KU LEUVEN

ARENBERG DOCTORAL SCHOOL
Faculty of Engineering Science

Robust Randomness Generation on Embedded Devices

Adriaan Peetermans

Supervisor:
Prof. dr. ir. Ingrid Verbauwhede

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

October 2024

Robust Randomness Generation on Embedded Devices

Adriaan PEETERMANS

Examination committee:

Prof. dr. ir. Jean Berlamont
Chair
Prof. dr. ir. Ingrid Verbauwhede
Supervisor
Prof. dr. ir. Wim Dehaene
Prof. dr. ir. Nele Mentens
(KU Leuven & Leiden University, The Netherlands)
Prof. dr. Werner Schindler
(BSI, Germany)
Dr. ir. Bohan Yang
(Tsinghua University, China)

Dissertation presented in partial fulfillment of the requirements for the degree of Doctor of Engineering Science (PhD): Electrical Engineering

October 2024

Preface

Although this is likely the first – and perhaps only – section many of you will read, it is the last one I wrote. In fact, according to my version manager, it has been 306 days since I started writing this thesis. It feels fitting that after all this technical work, I can now express my gratitude to the people who made this dissertation possible.

I would like to start by thanking my promotor, Prof. Ingrid Verbauwhede. Her trust and never-failing support have been instrumental in my growth as an engineer. I am especially thankful for the freedom she provided me throughout my PhD, allowing me to explore my own ideas and develop as an independent researcher. Additionally, I would like to genuinely thank my supervisors, Prof. Wim Dehaene and Prof. Nele Mentens, for their guidance throughout my PhD. I also wish to express my gratitude to my external jury members, Prof. Werner Schindler and Dr. Bohan Yang. Bohan, your mentorship during my master thesis and your introduction to the captivating field of TRNGs have profoundly shaped my research career. I still remember finding you late at night in the COSIC hallways during those early days. Finally, I thank Prof. Jean Berlamont for chairing the jury and overseeing this important milestone.

This work would not have been possible without the financial support of several funding agencies, particularly the Research Foundation - Flanders (FWO), which provided me with a PhD fellowship grant.

I could not have asked for a better environment to work during my PhD than COSIC, thanks to the many talented researchers working there. While I cannot thank everyone by name, I would like to give special recognition to the TRNG team: Vladimir and Miloš. Vladimir, your invaluable support during the initial phase of my PhD laid a strong foundation for my work, and Miloš, I greatly appreciated our numerous insightful discussions. I also want to thank my (former) office mates – Jan-Pieter, Sarani, Petr, Shuqin, and Zetao – with special thanks to Jan-Pieter, with whom I shared an office for nearly my entire

PhD. Additionally, I am grateful to Péla for her assistance in navigating all the practicalities of my PhD, as well as to the non-technical staff at COSIC – Saartje, Dana, Elsy, Wim, and Anouk – for ensuring our research group always ran smoothly. Furthermore, I would also like to thank my colleagues at the MICAS research group for the opportunity to participate in multiple tape-outs. Special thanks to Dragan, Carl, and Maxime for managing these tape-outs and ensuring the successful manufacture of my designs.

Buiten de meer *professionele* context ben ik ook veel dank verschuldigd aan verschillende vrienden. Mijn studiegenoten: Bernd & An Sofie, Evelien, Femke & Toon, Jonathan & Lotte, Luuk & Laurien, Simon & Nina, en Ward & Annabel. Velen van jullie ken ik al sinds de prille bachelorjaren. Daarnaast wil ik zeker enkele markante figuren extra in de verf zetten. Daan, de eerste persoon die ik ontmoette in Leuven, ondanks de afstand weten we op regelmatige basis contact te houden, wat ik enorm apprecieer. Robin, mijn langste onafgebroken vriendschap – waar moet ik beginnen? Bedankt om me wegwijs te maken in Leuven, specifiek in de Alma, waar we dan ook (te?) lang zijn blijven plakken, ondanks dat de Alma-hoogdagen ondertussen al ver achter ons lagen. Tot slot wil ik ook mijn sportieve vrienden bedanken, met in het bijzonder Jarne. Tijdens dit zes jaar durende doctoraat spendeerde ik 3334 uur al fietsend, lopend, wandelend of zwemmend, deels in jullie gezelschap. Ik ben er dan ook heilig van overtuigd dat deze sportieve afleiding een positieve invloed heeft gehad op dit proefschrift.

Als laatst, maar daarom zeker niet als minst heb ik nog veel dank over voor mijn familie. Mijn ouders, die ben ik eeuwig dankbaar; zonder jullie was er van dit proefschrift (en van ikzelf) helemaal geen sprake. Dankzij jullie goede zorgen in “hotel mama” – ik geef toe, mijn bijdragen in het huishouden waren en zijn beperkt – kon ik dit doctoraat steeds combineren met een mooie dosis sport. Helaas, mama, moet ik je teleurstellen: ik vrees dat ik de *sleutel* nog steeds niet heb gevonden. Ook mijn twee zussen Clara en Lucie en hun respectievelijke partners Giel en Jarne verdienen een woord van dank. Of jullie invloed op deze thesis uiteindelijk meer positief dan negatief was, laat ik maar in het midden.

Als allerlaatste is mijn vriendin Mira aan de beurt. Ik kan me dit doctoraat niet voorstellen zonder jou aan mijn zijde. Dat we samen in ditzelfde *doctoraatsschuitje* zaten maakte dat obstakels waar ik tijdens dit doctoraat op botste meteen zeer relateerbaar voor je waren. Ik kan je niet genoeg bedanken voor je constante steun, en ik kijk ernaar uit om samen alles wat ons te wachten staat, aan te gaan.

Adriaan Peetermans
Leuven, October 2024

Abstract

Cryptography plays a crucial role in securing communications and data in our modern society. High-quality randomness is essential for cryptographic systems, as it ensures security by being the only piece of information that a malicious party lacks to understand the entire system. Embedded devices, however, face significant challenges in generating this randomness due to their limited resources and minimal external input. As a result, these devices must rely heavily on on-chip generated randomness, making the design of TRNGs a critical area of research.

This dissertation makes several theoretical contributions to the field of TRNGs. Firstly, it proposes a more accurate entropy model that applies to a broader range of noise types. Our findings suggest that more noise types than previously recognized have potential utility in TRNGs, although further research is necessary to determine their precise entropy contributions. Additionally, this work offers a refined time-based characterization of various noise types on an ASIC platform, revealing that under practical operating conditions, a different noise type dominates compared to earlier assumptions.

On the practical side, this dissertation enhances TRNG design through increased control over key parameters, allowing designers to achieve optimal operating conditions with significantly less effort. The robustness of TRNGs against varying operating conditions has also been improved, making the output usable across a broader range of environments. Moreover, the research provides deeper insights into the trade-offs involved in performance optimization, facilitating a more streamlined design process compared to previous iterative approaches.

Beknopte Samenvatting

Cryptografie speelt een cruciale rol in het beveiligen van communicatie en data in onze moderne samenleving. De kwaliteit van willekeur is essentieel voor cryptografische systemen, omdat dit de beveiliging waarborgt door het enige stukje informatie te zijn dat een kwaadwillende partij ontbreekt om het systeem volledig te begrijpen. Ingebedde apparaten staan echter voor serieuze uitdagingen bij het genereren van deze willekeur, vanwege hun beperkte middelen en minimale invoer. Als gevolg hiervan zijn deze apparaten sterk afhankelijk van willekeur die op de chip wordt gegenereerd, wat het ontwerpen van werkelijk willekeurige getallengeneratoren (True Random Number Generators, TRNG's) tot een kritisch onderzoeksgebied maakt.

Dit proefschrift draagt op verschillende theoretische wijzen bij aan het onderzoeksgebied van TRNG's. Ten eerste introduceert het een nauwkeuriger entropiemodel dat van toepassing is op een breder scala aan ruistypes. Onze bevindingen suggereren dat meer ruistypes dan voorheen gedacht potentieel bruikbaar zijn in TRNG's, hoewel verder onderzoek nodig is om de exacte bijdragen van deze ruistypes te bepalen. Daarnaast biedt dit werk een verfijnde, op tijd gebaseerde karakterisering van verschillende ruistypes op een ASIC-platform. Hieruit blijkt dat onder praktische werkingsomstandigheden een ander ruistype domineert dan voorheen werd aangenomen.

Aan de praktische kant bevordert dit proefschrift het ontwerp van TRNG's door de controle over cruciale parameters te versterken, waardoor ontwerpers optimale prestaties kunnen bereiken met aanzienlijk minder inspanning. Daarnaast is de robuustheid van TRNG's tegen veranderende operationele omstandigheden verbeterd, wat de bruikbaarheid van de gegenereerde willekeur vergroot in een breder scala aan omgevingen. Bovendien biedt dit onderzoek diepgaand inzicht in de afwegingen die betrokken zijn bij het optimaliseren van de performantie, wat leidt tot een meer gestroomlijnd ontwerpproces in vergelijking met eerdere iteratieve benaderingen.

List of Abbreviations

ACF Auto-Correlation Function. 31–33, 35, 55, 60–64, 66, 84, 85, 177–179, 187

AIS Application notes and Interpretation of the Scheme. 13, 15, 17, 116, 120, 132

ALM Adaptive Logic Module. 134

ASIC Application-Specific Integrated Circuit. iii, v, 4–9, 22, 23, 27, 29, 44, 46, 47, 57, 58, 91–93, 106, 107, 113–115, 137, 138, 165–167

BSI Bundesamt für Sicherheit in der Informationstechnik. 13–15, 27

CCF Cross-Correlation Function. 177

CDF Cumulative Distribution Function. 145, 149, 151, 153, 174, 187

CMOS Complementary Metal-Oxide-Semiconductor. 29, 37, 44, 58, 94, 95, 106, 107, 110, 114, 138, 160, 163

COSO COherent Sampling ring Oscillator. 22, 23, 92, 106, 116, 117, 119, 123–126, 132–134

CSI Current-Starved Inverter. 107–111, 160

DAC Digital-to-Analog Converter. 107

DAS Digitized-Analog-Signal. 13, 15

DC Delay Chain. 20, 21, 28, 29, 37, 38, 43, 44, 47, 48, 50–52, 57, 58, 92, 134, 138–141, 145, 146, 148, 154–163

DFF Data Flip-Flop. 38, 117, 121, 146

DNL Differential Non-Linearity. 46

DSP Digital Signal Processor. 93, 116

E2L Edge To Level. 140, 145

ERO Elementary Ring Oscillator. 53, 61, 64, 66, 67, 75, 77, 78, 80, 83, 84, 118

ES Entropy Source. 4–9, 11–14, 16–24, 27–29, 52, 53, 58–61, 64, 66, 67, 75, 77, 78, 80, 83, 84, 91–93, 98, 106, 107, 116–120, 123–129, 132–135, 137–139, 154, 157–160, 162, 163, 165–167

FF Flip-Flop. 22, 38, 64, 97, 105, 117, 133–135, 163

FIPS Federal Information Processing Standard. 13, 116

FM Frequency Modulated. 5, 28, 31, 32, 36, 52, 55–57, 59–64, 75–78, 80, 81, 83, 84, 133, 158, 161, 165, 166, 171

FPGA Field-Programmable Gate Array. 6–9, 22, 23, 27, 44, 46, 56–58, 91–98, 100, 101, 103, 105, 106, 114–117, 120, 123–134, 137, 138, 166

FT Fourier Transform. 85, 171, 179

GP Global Placement. 98, 99, 101, 103, 106, 123–125, 127, 128, 132, 133

HDL Hardware Description Language. 22, 92, 93, 101, 103, 129, 166

HTP Min-entropy-Throughput Product. 119, 120, 125, 132

IC Integrated Circuit. 4, 92

iff if and only if. 187

IID Independent and Identically Distributed. 143, 153, 161

IQR InterQuartile Range. 98

ISO International Organization for Standardization. 14, 16, 27

LFSR Linear-Feedback Shift Register. 97, 124

LP Local Placement. 98, 99, 101, 103, 106, 124, 125, 127, 128, 132, 133

LSB Least Significant Bit. 118, 122, 124, 153

LUT LookUp table. 22, 23, 93–99, 101, 103, 105, 114, 117, 132–135

MOS Metal-Oxide-Semiconductor. 53, 107

MOSFET Metal-Oxide-Semiconductor Field-Effect Transistor. 53

MP Manual Placement. 133, 134

MR Manual Routing. 133, 134

MUX Multiplexer. 94, 95, 97

NIST National Institute of Standards and Technology. 13–15, 27, 116, 132, 161, 162

PC Personal Computer. 44, 97

PCB Printed Circuit Board. 44

PDF Probability Density Function. 68, 69, 71–74, 84, 145, 146, 151, 173, 174, 180–183, 187, 188

PLL Phase-Locked Loop. 22, 28, 92, 93, 116, 117, 134

PM Phase Modulated. 31, 32, 36, 56, 62, 63

PMF Probability Mass Function. 66–68, 71, 72, 156, 173, 174, 180–186

PRNG PseudoRandom Number Generator. 3, 4, 13, 16

PSD Power Spectral Density. 31–36, 53, 55, 62, 63, 85, 179

PUF Physical Unclonable Function. 93, 199

PVT Process, Voltage and Temperature. 21, 22, 167

RNG Random Number Generator. 3

RO Ring Oscillator. 4–9, 14, 20–23, 27, 28, 33, 36–38, 42–44, 47–49, 51–54, 56, 60, 64, 92–101, 103, 105–111, 113, 114, 116–135, 138–141, 143–146, 153, 154, 160–162, 166, 167

SoC System-on-Chip. 44

SRAM Static Random-Access Memory. 96, 164

STR Self-Timed Ring. 92, 134, 137, 166

TDC Time-to-Digital Converter. 20, 29, 46, 51, 52, 138–141, 146–149, 151, 152, 154–163

TERO Transition Effect Ring Oscillator. 22, 92, 134, 137

TFF Toggle Flip-Flop. 44, 141

TRNG True Random Number Generator. i, iii, v, 3, 4, 8, 9, 11–18, 23, 84, 92, 98, 115, 116, 165–167, 199

WSS Wide-Sense Stationary. 28, 31–33, 35, 36, 55–57, 63, 178, 179, 187

List of Symbols

Probability Mathematics

$\mathbf{\Sigma}$	Covariance matrix
$\mathbf{Cor}$	Correlation
$\mathbf{Cov}$	Covariance
$\mathbf{E}$	Expected value
$\mathbf{H_m}$	Min-entropy
$\mathbf{H_{worst}}$	Worst-case Shannon entropy
$\mathbf{H}$	Shannon entropy
$\mathbf{I}$	Information content
$\mathbf{P}$	Default probability measure
$\mathbf{Var}$	Variance
$\mathcal{B}$	Borel σ-algebra
$\mathcal{F}$	Fourier transform or event space
$\mathcal{IG}$	Inverse-Gaussian distribution
$\mathcal{N}$	Normal distribution
$\mathcal{U}$	Uniform distribution
μ	Distribution mean
Ω	Sample space

ω	Random outcome
Φ_s	Standard normal CDF
ϕ_s	Standard normal PDF
σ	Distribution standard deviation
F_X	Cumulative distribution function
f_X	Probability density/mass function
W	Wiener process

Electrical Physics

$\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}$	Random excess phase vector
$\vec{\boldsymbol{B}}$	Random bit vector
Δ	Random period length difference
Φ	Oscillator phase process
ϕ_0	Initial phase
Φ_e	Oscillator excess phase process
Φ_g	Global oscillator phase perturbation
Φ_Δ	TDC phase difference
τ	Time shift
φ	Phase realization
A	Oscillator relative phase acceleration process
B	Random bit
C	Random coherent sampling count
d_{LSB}	DC stage delay
e	Observable waveform function
f	Fourier frequency
f_h	Higher frequency limit

f_l	Lower frequency limit
f_n	Nominal oscillation frequency
h_α	Noise magnitude
p_w	Waveform period length
Q	Quantization error
R	Random counter output
R_X	Auto-correlation function
S_X	Power spectral density
s_{noise}	Linear noise strength
t	Continuous time
T_x^n	DC propagation delay
T_Δ^n	DC propagation delay difference
$t_{\max}$	Maximal accumulation time
T_π	TDC stopping time
t_{cor}	Noise corner accumulation time
Y	Oscillator relative frequency deviation process

Other Symbols

Ci	Cosine integral function	
δ	Dirac delta distribution	
$\emptyset$	Empty set	
γ	Euler-Mascheroni constant	0.577 215 665
$\mathbb{N}$	Set of natural numbers	
$\mathbb{R}$	Set of real numbers	
$\mathbb{Z}$	Set of integers	
π	Number pi	3.141 592 654

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Let me begin this thesis with a plea to the significance of *randomness* in our daily lives. When considering the presence of randomness in daily routines, the first examples that come to mind are often obvious ones such as flipping a coin, rolling a die, or spinning a roulette wheel. Others might associate randomness with luck, fortune, and the random chances that influence the course of one's life. Upon deeper reflection, concepts like *probability*, uncertainty, variability, surprise, and chaos often emerge. These aspects relate to various experiences individuals encounter, such as the emergence of specific weather patterns, traffic congestion and travel times, the outcomes of sports matches, unexpected breakdowns, last-minute cancellations, the quality of manufactured goods, and numerous other instances.

However, as I will elucidate in this introduction, randomness also plays a crucial role in many daily aspects and activities beyond those mentioned above. Consider the following actions, which almost everyone performs, often on a daily basis: sending and receiving messages via WhatsApp, Signal, or other chatting applications; making payments using a bank card; using a smart card to unlock office doors; browsing the internet; digitally signing documents with a (Belgian) identity card; connecting to a Wi-Fi hotspot; and unlocking cars with a key fob. Some actions, while less common, are highly relevant to more technologically inclined individuals: encrypting and signing emails; using SSH to connect to remote devices; and mining or transferring bitcoins.

The actions listed above share common requirements for *confidentiality* (e.g., ensuring that only intended recipients can read your chat messages), *integrity* (e.g., preventing alterations to the amount paid using your bank card), *authentication* (e.g., ensuring that your office door opens only when

your card is presented and not someone else's), or *non-repudiation* (e.g., preventing someone from denying that they signed a contract). *Cryptography*, a branch of mathematics, computer science, and electrical engineering, enables these everyday actions. For instance, encryption algorithms provide data confidentiality and integrity, while signature algorithms ensure authentication and non-repudiation.

These cryptographic algorithms and protocols rely heavily on basic building blocks (*primitives*), such as block ciphers, hash functions, and key exchange mechanisms. The design and operation of these cryptographic primitives are publicly available, meaning they do not depend on obscurity or hiding their internal workings to function effectively. Instead, cryptographic algorithms and protocols depend on the secrecy of small pieces of data, akin to passwords, often referred to as secret *keys*. This practice of making all details about cryptographic algorithms public is known as Kerckhoffs' principle, named after the cryptographer Auguste Kerckhoffs, who formulated it already in the 19th century [37].

Given the current landscape where anyone can perform any cryptographic operation, including the one used to encrypt your sensitive data, it is crucial to rely on the fact that only a select number of intended parties have knowledge of the secret key used for the encryption. Similar to passwords, it is essential to minimize the *likelihood* of anyone successfully guessing your secret key. Passwords that follow common keyboard patterns (such as `qwerty` or `12345678`) or include recognizable words or phrases (such as `Pa55w0rd` or `l3tmE1n`) are significantly more vulnerable to guessing compared to passwords without such patterns, such as `9pXJkL!zT$7fMn#` or `B8v^qR&5sW!yL2P`.

The same principle applies to secret keys composed of binary digits. A key with repetitive patterns such as `001001001001` is much easier to *predict* than a more randomized key such as `011101010100`. Additionally, it is crucial to minimize the *chance* of someone choosing a similar secret key to yours.

It should now be evident that the properties required for secret keys, such as unpredictability and *uniqueness*, are largely fulfilled when we select randomly generated data to serve as our keying material. A perfectly random bit string has the smallest probability of being guessed on the first try, specifically 2^{-n} for a length n. Additionally, it has the lowest probability of a collision, meaning two entities *independently* choosing the same key [23]. According to the birthday paradox, this probability is smaller than $\frac{k^2}{2^n}$ when k entities choose a key of length n and $k \leq 2^{n-1} + 1$.

One cannot quantify the randomness of a piece of data by itself. For instance, what makes the bit string `00000000` more or less random than any other bit

string, such as `10110111` or `10100011`? After all, any eight-bit pattern has an equal probability of 2^{-8} to be generated by an ideal Random Number Generator (RNG). *Randomness* is a property of the process that generates the random data, not of the data itself.

The concept of perfect randomness, along with related notions such as *entropy* and *information*, are introduced in the following section. For a formal definition of these concepts, interested readers are directed to appendix A.

1.1 Randomness and Integrated Circuits

In this thesis, perfect randomness refers to the property of devices capable of generating any n-bit pattern with a probability of 2^{-n}, independently of any past or future generated patterns. The term full entropy is often used to denote that the output of such a device carries a maximal amount of information, meaning it is impossible to compress the output to a smaller size (in terms of the number of symbols used to uniquely represent the data) without losing information. Perfect randomness, however, does not exist in reality, and we must be satisfied with far less ideal solutions. Rest assured, this is precisely where the engineer comes into play and why the research field addressed in this thesis exists.

Generating sufficiently robust randomness proves to be a challenging and often underestimated task in practice, as evidenced by instances where cryptographic guarantees failed due to deficient RNGs [14, 30]. This challenge is further exacerbated by the numerous design constraints found in real-world applications. Some systems leverage diverse randomness sources, including human input (e.g., mouse cursor movement), external input (e.g., network packet arrival times or hard drive access times), and internal timings (e.g., timing of interrupts), which are aggregated into a high-quality entropy pool, exemplified by `/dev/random` on Unix-based systems [25].

In contrast, certain devices do not possess external sources of randomness and thus rely solely on dedicated internal hardware for generating randomness. This is particularly pertinent for small *embedded* devices like smart cards, biomedical implants, and sensor nodes. The research findings presented in this dissertation primarily address solutions tailored for such embedded devices.

We distinguish between two flavors of randomness: *true* (also known as fresh or pure) and *pseudo* (also known as deterministic), generated respectively by devices known as True Random Number Generators (TRNGs) and PseudoRandom Number Generators (PRNGs). A TRNG produces new, unpredictable

information and offers *information-theoretic* security, ensuring that its output remains random even against adversaries with unlimited time and computational resources. In contrast, a PRNG expands a given amount of randomness in a deterministic manner and relies on the cryptographic strength of its underlying components for *computational* security. The research presented in this thesis exclusively addresses TRNGs.

TRNGs implemented on embedded devices frequently depend on extracting *electronic circuit noise*. Decades of research have been dedicated to minimizing the impact of electronic noise on Integrated Circuits (ICs). The entire IC design and manufacturing process has been developed to ensure reliability and repeatability. In some way, the TRNG designer must deviate from these paradigms and employ circuit techniques that are often considered peculiar by others.

The principal circuit block analyzed in this thesis is the Ring Oscillator (RO). ROs consist of a ring of simple circuit elements capable of producing a rail-to-rail oscillating voltage signal. The natural variations in the oscillation timing, referred to as *timing jitter* throughout this dissertation, will serve as the main source of randomness in the circuits discussed in the following chapters.

1.2 Research Questions and Objectives

This dissertation addresses a series of research questions, each tied to a publication where I served as the primary author, and discussed in a subsequent chapter. For each question, one or more objectives have been formulated. These objectives involve research actions, such as specific measurements, mathematical analyses, or the design of specific circuits or chips, all aimed at answering the posed research question. To support the results presented in this dissertation, a total of three distinct chips were manufactured.

Research Question 1: Oscillator Phase Noise Magnitude

Various *noise types*, often referred to as *noise colors*, can influence the phase of a free-running RO, implemented on an Application-Specific Integrated Circuit (ASIC) platform. When estimating the entropy for Entropy Sources (ESs), it is crucial to accurately measure the magnitude of each of the noise types involved. Furthermore, at a given *accumulation time* length, one noise type often *dominates* over the others, exhibiting a magnitude several orders greater than the other types.

Question. *What are the magnitudes of noise types relevant to oscillator-based ESs, and which noise type is the most* significant *on an ASIC platform?*

Objective 1.1. Develop an on-chip measurement circuit capable of quantifying the relevant noise types.

Objective 1.2. Construct a mathematical analysis of the measurement circuit to differentiate between the noise types from the measurement data.

This research question is addressed in my *IEEE TCAS I 2024* paper, which serves as the foundation for chapter 3 in this thesis.

> **Characterization of Oscillator Phase Noise Arising From Multiple Sources for ASIC True Random Number Generation**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IEEE Transactions on Circuits and Systems I (TCAS I)*, 2024

Research Question 2: Flicker FM Noise Contribution

Almost all attempts at modeling ESs over the last decade have been based on the assumption that white Frequency Modulated (FM) noise dominates in free-running ROs. White FM noise distinguishes itself from other noise types due to its inherent time-independence. Consecutive samples drawn from a white FM noise source can be considered independent, significantly reducing the complexity of proposed *entropy models*. However, as discussed in chapter 3 and recently verified on other hardware platforms, flicker FM noise can become dominant, particularly at longer accumulation time lengths.

Question. *What is the contribution of flicker FM noise to the overall* entropy rate *produced by oscillator-based ESs, in addition to the well-known contribution of white FM noise?*

Objective 2.1. Develop an analytical model that describes the oscillator phase influenced by multiple noise types.

Objective 2.2. Create a simulation model of an oscillator-based ES that can compare the impact of different noise types on the total produced entropy rate.

This research question is addressed in my *IACR TCHES 2024* paper, which serves as the foundation for chapter 4 in this thesis.

> **TRNG Entropy Model in the Presence of Flicker FM Noise**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, 2024

Research Question 3: Oscillator Frequency Control

Modeling the ES enables a precise determination of the required *design parameter* values. However, achieving these precise values in practice often demands significant *effort* from the designer. Examples of design parameters include the RO frequency and the *resolution* at which timing variations are sampled. Different techniques for realizing these design parameters are necessary, depending on whether the target hardware platform is a Field-Programmable Gate Array (FPGA) or an ASIC.

Question. *How can precise control over ES design parameters, particularly the RO frequency, be achieved on both FPGAs and ASICs?*

Objective 3.1. Develop a configurable RO topology controlled by a digital signal for both FPGA and ASIC platforms.

Objective 3.2. Characterize the configurable RO topology to determine the achievable period length *range* and resolution under a diverse set of operating conditions.

This research question is addressed in my *ACM TRETS 2021* paper, which serves as the foundation for chapter 5 in this thesis.

> **Design and Analysis of Configurable Ring Oscillators for True Random Number Generation Based on Coherent Sampling**
> Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
> *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2021

Research Question 4: ES Design Effort on FPGAs

Aside from precise oscillator tuning mechanisms, a complete ES design requires control logic to manage the RO configuration input. This control logic is essential for selecting an optimal RO configuration at device startup, thereby reducing the design effort, as the designer no longer needs to manually find a an RO that produces a suitable frequency. Additionally, the control logic must

continuously monitor the performance of the ES and update the configuration vector whenever the ES ceases to operate optimally.

Question. *Can the required design effort be significantly reduced when implementing an oscillator-based ES on FPGAs, by using the proposed RO tuning mechanisms?*

Objective 4.1. Formulate a controlling algorithm to select the optimal RO configuration and integrate this algorithm into a complete ES on an FPGA platform.

Objective 4.2. Experimentally verify the proper functioning of the ES, together with the controlling logic, under various operating conditions.

This research question is addressed in my *FPL 2019* paper, which serves as the foundation for chapter 6 in this thesis.

> **A Highly-Portable True Random Number Generator Based on Coherent Sampling**
> Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
> *International Conference on Field Programmable Logic and Applications (FPL)*, 2019

Research Question 5: Optimal Design Parameter Selection

Dynamic control over the design parameters is essential, but determining a suitable value range for these design parameters is equally important. This optimal range depends on measured *platform parameters* and constraints set by the targeted application. Leveraging insights from the stochastic model of the ES, an optimization procedure should be devised to select an optimal range for the design parameters.

Question. *Can the stochastic model be used to develop a strategy for selecting optimal design parameters?*

Objective 5.1. Develop a custom oscillator-based ES on an ASIC platform, along with an associated stochastic model.

Objective 5.2. Formulate an analytical design parameter optimization procedure based on the developed stochastic model.

Objective 5.3. Apply the derived optimal configuration to the manufactured ES and verify optimal functionality.

This research question is addressed in my *IACR TCHES 2022* paper, which serves as the foundation for chapter 7 in this thesis.

> **An Energy and Area Efficient, All Digital Entropy Source Compatible with Modern Standards Based on Jitter Pipelining**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, 2022

1.3 Structure of this Dissertation

The *modern* TRNG design flow is introduced and contrasted with the currently considered *obsolete* TRNG design procedure in chapter 2. Following chapter 2, this dissertation is divided into two main parts, both centered around the stochastic model, an indispensable element in this modern TRNG design flow.

Part I addresses the inputs required by the stochastic model. This includes verifying the assumptions made by the stochastic model, such as determining which type of noise most significantly affects the generated entropy and verifying the independence of individual circuit components. This verification is discussed in chapter 4 by analyzing the contributions of common noise types in an oscillator-based ES design. Another crucial input for the stochastic model is the experimental estimation of key platform parameters. The measurement of one such parameter, *phase noise* magnitude, is thoroughly examined in chapter 3.

The capabilities enabled by employing a stochastic model in TRNG design are discussed in part II. The stochastic model facilitates the formulation of a procedure for selecting optimal ES design parameters, including parameters set during the design phase, such as RO frequency and noise accumulation time length. Chapter 5 explores various RO topologies that enable the realization of selected design parameters, specifically RO frequency, on both FPGA and ASIC platforms. These RO topologies are implemented in complete ES architectures in chapters 6 and 7 for FPGA and ASIC hardware platforms, respectively. Additionally, chapter 6 introduces a method for dynamically updating the design parameters during device operation. Chapter 7 also provides a concrete example of the design parameter optimization procedure based on a stochastic model describing the ES.

Lastly, this dissertation is concluded in chapter 8, where the key contributions presented throughout the work are summarized, and their respective implications are discussed. Additionally, further research directions on this captivating topic are also provided.

Note. As could be observed from the chronology of the publications addressing the research questions introduced in section 1.2, the chapters in this dissertation are not arranged according to the timeline of the research. Instead, an order that closely follows the TRNG design flow was preferred. The reader should keep in mind that the results presented in the final chapters, particularly in chapters 6 and 7, date back to before the author gained the additional insights presented in part I of this dissertation.

1.4 Other Contributions

I have contributed to the following publications, which are not included in this dissertation. For a comprehensive list of publications to which I have contributed, please refer to the List of Publications on page 201.

Attacking Hardware Random Number Generators in a Multi-Tenant Scenario Implementing TRNGs in a multi-tenant scenario, where multiple users share a common FPGA die, introduces new vulnerabilities and potential attack vectors. In this study, we investigated the impact of three distinct attacks on two ES implementations on an FPGA. Notably, voltage manipulation attacks, by generating circuit activity-induced voltage variations, significantly disrupted entropy generation.

This publication is a result of a master thesis I co-supervised.

> **Attacking Hardware Random Number Generators in a Multi-Tenant Scenario**
> Yrjo Koyen, Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
> *Workshop on Fault Detection and Tolerance in Cryptography (FDTC)*, 2020

Contribution: *master thesis student supervision.*

SCALLER: Standard Cell Assembled and Local Layout Effect-based Ring Oscillators Particularly on an ASIC platform, numerous alternative techniques for adjusting the RO frequency are available beyond those outlined in chapter 5. This study introduces a standard-cell compatible RO configuration approach enabling highly accurate frequency tuning. The method leverages layout-induced effects on transistor drive current strength by manipulating the proximity of the transistor channel to the edge of the doped well. Experimental findings from fabricated devices validate the effectiveness of this tuning technique.

This publication is a result of a collaboration with the Tallinn University of Technology (TalTech), I was involved at the design phase and helped to prepare the measurement set-up.

SCALLER: Standard Cell Assembled and Local Layout Effect-Based Ring Oscillators
Muayad J. Aljafar, Zain Ul Abideen, Adriaan Peetermans, Benedikt Gierlichs, and Samuel Pagliarini
IEEE Embedded Systems Letters, 2024

Contribution: *design review and measurement set-up assistance.*

Chapter 2

Modern TRNG Design

The core perspective of this thesis revolves around the modern ES design and verification procedure, showcased in fig. 2.1. Central in this process is the stochastic model. Part I in this thesis addresses prerequisites of the model, these encompass the inputs necessary for the stochastic model, which include the assumptions made, and measurement of defined platform parameters. Part II in this thesis deals with capabilities of the model, covering the outputs generated by the stochastic model, which include design parameter optimization methods and discusses how to practically realize the required design parameter values across different hardware platforms.

2.1 TRNG Architecture and Terminology

The generic architecture of a TRNG is provided in fig. 2.2. The core building block of a TRNG is the ES, where *entropy* is generated by an analog phenomenon. This thesis focuses solely on *electrical noise sources*, such as *thermal* or *flicker* noise, for generating entropy. Whether entropy is genuinely being generated or is merely an artifact of the limitations in noise models accurately capturing circuit component behavior is considered a philosophical question and is, therefore, not further discussed in this thesis.

After digitization, we obtain *raw random numbers*, specifically raw random bits in this thesis. These raw random bits are often far from perfect, as they might be biased or contain dependencies. To improve their quality, post-processing is employed, which lowers throughput in exchange for higher *entropy density*. A

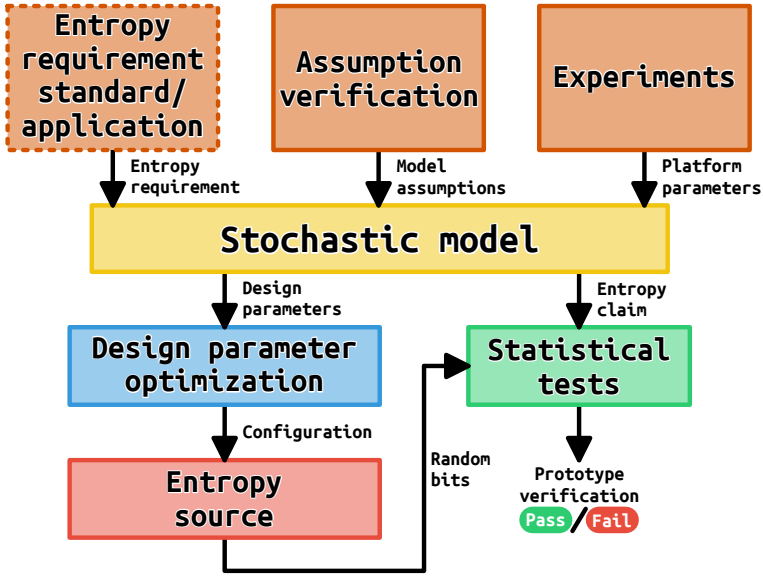

Figure 2.1: Modern ES verification approach.

third block provides testing functionality and alerts the application consuming the *randomness* if the output no longer contains the specified entropy density.

Table 2.1 provides a comparison between the relevant terms used in this thesis and those used by international standards [39, 83]. These standards are further introduced in section 2.2.2. The work presented in this dissertation will, to a large extent, focus on the ES rather than post-processing or health testing.

Note. In this thesis, the term *noise source* is defined according to its circuit-related context. Specifically, noise source refers to the hypothetical noise voltage or noise current sources added in series or parallel to an ideal circuit component (such as resistors or transistors) to model the behavior of real-world components. This definition contrasts with the standards [39, 83], where noise source refers to the sub-circuit of a TRNG identified as an *Entropy Source (ES)* in this text.

2.2 Obsolete versus Modern ES Design

Figure 2.3 illustrates the *obsolete* ES design and verification flow, which can be compared to the *modern* flow shown in fig. 2.1. The modern approach distinguishes itself from the obsolete one by incorporating a *stochastic model*

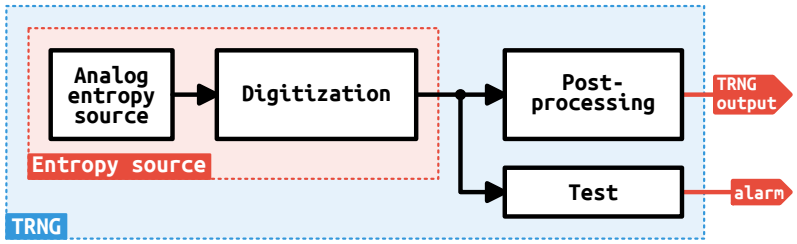

Figure 2.2: Generic TRNG architecture.

Table 2.1: Comparison in terminology.

This thesis	NIST SP 800-90B [83]	BSI AIS 20/31 [39]
Entropy Source (ES)	(Physical) Digital noise source	Physical noise source
TRNG	Nondeterministic random bit generator	Physical random number generator
PRNG	Deterministic random bit generator	Deterministic random number generator
Raw random numbers	Raw data	DAS-random numbers
TRNG output	Entropy source output	Internal random numbers
Entropy density	Entropy rate	Entropy per bit
Entropy rate	Entropy rate × throughput	Entropy rate

(also referred to as a mathematical model) for the ES. Before its inclusion became mandatory due to international standards, this stochastic model was generally absent in most ES designs. The stochastic model forms the cornerstone of the modern approach, offering a *probabilistic* description of how the entropy extraction mechanism operates, converting entropy from the underlying noise sources into the generated random data.

2.2.1 Obsolete Approach

Verification of the correct working of an ES often relied on execution of statistical test suits such as Diehard [53], National Institute of Standards and Technology (NIST) SP 800-22 [74] or the tests specified in Federal Information Processing Standard (FIPS) 140-2 [58]. The general workflow is illustrated in fig. 2.3. The validator generates a predetermined amount of test data from the ES, which is

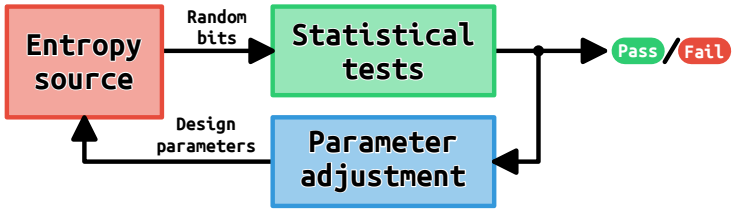

Figure 2.3: Obsolete ES verification approach.

then subjected to various statistical tests. These tests typically search for specific patterns within the data, the absence of these patterns is then interpreted as evidence of the ES functioning as intended. If substantial evidence is collected suggesting that the data cannot be considered random, i.e., if some tests fail, internal *design parameters* of the ES are tuned and the process was repeated with newly generated data. In the past, passing statistical tests was often perceived as a *proof* of the correct functioning of the ES under examination. However, it is now generally acknowledged that these tests are merely capable of proving the opposite; they indicate that an ES is not functioning as intended.

Several studies have revealed that relying solely on the results of statistical tests can lead to a dangerous overestimation of the actual entropy generated by the ES. A remarkable example is the design utilized by Motorola [82], which employs two free-running ROs. As elucidated by [19], an attack that significantly increases the success rate of *predicting* the generator output correctly is feasible, despite the generator passing multiple statistical test suites. Another study [10] demonstrated that statistical tests falsely suggested the presence of fresh entropy when applied to data produced by a deterministic model of an RO-based design, wherein all *random jitter* in the ROs was reduced to zero. Despite ample evidence highlighting the flaws in the approach outlined in this section, recent research [40, 60, 81] continuous to be published using this obsolete approach.

2.2.2 International Standards

TRNG standardization is a rapidly evolving domain. The research outlined in this thesis started in 2018 and over the past six years, three prominent standardization bodies: NIST, Bundesamt für Sicherheit in der Informationstechnik (BSI) and International Organization for Standardization (ISO) have adjusted the criteria for evaluating TRNGs. Both BSI and ISO now mandate the inclusion of a stochastic model as a fundamental criterion for TRNG evaluation. While NIST allows for the formulation of the model to be optional, it explicitly

acknowledges the stochastic model as a valid supporting argument for any entropy claim put forth. Efforts are ongoing to align the various standards, streamlining the process for a single design to achieve certification from multiple standardization bodies.

NIST SP 800-90B Section 3.2.2: *Requirements on the Noise Source* of the NIST SP 800-90B recommendation for entropy sources [83], published in 2018, lists the requirements imposed on a noise source. The third requirement reads:

> "Documentation **shall** provide an explicit statement of the expected entropy provided by the noise source outputs and provide a technical argument for why the noise source can support that entropy rate. To support this, documentation **may** include a stochastic model of the noise source outputs, and an entropy estimation based on this stochastic model **may** be included."

The submitter should make a minimal entropy estimate, which can be derived from a stochastic model.

Note. The formulation of a stochastic model is not obligatory.

BSI AIS 20/31 As early as 2001, the German BSI introduced the standardization of TRNGs, with the publication of the Application notes and Interpretation of the Scheme (AIS) standard: AIS 31 [38] and a conference paper: [77]. In this initial release of the AIS 31 standard, two TRNG classes were distinguished, with the stricter class (P2) mandating a mathematical model as an alternative requirement for certification. Only for specific TRNG designs under examination, where performing statistical testing on the Digitized-Analog-Signal (DAS)-random numbers proves infeasible, resorting to the mathematical model serves as an alternative requirement to identify dependencies in the generated data.

In a subsequent version of the standard [39], which has been in effect since 2011, the classification of TRNGs expanded to three hierarchical classes: PTG.1, PTG.2 and PTG.3. A TRNG meeting the criteria of a higher class, automatically fulfills the requirements of the lower classes. Both PTG.2 and PTG.3 classes now mandate the formulation of a stochastic model.

An updated draft of the AIS 20/31 standard became accessible in 2022 [69]. This draft eliminates the PTG.1 class, implying that all TRNG designs now necessitate the description of a stochastic model to adhere to the AIS 20/31 standard.

ISO 20543 The ISO standard, released in 2019 [34], addresses the evaluation of both PRNGs and TRNGs. The essential aspect of the evaluation process for TRNGs, is the utilization of a stochastic model to substantiate any entropy claims made.

2.2.3 Stochastic Model

In this thesis, the definition for a stochastic model was adopted from paragraph 622 in [69]:

> "A stochastic model provides a partial mathematical description (of the relevant properties) of a physical noise source using random variables. The stochastic model shall allow the verification of a lower entropy bound for the internal random numbers."

A stochastic model constructs a set of distributions encompassing the true distribution of the generated random numbers. The actual distribution may be influenced by parameters representing *process variations*, *operating conditions*, and *aging effects*. Hence, the stochastic model should accurately predict the entropy's lower bound for the ES across all manufactured devices, under all permissible operating conditions, and throughout the device's lifespan. To prevent undue complexity in the analysis, only relevant aspects should be incorporated. Consequently, the stochastic model inevitably presents a simplified version of reality.

In contrast to assessing the design of a PRNG, which can be carried out independently of its implementation, evaluating a TRNG and its associated ES using a stochastic model is inherently tied to the specific implementation and hardware platform utilized. The *security* of a PRNG is solely *computational*, implying that a pure PRNG would offer no defense against an adversary with unlimited computational resources and/or time. On the other hand, a TRNG provides security in an *information-theoretical* sense, ensuring that the freshly generated entropy remains unpredictable to such an adversary. Frequently, the terms *pseudo randomness* and *true randomness* are employed to denote these two concepts.

2.3 Part I: Model Prerequisites

This first part of the dissertation concentrates on the inputs needed by the stochastic model. As depicted in fig. 2.1, three distinct prerequisites are

fundamental when validating an ES using a stochastic model: determining the necessary entropy density, validating all *assumptions* made by the model, and experimentally confirming the values of physical quantities utilized in the model.

2.3.1 Entropy Requirement

Demanding a specific minimal entropy density in the random data produced by the TRNG sets a limit on the success rate of correctly predicting the output data. This defined lower bound on the entropy density may stem from either a TRNG standard that the designer aims to adhere to, the intended application consuming the random data, or both.

Standard

The AIS 20/31 standard [39] mandates a minimum Shannon entropy per generated random bit for TRNGs in class PTG.2. Requirement PTG.2.7 states:

> "The average Shannon entropy per internal random bit exceeds 0.997."

Implying that after post-processing, the random data supplied to the application must offer a Shannon entropy density of no less than 0.997 bit per output bit. The revised AIS 20/31 draft [69] has tightened the entropy criterion. Requirement PTG.2.3 now specifies that the average Shannon entropy for the generated data after post-processing should surpass 0.9998 bit per output bit.

Application

Depending on the intended application, the entropy requirement can be further increased. Various applications may necessitate diverse levels of security. A TRNG design providing data of appropriate quality for the application should be selected. Certainly, the TRNG should meet the minimum security standards demanded by the application. Conversely, developing a TRNG that delivers more entropy than strictly necessary for the application could result in a waste of resources.

2.3.2 Model Assumptions

A stochastic model inevitably relies on assumptions, which are evident statements regarded as true. For some assumptions, the model can rely on related research that offers evidence supporting the assumption's validity. Other assumptions are more foundational and are simply assumed true until evidence suggesting the opposite becomes available. When an assumption is proven to be invalid, any conclusions drawn from the model become uncertain. Hence, it is crucial for the model to explicitly state its assumptions, making it easier to challenge them during model validation.

This thesis focuses solely on oscillator-based ES designs. The common assumptions associated with these designs often revolve around the behavior of the noise sources impacting the oscillator. For instance, which *types of noise sources* are presumed to have the most significant effect on the characteristics of the generated random data, or how these noise sources are assumed to behave in the time and frequency domains. Chapter 4 handles these prevailing assumptions and offers a rationale that stochastic models can reference.

The subsequent subsections enumerate additional common assumptions employed in modeling oscillator-based TRNG designs.

Noise Sources Power Law

The analysis outlined in chapters 3 and 4 assumes, along with other noise models [28, 32, 59], that the frequency spectrum of the noise types involved follows a *power law* relation with the absolute frequency: f^{α}, for different $\alpha \in \mathbb{Z}$. Frequently, the term *colored noise* is employed, with various colors assigned to different noise types. For instance, white is associated with thermal noise ($\alpha = 0$), while pink is linked to flicker noise ($\alpha = -1$).

Independence

The output entropy density of a TRNG design can be affected by a combination of factors, including multiple noise sources, various circuit elements and external influences. While certain factors (e.g., specific types of noise sources) may contribute significantly to the output entropy, others (e.g., external influences) are considered known and potentially manipulable by an external observer, and therefore not contributing any meaningful entropy. Consequently, the stochastic model should evaluate the entropy generated by each contributing factor separately. To simplify this analysis, it is common to assume *independence*

between the different factors. The following paragraphs outline some popular independence assumptions (often implicitly) stated in the literature and also utilized in this thesis.

Between Noise Sources Noise sources characterized by distinct α values in the power law behave differently, some sources introduce dependencies between consecutively generated output samples, while others do not. The various noise types are presumed to operate independently of one another, implying that the behavior of one source does not affect the entropy generated by all other sources. As discussed in more detail in chapter 4, this commonly leads to the rationale that solely recognizing the entropy contribution of the thermal noise source is adequate to establish a lower bound on the entropy produced by the ES.

Between Noise Sources and External Influences Noise sources are presumed to be uncontrollable by any external means, implying that the instantaneous value of a given noise source is independent of any external factors. By definition, any deterministic variation observed in the circuit under investigation due to an external change (such as an increase in oscillation frequency due to a decrease in ambient temperature) is not classified as part of a noise source.

Note. Statistical characteristics of a noise source may be influenced by external factors, for instance, the magnitude of thermal noise increases with absolute temperature. However, temperature only impacts the *distribution* of the thermal noise source, not the instantaneous thermal noise value itself.

Between Different Subcircuits Even sources of the same type (with equal α), but acting on different subcircuits (such as two distinct oscillators), are assumed independent. While there may be a deterministic relation between the two subcircuits, particularly if they are physically close, this relation is not considered part of the noise sources affecting the subcircuits individually. For example, capacitive coupling, which is a deterministic effect that can be predicted at design time, falls under this category. This assumption is also commonly encountered in the context of presuming that any randomness introduced in the sampling circuitry (such as *metastability* in the samplers) is unrelated to the timing jitter generated by the oscillators being sampled.

2.3.3 Platform Parameter Estimation

As discussed in section 2.2.3, evaluating an ES using a stochastic model cannot be considered independent of the physical implementation and the underlying

hardware platform. The stochastic model necessitates the estimation of various physical platform-related quantities (referred to as *platform parameters*) before an entropy estimate can be derived. Estimating these parameters requires conducting physical experiments on the intended hardware platform, closely matching the operating conditions of the ES as much as possible. The following subsections address the two most commonly encountered platform parameters found in stochastic models for oscillator-based ESs: *noise magnitude* and *propagation delay*.

Noise Magnitude

Chapter 3 addresses the measurement of a fundamental parameter for oscillator-based ESs: noise magnitude, which is also commonly referred to as noise strength, linear noise strength, jitter strength, or phase noise magnitude. This parameter determines the linear rate at which jitter accumulates in a free-running oscillator and is therefore closely associated with the strength of the thermal noise sources affecting the oscillator under examination. Additionally, chapter 3 also discusses the accumulation of timing jitter in a non-linear manner, which arises from noise sources other than thermal noise *dominating* the oscillator's behavior. The impact that dominating flicker noise has on the ES is further elaborated upon in chapter 3 as well.

Propagation Delay

The time taken for the output of a logic gate to respond to a change at one of its inputs is quantified by the propagation delay. For ROs, the summation of the propagation delays of the individual stages determines the final oscillation frequency. Additionally, the smallest achievable propagation delay determines the time *resolution* for a Time-to-Digital Converter (TDC) based on Delay Chains (DCs), as employed in the measurements described in chapter 3 and the ES architecture presented in chapter 7. It is often valuable to distinguish between the propagation delay affecting positive input edges and that affecting negative input edges, as these two quantities are not inherently equal. Operating conditions can influence the measured propagation delay, for instance, higher ambient temperatures or lower supply voltages often result in an increased propagation delay. This parameter was estimated multiple times throughout the work presented in this thesis, employing various methods ranging from indirectly measuring the propagation delay by observing the oscillation frequency of an RO to utilizing a *Monte Carlo* method by observing sampled edge positions.

2.4 Part II: Model Capabilities

The second part of this dissertation focuses on the outputs generated by the stochastic model. Figure 2.1 shows that the stochastic model, as previously described, facilitates the estimation of the expected output entropy density. Furthermore, the stochastic model provides insights into how the selected design parameters impact key ES performance metrics (such as entropy density, throughput, and energy consumption per produced amount of data). This understanding enables precise optimization of the design parameters to achieve optimal performance according to the requirements set for the ES.

2.4.1 Design Parameter Realization

Design parameters refer to the parameters within the ES architecture that are under the control of the designer.

Note. Design parameters differ fundamentally from platform parameters, as introduced in section 2.3.3. While design parameters are intentionally selected by the designer, platform parameters are inherent physical quantities derived from the intended hardware platform.

Once the overall architecture for the ES is determined, the design parameters dictate the remaining configuration options. In oscillator-based ES architectures, several key design parameters are typically identifiable: the length of the jitter accumulation interval (which determines the sampling rate and eventual ES throughput), the oscillation frequencies of the involved ROs, or the number of DC stages. Precise control over these design parameters is crucial, as they significantly impact the resolution at which random timing jitter can be sampled. As emphasized by [73], enhancing this timing resolution noticeably reduces the time required for accumulating random jitter, thereby benefiting the ES throughput or augmenting the entropy gathered per sample.

Improving Realization Effort and Robustness

While the stochastic model can accurately predict the required value for the design parameters to achieve optimal performance, it does not prescribe a method for physically attaining these values. Particularly under Process, Voltage and Temperature (PVT) variations, the optimal value for certain design parameters may vary. This stands in contrast to traditional digital circuits, which are generally less impacted by PVT variations.

The struggle required to achieve appropriate values for the design parameters was quantified using a *feasibility and repeatability* score by [70]. In this thesis, the term *design effort*, as employed by [93], or alternatively realization effort will be utilized to denote the effort demanded from the designer to implement a given ES architecture and realize all design parameters within an appropriate range.

Certain ES architectures (such as the Transition Effect Ring Oscillator (TERO)-ES [86] or the COherent Sampling ring Oscillator (COSO)-ES [43]) are notoriously hard to implement on FPGAs. Designers often had to resort to exhaustive trial and error across many physical locations on the FPGA fabric until, by chance, a suitable location was found that yielded an acceptable realization of design parameters. This painstaking process had to be repeated for each individual device. Additionally, as demonstrated by [11], any PVT variation might render the chosen FPGA location unsuitable. A robust ES design should function regardless of its position on the die and should be able to handle PVT variations. Consequently, this thesis advocates for ES designs to be *dynamically configurable*, enabling them to adapt to varying operating conditions without necessitating extensive design effort during implementation.

Configurable ROs

An important design parameter for oscillator-based ESs is the frequency at which the ROs operate. This frequency is determined by the sum of propagation delays of its individual stages. As previously mentioned, precise control over the RO frequency enhances both the performance and robustness of the ES. Depending on the hardware platform, various methods are available to implement a *configurable* RO architecture. This thesis considers two hardware platforms: FPGAs and ASICs.

FPGA Especially on FPGAs, the options for creating a configurable RO are limited: the only hardware elements available are LookUp tables (LUTs) and Flip-Flops (FFs), the wiring must utilize existing interconnects which significantly impact the RO frequency, and the architecture should ideally be described using only a Hardware Description Language (HDL). While many FPGAs support the use of Phase-Locked Loops (PLLs) to implement a tunable oscillator, the number of PLLs available on the device is often limited, and their primary purpose is to serve as sources of digital clocking signals. Given these constraints, chapter 5 presents and compares several highly portable, tunable RO architectures compatible with most popular FPGA families. All RO designs

demonstrated in chapter 5 exploit the inherent process variations in the LUTs and interconnects that constitute the FPGA fabric.

ASIC Designing a configurable RO on an ASIC platform offers considerable flexibility. However, in this thesis, it is essential that the RO architectures easily integrate with other digital logic. Therefore, the designs are restricted to using only digital voltages for control, consisting solely of transistors and interconnects (i.e., no dedicated capacitors or resistors), and having a layout capable of fitting within a standard cell row. Chapter 5 presents a configurable RO architecture that meets these constraints and demonstrates the achieved tunability across multiple ASIC technology nodes.

2.4.2 Exploiting Parameter Realization

The configurable RO topology allows TRNGs to be constructed precisely meeting the design parameters as determined by using a stochastic model. This ensures that the TRNG operates near optimal performance, maximizing efficiency given the available resources. This thesis introduces two distinct ES designs, each tailored for either the FPGA or the ASIC hardware platform in chapters 6 and 7 respectively. Furthermore, chapter 6 introduces a controller that continuously monitors a COSO-ES, dynamically adjusting its configuration inputs to maintain operation within a predetermined region. The introduction of the controller, alongside the configurable RO design, eliminates the necessity for user-defined placement constraints, significantly diminishing the demanded design effort. Chapter 7 provides a comprehensive demonstration, illustrating the utilization of a stochastic model in the process of selecting optimal design parameters for a custom ES.

2.5 Conclusion

This chapter contrasts the modern TRNG design flow with the now-considered obsolete TRNG design flow. The latter heavily relied on statistical testing to support any claims regarding the quality of the produced randomness, whereas the former uses a stochastic model description as the centerpiece of the ES verification procedure. The rise in adoption of this modern flow is mainly driven by international standards. Although progress is modest, more and more proposed ES designs now include a stochastic model [24], or at least provide clear outlines for constructing one [61].

The twofold contribution of this dissertation is illustrated by sections 2.3 and 2.4. Section 2.3 outlines part I of this thesis, focussing on the three main prerequisites required by the stochastic model: establishing the necessary entropy density, validating the assumptions made by the model, and experimentally determining the values of key platform parameters.

The contents of part II are discussed in section 2.4, where two main contributions are highlighted. Firstly, the stochastic model enables the exploitation of available ES design parameters to achieve a well-tuned ES implementation, enhancing performance at an equal or lower cost. Secondly, section 2.4.1 explains that realizing an optimal design parameter, given the limitations imposed by physical hardware, necessitates the use of dedicated circuit techniques.

Part I

Model Prerequisites

Characterizing Oscillator Phase Noise

Chapter 3

Jitter Measurement

This chapter is based on the following publication:

> **Characterization of Oscillator Phase Noise Arising From Multiple Sources for ASIC True Random Number Generation**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IEEE Transactions on Circuits and Systems I (TCAS I)*, 2024

Contribution: *main author*.

3.1 Background and Context

In *fully digital* architectures, an attractive choice is an ES topology based on free-running ROs [6, 43, 80, 86]. This approach permits to construct the ROs, together with the sampling circuitry, by exclusively using digital logic. Due to their entirely digital structure, these architectures are well-suited for use in both FPGAs and ASICs.

As demonstrated in chapter 2, international standardization bodies, such as NIST [83], BSI [69], and ISO [34], mandate the presence of a *stochastic model*, capable of estimating the available *entropy density* in the ES output stream. In order to generate an accurate entropy estimate, these models rely on a precise measurement of *platform-specific parameters*, e.g., gate delay or oscillator frequency [93]. Constituting a crucial subset of these parameters are the ones related to the rate at which *random* oscillator period variations accumulate through time. These parameters are often referred to as *timing jitter strength* or

phase noise [94]. Estimating the precise value of these parameters is essential, considering that in oscillator-based ESs, the oscillator period variations often are the sole source of fresh entropy and will consequently have a direct impact on key performance aspects of the ES, such as *entropy rate*, *throughput* and *energy efficiency*.

Oscillator *period jitter* or oscillator phase noise has been a subject of study for numerous years. Studies analyzing the use of ROs in PLLs frequently disregard, *flicker* FM noise, based on the rationale that low frequency variations would be attenuated by the PLL feedback loop [90]. However, in the context of the ES application, where such a feedback loop is absent, flicker noise can introduce long term dependencies in the ES output. In [55], a time domain study derives the magnitude of period length variations in a *differential* RO, taking into account various *noise sources* within the oscillator circuit. The analysis focused exclusively on *white* (uncorrelated) noise.

A frequency domain approach was presented in [29], building further on the theory outlined in [28]. Once more, only white noise sources are considered. Additionally, [29] relates the frequency and time domain concepts: phase noise and timing jitter respectively, under the *assumption* of a Wide-Sense Stationary (WSS) excess oscillator phase. Section 3.2.2 in this chapter demonstrates that this assumption, nonetheless, does not hold in the presence of white FM noise.

An approach to estimate the *phase diffusion rate* under the action of white FM noise was proposed by [20]. The study revealed that the time constant, at which the ensemble average amplitude decays, is closely linked to the linear (jitter variance that increases linearly with absolute time) jitter strength in the presence of white FM noise. Other attempts of estimating the linear jitter strength have been carried out in previous research: [84, 94]. Although [94] presents a promising topology, using DCs, both only consider the existence of *thermal* noise, whilst other *noise types*, e.g., flicker noise or *random walk* noise, might affect the free-running oscillator as well.

The existence of other noise types was acknowledged by [27, 46, 59]. In [59], the use of counters limited the available *time resolution*, resulting in a high *quantization noise floor* in the measurements. An alternative counter method outlined in [49] employs two counters to effectively double the resolution. As also concluded by [56], the extended *accumulation time* necessary for the counter method renders it impractical for thermal noise measurement. The long accumulation times are essential to mitigate the impact of the high quantization noise floor, however, they consequently also result in flicker noise *dominating* over thermal noise. The existence of a *noise corner* (accumulation time lengths above which flicker noise dominates over thermal noise) was recognized by both [27] and [46]. However, in these works, no measurements have been

presented for accumulation time lengths below 120 µs and 3.5 µs, respectively. Modern oscillator-based ES designs often achieve a throughput higher than $1\,\mathrm{Mbit\,s^{-1}}$ [24, 93], which means jitter accumulation times lower than 1 µs are relevant.

Considering the room for improvement left by previous jitter strength estimates, this chapter introduces a jitter measurement methodology implemented in a 65 nm Complementary Metal-Oxide-Semiconductor (CMOS) ASIC technology. The TDC is based on DCs, as was proposed by [94]. By using DCs, a time resolution less than 100 ps could be achieved, which allows investigating jitter strength for accumulation time lengths as short as 30 ns.

The main contributions of this chapter are:

- A 65 nm CMOS ASIC chip is fabricated, capable of estimating the available *noise strength* in free-running oscillators dedicated for use in ESs.
- Using DCs, a time resolution less than 100 ps, determined by the intrinsic gate delay of one inverter gate, is achieved. This resolution allows for more than a factor of ten increase in measurement precision compared to other techniques [27, 46, 84] and is comparable with previous jitter measurement attempts using DCs [94].
- A wide selection of oscillator phase noise sources are considered. For each noise type, a time-based *analytical model* is proposed, allowing to determine the region where one particular type dominates over other noise types.
- A broad spectrum of accumulation time lengths has been investigated, covering a range from 30 ns to 0.3 s. This enabled to detect long-term dependencies, while simultaneously accommodating high-throughput ESs.

This chapter is structured as follows: section 3.2 lists five different noise types and derives the oscillator phase variance for each of these noise types analytically. The differential jitter measurement platform, together with a model describing the measurement results is shown in section 3.3. Section 3.4 presents the set-up calibration process and shows the jitter measurement results. The phase stationarity misconception, often found in literature is discussed and a detailed comparison to previous attempts of oscillator jitter strength measurement are shown in section 3.5. This chapter is then concluded in section 3.6.

3.2 Noisy Oscillator Model

This section explains the application of a *power law* model to describe the phase of a free-running oscillator. As discussed in section 2.3.2, this assumption forms the foundation of the oscillator model developed in this section.

For a formal introduction to the mathematical concepts used in this chapter, please refer to appendix A.

3.2.1 Power Law Model

The phase of a noisy oscillator can be described as a *random process* through time, $\Phi : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ by

$$\Phi(\omega, t) = 2\pi f_n t + \phi_0 + \Phi_e(\omega, t),$$

with f_n the nominal oscillator frequency, ϕ_0 the initial phase at time $t = 0$, and $\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$ a random process, describing the excess phase through time. We now define the relative frequency deviation.

Definition 3.1. (Relative frequency deviation) The relative frequency deviation is a random process: $Y : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ by

$$Y(\omega, t) = \frac{\frac{\mathrm{d}\Phi}{\mathrm{d}t}(\omega, t) - 2\pi f_n}{2\pi f_n} = \frac{1}{2\pi f_n}\frac{\mathrm{d}\Phi_e}{\mathrm{d}t}(\omega, t).$$

The expected value for the excess phase is equal to, $\forall t \in \mathbb{R}_{\geq 0}$:

$$\mathbf{E}\big[\Phi_e(t)\big] = \mathbf{E}\left[2\pi f_n \int_0^t Y(\theta)\mathrm{d}\theta\right] = 2\pi f_n \int_0^t \mathbf{E}\big[Y(\theta)\big]\mathrm{d}\theta = 0, \tag{3.1}$$

as is assumed that $\forall\theta \in \mathbb{R}_{\geq 0} : \mathbf{E}[Y(\theta)] = 0$, by the definition of f_n equaling the expected frequency constant: $2\pi f_n = \mathbf{E}\big[\frac{\mathrm{d}\Phi}{\mathrm{d}t}(t)\big]$, and it is further assumed that

$\Phi_e(0) \sim 0$. The variance for the excess phase is then equal to

$$\begin{aligned}\forall t \in \mathbb{R}_{\geq 0} : \mathbf{Var}\big[\Phi_e(t)\big] = \mathbf{E}\big[\Phi_e^2(t)\big] &= \mathbf{E}\left[\left(2\pi f_n \int_0^t Y(\theta)\mathrm{d}\theta\right)^2\right] \\ &= \mathbf{E}\left[4\pi^2 f_n^2 \int_0^t Y(\theta_i)\mathrm{d}\theta_i \int_0^t Y(\theta_j)\mathrm{d}\theta_j\right] \\ &= 4\pi^2 f_n^2 \int_0^t \int_0^t \mathbf{E}\big[Y(\theta_i)Y(\theta_j)\big]\mathrm{d}\theta_i\mathrm{d}\theta_j \\ &= 4\pi^2 f_n^2 \int_0^t \int_0^t R_Y(\theta_i,\theta_j)\mathrm{d}\theta_i\mathrm{d}\theta_j,\end{aligned} \tag{3.2}$$

with R_Y the Auto-Correlation Function (ACF) for the relative frequency deviation. If $\{Y(t)\}_{t\in\mathbb{R}_{\geq 0}}$ is a WSS random process, this ACF only depends on the time shift: $\tau = \theta_j - \theta_i$.

The following relation holds for the Power Spectral Density (PSD):

$$S_Y(f) = \left(\frac{1}{2\pi f_n}\right)^2 (2\pi f)^2 S_{\Phi_e}(f) = \left(\frac{f}{f_n}\right)^2 S_{\Phi_e}(f), \tag{3.3}$$

with S_Y, S_{Φ_e} and f the PSD of $\{Y(t)\}_{t\in\mathbb{R}_{\geq 0}}$, $\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$ and the Fourier frequency respectively. According to [32], the relative frequency deviation PSD can be accurately modeled by a power law:

$$S_Y(f) = \sum_{\alpha=-2}^{2} h_\alpha |f|^\alpha, \tag{3.4}$$

with h_α, the proportionality constant for the corresponding noise type. Table 3.1 provides an overview of the most common noise types and the corresponding value for α in eq. (3.4). The noise type may either refer to the oscillator frequency: *Frequency Modulated (FM)*, or to the oscillator phase: *Phase Modulated (PM)*. The noise type name refers to the shape of the oscillator frequency or phase PSD.

A physical interpretation for each of the five most common noise types is presented in [32], section XII. In summary: due to the inherent integration action of phase errors (the influence of past phase disturbances persists, as there is no restoring force bringing the excess phase back to zero [28]), circuit-level noise sources appear as close in-band phase noise in the oscillator output voltage spectrum. *White-*, flicker- and *burst* noise sources at the circuit level therefore appear as $1/|f|^2$ ($\alpha = 0$), $1/|f|^3$ ($\alpha = -1$) and $1/|f|^4$ ($\alpha = -2$) respectively in

Table 3.1: Power law noise types.

Noise type	α in eq. (3.4)
Random walk FM	-2
Flicker FM	-1
White FM	0
Flicker PM	1
White PM	2

the phase spectrum (note the relation between the phase and frequency spectra given by eq. (3.3)). Output stage nonlinearities upconvert white- and flicker noise sources to a constant ($\alpha = 2$) and $1/|f|$ ($\alpha = 1$) in-band phase spectrum respectively.

For each type of noise, the excess phase variance function, $\mathbf{Var}\big[\Phi_e(t)\big]$ for $t \in \mathbb{R}_{\geq 0}$, can be derived. The final excess phase variance can be obtained by summing the contributions of each individual noise type. The following subsections will handle each of the noise types listed in table 3.1 in order of relevance.

3.2.2 White FM Noise

The relative frequency deviation PSD is constant: $S_Y(f) = h_0$. The assumption is made that, in the presence of only white FM noise, $\{Y(t)\}_{t\in\mathbb{R}_{\geq 0}}$ is a WSS process. Often (e.g., in [16, 27, 29]), the assumption of a WSS phase signal is made. However, as will be shown, any noise type other than white Phase Modulated (PM) noise and flicker PM noise will violate this assumption.

The Wiener-Khinchin theorem allows to calculate the ACF in terms of the PSD:

$$R_Y(\tau) = \int_{-\infty}^{\infty} S_Y(f) \mathrm{e}^{2\pi \mathrm{j} \tau f} \mathrm{d}f = h_0 \delta(\tau). \tag{3.5}$$

The variance of the excess phase can now be determined, by substituting the result of eq. (3.5) into eq. (3.2), $\forall t \in \mathbb{R}_{\geq 0}$:

$$\mathbf{Var}\big[\Phi_e(t)\big] = 4\pi^2 f_n^2 \int_0^t \int_0^t h_0 \delta(\theta_j - \theta_i) \mathrm{d}\theta_i \mathrm{d}\theta_j = 4\pi^2 f_n^2 h_0 t. \tag{3.6}$$

Equation (3.6) makes it evident that $\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$ cannot be regarded a WSS process since, in the presence of white FM noise, its variance function depends on the absolute time, t.

The linear relation given by eq. (3.6) is a well-known result in time domain jitter analysis of free-running ROs, when only white noise sources are considered. In [90], the cycle-to-cycle jitter was acknowledged to scale linearly with the number of inverter stages in the RO loop, which by itself is proportional to the oscillation period length. In [20, 55, 66], the quantity $\sqrt{4\pi^2 f_n^2 h_0}$ is referred to as $2D$, κ and $\sqrt{F_{noise}}$ in the respective notation.

3.2.3 Flicker FM Noise

The relative frequency deviation PSD is inversely proportional to the Fourier frequency: $S_Y(f) = \frac{h_{-1}}{|f|}$. As noticed by [52], the general random process $\{Y(t)\}_{t\in\mathbb{R}_{\geq 0}}$, described in eq. (3.4), cannot be regarded as WSS for $\alpha \leq -1$. In practice however, only a bandlimited version of $\{Y(t)\}_{t\in\mathbb{R}_{\geq 0}}$ can be measured [88]. Assuming the bandlimited process is WSS, the ACF for the relative frequency deviation is computed, using the Wiener-Khinchin theorem, as

$$\begin{aligned} R_Y(\tau) &= \int_{-\infty}^{\infty} S_Y(f) \mathrm{e}^{2\pi \mathrm{j}\tau f} \mathrm{d}f = \int_{-f_h}^{-f_l} \frac{h_{-1}}{|f|} \mathrm{e}^{2\pi \mathrm{j}\tau f} \mathrm{d}f + \int_{f_l}^{f_h} \frac{h_{-1}}{|f|} \mathrm{e}^{2\pi \mathrm{j}\tau f} \mathrm{d}f \\ &= 2h_{-1} \int_{f_l}^{f_h} \frac{\cos(2\pi f\tau)}{f} \mathrm{d}f, \end{aligned} \tag{3.7}$$

with f_h, f_l the high, and low frequency limits on S_Y respectively. Substituting the result of eq. (3.7) into eq. (3.2) results in, $\forall t \in \mathbb{R}_{\geq 0}$:

$$\mathbf{Var}\big[\Phi_e(t)\big] = 8\pi^2 f_n^2 h_{-1} \int_0^t \int_0^t \int_{f_l}^{f_h} \frac{\cos\big(2\pi f(\theta_j - \theta_i)\big)}{f} \mathrm{d}f \mathrm{d}\theta_i \mathrm{d}\theta_j. \tag{3.8}$$

Solving eq. (3.8), by changing the order of integration, results in the following expression for the variance of the excess phase depending on the absolute time t, $\forall t \in \mathbb{R}_{>0}$:

$$\begin{aligned} \mathbf{Var}\big[\Phi_e(t)\big] = 8\pi^2 f_n^2 h_{-1} t^2 \Bigg(&- \frac{\sin^2(\pi f_h t)}{2(\pi f_h t)^2} - \frac{\sin(2\pi f_h t)}{2\pi f_h t} + \mathrm{Ci}(2\pi f_h t) \\ &+ \frac{\sin^2(\pi f_l t)}{2(\pi f_l t)^2} + \frac{\sin(2\pi f_l t)}{2\pi f_l t} - \mathrm{Ci}(2\pi f_l t) \Bigg), \end{aligned} \tag{3.9}$$

and $\mathbf{Var}\big[\Phi_e(0)\big] = 0$. The cosine integral function, Ci, is defined in appendix A.1. Letting $f_h \to \infty$ and using the property: $\lim_{x\to\infty} \mathrm{Ci}(x) = 0$, eq. (3.9) is simplified to, $\forall t \in \mathbb{R}_{>0}$:

$$\mathbf{Var}\big[\Phi_e(t)\big] = 8\pi^2 f_n^2 h_{-1} t^2 \left(\frac{\sin^2(\pi f_l t)}{2(\pi f_l t)^2} + \frac{\sin(2\pi f_l t)}{2\pi f_l t} - \mathrm{Ci}(2\pi f_l t) \right). \tag{3.10}$$

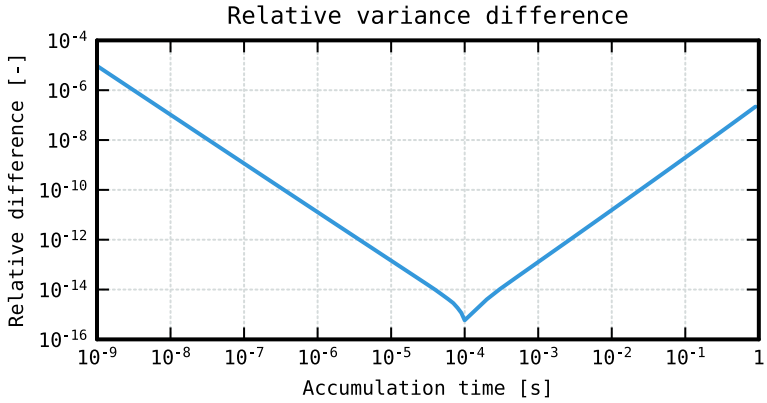

Figure 3.1: Relative difference between eq. (3.9) and eq. (3.11). Lower and upper frequency limits, f_l and f_h, equal 1 mHz and 10 GHz respectively.

Additionally, the same assumption as in [88] is made here, namely: the lower frequency limit, f_l, is much smaller than the reciprocal of the maximal measurement time, $t_{\max}$. Any contribution to $\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$, of frequencies sufficiently lower than $\frac{1}{t_{\max}}$, will be perceived as a constant frequency offset and therefore do not contribute to the measured variance estimate, as this offset is captured by the nominal frequency, f_n, estimate. The time, t, at which $\mathbf{Var}\big[\Phi_e(t)\big]$ is evaluated will be bounded by $t_{\max}$: $t \leq t_{\max}$. Therefore: $2\pi f_l t \leq 2\pi f_l t_{\max} \ll 1$. Using the Taylor series of $\mathrm{Ci}(x)$ around $x = 0$, eq. (3.10) can be further simplified to

$$\forall t \in \mathbb{R}_{>0} : \mathbf{Var}\big[\Phi_e(t)\big] \approx 4\pi^2 f_n^2 h_{-1} t^2 \big(3 - 2\gamma - 2\ln(2\pi f_l t)\big). \tag{3.11}$$

Figure 3.1 compares the outcome of eq. (3.9) to the simplified expression in eq. (3.11) for realistic values of f_l and f_h. Within the time interval: 1 ns to 1 s, eq. (3.11) approximates eq. (3.9) very well, having a relative error of at most 1×10^{-5}.

3.2.4 Random Walk FM Noise

The relative frequency deviation PSD is now inversely proportional to the Fourier frequency squared: $S_Y(f) = \frac{h_{-2}}{f^2}$. We now define the relative phase acceleration.

Definition 3.2. (Relative phase acceleration) The relative phase acceleration is a random process: $A : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ by

$$A(\omega, t) = \frac{\mathrm{d}Y}{\mathrm{d}t}(\omega, t) = \frac{1}{2\pi f_n} \frac{\mathrm{d}^2 \Phi_e}{\mathrm{d}t^2}(\omega, t).$$

The relative phase acceleration PSD is therefore constant:

$$S_A(f) = (2\pi f)^2 S_Y(f) = (2\pi)^2 h_{-2}.$$

Following an identical reasoning as in section 3.2.1 and assuming $\{A(t)\}_{t \in \mathbb{R}_{\geq 0}}$ is WSS, the relative frequency deviation ACF can be related to the relative phase acceleration ACF:

$$R_Y(\theta_i, \theta_j) = \int_0^{\theta_j} \int_0^{\theta_i} R_A(\nu_j - \nu_i) \mathrm{d}\nu_i \mathrm{d}\nu_j,$$

with R_A, the relative phase acceleration ACF. Similar as in section 3.2.2, this ACF is described by a scaled Dirac delta distribution: $R_A(\tau) = 4\pi^2 h_{-2} \delta(\tau)$. The relative frequency deviation ACF is therefore equal to

$$R_Y(\theta_i, \theta_j) = 4\pi^2 h_{-2} \int_0^{\theta_j} \int_0^{\theta_i} \delta(\nu_j - \nu_i) \mathrm{d}\nu_i \mathrm{d}\nu_j = 4\pi^2 h_{-2} \min(\theta_i, \theta_j).$$

Substituting this result into eq. (3.2), $\forall t \in \mathbb{R}_{\geq 0}$:

$$\mathbf{Var}\big[\Phi_e(t)\big] = 16\pi^4 f_n^2 h_{-2} \int_0^t \int_0^t \min(\theta_i, \theta_j) \mathrm{d}\theta_i \mathrm{d}\theta_j = \frac{16}{3} \pi^4 f_n^2 h_{-2} t^3.$$

3.2.5 Flicker PM Noise

The relative frequency deviation PSD is now proportional to the absolute Fourier frequency: $S_Y(f) = h_1 |f|$. Using eq. (3.3), the excess phase PSD becomes: $S_{\Phi_e}(f) = f_n^2 h_1 \frac{1}{|f|}$. Similar as in section 3.2.3, $\{\Phi_e(t)\}_{t \in \mathbb{R}_{\geq 0}}$ is assumed WSS by bandlimiting the PSD to the frequency interval $[f_l, f_h]$. The excess phase variance can then be found, with R_{Φ_e}, the excess phase ACF, $\forall t \in \mathbb{R}_{>0}$:

$$\mathbf{Var}\big[\Phi_e(t)\big] = \mathbf{E}\big[\Phi_e^2(t)\big] = R_{\Phi_e}(0) = \int_{-\infty}^{\infty} S_{\Phi_e}(f) \mathrm{d}f = 2 f_n^2 h_1 \int_{f_l}^{f_h} \frac{\mathrm{d}f}{f}$$

$$= 2 f_n^2 h_1 \ln\Big(\frac{f_h}{f_l}\Big).$$

Table 3.2: Noise types overview.

Noise type	α in eq. (3.4)	$\mathbf{Var}\big[\Phi_e(t)\big]$
Random walk FM	-2	$h_{-2}\frac{16}{3}\pi^4 f_n^2 t^3$
Flicker FM	-1	$h_{-1}4\pi^2 f_n^2 t^2\big(3-2\gamma-2\ln(2\pi f_l t)\big)$
White FM	0	$h_0 4\pi^2 f_n^2 t$
Flicker PM	1	$h_1 2 f_n^2 \ln\big(\frac{f_h}{f_l}\big)$
White PM	2	$h_2 2 f_n^2 f_h$

3.2.6 White PM Noise

The relative frequency deviation PSD is now proportional to the Fourier frequency squared: $S_Y(f) = h_2 f^2$. Using eq. (3.3), the excess phase PSD becomes constant: $S_{\Phi_e}(f) = f_n^2 h_2$. The excess phase is now modeled by WSS white noise with infinite variance. By bandlimiting the PSD to the frequency interval $[0, f_h]$, a finite excess phase variance is obtained, $\forall t \in \mathbb{R}_{>0}$:

$$\mathbf{Var}\big[\Phi_e(t)\big] = \int_{-\infty}^{\infty} S_{\Phi_e}(f)\mathrm{d}f = 2f_n^2 h_2 \int_0^{f_h} \mathrm{d}f = 2f_n^2 h_2 f_h.$$

3.2.7 Power Law Model Summary

Table 3.2 overviews the excess phase variance for all noise types discussed. Figure 3.2 compares the contributions of these noise types, for increasing accumulation time, t. Depending on the magnitudes of h_α, each noise type can become dominant in a certain accumulation time interval. The total excess phase variance is only determined by the dominant noise type.

3.3 Jitter Measurement Model

This section proposes an analytical model, describing phase variance accumulation in a differential RO arrangement. Initially, the architecture and principle of operation is discussed. Subsequently, an analytical model is derived for the architecture and finally, the impact of quantization noise on the measurements is addressed.

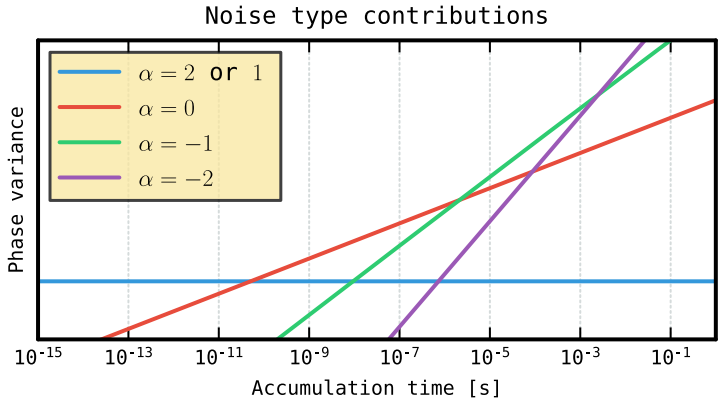

Figure 3.2: Exemplifying contribution of the different noise types to the total phase variance, using log-log axes. The relative values used for the proportionality constants are: $h_{-2} = 1 \times 10^{27}\,\mathrm{s}^{-1}$, $h_{-1} = 1 \times 10^{24}$, $h_0 = 1 \times 10^{20}\,\mathrm{s}$, $h_2 = 1\,\mathrm{s}^3$. The noise contribution corresponding with h_1 was not drawn as it is similar to h_2. The lower and upper frequency limits used are 10 µHz and 100 GHz, respectively.

3.3.1 Differential Jitter Measurement Set-up

Architecture

A differential jitter measurement architecture, comparable to the one proposed by [94], was used in this chapter. Conducting the measurements using a differential set-up reduces the influence of external (via the substrate or via the supply network) noise sources. As only the accumulated phase difference between the two ROs is of interest, any perturbation equally affecting both ROs will be canceled out in the phase difference.

As depicted by fig. 3.3, the set-up consists of two free-running ROs: `RO0` and `RO1`, connected to two DCs: `DC0` and `DC1`. Both ROs are designed in an identical manner, comprising a NAND gate followed by two *configurable inverters*. The signal: `RO_ENABLE`, connected to both RO's NAND gates allows to switch the ROs on or off. The drive strength of the configurable inverter can be modified by activating or deactivating additional CMOS transistor pairs, as illustrated in fig. 3.4. This technique allows for a precise control over the produced RO oscillation frequency and will be analyzed in greater detail in chapter 5. Additionally, a coarse-grained control over the oscillation frequency is achieved by attaching a frequency scaler to the RO output, enabling the reduction of the

RO frequency by a power of two.

Both DCs comprise 128 stages, and each stage is implemented as a single inverter. The output of each DC stage is connected to the data input of a Data Flip-Flop (DFF). To counteract the inversion created by employing a single inverter in each DC stage, the inverted or non-inverted data output of the FF is utilized alternately. A shared clock signal, `DC_CLK`, is used to clock each FF.

A NOR gate links the last DC stage output to the input of an asynchronous counter. The NOR gate disallows further clocking of the counter, once a positive edge of the `DC_CLK` signal occurs. Data from both DCs and counters is collected for subsequent analysis.

Working

A timing diagram in fig. 3.5 shows a phase measurement action. First, both ROs are enabled at the same time by raising the `RO_ENABLE` signal. Edges generated by both ROs propagate through the DCs. The counters keep track of the occurred number of edges. After some accumulation time, the state of the DCs is captured by raising the `DC_CLK` signal. The captured DC state, acts as a time window, in which the precise location of the RO edge can be determined. The state of the counter enables to determine the number of edges that precede the captured DC time window. Combining the DC and counter states allows for accurately determining the RO phase at the exact capturing moment. In [94], this principle is discussed in greater detail.

3.3.2 Phase Difference Variance Model

An analytical model is composed to describe the variance of the RO phase difference as a function of the applied accumulation time length. The phase for both ROs, is described as follows: $\Phi_0, \Phi_1 : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ by

$$\Phi_0(\omega, t) = 2\pi f_{n0} t + \Phi_{e0}(\omega, t) + \Phi_g(\omega, t),$$
$$\Phi_1(\omega, t) = 2\pi f_{n1} t + \Phi_{e1}(\omega, t) + \Phi_g(\omega, t),$$

with f_{n0}, f_{n1} the nominal frequency, and $\{\Phi_{e0}(t)\}_{t\in\mathbb{R}_{\geq 0}}$, $\{\Phi_{e1}(t)\}_{t\in\mathbb{R}_{\geq 0}}$ the excess phase random process for `RO0`, `RO1`, respectively. Both phases start at initial phase zero: $\Phi_{e0}(0) \sim 0$ and $\Phi_{e1}(0) \sim 0$. The global phase perturbation, common to both ROs, is represented by $\{\Phi_g(t)\}_{t\in\mathbb{R}_{\geq 0}}$. By working with the phase difference, any global contribution to the oscillator phase can be eliminated:

$$\forall t \in \mathbb{R}_{\geq 0} : \Phi_0(\omega, t) - \Phi_1(\omega, t) = 2\pi t(f_{n0} - f_{n1}) + \Phi_{e0}(\omega, t) - \Phi_{e1}(\omega, t).$$

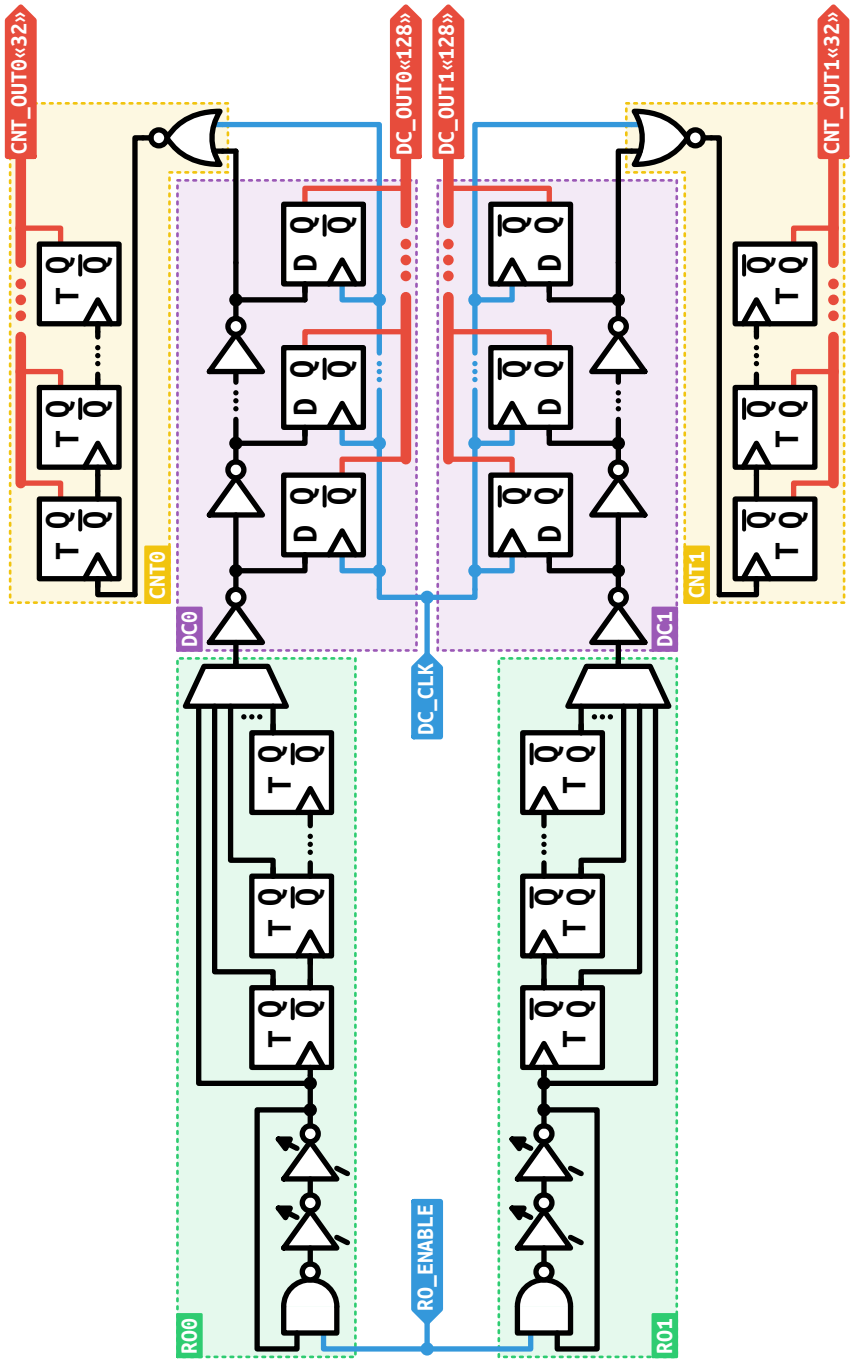

Figure 3.3: Detailed jitter measurement architecture.

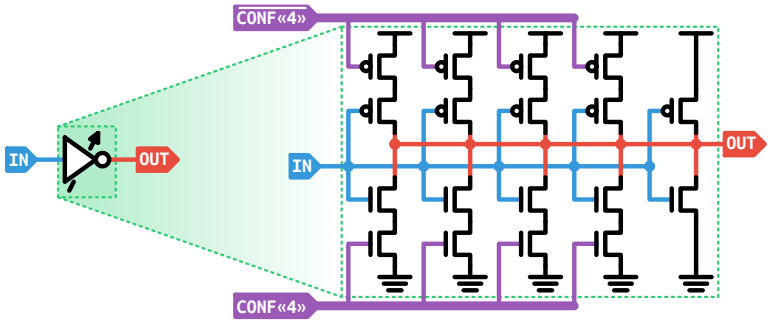

Figure 3.4: Detailed configurable inverter architecture.

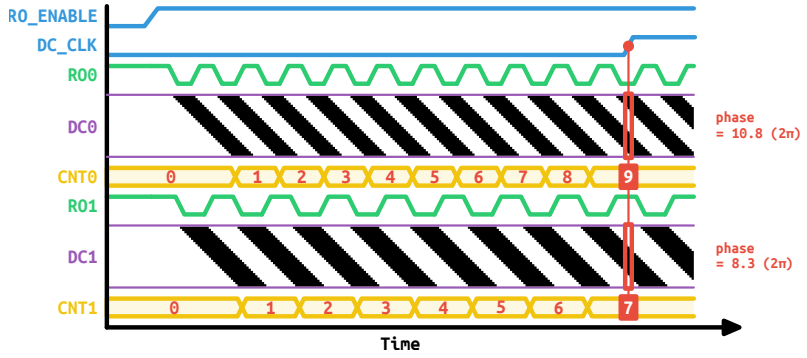

Figure 3.5: Phase measurement timing.

This phase difference is evaluated at a moment in time: $t_a \in \mathbb{R}_{\geq 0}$. However, due to measurement uncertainty, this moment in time will be a realization of a *random variable*, T_a, assumed *independent* for each measurement. The expected value for T_a equals μ_{T_a}, which is the accumulation time length aimed by the measurement set-up and is therefore always strictly positive: $\mu_{T_a} > 0$. The expected value for the phase difference, evaluated at T_a is

$$\mathbf{E}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = 2\pi(f_{n0} - f_{n1})\mu_{Ta},$$

as from eq. (3.1), $\forall t \in \mathbb{R}_{\geq 0} : \mathbf{E}\big[\Phi_{e0}(t)\big] = \mathbf{E}\big[\Phi_{e1}(t)\big] = 0$. The variance is calculated as

$$\begin{aligned}\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = \mathbf{E}\Big[\big(&2\pi T_a(f_{n0} - f_{n1}) + \Phi_{e0}(T_a) - \Phi_{e1}(T_a) \\ &- 2\pi(f_{n0} - f_{n1})\mu_{Ta}\big)^2\Big].\end{aligned} \tag{3.13}$$

Working out the square in eq. (3.13) and using: $\mathbf{E}\big[\Phi_{e0}(T_a)\big] = \mathbf{E}\big[\Phi_{e1}(T_a)\big] = 0$, the following is obtained:

$$\begin{aligned}\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = {}& 4\pi^2(f_{n0} - f_{n1})^2\mathbf{Var}[T_a] \\ &+ \mathbf{Var}\big[\Phi_{e0}(T_a)\big] + \mathbf{Var}\big[\Phi_{e1}(T_a)\big] \\ &+ 4\pi(f_{n0} - f_{n1})\Big(\mathbf{E}\big[T_a\Phi_{e0}(T_a)\big] - \mathbf{E}\big[T_a\Phi_{e1}(T_a)\big]\Big) \\ &- 2\mathbf{E}\big[\Phi_{e0}(T_a)\Phi_{e1}(T_a)\big].\end{aligned} \tag{3.14}$$

For the fourth term: $\mathbf{E}\big[T_a\Phi_{e0}(T_a)\big] = \mathbf{E}\big[T_a\Phi_{e1}(T_a)\big] = 0$, again due to $\forall t \in \mathbb{R}_{\geq 0} : \mathbf{E}\big[\Phi_{e0}(t)\big] = \mathbf{E}\big[\Phi_{e1}(t)\big] = 0$, from eq. (3.1).

Assuming the excess phases of `ROO` and `RO1` are independent, zeros the fifth term as well, as shown by the following relation:

$$\begin{aligned}\mathbf{E}\big[\Phi_{e0}(T_a)\Phi_{e1}(T_a)\big] &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\phi_0\phi_1 f_{\Phi_{e0}(t_a)|T_a=t_a}(\phi_0) \\ &\qquad f_{\Phi_{e1}(t_a)|T_a=t_a}(\phi_1) f_{T_a}(t_a)\mathrm{d}t_a\mathrm{d}\phi_0\mathrm{d}\phi_1 \\ &= \int_{-\infty}^{\infty} f_{T_a}(t_a)\int_{-\infty}^{\infty}\phi_1 f_{\Phi_{e1}(t_a)|T_a=t_a}(\phi_1) \\ &\qquad \mathbf{E}\big[\Phi_{e0}(t_a) \mid T_a = t_a\big]\mathrm{d}\phi_1\mathrm{d}t_a \\ &= 0,\end{aligned}$$

because $\forall t_a \in \mathbb{R}_{\geq 0} : \mathbf{E}\big[\Phi_{e0}(t_a) \mid T_a = t_a\big] = 0$. Now, eq. (3.14) can be simplified to

$$\begin{aligned}\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = {}& 4\pi^2(f_{n0} - f_{n1})^2\mathbf{Var}[T_a] \\ &+ \mathbf{Var}\big[\Phi_{e0}(T_a)\big] + \mathbf{Var}\big[\Phi_{e1}(T_a)\big].\end{aligned} \tag{3.15}$$

Equation (3.15) shows an additional advantage of performing differential phase measurements. The contribution of the measurement uncertainty, $\mathbf{Var}[T_a]$, to

the total measured variance of the phase difference is reduced by a frequency offset: $f_{n0} - f_{n1}$. By choosing a small enough offset, the measurement results are dominated by the second and third terms in eq. (3.15).

The variances: $\mathbf{Var}\big[\Phi_{e0}(T_a)\big]$ and $\mathbf{Var}\big[\Phi_{e1}(T_a)\big]$, are still dependent on the measurement uncertainty, represented by T_a, and might therefore overestimate the variance at a fixed point in time, $t \in \mathbb{R}_{\geq 0}$: $\mathbf{Var}\big[\Phi_{e0}(t)\big]$, $\mathbf{Var}\big[\Phi_{e0}(t)\big]$.

The excess phase variance of an oscillator that can be modeled by a power law, eq. (3.4), is equal to the sum of the contributions of each different noise type. In section 3.2, the relation between the accumulation time and each noise contribution was described. The excess phase variance is then estimated as a sum of components contributed by each noise type:

$$\forall t \in \mathbb{R}_{\geq 0} : \mathbf{Var}\big[\Phi_e(t)\big] = f_n^2\big(c_0 + c_1 t + c_{20} t^2 + c_{21} t^2 \ln(t) + c_3 t^3\big), \tag{3.16}$$

with the coefficients c_0, c_1, c_{20} and c_{21}, c_3, corresponding to the noise types: $\alpha = 1, 2$, $\alpha = 0$, $\alpha = -1$, $\alpha = -2$ in table 3.2, respectively. All constant factors in the expressions for the excess phase variance are incorporated into a single constant: c_i, except for the nominal oscillating frequency, f_n, which is common to all noise types. The accumulation time, T_a, is assumed normally distributed: $T_a \sim \mathcal{N}(\mu_{T_a}, \sigma^2_{T_a})$. The excess phase variance at a varying accumulation time becomes:

$$\begin{aligned}
\mathbf{Var}\big[\Phi_e(T_a)\big] &= \mathbf{E}\Big[\mathbf{Var}\big[\Phi_e(t_a) \mid T_a = t_a\big]\Big] \\
&= \mathbf{E}\Big[f_n^2\big(c_0 + c_1 T_a + c_{20} T_a^2 + c_{21} T_a^2 \ln(T_a) + c_3 T_a^3\big)\Big] \\
&= f_n^2\Bigg(c_0 + c_1 \mu_{T_a} + c_{20}\big(\mu^2_{T_a} + \sigma^2_{T_a}\big) \\
&\qquad + c_{21}\Big(\frac{3}{2}\sigma^2_{T_a} + \ln(\mu_{T_a})\big(\mu^2_{T_a} + \sigma^2_{T_a}\big)\Big) \\
&\qquad + c_3\big(\mu^3_{T_a} + 3\mu_{T_a}\sigma^2_{T_a}\big)\Bigg),
\end{aligned}$$

where $\mathbf{E}\big[T_a^2 \ln(T_a)\big] \approx \frac{3}{2}\sigma^2_{T_a} + \ln(\mu_{T_a})\big[\mu^2_{T_a} + \sigma^2_{T_a}\big]$, using the Taylor expansion of $t^2 \ln(t)$ around $t = \mu_{T_a}$. Equation (3.15), again under the assumption of T_a normally distributed and assuming equal noise strengths in both ROs, now

becomes:

$$\begin{aligned}\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = 4\pi^2(f_{n0} - f_{n1})^2\sigma_{T_a}^2 \\ + \big(f_{n0}^2 + f_{n1}^2\big)\Big(c_0 + c_1\mu_{T_a} + c_{20}\big(\mu_{T_a}^2 + \sigma_{T_a}^2\big) \\ + c_{21}\Big(\frac{3}{2}\sigma_{T_a}^2 + \ln(\mu_{T_a})\big(\mu_{T_a}^2 + \sigma_{T_a}^2\big)\Big) \\ + c_3\big(\mu_{T_a}^3 + 3\mu_{T_a}\sigma_{T_a}^2\big)\Big).\end{aligned} \tag{3.17}$$

3.3.3 Quantization Noise

The DCs only allow to measure the time elapsed since the last edge before the capturing moment, T_a, was outputted out of the RO. This measured elapsed time is denoted as a random process through time: $\{\hat{T}_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$. The total time elapsed since the RO was enabled from the perspective of the RO is given as

$$\forall t \in \mathbb{R}_{\geq 0} : \hat{T}_{e_{tot}}(\omega, t) = \frac{N(\omega, t)}{f_n} + \hat{T}_e(\omega, t),$$

where $\{N(t)\}_{t\in\mathbb{R}_{\geq 0}}$ represents the number of full periods counted by the counters since the RO was enabled. The random process, $\{T_{e_{tot}}(t)\}_{t\in\mathbb{R}_{\geq 0}}$, describes the time elapsed as if the RO is a perfectly stable clocking source. Only in a *noise-free* environment, will this random process exactly equal the absolute time, t. The measured RO phase is then derived as follows, $\forall t \in \mathbb{R}_{\geq 0}$:

$$\hat{\Phi}(\omega, t) = 2\pi f_n \hat{T}_{e_{tot}}(\omega, t) = 2\pi N(\omega, t) + 2\pi f_n \hat{T}_e(\omega, t). \tag{3.18}$$

The DCs enable to measure an edge position up to a precision of one inverter gate delay. The difference between the measured elapsed time, $\{\hat{T}_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$, and the actual elapsed time, $\{T_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$, is modeled by an independent, uniform quantization error, $Q : \Omega \to [-d_{LSB}, 0]$:

$$\forall t \in \mathbb{R}_{\geq 0} : \hat{\Phi}(\omega, t) = 2\pi N(\omega, t) + 2\pi f_n\big(T_e(\omega, t) + Q(\omega)\big),$$

with Q uniformly distributed: $Q \sim \mathcal{U}(-d_{LSB}, 0)$ and d_{LSB} indicating the DC stage (inverter gate) delay. The variance of a phase measurement is then

calculated as, $\forall t \in \mathbb{R}_{\geq 0}$:

$$\begin{aligned}\mathbf{Var}\big[\hat{\Phi}(t)\big] &= \mathbf{Var}\Big[2\pi N(t) + 2\pi f_n\big(T_e(t) + Q\big)\Big] \\ &= \mathbf{Var}\Big[2\pi\big(N(t) + f_n T_e(t)\big)\Big] + \mathbf{Var}\big[2\pi f_n Q\big] \\ &= \mathbf{Var}\big[\Phi(t)\big] + \mathbf{Var}\big[2\pi f_n Q\big] \\ &= \mathbf{Var}\big[\Phi(t)\big] + 4\pi^2 f_n^2 \frac{d_{LSB}^2}{12},\end{aligned} \tag{3.19}$$

where $\forall t \in \mathbb{R}_{\geq 0} : 2\pi\big(N(\omega, t) + f_n T_e(\omega, t)\big) = \Phi(\omega, t)$ from eq. (3.18) with no measurement error. The term: $4\pi^2 f_n^2 \frac{d_{LSB}^2}{12}$, functions as a quantization noise floor, below which no accurate measurement can be made using a DC with stage delays equal to d_{LSB}.

In [56], although not derived analytically, but obtained via model simulations, the DC stage delay was also shown to have an influence on the measurement precision. The authors of [56] conclude that an average DC stage delay of no more than 18 ps is necessary to have an accurate (less than 10 % relative error) jitter measurement.

3.4 ASIC Measurements

3.4.1 Test Set-up

The test set-up is shown in fig. 3.6. A Xilinx Zynq System-on-Chip (SoC) FPGA controls a 65 nm CMOS ASIC. A scan chain on the ASIC is used to serially read out the sampled DC and counter values. The FPGA drives the signals `RO_ENABLE` and `DC_CLK` and forwards the scan chain data to a Personal Computer (PC) for further analysis. Prior to each measurement, the PC sends a configuration vector to the FPGA, which drives an input scan chain on the ASIC, that configures the inverters and frequency scaler inside the ROs. Unless stated otherwise, for all measurements, the ROs are configured such that the configurable inverters have minimal drive strength and the frequency scaler outputs a frequency divided by two, i.e., the output of the first Toggle Flip-Flop (TFF) in fig. 3.3.

A photograph illustrating the ASIC die along with the measurement set-up is presented in fig. 3.7. A Printed Circuit Board (PCB) is used to distribute the 3.3 V supply voltage to the various voltage levels required by the ASIC. The

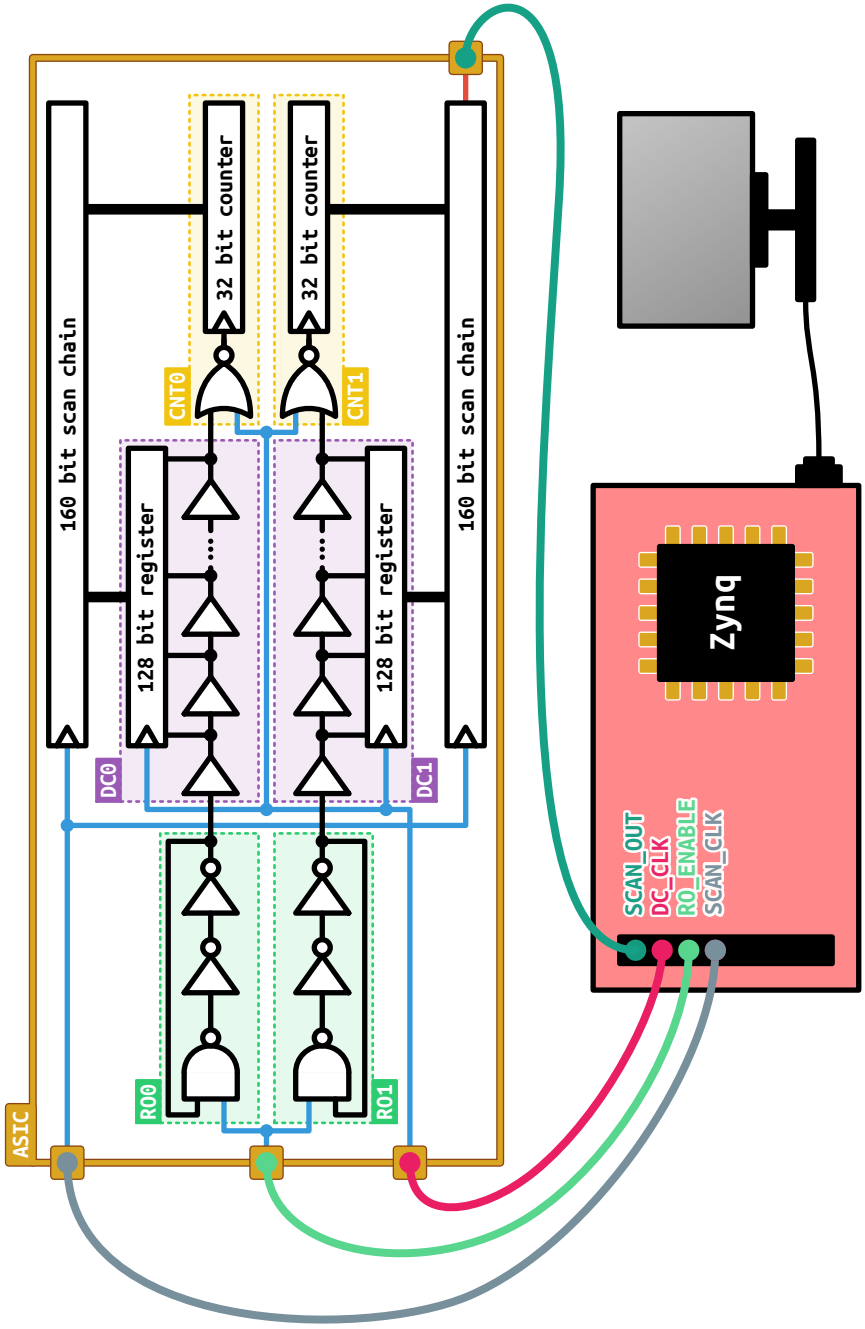

Figure 3.6: Measurement test set-up.

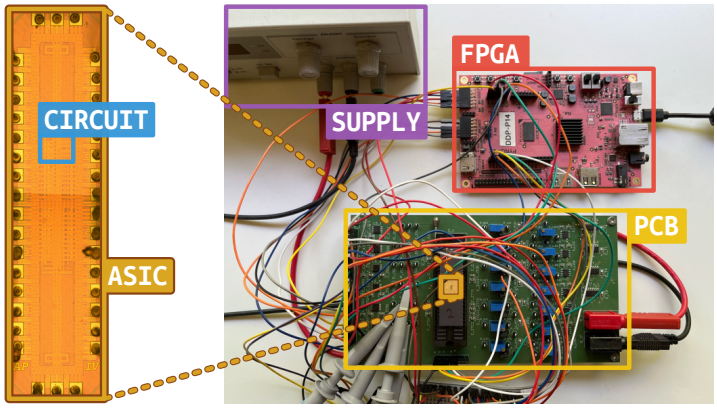

Figure 3.7: Chip photograph and practical measurement test set-up.

measurement circuit on the ASIC die occupies a rectangular area measuring 170 µm by 125 µm.

Test Set-up Induced Noise

The analysis presented in section 3.3.2 indicates that the non-stable timing signals, produced by the test set-up, may influence the calculated phase variance. Especially fluctuations in the rise times of the `RO_ENABLE` and `DC_CLK` signals can result in variations in the accumulation time, T_a, leading to an overestimation of the phase variance as determined by eq. (3.17). To quantify the test set-up induced variations, the accumulation time, difference between the rising edges of `RO_ENABLE` and `DC_CLK`, is measured using an oscilloscope. The assumption is made that the timing reference used by the oscilloscope is independent of the timing reference used by the FPGA. Therefore, all variations present in the accumulation time can be captured. Figure 3.8 shows the measured accumulation time variance versus the set accumulation time. The measured variance increases with set accumulation time and is in the order of $(0.5\,\text{ns})^2$.

DC Characterization

Due to *process variations*, not all delay chain stages will have equal *propagation delay*. Irregularities in the stage propagation delays will give rise to unequal steps in the resolution of the time to digital conversion, often called Differential Non-Linearity (DNL) in the TDC-community. To increase the accuracy of the

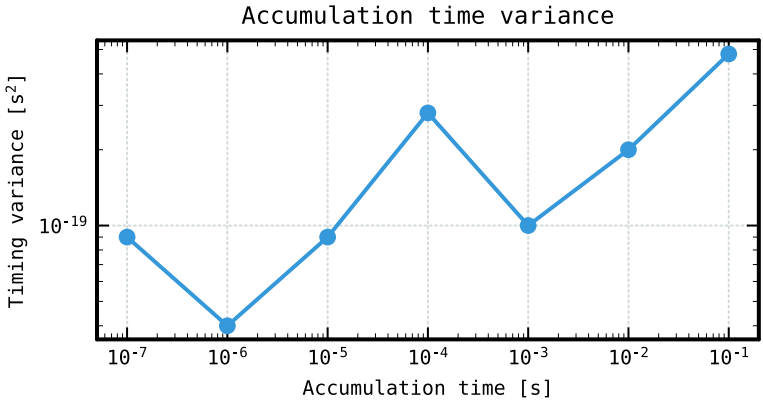

Figure 3.8: Measured accumulation time variance.

time measurements performed, the delay chain stage propagation delays are characterized using a *Monte Carlo* method, a similar approach as in [94].

Both ROs are enabled by raising `RO_ENABLE`. The DCs are periodically sampled, by pulsing `DC_CLK`, at a low sampling rate: 10 Hz or less. The sampling clock is independent of the ROs on the ASIC. In this way, it is assumed the ROs are sampled at a random phase. In case all DC stages would have equal delay, one expects the position of a captured edge to be uniformly distributed over the DC. However, in practice, stages with a large propagation delay will have a larger *probability* of propagating an edge at the moment of sampling. Using this relation, stage delay variations can be obtained by observing the distribution of captured edge positions over multiple samples. Knowing the absolute period length of the RO, allows for determining the absolute DC stage propagation delays, both for rising as for falling edges. Box plots, visualizing the measured stage delay distributions, are shown in fig. 3.9 for three different chips tested. To estimate these distributions, 1000 Monte Carlo samples are collected for each chip.

DC Quantization Noise

From section 3.3.3, the quantization noise floor depends on the resolution of the DCs. An upper bound for the quantization noise floor is found by substituting the largest measured DC stage propagation delay, $d_{LSB}^{\max}$, in eq. (3.19). Table 3.3 gives the largest measured DC stage propagation delay and corresponding time domain quantization noise floor magnitudes for three chips tested.

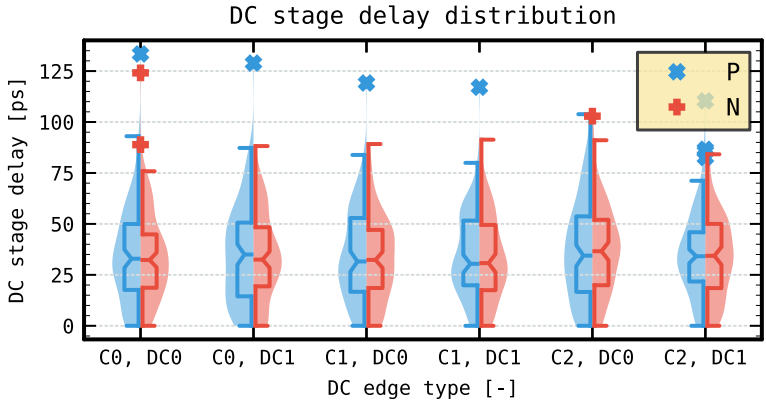

Figure 3.9: Measured stage delay distributions, using 1000 Monte Carlo samples. The x-axis labels indicate the chip number (`C0`, `C1` or `C2`) and the DC (`DC0` or `DC1`). Delay distributions are given for rising and falling edge types: `P` and `N`, respectively.

Note. The $d_{LSB}^{\max}$ values shown in table 3.3 are notably larger than the 18 ps, as required by [56]. However, the *assumptions* made by [56] are rather conservative. The anticipated linear noise strength is small (2 fs, using assumption *c)* in [56], contrasted to 20 fs as obtained in section 3.4.2, for an oscillation frequency of 500 MHz). Furthermore, requiring the quantization error to be ten times smaller than the expected period variations represents a significant margin of safety. Particularly since the measured period variation will only approach the quantization noise floor at the lower end of the tested accumulation time range. With more realistic values for the linear noise strength (20 fs) and safety margin (two times smaller), the average DC resolution should be below 42 ps for an accumulation time of 30 ns. The largest average DC resolution observed across all three devices was 37 ps, obtained for chip 2.

Quantization errors are assumed independent between both DCs, therefore the total phase difference quantization noise floor is found by summing up the contribution of each DC:

$$\mathbf{Var}\left[\hat{\Phi}_0(T_a) - \hat{\Phi}_1(T_a)\right] = \mathbf{Var}\left[\Phi_0(T_a) - \Phi_1(T_a)\right] + 4\pi^2 f_{n0}^2 \mathbf{Var}[Q_0] + 4\pi^2 f_{n1}^2 \mathbf{Var}[Q_1],$$

with Q_0 and Q_1 the quantization error contributions of `DC0` and `DC1` respectively. Table 3.4 shows the measured RO frequencies and the contributions to the total phase quantization noise floor for each DC individually.

Table 3.3: DC resolution.

Chip	DC0		DC1	
	$d_{LSB}^{\max}$ [ps]	$\mathbf{Var}[Q_0]$ [s^2]	$d_{LSB}^{\max}$ [ps]	$\mathbf{Var}[Q_1]$ [s^2]
0	130	1.40×10^{-21}	96	7.67×10^{-22}
1	98	7.94×10^{-22}	99	8.20×10^{-22}
2	115	1.10×10^{-21}	109	9.97×10^{-22}

Table 3.4: Measured frequency and quantization noise floor.

Chip	DC0		DC1	
	f_{n0} [MHz]	$4\pi^2 f_{n0}^2 \mathbf{Var}[Q_0]$ [rad^2]	f_{n1} [MHz]	$4\pi^2 f_{n1}^2 \mathbf{Var}[Q_1]$ [rad^2]
0	508	1.43×10^{-2}	536	8.71×10^{-3}
1	521	8.50×10^{-3}	547	9.68×10^{-3}
2	556	1.34×10^{-2}	545	1.17×10^{-2}

3.4.2 Phase Difference Measurements

Dominant Noise Type Estimation

The measurement results from section 3.4.1 show that $\sigma_{T_a}^2 \ll \mu_{T_a}^2$, for μ_{T_a} in the range from 100 ns to 0.1 s. This knowledge allows to further simplify eq. (3.17):

$$\begin{aligned}\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = 4\pi^2 (f_{n0} - f_{n1})^2 \sigma_{T_a}^2 \\ + \big(f_{n0}^2 + f_{n1}^2\big)\big(c_0 + c_1 \mu_{T_a} + c_{20} \mu_{T_a}^2 \\ + c_{21} \ln(\mu_{T_a}) \mu_{T_a}^2 + c_3 \mu_{T_a}^3\big).\end{aligned} \tag{3.20}$$

The variance of the phase difference was estimated by measuring both RO phases 100 times at accumulation time lengths ranging from 30 ns to 0.3 s. For each measurement, both RO phases are subtracted to obtain the measured phase difference. The phase difference variance was estimated for each accumulation time length, using the sample variance:

$$\big(\hat{\sigma}_{\Delta\Phi}^a\big)^2 = \sum_{i=0}^{99} \frac{\big(\hat{\phi}_0^{i,a} - \hat{\phi}_1^{i,a} - \hat{\mu}_{\Delta\Phi}^a\big)^2}{100},$$

with $\hat{\mu}_{\Delta\Phi}^a = \sum_{i=0}^{99} \frac{\hat{\phi}_0^{i,a} - \hat{\phi}_1^{i,a}}{100}$ the sample mean at accumulation time length measurement a, and $\hat{\phi}_0^{i,a}$ and $\hat{\phi}_1^{i,a}$, the measured phase for measurement i

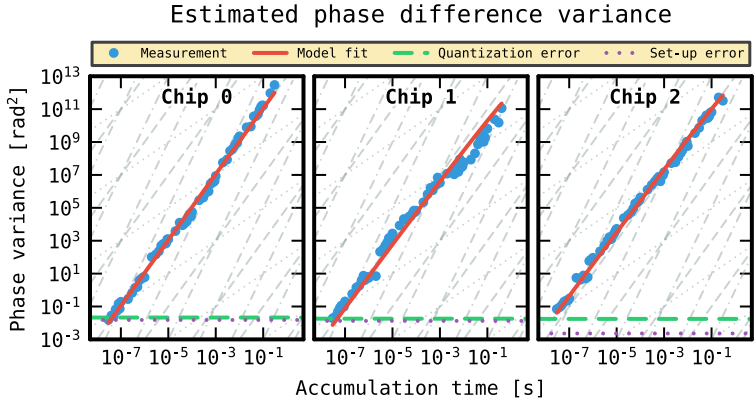

Figure 3.10: Estimated variance of the phase difference.

Table 3.5: Test set-up noise floor.

Chip	$4\pi^2(f_{n0} - f_{n1})^2\sigma_{T_a}^2$ [rad^2]
0	1.52×10^{-2}
1	1.33×10^{-2}
2	2.30×10^{-3}

and accumulation time length measurement a, for `R00` and `R01` respectively. Figure 3.10 shows the measurement results for three chips tested. Two effects limit the measurement precision:

- *Quantization noise floor*: obtained by summing the quantization noise contributions of both DCs, given in table 3.4.
- *Test set-up noise floor*: obtained by evaluating the term: $4\pi^2(f_{n0} - f_{n1})^2\sigma_{T_a}^2$ in eq. (3.20) for the largest timing variance, $\sigma_{T_a}^2$, observed in the measurement described in section 3.4.1: 4.8×10^{-19} s^2. Table 3.5 provides the calculated set-up noise floor for each chip tested.

From fig. 3.10, it is clear that almost all measurements fall above both noise floors and the results should therefore be contributed to the phase difference variance. Equation (3.20) can then be further simplified, as the test set-up induced variance is small compared to the measured variance:

$$\begin{aligned}\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = \big(f_{n0}^2 + f_{n1}^2\big)\big(c_0 + c_1\mu_{T_a} + c_{20}\mu_{T_a}^2 \\ + c_{21}\ln(\mu_{T_a})\mu_{T_a}^2 + c_3\mu_{T_a}^3\big).\end{aligned} \tag{3.21}$$

Table 3.6: Phase variance model fit.

Chip	c_{20} [rad^2 s^{-2}]	c_{21} [rad^2 s^{-2}]
0	2.04×10^{-5}	-5.63×10^{-13}
1	1.73×10^{-6}	-7.48×10^{-7}
2	7.94×10^{-6}	-3.93×10^{-6}

The gray dashed lines in fig. 3.10 show polynomials of first, second and third degree. The estimated variance data show the greatest correspondence to a second degree polynomial. This enables to rule out the terms corresponding with a first and third degree polynomial in eq. (3.21), as a change of slope should be observed in fig. 3.10 when these terms would become dominant. Equation (3.21) can be further reduced to only include the second degree terms:

$$\mathbf{Var}\big[\Phi_0(T_a) - \Phi_1(T_a)\big] = \big(f_{n0}^2 + f_{n1}^2\big)\big(c_{20}\mu_{T_a}^2 + c_{21}\ln(\mu_{T_a})\mu_{T_a}^2\big).$$

For the simplification used in section 3.2.3 to hold, the maximal accumulation time, $t_{\max}$, is bounded by the lower frequency limit: $2\pi f_l t_{\max} \ll 1$. Combining eqs. (3.11) and (3.16), the lower frequency limit, f_l, is related to the coefficients: c_{20} and c_{21} as follows:

$$2\pi f_l = \exp\Big(\frac{c_{20}}{c_{21}} + \frac{3}{2} - \gamma\Big). \tag{3.22}$$

The bound on the lower frequency limit implies a bound on the coefficients c_{20} and c_{21}:

$$\exp\Big(\frac{c_{20}}{c_{21}}\Big) \ll \frac{\exp\big(\gamma - \frac{3}{2}\big)}{t_{\max}} = 1.32, \tag{3.23}$$

for a maximal accumulation time equal to 0.3 s. The coefficients: c_{20} and c_{21} are estimated by minimizing the fitting error:

$$\min_{c_{20},c_{21}} \sum_{a=0}^{a_{\max}-1} \left(\ln\Big(\big(\hat{\sigma}_{\Delta\Phi}^a\big)^2\Big) - \ln\Big(\big(f_{n0}^2 + f_{n1}^2\big)\big(c_{20}\mu_{T_a}^2 + c_{21}\ln(\mu_{T_a})\mu_{T_a}^2\big)\Big)\right)^2,$$

subject to the bound in eq. (3.23) and with μ_{T_a}, the set accumulation time length for accumulation time measurement a, and $a_{\max}$, the number of accumulation time length measurements performed. The optimal fit is plotted as a solid line in fig. 3.10. The optimal values for c_{20} and c_{21} are provided in table 3.6.

Flicker Noise Corner

As shown in fig. 3.3, the measurement architecture uses both DCs and counters to resolve the RO phase. The DCs are used as a fine-grained TDC, to measure the

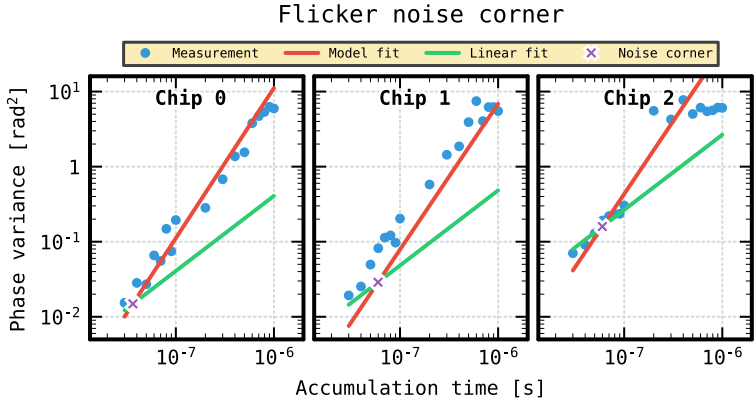

Figure 3.11: Flicker noise corner.

RO phase within one RO period, while the counters are used as a course-grained TDC, to keep track of the RO phase as a multiple of 2π, i.e., one RO period. For small set accumulation time lengths, μ_{T_a}, the variations measured are dominated by the DCs, as the phase variations are within the same RO period and the counters will show almost no variations. For large set accumulation time lengths, the counter variations will dominate the measured variations, as the DC variations are limited to a few RO periods.

Figure 3.11 zooms in on the area where the measured variance is dominated by the DCs: accumulation time in the interval 30 ns to 1 µs. The model fit from section 3.4.2 is shown as well. For the data in this accumulation time length range, a second fit is made, similar to the model fit in section 3.4.2, but also allowing a linear component: $c_1\mu_{T_a}$. This linear component, together with the intersection point between the linear component and the quadratic model fit are shown as well in fig. 3.11. Below the intersection point, the linear component dominates the phase variance. Table 3.7 provides derived values for the flicker noise corner time and the linear fitted noise strength, s_{noise}:

$$\mathbf{Var}\big[T_{per}\big] \approx s_{noise}\mathbf{E}\big[T_{per}\big] = \frac{c_1}{4\pi^2}\mathbf{E}\big[T_{per}\big], \tag{3.24}$$

with T_{per}, a random variable representing the period length of the RO in case of only white FM noise.

For accumulation time lengths above the flicker noise corner, flicker FM noise will dominate over white FM noise. Meaning that for accumulation time lengths larger than this corner, accumulated RO jitter cannot be regarded as independent anymore. This has an important consequence for ES designs, as many ES stochastic models assume only the existence of white FM noise, e.g.,

Table 3.7: Flicker noise corner and linear noise strength.

Chip	Flicker noise corner [ns]	s_{noise} [fs]
0	36.5	18.9
1	59.7	21.4
2	59.9	111

in [4, 6]. Especially for ES designs that require long accumulation lengths, this assumption can become problematic. In [27], the authors came to the same conclusion.

A concrete example highlights the consequence of wrongly assuming independent jitter on the entropy estimation of an Elementary Ring Oscillator (ERO)-ES [6]. Assume a technology with a linear noise strength (s_{noise}) and a flicker noise corner equal to 20 fs and 50 ns, respectively. Measuring the accumulated period variance at an accumulation time of 1 µs, yields an overestimated linear noise strength of 400 fs. Assuming ES throughput of 1 MHz and an oscillator frequency of 1 GHz, and using the overestimated noise strength results in a significant overestimation of the produced min-entropy per output bit, yielding 0.9993 bit, as opposed to the more accurate estimation of 0.1144 bit.

Long Term Dependency and Use of Allan Variance

The trapping and detrapping action of inversion carriers in the Metal-Oxide-Semiconductor (MOS) channel is believed to be the main source of flicker noise [35]. Each trapping site has the potential of modulating the Metal-Oxide-Semiconductor Field-Effect Transistor (MOSFET) channel conductance, this effect is often modeled by a *random telegraph* random process. The superposition of the random telegraph processes from multiple trapping sites produces the $1/|f|$ dependency of the PSD of output referred current noise sources in MOSFETs [41]. This noise current is injected at the internal RO nodes and therefore gets integrated to generate an $1/|f|^3$ component in the RO excess phase PSD: S_{Φ_e}, for $\alpha = -1$ in eq. (3.4). The theory explaining how low frequency noise sources manifest as integrated in the excess phase spectrum was developed in [28]. To alleviate the impact of device flicker noise, [28] suggests enhancing the symmetry of voltage waveforms at the oscillator nodes. This can be accomplished by appropriately scaling the oscillator transistors.

Depending on the time constant (release time) of the trapping site, its influence on the MOSFET drive current might stretch for a long time interval [18], hence the presence of low frequencies in flicker noise. Theoretically, its influence

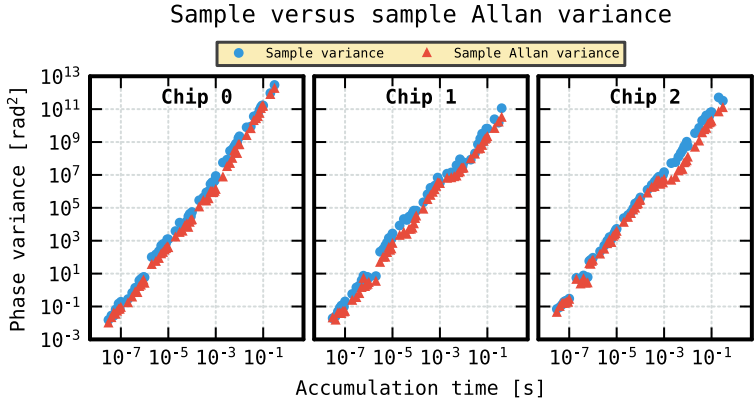

Figure 3.12: Sample variance versus sample Allan variance.

might even span multiple consecutive phase measurements of a single RO, despite the RO being disabled in between two measurements, by driving the `RO_ENABLE` signal low. The trapping, detrapping actions might therefore induce mutual dependencies between the separate phase measurements performed in section 3.4.2. The model proposed in this chapter, however, assumes no *state information* is transferred between two consecutive phase measurements.

In previous work [27, 59], the *Allan variance* [3] is used to reduce the measured influence of long term dependencies in the oscillator excess phase. Instead of looking at the sample variance of a data set containing measured oscillator phase differences: $\{\hat{\phi}_0^{0,a} - \hat{\phi}_1^{0,a}, \hat{\phi}_0^{1,a} - \hat{\phi}_1^{1,a}, \ldots, \hat{\phi}_0^{99,a} - \hat{\phi}_1^{99,a}\}$, the sample Allan variance is calculated as half the sample variance of a data set, containing the difference between two consecutive phase difference measurements: $\{\hat{\phi}_0^{0,a} - \hat{\phi}_0^{1,a} - \hat{\phi}_1^{0,a} + \hat{\phi}_1^{1,a}, \hat{\phi}_0^{1,a} - \hat{\phi}_0^{2,a} - \hat{\phi}_1^{1,a} + \hat{\phi}_1^{2,a}, \ldots, \hat{\phi}_0^{98,a} - \hat{\phi}_0^{99,a} - \hat{\phi}_1^{98,a} + \hat{\phi}_1^{99,a}\}$. *Flicker* noise events that affect consecutive phase difference measurements will hence be filtered out by using the sample Allan variance.

Figure 3.12 compares the standard sample variance, as it was used in section 3.4.2 with the sample Allan variance on the measured phase difference data. Although the sample Allan variance is consistently smaller than the standard sample variance, the difference between the two sample variances is minor, indicating that the impact of long term dependencies on the phase measurements is of a similar magnitude to the measured flicker noise component within a measurement interval.

Table 3.8: Noise type magnitudes.

Chip	$h_{-2} \ll [\mathrm{s}^{-1}]$	h_{-1} [-]	h_0 [s]	$h_1 \ll [\mathrm{s}^2]$	$h_2 \ll [\mathrm{s}^3]$
0	1.31×10^{-7}	7.14×10^{-15}	1.89×10^{-14}	5.71×10^{-22}	2.76×10^{-28}
1	1.69×10^{-8}	9.48×10^{-9}	2.14×10^{-14}	4.10×10^{-22}	1.98×10^{-28}
2	8.13×10^{-8}	4.97×10^{-8}	1.11×10^{-13}	2.12×10^{-21}	1.03×10^{-27}

Noise Type Contribution Summary

Table 3.8 summarizes the estimates for the coefficients used in eq. (3.4), describing the relative frequency deviation PSD for an oscillator modeled by a power law. For the coefficients: h_{-2}, h_1 and h_2, only upper bounds are estimated as these noise types are not observed in the measurements. The values used for the high and low frequency limits are determined from the accumulation measurement range: $f_l = 1/(0.3\,\mathrm{s}) = 3.33\,\mathrm{Hz}$ and $f_h = 1/(30\,\mathrm{ns}) = 33.3\,\mathrm{MHz}$. In reality f_l, f_h is much smaller, larger respectively, however the values given in table 3.8 remain valid upper bounds.

In contrast to the value obtained for h_0, which is directly related to the available noise strength via $c_1 = 4\pi^2 h_0$ and eq. (3.24), the reported values for h_{-1} should be considered as rough estimates, without a direct physical implication. The magnitude of the derived lower frequency limit, from eq. (3.22), heavily determines the final estimate for h_{-1}. The first row in table 3.8 reflects this, showing a low estimate for h_{-1} compared to the other chips tested, due to a reduced lower frequency limit obtained from eq. (3.22).

3.5 Oscillator Noise in Other Work

3.5.1 Stationarity Misconception

An assumption, commonly found in literature [16, 20, 27, 29, 31], when relating frequency domain concepts (e.g., PSD) to time domain concepts (e.g., period length variation), is WSS of the signals under study (assuming *ergodicity* implies assuming stationarity). This assumption allows for the use of the Wiener-Khinchin theorem to relate the PSD to the ACF. As shown by the analysis in section 3.2, care has to be taken to ascertain this assumption remains valid when subjected to the relevant noise types.

In [16, 27, 29], the excess phase was wrongly assumed WSS under the action of white FM noise. As widely acknowledged in literature and shown once more

in section 3.2.2 in this chapter, the excess phase exhibits a time dependent variance function, $\mathbf{Var}\left[\Phi_e(t)\right]$, and therefore cannot be WSS.

To clarify, for the oscillators under study, three related processes exist:

- The excess phase ($\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$): only WSS for white and flicker PM noise.
- The relative frequency deviation ($\{Y(t)\}_{t\in\mathbb{R}_{\geq 0}}$): only WSS for flicker FM (bandlimited), white FM, white and flicker PM noise.
- The oscillator output voltage: when starting from a non-uniform phase distribution (e.g., a fixed initial phase), the output voltage only becomes stationary in the limit when a sufficient amount of time has passed to allow the excess phase to diffuse to approximately a uniform distribution. This concept was demonstrated by [20], where the influence of the initial phase was shown to follow an exponential decay. Phase diffusion occurs with random walk, flicker and white FM noise.

Note. Given the oscillator noise model and starting from a non-uniform initial phase distribution, the output voltage cannot be stationary when the excess phase is stationary and vice versa.

3.5.2 Jitter Measurements in Other Work, Comparison

Previous studies have characterized the oscillator phase noise in the context of random number generation. Table 3.9 provides a comparison of the linear (thermal or white) noise strength estimates found in other research.

In [6], a differential jitter measurement set-up is proposed consisting of two free-running ROs. Both RO outputs are brought off-chip and the toggle time instances are measured using an oscilloscope. This approach, likely leads to an overestimation of the available jitter strength, as external noise influences induced by the output chain (e.g., output drivers, voltage level shifters, off-chip wires, etc.) cannot be ruled out. The reported linear jitter strength value by [6], was the largest value found in recent literature, indicating an overestimation. Only thermal noise was considered, however, fig. 3 in [6] shows a slight deviation from linear jitter accumulation, indicating the presence of other noise types.

In [59], the Allan variance is used to characterize oscillator frequency stability in an FPGA platform, using counter values. As counters are used to measure the oscillator phase, the resolution of the phase measurement is limited, reflected in the high quantization noise floor visible in figs. 7 and 8 in [59]. Nevertheless, the same quadratic dependency on accumulation time length was observed

in the counter variance, indicating the flicker FM noise is dominating for the accumulation times of interest.

In [27], the excess phase, $\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$, was wrongly assumed to be WSS in the presence of thermal- and flicker noise. Despite this wrong assumption, the authors of [27] identified the existence of a noise corner, under which the thermal noise component dominates and accumulated jitter samples can be regarded as independent. This noise corner, calculated from the data in [27], equals 104 µs. The measurement results, shown here in fig. 3.10, do not show a noise corner at this accumulation time length and the flicker noise component remains dominant well below 104 µs. Apart from the fact that a different technology was used in [27]: FPGA versus ASIC in this chapter, this discrepancy in measurement results could also be induced by the limited range of accumulation time lengths tested. From fig. 7 in [27], the obtained noise corner, $N = 5354$, is near the edge of the measurement interval. The obtained thermal noise is therefore the result of an extrapolation of the measurement results, as all measurements were performed in the region where the flicker noise component dominates. It has to be noted, that the same argument holds for the measurement presented in this chapter. However, the smallest accumulation time length here, 30 ns, is considerably smaller than in [27].

In both [21] and [46], the coherent sampling technique is used on an FPGA platform, to measure the linear jitter strength for accumulation time lengths small enough, such that the flicker noise is not dominant. Measurement results are presented for accumulation time lengths down to 2.0 µs ($M = 250$, assuming $T_2 \approx T_1$ in [21]) and 3.5 µs respectively. Although a flicker noise corner at approximately 3.5 µs ($M = 450$ in [21]) and 5 µs are recognized by the authors, they did not conduct a formal analysis to ascertain its precise value.

In [84], a basic counter method was utilized for on-chip linear jitter strength measurement. The precision of the time measurement was constrained by the duration of the oscillator period, which was 10.2 ns. Only thermal noise was considered, and a single accumulation time of 333 µs was employed.

In [94], DCs are used to resolve the timing jitter, which is similar to the approach used in this chapter. In addition, the timing resolution is increased using carry logic that is available in certain FPGAs. The accumulation time lengths that were tested range from 10 ns to 80 ns. Only thermal noise was considered, however, fig. 6 in [94] suggests a quadratic behavior above accumulation time lengths of 60 ns, as the measured variance deviates from the linear approximation.

In [66], a fast oscillator, together with a counter are used to resolve the differential timing jitter between two DCs. The accumulation time lengths that were tested

Table 3.9: Linear jitter strength in other work comparison.

Work	Device	s_{noise} [fs]	resolution [ns]	Measurement range From [µs]	Measurement range To [ms]
This work	**65 nm CMOS ASIC**	**18.9**	**0.109**	**0.03**	**300**
[6]	Altera (Intel) Stratix II FPGA	281.4	-[a]	29	0.058
[27]	Altera (Intel) Cyclone III FPGA	26.0	9.7	117[b]	1.94[b]
[66]	28 nm CMOS ASIC	30	0.164	0.016	1.31×10^{-4}
[94]	Xilinx (AMD) Spartan 6 FPGA	21.98	0.057[c]	0.01	8×10^{-5}
[94]	Altera (Intel) Cyclone IV FPGA	0.89	-[d]	0.04	4×10^{-5}
[21]	Altera (Intel) Cyclone III FPGA	3.12	-[e]	1.95	9.37×10^{-3}
[46]	Altera (Intel) Cyclone III FPGA	78.68	2.62	3.48	0.019
[84]	Actel (Microchip) Fusion FPGA	107	10.2	333	0.333

[a] Resolution of the oscilloscope was not reported.
[b] Values calculated from fig. 7 in [27].
[c] Value based on the maximal delay from fig. 5 in [94].
[d] Resolution of the DC was not reported.
[e] Period length, T_2, was not reported.

ranged from 20 ns to 160 ns. Only thermal noise was considered, however, as identified by [18]: fig. 12 in [66] suggests a super linear behavior at the higher end of the tested accumulation interval, as the measured variance deviates from the linear approximation.

3.6 Conclusion

This study presents a comprehensive review of the prevalent noise types in free-running oscillators, which are characterized using a power law. In this chapter, the analytical variance of the excess phase for each noise type is derived, and these findings are employed to develop a model for measuring differential oscillator phase variance. The model incorporates the variances introduced by the quantization action and the test set-up.

A 65 nm CMOS technology was used to implement the phase measurement set-up, and three different devices were measured out. Using DCs in an ASIC technology allows for a fine time measurement resolution below 100 ps, reducing the quantization noise floor. The measurement results from all three devices showed that the oscillator phase noise was primarily dominated by flicker noise for most accumulation time lengths of interest. For accumulation time lengths less than 100 ns, a noise corner was observed, suggesting an upper limit on the jitter accumulation time used by ES designs that assume independent jitter accumulation.

Chapter 4

Modelling Phase Noise

This chapter is based on the following publication:

> **TRNG Entropy Model in the Presence of Flicker FM Noise**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, 2024

Contribution: *main author*.

4.1 Background and Context

As thoroughly discussed in sections 2.2.2 and 2.2.3, international standards strongly recommend including a *stochastic model* capable of estimating the ES *entropy* production. Despite the abundance of ES stochastic models, many relying on the presence of only *white* FM noise as seen in [6, 26, 66], the significance of *flicker* FM noise was often disregarded or deemed less important. This is typically justified by the rationale that accounting solely for the entropy provided by the white FM *noise component* is adequate for establishing a lower bound on the entropy produced by the ES [47]. Any entropy generated by other *independent* noise components was regarded as a surplus that would not invalidate the entropy bound previously declared.

Increased attention for the flicker FM noise component [27, 46, 67] underscores the need to incorporate this *type of noise* into ES models. Particularly, recent research [8] has indicated that flicker FM noise can make a meaningful

contribution to the ES's entropy production, thus emphasizing the importance of accurately describing it. Although [8] proposes a method for generating period length samples for an RO under the influence of white FM and flicker FM noise, it lacks a rigorous mathematical analysis to derive the ES output *entropy density*. This absence is particularly unfortunate given the intriguing claim that an increased flicker FM magnitude tends to reduce the observed correlation of the generated output bits. Instead, the study relies on simulation results fitted to empirical data. Moreover, it *assumes* that the absence of linear correlation in the output bits proves independence, which is crucial for the validity of the reported entropy results.

Precisely modeling flicker FM noise is challenging due to its inherent long-term dependencies, which arise from the physical nature of the charge carrier trapping and detrapping process in the transistors that comprise the oscillator [22]. A trapped charge carrier can affect the transistor's drive strength over multiple oscillation periods, leading to a sustained increase or decrease in the oscillating frequency. A notable effort was made by [67], chapter 3 in this thesis, where a time-based analysis revealed the dependency of an oscillator's excess phase variance on the *accumulation time length*. However, the work presented in chapter 3 lacks a comprehensive description of the oscillator phase and a method for characterizing phase dependencies. Additionally, it does not provide a method for estimating the entropy induced by flicker FM noise. Instead, chapter 3 focused on quantifying the magnitudes of the prevalent noise types in free-running oscillators.

Apart from [67], there are only a limited number of flicker FM magnitude estimates available: [8, 21, 27, 46], which span multiple orders of magnitude. As demonstrated in this chapter, the specific magnitude employed, notably impacts the resulting entropy estimate, thereby dictating the extent to which the contribution of flicker FM noise outweighs that of white FM noise in the total ES *entropy rate*.

The primary contributions presented in this chapter are as follows:

- An *analytical* derivation of the oscillator excess phase ACF, influenced by flicker FM-shaped *noise sources*, is presented as a generalization of the phase variance derivation in chapter 3.
- This chapter proposes a simulation method for the phase process, similar to the approach presented by [8]. However, our method offers a more robust mathematical foundation. We achieve this by developing a unified Gaussian process model that incorporates the ACF for both white FM and flicker FM noise.

- Simulation results using the constructed model compare the entropy produced in an ERO-ES under the influence of white FM and flicker FM noise, *conditioned* on the oscillator's phase or previously produced bit values.

Section 4.2 presents the excess phase Gaussian process and how this process is shaped by its ACF. The ACFs for white FM and flicker FM noise are used to formulate an *entropy model* for an ERO-ES in section 4.3. Simulation results presenting entropy values, obtained by applying the proposed model to typical *noise magnitudes* found in literature are discussed in section 4.4. Finally, section 4.5 summarizes this chapter and suggests avenues for future research.

For an introduction to, and detailed descriptions of the mathematical notions and concepts utilized throughout this chapter, readers are referred to appendix A.

4.2 Excess Phase Process

As in section 3.2, the phase of a noisy oscillator, starting at time $t = 0$, is modeled as a real-valued *random process* over continuous time $\Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$:

$$\Phi(\omega, t) = 2\pi f_n t + \phi_0 + \Phi_e(\omega, t),$$

with f_n, the nominal oscillator frequency, ϕ_0, the initial phase at time $t = 0$ and $\{\Phi_e(t)\}_{t \in \mathbb{R}_{\geq 0}}$, a real-valued random process describing the oscillator's excess phase. The excess phase is assumed to be unbiased (equal *probability* to become positive as to become negative in value) through time. Therefore: $\forall t \in \mathbb{R}_{\geq 0} : \mathbf{E}\big[\Phi_e(t)\big] = 0$.

Instead of the phase itself, properties of the oscillator are often described using the instantaneous relative frequency deviation: $\forall \omega \in \Omega, \forall t \in \mathbb{R}_{\geq 0} : Y(\omega, t) = \frac{\frac{\mathrm{d}}{\mathrm{d}t}\Phi(\omega,t) - 2\pi f_n}{2\pi f_n}$. The phase ACF can be written in terms of the ACF for this relative frequency deviation, by generalizing eq. (3.2) on page 31 in chapter 3,

$\forall (t_i, t_j)^\intercal \in \mathbb{R}^2_{\geq 0}$:

$$\begin{aligned} R_{\Phi_e}(t_i, t_j) &= \mathbf{E}\big[\Phi_e(t_i)\Phi_e(t_j)\big] = \mathbf{E}\left[2\pi f_n \int_0^{t_i} Y(\theta_i)\mathrm{d}\theta_i 2\pi f_n \int_0^{t_j} Y(\theta_j)\mathrm{d}\theta_j\right] \\ &= 4\pi^2 f_n^2 \int_0^{t_i}\int_0^{t_j} \mathbf{E}\big[Y(\theta_i)Y(\theta_j)\big]\mathrm{d}\theta_j\mathrm{d}\theta_i \\ &= 4\pi^2 f_n^2 \int_0^{t_i}\int_0^{t_j} R_Y(\theta_i, \theta_j)\mathrm{d}\theta_j\mathrm{d}\theta_i. \end{aligned} \tag{4.1}$$

4.2.1 Gaussian Process

The excess phase: $\{\Phi_e(t)\}_{t\in\mathbb{R}_{\geq 0}}$ is assumed to behave as a Gaussian process, with zero mean function: $\forall t \in \mathbb{R}_{\geq 0} : \mu(t) = 0$. The ACF fully describes the behavior of the excess phase process. Sampling the excess phase at n time instances: $(t_0, t_1, \ldots, t_{n-1})^\intercal \in \mathbb{R}^n_{\geq 0}$, produces an n-dimensional multivariate normal distributed vector:

$$\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}} = \big(\Phi_e(t_0), \Phi_e(t_1), \ldots, \Phi_e(t_{n-1})\big)^\intercal \sim \mathcal{N}_n\big(\vec{\mathbf{0}}_{\boldsymbol{n}}, \boldsymbol{\Sigma_e}\big),$$

with mean vector: $\vec{\mathbf{0}}_{\boldsymbol{n}}$, an $n \times 1$ all-zero vector and covariance matrix given by

$$\boldsymbol{\Sigma_e} = \begin{bmatrix} R_{\Phi_e}(t_0,t_0) & R_{\Phi_e}(t_0,t_1) & \ldots & R_{\Phi_e}(t_0,t_{n-1}) \\ R_{\Phi_e}(t_1,t_0) & R_{\Phi_e}(t_1,t_1) & \ldots & R_{\Phi_e}(t_1,t_{n-1}) \\ \vdots & \vdots & \ddots & \vdots \\ R_{\Phi_e}(t_{n-1},t_0) & R_{\Phi_e}(t_{n-1},t_1) & \ldots & R_{\Phi_e}(t_{n-1},t_{n-1}) \end{bmatrix}. \tag{4.2}$$

4.2.2 Source of Noise

The PSD for the relative frequency deviation, denoted by $S_Y(f)$, is assumed to be composed of a sum of *noise contributions* of different type α [32]:

$$S_Y(f) = \sum_{\alpha=-2}^{2} h_\alpha |f|^\alpha = \sum_{\alpha=-2}^{2} S_{Y^\alpha}(f). \tag{4.3}$$

Following the reasoning in section 3.2.1, the noise magnitudes, h_α, represent the magnitudes of the five most prevalent noise types in oscillators: *random walk* FM ($\alpha = -2$), flicker FM ($\alpha = -1$), white FM ($\alpha = 0$), flicker PM

($\alpha = 1$), and white PM ($\alpha = 2$). The terms: *white* and *flicker* refer to the shape of the oscillator's frequency or phase spectrum, hence the adjectives: *Frequency Modulated (FM)* and *Phase Modulated (PM)* are used in this text. Additionally, the contributions are assumed mutually independent. Using the derivation outlined in appendix 4.A, the excess phase random process is shown to consist of a sum of individual excess phase noise contributions, $\left\{\Phi_e^\alpha(t)\right\}_{t\in\mathbb{R}_{\geq 0}}$: $\forall\omega\in\Omega, \forall t\in\mathbb{R}_{\geq 0} : \Phi_e(\omega,t) = \sum_{\alpha=-2}^{2}\Phi_e^\alpha(\omega,t)$. Each of these contributions has a PSD following a *power law*: $S_{\Phi_e^\alpha}(f) = \left(\frac{f_n}{f}\right)^2 h_\alpha |f|^\alpha$.

As the experiments in section 3.4 have shown, the noise types of interest when studying an oscillator for a reasonable time frame are white FM ($\alpha = 0$) and flicker FM ($\alpha = -1$) noise. In this chapter, these two noise types are exclusively studied and indicated by the indices: $\cdot^w$ and $\cdot^f$ for white FM and flicker FM components respectively. The constants: $h_w = h_0$ and $h_f = h_{-1}$ are used to indicate the noise magnitudes.

4.2.3 Excess Phase ACF

White FM Noise

As assumed in section 3.2.2, the relative frequency deviation, $\left\{Y(t)\right\}_{t\in\mathbb{R}_{\geq 0}}$, is a WSS process. Its ACF is therefore only dependent on the time shift and a simplified notation is used: $\forall(t_i,t_j)^\intercal\in\mathbb{R}^2_{\geq 0} : R_Y(t_i,t_j) = R_Y(t_j-t_i) = R_Y(\tau)$. Derived in section 3.2.2, under the influence of only white FM noise, the relative frequency deviation ACF equals a scaled Dirac delta distribution function: $R_Y(\tau) = h_0\delta(\tau)$. Using eq. (4.1), the excess phase ACF becomes

$$\begin{aligned}\forall\,(t_i,t_j)^\intercal\in\mathbb{R}^2_{\geq 0} : R_{\phi_e}(t_i,t_j) &= 4\pi^2 f_n^2 h_w \int_0^{t_i}\int_0^{t_j}\delta(\theta_j-\theta_i)\mathrm{d}\theta_j\mathrm{d}\theta_i \\ &= 4\pi^2 f_n^2 h_w \min(t_i,t_j).\end{aligned} \tag{4.4}$$

Flicker FM Noise

Generalizing the derivation made in section 3.2.3, the excess phase ACF can be obtained. The same assumption is made here: a band-limited version of the relative frequency deviation, $\left\{Y(t)\right\}_{t\in\mathbb{R}_{\geq 0}}$, to the frequency interval $[f_l, f_h]$, under the action of flicker FM noise, is WSS. From eq. (3.7) on page 33 in

chapter 3, the relative frequency deviation ACF equals

$$R_Y(\tau) = 2h_f \int_{f_l}^{f_h} \frac{\cos(2\pi f\tau)}{f} \mathrm{d}f.$$

Using eq. (4.1), the excess phase ACF can be obtained, $\forall (t_i, t_j)^\intercal \in \mathbb{R}^2_{\geq 0}$:

$$R_{\Phi_e}(t_i, t_j) = 4\pi^2 f_n^2 \int_0^{t_i} \int_0^{t_j} 2h_f \int_{f_l}^{f_h} \frac{\cos\big(2\pi f(\theta_j - \theta_i)\big)}{f} \mathrm{d}f \mathrm{d}\theta_j \mathrm{d}\theta_i. \tag{4.5}$$

The details of working out the integral and assuming $f_h \to \infty$ are shown in appendix 4.B. The following could be obtained, $\forall (t_i, t_j)^\intercal \in \mathbb{R}^2_{>0}$:

$$R_{\Phi_e}(t_i, t_j) = 4\pi^2 f_n^2 h_f t_i t_j \Bigg(3 - 2\gamma - 2\ln\big(2\pi f_l |t_j - t_i|\big) + \frac{t_i}{t_j} \ln\left(\frac{|t_j - t_i|}{t_i}\right) + \frac{t_j}{t_i} \ln\left(\frac{|t_j - t_i|}{t_j}\right) \Bigg). \tag{4.6}$$

The ACF becomes zero whenever $t_i = 0$ or $t_j = 0$. When $t_i = t_j = t$, the $\ln|t_j - t_i|$ terms cancel out and eq. (4.6) reduces to eq. (3.11) on page 34 in chapter 3.

Example 4.1. Take the following realistic values: $f_n = 520\,\mathrm{MHz}$, $h_w = 18.9\,\mathrm{fs}$, $h_f = 1 \times 10^{-10}$ and $f_l = 1\,\mathrm{mHz}$. Figure 4.1 illustrates an example white FM and flicker FM noise Gaussian process: $\big\{\Phi_e^w(t)\big\}_{t\in\mathbb{R}_{\geq 0}}$ (blue) and $\big\{\Phi_e^f(t)\big\}_{t\in\mathbb{R}_{\geq 0}}$ (red). The markers show four realizations: $\varphi_e^y(t_i)$ for $t_i \in \{0\,\mu\mathrm{s}, 1\,\mu\mathrm{s}, 3\,\mu\mathrm{s}, 4\,\mu\mathrm{s}\}$ and $y \in \{w, f\}$. The confidence region (one standard deviation above and below the mean containing 68 % of the samples) and mean through time: $\sqrt{\mathbf{Var}\big[\Phi_e^y(t)\big]}$ and $\mathbf{E}\big[\Phi_e^y(t)\big]$, given these four realizations, are represented by a shaded area and a solid line, respectively.

Note. The standard deviation through time for flicker FM and white FM noise increases linearly and with a square root law, respectively.

4.3 ERO-ES Entropy Model

To study the effects of flicker FM noise on the entropy generated by an ES, the ERO-ES is selected as the reference architecture in this chapter. As shown in fig. 4.2, the ERO-ES being studied consists of a single free-running RO, a reference clocking signal and a sampling FF.

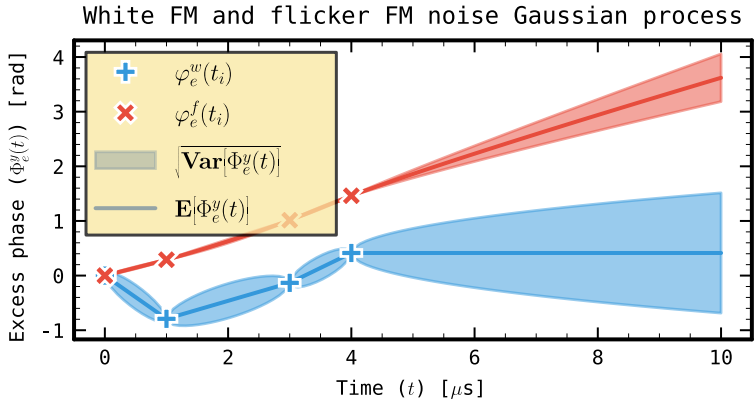

Figure 4.1: Gaussian process for white FM (blue) and flicker FM (red) noise in example 4.1. The markers represent realized values for the excess phase process. The shaded area and the solid line represent the standard deviation and mean for the process at any time instant respectively, given the realizations: $\Phi_e^y(t_i) = \varphi_e^y(t_i)$ for $t_i \in \{0\,\mu s, 1\,\mu s, 3\,\mu s, 4\,\mu s\}$ and $y \in \{w, f\}$.

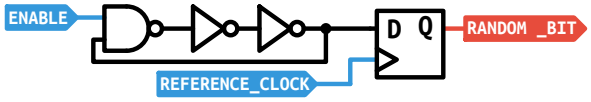

Figure 4.2: ERO-ES reference architecture.

Note. Although the derivations in this chapter assume the reference clock is *jitter-free*, in reality, the reference clock may also exhibit jitter.

Techniques exist to transfer the *jitter* from the reference clock to the free-running oscillator under study [21], allowing the results presented here to remain valid. However, when the origin of the jitter in the reference clock is unknown, it should not be considered in the entropy estimate.

From section 4.2, both the white and flicker excess phase components can be considered a Gaussian process. A random excess phase vector is defined as

$$\vec{\mathbf{\Phi}}_{\boldsymbol{e}} = \left(\Phi_e^w(t_1), \Phi_e^w(t_2), \ldots, \Phi_e^w(t_n), \Phi_e^f(t_1), \Phi_e^f(t_2), \ldots, \Phi_e^f(t_n)\right)^\intercal,$$

describing the sampled excess phase at time instances $(t_1, t_2, \ldots, t_n)^\intercal \in \mathbb{R}^n_{\geq 0}$. The excess phase vector is then multivariate normal distributed, as the white and

flicker noise components are independent: $\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}} \sim \mathcal{N}_{2n}\left(\vec{\mathbf{0}}_{\boldsymbol{2n}}, \boldsymbol{\Sigma}_{\boldsymbol{e}}\right)$. The covariance matrix is constructed as

$$\boldsymbol{\Sigma}_{\boldsymbol{e}} = \begin{bmatrix} \boldsymbol{\Sigma}_{\boldsymbol{e}}^{\boldsymbol{w}} & \mathbf{0}_{\boldsymbol{n}\times\boldsymbol{n}} \\ \mathbf{0}_{\boldsymbol{n}\times\boldsymbol{n}} & \boldsymbol{\Sigma}_{\boldsymbol{e}}^{\boldsymbol{f}} \end{bmatrix}, \tag{4.7}$$

with $\mathbf{0}_{\boldsymbol{n}\times\boldsymbol{n}}$, an $n \times n$ all zero matrix. The $n \times n$ matrices $\boldsymbol{\Sigma}_{\boldsymbol{e}}^{\boldsymbol{w}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{e}}^{\boldsymbol{f}}$ are the covariance matrices for the individual white and flicker noise components respectively, constructed using the noise ACF, as illustrated in eq. (4.2).

4.3.1 Bit Distribution

In an ERO-ES, the oscillator is sampled at regular time intervals: $t_i = it_{acc}$ for $i \in \{1, 2, \ldots, n\}$ and accumulation time, t_{acc}. The sampled value of the i-th bit, B_i, equals

$$\begin{aligned} B_i &= \left\lfloor \frac{\Phi(t_i)}{\pi} \right\rfloor \bmod 2 = \left\lfloor \frac{2\pi f_n t_i + \phi_0 + \Phi_e(t_i)}{\pi} \right\rfloor \bmod 2 \\ &= b_i^d \oplus \left\lfloor \frac{\phi_i + \Phi_e(t_i)}{\pi} \right\rfloor \bmod 2, \end{aligned} \tag{4.8}$$

with $b_i^d = \left\lfloor \frac{2\pi f_n t_i + \phi_0}{\pi} \right\rfloor \bmod 2$ and $\phi_i = (2\pi f_n t_i + \phi_0) \bmod \pi$, the number of completed half cycles modulo two and fractional part of the current oscillator half cycle, both deterministic quantities and $\oplus$, the binary XOR operator. XORing with b_i^d can be considered a form of post-processing, as it does not alter the entropy content of the bit B_i. Remove the XOR post-processing to obtain an adjusted bit with identical entropy content:

$$B_i' = \left\lfloor \frac{\phi_i + \Phi_e(t_i)}{\pi} \right\rfloor \bmod 2 = \left\lfloor \frac{\phi_i + \Phi_e^w(t_i) + \Phi_e^f(t_i)}{\pi} \right\rfloor \bmod 2.$$

B_i' is a discrete binary *random variable*, dependent on the oscillator excess phase, $\Phi_e(t_i)$, at sampling time t_i. The conditional Probability Mass Function (PMF), given the excess phase equals

$$f_{B_i'|\Phi_e(t_i)}(b \mid \varphi) = \begin{cases} 1 & \text{if } b = \left\lfloor \frac{\phi_i + \varphi}{\pi} \right\rfloor \bmod 2 \\ 0 & \text{otherwise} \end{cases}.$$

When considering m sample time instances: $\forall j \in \mathbb{N}_m, \forall i_j \in \{1, 2, \ldots, n\}$: $(t_{i_0}, t_{i_1}, \ldots, t_{i_{m-1}})^\intercal \in \mathbb{R}_{\geq 0}^m$, define the random bit vector as follows: $\vec{\boldsymbol{B}} =$

$(B'_{i_0}, B'_{i_1}, \ldots, B'_{i_{m-1}})^\intercal$, with conditional PMF equal to

$$f_{\vec{\boldsymbol{B}}|\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}}) = \prod_{j=0}^{m-1} f_{B'_{i_j}|\Phi_e(t_{i_j})}(b_{i_j} \mid \varphi_{i_j-1} + \varphi_{i_j-1+n}), \tag{4.9}$$

where b_{i_j}, φ_{i_j-1}, and φ_{i_j-1+n} are the elements at the j-th, (i_j-1)-th, and (i_j-1+n)-th position in the vectors $\vec{\boldsymbol{b}}$, and $\vec{\boldsymbol{\varphi}}$, respectively. This PMF only equals one if for all m indexes, i_j, the bit b_{i_j} equals $\left\lfloor \frac{\phi_{i_j}+\varphi_{i_j-1}+\varphi_{i_j-1+n}}{\pi} \right\rfloor \bmod 2$, with φ_{i_j-1} and φ_{i_j-1+n}, the given excess white and flicker phase respectively for the index corresponding with sampling time t_{i_j}.

4.3.2 Conditional Distributions

The model presented in this chapter assumes that an entity has observed or will observe a certain amount of information about the ERO-ES. This information could include the exact oscillator phase value, $\Phi_e^w(t_i)$ and $\Phi_e^f(t_i)$, or the produced output bit B'_i at specific sampling time instances: $t_i \in \mathbb{R}_{\geq 0}$. Take as an example: person `A` collecting future or previous ES output. The collection of already observed random variables is referred to as the observed part. The conditional distributions developed in this study, describe the distribution for an unobserved part (phase or bit values), from the perspective of that entity who already possesses knowledge of the observed part. For instance: the current bits' distribution for person `A`, given this person has the knowledge of different bits produced in the future or past.

The n sampling time instances of interest are now partitioned in an observed part: $\{t_{i_0^o}, t_{i_1^o}, \ldots, i_{n^o-1}^o\}$ and an unobserved part: $\{t_{i_0^u}, t_{i_1^u}, \ldots, t_{i_{n^u-1}^u}\}$, containing n^o and n^u time instances, respectively and $n = n^o + n^u$, the total number of samples under study. A sample instance cannot be both observed and unobserved: $\forall (k,j) \in \mathbb{N}_{n^u} \times \mathbb{N}_{n^o} : i_k^u \neq i_j^o$.

The excess phase vector can be similarly partitioned (by reordering the indexes) into observed and unobserved parts:

$$\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}} = \left(\boldsymbol{\Phi}_{\boldsymbol{e}}^{\vec{\boldsymbol{o}},\boldsymbol{w}^\intercal}, \boldsymbol{\Phi}_{\boldsymbol{e}}^{\vec{\boldsymbol{u}},\boldsymbol{w}^\intercal}, \boldsymbol{\Phi}_{\boldsymbol{e}}^{\vec{\boldsymbol{o}},\boldsymbol{f}^\intercal}, \boldsymbol{\Phi}_{\boldsymbol{e}}^{\vec{\boldsymbol{u}},\boldsymbol{f}^\intercal}\right)^\intercal,$$

with: $\forall (x,y) \in \{o,u\} \times \{w,f\} : \boldsymbol{\Phi}_{\boldsymbol{e}}^{\vec{\boldsymbol{x}},\boldsymbol{y}} = \left(\Phi_e^y(t_{i_0^x}), \Phi_e^y(t_{i_1^x}), \ldots, \Phi_e^y(t_{i_{n^x-1}^x})\right)^\intercal$. Similarly, the random bit vectors representing the observed and unobserved ES output equal $\forall x \in \{o,u\} : \vec{\boldsymbol{B}}^{\vec{\boldsymbol{x}}} = (B'_{i_0^x}, B'_{i_1^x}, \ldots, B'_{i_{n^x-1}^x})^\intercal$.

Conditioned on the Oscillator Phase

The unobserved excess phase vector, $\vec{\boldsymbol{\Phi}}_e^{\vec{u}} = \left(\boldsymbol{\Phi}_e^{\vec{u},w^\mathsf{T}}, \boldsymbol{\Phi}_e^{\vec{u},f^\mathsf{T}}\right)^\mathsf{T}$, given a realization of the observed excess phase, $\vec{\boldsymbol{\Phi}}_e^{o} = \left(\boldsymbol{\Phi}_e^{\vec{o},w^\mathsf{T}}, \boldsymbol{\Phi}_e^{\vec{o},f^\mathsf{T}}\right)^\mathsf{T} = \left(\vec{\boldsymbol{\varphi}}^{w^\mathsf{T}}, \vec{\boldsymbol{\varphi}}^{f^\mathsf{T}}\right)^\mathsf{T} = \vec{\boldsymbol{\varphi}}^{o}$, has a multivariate normal conditional Probability Density Function (PDF):

$$f_{\vec{\boldsymbol{\Phi}}_e^{u}|\vec{\boldsymbol{\Phi}}_e^{o}}\left(\vec{\boldsymbol{\varphi}}^{u} \mid \vec{\boldsymbol{\varphi}}^{o}\right) = \phi_{\mathcal{N}_{2n^u}}\left(\vec{\boldsymbol{\varphi}}^{u}; \vec{\boldsymbol{\mu}}_e^{u|o}, \boldsymbol{\Sigma}_e^{u|o}\right), \tag{4.10}$$

with conditional covariance matrix equal to

$$\boldsymbol{\Sigma}_e^{u|o} = \begin{bmatrix} \boldsymbol{\Sigma}_e^{uu,w} - \boldsymbol{\Sigma}_e^{uo,w}\boldsymbol{\Sigma}_e^{oo,w\,-1}\boldsymbol{\Sigma}_e^{ou,w} & \mathbf{0}_{n^u \times n^u} \\ \mathbf{0}_{n^u \times n^u} & \boldsymbol{\Sigma}_e^{uu,f} - \boldsymbol{\Sigma}_e^{uo,f}\boldsymbol{\Sigma}_e^{oo,f\,-1}\boldsymbol{\Sigma}_e^{ou,f} \end{bmatrix}, \tag{4.11}$$

and conditional mean vector equal to

$$\vec{\boldsymbol{\mu}}_e^{u|o} = \begin{bmatrix} \boldsymbol{\Sigma}_e^{uo,w}\boldsymbol{\Sigma}_e^{oo,w\,-1}\vec{\boldsymbol{\varphi}}^{w} \\ \boldsymbol{\Sigma}_e^{uo,f}\boldsymbol{\Sigma}_e^{oo,f\,-1}\vec{\boldsymbol{\varphi}}^{f} \end{bmatrix}.$$

The submatrices are derived from the excess phase covariance matrix in eq. (4.7):

$$\boldsymbol{\Sigma}_e = \begin{bmatrix} \boldsymbol{\Sigma}_e^{oo,w} & \boldsymbol{\Sigma}_e^{ou,w} & \mathbf{0}_{n\times n} & \\ \boldsymbol{\Sigma}_e^{uo,w} & \boldsymbol{\Sigma}_e^{uu,w} & & \\ & \mathbf{0}_{n\times n} & \boldsymbol{\Sigma}_e^{oo,f} & \boldsymbol{\Sigma}_e^{ou,f} \\ & & \boldsymbol{\Sigma}_e^{uo,f} & \boldsymbol{\Sigma}_e^{uu,f} \end{bmatrix},$$

Note. Only the conditional mean is dependent on the phase realization.

Conditioned on the Output Bits

Using Bayes' rule, the conditional PDF for the random excess phase vector, $\vec{\boldsymbol{\Phi}}_e$, given the observation of n^o bits $\vec{\boldsymbol{B}}^o = \vec{\boldsymbol{b}}$, equals

$$f_{\vec{\boldsymbol{\Phi}}_e|\vec{\boldsymbol{B}}^o}(\vec{\boldsymbol{\varphi}} \mid \vec{\boldsymbol{b}}) = \frac{f_{\vec{\boldsymbol{B}}^o|\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}}) f_{\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{\varphi}})}{f_{\vec{\boldsymbol{B}}^o}(\vec{\boldsymbol{b}})}, \tag{4.12}$$

with the *unconditional* excess phase PDF equal to the $2n$-dimensional multivariate normal distribution PDF, $f_{\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{\varphi}}) = \phi_{\mathcal{N}_{2n}}(\vec{\boldsymbol{\varphi}}; \vec{\mathbf{0}}_{\vec{2n}}, \boldsymbol{\Sigma}_e)$. The unconditional marginal PMF, for a bit vector, $f_{\vec{\boldsymbol{B}}}(\vec{\boldsymbol{b}})$ or equivalently $f_{\vec{\boldsymbol{B}}^o}(\vec{\boldsymbol{b}})$, can be found by integrating the unconditional excess phase PDF over the subspace where $f_{\vec{\boldsymbol{B}}|\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}})$, from eq. (4.9), equals one:

$$f_{\vec{\boldsymbol{B}}}(\vec{\boldsymbol{b}}) = \int_{\mathbb{R}^{2n}} f_{\vec{\boldsymbol{B}}|\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}}) f_{\vec{\boldsymbol{\Phi}}_e}(\vec{\boldsymbol{\varphi}}) \mathrm{d}\vec{\boldsymbol{\varphi}}. \tag{4.13}$$

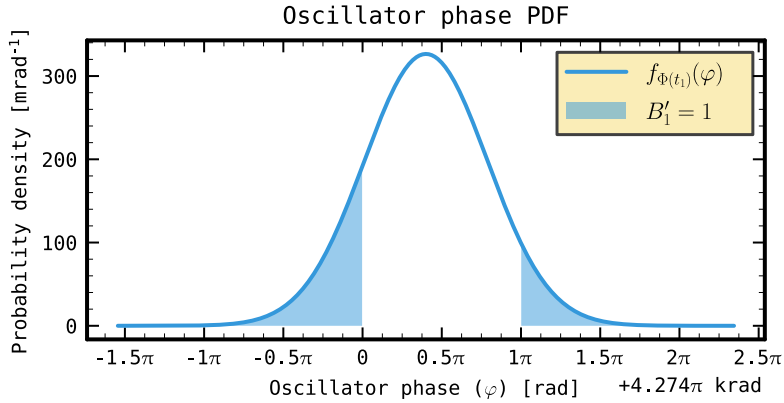

Figure 4.3: Oscillator total phase PDF and integration area for the first sample being equal to one.

Example 4.2. Take the following realistic parameter values: $f_n = 520\,\text{MHz}$, $t_{acc} = 4.11\,\mu\text{s}$, $\phi_0 = 0\,\text{rad}$ (initial oscillator phase), $h_w = 18.9\,\text{fs}$ (white *noise strength*, as measured in chapter 3), and $h_f = 100 \times 10^{-12}$ (flicker noise strength, with a *noise corner* around $5\,\mu\text{s}$). In this example, two sampling time instances are considered: $t_1 = t_{acc}$ and $t_2 = 2t_{acc}$. The random excess phase vector becomes $\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}} = \left(\Phi_e^w(t_1), \Phi_e^w(t_2), \Phi_e^f(t_1), \Phi_e^f(t_2)\right)^\intercal$. The generated bit from the first sample is observed equal to one, $b_{i_0^o} = 1$ for $i_0^o = 1$ and $\vec{\boldsymbol{B}}^{\boldsymbol{o}} = [B'_{i_0^o}]$. The probability of sampling a one at the first sampling time instance, is obtained by integrating the unconditional total oscillator phase PDF over the area highlighted in fig. 4.3 and equals

$$f_{\vec{\boldsymbol{B}}^{\boldsymbol{o}}}([1]) = \int_{\left\lfloor \frac{\phi_1+\varphi_0+\varphi_2}{\pi} \right\rfloor \bmod 2=1} \phi_{\mathcal{N}_4}(\vec{\boldsymbol{\varphi}}; \vec{\mathbf{0}}_{\mathbf{4}}, \boldsymbol{\Sigma}_{\boldsymbol{e}}) \mathrm{d}\vec{\boldsymbol{\varphi}} = 21.3\,\%,$$

with the unconditional excess phase covariance matrix equal to

$$\boldsymbol{\Sigma}_{\boldsymbol{e}} = \begin{bmatrix} R^w_{\Phi_e}(t_1,t_1) & R^w_{\Phi_e}(t_1,t_2) & 0 & 0 \\ R^w_{\Phi_e}(t_2,t_1) & R^w_{\Phi_e}(t_2,t_2) & 0 & 0 \\ 0 & 0 & R^f_{\Phi_e}(t_1,t_1) & R^f_{\Phi_e}(t_1,t_2) \\ 0 & 0 & R^f_{\Phi_e}(t_2,t_1) & R^f_{\Phi_e}(t_2,t_2) \end{bmatrix}.$$

Equation (4.12) determines that the knowledge of the first bit being equal to one, changes the distributions for the white- and flicker excess phase at the second sample, as illustrated by figs. 4.4 and 4.5.

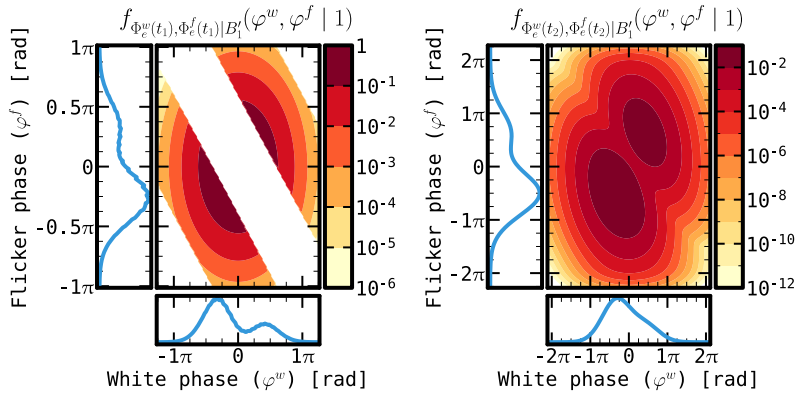

Figure 4.4: Joint PDFs for the white and flicker excess phase at the first sampling time instance: t_1 (left) and second sampling time instance: t_2 (right), conditioned on the first sampled bit being equal to one.

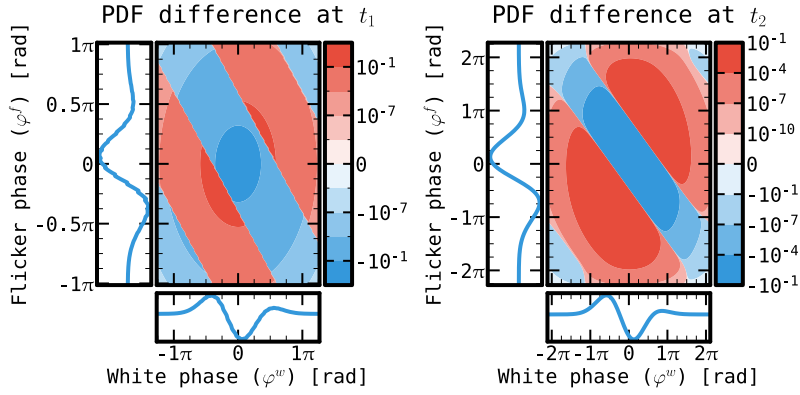

Figure 4.5: Difference between the unconditioned joint PDF and the conditioned joint PDF: $f_{\Phi_e^w(t_i),\Phi_e^f(t_i)|B_1'}(\varphi^w,\varphi^f \mid 1) - f_{\Phi_e^w(t_i),\Phi_e^f(t_i)}(\varphi^w,\varphi^f)$, for t_i equal to t_1 (left) or t_2 (right).

4.3.3 Monte Carlo Integration

Example 4.2 was generated by evaluating the four-dimensional multivariate normal PDF, $\phi_{\mathcal{N}_4}$, at a four-dimensional grid. While providing accurate results, this approach quickly becomes intractable when dealing with more dimensions (i.e. the number of samples at interest increases). When evaluating eq. (4.12), only the denominator poses a problem. The unconditional bit PMF, $f_{\vec{\boldsymbol{B}}}(\vec{\boldsymbol{b}})$, requires the integration of a multivariate normal PDF in a $2n$-dimensional subspace, as shown by eq. (4.13). No closed form solution exists when there is correlation between the individual random variables or the integration boundaries depend on more than one random variable.

The knowledge of the generated bit at time t_{i_j}: $B'_{i_j} = b_{i_j}$, leads to an additional factor in eq. (4.9). The excess phase becomes additionally constrained by the relation: $\left\lfloor \frac{\phi_{i_j} + \Phi_e(t_{i_j})}{\pi} \right\rfloor \bmod 2 = b_{i_j}$, which generates a periodic tilted bands integration region, with period 2π, in the $\Phi_e^w(t_{i_j})$, $\Phi_e^f(t_{i_j})$-plane, as could be seen in fig. 4.4 (left).

For evaluating the integral in eq. (4.13), a *Monte Carlo* integration method is used. Generate s random samples, $\vec{\boldsymbol{\varphi}}_{\boldsymbol{i}}$ for $i \in \mathbb{N}_s$, from the excess phase distribution, $\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}$ (multivariate normal). The ratio between the number of samples that fall inside the integration region to the total number of generated samples is used as an approximation to the integral in eq. (4.13):

$$f_{\vec{\boldsymbol{B}}}(\vec{\boldsymbol{b}}) \approx \frac{1}{s} \left| \left\{ \vec{\boldsymbol{\varphi}}_{\boldsymbol{i}} \mid f_{\vec{\boldsymbol{B}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}}_{\boldsymbol{i}}) = 1, i \in \mathbb{N}_s \right\} \right|. \tag{4.14}$$

For all the results presented in this chapter, the `multivariate_normal()` method from the `numpy.random` Python library was used to generate one million (s) samples.

4.3.4 Entropy Study

Using eq. (4.13) or eq. (4.14) as an approximation, to obtain the unconditional PMF for the bit vector $\vec{\boldsymbol{B}}^{\boldsymbol{u}}$, the unconditional Shannon entropy for the unobserved bits can be determined as usual:

$$\mathbf{H}[\vec{\boldsymbol{B}}^{\boldsymbol{u}}] = - \sum_{\vec{\boldsymbol{b}} \in \{0,1\}^{n_u}} f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}}(\vec{\boldsymbol{b}}) \log_2 \left(f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}}(\vec{\boldsymbol{b}}) \right). \tag{4.15}$$

Entropy Conditioned on the Oscillator Phase

Given the realization of n^o phase values, at sampling time instances $(t_{i_0^o}, t_{i_1^o}, \ldots, t_{i_{n^o-1}^o})^\intercal \in \mathbb{R}_{\geq 0}^{n^o}$: $\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o}} = \vec{\boldsymbol{\varphi}}$ and combining eqs. (4.9) and (4.10), the unobserved bits PMF, given the observed oscillator phase at n^o time instances equals

$$f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}}^{\boldsymbol{o}}) = \int_{\mathbb{R}^{2n^u}} f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}}}(\vec{\boldsymbol{b}} \mid \vec{\boldsymbol{\varphi}}^{\boldsymbol{u}}) f_{\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o}}}(\vec{\boldsymbol{\varphi}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{\varphi}}^{\boldsymbol{o}}) \mathrm{d}\vec{\boldsymbol{\varphi}}^{\boldsymbol{u}}.$$

From this conditional PMF, the conditional Shannon entropy, given the observed phase values, $\mathbf{H}[\vec{\boldsymbol{B}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o}} = \vec{\boldsymbol{\varphi}}^{\boldsymbol{o}}]$, can be determined similarly to eq. (4.15).

Entropy Conditioned on the Output Bits

Given again n^o bit observations $\vec{\boldsymbol{b}}^{\boldsymbol{o}} = (b_0^o, b_1^o, \ldots, b_{n^o-1}^o)^\intercal \in \{0,1\}^{n^o}$, at sampling time instances $(t_{i_0^o}, t_{i_1^o}, \ldots, t_{i_{n^o-1}^o})^\intercal \in \mathbb{R}_{\geq 0}^{n^o}$, we therefore have a realization: $\vec{\boldsymbol{B}}^{\boldsymbol{o}} = \vec{\boldsymbol{b}}^{\boldsymbol{o}}$. Combining eqs. (4.9) and (4.12), the conditional unobserved bits PMF, given the realization of $\vec{\boldsymbol{B}}^{\boldsymbol{o}}$ can be determined as follows:

$$f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{b}}^{\boldsymbol{o}}) = \int_{\mathbb{R}^{2n}} f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}}(\vec{\boldsymbol{b}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{\varphi}}) f_{\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}|\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{\varphi}} \mid \vec{\boldsymbol{b}}^{\boldsymbol{o}}) \mathrm{d}\vec{\boldsymbol{\varphi}}.$$

Using the Monte Carlo integration method from section 4.3.3, this conditional PMF can be estimated by the following ratio:

$$f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{b}}^{\boldsymbol{o}}) = \frac{f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}},\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}}^{\boldsymbol{u}}, \vec{\boldsymbol{b}}^{\boldsymbol{o}})}{f_{\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}}^{\boldsymbol{o}})},$$

with

$$f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}},\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}}^{\boldsymbol{u}}, \vec{\boldsymbol{b}}^{\boldsymbol{o}}) \approx \frac{1}{s}\left|\left\{\vec{\boldsymbol{\varphi}}_{\boldsymbol{i}} \mid f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}}(\vec{\boldsymbol{b}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{\varphi}}_{\boldsymbol{i}}) = 1, f_{\vec{\boldsymbol{B}}^{\boldsymbol{o}}|\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}}(\vec{\boldsymbol{b}}^{\boldsymbol{o}} \mid \vec{\boldsymbol{\varphi}}_{\boldsymbol{i}}) = 1, i \in \mathbb{N}_s\right\}\right|,$$

and with $f_{\vec{\boldsymbol{B}}^{\boldsymbol{o}}}(\vec{\boldsymbol{b}}^{\boldsymbol{o}})$ from eq. (4.14). Use s randomly generated samples from the unconditional multivariate normal $\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}$ distribution, $\vec{\boldsymbol{\varphi}}_{\boldsymbol{i}}$ for $i \in \mathbb{N}_s$. From the conditional PMF, $f_{\vec{\boldsymbol{B}}^{\boldsymbol{u}}|\vec{\boldsymbol{B}}^{\boldsymbol{o}}}$, the conditional Shannon entropy for the unobserved bits, given the observed bits, $\mathbf{H}[\vec{\boldsymbol{B}}^{\boldsymbol{u}} \mid \vec{\boldsymbol{B}}^{\boldsymbol{o}} = \vec{\boldsymbol{b}}^{\boldsymbol{o}}]$, can be determined similar to eq. (4.15).

Example 4.3. Given the scenario from example 4.2, the total oscillator phase PDF is shown in fig. 4.6. Both the unconditional PDF as the conditional PDF, given the first sample equaled one, are shown. The conditional probability of obtaining a one at the second sample equals 52.8 % and the conditional Shannon entropy is $\mathbf{H}[B_2' \mid B_1' = 1] = 0.998$ bit.

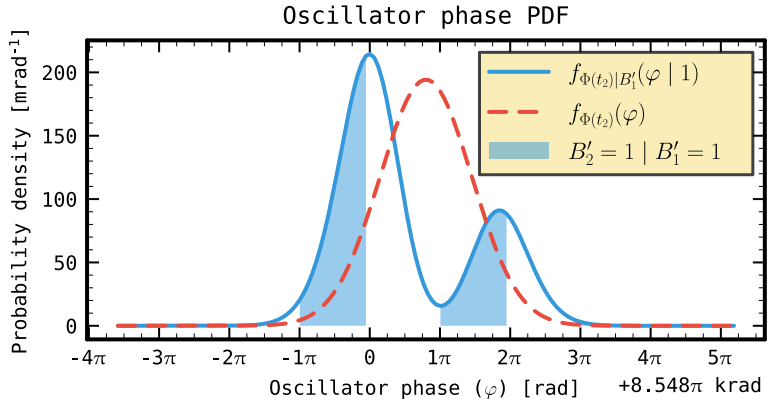

Figure 4.6: Oscillator total phase PDF and integration area for the second sample being equal to one, conditioned on the first sample being equal to one (solid blue) and unconditioned (dashed red).

Worst-case Entropy

Despite the bimodal shape of the conditioned total oscillator phase PDF from fig. 4.6 at example 4.3, the obtained Shannon entropy for the second sample, given the first sample equals one is significantly larger than the entropy for the first sample, $\mathbf{H}[B_1'] = 0.747\,\text{bit}$ vs. $\mathbf{H}[B_2' \mid B_1' = 1] = 0.998\,\text{bit}$. The area under the PDF curve used for determining the bit probability is highly influenced by the horizontal position of the curve. This horizontal position is determined by the nominal phase at the sampling time: $2\pi f_n i t_{acc} + \phi_0 \bmod 2\pi$ for the i-th sample, related to ϕ_i in eq. (4.8). To eliminate the influence of the nominal phase on the entropy, a *worst-case* entropy function is defined.

Definition 4.1. (Worst-case entropy) Given an oscillator total phase random variable at some time instance $\Phi : \Omega \to \mathbb{R}$ and its corresponding random bit, $B = \lfloor \frac{\Phi}{\pi} \rfloor \bmod 2$. A worst-case random bit function, $B_{worst}(\delta_\phi) : \Omega \times [0, 2\pi) \to \{0, 1\}$, is defined as

$$B_{worst}(\delta_\phi) = \left\lfloor \frac{\Phi + \delta_\phi}{\pi} \right\rfloor \bmod 2,$$

with δ_ϕ, a deterministic phase offset. The worst-case Shannon entropy for B, conditioned on an event, $E \in \mathcal{F}$, is equal to

$$\mathbf{H_{worst}}[B \mid E] = \mathbf{H}\big[B_{worst}(\delta_\phi^m) \mid E\big],$$

with: $\delta_\phi^m = \arg\max_{\delta_\phi \in [0,2\pi)} \mathbf{P}\big[B_{worst}(\delta_\phi) = 1\big]$.

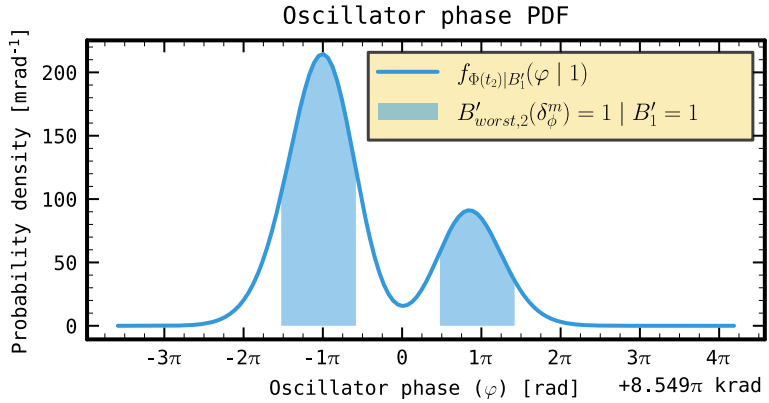

Figure 4.7: Oscillator total phase PDF and worst-case entropy integration area for the second sample being equal to one, given the first sample was one.

As the worst-case entropy is independent of a phase shift, one can assign a worst-case entropy value to a phase random variable. Both the total oscillator phase as only the excess oscillator phase have an equal worst-case entropy content, as they only differ in a deterministic phase offset (nominal phase).

Definition 4.2. (Worst-case entropy for a phase random variable) Given an oscillator's total phase and excess phase random variables at some time instance $\Phi : \Omega \to \mathbb{R}$ and $\Phi_e : \Omega \to \mathbb{R}$. The worst-case Shannon entropy for Φ and Φ_e, conditioned on an event, $E \in \mathcal{F}$, is equal to

$$\mathbf{H_{worst}}[\Phi \mid E] = \mathbf{H_{worst}}[\Phi_e \mid E] = \mathbf{H_{worst}}[B \mid E],$$

with $B : \Omega \to \{0, 1\}$, a random bit extracted from that oscillator, $B = \left\lfloor \frac{\Phi + \delta_\phi}{\pi} \right\rfloor \bmod 2$, for any phase shift $\delta_\phi \in \mathbb{R}$.

Example 4.4. Given the scenario from example 4.2, fig. 4.7 provides the conditional PDF for the total oscillator phase at the second sampling moment, given the first sample obtained a one. The shaded area indicates the worst-case probability of obtaining a one, equaling 76.6 %. The worst-case Shannon entropy now becomes $\mathbf{H_{worst}}[B_2' \mid B_1' = 1] = 0.785\,\text{bit}$, which is significantly reduced compared to the regular Shannon entropy in example 4.3, and independent of the nominal oscillator phase.

Table 4.1: Numerical values used for: h_w, h_f and t_{acc} and obtained white FM noise $\mathbf{H_{worst}}$.

Estimate	h_w	h_f	Corner	Sampling speed 25 %		100 %		400 %	
				t_{acc}	$\mathbf{H_{worst}}$	t_{acc}	$\mathbf{H_{worst}}$	t_{acc}	$\mathbf{H_{worst}}$
	[fs]	$[1 \times 10^{-12}]$	[µs]	[µs]	[bit]	[µs]	[bit]	[µs]	[bit]
Low	-	6.22	100	25.0	0.992	100	> 0.999	400	> 0.999
Mid	18.9	104	5.00	1.25	0.0186	5.00	0.523	20.0	0.979
High	-	9480	0.0434	0.0109	< 0.001	0.0434	< 0.001	0.174	< 0.001

4.4 Model Simulation

The Gaussian process model, developed in section 4.3, will now be used to generate entropy estimates for an ERO-ES affected by both white FM and flicker FM noise under realistic *operating conditions*.

4.4.1 Noise Magnitude

This section explains how the scaling constants h_w and h_f, used to approximate both the white FM and flicker FM noise magnitude, are selected. Based on the obtained noise corner, the ERO-ES sampling speed is determined as well. Table 4.1 lists the numerical values used in the remainder of this chapter.

White FM Noise Magnitude

Throughout this chapter, a single magnitude for the white FM noise is used: $h_w = 18.9\,\text{fs}$. As provided by table 3.9 on page 58 in chapter 3, this value is lower than most other estimates in the field and can therefore be considered as a conservative estimate.

There exists a one-to-one relation between the white FM noise component worst-case Shannon entropy and the accumulation time. Figure 4.8 depicts this relation for the selected white FM noise magnitude (solid red) and other magnitudes (dashed gray). Higher noise magnitudes give a higher entropy value at a given accumulation time.

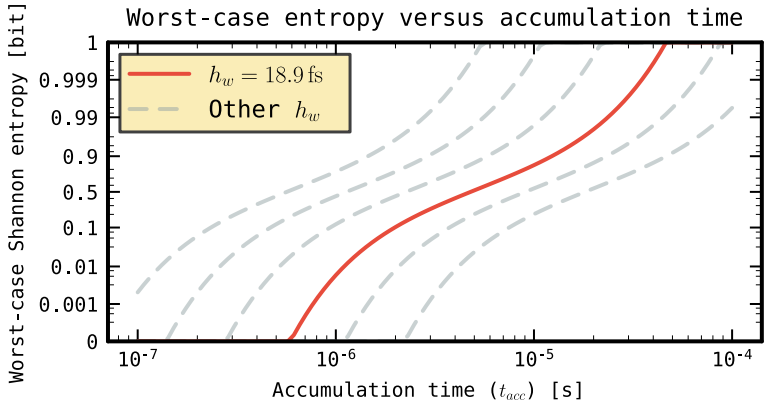

Figure 4.8: Worst-case Shannon entropy for a white FM noise source, versus accumulation time. Entropy curves for noise magnitudes $h_w = 18.9\,\mathrm{fs}$ (solid red) and other magnitudes (dashed gray) for reference: $\{5\,\mathrm{fs}, 10\,\mathrm{fs}, 40\,\mathrm{fs}, 80\,\mathrm{fs}, 160\,\mathrm{fs}\}$ are shown.

Flicker FM Noise Magnitude

This chapter considers three different magnitudes for the flicker FM noise component. At the higher end of the spectrum, there is the magnitude as measured in section 3.4, $h_f = 9.48 \times 10^{-9}$. At the lower end, the noise corner derived from the measurements in [27] lead to a magnitude $h_f = 6.22 \times 10^{-12}$, when using the white noise strength estimate from section 4.4.1. The third value is selected in between, $h_f = 1.04 \times 10^{-10}$, and approaches what has been reported by [46] and [21]. A frequency value $f_l = 1\,\mathrm{mHz}$ is used for the lower frequency bound in eq. (4.6).

White FM - Flicker FM Noise Corner

The presence of both white FM and flicker FM noise gives rise to a noise corner. The noise corner represents a pair $\big(t_{cor}, \mathbf{Var}\big[\Phi(t_{cor})\big]\big)$, for which the accumulated oscillator white FM phase variance equals the accumulated flicker FM phase variance. From eqs. (4.4) and (4.6), the corner accumulation time satisfies the relation: $t_{cor}\big(3 - 2\gamma - 2\ln(2\pi f_l t_{cor})\big) = \frac{h_w}{h_f}$.

Figure 4.9 depicts the accumulated oscillator phase variance versus accumulation time, for the three flicker FM noise magnitudes considered in this chapter. For accumulation times below the noise corner, the white FM noise component is *dominant* and the oscillator phase variance increases linearly. Above the noise

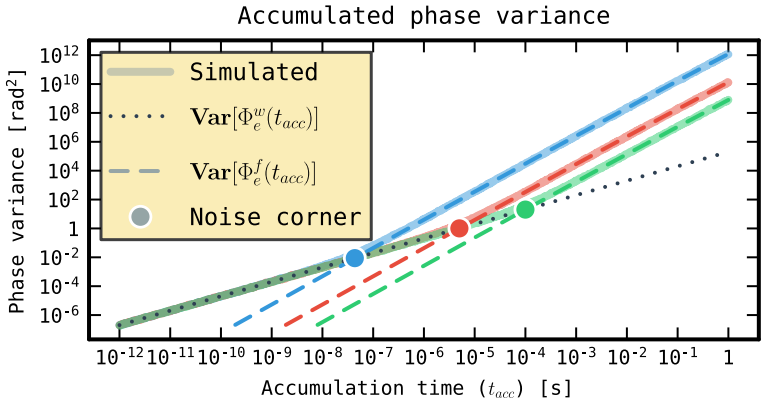

Figure 4.9: Oscillator phase variance versus accumulation time for the three different flicker FM noise magnitudes: $\{6.22 \times 10^{-12}, 1.04 \times 10^{-10}, 9.48 \times 10^{-9}\}$, represented by distinct colors, used in this work.

corner, the flicker FM noise component dominates and the variance increases in a quadratic way. The dotted and dashed lines in fig. 4.9 represent the theoretical phase variance from eqs. (4.4) and (4.6) respectively. The simulation results, when using the Gaussian process model are shown as solid opaque curves. The noise corner accumulation time values obtained for the flicker FM magnitudes are provided in table 4.1.

Sampling Speed

Depending on the flicker FM magnitude, the accumulation time is selected as 25 %, 100 % and 400 % of the value of the noise corner. At 25 %, the white FM noise component will dominate and at 400 %, the flicker FM noise component dominates. Table 4.1 provides the accumulation times at interest and the corresponding worst-case Shannon entropy for the white FM component, also visible in fig. 4.8, when the ERO-ES is sampled at t_{acc} time intervals.

4.4.2 Entropy Estimation

This section presents numerical results for the conditional ERO-ES worst-case entropy, from implementing the theory outlined in sections 4.3.2 and 4.3.4. The subsections are arranged in decreasing knowledge of the oscillator's state: in the first subsection, we assume the observation of the complete oscillator phase,

whereas in the second subsection only the produced ES output bits are assumed to be known.

Knowledge of the Previous Phase Values

The worst-case Shannon entropy is evaluated when p previous sample phase values are known, $\mathbf{H_{worst}}[\vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}} \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}} = \vec{\boldsymbol{\varphi}}]$. The entropy for the sixth bit from an ERO-ES is calculated, given the knowledge of the previous p sample phases, for p ranging from zero up to five. The unobserved/observed excess phase vectors become

$$\vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}} = \left(\Phi_e^w(t_6), \Phi_e^f(t_6)\right)^\intercal,$$

$$\vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}} = \left(\Phi_e^w(t_5), \Phi_e^w(t_4), \ldots, \Phi_e^w(t_{6-p}), \Phi_e^f(t_5), \Phi_e^f(t_4), \ldots, \Phi_e^f(t_{6-p})\right)^\intercal,$$

for $p \in \mathbb{N}_6$. When p equals zero, no phase information is known and the entropy becomes unconditioned.

Note. From eq. (4.11), the conditional covariance matrix for $\vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}} \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}} = \vec{\boldsymbol{\varphi}}$ is independent of the actual realized value of the previous sample phases, $\vec{\boldsymbol{\varphi}}$, and note that the worst-case entropy is independent of a phase shift introduced by the conditional mean. The worst-case entropy, given the knowledge of p previously observed sample phases is therefore not influenced by the realized value itself: $\mathbf{H_{worst}}[\vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}} \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}} = \vec{\boldsymbol{\varphi}}] = \mathbf{H_{worst}}[\vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}} \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}}]$.

Knowledge of p Previous Sample Phases Figures 4.10 to 4.12 provide the worst-case Shannon entropy and phase standard deviation for the sixth bit, given knowledge of p previous phase values, for flicker FM noise magnitudes 6.22×10^{-12}, 1.04×10^{-10} and 9.48×10^{-9}, respectively. The entropy and phase standard deviation values are given for white FM and flicker FM noise separately, $\mathbf{H_{worst}}\left[\Phi_e^y(t_6) \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}}\right]$ and $\sqrt{\mathbf{Var}\left[\Phi_e^y(t_6) \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}}\right]}$, for $y \in \{w, f\}$, respectively.

As seen from these figures, the entropy reduces significantly when the previous sample phase is known, both for white FM and flicker FM noise. For white FM noise, the entropy remains constant for $p \geq 1$ and the phase variance equals the variance accumulated between the fifth and sixth sample: $\mathbf{Var}[\Phi_e^w(t_6) \mid \vec{\mathbf{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{o},\boldsymbol{p}}] = 4\pi^2 f_n^2 h_w t_{acc}$. For flicker FM noise, the phase variance and therefore also the worst-case entropy keep reducing for increasing p, although the reduction is minor compared to the reduction for p from zero to one and reduces for higher p.

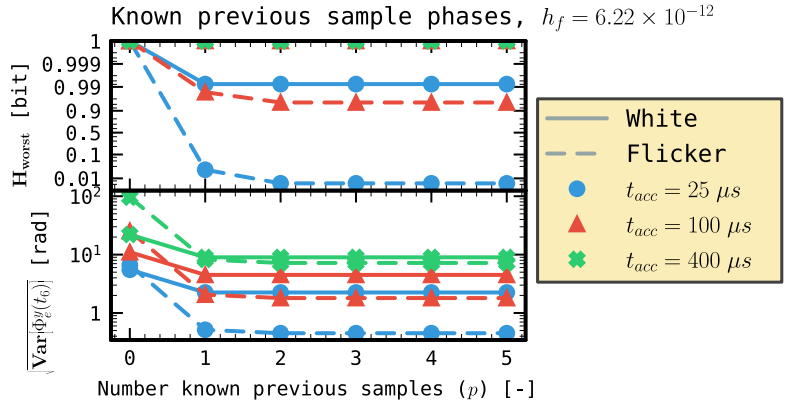

Figure 4.10: Worst-case Shannon entropy (top) and oscillator phase standard deviation (bottom), given the knowledge of p previous sample phase values, for $p \in \mathbb{N}_6$ and a flicker FM noise magnitude $h_f = 6.22 \times 10^{-12}$. Results are provided both for white FM and flicker FM noise and for accumulation lengths: $t_{acc} \in \{25.0\,\mu s, 100\,\mu s, 400\,\mu s\}$.

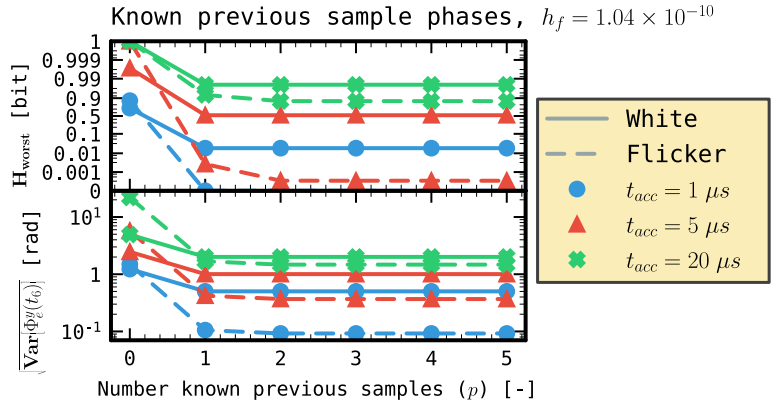

Figure 4.11: Worst-case Shannon entropy (top) and oscillator phase standard deviation (bottom), given the knowledge of p previous sample phase values, for $p \in \mathbb{N}_6$ and a flicker FM noise magnitude $h_f = 1.04 \times 10^{-10}$. Results are provided both for white FM and flicker FM noise and for accumulation lengths: $t_{acc} \in \{1.25\,\mu s, 5.00\,\mu s, 20.0\,\mu s\}$.

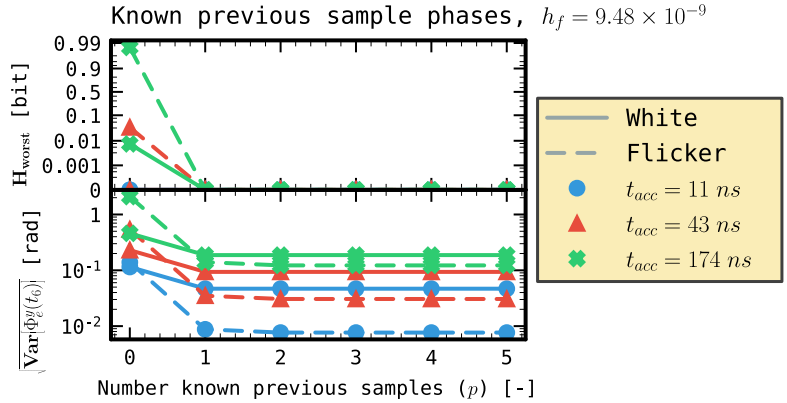

Figure 4.12: Worst-case Shannon entropy (top) and oscillator phase standard deviation (bottom), given the knowledge of p previous sample phase values, for $p \in \mathbb{N}_6$ and a flicker FM noise magnitude $h_f = 9.48 \times 10^{-9}$. Results are provided both for white FM and flicker FM noise and for accumulation lengths: $t_{acc} \in \{10.9\,\text{ns}, 43.4\,\text{ns}, 174\,\text{ns}\}$.

Knowledge of Only the Previous Sample Phase Elaborating on knowing only the phase of the previous sample ($p = 1$), fig. 4.13 depicts the worst-case Shannon entropy for white FM and flicker FM noise separately, versus the accumulation time between the samples. Curves are given for three time instances: t_6, t_{1000} and $t_{1000000}$, the sixth (given in previous paragraph), thousandth and millionth bit, respectively. The worst-case white FM noise Shannon entropy shown in fig. 4.13 is identical to the solid red curve from fig. 4.8.

As seen from this figure, given the knowledge of the previous sample's phase, the worst-case entropy for the flicker FM noise component is significantly higher, comparable or significantly lower than the worst-case entropy for the white FM noise component, when using the high, mid or low flicker FM noise magnitude estimate respectively from table 4.1. Additionally, increasing from the sixth to the millionth sampled bit, reduces the knowledge gained over the current sample, when observing the previous sample's phase value.

Knowledge of the Previous Bit Values

In this section, only the value for the previous p sampled bits instead of the full oscillator phase is assumed to be known. The entropy for the 300-th bit from an ERO-ES is calculated, given the knowledge of the previous p sampled bits, for p ranging from zero up to ten. When p equals zero, the entropy for the

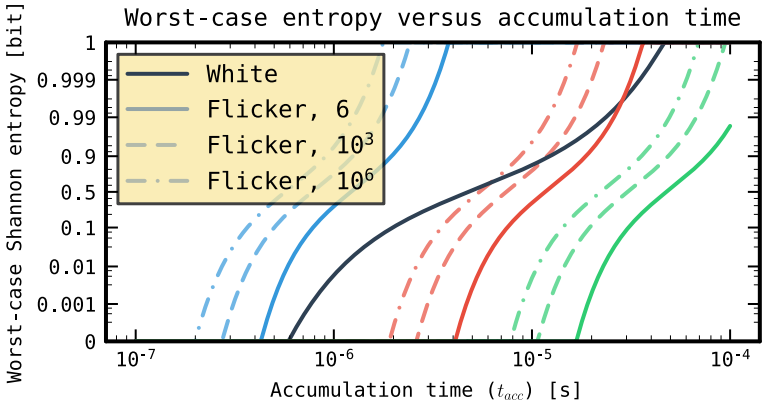

Figure 4.13: Worst-case flicker FM noise Shannon entropy, given the knowledge of the previous sample's phase value versus the accumulation time (t_{acc}) between two samples. Curves are plotted for three different flicker FM noise magnitudes: $h_f \in \{6.22 \times 10^{-12}, 1.04 \times 10^{-10}, 9.48 \times 10^{-9}\}$, a higher noise magnitude gives a higher entropy value. For each flicker FM noise magnitude, three curves corresponding to the sixth, thousandth and millionth bit are shown. The white FM noise entropy, from fig. 4.8, is shown for reference.

unconditioned distribution is given. The unobserved excess phase and observed bit vector become

$$\vec{\boldsymbol{\Phi}}_{\boldsymbol{e}}^{\boldsymbol{u}} = \left(\Phi_e^w(t_{300}), \Phi_e^f(t_{300})\right)^\intercal,$$

$$\boldsymbol{B}^{\vec{\boldsymbol{o}},\boldsymbol{p}} = (B_{299}, B_{298}, \ldots, B_{300-p})^\intercal.$$

Figures 4.14 to 4.16 show the worst-case Shannon entropy for the 300-th bit, given the knowledge of p previous sample bits, for flicker FM noise magnitudes 6.22×10^{-12}, 1.04×10^{-10} and 9.48×10^{-9}, respectively. The entropy values are given for white FM and flicker FM noise separately, $\mathbf{H_{worst}}\left[\Phi_e^y(t_{300}) \mid \boldsymbol{B}^{\vec{\boldsymbol{o}},\boldsymbol{p}}\right]$, for $y \in \{w, f\}$, respectively.

Figures 4.14 and 4.15 show high worst-case Shannon entropy values for both white FM and flicker FM noise. The entropy reduces slightly with increasing number of known bits, as each bit reveals some amount of information on the current oscillator phase. For the higher sampling speeds in fig. 4.16, the flicker FM worst-case Shannon entropy drastically reduces even when a single bit is known.

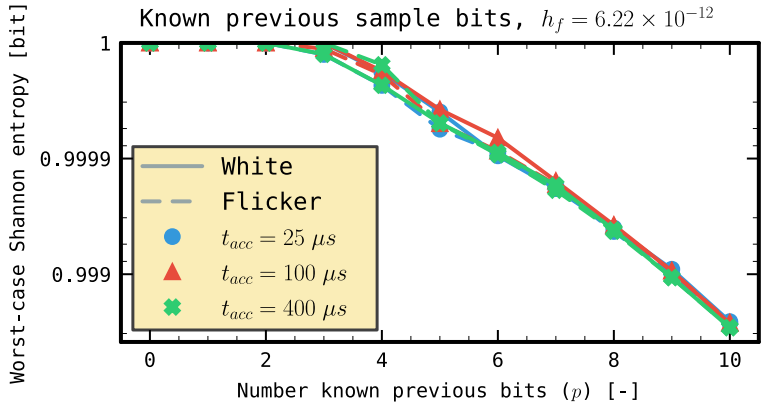

Figure 4.14: Worst-case Shannon entropy, given the knowledge of p previous sample bit values, for $p \in \mathbb{N}_{11}$ and a flicker FM noise magnitude: $h_f = 6.22 \times 10^{-12}$. Results are provided both for white FM and flicker FM noise and for accumulation lengths: $t_{acc} \in \{25.0\,\mu s, 100\,\mu s, 400\,\mu s\}$.

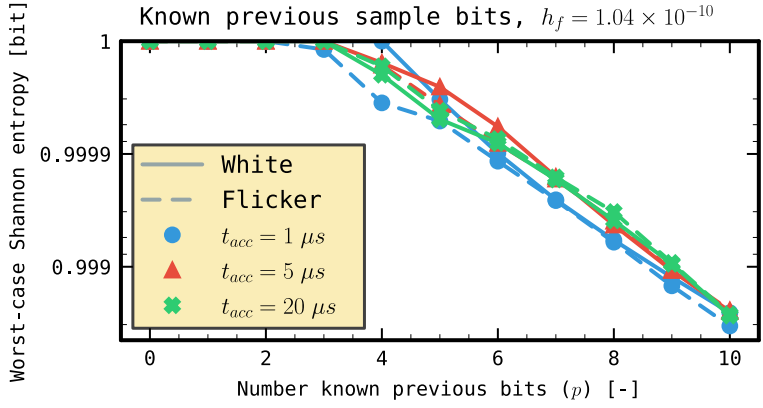

Figure 4.15: Worst-case Shannon entropy, given the knowledge of p previous sample bit values, for $p \in \mathbb{N}_{11}$ and a flicker FM noise magnitude: $h_f = 1.04 \times 10^{-10}$. Results are provided both for white FM and flicker FM noise and for accumulation lengths: $t_{acc} \in \{1.25\,\mu s, 5.00\,\mu s, 20.0\,\mu s\}$.

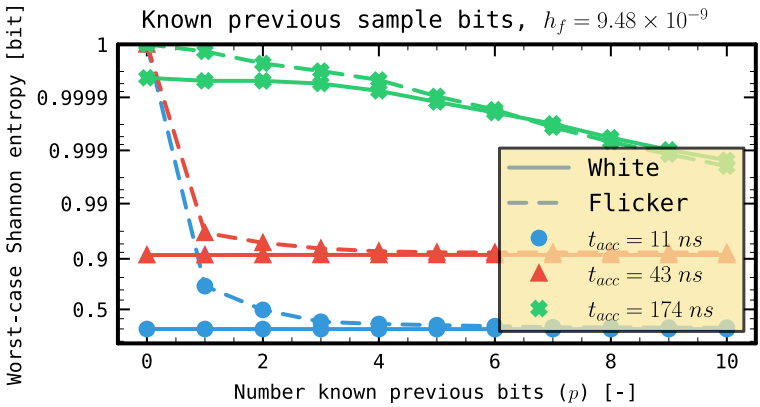

Figure 4.16: Worst-case Shannon entropy, given the knowledge of p previous sample bit values, for $p \in \mathbb{N}_{11}$ and a flicker FM noise magnitude: $h_f = 9.48 \times 10^{-9}$. Results are provided both for white FM and flicker FM noise and for accumulation lengths: $t_{acc} \in \{10.9\,\text{ns}, 43.4\,\text{ns}, 174\,\text{ns}\}$.

4.4.3 Summary of Simulation Findings

The simulation results reveal several key insights regarding the worst-case entropy generated by white FM and flicker FM noise in an ERO-ES. Firstly, the worst-case entropy decreases more significantly when previous phase values are known for flicker FM noise compared to white FM noise. This could mainly be attributed to the dependencies in consecutive period lengths caused by flicker FM noise. Additionally, at higher sampling speeds, flicker FM noise may appear to contain more entropy than it actually does. This overestimation of entropy is only detected when previous samples are also considered. Lastly, the h_f value plays a crucial role in determining the ratio of entropy contribution between white and flicker noise, making it impossible to assert that flicker noise always provides a meaningful contribution to the output entropy.

4.5 Conclusion and Further Research Directions

This chapter presented a method for modeling the excess phase process of a free-running oscillator. The time-domain model is based on the theory of Gaussian processes and is specifically tailored for use of estimating the entropy produced by an ES. The focus of this chapter was on the most prevalent noise types: white FM and flicker FM noise, but the proposed model could be applied

to noise sources with other spectral shapes (e.g. random walk FM noise as described by [32]) as well. For the two noise types, the ACF was analytically derived from the shape of the oscillator's relative frequency deviation spectrum.

Using Bayes' theorem, the conditional ERO-ES output bit distribution is analytically derived from the Gaussian process excess phase model. These distributions allow observing the change in phase PDF shape, when further knowledge on the ES state becomes available. Additionally, the entropy produced by the ES is derived from the obtained phase PDFs and the worst-case entropy concept was introduced to remove the deterministic influence of the phase offset on the derived entropy figure.

Finally, this chapter presents some exploratory simulation results for the proposed entropy model, using three different magnitudes for the flicker FM noise component, encountered in the literature. The results show that flicker FM noise can indeed in some cases be a valid source of TRNG entropy. However, due to the inherent long-lasting dependency, this noise should be harvested with great care. Given a low flicker FM noise magnitude, we conclude from fig. 4.13 that the flicker FM noise only bears minimal entropy compared to white FM noise at practical sampling speeds. Especially as there is a wide range of flicker FM noise estimates available in literature, ranging from $h_f = 9.48 \times 10^{-9}$ in [67], chapter 3 in this thesis, down to $h_f = 6.22 \times 10^{-12}$ in [27], more experimental evidence on potentially a wider range of platforms should become available before flicker FM noise could be widely accepted as a reliable source of TRNG entropy.

Besides from working on a more profound experimental validation of the flicker FM noise magnitude, we believe further research should be focused on applying the Gaussian process model on a more extended set of ES architectures, e.g. situations where multiple oscillators are used. Additionally, studying the stopping time, when a specified phase level is reached by the random process, is necessary to determine the distribution for the oscillator's period length, which in turn enables to augment existing ES stochastic models with the existence of flicker FM noise.

Appendix 4.A Combining Multiple Noise Sources

In terms of the relative phase acceleration, $\forall \omega \in \Omega, t \in \mathbb{R}_{\geq 0} : A(\omega, t) = \frac{\mathrm{d}}{\mathrm{d}t} Y(\omega, t)$, the relation from eq. (4.3) becomes $S_A(f) = (2\pi f)^2 S_Y(f) = \sum_{\alpha=-2}^{2} (2\pi f)^2 S_{Y^\alpha}(f) = \sum_{\alpha=-2}^{2} S_{A^\alpha}(f)$. The relative phase acceleration is assumed *stationary* in section 3.2.4, therefore $S_A(f) = \mathcal{F}\{R_A(\tau)\}$. Combine this

by using the linearity of the Fourier Transform (FT): $S_A(f) = \sum_{\alpha=-2}^{2} S_{A^\alpha}(f) = \sum_{\alpha=-2}^{2} \mathcal{F}\{R_{A^\alpha}(\tau)\} = \mathcal{F}\left\{\sum_{\alpha=-2}^{2} R_{A^\alpha}(\tau)\right\} = \mathcal{F}\{R_A(\tau)\}$, therefore, we have $R_A(\tau) = \sum_{\alpha=-2}^{2} R_{A^\alpha}(\tau)$, with $R_{A^\alpha}(\tau) = \mathcal{F}^{-1}\{(2\pi f)^2 S_{Y^\alpha}(f)\}$. The relative phase acceleration ACF is similarly composed of a sum of independent contributions.

Satisfying this relation, we assume the relative phase acceleration equals $\forall \omega \in \Omega, t \in \mathbb{R}_{\geq 0} : A(\omega, t) = \sum_{\alpha=-2}^{2} A^\alpha(\omega, t)$, with $\forall (t_i, t_j)^\intercal \in \mathbb{R}^2_{\geq 0} : R_{A^\alpha}(t_i, t_j) = \mathbf{E}\big[A^\alpha(t_i)A^\alpha(t_j)\big]$. Indeed, the ACF now equals

$$\begin{aligned} R_A(t_i, t_j) &= \mathbf{E}\big[A(t_i)A(t_j)\big] = \mathbf{E}\left[\sum_{\alpha_i=-2}^{2} A^{\alpha_i}(t_i) \sum_{\alpha_j=-2}^{2} A^{\alpha_j}(t_j)\right] \\ &= \sum_{\alpha_i=-2}^{2} \sum_{\alpha_j=-2}^{2} \mathbf{E}\big[A^{\alpha_i}(t_i)A^{\alpha_j}(t_j)\big] = \sum_{\alpha=-2}^{2} \mathbf{E}\big[A^{\alpha}(t_i)A^{\alpha}(t_j)\big] \\ &= \sum_{\alpha=-2}^{2} R_{A^\alpha}(t_i, t_j), \end{aligned}$$

using $\forall (t_i, t_j)^\intercal \in \mathbb{R}^2_{\geq 0}, \alpha_i \neq \alpha_j : \mathbf{E}\big[A^{\alpha_i}(t_i)A^{\alpha_j}(t_j)\big] = 0$, due to the mutual independence of the noise contributions.

We now define the individual excess phase noise contributions: $\Phi_e^\alpha : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ by $\Phi_e^\alpha(\omega, t) = 2\pi f_n \int_0^t \int_0^\theta A^\alpha(\omega, \nu)\mathrm{d}\nu\mathrm{d}\theta$, or equivalently by $\frac{\mathrm{d}^2}{\mathrm{d}t^2}\Phi_e^\alpha(\omega, t) = 2\pi f_n A^\alpha(\omega, t)$. The total excess phase then becomes

$$\begin{aligned} \Phi_e(\omega, t) &= 2\pi f_n \int_0^t \int_0^\theta A(\omega, \nu)\mathrm{d}\nu\mathrm{d}\theta = 2\pi f_n \int_0^t \int_0^\theta \sum_{\alpha=-2}^{2} A^\alpha(\omega, \nu)\mathrm{d}\nu\mathrm{d}\theta \\ &= \sum_{\alpha=-2}^{2} 2\pi f_n \int_0^t \int_0^\theta A^\alpha(\omega, \nu)\mathrm{d}\nu\mathrm{d}\theta = \sum_{\alpha=-2}^{2} \Phi_e^\alpha(\omega, t), \end{aligned}$$

with the excess phase noise contribution PSD equal to

$$S_{\Phi_e^\alpha}(f) = \frac{(2\pi f_n)^2}{(2\pi f)^4} S_{A^\alpha}(f) = \left(\frac{f_n}{f}\right)^2 S_{Y^\alpha}(f) = \left(\frac{f_n}{f}\right)^2 h_\alpha |f|^\alpha.$$

The total oscillator phase therefore equals a sum of independent noise contributions, added to a deterministic part, determined by the nominal frequency, f_n: $\Phi(\omega, t) = 2\pi f_n t + \phi_0 + \sum_{\alpha=-2}^{2} \Phi_e^\alpha(\omega, t)$.

Appendix 4.B Flicker FM Noise ACF

Changing the order of integration in eq. (4.5) and working out the integral obtains $\forall (t_i, t_j)^\intercal \in \mathbb{R}^2_{\geq 0}$:

$$\begin{aligned}
R_{\Phi_e}(t_i, t_j) &= 8\pi^2 f_n^2 h_f \int_{f_l}^{f_h} \frac{1}{f} \int_0^{t_i} \int_0^{t_j} \cos\big(2\pi f(\theta_j - \theta_i)\big) \mathrm{d}\theta_j \mathrm{d}\theta_i \mathrm{d}f \\
&= 4\pi f_n^2 h_f \int_{f_l}^{f_h} \frac{1}{f^2} \int_0^{t_i} \Big(\sin\big(2\pi f(t_j - \theta_i)\big) + \sin(2\pi f \theta_i)\Big) \mathrm{d}\theta_i \mathrm{d}f \\
&= 2 f_n^2 h_f \int_{f_l}^{f_h} \frac{1}{f^3} \Big(\cos\big(2\pi f(t_j - t_i)\big) - \cos(2\pi f t_j) \\
&\qquad - \cos(2\pi f t_i) + 1\Big) \mathrm{d}f.
\end{aligned}$$

When $t_i \neq t_j$, $t_i \neq 0$ and $t_j \neq 0$, the integral becomes

$$\begin{aligned}
R_{\Phi_e}(t_i, t_j) = 2 f_n^2 h_f \Bigg(& - \frac{\cos\big(2\pi f_h (t_j - t_i)\big)}{2 f_h^2} + \pi (t_j - t_i) \frac{\sin\big(2\pi f_h (t_j - t_i)\big)}{f_h} \\
& - 2\pi^2 (t_j - t_i)^2 \operatorname{Ci}\big(2\pi f_h |t_j - t_i|\big) - \frac{1}{2 f_h^2} \\
& + \frac{\cos\big(2\pi f_l (t_j - t_i)\big)}{2 f_l^2} - \pi (t_j - t_i) \frac{\sin\big(2\pi f_l (t_j - t_i)\big)}{f_l} \\
& + 2\pi^2 (t_j - t_i)^2 \operatorname{Ci}\big(2\pi f_l |t_j - t_i|\big) + \frac{1}{2 f_l^2} \\
& + \frac{\cos(2\pi f_h t_j)}{2 f_h^2} - \pi t_j \frac{\sin(2\pi f_h t_j)}{f_h} + 2\pi^2 t_j^2 \operatorname{Ci}(2\pi f_h t_j) \\
& - \frac{\cos(2\pi f_l t_j)}{2 f_l^2} + \pi t_j \frac{\sin(2\pi f_l t_j)}{f_l} - 2\pi^2 t_j^2 \operatorname{Ci}(2\pi f_l t_j) \\
& + \frac{\cos(2\pi f_h t_i)}{2 f_h^2} - \pi t_i \frac{\sin(2\pi f_h t_i)}{f_h} + 2\pi^2 t_i^2 \operatorname{Ci}(2\pi f_h t_i) \\
& - \frac{\cos(2\pi f_l t_i)}{2 f_l^2} + \pi t_i \frac{\sin(2\pi f_l t_i)}{f_l} - 2\pi^2 t_i^2 \operatorname{Ci}(2\pi f_l t_i) \Bigg),
\end{aligned} \tag{4.18}$$

with: Ci, the cosine integral, defined in appendix A. When either $t_i = t_j$, $t_i = 0$ or $t_j = 0$, the corresponding term simplifies to

$$-\frac{\cos(2\pi f_x t_y)}{2f_x^2} + \pi t_y \frac{\sin(2\pi f_x t_y)}{f_x} - 2\pi^2 t_y^2 \operatorname{Ci}(2\pi f_x t_y) = -\frac{1}{2f_x^2},$$

for $x \in \{l, h\}$ and $t_y \in \{t_i, t_j, |t_j - t_i|\}$.

Similar as in section 3.2.3, the upper frequency bound, f_h, is assumed very large: $f_h \to \infty$. Using the property of the cosine integral: $\lim_{x\to\infty} \operatorname{Ci}(x) = 0$, eq. (4.18) is simplified:

$$\begin{aligned} R_{\Phi_e}(t_i, t_j) = 2f_n^2 h_f \Bigg(& \frac{\cos\big(2\pi f_l(t_j - t_i)\big)}{2f_l^2} - \pi(t_j - t_i)\frac{\sin\big(2\pi f_l(t_j - t_i)\big)}{f_l} \\ & + 2\pi^2 (t_j - t_i)^2 \operatorname{Ci}\big(2\pi f_l |t_j - t_i|\big) + \frac{1}{2f_l^2} \\ & - \frac{\cos(2\pi f_l t_j)}{2f_l^2} + \pi t_j \frac{\sin(2\pi f_l t_j)}{f_l} - 2\pi^2 t_j^2 \operatorname{Ci}(2\pi f_l t_j) \\ & - \frac{\cos(2\pi f_l t_i)}{2f_l^2} + \pi t_i \frac{\sin(2\pi f_l t_i)}{f_l} - 2\pi^2 t_i^2 \operatorname{Ci}(2\pi f_l t_i) \Bigg). \end{aligned}$$

Reordering further to

$$\begin{aligned} R_{\Phi_e}(t_i, t_j) = {} & 4\pi^2 (t_j - t_i)^2 f_n^2 h_f \Bigg(-\frac{1}{2}\frac{\sin^2\big(\pi f_l(t_j - t_i)\big)}{\pi^2 f_l^2 (t_j - t_i)^2} - \frac{\sin\big(2\pi f_l(t_j - t_i)\big)}{2\pi f_l (t_j - t_i)} \\ & + \operatorname{Ci}\big(2\pi f_l |t_j - t_i|\big) + \frac{1}{4\pi^2 f_l^2 (t_j - t_i)^2} \Bigg) \\ & + 4\pi^2 t_j^2 f_n^2 h_f \Bigg(\frac{1}{2}\frac{\sin^2(\pi f_l t_j)}{\pi^2 f_l^2 t_j^2} + \frac{\sin(2\pi f_l t_j)}{2\pi f_l t_j} \\ & - \operatorname{Ci}(2\pi f_l t_j) - \frac{1}{4\pi^2 f_l^2 t_j^2} \Bigg) \\ & + 4\pi^2 t_i^2 f_n^2 h_f \Bigg(\frac{1}{2}\frac{\sin^2(\pi f_l t_i)}{\pi^2 f_l^2 t_i^2} + \frac{\sin(2\pi f_l t_i)}{2\pi f_l t_i} \\ & - \operatorname{Ci}(2\pi f_l t_i) - \frac{1}{4\pi^2 f_l^2 t_i^2} \Bigg) + \frac{1}{f_l^2} f_n^2 h_f. \end{aligned} \tag{4.19}$$

As in section 3.2.3, it is assumed that the lower frequency limit, f_l, is much smaller than the inverse of the observed time, $t_y \in \{t_i, t_j, |t_j - t_i|\} : \forall t_y \in \mathbb{R}_{>0} : f_l \ll \frac{1}{t_y}$. Therefore, $\pi f_l t_y \ll 1$. Using the property $\lim_{x\to 0} \frac{\sin(x)}{x} = 1$, and using the Taylor expansion for $\mathrm{Ci}(x)$ around $x = 0$, eq. (4.19) is further simplified for $t_i > 0$, $t_j > 0$ and $t_i \neq t_j$:

$$\begin{aligned} R_{\Phi_e}(t_i, t_j) = 4\pi^2 (t_j - t_i)^2 f_n^2 h_f \left(-\frac{3}{2} + \gamma + \ln\big(2\pi f_l |t_j - t_i|\big) \right) \\ + 4\pi^2 t_j^2 f_n^2 h_f \left(\frac{3}{2} - \gamma - \ln(2\pi f_l t_j) \right) \\ + 4\pi^2 t_i^2 f_n^2 h_f \left(\frac{3}{2} - \gamma - \ln(2\pi f_l t_i) \right). \end{aligned} \tag{4.20}$$

When $t_y = 0$ for $t_y \in \{t_i, t_j, |t_j - t_i|\}$, the corresponding term in eq. (4.20) reduces to zero: $4\pi^2 t_y^2 f_n^2 h_f \big(\frac{3}{2} - \gamma - \ln(2\pi f_l t_y)\big) = 0$. Equation (4.20) can then further be simplified to obtain eq. (4.6).

Part II

Model Capabilities

Tuning of Design Parameters

Chapter 5

Configurable ROs

This chapter is based on the following publications:

> **Design and Analysis of Configurable Ring Oscillators for True Random Number Generation Based on Coherent Sampling**
> Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
> *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2021

Contribution: *main author*.

> **An Energy and Area Efficient, All Digital Entropy Source Compatible with Modern Standards Based on Jitter Pipelining**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, 2022

Contribution: *main author*.

5.1 Background and Context

Engineers often rely on *digital* circuit techniques to build ESs because it simplifies integration within digital systems, enhances portability across technology nodes, between ASIC and FPGA platforms, and among different FPGA families and vendors. Additionally, digital logic benefits from further technology scaling. However, implementing ESs with only digital logic is challenging due to limited ES-specific resources and techniques available. Typically, these ESs are based

on either the *unpredictability* of *metastable* memory elements [17, 91] or the *timing jitter* in free-running oscillators [6, 86].

As discussed in section 2.4.1, TRNG designers must consider the feasibility of the ES on the given hardware platform (FPGA or ASIC), portability across different FPGA families and vendors, different ASIC technology nodes, *design constraints*, and *design effort*. Unlike most digital designs, ESs are typically not solely described using an HDL. Beyond the HDL description, many ESs require manual set-up or placement and routing *constraints*. For example, designs based on Self-Timed Ring (STR) oscillators [13] and DCs [73, 93] require placement constraints to be set up for each FPGA family, limiting their portability. Additionally, some ESs [86] do not work correctly at all locations on an FPGA, necessitating a search procedure for each individual device to find a suitable placement.

ASIC designs also require care to ensure the circuit operates as intended. The TERO-ES [96] requires a *dynamic tuning* mechanism to handle the inherent *frequency variability* of the manufactured oscillator circuits due to *process variations*. The tuning mechanism ensures a sufficiently small spread in realized *design parameters*, which in turn leads to a desirable yield when manufacturing the ES.

Chapter 6 will cover the design of a *configurable* COSO-ES, a popular ES due to its minimal chip area requirement and implementation using only digital logic. The COSO-ES generates two oscillating signals with similar periods, typically created using two identically designed ROs. However, process and interconnect delay variations make matching the periods of the two ROs challenging [70]. On an FPGA platforms, this necessitates a labor-intensive search procedure to find two well-matched ROs, which is impractical as it must be repeated for each individual device, even within the same FPGA family.

As briefly touched in section 2.4.1 and discussed in more detail in section 6.2.2, the *stochastic model* requires a small oscillation period difference (tens of picoseconds) to obtain a high *entropy density* at the ES output. Achieving this small period difference necessitates precise frequency tuning of clock signals. There are various ways to achieve this tuning, with the most straightforward being the use of PLLs on both FPGA and ASIC platforms. PLLs are often available as *primitive circuit elements* in FPGA devices and are designed for clock generation. A PLL enables the creation of multiple clock signals with a designer-chosen frequency relation, based on a reference input clock signal. This input can either come from an external source, such as a crystal oscillator, or be generated by another clock resource on the IC.

Another approach would be to generate the required clock signals using ROs.

The RO can be constructed solely utilizing the primitive elements that make up the combinatorial logic in the FPGA fabric: LUTs. This removes the dependency of the design on often-desired circuit blocks such as PLLs and simplifies the design effort, as LUTs are among the most abundant resources in most common FPGA families. Additionally, ROs can be made very small, comprising only a handful of logic gates.

Once implemented, the frequency of this RO is often fixed to a suboptimal value. Additionally, predicting this value precisely before programming the FPGA or fabricating the ASIC is challenging due to *unique* characteristics resulting from process variations in each device. Even within a single device, the location of the RO can significantly alter its frequency by up to 10 % [57]. This variability in RO frequency is utilized by RO-based Physical Unclonable Functions (PUFs) to provide a *device-unique fingerprint* [50].

To precisely control the frequency of the RO and achieve sufficient entropy density at the ES output, designers employ various techniques post-manufacturing or programming. These methods include dynamically adjusting the load capacitance and modifying the drive current strength of individual stages. While effective in ASIC designs, these approaches are impractical for FPGA systems due to limitations of the fixed fabric design. Consequently, FPGA designers resort to leveraging inherent variations in wiring and logic delay for RO frequency control.

This chapter is divided into two parts: section 5.2 discusses circuit techniques for tuning the RO frequency specifically for use in FPGAs, while section 5.3 focuses on the ASIC platform.

5.2 Configurable ROs for FPGAs

To provide the desired RO frequency tuning control, this section proposes the following three concepts of *delay variability* in FPGAs: *Gate delay* variability, *Wiring delay* variability, and *Intra-LUT delay* variability, embodied in the `GateVar`, `WireVar` and `LUTVar` RO, respectively. All three concepts facilitate highly portable, easy-to-use architectures suitable for constructing RO-based ESs without requiring specific device blocks such as PLLs, carry-chains, or Digital Signal Processors (DSPs). Moreover, these architectures eliminate the need for placement and routing constraints, as well as manual search procedures, and can be described solely using an HDL, narrowing the gap between the design of ESs and conventional digital logic.

5.2.1 Architecture

All three architectures form a single RO that has a configuration input. Altering the value assigned to the configuration input directly affects the oscillation frequency. The required area to implement the RO architectures on an FPGA, consisting of $n + 1$ stages, is shown in table 5.1.

Gate Delay Variability

Process variations in nanoscale CMOS technologies ensure that every transistor has unique characteristics. These variations manifest themselves in what is known as device mismatch in the analog design community [48]. Mismatch is the transistor parameter variation between identically designed transistors. It can be either systematic, due to e.g. a poor design layout, or caused by process variations. These device variations manifest themselves as variations in the LUT *propagation delay*.

In this chapter, we propose to use device variations to our advantage. Figure 5.1 shows the proposed `GateVar` configurable RO architecture, that uses the LUT propagation delay variability to generate a tuneable oscillating signal. Each column of four Multiplexers (MUXs) represents one RO stage. A select signal enables the choice of the output of one of the four MUXs that make up the previous stage. In this way, one MUX can be chosen from every column and a custom chain of MUXs can be selected through this network. Due to process variations, each chain has a unique propagation delay. An additional column containing only NAND gates is added to provide the necessary inversion and enable functionality. Hereby, an RO with $n + 1$ stages is obtained, where n represents the number of MUX columns. Every MUX stage enables for four different configurations (select one out of the four previous stage MUXs), which produces a total of 4^n number of delay configurations for an $n + 1$-stage RO.

Wire Delay Variability

Process variations affect not only transistor properties but also the interconnection circuitry. In FPGA devices, this includes both the wiring and the switching matrices. The wires interconnecting the logic elements on a chip function as a distributed network of resistors, capacitors, and inductors. Variations in these circuit elements occur due to factors like line edge roughness [45] or the materials used [89].

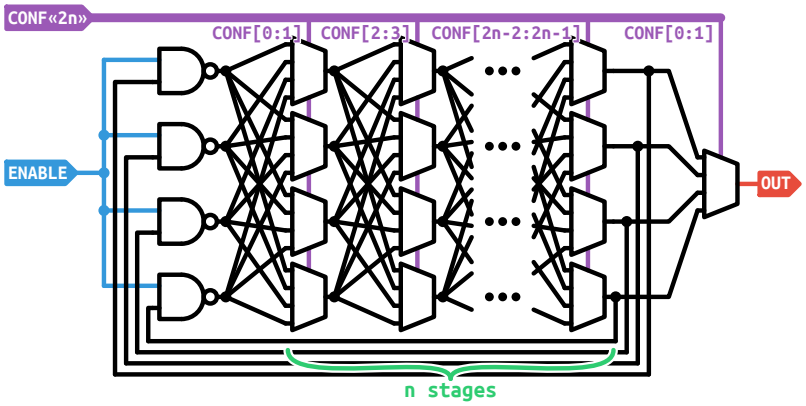

Figure 5.1: Architecture of a configurable RO (indicated as `GateVar` in the remainder of this thesis), using gate delay variability.

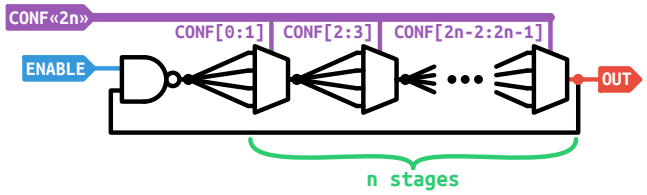

Figure 5.2: Architecture of a configurable RO (indicated as `WireVar` in the remainder of this thesis), using wire delay variability.

Figure 5.2 shows the proposed `WireVar` RO. Each MUX represents an RO stage and selects one of four wires originating from the previous stage's output. An additional NAND gate enables or disables the RO. By changing the select input signals of the individual MUXs, different wire combinations propagate the running edge, resulting in different RO frequencies. Each RO stage offers four wire options, yielding 4^n possible configurations for an $n+1$-stage RO.

Intra-LUT Delay Variability

Variations also exist within the internal structure of LUTs. Figure 5.3 shows the possible internal layout of a three-input LUT. The exact configuration of LUTs in the FPGA fabric is often proprietary, but the depicted structure, using CMOS transmission gates, is a common *assumption* [51], and a version of it was patented by Xilinx [71]. Adding more inputs to the LUT increases the depth of

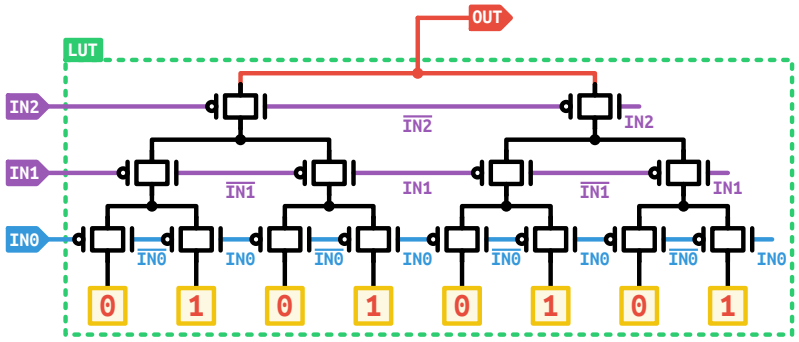

Figure 5.3: Possible internal structure of a three-input LUT.

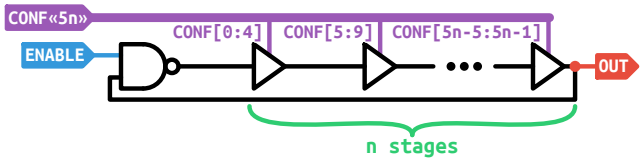

Figure 5.4: Architecture of a configurable RO (indicated as `LUTVar` in the remainder of this thesis), using internal LUT delay variability.

the transmission gate tree, necessitating additional buffers to maintain signal integrity. The shaded boxes represent Static Random-Access Memory (SRAM) cells storing the FPGA configuration.

In fig. 5.3, the LUT is configured to act as a buffer to input `in0`. Although inputs `in1` and `in2` do not affect the logical output value, they do influence the internal path selected to propagate the content of the SRAM cell to the output. As proposed by [51], these unused inputs can be used as select signals to fine-tune the LUT propagation delay.

The `LUTVar` RO, built from LUT buffers, is shown in fig. 5.4. For six-input LUTs, each LUT configured as a buffer has five select inputs and one data input, resulting in 32 configurations per stage and a total of 32^n configurations for an $n+1$-stage RO. A NAND gate is added to provide an inversion and an enable input to the RO. Additional flexibility is achieved by selecting which physical LUT input port serves as the data input. For a six-input LUT, there are six possible assignments for the data input, with the remaining five physical input ports used for select inputs.

Table 5.1: Detailed FPGA area breakdown.

	Spartan 7		SmartFusion2	
	[LUT]	[%LUT]	[LUT]	[%LUT]
`GateVar`	$4n+2$	$0.035^{(a)}$	$8n-1$	$0.083^{(b)}$
`WireVar`	$n+2$	$0.012^{(a)}$	$2n$	$0.022^{(b)}$
`LUTVar`	$n+2$	$0.012^{(a)}$	$2n$	$0.022^{(b)}$

(a) Utilized chip area proportion for four stages.
(b) Utilized chip area proportion for three stages.

5.2.2 Experimental Evaluation

All three proposed RO designs are implemented on FPGAs from two different vendors: a Xilinx Spartan 7 (with six-input LUTs) and a Microsemi SmartFusion2 (with four-input LUTs). Each four-input MUX, the NAND gates, and configurable `LUTVar` buffers fit into a six-input LUT on the Spartan 7. However, on the SmartFusion2, a four-input MUX no longer fits into a single LUT. To accommodate this, the design is adapted by redefining the primitives used (only FFs and LUTs) to match the different library naming conventions of each vendor. The synthesis tool then automatically constructs the four-input MUXs using multiple four-input LUTs on the SmartFusion2 FPGA.

RO frequency measurements are performed using a counter method. Both the RO output and the system clock are applied to a 16 bit asynchronous counter. These counters are sampled at regular intervals and sent to a PC for analysis. The number of configurable stages (n) in each experiment ranged from one to four, except for the congestion experiment, where only four-stage ROs were measured out. In the four-stage case, this results in $4^4 = 256$ configuration options for each `GateVar` and `WireVar` RO and $32^4 = 1\,048\,576$ configuration options for each `LUTVar` RO. To reduce measurement time, not every `LUTVar` configuration was tested; instead, a Linear-Feedback Shift Register (LFSR) generated $2^{15} = 32\,768$ configuration inputs. Table 5.2 provides basic statistics of the measured oscillation periods over all applied configuration input values for the different experiments performed and for ROs consisting of four stages.

The experimental results presented in this section aim to answer the following questions:

Question 1: Can the proposed configurable RO architectures produce a wide range of frequencies?

Question 2: Is searching for an optimal Global Placement (GP) inside the FPGA chip still necessary?

Question 3: Are Local Placement (LP) constraints still necessary?

Question 4: Can the proposed configurable RO architectures also work on other FPGAs?

Question 5: How many stages are necessary?

Question 6: What is the influence of the implementation strategy?

Question 7: What will happen if the FPGA routing becomes highly congested?

Note. A distinction is made between GP and LP constraints: GP constraints determine the absolute location of the circuit on the FPGA die, while LP constraints ensure relative *symmetry* of wiring and LUT placement.

Each question in the list above is answered in the following subsections. First, the two key RO performance metrics are defined, and the performance of the different physical input ports for the `LUTVar` architecture is compared.

Performance Metrics

To evaluate and compare the performance of various configurable RO designs, this thesis utilizes the concepts of RO *range* and RO *resolution*.

Definition 5.1. (Configurable RO range) *Range* is defined as the InterQuartile Range (IQR) of the measured oscillation period distribution when iterating over all select input values:

$$\text{range} = Q(0.75) - Q(0.25),$$

with $Q : \mathbb{R} \to \mathbb{R}$, representing the quantile function (inverse cumulative distribution function) of the measured RO oscillation period distribution. The *normalized range* is calculated by dividing the range by the median oscillation period length.

Definition 5.2. (Configurable RO resolution) *Resolution* is determined by sorting the measured oscillation periods for all select input values and then calculating the median distance between these sorted period lengths.

To function effectively as an ES in TRNGs, it is desirable for the oscillator to have a wide range and a small resolution value.

Table 5.2: Measured RO (four stages) statistics on FPGA.

Experiment	RO type	Period mean [ns]	Frequency mean [MHz]	Period standard deviation [ns]
Placement constraints	LUTVar0	4.347	230.1	0.015 16
	LUTVar5	2.274	439.7	0.000 361 0
	WireVar	3.721	272.0	0.3992
	GateVar	3.733	270.8	0.3861
No GP or LP	LUTVar0	5.012	199.5	0.018 69
	LUTVar5	2.844	351.6	0.003 573
	WireVar	4.090	248.6	0.5135
	GateVar	3.551	285.9	0.4268
SmartFusion2	LUTVar0	5.430	184.2	0.051 95
	LUTVar3	5.523	181.2	0.1041
	WireVar	6.553	165.2	1.363
	GateVar	6.918	150.3	1.183
Implementation strategy	LUTVar0	4.971	201.2	0.017 78
	LUTVar5	2.478	403.6	0.004 441
	WireVar	4.004	253.5	0.4856
	GateVar	3.730	270.9	0.3731
Congestion	LUTVar0	5.707	175.2	0.020 52
	LUTVar5	3.195	313.0	0.002 635
	WireVar	4.242	244.8	0.7960
	GateVar	6.221	161.9	0.5252

LUTVar Physical Input Port Comparison

Figures 5.5 and 5.6 display the range and resolution for LUTVar ROs with one to four stages and physical input port numbers from zero to five on a Spartan 7 (using six-input LUTs). The figures show that the normalized range tends to slightly decrease with more stages and higher physical input port numbers. Conversely, the resolution improves with both an increasing number of stages and higher physical input port numbers.

Note. The measured resolution for a four-stage LUTVar RO was minimal (less than 0.1 ps), making accurate measurement difficult.

The initial experiments demonstrated that signal physical input ports zero and five represent the extremes in terms of normalized range and resolution. To save time in the subsequent experiments, only ports zero and five are tested,

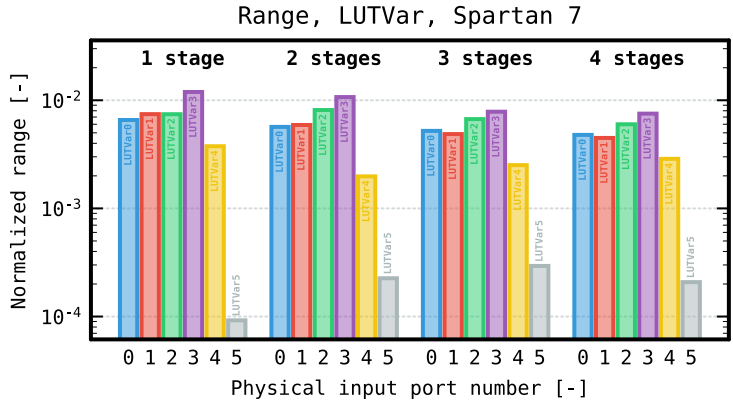

Figure 5.5: Normalized range comparison of `LUTVar` ROs with physical data input port ranging from zero to five, and number of stages ranging from one to four on a Spartan 7 FPGA.

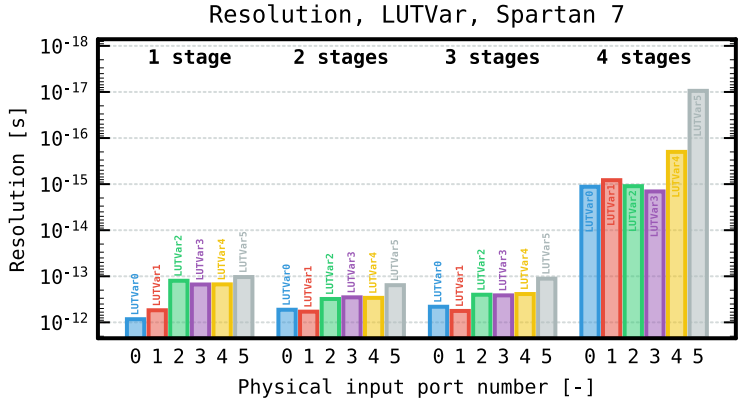

Figure 5.6: Resolution comparison of `LUTVar` ROs with physical data input port ranging from zero to five, and number of stages ranging from one to four on a Spartan 7 FPGA.

assuming the other ports fall within these extremes.

Fixed LP, Variable GP

To address questions 1, 2 and 5 on pages 97 to 98, we first implemented all three proposed RO architectures using manual placement on a Spartan 7 FPGA. The

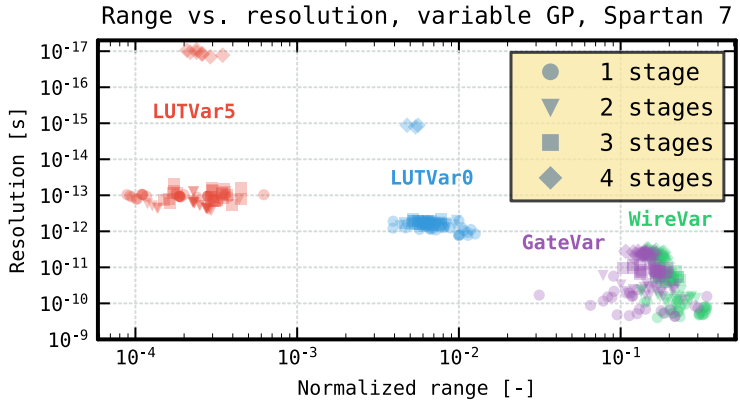

Figure 5.7: Normalized range versus resolution scatter plot for all three RO architectures, for the number of stages ranging from one to four. Each dot presents one out of 25 locations, uniformly selected over a Spartan 7 FPGA die.

LUTs containing the RO stages were placed symmetrically. Figure 5.7 presents a scatter plot of normalized range versus resolution, obtained by sweeping all the configurations. Each dot represents one of 25 test locations uniformly distributed across the FPGA die. The shapes of the dots indicate the number of RO stages, ranging from one to four.

As observed from fig. 5.7, all three RO architectures achieve a resolution smaller than 1 ns. The `WireVar` and `GateVar` architectures achieve a normalized range exceeding 10 %. Although the `LUTVar` architectures provide a finer resolution, their normalized range is smaller compared to the `WireVar` and `GateVar` designs. GP has minimal impact on the RO performance for all three designs, with the `LUTVar` ROs achieving the best resolution with four stages.

No Placement Constraints

To answer question 3 on page 98, the previous experiment was repeated on a Spartan 7 FPGA, this time without applying any LP and GP constraints. For all three architectures, the ROs are described by solely using an HDL.

Figures 5.8 and 5.9 display the normalized range and the resolution for all three RO architectures. Despite the absence of GP and LP constraints, the `WireVar` and `GateVar` architectures achieved a resolution finer than 0.1 ns. The `LUTVar` exhibited an even better resolution. For the `WireVar` and `GateVar` ROs

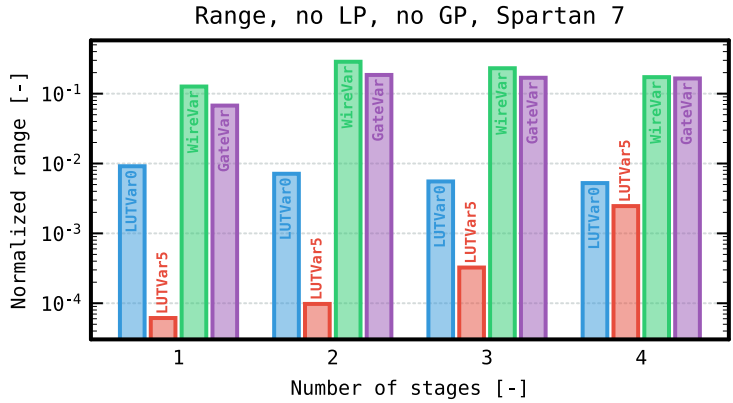

Figure 5.8: Normalized range versus number of stages plot for all three RO architectures, for the number of stages ranging from one to four. No placement constraints are applied to the ROs on a Spartan 7 FPGA.

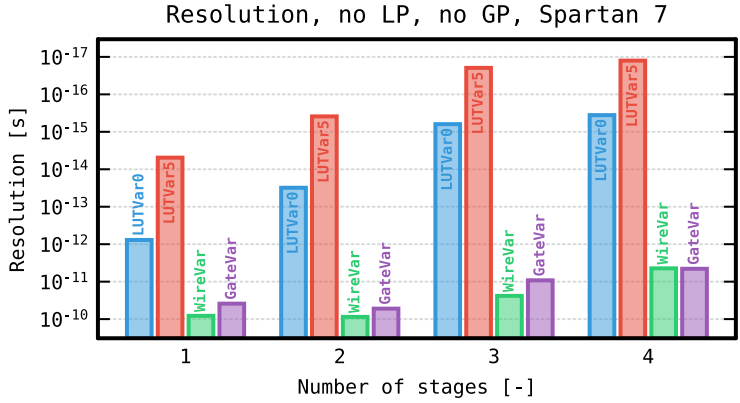

Figure 5.9: Resolution versus number of stages plot for all three RO architectures, for the number of stages ranging from one to four. No placement constraints are applied to the ROs on a Spartan 7 FPGA.

with two or more stages, a normalized range exceeding 10 % was attained. The `LUTVar` continued to show a relatively narrow normalized range.

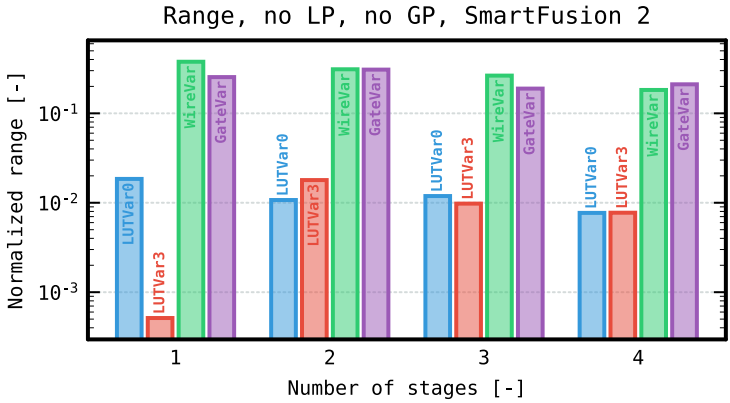

Figure 5.10: Normalized range versus number of stages plot for all three RO architectures, for the number of stages ranging from one to four. No placement constraints are applied to the ROs on a SmartFusion2 FPGA.

FPGA Vendor Portability

Focussing on question 4 on page 98, the previous experiment (with no GP or LP constraints) was repeated on a Microsemi SmartFusion2 FPGA. The `LUTVar` ROs used four-input LUTs and signal physical input ports zero and three were considered.

Figures 5.10 and 5.11 present the measured normalized range and resolution, respectively, which are comparable to the results on the Spartan 7 FPGA. These findings demonstrate the portability of all three proposed RO architectures.

Implementation Strategy

For question 6 on page 98, the experiment was repeated on a Spartan 7 FPGA, using the Area Explore implementation strategy instead of the default strategy in the Xilinx Vivado design tool. This strategy performs multiple optimization runs to minimize circuit area [92]. The HDL implementation of the RO circuit remained unchanged.

From the experimental results shown in fig. 5.12, we conclude that the implementation strategy choice minimally impacts the performance of all three proposed RO designs. The range-resolution curves obtained with the default strategy (dashed curve) closely match those for the Area Explore strategy (solid curve) across ROs with one to four stages.

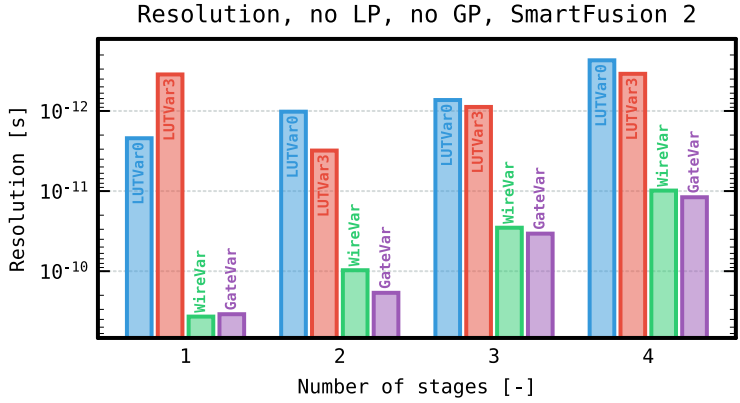

Figure 5.11: Resolution versus number of stages plot for all three RO architectures, for the number of stages ranging from one to four. No placement constraints are applied to the ROs on a SmartFusion2 FPGA.

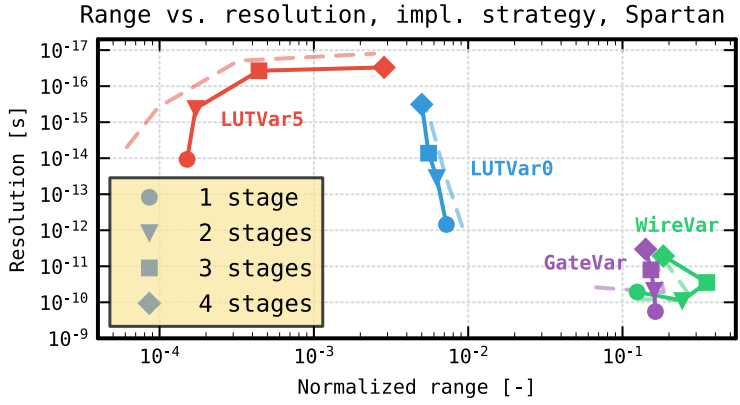

Figure 5.12: Normalized range versus resolution plot for all three RO architectures, for the number of stages ranging from one to four on a Spartan 7 FPGA. The solid lines represent the Area Explore implementation strategy in the Xilinx Vivado design tool. The dashed lines represent the default strategy (equal to the data from the omitted placement constraints experiment).

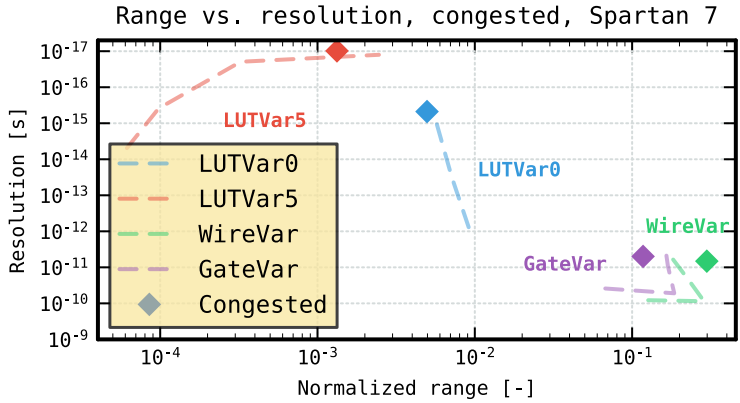

Figure 5.13: Normalized range versus resolution plot for all three RO architectures, with only four stages on a congested Spartan 7 FPGA. The dashed lines represent a non-congested FPGA (equal to the data from the omitted placement constraints experiment).

High Resource Utilization

Finally, to address question 7 on page 98, we augmented the FPGA utilization to assess the robustness of each RO architecture against high routing congestion. This was achieved by implementing a substantial number of 64-bit adder structures alongside the RO circuitry, resulting in 96 % and 72 % utilization of available LUTs and FFs, respectively. The rationale behind this approach is that increased congestion would force the place and route tools to lengthen the physical distance (and thus routing delay) between the various components constituting the RO circuitry.

The outcomes of this experiment are depicted in fig. 5.13, illustrating the range-resolution points for the different RO architectures. Only four stage-ROs have been considered to reduce measurement time. Remarkably, all three RO architectures exhibit comparable performance to the non-congested FPGA results (dashed curve).

Note. While fig. 5.13 implies that the degree of congestion has minimal impact on RO performance, table 5.2 highlights a notable reduction in the RO frequency for the *GateVar* architecture.

Table 5.3: Summary of FPGA experimental results.

Metric	`GateVar`	`WireVar`	`LUTVar`
Range	⊕	⊕⊕	⊖⊖
Resolution	⊕⊕	⊕	⊕⊕⊕
Omit GP	⊕⊕	⊕⊕	⊖
Omit LP	⊕⊕	⊕⊕	⊖
Portable	⊕⊕	⊕⊕	⊖⊖
Implementation robustness	⊕	⊕	⊖⊖
Congestion resilience	⊕	⊕⊕	⊖

5.2.3 FPGA Conclusion

Since all the questions posed on pages 97 to 98 have been positively answered in this section, we conclude that the three proposed RO architectures provide a robust foundation for constructing reliable ESs on FPGA platforms.

However, not all three presented RO architectures performed equally well. Both the `WireVar` and `GateVar` topologies exhibit a larger normalized range across all experiments conducted in this section compared to the `LUTVar` topology. On the other hand, the `LUTVar` topology achieves a finer resolution overall compared to the `WireVar` and `GateVar` architectures. As will be demonstrated in chapter 6, the advantage of the `WireVar` and `GateVar` architectures having a larger range outweighs the finer resolution achieved by the `LUTVar` topology when implementing a COSO-ES. In conclusion, table 5.3 summarizes the FPGA experimental results, comparing the strengths and weaknesses of each RO topology.

5.3 Configurable ROs for ASICs

Frequency tuning of CMOS ROs in ASICs is a well-developed field of research and is significantly more established in ASICs than in FPGAs. Various frequency-controlling techniques have been proposed, largely due to the greater design freedom available in ASICs compared to FPGAs. Some techniques utilize a variable capacitive load on each RO-stage [76], while others employ a variable resistor between consecutive stages using a transmission gate structure [98]. Another approach involves limiting the RO stage's drive current [33]. This brief summary is far from exhaustive.

For the purpose constructing ESs, the proposed configurable RO designs in this thesis must consider the following constraints:

- *Digitally controllable*: Eliminate the requirement for a Digital-to-Analog Converter (DAC) to convert the digital control signal into an analog controlling voltage.
- *Transistor only*: The RO should be constructed exclusively using MOS transistors, without the need for capacitors, resistors, or inductors.
- *Standard cell compatible*: The RO layout should be compatible with other digital circuitry by using a row-based design similar to standard cells.

5.3.1 Architecture

This thesis describes two configurable RO styles specifically designed for use with CMOS technologies. In both architectures, a digital configuration input directly influences the oscillation frequency by modifying the stage's current drive strength, either linearly or exponentially. The area required to implement the RO architectures on three different ASIC technologies is presented in table 5.4.

Current-Starved Inverter

The central building block for both described RO architectures is the Current-Starved Inverter (CSI), as described by [72]. As shown in fig. 5.14 (bottom left), a configuration input, `conf`, controls the drive current for this inverter stage. When the `conf` signal is low, both transistors `M0` and `M3` are off, resulting in a reduced drive strength for the gate. When the `conf` signal is high, transistors `M1` and `M2` create a typical inverter pair.

By scaling the width of the transistors, a CSI stage with increased current drive strength can be achieved, as illustrated in fig. 5.14 (middle). Conversely, by chaining multiple minimal-sized transistors, as shown in fig. 5.14 (right), a CSI with reduced drive strength is obtained. The symbol representing a CSI of size n, as used throughout this thesis, is shown in fig. 5.14 (top left).

Linear Drive Strength Increase

Utilizing the CSI, an RO can be constructed where the drive strength increases linearly with the applied configuration value. Figure 5.15 illustrates the principle of combining multiple CSIs in parallel, along with an optional inverter that

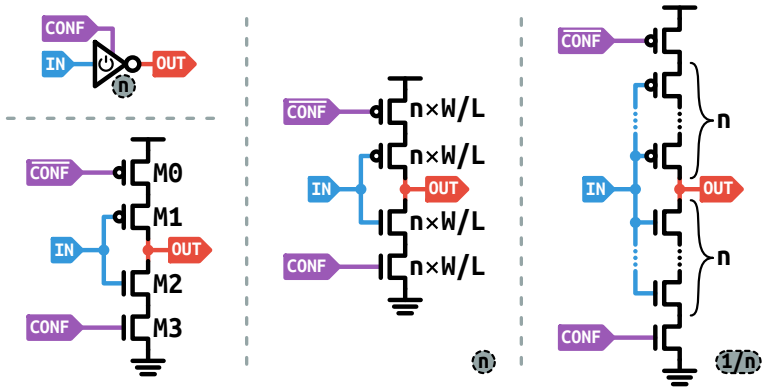

Figure 5.14: The symbol for a CSI of size n (top left), the CSI structure (bottom left), a CSI of size n (middle) and a CSI of size $\frac{1}{n}$ (right).

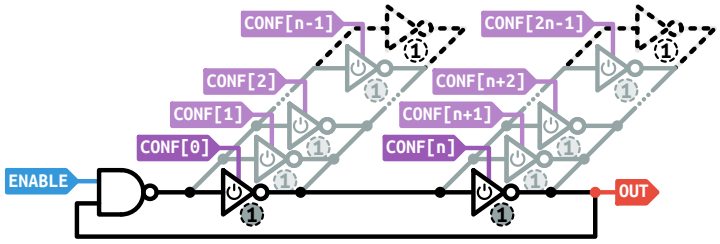

Figure 5.15: An example of a configurable RO architecture with a linear increase in drive strength.

ensures the RO oscillates when the all-zero configuration input is applied. Each parallel CSI is identical in size, ensuring that each bit of the `conf` signal has a consistent impact on the resulting oscillation frequency. The realized oscillation frequency primarily depends on the Hamming weight of the applied `conf` signal.

Throughout the rest of this thesis, this RO will be referred to as `LinX×Y`, where `X` denotes the number of stages and `Y` denotes the number of parallel CSIs per RO stage.

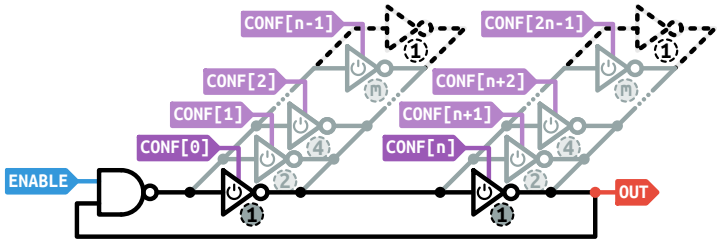

Figure 5.16: An example of a configurable RO architecture with an exponential increase in drive strength.

Table 5.4: Detailed ASIC area breakdown.

Technology	RO type	Transistor pairs [-]	Normalized area [kF]	Area [µm^2]
65 nm	`Lin2×4`	20	3.980	16.817
40 nm	`Lin2×8`	34	7.335	11.736
	`Exp4×4`	430	88.754	142.006
28 nm	`Lin2×8`	36	11.733	9.199
	`Exp4×4`	142	29.484	23.115

Exponential Drive Strength Increase

Instead of instantiating each CSI with equal size, each bit of the `conf` signal can be applied to a CSI of varying size. These sizes can be chosen exponentially, as shown by fig. 5.16. This approach makes the oscillation frequency predominantly dependent on the configuration value applied to the RO. An increasing configuration value should theoretically result in a higher oscillation frequency. However, due to *manufacturing variations*, a smaller configuration value with a larger Hamming weight (e.g., a value of 7) might produce a higher oscillation frequency than a larger configuration value with a smaller Hamming weight (e.g., a value of 8).

This type of configurable RO will be denoted as `ExpX×Y`, where `X` represents the number of stages and `Y` indicates the number of parallel CSIs per RO stage.

Table 5.5: Measured RO statistics on ASIC.

Technology	RO type	Period mean [ps]	Frequency mean [GHz]	Period standard deviation [ps]
65 nm	`Lin2×4`	579.5	1.813	80.37
40 nm	`Lin2×8`	206.6	5.192	49.07
	`Exp4×4`	715.8	1.698	462.5
28 nm	`Lin2×8`	187.3	5.613	33.93
	`Exp4×4`	2809	0.4332	1498

5.3.2 Experimental Evaluation

The described RO types have been implemented using three different CMOS technologies: 65 nm, 40 nm and 28 nm. In each technology except for the 65 nm technology, both the linear and exponential RO types have been implemented.

RO frequency measurements were conducted by downscaling the RO output frequency using an on-chip frequency scaler and analyzing the scaled oscillating signal off-chip with an oscilloscope. In the case of the 65 nm technology, on-chip frequency counters were also employed. Table 5.5 presents basic statistics of the measured oscillation periods across all applied configuration input values for the various CMOS technologies and RO types investigated.

65 nm Technology

We implemented a linear configurable RO architecture with two stages and four configurable CSIs per stage, denoted as `Lin2×4`. Additionally, each RO stage includes one inverter in parallel, which remains active irrespective of the `conf` signal value. Three sample chips have been measured out.

Figure 5.17 illustrates the median oscillation period length for each of the eight bits in the `conf` input signal. Each bit is represented by a vertical line, with the bottom dot for the bit set to zero and a top dot for the bit set to one. Additionally, dashed horizontal lines depict the median RO period length obtained when sweeping across all values of the `conf` signal. The vertical lines are determined by varying all other bits in the `conf` signal. As observed in fig. 5.17, each bit in the `conf` signal exerts a similar influence on the RO oscillation period length.

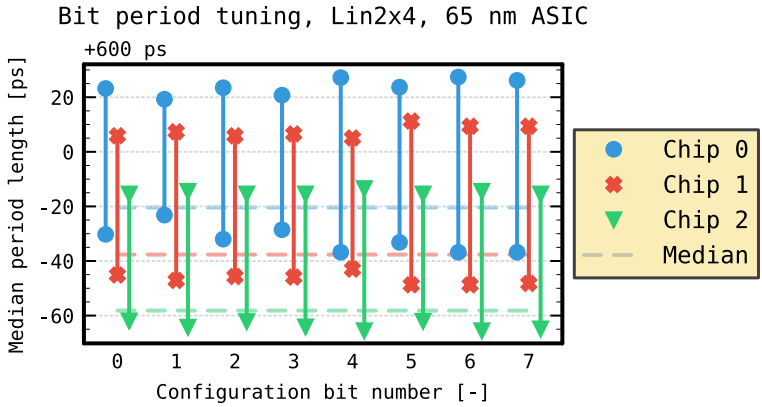

Figure 5.17: Configuration bit influence on the RO period length for the `Lin2×4` architecture in a 65 nm ASIC technology.

40 nm Technology

Two types of ROs were implemented: a linear architecture with two stages and eight CSIs per stage, denoted as `Lin2×8`, and an exponential architecture with four stages, each containing four CSIs and one regular inverter of size 1, denoted as `Exp4×4`. The four parallel CSIs in each stage of the `Exp4×4` RO are sized according to the sequence 1, 4, 16, and 32. Measurements were taken from two sample chips.

The influence of the individual bits of the `conf` signal is shown in figs. 5.18 and 5.19 for the `Lin2×8` and `Exp4×4` designs, respectively. The `Lin2×8` architecture performs similarly to the `Lin2×4` presented in the previous section. In the `Exp4×4` RO, the bits connected to the larger CSIs have a greater influence on the oscillation period than the bits connected to the smallest CSIs.

28 nm Technology

Similar to the previous subsection, two ROs were implemented: a linear architecture with two stages, each stage having eight CSIs and one regular inverter, denoted as `Lin2×8` and an exponential architecture with four stages, each stage having four CSIs and one regular inverter of size $\frac{1}{16}$, denoted as `Exp4×4`. The four parallel CSIs in each stage of the `Exp4×4` RO are sized according to the sequence: $\frac{1}{8}$, $\frac{1}{4}$, $\frac{1}{2}$ and 1. Both ROs include a regular inverter in parallel for each stage to ensure oscillation when the all-zero configuration input is applied. Measurements were taken from three sample chips.

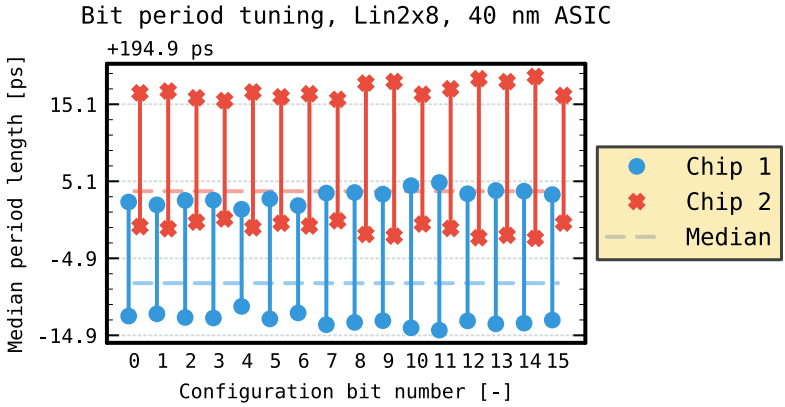

Figure 5.18: Configuration bit influence on the RO period length for the `Lin2×8` architecture in a 40 nm ASIC technology.

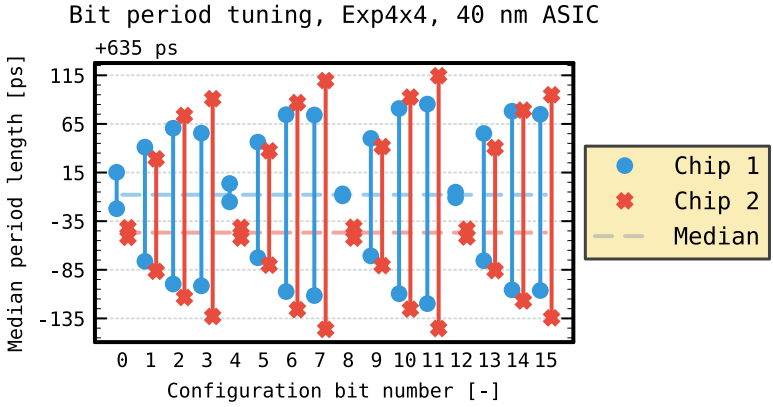

Figure 5.19: Configuration bit influence on the RO period length for the `Exp4×4` architecture in a 40 nm ASIC technology.

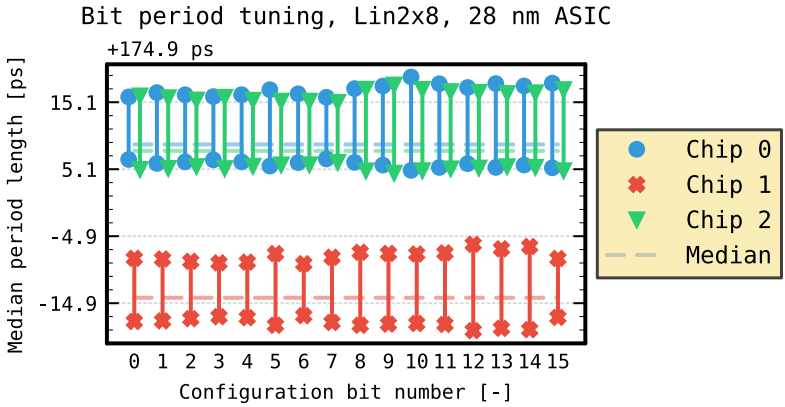

Figure 5.20: Configuration bit influence on the RO period length for the `Lin2×8` architecture in a 28 nm ASIC technology.

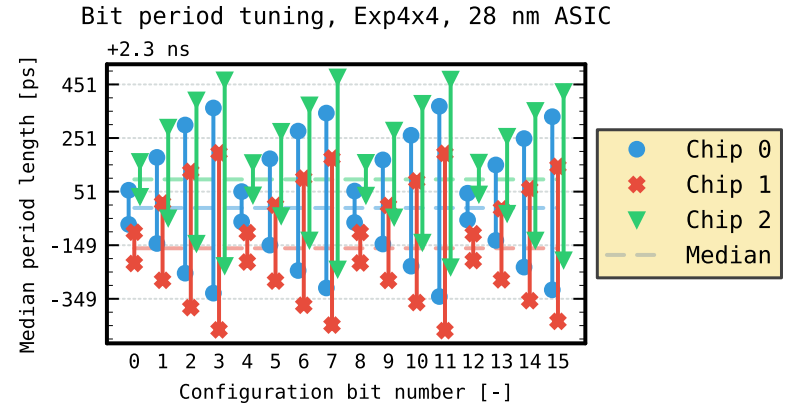

Figure 5.21: Configuration bit influence on the RO period length for the `Exp4×4` architecture in a 28 nm ASIC technology.

As shown in figs. 5.20 and 5.21, which present the measurement results for the `Lin2×8` and `Exp4×4` architectures respectively, both architectures perform similarly to those described in the previous subsections.

5.3.3 ASIC Conclusion

Figure 5.22 summarizes the ASIC configurable RO measurements by plotting the normalized range versus the obtained resolution for all RO architectures

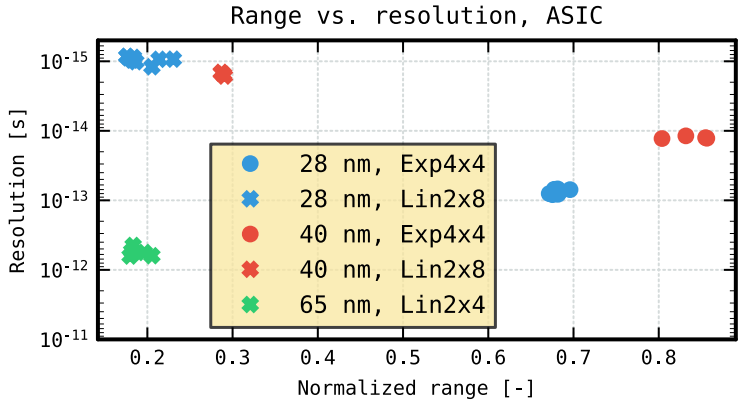

Figure 5.22: Normalized range versus resolution plot for all three ASIC technologies tested.

and CMOS technologies discussed in this section. Different dots in the figure represent different measured chips. Both the `Exp4×4` architectures in the 28 nm and 40 nm technologies exhibit a large normalized range of over 65 %. The finest resolution is achieved with the `Lin2×8` architectures in the 28 nm and 40 nm technologies. While having a similar normalized range of around 20 %, the `Lin2×4` architecture in the 65 nm technology produces a less fine resolution compared to the other linear architectures, primarily due to the smaller number of configuration input bits (8 bit compared to 16 bit).

5.4 Conclusion

This chapter demonstrates that highly tunable RO designs can be achieved on both FPGA and ASIC platforms. We achieved normalized range values of up to 10 % on FPGA and up to 80 % on ASIC, showcasing the controllability of the designs. Additionally, resolutions finer than 1 ps were obtained across all ASIC technologies and FPGA vendors tested. The presented architectures utilize only digital control signals, ensuring simplicity and area efficiency. Furthermore, their high portability is asserted by the exclusive use of basic circuit components: LUTs for FPGA and CMOS transistors for ASIC.

Future research should concentrate on examining the jitter properties of the RO architectures discussed in this chapter to further improve their reliability and assist in selecting the optimal RO topology based on the specific application requirements.

Chapter 6

Configurable TRNGs for FPGAs

This chapter is based on the following publications:

> **A Highly-Portable True Random Number Generator Based on Coherent Sampling**
> Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
> *International Conference on Field Programmable Logic and Applications (FPL)*, 2019

Contribution: *main author*.

> **Design and Analysis of Configurable Ring Oscillators for True Random Number Generation Based on Coherent Sampling**
> Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
> *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, 2021

Contribution: *main author*.

6.1 Background and Context

Implementing TRNGs on FPGAs is becoming increasingly popular, due to the FPGA's easy reconfigurability and significantly shorter design turnaround time compared to ASIC platforms. As discussed in section 2.4.1, ensuring good performance and robustness often requires a complex implementation procedure

for many TRNG designs, frequently involving manual placement and routing. In this chapter, we implement, analyze, and compare the three *dynamic* RO calibration mechanisms introduced in section 5.2, embedded in a COSO-based ES.

The ES set-up procedure automatically selects a *configuration* that meets security requirements. In our experiments, we demonstrate that two of the three proposed mechanisms can ensure correct ES operation even with automatic placement and when porting the design to another FPGA family. We generated random bits on both a Xilinx Spartan 7 FPGA and a Microsemi SmartFusion2 FPGA, achieving *throughputs* of 4.65 Mbit s^{-1} and 1.47 Mbit s^{-1}, respectively. Without post-processing, the generated bits passed the AIS 31 statistical tests.

As discussed in section 2.2, the output of a TRNG should pass statistical tests such as NIST SP 800-22 [74], NIST SP 800-90B [83], or FIPS 140-2 [58]. However, merely passing these statistical tests is insufficient to guarantee the *security* of a TRNG. According to the AIS 31 [39] and NIST SP 800-90B [83] recommendations, the security of a TRNG should preferably be justified by a *stochastic model* of the ES.

This chapter focuses on the COSO-based ES, originally proposed by [43]. The stochastic model for this ES was first developed by [9] for PLL-based implementations. A more general stochastic model, proposed by [95], does not specify the source of the *random jitter*, as it is based on the period difference between two oscillators of any kind.

The COSO-ES combines substantial throughput with minimal area requirement, making it an attractive option for an FPGA-compatible ES architecture. Existing COSO-ES implementations can achieve a throughput on the order of 1 Mbit s^{-1} [70], but they require an extensive *design effort*, a term introduced in section 2.4.1. The ES generates two oscillating signals with similar periods, typically using two identically designed ROs. As noted in section 5.1, matching RO periods on FPGA platforms is particularly challenging due to *process* and interconnect delay *variations* [70]. This difficulty necessitates a significant design effort, involving a search procedure to find two well-matched ROs, making the COSO-ES implementation impractical as this procedure must be repeated for every device, even within the same FPGA family.

In this chapter, we propose a highly portable and user-friendly architecture for a COSO-ES that does not require any device-specific blocks such as PLLs, carry-chains, or DSPs. Additionally, it demands no placement and routing *constraints* and eliminates the need for a manual search procedure. The main contributions are as follows:

- We propose a new ES featuring reconfigurable ROs that enable matching two oscillating periods with precision in the range of a few picoseconds. This ES utilizes only LUTs and FFs, avoiding the need for device-specific resources and ensuring portability across different FPGA families and vendors.
- We experimentally demonstrate that precise matching is achievable without placement and routing constraints. The Verilog source code for this ES is open source and publicly available[1].
- We have developed a control circuit that monitors the health of the ES, dynamically adjusts matching conditions, and notifies the user if satisfactory matching cannot be achieved.

This chapter is structured as follows: in section 6.2, we introduce the working principles and stochastic model of the COSO-ES. Section 6.3 details the architecture of our novel ES. Experimental validation of its functionality is provided in section 6.4. Finally, section 6.5 offers a discussion on the results and compares performance with existing literature. The chapter concludes with a summary and outlines future directions in section 6.6.

6.2 Coherent Sampling

The COSO designs incorporate two oscillators within the ES. These oscillators' periods must satisfy one of two conditions: either their ratio matches a known rational fraction, typical in PLL-based designs, or the ratio is finely *tuned* to approach unity, as is the case with ROs. Here, our focus is on the latter scenario to avoid dependency on device-specific components like PLLs.

6.2.1 Stochastic Model

Figure 6.1 depicts the architecture of the COSO-ES. A DFF is employed for sampling, which generates a low-frequency beat signal labeled as $\texttt{S}_{\texttt{beat}}$. The period of $\texttt{S}_{\texttt{beat}}$ is measured using a counter clocked by the sampling signal. This counter resets every period of $\texttt{S}_{\texttt{beat}}$. The counter's output signal, denoted as `CSCnt`, represents a discrete *random variable*, C, influenced by the *independent* random jitter present in both oscillators.

[1] https://github.com/KULeuven-COSIC/COSO-TRNG

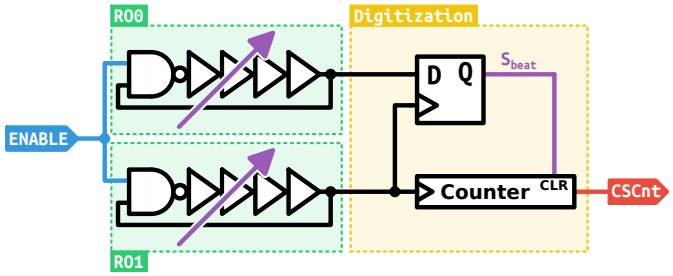

Figure 6.1: Architecture of the COSO-ES.

According to the model presented in [95], the mean and the variance of this random variable are

$$\mathbf{E}[C] = \frac{\mathbf{E}[T_{RO0}]}{\mathbf{E}[\Delta]}, \tag{6.1}$$

$$\mathbf{Var}[C] = \mathbf{E}[C]\frac{\mathbf{Var}[\Delta]}{\mathbf{E}[\Delta]^2}. \tag{6.2}$$

Here, T_{RO0} and T_{RO1} represent the period lengths for the `RO0` and `RO1` ROs, respectively. Mean and variance of the period difference Δ are equal to:

$$\mathbf{E}[\Delta] = \big|\mathbf{E}[T_{RO1}] - \mathbf{E}[T_{RO0}]\big|,$$

$$\mathbf{Var}[\Delta] = \mathbf{Var}[T_{RO0}] + \mathbf{Var}[T_{RO1}]. \tag{6.3}$$

The random variable B, representing the Least Significant Bit (LSB) of the `CSCnt` signal, is now utilized as the generated random bit.

6.2.2 Entropy Rate Optimization

Calculating the *entropy density* per generated bit requires knowledge of the exact distribution of C. Deriving an *analytical* expression for this distribution starting from the stochastic model is a challenging task. In [95], this problem was addressed by assuming that the resulting distribution follows a normal distribution. This *assumption* simplifies the stochastic model to the ERO-ES model as presented in [49]. However, our findings indicate that the approximation employed in [95] does not hold true when the oscillators are highly matched, meaning that the period difference is of the same order of magnitude as the accumulated jitter.

Therefore, we estimate the distribution of C conducting an event-driven simulation of the COSO-ES. Variables for this model include:

- The sampled signal's average period length: $\mathbf{E}[T_{RO0}]$
- The average period length difference: $\mathbf{E}[\Delta]$
- The linear *jitter strength*: $\frac{\mathbf{Var}[T_{RO0}]}{\mathbf{E}[T_{RO0}]}$

In the model, we assumed both random variables T_{RO0} and T_{RO1} to be independent and normally distributed:

$$T_{RO0} \sim \mathcal{N}\big(\mathbf{E}[T_{RO0}], \mathbf{Var}[T_{RO0}]\big),$$

$$T_{RO1} \sim \mathcal{N}\big(\mathbf{E}[T_{RO1}], \mathbf{Var}[T_{RO1}]\big).$$

Another assumption is made by equating the variance of both oscillating signals:

$$\mathbf{Var}[T_{RO0}] = \mathbf{Var}[T_{RO1}].$$

This assumption can be justified by the fact that both ROs are implemented in the same technology, implying they have similar jitter characteristics. Additionally, the period difference, $\mathbf{E}[\Delta]$, is small.

Based on the estimated distribution of C, we calculate the *probabilities* of the random output bit B being zero or one as

$$\forall i \in \{0,1\} : \mathbf{P}[B = i] = \sum_{j=0}^{\infty} \mathbf{P}[C = 2j + i].$$

The *min-entropy* is then calculated as

$$\mathbf{H_m}[B] = -\log_2\big(\max_{i\in\{0,1\}} \mathbf{P}[B = i]\big).$$

The expected throughput is equal to

$$tp = \frac{1}{\mathbf{E}[C]\mathbf{E}[T_{RO1}]}.$$

The equations above enable us to estimate the Min-entropy-Throughput Product (HTP).

Figures 6.2 and 6.3 present the simulation results for *platform parameters* obtained for a Spartan 7 and SmartFusion2 implementation, respectively. In the upper part, the shaded region represents the min-entropy exceeding a value

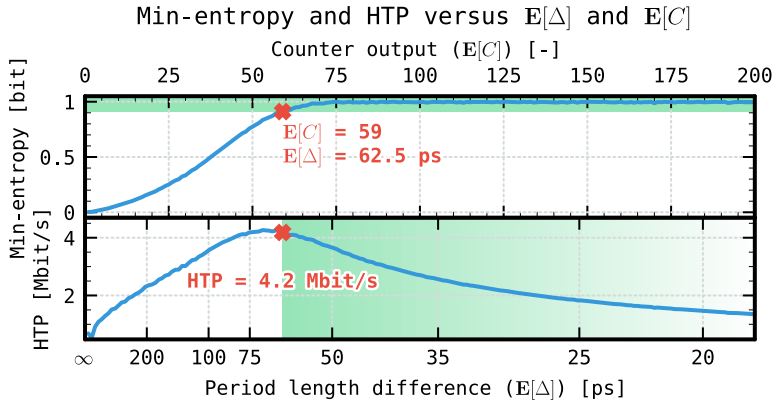

Figure 6.2: Estimated min-entropy (top) and HTP (bottom) versus $\mathbf{E}[\Delta]$ and $\mathbf{E}[C]$ for a Spartan 7 implementation, $\mathbf{E}[T_{RO1}] = 3.69\,\text{ns}$ and $\mathbf{Var}[T_{RO1}] = (4.1\,\text{ps})^2$.

of 0.91 bit per output bit, required by the AIS 31 standard [39]. The cross markers denote the point with the minimum $\mathbf{E}[C]$ that meets this requirement. Configuration values of $\mathbf{E}[C]$ and $\mathbf{E}[\Delta]$ to the right of these markers are compliant with the AIS 31 standard. The shading gradient indicates that configurations further to the right have reduced throughput, making them less desirable. To achieve maximum throughput, configurations as close as possible to the markers are preferred. The lower part of the figures displays the HTP, indicating a peak for a specific value of $\mathbf{E}[\Delta]$. This peak represents a trade-off between ensuring sufficient *accumulation time* for entropy generation while maintaining high throughput.

6.3 COSO-ES Architecture

6.3.1 Entropy Source

The two ROs, `RO0` and `RO1`, that form the ES as shown in fig. 6.1, are built using the configurable RO architectures detailed in section 5.2. Four RO topologies are implemented: `GateVar`, `WireVar`, `LUTVar0`, and `LUTVar5`.

Note. When using a SmartFusion2 instead of a Spartan 7 FPGA platform, the `LUTVar5` topology is replaced by the `LUTVar3` topology.

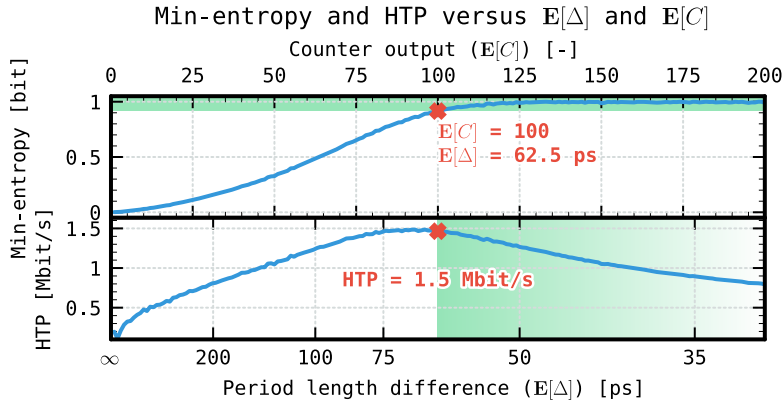

Figure 6.3: Estimated min-entropy (top) and HTP (bottom) versus $\mathbf{E}[\Delta]$ and $\mathbf{E}[C]$ for a SmartFusion2 implementation, $\mathbf{E}[T_{RO1}] = 6.25\,\text{ns}$ and $\mathbf{Var}[T_{RO1}] = (3.2\,\text{ps})^2$.

Table 6.1: Total number of configuration values and number of configuration bits per RO when both ROs consist of $n+1$ stages.

RO type	GateVar	WireVar	LUTVar
Configurations	$(4^n)^2$	$(4^n)^2$	$(32^n)^2$
Configuration bits (k)	$2n$	$2n$	$5n$

Table 6.1 provides the total number of configuration values for both `RO0` and `RO1` combined when both ROs are implemented using $n+1$ stages. To achieve optimal period length matching, both ROs always have an equal number of stages. Practical implementation details for each RO topology are given in section 5.2.1 of the previous chapter.

6.3.2 Digitization

A detailed hardware diagram of the digitization module is provided in fig. 6.4. In this set-up, `RO0` is sampled by a clock signal generated by `RO1` using `DFF0`. The output of `DFF0` ($\mathtt{S}_{\mathtt{beat}}$) is used to clock `DFF1`, `DFF2`, and `DFF3`. These DFFs regulate the enabling and clearing of the asynchronous counter, which measures the period length of $\mathtt{S}_{\mathtt{beat}}$ relative to the period length of the oscillating signal produced by `RO1`. Synchronization with other logic occurs via a two-way handshake, using the signals `ack` and `req` generated by `DFF3`. The asynchronous

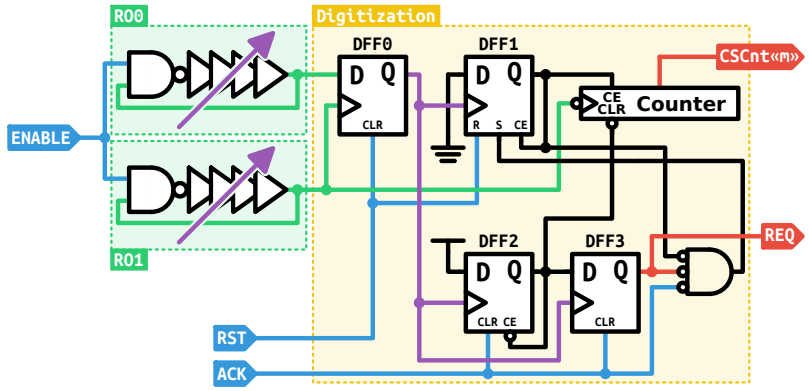

Figure 6.4: Detailed architecture of the COSO-ES digitization.

counter produces an m-bit signal (`CSCnt`) that the controller uses to configure the ROs, with the LSB of this value being used to generate random bits.

6.3.3 Controller

Simplified pseudocode describing the controller is depicted in algorithm 6.1. Its function is monitoring if the value of the `CSCnt` signal (C) is still within predefined boundaries (l and h in the algorithm, representing the lower and upper bound respectively) and selecting a new $2k$-bit RO configuration (S) if necessary, each RO receives a k-bit configuration signal. The relation between the number of configuration bits, k and the number of RO stages, $n+1$ is provided in table 6.1. $\mathbf{E}[C]$ is directly related to the matching of the two ROs ($\mathbf{E}[\Delta]$) via eq. (6.1). The controller sequentially goes through the possible configurations until a suitable one is found. Optimal boundaries have to be chosen to maximize the throughput and provide sufficient entropy from figs. 6.2 and 6.3. A smaller range $[l, h)$ enables finer control, but increases the controller latency to find a suitable configuration.

This controller activates at start-up to find a configuration that validates the stochastic model. After start-up, the controller remains actively checking the produced C values and dynamically recalibrates the ROs when necessary.

Algorithm 6.1 Controller pseudocode.

Input: C[m-1:0], req
Output: S[2k-1:0], matched
Global constant: l, h

```
1: good_samples[6:0] ← 0
2: sample_cnt[6:0] ← 0
3: S[2k-1:0] ← 0
4: matched ← 0
5: while true do
6:   if req then
7:     if l ≤ C[m-1:0] < h then
8:       good_samples[6:0] ← good_samples[6:0] + 1
9:       matched ← 1
10:    end if
11:    if sample_cnt[6:0] == 2^7 - 1 then
12:      if good_samples[6:0] == 0 then
13:        S[2k-1:0] ← S[2k-1:0] + 1
14:        matched ← 0
15:      end if
16:      good_samples[6:0] ← 0
17:    end if
18:    sample_cnt[6:0] ← sample_cnt[6:0] + 1
19:  end if
20: end while
```

6.4 Experimental Results

In this section, we will experimentally verify the correct operation of the COSO-ES using the three proposed RO architectures. We will revisit the questions posed in section 5.2.2 on pages 97 to 98, this time focusing on the obtained C values to assess the performance of the ES.

The questions on pages 97 to 98 are repeated here, rephrased to focus on evaluating the ES:

Question 1: Can the proposed configurable COSO-ES produce a wide range of C values?

Question 2: Is searching for an optimal GP inside the FPGA chip still necessary?

Question 3: Are LP constraints still necessary?

Question 4: Can the proposed configurable COSO-ES also work on other FPGAs?

Question 5: How many RO stages are necessary?

Question 6: What is the influence of the implementation strategy?

Question 7: What will happen if the FPGA routing becomes highly congested?

Each of these questions is addressed in the following subsections. First, the experimental set-up is further explained.

Measured C distributions will be visualized using box plots. A box plot displays the data distribution, with 50 % of the data contained within the box. The median of the distribution is represented by a horizontal line inside the box. A data point is considered an outlier (marked with a dot in the plot) if it is more than 1.5 times the interquartile range from the nearest quartile. All box plot figures depicting a C distribution will be overlaid on a gradient-shaded background representing configurations with sufficient entropy density per bit.

6.4.1 Experimental Set-up

We first implemented the proposed ES and digitization as depicted in Fig. 6.4 on a Xilinx Spartan 7 FPGA. Unless otherwise indicated, the number of configurable stages (n) in any experiment, except in sections 6.4.3 and 6.4.5 (where only three stages were used to reduce measurement time), is set to four. This results in a total of $4^4 = 256$ configurations for each `GateVar` and `WireVar` RO, and $32^4 = 1\,048\,576$ configurations for each `LUTVar` RO. To reduce measurement time, similar to section 5.2.2, we did not test every `LUTVar` configuration but instead used an LFSR to generate $2^{15} = 32\,768$ configuration inputs.

For realistic values of $\mathbf{Var}[T_{RO1}]$ and $\mathbf{E}[\Delta]$, $\mathbf{E}[C]$ is below 256, allowing the use of an 8-bit asynchronous counter in the digitization module. However, for the initial experiments, we used a 16-bit counter. This counter is reset every period of $\mathtt{S}_{\mathtt{beat}}$ and read by the controller. The LSB of this counter is used as the generated random bit.

6.4.2 Feasibility of the Architecture

Addressing question 1 on page 123, we first implemented the ROs and digitization, with manual GP and LP constraints on a Xilinx Spartan 7 FPGA.

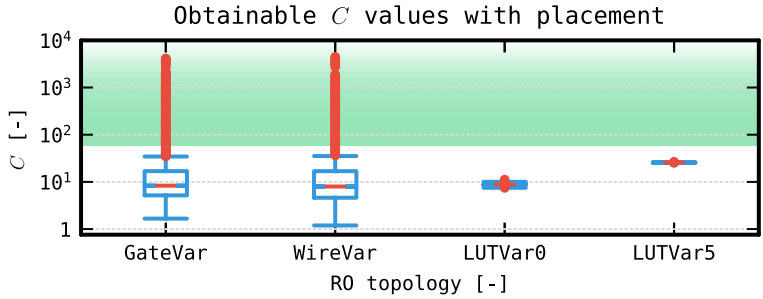

Figure 6.5: Obtainable C values with fixed GP and LP constraints on a Spartan 7 FPGA.

In fig. 6.5, the box plot shows the obtained C realizations. The `GateVar` and `WireVar` architectures can achieve a wide range of values up to 1×10^4 (Δ as low as 0.4 ps), allowing the controller to find a configuration near the optimal HTP. In contrast, the `LUTVar` ROs do not provide a sufficiently wide range of oscillation frequencies to achieve high C values. This limitation is due to the restricted range of frequencies a single `LUTVar` RO can produce, making it difficult to compensate for the inherent mismatch between the two ROs by reconfiguring the oscillator pair.

6.4.3 Global Placement

To verify question 2 on page 123, assessing whether the RO architectures can achieve RO matching without dependence on GP constraints, we conducted the previous experiment across 25 manually selected locations spanning the entire FPGA. LP constrains were maintained to enhance RO matching.

Figures 6.6 to 6.9 depict box plots for each tested location using the `GateVar`, `WireVar`, `LUTVar0`, and `LUTVar5` architectures, respectively. In this experiment, C values are computed based on measured RO frequencies rather than testing every possible RO pair, reducing measurement time. Each ES employs ROs with three configurable stages.

This experiment demonstrates that adequate matching can be achieved at every tested location for the `GateVar` and `WireVar` architectures. This stands in stark contrast to earlier designs utilizing the COSO-ES with ROs [70, 85, 95], where a significant design effort was required to manually identify FPGA locations with satisfactory matching between the ROs.

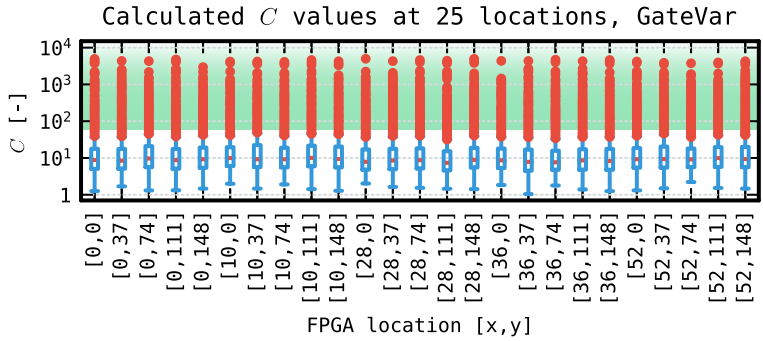

Figure 6.6: Calculated C values using the `GateVar` topology at 25 different locations on a Spartan 7 FPGA.

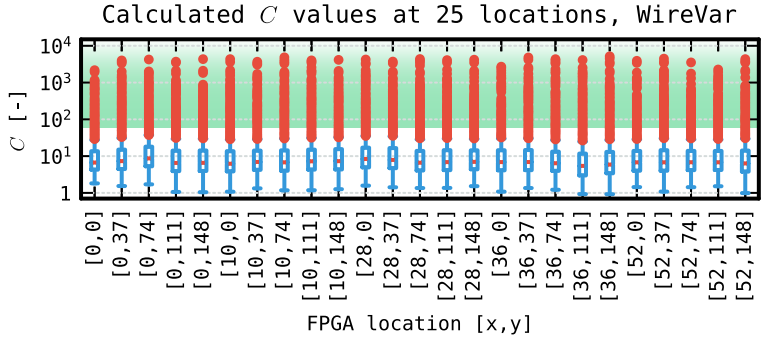

Figure 6.7: Calculated C values using the `WireVar` topology at 25 different locations on a Spartan 7 FPGA.

The results from the `LUTVar` architectures resemble those observed in previous COSO-ES designs in literature: certain locations exhibit well-matched ROs and achieve high C values. However, these locations are still dispersed across the FPGA fabric (12 out of 25 locations for `LUTVar0` and 13 out of 25 locations for `LUTVar5`).

Note. The `LUTVar0` topology offers a broader range of C values compared to the `LUTVar5` topology, consistent with the larger period length range observed in section 5.2.2 of the previous chapter.

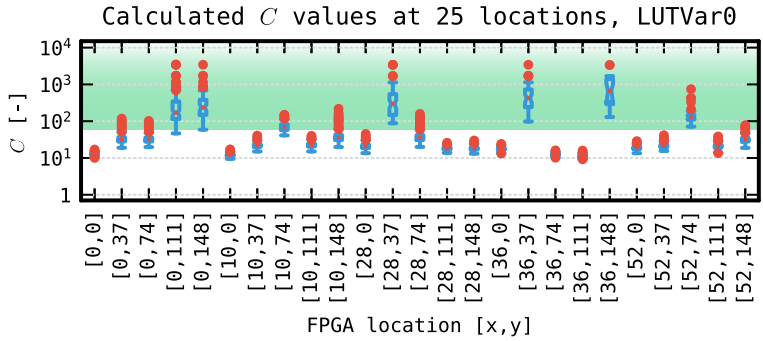

Figure 6.8: Calculated C values using the `LUTVar0` topology at 25 different locations on a Spartan 7 FPGA.

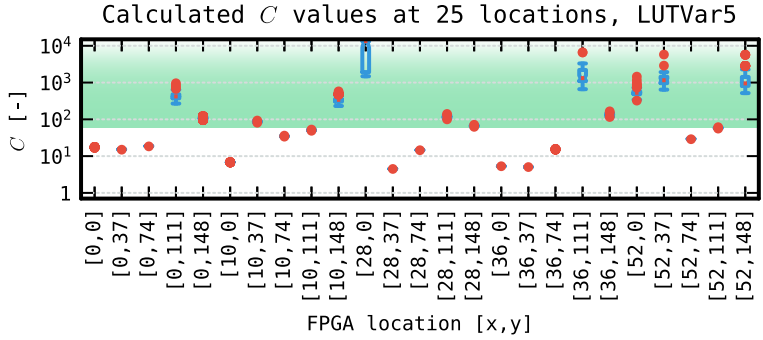

Figure 6.9: Calculated C values using the `LUTVar5` topology at 25 different locations on a Spartan 7 FPGA.

6.4.4 Local Placement

The next experiment addresses question 3 on page 124 and demonstrates that the correct operation of the ES does not necessitate LP or GP constraints.

Figure 6.10 illustrates the measured C values for an implementation on a Spartan 7 FPGA, without any specified placement constraints. The figure demonstrates that matching can still be achieved for the `GateVar` and `WireVar` RO architectures. In this particular instance, both `LUTVar` architectures also achieve sufficiently high C values. However, as will be discussed in section 6.4.7, these outcomes are highly affected by the FPGA implementation strategy and thus challenging for the designer to control.

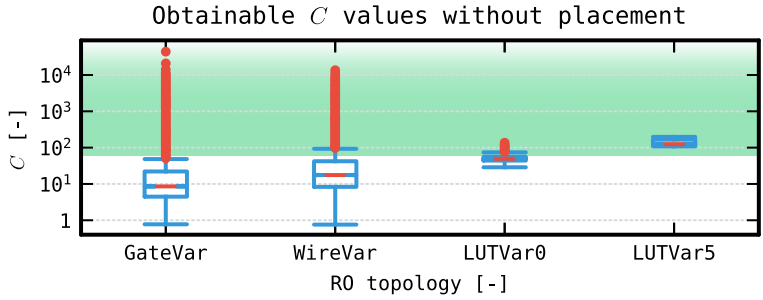

Figure 6.10: Obtained C values without specified GP and LP constraints on a Spartan 7 FPGA.

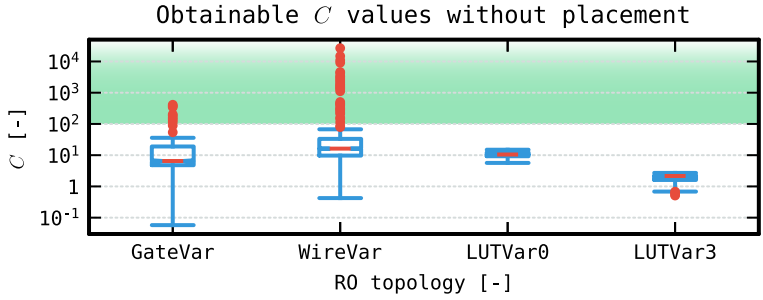

Figure 6.11: Obtained C values without specified GP and LP constraints on a SmartFusion2 FPGA.

6.4.5 Portability

To evaluate the portability of the ES, as considered by question 4 on page 124, we repeated the previous experiment without any GP or LP constraints, this time using a Microsemi SmartFusion2 FPGA with three-stage ROs.

The results depicted in fig. 6.11 are similar to those obtained on the Spartan 7 FPGA. Notably, the `GateVar` and `WireVar` architectures demonstrate consistent performance, confirming the portability claim.

6.4.6 Optimal Number of Stages

In this experiment, we varied the number of stages to address question 5 on page 124, assessing the impact on achievable C values. Increasing the number

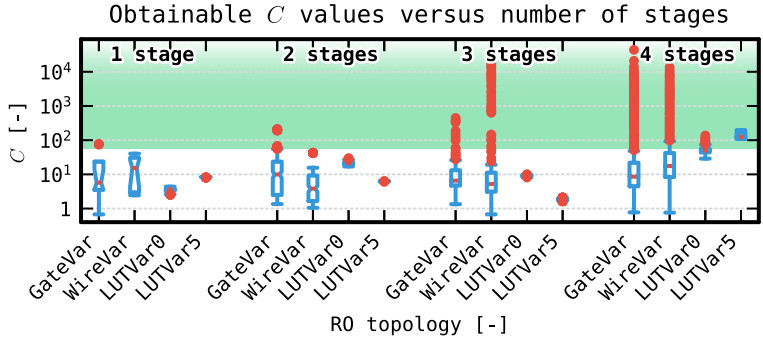

Figure 6.12: Obtained C values for different number of RO stages and omitted GP and LP constraints on a Spartan 7 FPGA.

of stages expands the configuration space, enhancing the *likelihood* of achieving adequate matching.

Figure 6.12 presents the outcomes of this experiment conducted on a Spartan 7 FPGA, where placement constraints were omitted. The `GateVar` and `WireVar` architectures exhibit minimal or no C values falling within the shaded region when utilizing one or two stages. However, configurations employing three or more stages demonstrate a substantial number of C values within this desirable range. Conversely, the `LUTVar` architectures only achieve sufficient C values in the desirable range when employing four stages.

6.4.7 Implementation Strategy

To address question 6 on page 124, we conducted a repeat of the previous experiment using the Area Explore implementation strategy instead of the default strategy in the Xilinx Vivado design tool. The HDL implementation of the ES circuit remained unchanged.

Figure 6.13 presents the results of the experiment. The `GateVar` and `WireVar` architectures with four stages consistently yield C values within the shaded region. For `WireVar` with three stages, only a few configurations achieve sufficiently high C values. However, `LUTVar5` fails to achieve adequate RO matching across all number of stages tested.

Based on these findings, we recommend employing the `GateVar` or `WireVar` architecture with four stages to ensure reliable matching of the ROs, while

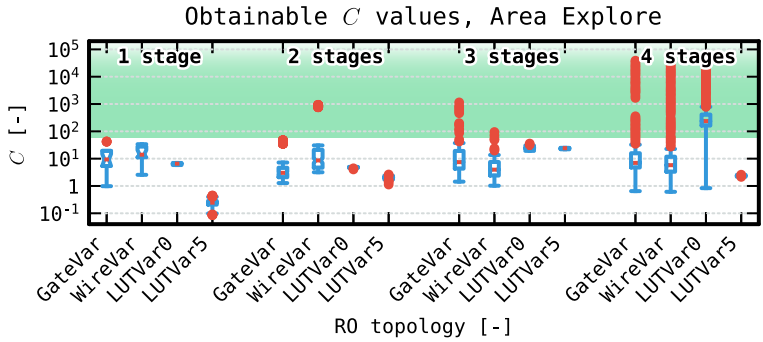

Figure 6.13: Obtained C values for different number of RO stages and omitted GP and LP constraints on a Spartan 7 FPGA, using the Area Explore implementation strategy.

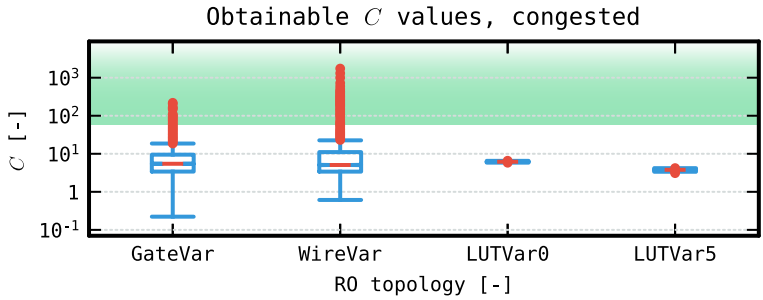

Figure 6.14: Obtained C values for omitted GP and LP constraints on a heavily congested Spartan 7 FPGA.

maintaining a smaller RO footprint compared to configurations with more than four stages.

6.4.8 FPGA Congestion

Finally, addressing question 7 on page 124, we increased the FPGA utilization by employing 64-bit adder structures, similar to section 5.2.2.

The results of this experiment are depicted in fig. 6.14, showing the achieved C values. Despite the FPGA congestion, both `GateVar` and `WireVar` architectures still manage to attain C values within the desired range.

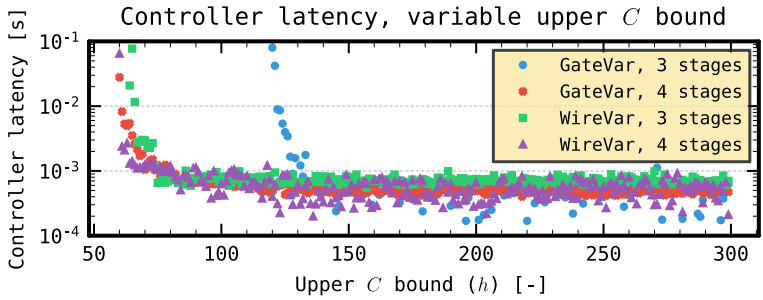

Figure 6.15: Measured controller latency when selecting a new RO configuration with variable upper C bound (h) and fixed lower C bound (l) equal to 59 on a Spartan 7 FPGA.

6.4.9 Controller Latency

Under changing *operating conditions*, the current RO configuration may cease to generate a valid C value. In such cases, the controller is required to update the configuration select signal by sequentially scanning through subsequent configurations. This procedure is also executed during device startup and unavoidably introduces latency due to the scanning process. The time taken for the selection procedure depends on the lower and upper bounds of C (l and h respectively in algorithm 6.1). Broader bounds facilitate the selection process since more configurations meet the criteria.

In this final experiment, we measure the latency of the selection procedure using a variable upper bound, h. The lower bound, l, is fixed at 59 to meet the *entropy requirement* illustrated in fig. 6.2. The latency results are presented in fig. 6.15. For the `GateVar` topology with four stages and the `WireVar` topology with both four and three stages, latency decreases as the upper bound increases, stabilizing around an upper bound of 100. However, for the `GateVar` architecture with three stages, suitable configurations are found only when the upper bound exceeds 120, with a latency stabilizing at approximately 500 µs.

6.5 Results and Comparison

To theoretically estimate the entropy density per bit, we require the magnitude of the jitter strength: $\frac{\mathbf{Var}[T_{RO0}]}{\mathbf{E}[T_{RO0}]}$. For the SmartFusion2 implementation, we refer to values from [70], which used similar FPGA devices. To ensure a conservative entropy estimate, we consider the minimum jitter strength observed across all

tested frequencies in [70]. This minimal jitter strength is measured as 1.6 fs, resulting in a period jitter of 3.2 ps for an RO oscillating at 160 MHz. For the Spartan 7 implementation, we estimate the jitter strength using the variance of the measured C values via eqs. (6.2) and (6.3), yielding a jitter strength of 4.6 fs. Given these jitter strengths, the acceptable range of C is depicted by the shaded area in the lower part of figs. 6.2 and 6.3. We assume that the jitter strength is independent of the RO architecture, hence the computed bounds in these figures are applicable irrespective of the specific RO architecture. This assumption can be justified by the fact that all proposed RO architectures utilize the same underlying FPGA circuitry, including LUTs and switching matrices.

Random bits were generated using the `GateVar` and `WireVar` RO architectures in a Spartan 7 implementation. To assess portability, the `GateVar` architecture was also employed for random bit generation on a SmartFusion2 platform. We did not utilize the `LUTVar` architecture due to its inability, as demonstrated in section 6.4, to reliably match the two ROs. All random bits were generated without specific GP and LP constraints, using ROs with four configurable stages for the Spartan 7 implementation and three configurable stages for the SmartFusion2 implementation. Given a successful implementation of the `GateVar` architecture on SmartFusion2, we do not explore the `WireVar` architecture on this platform.

Both the Spartan 7 implementations (`GateVar` and `WireVar`) and the SmartFusion2 implementation (`GateVar`) achieved configurations with average C values falling within the shaded region, close to the maximum HTP: 60.6, 77.8, and 107.9 respectively, resulting in throughputs of 4.65 Mbit s^{-1}, 2.34 Mbit s^{-1}, and 1.47 Mbit s^{-1}. According to [39], the Shannon entropy density per output bit should exceed 0.997 bit. Converted to min-entropy, it should be higher than 0.91 bit per output bit. At measured oscillation frequencies of 271 MHz and 160 MHz, the obtained min-entropy values from the stochastic model are 0.92 bit, 0.99 bit, and 0.95 bit per output bit for the Spartan 7 (`GateVar` and `WireVar`) and SmartFusion2 (`GateVar`) implementations, respectively. The bit streams successfully passed test procedures B (T6 to T8) as described in AIS 31 [39], using 120 Mbit of generated data, with estimated Shannon entropy densities of 0.998 bit, 0.999 bit, and 0.997 bit per output bit based on the results of test T8. Both implementations are thus PTG.3 compliant upon the addition of cryptographic post-processing. We utilized CBC-MAC post-processing as specified by the NIST SP 800-90B standard [83]. Post-processing reduced the throughput by half, and all three bit streams passed the NIST SP 800-22 test suite.

A comparison with previous COSO-ES designs and other ES implementations compliant with the AIS 31 standard is presented in table 6.2. The results from [63], where the `GateVar` RO was implemented on a Xilinx Spartan 6 FPGA,

are included for reference. Compared to other designs, the ES proposed in this chapter achieves a throughput exceeding $1\,\mathrm{Mbit\,s^{-1}}$ with a larger area footprint, although still a lot smaller than the design presented in [95]. Notably, the ES in this chapter incorporates an on-line testing module that alerts users when sufficient matching cannot be achieved. This feature aligns with certification requirements outlined in both standards [39, 83], and contributes to an increased area usage compared to other designs.

A precise breakdown of the area requirements for the proposed ES design is provided in table 6.3. The table presents both the absolute numbers of used FFs/LUTs and their proportions relative to the total available FFs/LUTs in the FPGA, for different number of stages, n.

The design effort is significantly reduced in this approach compared to previous methods that typically involved Manual Placement (MP) and often Manual Routing (MR). Such efforts needed to be repeated each time the design was implemented on a different device, even within the same FPGA family.

For applications requiring higher throughput, techniques introduced in [85] can be integrated with our proposed ES. These methods enhance throughput by a factor of four. However, the stochastic model must be expanded to accurately estimate the entropy density when employing mutual sampling.

6.6 Conclusion and Future Work

In this chapter, we introduced and evaluated three new RO topologies for the COSO-ES, which significantly reduce the required design effort. We experimentally validated the feasibility of the `GateVar` and `WireVar` architectures, demonstrating that they consistently meet entropy requirements derived from the stochastic model even when GP and LP constraints are omitted or when the design is ported to a different FPGA family. This characteristic makes the proposed ES highly suitable for integration into larger cryptographic systems. Random bits were generated with maximum throughputs of $4.65\,\mathrm{Mbit\,s^{-1}}$ on a Spartan 7 and $1.47\,\mathrm{Mbit\,s^{-1}}$ on a SmartFusion2 FPGA, both passing the necessary statistical tests. Although the design requires a modest area, it comes with a cost of increased latency when the controller searches for an optimal configuration.

Note. The results presented in this chapter, particularly figs. 6.2 and 6.3, do not account for dependencies on the previous ES state or output. Given the insights from the research presented in part I, especially considering the presence of flicker FM noise, the COSO-ES stochastic model must incorporate these

Table 6.2: Comparison with related work.

Architecture	FPGA family	Area [FFs/LUTs]	Throughput [Mbit s^{-1}]	Design effort
This work	**Spartan 6**[a][b]	**39/108**[c]	**3.30**	**-**
	SmartFusion2[a]	**38/111**[c]	**1.47**	**-**
	Spartan 7[a]	**62/82**[d]	**4.65**	**-**
	Spartan 7[e]	**62/58**[d]	**2.34**	**-**
Original COSO [70]	Spartan 6	3/18	0.54	MP
	Cyclone V	3/13	1.44	MP
	SmartFusion2	3/23	0.328	MP
COSO: one bit per half cycle [85]	Actel Fusion AFS600	7/24[f]	2	MP & MR
	Spartan 3	7/18[f]	1.6	MP & MR
COSO: mutual sampling [85]	Actel Fusion AFS600	14/29[f]	4	MP & MR
	Spartan 3	14/23[f]	3.2	MP & MR
COSO: parameter adjustment [95]	Virtex-5	109 slices	4.08	MP & MR
DC-ES [5]	Spartan 6	128 slices	1.1	MP
	Cyclone V	273 ALMs	1.116	MP & MR
PLL-ES [5]	Spartan 6	190 slices[g]	1.0416	PLL required
	Cyclone V	273 ALMs	1.04	PLL required
Edge sampling-ES [93]	Spartan 6	5/10	1.15	MP
	Cyclone V	6/10	1.067	MP
TERO-ES [70]	Spartan 6	12/39	0.625	MP & MR
	Cyclone V	12/46	1	MP & MR
	SmartFusion2	12/46	1	MP & MR
STR-ES [70]	Spartan 6	256/346	154	MP & MR
	Cyclone V	256/352	245	MP & MR
	SmartFusion2	256/350	188	MP & MR

[a] Using the `GateVar` RO architecture.
[b] Results reported in [63].
[c] Values calculated for three-stage ROs, controller hardware included.
[d] Values calculated for four-stage ROs, controller hardware included.
[e] Using the `WireVar` RO architecture.
[f] Values calculated using shown circuit diagram.
[g] Including embedded tests and data interface.

Table 6.3: Detailed area breakdown.

	Spartan 7		SmartFusion2	
	[FFs/LUTs]	[%FFs/%LUTs]	[FFs/LUTs]	[%FFs/%LUTs]
Controller:	38/38[a]	0.058/0.073[a]	38/78[b]	0.137/0.282[b]
RO:				
`GateVar`	$0/4n+2$	0/0.035[c]	$0/8n-1$	0/0.083[d]
`WireVar`	$0/n+2$	0/0.012[c]	$0/2n$	0/0.022[d]
`LUTVar`	$0/n+2$	0/0.012[c]	$0/2n$	0/0.022[d]
Digitization:	20/17	0.031/0.033	20/16	0.072/0.058

[a] Valid for a four stage `GateVar` or `WireVar` RO architecture.
[b] Valid for a three stage `GateVar` or `WireVar` RO architecture.
[c] Valid for four stages.
[d] Valid for three stages.

dependencies to accurately estimate the entropy density *conditioned* on previous ES state information.

Future work will explore the applicability of this methodology to other ES designs that demand significant design effort, optimize the controller's search strategy to reduce latency, examine the behavior of this ES under active manipulation attacks, and update the stochastic model to include a wider range of *noise colors*.

Chapter 7

Configurable TRNGs for ASICs

This chapter is based on the following publication:

> **An Energy and Area Efficient, All Digital Entropy Source Compatible with Modern Standards Based on Jitter Pipelining**
> Adriaan Peetermans, and Ingrid Verbauwhede
> *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, 2022

Contribution: *main author*.

7.1 Background and Context

As discussed with great detail in section 2.2, validation of the correct functioning of an ES must adhere to strict rules set by international standards on random number generation [34, 39, 83]. This process is centered around the existence of a *stochastic model*.

ES designs implemented on ASICs that are compatible with this workflow include the TERO-ES, initially proposed by [86] and implemented in [96] (*oscillator jitter*-based), a cross-coupled inverter pair ES, proposed and implemented by [54] (*metastability*-based), and the STR-ES, first proposed by [12] and implemented in [15] (oscillator jitter-based). Many of these designs, however, were initially introduced for use on FPGA platforms and only later adopted by the circuit

community for ASIC implementations. As these ESs originated from FPGA implementations, they make only limited use of the extensive *design flexibility* that the ASIC platform offers.

The designs presented in [15, 54, 96] represent a limited subset of ASIC implementations available in the literature that follow the design procedures mandated by international standards, as detailed in sections 2.3 and 2.4. Despite the increasing availability of numerous high-performance ASIC ES designs, such as those in [40, 81], which showcase innovative circuit techniques, many of these ASIC-tailored designs lack a proper *security evaluation*. Specifically, they often fail to specify a stochastic model or perform a profound theoretical *entropy assessment*.

Following the *modern* approach to ES design, this chapter introduces an ES architecture, explicitly constructed for use on an ASIC platform. The contributions of the work presented in this chapter are as follows:

- A novel *all-digital* ES architecture based on the unpredictable timing jitter in inverter DC ROs is proposed. The timing jitter is resolved by a TDC using two free-running ROs, similar to the method in [43]. This all-digital architecture offers two main advantages: it facilitates easy integration into more complex digital systems and benefits from further CMOS scaling without necessitating a complete circuit redesign.
- Reducing the TDC *resolution* and simultaneously accumulating *independent* timing jitter (creating a *jitter pipeline*) decreases the required jitter *accumulation time*, allowing for a *throughput* of several hundred megabits per second. This throughput is significantly higher than previously reported for oscillator jitter based ESs [15, 40, 42, 96, 97].
- A detailed stochastic model is provided, capable of estimating the generated output *entropy*.
- Efforts were made to measure and estimate the magnitude of the linear *jitter strength platform parameter*, a required input for the stochastic model.
- A *design parameter* optimization strategy is provided to maximize the ES throughput.
- The design is implemented in a 28 nm CMOS technology, and measurement results are available.

This chapter is structured as follows: section 7.2 introduces the ES architecture, including the principle of the jitter pipeline. A detailed mathematical description

of the stochastic model for the proposed ES is available in section 7.3. Section 7.4 presents a rudimentary jitter measurement set-up and experimental jitter measurement results, utilizing only the available on-chip ES hardware. An optimization strategy for selecting optimal design parameters, such as the DC jitter accumulation lengths or the TDC oscillation frequency, is detailed in section 7.5. Experimental results showcasing the performance of the proposed ES are presented in section 7.6. Finally, the chapter concludes with a comparison to related work in section 7.7.

7.2 ES Architecture

This section provides a high-level overview of the ES architecture and jitter pipeline principle. A more detailed mathematical analysis of the design follows in section 7.3.

7.2.1 Jitter Pipeline

The proposed ES architecture is shown in fig. 7.1, which includes three main components: a DC, a TDC and a digitization block. The DC and TDC each consist of two ROs: `DC0`, `DC1` and `TDC0`, `TDC1`, respectively. Timing jitter naturally accumulates in all four ROs during a specified accumulation interval. The TDC ROs resolve the timing jitter generated by the DC ROs with a resolution determined by the period difference between the two TDC ROs. This process produces a digital representation of the timing difference created by the DC, which is then used to generate a *random* output bit.

To minimize idle time, the DC can begin accumulating jitter for the next output bit during the resolvement phase of the current bit. During this resolvement phase, timing jitter continues to accumulate since the TDC also contains free-running ROs. Both phases provide independent contributions to the output bit entropy, effectively creating a jitter pipeline. In this pipeline, jitter is first generated in the DC stage and then handed over to the TDC stage for further accumulation. This pipelining process is illustrated by the shaded boxes in fig. 7.2.

The concept of jitter pipelining extends beyond the specific use of jitter accumulation and resolvement with a DC and TDC structure as presented in this chapter. It should be viewed as a broader principle that could be applied to other ES architectures as well.

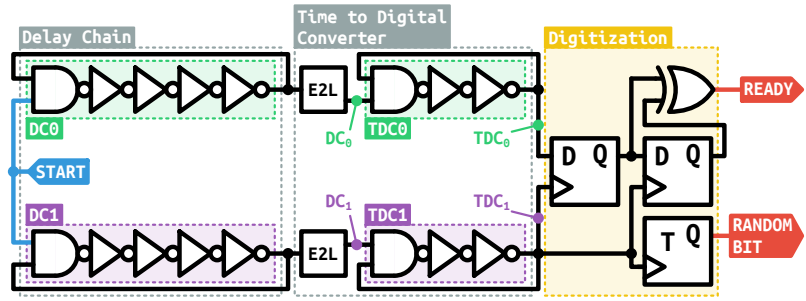

Figure 7.1: ES architecture, containing the jitter pipeline.

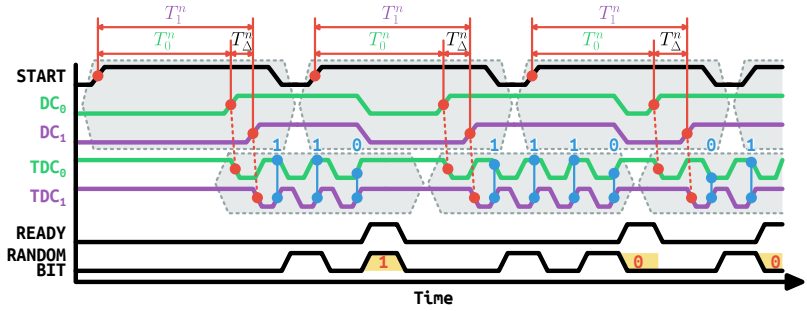

Figure 7.2: ES and jitter pipeline timing diagram.

7.2.2 Architecture Timing Description

A start edge is applied to both DC ROs, each implemented using a *configurable* `Exp4×4` topology, as detailed in section 5.3. The Edge To Level (E2L) blocks in fig. 7.1 respond to the n-th positive edge generated by the DC ROs by disabling them and outputting a positive edge ($\texttt{DC}_0$ and $\texttt{DC}_1$). The time it takes for the start edge to propagate through the DC ROs for n cycles to the output of the E2L block is denoted as T_0^n and T_1^n for `DC0` and `DC1`, respectively, as shown in the timing diagram in fig. 7.2. Random timing jitter variations make the timing difference $T_\Delta^n = T_0^n - T_1^n$ a *random variable* over multiple evaluations. The E2L outputs ($\texttt{DC}_0$ and $\texttt{DC}_1$) then enable the TDC ROs to start oscillating ($\texttt{TDC}_0$ and $\texttt{TDC}_1$).

Both TDC ROs use the configurable `Lin2×8` topology and are set to have slightly different oscillation periods, defining the TDC resolution as $res = |p_{TDC_0} - p_{TDC_1}|$, with p_w the average period for $w \in \{TDC_0, TDC_1\}$. The ROs begin with an initial phase difference determined by T_Δ^n and continue

oscillating until the phase difference reaches either 0° or 180° (0 or π radians). The digitization circuitry detects this *phase synchronization* when the bottom RO (`TDC1`) starts sampling a different logic value from the top RO (`TDC0`) via an XOR gate. A TFF then determines if `TDC1` experienced an odd or an even number of cycles during phase synchronization. The output of this TFF is used as the random output bit.

7.3 Stochastic Model

This section provides a mathematical characterization of the circuit proposed in section 7.2 and quantifies the entropy extracted from the available timing jitter. The entropy estimation in this chapter is based solely on the presence of unmanipulatable *thermal noise*. As demonstrated in chapter 3, other *noise sources* will inevitably be present as well. *Assuming* thermal noise is independent of all other noise sources, their coexistence will not reduce entropy. Therefore, the estimation provided here inevitably represents a conservative lower bound.

For the notation and definitions of the probability concepts used throughout this chapter, please refer to appendix A.

7.3.1 Model Assumptions

Following the reasoning in section 2.3.2, the stochastic model is based on four main assumptions listed below:

- Thermal noise is unmanipulatable and independent of other noise sources.
- DC and TDC ROs are all *mutually independent* oscillators, affected by thermal noise.
- RO phase affected by thermal noise behaves as a Wiener process with drift.
- Linear jitter strength (defined in section 7.3.2) is small: $s_{noise} \ll 1\,\mathrm{s}$.

7.3.2 Description of a Noisy Oscillating Signal

Before diving into the intricate details of the stochastic model, it is constructive to establish fundamental knowledge on timing jitter in free-running ROs.

Noiseless Oscillator

The phase of an oscillating *noiseless* signal is a continuous linear function through time t.

Definition 7.1. (Noiseless oscillator phase) The phase of a noiseless oscillator is defined as $\varphi : \mathbb{R} \to \mathbb{R}$ by $\varphi(t) = \mu t + \phi_0$, with μ defining the oscillation speed or angular frequency and ϕ_0 determining the phase at time zero.

The phase of an oscillator cannot be explicitly observed. Instead, we observe an observable waveform $e(\varphi)$ (current flow or node voltage), which is defined as a function of the implicit phase. Examples of such waveforms include:

$$e(\varphi) = a \sin(\varphi),$$

$$e(\varphi) = \begin{cases} a & \text{if } \varphi \bmod 2\pi < \pi \\ 0 & \text{if } \varphi \bmod 2\pi \geq \pi \end{cases},$$

$$e(\varphi) = \frac{a}{2\pi}(\varphi \bmod 2\pi),$$

representing sinusoidal, square and sawtooth waveforms respectively, with amplitude a.

Definition 7.2. (Oscillator waveform) The waveform of an oscillator is a composite function of time, defined as $w : \mathbb{R} \to R \subseteq \mathbb{R}$ by $w(t) = (e \circ \varphi)(t)$, with R the range of the waveform function e.

Each of these waveforms has a period: $p_w = \frac{2\pi}{\mu}$, meaning that $\forall t \in \mathbb{R} : w(t + p_w) = w(t)$. The waveform frequency is the inverse of the period: $f_w = \frac{\mu}{2\pi}$.

Noisy Oscillator

Assuming the influence of thermal noise alone, likewise to section 3.2.1, we assume here that the phase of an oscillator subjected only to thermal noise, $\{\Phi(t)\}_{t \in \mathbb{R}_{\geq 0}}$, behaves as a Wiener process with drift.

Definition 7.3. (Noisy oscillator phase) The phase of an oscillator influenced by thermal noise is a *random process*, defined as $\Phi : \Omega \times \mathbb{R}_{\geq 0} \to \mathbb{R}$ by

$$\Phi(\omega, t) = \mu t + \phi_0 + \sigma W(\omega, t),$$

with ϕ_0 again the phase at time zero, μ the drift and σ^2 the infinitesimal variance. $\{W(t)\}_{t \in \mathbb{R}_{\geq 0}}$ represents a Wiener random process without drift.

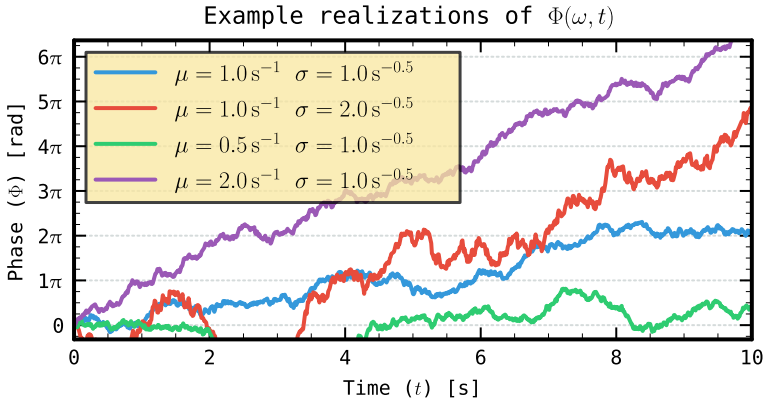

Figure 7.3: Example instances of a random phase process.

The oscillator is assumed to initiate at time zero, as the Wiener process is not defined for negative time. The assumption is based on the description of a Wiener process with drift, which models the integration of currents with a thermal noise (*white*) component onto a load capacitor, as detailed by [1]. Example instances of this phase process are illustrated in fig. 7.3.

Note. As a Wiener process with drift is a special case of a Gaussian process, at any moment in time $t_a \in \mathbb{R}_{\geq 0}$, the value of the phase is normally distributed:

$$\Phi(t_a) \sim \mathcal{N}(\mu t_a + \phi_0, \sigma^2 t_a).$$

Noisy RO

A square waveform with an amplitude of one, for the observable waveform phase function, $e(\varphi)$, is used to model an RO. The phase itself is modelled by a Wiener process with drift and zero initial phase: $\forall \omega \in \Omega, \forall t \in \mathbb{R}_{\geq 0} : \Phi(\omega, t) = \mu t + \sigma W(\omega, t)$, as depicted in fig. 7.4.

The duration of the i-th half period is denoted by the random variable X_i. Because of the *independent increment* property of the Wiener process, each half-period duration of the RO output is Independent and Identically Distributed (IID) compared to all other half periods, and can be represented by a single random variable, X. This duration corresponds to the time it takes for the oscillator phase to reach a multiple of π. Again, due to the independent increment property, we focus solely on the time required to reach a phase of π radians starting from phase zero. All other half periods have the same distribution.

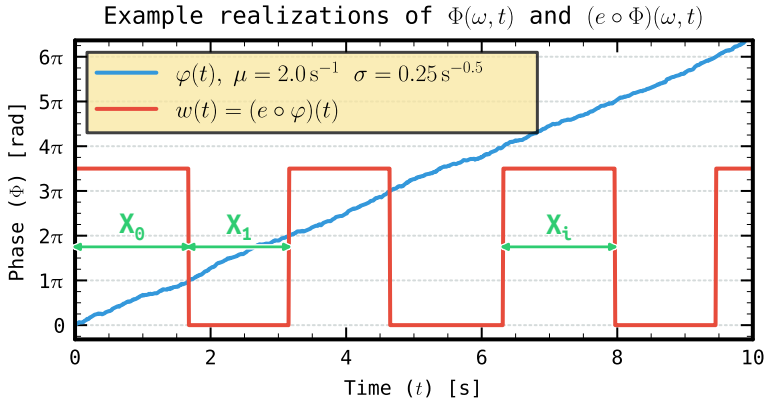

Figure 7.4: RO waveform and corresponding phase versus time.

The time required for a Wiener process with drift to hit a certain level, α, for the first time is inverse-Gaussian distributed: $\mathcal{IG}\left(\frac{\alpha}{\mu},\left(\frac{\alpha}{\sigma}\right)^2\right)$. The half-period duration distribution is then given as

$$X\sim\mathcal{IG}\left(\frac{\pi}{\mu},\left(\frac{\pi}{\sigma}\right)^2\right).$$

From this, the expected value and variance for X can be calculated as $\mathbf{E}[X]=\frac{\pi}{\mu}$ and $\mathbf{Var}[X]=\frac{\pi\sigma^2}{\mu^3}$. The jitter strength, controlling the rate at which linear jitter accumulates in the RO is then equal to

$$s_{noise}=\frac{\mathbf{Var}[X]}{\mathbf{E}[X]}=\left(\frac{\sigma}{\mu}\right)^2, \tag{7.1}$$

with units of time. In practical applications, this quantity typically ranges in the order of femtoseconds [94].

Note. An assumption was made that if the drift μ is positive and the phase starts at zero, it would not return and cross zero into negative values. However, zero is also a multiple of π and will therefore produce an edge at the output when crossed. This is because the inverse-Gaussian distribution only describes the first passage time. For small values of drift relative to the infinitesimal variance ($\frac{\mu}{\sigma}\ll 1\,\mathrm{s}^{-1/2}$), the phase could pass a certain level multiple times, with each passage creating an edge at the output.

The assumptions will only hold true when $\frac{\mu}{\sigma}\gg 1\,\mathrm{s}^{-1/2}$, which is typically the case in most applications where $s_{noise}\ll 1\,\mathrm{s}$. The *probability* of the phase

returning to its starting value and crossing it is equal to

$$\mathbf{P}\big[\Phi(t) \leq 0\big] = \Phi_s\Big(-\frac{\mu}{\sigma}\sqrt{t}\Big),$$

This probability diminishes rapidly over time when $s_{noise} \ll 1\,\mathrm{s}$. For very small time instances ($t \approx (\frac{\sigma}{\mu})^2$ or lower), the assumption also does not hold, as the RO output waveform cannot be considered as an ideal digital signal anymore.

7.3.3 DC Time Difference Distribution

The DC comprises two noisy free-running ROs, with the RO phase described as a random process:

$$\forall\omega \in \Omega, \forall t \in \mathbb{R}_{\geq 0} : \Phi_{DC_0}(\omega, t) = \mu_{DC_0} t + \sigma_{DC_0} W_{DC_0}(\omega, t),$$

$$\forall\omega \in \Omega, \forall t \in \mathbb{R}_{\geq 0} : \Phi_{DC_1}(\omega, t) = \mu_{DC_1} t + \sigma_{DC_1} W_{DC_1}(\omega, t).$$

The DCs start at time zero with an initial phase equal to zero. Both DCs run for a specified number of periods, n, triggering the E2Ls at times T_0^n and T_1^n, respectively:

$$\Phi_{DC_0}(T_0^n) = n2\pi,$$

$$\Phi_{DC_1}(T_1^n) = n2\pi.$$

The first passage time at level $n2\pi$ of a Wiener process with drift is described by the inverse-Gaussian distribution:

$$\begin{aligned} T_0^n &\sim \mathcal{IG}\left(\frac{n2\pi}{\mu_{DC_0}}, \left(\frac{n2\pi}{\sigma_{DC_0}}\right)^2\right), \\ T_1^n &\sim \mathcal{IG}\left(\frac{n2\pi}{\mu_{DC_1}}, \left(\frac{n2\pi}{\sigma_{DC_1}}\right)^2\right). \end{aligned} \tag{7.2}$$

The DC time difference distribution, T_Δ^n, after n periods is given by

$$T_\Delta^n = T_0^n - T_1^n,$$

which is defined by the subtraction of two independent random variables. Its Cumulative Distribution Function (CDF) can be determined by integrating the PDFs of T_0^n and T_1^n.

$$\begin{aligned} F_{T_\Delta^n}(t) &= \mathbf{P}[T_\Delta^n \leq t] = \mathbf{P}[T_0^n \leq t + T_1^n] \\ &= \begin{cases} \int_0^\infty f_{T_1^n}(t_1) \int_0^{t+t_1} f_{T_0^n}(t_0)\,\mathrm{d}t_0\,\mathrm{d}t_1 & \text{if } t \in \mathbb{R}_{\geq 0} \\ \int_0^\infty f_{T_0^n}(t_0) \int_{t_0-t}^\infty f_{T_1^n}(t_1)\,\mathrm{d}t_1\,\mathrm{d}t_0 & \text{if } t \in \mathbb{R}_{<0} \end{cases}. \end{aligned} \tag{7.3}$$

The PDF for T_Δ^n is then equal to

$$f_{T_\Delta^n}(t) = \frac{\partial F_{T_\Delta^n}(t)}{\partial t}. \tag{7.4}$$

Note. In this model, T_0^n and T_1^n are assumed to be independent. Significant effort was made in the design and layout of all four ROs to minimize coupling by introducing separate supply networks and placing each RO in its own N-well. If any dependency remains, it would lead to a reduced jitter strength estimate in section 7.4 thus reducing the entropy claim made by this model.

7.3.4 TDC Run Time Distribution

The TDC oscillators begin oscillating once the respective DC has completed n cycles (at times T_0^n and T_1^n, respectively). Each TDC is a free-running RO, and their phases can be described by a random process, $\forall \omega \in \Omega$:

$$\begin{aligned} &\forall t \in \mathbb{R}_{\geq T_0^n} : \Phi_{TDC_0}(\omega, t) = \mu_{TDC_0}(t - T_0^n) + \sigma_{TDC_0} W_{TDC_0}(\omega, t - T_0^n), \\ &\forall t \in \mathbb{R}_{\geq T_1^n} : \Phi_{TDC_1}(\omega, t) = \mu_{TDC_1}(t - T_1^n) + \sigma_{TDC_1} W_{TDC_1}(\omega, t - T_1^n). \end{aligned} \tag{7.5}$$

Note. Both T_0^n and T_1^n are random variables that follow the distributions given in eq. (7.2). Consequently, $\left\{\Phi_{TDC_0}(t)\right\}_{t \in \mathbb{R}_{\geq T_0^n}}$ and $\left\{\Phi_{TDC_1}(t)\right\}_{t \in \mathbb{R}_{\geq T_1^n}}$ represent random Wiener processes with drift and a random starting time instance.

`TDC1` samples `TDC0` using a DFF. Figure 7.5 illustrates the relation between the TDC phases and the sampling time instances. The TDCs stop oscillating whenever the sampled value (output of the DFF) toggles, and the number of `TDC1` periods is recorded. This toggling occurs whenever the two TDC phases have diverged by more than π. The TDCs stop at the next positive edge of `TDC1`. The TDC phase difference, $\Phi_\Delta(t)$, is defined only for time instances after the second TDC has started, $\forall \omega \in \Omega, \forall t \in \mathbb{R}_{\geq \max(T_0^n, T_1^n)}$:

$$\begin{aligned} \Phi_\Delta(\omega, t) &= \Phi_{TDC_0}(\omega, t) - \Phi_{TDC_1}(\omega, t) \\ &= (\mu_{TDC_0} - \mu_{TDC_1})t + \mu_{TDC_1} T_1^n - \mu_{TDC_0} T_0^n \\ &\quad + \mathcal{N}\big(0, \sigma_{TDC_0}^2(t - T_0^n) + \sigma_{TDC_1}^2(t - T_1^n)\big). \end{aligned} \tag{7.6}$$

Note. The notation $\cdot + \mathcal{N}\big(a, b^2\big)$ in eq. (7.6) indicates the addition of a normally distributed random variable X such that $X \sim \mathcal{N}\big(a, b^2\big)$. This normally distributed variable arises from the properties of Wiener processes and the addition of normal variables: $aW(b) \sim \mathcal{N}\big(0, a^2 b\big)$, for any $b \in \mathbb{R}_{\geq 0}$, and

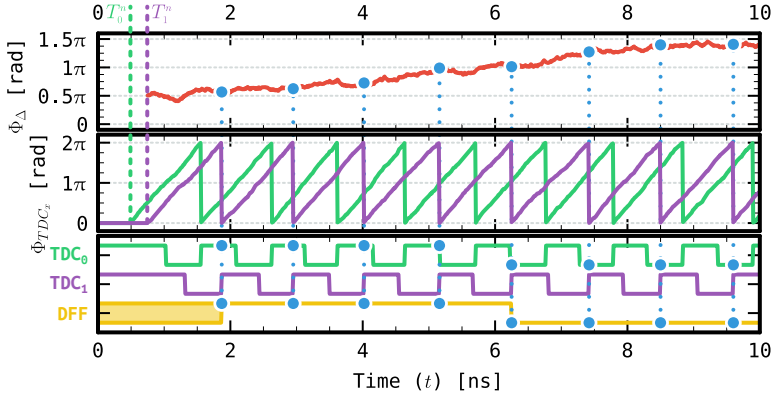

Figure 7.5: Relation between the TDC phases and the sampling time instances. The middle graph shows two realizations of the TDC phase modulo 2π ($\Phi_{TDC_0}(\omega,t)$ mod 2π and $\Phi_{TDC_1}(\omega,t)$ mod 2π), the top graph shows the corresponding realized TDC phase difference ($\Phi_{TDC_0}(\omega,t) - \Phi_{TDC_1}(\omega,t)$), and the bottom graph shows the output waveforms for `TDC0` (top), `TDC1` (middle), and the sampling DFF (bottom).

$\mathcal{N}(a,b^2) - \mathcal{N}(c,d^2) \sim \mathcal{N}(a-c, b^2+d^2)$, for two independent normally distributed random variables.

Now, three scenarios can be distinguished:

1. $\boldsymbol{T_\Delta^n \in \mathbb{R}_{>0}}$ or $\boldsymbol{T_0^n > T_1^n}$: The clocking TDC (`TDC1`) begins oscillating first. A time shift is applied to $\Phi_\Delta(\omega,t)$, where $t' = t - T_0^n$. Substituting this into eq. (7.6) results in, $\forall\omega \in \Omega, \forall t' \in \mathbb{R}_{\geq 0}$:

$$\Phi_\Delta(\omega, t' + T_0^n) = (\mu_{TDC_0} - \mu_{TDC_1})(t' + T_0^n) + \mu_{TDC_1} T_1^n - \mu_{TDC_0} T_0^n + \mathcal{N}\big(0, \sigma_{TDC_0}^2 t' + \sigma_{TDC_1}^2 (t' + T_0^n - T_1^n)\big).$$

This can be further simplified to, $\forall\omega \in \Omega, \forall t' \in \mathbb{R}_{\geq 0}$:

$$\Phi_\Delta(\omega, t' + T_0^n) = (\mu_{TDC_0} - \mu_{TDC_1})t' + \sqrt{\sigma_{TDC_0}^2 + \sigma_{TDC_1}^2}\, W'_{TDC}(\omega, t') - \Phi_{TDC_1}(\omega, T_0^n). \tag{7.7}$$

Equation (7.7) demonstrates that the TDC phase difference, $\Phi_\Delta(\omega,t)$, for $t \in \mathbb{R}_{\geq T_0^n}$, can be expressed as a new Wiener process with drift:

$\mu_\Delta = \mu_{TDC_0} - \mu_{TDC_1}$, and infinitesimal variance: $\sigma^2_\Delta = \sigma^2_{TDC_0} + \sigma^2_{TDC_1}$, subtracted by the accumulated phase $\Phi_{TDC_1}(\omega, T_0^n)$ (from time T_1^n to T_0^n). This accumulated phase in `TDC1` remains independent of the Wiener process $W'_{TDC}(\omega, t')$, as this process only starts at time $t' = 0$ or $t = T_0^n$. Due to the properties of Wiener processes, phase accumulated over non-overlapping time intervals is independent.

2. $\boldsymbol{T_\Delta^n \in \mathbb{R}_{<0}}$ or $\boldsymbol{T_0^n < T_1^n}$: The clocking TDC (`TDC1`) begins oscillating last. The reasoning from the scenario $T_\Delta^n \in \mathbb{R}_{>0}$, can be repeated with a time shift $t' = t - T_1^n$. This will result in, $\forall\omega \in \Omega, \forall t' \in \mathbb{R}_{\geq 0}$:

$$\Phi_\Delta(\omega, t' + T_1^n) = (\mu_{TDC_0} - \mu_{TDC_1})t' + \sqrt{\sigma^2_{TDC_0} + \sigma^2_{TDC_1}} W'_{TDC}(\omega, t') + \Phi_{TDC_0}(\omega, T_1^n).$$

   Again, the phase accumulated in `TDC0`, $\Phi_{TDC_0}(\omega, T_1^n)$, from T_0^n to T_1^n remains independent of the Wiener process $W'_{TDC}(\omega, t')$, which starts at time $t' = 0$ or $t = T_1^n$.

3. $\boldsymbol{T_\Delta^n = 0}$ or $\boldsymbol{T_0^n = T_1^n}$: Both TDCs start oscillating simultaneously.

   Note. This is a purely theoretical scenario, as T_Δ^n follows a continuous distribution, making the probability of this case effectively zero.

   To be thorough, however, the TDC phase difference is now equal to (with a time shift $t' = t - T_0^n = t - T_1^n$), $\forall\omega \in \Omega, \forall t' \in \mathbb{R}_{\geq 0}$:

$$\Phi_\Delta(\omega, t' + T_0^n) = (\mu_{TDC_0} - \mu_{TDC_1})t' + \sqrt{\sigma^2_{TDC_0} + \sigma^2_{TDC_1}} W'_{TDC}(\omega, t') = \Phi_\Delta(\omega, t' + T_1^n).$$

   In this scenario, the subtraction involving the phase accumulated in one of the TDCs disappears, as neither TDC had been running before the second one starts.

The shifted TDC phase difference, $\left\{\Phi'_\Delta(t)\right\}_{t\in\mathbb{R}_{\geq 0}}$, is now introduced:

$$\forall\omega \in \Omega, \forall t \in \mathbb{R}_{\geq 0} : \Phi'_\Delta(\omega, t) = \Phi_\Delta\big(\omega, t + \max(T_0^n, T_1^n)\big).$$

The TDC phase difference begins at a random variable, determined by the DC time difference T_Δ^n:

$$\Phi^0_\Delta = \Phi'_\Delta(0) = \begin{cases} -\Phi_{TDC_1}(T_0^n) & \text{if } T_\Delta^n \in \mathbb{R}_{>0} \\ \Phi_{TDC_0}(T_1^n) & \text{if } T_\Delta^n \in \mathbb{R}_{<0} \\ 0 & \text{if } T_\Delta^n = 0 \end{cases}. \tag{7.8}$$

From the random process description in eq. (7.5), Φ^0_Δ *conditioned* on a realization t of T^n_Δ, is distributed as

$$\Phi^0_\Delta \mid T^n_\Delta = t \sim \begin{cases} \mathcal{N}\big(-\mu_{TDC_1} t, \sigma^2_{TDC_1} t\big) & \text{if } t \in \mathbb{R}_{>0} \\ \mathcal{N}\big(-\mu_{TDC_0} t, -\sigma^2_{TDC_0} t\big) & \text{if } t \in \mathbb{R}_{<0} \\ 0 & \text{if } t = 0 \end{cases}, \tag{7.9}$$

The TDC phase difference $\big\{\Phi'_\Delta(t)\big\}_{t\in\mathbb{R}_{\geq 0}}$ will behave as a Wiener process with drift, added to this initial phase difference. The TDCs will cease oscillating when $\Phi'_\Delta(\omega, t)$ first crosses a multiple of π at a random time T_π:

$$T_\pi : \Omega \to \mathbb{R}_{\geq 0} \text{ by } T_\pi(\omega) = \min_{t\in\mathbb{R}_{\geq 0}} \big(t \mid \Phi'_\Delta(\omega, t) = m\pi \text{ and } m \in \mathbb{Z}\big).$$

An example phase instance of the two TDCs is illustrated in fig. 7.6.

Because our focus is solely on the first passage time of $\Phi'_\Delta(\omega, t)$ at a multiple of π, the initial phase difference can be reduced modulo π:

$$\forall \omega \in \Omega, \forall t \in \mathbb{R}_{\geq 0} : \bar{\Phi}'_\Delta(\omega, t) = \big(\Phi'_\Delta(\omega, t) - \Phi^0_\Delta\big) + \Phi^0_\Delta \bmod \pi.$$

Both $\bar{\Phi}'_\Delta(\omega, t)$ and $\Phi'_\Delta(\omega, t)$ reach their first passage times at a multiple of π simultaneously: T_π.

Note. For the reduced phase difference $\big\{\bar{\Phi}'_\Delta(t)\big\}_{t\in\mathbb{R}_{\geq 0}}$, the first passage level at a multiple of π will be either zero or π: $\bar{\Phi}'_\Delta(T_\pi) = 0$ or π.

Depending on the sign of the phase drift difference μ_Δ, $\big\{\bar{\Phi}'_\Delta(t)\big\}_{t\in\mathbb{R}_{\geq 0}}$ will drift towards one of the two boundaries when $\mu_\Delta \in \mathbb{R}_{>0}$ or $\mu_\Delta \in \mathbb{R}_{<0}$, as depicted in fig. 7.6. If initiated near the opposite boundary (either 0 for $\mu_\Delta \in \mathbb{R}_{>0}$ or π for $\mu_\Delta \in \mathbb{R}_{<0}$), the phase difference may reach this boundary prematurely, ending the oscillations.

We can now establish the CDF for T_π, conditioned on a realization φ of Φ^0_Δ:

$$\begin{aligned} F_{T_\pi \mid \Phi^0_\Delta}(t \mid \varphi) &= \mathbf{P}\big[T_\pi \leq t \mid \Phi^0_\Delta = \varphi\big] = 1 - \mathbf{P}\big[T_\pi > t \mid \Phi^0_\Delta = \varphi\big] \\ &= \begin{cases} 1 - \mathbf{P}\big[0 < \bar{\Phi}'_\Delta(t) \leq \pi \mid \Phi^0_\Delta = \varphi\big] & \text{if } t \in \mathbb{R}_{\geq 0} \\ 0 & \text{otherwise} \end{cases}. \end{aligned} \tag{7.10}$$

The condition for the oscillations to continue can be expressed more explicitly as

$$0 < \mu_\Delta t + \sigma_\Delta W'_{TDC}(t) + \varphi \bmod \pi \leq \pi.$$

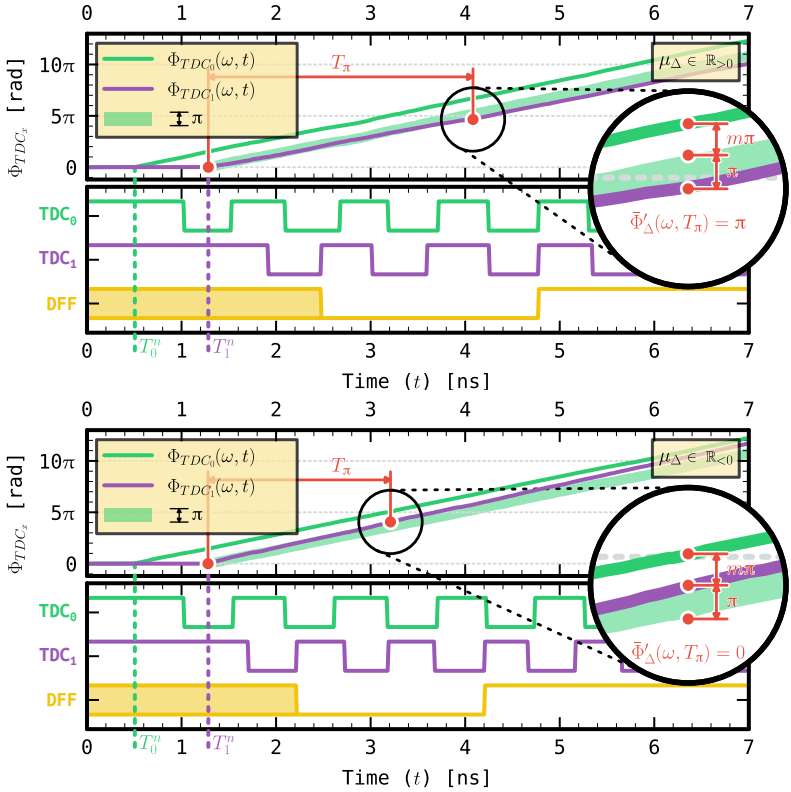

Figure 7.6: Relation between the TDC phases and the sampling time instances for $\mu_\Delta \in \mathbb{R}_{>0}$ (top) and $\mu_\Delta \in \mathbb{R}_{<0}$ (bottom).

Moving all deterministic components to the outside to obtain

$$-\frac{\mu_\Delta t + \varphi \bmod \pi}{\sigma_\Delta} < W'_{TDC}(t) \leq \frac{\pi - \mu_\Delta t - \varphi \bmod \pi}{\sigma_\Delta}. \tag{7.11}$$

From the property of Wiener processes, $W(a) \sim \sqrt{a}\mathcal{N}(0,1)$, we can rewrite the boundaries from eq. (7.11) as

$$-\frac{\mu_\Delta t + \varphi \bmod \pi}{\sigma_\Delta \sqrt{t}} < X \leq \frac{\pi - \mu_\Delta t - \varphi \bmod \pi}{\sigma_\Delta \sqrt{t}}, \tag{7.12}$$

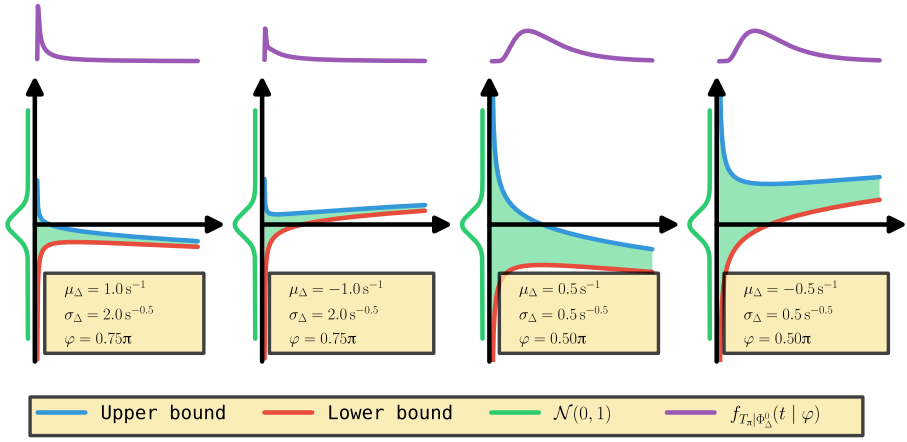

Figure 7.7: Lower and upper boundaries versus time (t) from eq. (7.12), for different μ_Δ, σ_Δ, and $\Phi^0_\Delta = \varphi$.

with $X \sim \mathcal{N}(0,1)$ (standard normal distributed). Substituting this result into eq. (7.10) gives

$$F_{T_\pi|\Phi^0_\Delta}(t \mid \varphi) = \begin{cases} 1 - \Phi_s\left(\dfrac{\pi - \mu_\Delta t - \varphi \bmod \pi}{\sigma_\Delta \sqrt{t}}\right) \\ \quad + \Phi_s\left(-\dfrac{\mu_\Delta t + \varphi \bmod \pi}{\sigma_\Delta \sqrt{t}}\right) & \text{if } t \in \mathbb{R}_{\geq 0} \\ 0 & \text{otherwise} \end{cases}.$$

Figure 7.7 illustrates the evolution of these boundaries over time for different drift difference μ_Δ, infinitesimal difference variance σ_Δ, and initial phase difference condition $\Phi^0_\Delta = \varphi$. The conditional PDF for T_π can then be derived by differentiating the CDF.

7.3.5 Output Bit Probability Distribution

The TDCs cease oscillating at the first positive edge of `TDC1` after time T_π. The number of cycles of `TDC1` up to this point, denoted as the random variable R, will then be used to construct the output random bit. This number of completed cycles is equal to

$$R : \Omega \to \mathbb{Z} \text{ by } R(\omega) = \left\lceil \frac{\Phi_{TDC_1}\left(\omega, T_\pi + \max(T_0^n, T_1^n)\right)}{2\pi} \right\rceil. \tag{7.13}$$

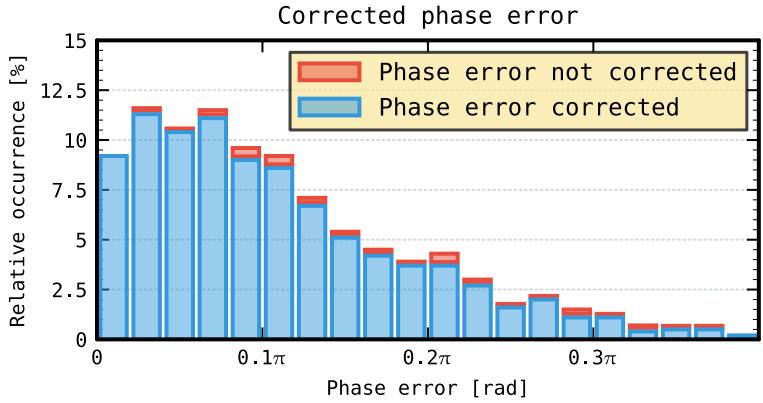

Figure 7.8: Histogram of 1000 repeated simulations illustrating the absolute phase error in $\Phi_{TDC_1}(\omega, t)$ when employing the simplified relation in eq. (7.14). Typically, this phase error is corrected by applying the ceil operation. The simulations were conducted assuming a linear jitter strength of $s_{noise} = 30\,\text{fs}$.

The term $\max(T_0^n, T_1^n)$ is added to T_π, as T_π was defined for the shifted phase difference. This term accounts for the accumulated phase in the scenario `TDC1` was started first.

Note. The range of the random cycle count, R, includes negative values. However, for practical jitter strengths where $s_{noise} \ll 1\,\text{s}$, the probability of R being negative is negligible, and the range effectively covers only positive counts: $R : \Omega \to \mathbb{N}$.

Note. There exists a dependency between T_π and the Wiener process determining $\left\{\Phi_{TDC_1}(t)\right\}_{t \in \mathbb{R}_{\geq T_1^n}}$, complicating the derivation of an *analytical* expression for the distribution of R. To address this, the noise contributing to $\left\{\Phi_{TDC_1}(t)\right\}_{t \in \mathbb{R}_{\geq T_1^n}}$ is neglected: $\Phi_{TDC_1}(\omega, t) \approx \mu_{TDC_1}(t - T_1^n)$. This simplification is justified because the jitter strength in the phase difference signal is significantly greater than in a single TDC: $\frac{\sigma_\Delta}{\mu_\Delta} \gg \frac{\sigma_{TDC_1}}{\mu_{TDC_1}}$. This condition holds well for closely matched TDCs: $\mu_{TDC_0} \approx \mu_{TDC_1}$ and $|\mu_\Delta| \ll \mu_{TDC_1}$.

Simulation results depicted in fig. 7.8 provide additional validation for the simplification, as the introduced error in the phase process $\Phi_{TDC_1}(\omega, t)$ is minimal. The relative error (average deviation for R) is 0.2 %.

Using this simplification, eq. (7.13) can be rewritten as

$$R = \left\lceil \frac{\mu_{TDC_1}\left(T_\pi + \max(T_0^n, T_1^n) - T_1^n\right)}{2\pi} \right\rceil. \tag{7.14}$$

Depending on the sign of T_Δ^n, we have

$$R = \begin{cases} \left\lceil \frac{\mu_{TDC_1}(T_\pi + T_\Delta^n)}{2\pi} \right\rceil & \text{if } T_\Delta^n \in \mathbb{R}_{>0} \\ \left\lceil \frac{\mu_{TDC_1} T_\pi}{2\pi} \right\rceil & \text{if } T_\Delta^n \in \mathbb{R}_{\leq 0} \end{cases}.$$

The term $\mu_{TDC_1} T_\Delta^n$ can be more accurately replaced by $-\Phi_\Delta^0$ for $T_\Delta^n \in \mathbb{R}_{>0}$ from eq. (7.8), as this term represents the accumulated phase in `TDC1` before `TDC0` was started. The conditional CDF for R can now be calculated as, $\forall r \in \mathbb{N}$:

$$\begin{aligned} F_{R|\Phi_\Delta^0, T_\Delta^n}(r \mid \varphi, t) &= \mathbf{P}\left[R \leq r \mid \Phi_\Delta^0 = \varphi, T_\Delta^n = t\right] \\ &= \begin{cases} \mathbf{P}\left[T_\pi \leq \frac{2\pi r + \varphi}{\mu_{TDC_1}} \mid \Phi_\Delta^0 = \varphi\right] & \text{if } t \in \mathbb{R}_{>0} \\ \mathbf{P}\left[T_\pi \leq \frac{2\pi r}{\mu_{TDC_1}} \mid \Phi_\Delta^0 = \varphi\right] & \text{if } t \in \mathbb{R}_{\leq 0} \end{cases}. \end{aligned}$$

R is a discrete random variable, $\forall r \in \mathbb{N}$:

$$\begin{aligned} f_{R|\Phi_\Delta^0, T_\Delta^n}(r \mid \varphi, t) &= \mathbf{P}\left[R = r \mid \Phi_\Delta^0 = \varphi, T_\Delta^n = t\right] \\ &= \begin{cases} F_{R|\Phi_\Delta^0, T_\Delta^n}(r \mid \varphi, t) - F_{R|\Phi_\Delta^0, T_\Delta^n}(r-1 \mid \varphi, t) & \text{if } r \in \mathbb{N}_{\neq 0} \\ F_{R|\Phi_\Delta^0, T_\Delta^n}(0 \mid \varphi, t) & \text{if } r = 0 \end{cases}. \end{aligned}$$

Removing the conditionals, to obtain the joint distribution:

$$f_{R,\Phi_\Delta^0, T_\Delta^n}(r, \varphi, t) = f_{R|\Phi_\Delta^0, T_\Delta^n}(r \mid \varphi, t) f_{\Phi_\Delta^0 | T_\Delta^n}(\varphi \mid t) f_{T_\Delta^n}(t),$$

with $f_{T_\Delta^n}(t)$ and $f_{\Phi_\Delta^0 | T_\Delta^n}(\varphi \mid t)$ obtained from eqs. (7.4) and (7.9), respectively. The random variables Φ_Δ^0 and T_Δ^n are integrated out to obtain the distribution for R:

$$f_R(r) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f_{R,\Phi_\Delta^0, T_\Delta^n}(r, \varphi, t) \mathrm{d}t \mathrm{d}\varphi.$$

The produced random bit, B, is now defined as the LSB of R. Therefore, the bit probability can be calculated as

$$\forall b \in \{0, 1\} : \mathbf{P}[B = b] = \sum_{i=0}^{\infty} f_R(2i + b).$$

Because the system does not retain a state between bit generations, individual bits are inherently IID, assuming that thermal noise is the sole source of RO jitter.

7.4 Jitter Strength Measurement

The entropy estimate provided by the model in section 7.3 is significantly influenced by the platform-dependent parameter known as jitter strength (s_{noise}). This parameter dictates the rate at which timing jitter accumulates in a free-running RO. Unlike design parameters (e.g., RO frequency), the jitter strength cannot be directly measured or controlled. As proposed by [94], jitter measurements should be conducted on-chip and preferably with a *differential* measurement set-up to minimize external (potentially manipulable) influences that might lead to an overestimation of the available timing jitter.

It is crucial not to overestimate the jitter strength parameter and to use a conservative method for two key reasons. Firstly, measurement errors, such as external noise sources other than thermal noise, will always introduce a positive bias. This happens because the jitter strength is determined based on observed measurement variance, and any external, independent sources of error will increase this variance (adding two independent random variables will increase variance), leading to overestimation. Secondly, the estimated output entropy forms the basis of a *security* claim. Overestimating the jitter strength results in an overestimation of the produced output entropy, potentially invalidating the security claim.

To ensure accuracy, the jitter measurement experiment was repeated on five separate devices (chips). The most conservative estimate from these tests will be used to estimate the entropy for all devices.

7.4.1 On-chip Measurement Set-up

The proposed ES architecture in section 7.2 also supports on-chip differential jitter strength measurement. A circuit diagram and timing diagram for this measurement are shown in figs. 7.9 and 7.10. By configuring `TDC0` and `TDC1` to have long and short oscillation periods respectively ($p_{TDC_0} > 2p_{TDC_1}$), a positive edge of `TDC1` will occur each half-period of `TDC0`. Each half-period of `TDC0` is sampled, causing the TDCs to stop oscillating once both DCs have finished propagating. `DC0` and `DC1` are configured so that `DC1` has a shorter *propagation delay* than `DC0` ($T_0^n > T_1^n$). Thus, `TDC1` oscillates during the interval when `DC1` has finished propagating but `DC0` has not. A counter records the number of `TDC1` oscillations during this interval, producing an output proportional to the propagation delay difference between `DC0` and `DC1`. By analyzing the counter output variance over multiple evaluations, an estimation

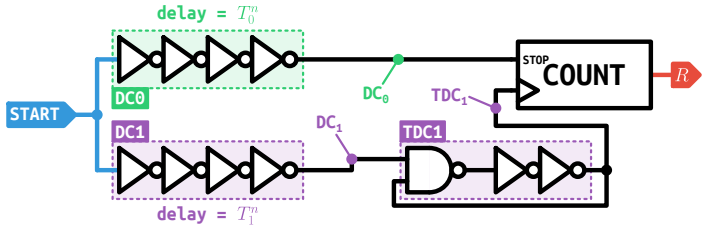

Figure 7.9: Jitter measurement circuit architecture.

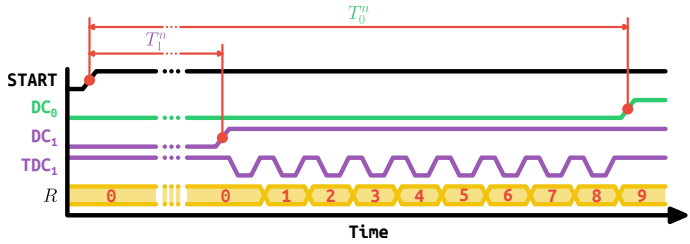

Figure 7.10: Jitter measurement timing diagram.

of the differential DC propagation variance, and therefore the available jitter strength in DC0 and DC1, is obtained.

7.4.2 Theoretical Jitter Analysis

Based on the stochastic model from section 7.3, an estimate for the observed counter output variance can be made dependent on the jitter strength value, s_{noise}. In this chapter, we select the highest value for s_{noise} that still results in an underestimation of the observed variance, using this as the final jitter strength estimate. The timing jitter accumulated by TDC1 also impacts the counter output variance. Therefore, the model from section 7.3 is extended here to estimate the counter output variance accurately.

The DC timing difference distribution, T_Δ^n, is given by eq. (7.3). Due to a hardware constraint, the TDCs are only permitted to stop oscillating after both have completed two full periods. Consequently, the jitter accumulation time interval is given by

$$T_A^n = T_\Delta^n + T_{2TDC_0},$$

where T_{2TDC_0} is a random variable representing the time required for TDC0 to oscillate for two full periods. According to section 7.3, T_{2TDC_0} follows an

inverse-Gaussian distribution:

$$T_{2TDC_0} \sim \mathcal{IG}\left(\frac{4\pi}{\mu_{TDC_0}}, \left(\frac{4\pi}{\sigma_{TDC_0}}\right)^2\right).$$

During the accumulation time interval T_A^n, `TDC1` will oscillate. Assuming it starts with zero phase, the phase of `TDC1` at the end of this interval, denoted as Φ_J, conditioned on the length of the accumulation time interval, follows a Gaussian distribution:

$$\forall t_a \in \mathbb{R}_{\geq 0} : (\Phi_J \mid T_A^n = t_a) = \Phi_{TDC_1}(t_a) \sim \mathcal{N}\left(\mu_{TDC_1} t_a, \sigma^2_{TDC_1} t_a\right).$$

The condition to the accumulation time interval length can be removed in a manner similar to what was done in section 7.3:

$$f_{\Phi_J}(\varphi) = \int_0^\infty \frac{1}{\sigma_{TDC_1}\sqrt{t_a}} \phi_s\left(\frac{\varphi - \mu_{TDC_1} t_a}{\sigma_{TDC_1}\sqrt{t_a}}\right) f_{T_A^n}(t_a) \mathrm{d}t_a.$$

The TDC oscillations will only stop after a positive edge occurs in `TDC1`. Consequently, all phases of `TDC1` within the interval $\left(2\pi(r-1), 2\pi r\right]$ will result in the same counter output: r. The probability of the counter output R being equal to a value r (PMF) is then given by

$$\forall r \in \mathbb{N}_{\neq 0} : f_R(r) = \mathbf{P}[R = r] = \int_{2\pi(r-1)}^{2\pi r} f_{\Phi_J}(\varphi) \mathrm{d}\varphi.$$

From this result, the variance $\mathbf{Var}[R]$ can be calculated and compared with the measured values.

7.4.3 Measurement Results

Figure 7.11 presents the jitter measurement results for five different devices. The experiment was conducted with four different DC accumulation time differences, T_Δ^n, determined by the parameter: $n \in \{1, 2, 4, 8\}$. For each device, 100 sets of 2^{16} counter outputs were collected. The counter output variance, $\mathbf{Var}[R]$, was calculated from the 2^{16} samples. Each device at each value of n is represented as a box plot, showing the experimental distribution for $\mathbf{Var}[T_0^n]$, derived from measuring $\mathbf{Var}[R]$ and using the model from section 7.4.2, over the 100 repeated measurements. The straight lines in fig. 7.11 represent the theoretical `DCO` period length variance, $\mathbf{Var}[T_0^n]$, for different jitter strength magnitudes, s_{noise}, calculated as: $\mathbf{Var}[T_0^n] = s_{noise}\mathbf{E}[T_0^n]$. A conservative estimate of 30 fs was obtained.

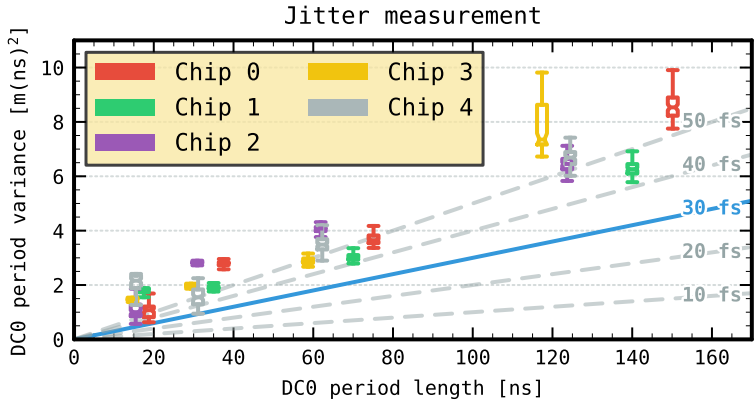

Figure 7.11: Jitter measurement results.

7.5 Design Parameter Selection Criteria

The proposed ES design features four design parameters that can be freely chosen by the designer: μ_{DC_0}, μ_{DC_1}, μ_{TDC_0}, and μ_{TDC_1}. The infinitesimal variances (σ_w for $w \in \{DC_0, DC_1, TDC_0, TDC_1\}$) are linked to the phase drifts by the determined jitter strength and eq. (7.1). Following the strategy outlined in section 2.4.2 in chapter 2, this section provides a selection strategy for these four design parameters.

7.5.1 Pipeline Balance

As with all pipelined architectures, it is necessary to balance the propagation times for both stages. The average propagation delay of the DC stage is given by:

$$d_{DC} = \max\big(\mathbf{E}[T_0^n], \mathbf{E}[T_1^n]\big).$$

The slowest DC will dictate when the TDCs can begin resolving the timing difference. The maximal TDC resolving time is determined by the TDC resolution (res), which is defined as

$$res = |p_{TDC_0} - p_{TDC_1}| = \left| \frac{2\pi}{\mu_{TDC_0}} - \frac{2\pi}{\mu_{TDC_1}} \right|.$$

Each period of `TDC0`, the positive edge of `TDC1` shifts by an amount of res relative to the positive edge of `TDC0`. The TDCs will stop oscillating as soon

as TDC1 samples a different value from TDC0. This implies that at most $\frac{p_{TDC_0}}{2res}$ cycles of TDC1 are required. The maximal TDC resolving time is then given as

$$d_{TDC} = \frac{p_{TDC_0} p_{TDC_1}}{2res}.$$

To ensure the TDCs finish resolving before the DCs have accumulated enough jitter for the next output bit, the TDC resolving time must be shorter than the maximal DC propagation delay: $d_{TDC} < d_{DC}$. This *constraint* sets an upper limit on the TDC resolution:

$$res > \frac{p_{TDC_0} p_{TDC_1}}{2 \max\left(\mathbf{E}[T_0^n], \mathbf{E}[T_1^n]\right)}. \tag{7.15}$$

7.5.2 Entropy Density

According to [39], a minimum Shannon *entropy density* of 0.997 bit per output bit is required, when no additional post-processing is added. The stochastic model from section 7.3 is used to determine the theoretical entropy density at the output. Linear timing jitter, originating from thermal (white) FM noise, accumulates proportionally to the square root of the accumulation time (addition of independent variances). A maximum TDC resolution size is needed to extract the required entropy from the accumulated DC timing jitter. This requirement leads to an upper bound on the TDC resolution, given as

$$res < \alpha \sqrt{s_{noise} \max\left(\mathbf{E}[T_0^n], \mathbf{E}[T_1^n]\right)}, \tag{7.16}$$

with the constant α related to the required entropy density and the shape of the accumulated timing jitter distribution. The value of α can be determined by evaluating the model from section 7.3 for various DC accumulation times and identifying the upper bound on the required resolution.

Figure 7.12 presents the model results. It is evident that the obtained value of α is not perfectly constant. To ensure that eq. (7.16) consistently produces a valid upper bound for the TDC resolution, a lower bound (indicated by the horizontal line in fig. 7.12) is selected. The value of α used throughout this chapter equals 1.94.

7.5.3 ES Throughput

The ES throughput is related to the DC accumulation time:

$$throughput = \frac{1}{\max\left(\mathbf{E}[T_0^n], \mathbf{E}[T_1^n]\right)}.$$

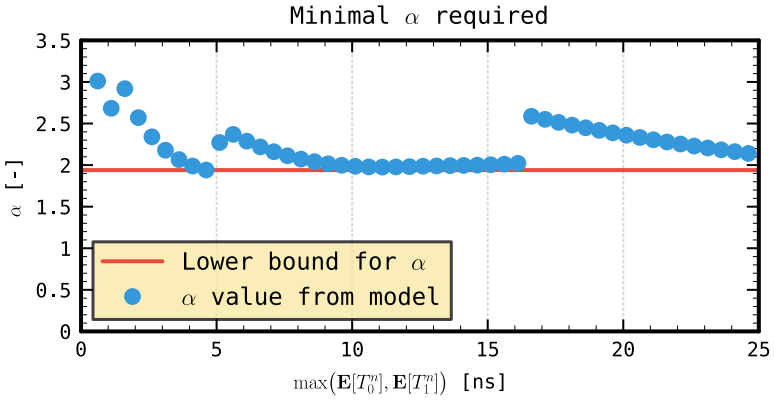

Figure 7.12: Minimal α required for eq. (7.16) to produce a high enough entropy density, with $s_{noise} = 30\,\text{fs}$.

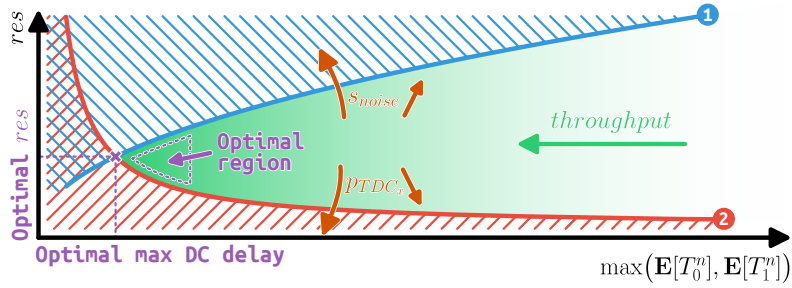

Figure 7.13: TDC resolution versus DC accumulation time optimization.

A sketch illustrating both constraints for the TDC resolution versus the DC accumulation time is shown in fig. 7.13. The top curve (1) and bottom curve (2) represent eqs. (7.16) and (7.15), respectively. Valid values for the TDC resolution are indicated by the shaded region between the two constraint curves. Higher ES throughput favors points more to the left of the graph. The optimal resolution/accumulation time point is at the intersection of both constraint curves. To ensure a sufficiently stable ES implementation, a margin from the constraint borders is necessary, which is indicated by the optimal region in fig. 7.13.

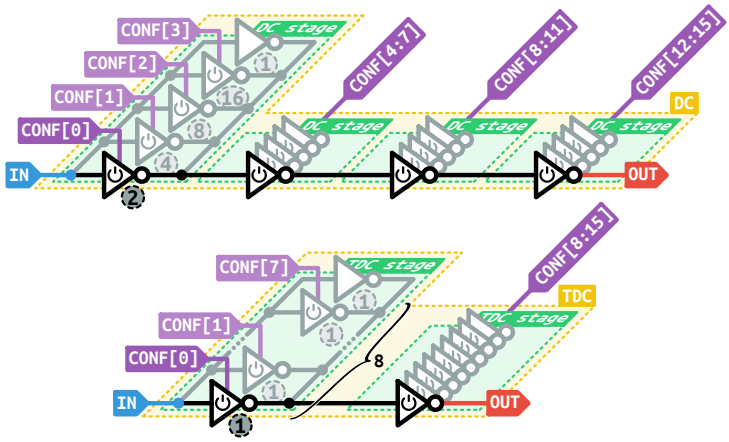

Figure 7.14: Detailed DC/TDC architecture.

7.5.4 Delay Control Circuit

To optimize throughput, a precise control over the DC accumulation time ($\mathbf{E}[T_0^n]$ and $\mathbf{E}[T_1^n]$) and the TDC resolution is essential. The DC and TDC implementation leverage the `Exp4×4` and `Lin2×8` RO topologies, respectively, introduced in section 5.3. Figure 7.14 illustrates the circuit breakdown, including the CSI sizing, for both the DC and TDC ROs.

Both DC and TDC ROs feature 16 configuration bits each, controlled by an off-chip controller circuit. With the architecture depicted in fig. 7.14, a configuration within the optimal region was consistently achievable across all tested devices.

7.6 Experimental Results

Five devices with the proposed ES architecture were fabricated using a 28 nm CMOS technology. All measurements are conducted under standard conditions: 20 °C ambient temperature and 0.9 V supply voltage unless specified otherwise. Each device's DC and TDC ROs were configured to operate within the optimal region, as detailed in section 7.5. This involved scanning through various DC and TDC RO frequencies to select the most suitable combination for each device.

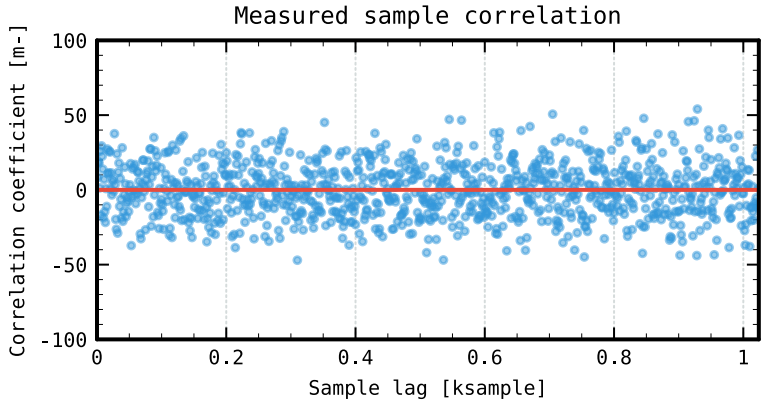

Figure 7.15: Measured sample correlation for 4096 samples, obtained from chip 0.

7.6.1 IID Claim Verification

As stated in section 7.3.5, the output bits are inherently IID under the assumption that only thermal FM noise affects the DC and TDC ROs. To validate this assertion, two experiments were conducted: a correlation analysis of the generated counter values (R) and the NIST SP 800-90B IID test [83].

Correlation Analysis The sample correlation coefficient of 4096 consecutively generated counter samples (realizations of R) from chip 0 is computed for sample lags ranging from 1 to 1024. The sample correlation coefficient is calculated using the formula:

$$correlation(lag) = \frac{\sum_{i=1}^{3072}(r_i - \bar{r})(r_{i+lag} - \bar{r})}{\sqrt{\sum_{i=1}^{3072}(r_i - \bar{r})^2 \sum_{i=1}^{3072}(r_{i+lag} - \bar{r})^2}},$$

with r_i the i-th generated sample and $\bar{r}$ the sample mean. The results depicted in fig. 7.15 show no significant sample correlation, which further supports the assumption that the generated samples are IID.

NIST SP 800-90B IID Test All five devices successfully pass the NIST SP 800-90B IID test, using 1 Mbit of consecutively generated bits.

Table 7.1: Min-entropy estimates, smallest value in bold.

Chip	Model [bit]	Test [bit]
0	0.999 88	**0.933 41**
1	0.998 61	**0.944 75**
2	0.998 11	**0.947 22**
3	0.998 95	**0.952 55**
4	0.999 63	**0.962 21**

7.6.2 Entropy Validation

As imposed by [39], the estimated min-entropy should be greater than 0.91 bit per output bit (equivalent to 0.997 bit of Shannon entropy), when no further post-processing is utilized. In section 7.5, the ES design parameters were chosen to achieve at least 0.91 bit of min-entropy per output bit, with higher entropy achievable at the expense of a reduced throughput. Table 7.1 provides an overview of the min-entropy estimates for all five tested devices, based on 1 Mbit of consecutive data collected under nominal conditions. Each device meets the required min-entropy level. The entropy estimate derived from the NIST SP 800-90B test, also using 1 Mbit of data, is more conservative compared to the stochastic model estimate, which aligns with the tendency of these tests to underestimate the available entropy [75]. The counter output R serves as a health metric, indicating a potential entropy reduction.

7.6.3 Power and Throughput

All five devices tested achieve a throughput exceeding 250 Mbit s^{-1} under nominal conditions, as illustrated in the left graph of fig. 7.16. *Process variations* in the DC/TDC ROs can cause performance differences among devices. One device (chip 1) was extensively tested across various voltage conditions. The experimental results, shown in the middle graph of fig. 7.16, indicate that for all tested supply voltage levels, the output bit entropy remained above the 0.91 bit threshold per output bit.

The right graph in fig. 7.16 displays the power consumption breakdown and *energy efficiency* per generated bit. The best energy efficiency is achieved at a 0.8 V supply, with 1.46 pJ bit^{-1}, which is lower than previously reported. The power breakdown reveals that at nominal conditions, the Core, DC, and TDC

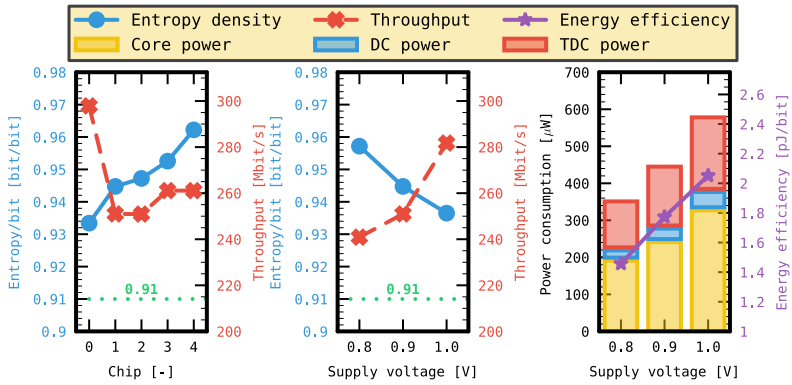

Figure 7.16: Measurement results.

consume 54.2 %, 8.2 %, and 37.6 % of the total power, respectively. The Core module includes the digitization and synchronization circuitry.

7.7 Conclusion and Comparison

7.7.1 Comparison

Compared to previous work listed in table 7.2, the proposed design achieves the best energy efficiency and the second-best *area efficiency* (throughput generated per unit of normalized area). The jitter pipelining architecture, combined with high TDC time resolution, allows for high throughput with modest area and power requirements. A chip photo is shown in fig. 7.17. The ES circuitry, comprising the DC, TDC, and Core, occupies 750.7 µm^2. Additional configuration FFs, to store the DC/TDC configuration (`Conf`), and interfacing logic (`Send`) are included for device measurement.

7.7.2 Conclusion

The proposed ES architecture was designed and verified using an approach aligned with modern standards. Leveraging the digital nature of the circuits used, this design benefits from the advantages of digital CMOS, such as scaling and integration. A stochastic model is introduced to estimate the output bit entropy, along with an on-chip jitter measurement methodology to quantify the

Table 7.2: Comparison with previous work.

	This work	*ISSCC* 2021 [81]	*ISSCC* 2017 [40]	*JSSC* 2016 [96]	*JSSC* 2012 [54]	*Cryptogr.* 2021 [42]
Technology [nm]	28	28	65	40	45	65
Entropy source	Edge jitter	SRAM leakage jitter	Edge jitter	Edge jitter	Meta-stability	Edge jitter
Stochastic model available	✓	✗	✗	✓	✓	✓
All digital	✓	✓	✗	✓	✗	✓
Area [kF^2]	957.5	**36**[(a)]	218	522.5	1977	59.2
Max throughput [Mbit s^{-1}]	298	3.6	9.9	2	**2400**	8.27
Best energy efficiency [pJ bit^{-1}]	**1.46**	9.6	35.5	11	2.9	128.2
Best area efficiency [bit s^{-1} F^{-2}]	311.2	100	45.4	3.83	**1214**	139.7
Supply voltage range [V]	0.8 - 1.0	0.8 - 1.0	1.08 - 1.2	0.6 - 0.9	0.28 - 1.35	-

[(a)] SRAM area not included

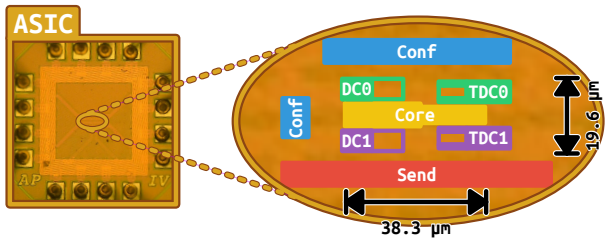

Figure 7.17: Chip photo with zoomed in region on the ES area.

jitter strength platform parameter. An optimization scheme guides the selection of design parameters, ensuring maximal throughput for the given platform parameters. The jitter pipelining structure enables efficient on-chip entropy generation in terms of both area and energy usage.

Chapter 8

Conclusion

The final chapter of this dissertation summarizes the key contributions presented in this work and discusses the implications of the obtained results. Additionally, it provides directional suggestions for further research in this field.

8.1 Overview of Contributions

For a detailed summary of the contributions presented in the different chapters, please refer to the conclusion section of each respective chapter. This section instead highlights several *global research trends* in the field of TRNG design and explains how this dissertation has contributed to these trends.

Trend 1: Increased Awareness of Other Oscillator Noise Types The noticeable increase in the rate of publications [7, 8, 18] addressing oscillator *noise types* other than the established *thermal* FM noise, mainly *flicker* FM noise, highlights the growing importance of this topic in the TRNG and ES design community. To the best of our knowledge, the work presented in part I of this dissertation is the first to analyze and describe the influence of *noise sources* other than thermal FM noise on an ASIC platform in a TRNG context. This new description of flicker FM noise enables a more profound analysis of the different ways in which *entropy* gets generated in oscillator-based ESs.

Trend 2: Reduced Design Effort The days of designers endlessly tinkering with the ES implementation for only marginal performance gains are now behind us. Additionally, these same ESs were often very susceptible to changing *operating conditions*, rapidly degrading with changes in supply voltage [11] or

surrounding temperature [79]. By introducing a *configurable* approach, the work presented in part II of this dissertation allows for the construction of *modern* ESs that provide increased *robustness* against environmental changes while simultaneously reducing the *design effort* required for correct implementation.

Trend 3: Increased Performance, While Adhering to Standards Recently published ES designs [24, 42] continue to push the performance boundaries, while simultaneously remaining compatible with international standards on TRNG design. Mainly two figures of merit are distinguished in this thesis: generated *entropy rate* per required area unit, and produced entropy per required unit of energy. Certainly, depending on the application, individual metrics such as: area usage, power consumption, or throughput can be the main topic of optimization as well. The work presented in chapter 7 contributes to this trend by proposing an ES architecture, achieving the highest (at the time of publishing) energy efficiency, and additionally providing a detailed *stochastic model* description.

8.2 Further Research Pathways

For each of the trends indicated in the previous section, some potential directions for further research are provided.

Trend 1: Increased Awareness of Other Oscillator Noise Types Looking ahead, a more in-depth analysis of the *stopping times* for oscillator phase processes (the time required for the phase to hit a certain level), particularly under the influence of noise types beyond thermal FM noise, will be crucial for providing a detailed description of the distribution and dependency of consecutive oscillator period lengths. Additionally, the flicker FM noise analysis should be extended across a wider range of ES architectures. For example, for the STR-ES [13], we can validly question whether assumptions of *independence* for all individual events still hold true.

Moreover, despite existing measurements of the flicker FM *noise magnitude* [8, 21, 27, 46], the wide variation in reported values necessitates further investigation. Gathering a larger number of measurements across diverse hardware platforms will be essential to fully comprehend the behavior of flicker FM noise and its suitability for entropy generation.

Trend 2: Reduced Design Effort The results presented in chapter 6 demonstrated that using configurable RO topologies, the ES can be fully described using an HDL alone, significantly reducing the design effort required on the FPGA platform. The question now arises if the same principle is possible on an ASIC hardware platform as well. Advancing further in the direction of

reducing design effort, the development of a first *standard-cell* ASIC ES design should not be far off. Additionally, further attention can be given to applying the proposed techniques of RO configurability to non-oscillator-based ESs, such as an inverter *metastability*-based design [87], as well.

Trend 3: Increased Performance, While Adhering to Standards The trend of chasing after ever-increasing TRNG performance will continue to motivate engineers to publish novel architectures and concepts, especially within more circuit-oriented communities. While this trend is to be encouraged, *safeguards* should remain in place to prevent this pursuit of performance gains from leading to unintended side effects:

- Ensure that the continuous pursuit of performance gains does not result in needlessly complex topologies, which are hard to analyze and could significantly increase the design effort.
- Ensure that future proposed designs consider resistance to PVT variations from the design phase, in addition to performance.
- Ensure that no severe compromises affecting the security of the proposed designs are made in exchange for performance gains.

While history has proven numerous times that expecting no future TRNG design to wander outside these safeguards is almost foolish, we can find reassurance in the guiding influence of international *standardization bodies* today, steering the trajectory of TRNG design toward a promising future.

Appendix A

Mathematical Framework

This appendix offers essential mathematical background knowledge. Most of the concepts introduced here are extensively covered in [36]. The notation has been slightly adapted to accommodate the requirements of this thesis.

A.1 Notation and Definitions

A.1.1 Notation

Table A.1 displays the notation for the mathematical objects utilized in this thesis. Both Latin and Greek symbols are used interchangeably. When the dimension of an object is not specified, scalar notation is employed. Sets are denoted by capital symbols, and within the given context, there should be no potential confusion with the notation for random variables.

The notation: $\mathbb{R}_{\geq a}$, signifies the subset of real numbers greater than or equal to a: $\{x \in \mathbb{R} \mid x \geq a\}$. Similarly, this notation is applied to the relations: $>$, $\leq$, $<$, and $\neq$ as well, as to the sets of integers, $\mathbb{Z}$, and natural numbers, $\mathbb{N}$. A finite range of integers is denoted as $\mathbb{N}_n = \{i \in \mathbb{N} \mid i < n\} = \{0, 1, \ldots, n-1\}$. The Cartesian power of a set is represented as follows: $\bigtimes_{i=0}^{n-1} S = S^n$ and the power set, 2^S, equals the set containing all subsets of S, including the empty set, $\emptyset$, and S itself. A random variable, A, distributed according to some parameterized distribution, $\mathcal{D}(p_0, p_1, \ldots p_n)$ is denoted as $A \sim \mathcal{D}(p_0, p_1, \ldots, p_n)$. The operator

Table A.1: Notation of mathematical objects.

	Deterministic	Random
Scalar	a	A
Vector	$\vec{\boldsymbol{a}}$	$\vec{\boldsymbol{A}}$
Matrix	$\boldsymbol{A}$	$\mathbb{A}$
Matrix element	$A_{i,j}$	$\mathbb{A}_{i,j}$
Function	$a(t)$	$\{A(t)\}_{t\in T}$
Measurement	$\hat{a}$	$\hat{A}$

$\cdot \mod a$, is shorthand notation for $\cdot - \left\lfloor \frac{\cdot}{a} \right\rfloor a$, or the positive remainder after division by a.

A.1.2 Elementary Functions

This subsection defines elementary functions and operators used in this thesis.

Indicator Function

The set indicator function assigns a zero, one to elements that are not in, are in a given subset.

Definition A.1. (Indicator function) Given a set and subset: $S \subseteq T$, the indicator function, $1_S : T \to \{0, 1\}$, is defined as

$$\forall t \in T : 1_S(t) = \begin{cases} 1 & \text{if } t \in S \\ 0 & \text{if } t \notin S \end{cases}.$$

Window Function

The window function only equals one in an interval of length Δ.

Definition A.2. (Window function) The window function on the real line: $w_\Delta : \mathbb{R} \to \{0, \frac{1}{2}, 1\}$, is defined as

$$\forall t \in \mathbb{R} : w_\Delta(t) = \begin{cases} 1 & \text{if } |t| \in \mathbb{R}_{<\frac{\Delta}{2}} \\ \frac{1}{2} & \text{if } |t| = \frac{\Delta}{2} \\ 0 & \text{otherwise} \end{cases}.$$

Cosine Integral

The cosine integral is used to model an oscillator influenced by flicker Frequency Modulated (FM) noise throughout this thesis.

Definition A.3. (Cosine integral) The cosine integral function, $\mathrm{Ci} : \mathbb{R}_{>0} \to \mathbb{R}$, is defined as

$$\forall x \in \mathbb{R}_{>0} : \mathrm{Ci}(x) = -\int_x^\infty \frac{\cos(\theta)}{\theta} \mathrm{d}\theta.$$

We have the following property: $\lim_{x\to\infty} \mathrm{Ci}(x) = 0$ and the *Taylor* expansion of $\mathrm{Ci}(x)$ around $x = 0$ is given as [88]:

$$\mathrm{Ci}(x) = \gamma + \ln(x) + \sum_{k=1}^{\infty} \frac{\left(-x^2\right)^k}{2k(2k)!},$$

with $\gamma \approx 0.577$ representing the *Euler-Mascheroni* constant.

Fourier Transform

The Fourier Transform (FT) of a scalar function breaks down the function into its frequency composition.

Definition A.4. (FT) The FT of a scalar function, $a : \mathbb{R} \to \mathbb{C}$, results into a different scalar function, $c(f) = \mathcal{F}\{a(t)\} : \mathbb{R} \to \mathbb{C}$, defined as

$$\forall f \in \mathbb{R} : c(f) = \mathcal{F}\{a(t)\} = \int_{-\infty}^{\infty} a(t) \mathrm{e}^{-\mathrm{j}2\pi f t} \mathrm{d}t,$$

with j the imaginary unit.

The inverse FT is similarly defined.

Definition A.5. (Inverse FT) The inverse FT of a scalar function, $c : \mathbb{R} \to \mathbb{C}$, results into a different scalar function, $a(t) = \mathcal{F}^{-1}\{c(f)\} : \mathbb{R} \to \mathbb{C}$, defined as

$$\forall t \in \mathbb{R} : a(t) = \mathcal{F}^{-1}\{c(f)\} = \int_{-\infty}^{\infty} c(f) \mathrm{e}^{\mathrm{j}2\pi t f} \mathrm{d}f.$$

It can be shown that for a continuous function, a: $\mathcal{F}^{-1}\Big\{\mathcal{F}\{a(t)\}\Big\} = a(t)$, when a and $\mathcal{F}\{a\}$ are absolutely integrable. The functions, $a(t)$ and $c(f) = \mathcal{F}\{a(t)\}$ form a FT pair.

A.2 Fundamentals of Probability Mathematics

This section offers prerequisite knowledge on probability mathematics and measure theory.

A.2.1 Probability Space

A probability space provides a framework for managing real-world scenarios where outcomes cannot be predicted with absolute certainty.

Definition A.6. (Probability space) A probability space is a measure space, $(\Omega, \mathcal{F}, \mathbf{P})$, having the following three elements:

- A sample space: Ω, the set containing all possible outcomes.
- A σ-algebra in Ω: $\mathcal{F}$, a set containing subsets of Ω. Each subset of Ω is called an event.
- A probability measure: $\mathbf{P} : \mathcal{F} \to [0, 1]$, a countably additive set function on $\mathcal{F}$. This function assigns a probability to each element of $\mathcal{F}$.

A.2.2 Random Variable

Random variables represent mappings from the space of possible outcomes to a measurable space.

Definition A.7. (Random variable) An $\mathcal{A}$-valued random variable, A, is a measurable function from the sample space, Ω, to a measurable space: $(\mathcal{A}, \Sigma_A)$: $A : \Omega \to \mathcal{A}$.

Note. The set: $\mathcal{A}$ contains all possible realizations of A, this set is often referred to as the alphabet of A.

Note. In this thesis, the measurable space of interest is the set of real numbers (vectors), or a union of subintervals thereof. For continuous random variables (vectors), the corresponding measurable space is $\big(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n)\big)$, with $\mathcal{B}(\mathbb{R}^n)$, the Borel σ-algebra. For discrete random variables (vectors), the measurable space consists of some discrete subset, $\mathcal{A} \subset \mathbb{R}^n$, containing a finite or infinitely countable number of elements and a corresponding σ-algebra, $\Sigma_A \subseteq 2^{\mathcal{A}}$.

Definition A.8. (Real-valued random variable) A real-valued random variable, A, is a measurable function from the sample space, Ω, to a subset of the real numbers: $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^n$.

Note. Random variables mapping to the set of real numbers are called real-valued random variables. In this thesis, only real-valued random variables are considered and the prefix *real-valued* is therefore further omitted.

Note. For a random variable, A, a set of realizations, $S \in \Sigma_A$, can be related to the event: $A^{-1}(S) = \{\omega \in \Omega \mid A(\omega) \in S\}$. The notation: $\mathbf{P}[A \in S] = A_*(\mathbf{P})[S] = \mathbf{P}\big[A^{-1}(S)\big]$, is used to denote the probability of observing a realization in S. The push-forward measure of $\mathbf{P}$ on $(\mathcal{A}, \Sigma_A)$ by A, is denoted by $A_*(\mathbf{P})$. When the set: S, contains only one element, e.g. $S = \{a\}$, the notation: $\mathbf{P}[A = a] = \mathbf{P}\big[A \in \{a\}\big]$, is used as well.

A.2.3 Probability Density/Mass Function

The Probability Density Function (PDF) for a continuous or Probability Mass Function (PMF) for a discrete random variable assigns a positive real value to each realization.

Definition A.9. (PDF/PMF) The PDF/PMF for an $\mathcal{A}$-valued random variable (scalar, vector or matrix), $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^{n \times m}$, is a measurable function, $f_A : \mathcal{A} \to \mathbb{R}_{\geq 0}$, with the following property:

$$\forall S \in \Sigma_A : \mathbf{P}[A \in S] = \int_{A^{-1}(S)} \mathbf{P}[\mathrm{d}\omega] = \int_S f_A(a)\mathbf{M}[\mathrm{d}a],$$

with $\mathbf{M}[\cdot]$, the reference measure on $(\mathcal{A}, \Sigma_A)$.

Note. In this thesis, the reference measure, $\mathbf{M}[\cdot]$, equals the Lebesgue measure for continuous random variables or the counting measure for discrete random variables. The counting measure equals the set cardinality if it has a finite number of elements.

Note. The PMF for a discrete random variable, $A : \Omega \to \mathcal{A} \subset \mathbb{R}^{n \times m}$, has the following form:

$$f_A(a) = \mathbf{P}[A = a] = \mathbf{P}\Big[A^{-1}\big(\{a\}\big)\Big].$$

This can be seen by the fact that S is a discrete set and therefore $\mathbf{P}[A \in S] = \sum_{a \in S} \mathbf{P}[A = a]$.

Definition A.10. (Joint PDF/PMF) Consider the individual elements: $A_i : \Omega \to \mathcal{A}_i \subseteq \mathbb{R}^n$ for $i \in \mathbb{N}_m$, of an $n \times m$-dimensional random vector or matrix: $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^{n \times m}$, $A = [A_0, A_1, \ldots, A_{m-1}]$. A joint PDF/PMF is defined as $f_{A_0, A_1, \ldots, A_{m-1}} : \bigtimes_{i=0}^{m-1} \mathcal{A}_i \to \mathbb{R}_{\geq 0}$ by

$$f_{A_0, A_1, \ldots, A_{m-1}}(a_0, a_1, \ldots, a_{m-1}) = f_A\big([a_0, a_1, \ldots, a_{m-1}]\big).$$

A.2.4 Cumulative Distribution Function

The Cumulative Distribution Function (CDF) for a random variable gives the probability that a realization less than or equal to some value will be observed.

Definition A.11. (CDF) The CDF for a random variable (scalar, vector or matrix), $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^{n\times m}$, is a right-continuous monotone increasing function, $F_A : \mathbb{R}^{n\times m} \to [0, 1]$, satisfying the following limits:

$$\forall i \in \mathbb{N}_n, \forall j \in \mathbb{N}_m : \lim_{x_{i,j}\to-\infty} F_A(x) = 0,$$

$$\lim_{x\to\boldsymbol{\infty}_{\boldsymbol{n\times m}}} F_A(x) = 1,$$

with $x \to \boldsymbol{\infty}_{\boldsymbol{n\times m}}$, indicating that all elements of x approach positive infinity. The CDF evaluates to

$$\forall x \in \mathbb{R}^{n\times m} : F_A(x) = \mathbf{P}\big[\{\omega \in \Omega \mid \forall i \in \mathbb{N}_n, \forall j \in \mathbb{N}_m : A_{i,j}(\omega) \leq x_{i,j}\}\big].$$

As for the PDF, the joint CDF is defined by using the vector or matrix CDF.

Definition A.12. (Joint CDF) Consider the individual elements: $A_i : \Omega \to \mathcal{A}_i \subseteq \mathbb{R}^n$ for $i \in \mathbb{N}_m$, of an $n \times m$-dimensional random vector or matrix: $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^{n\times m}$, $A = [A_0, A_1, \ldots, A_{m-1}]$. A joint CDF is defined as $F_{A_0,A_1,\ldots,A_{m-1}} : \bigtimes_{i=0}^{m-1} \mathcal{A}_i \to [0, 1]$ by

$$F_{A_0,A_1,\ldots,A_{m-1}}(x_0, x_1, \ldots, x_{m-1}) = F_A\big([x_0, x_1, \ldots, x_{m-1}]\big).$$

Note. For a one-dimensional random variable, by the Radon-Nikodym theorem and using definition A.9, the PDF/PMF equals

$$f_A(a) = \frac{\mathrm{d}A_*(\mathbf{P})}{\mathrm{d}\mathbf{M}} = \frac{A_*(\mathbf{P})[\mathrm{d}x]}{\mathbf{M}[\mathrm{d}x]}.$$

When the random variable is continuous:

$$f_A(a) = \frac{A_*(\mathbf{P})[\mathrm{d}x]}{\mathbf{M}[\mathrm{d}x]}(a) = \frac{\mathrm{d}F_A}{\mathrm{d}x}(a).$$

When the random variable is discrete:

$$f_A(a) = \frac{A_*(\mathbf{P})[\mathrm{d}x]}{\mathbf{M}[\mathrm{d}x]}(a) = A_*(\mathbf{P})[\mathrm{d}x](a) = F_A(a) - \lim_{x\to a^-} F_A(x) = \mathbf{P}\big[\{a\}\big].$$

Note. Given a set of realizations: $S \subset \mathbb{R}$, in the form of an interval, $S = (x_0, x_1]$, the probability for a realization in this set can be described using the CDF: $\mathbf{P}[A \in S] = F_A(x_1) - F_A(x_0)$. For a discrete random variable, $A : \Omega \to \mathcal{A} \subset \mathbb{R}$, the CDF becomes $F_A(x) = \sum_{a\in S} \mathbf{P}[A = a]$, with S the set containing all possible realizations smaller than or equal to x: $S = \{a \in \mathcal{A} \mid a \leq x\}$.

A.2.5 Expectation

The expectation operator enables the description of the expected (average) outcome for a random variable. Additionally, it provides a means of expressing the expected spread and similarity between two random variables.

Expected Value

The expected value operator, $\mathbf{E}$, assigns a mean value to a random variable.

Definition A.13. (Expected value) The expected value is a linear function, $\mathbf{E} : \{\Omega \to \mathbb{R}^{n\times m}\} \to \mathbb{R}^{n\times m}$, working on random variables (scalars, vectors or matrices):

$$\mathbf{E}[A] = \int_{\Omega} A \mathrm{d}\mathbf{P} = \int_{\Omega} A(\omega)\mathbf{P}[\mathrm{d}\omega] = \int_{\mathcal{A}} a f_A(a)\mathbf{M}[\mathrm{d}a],$$

for any random variable: $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^{n\times m}$.

Note. The expected value, $\mathbf{E}[A]$, has the same shape as the original random variable, A.

Note. In case of a discrete random variable, the integral can be replaced by a weighted sum over all realizations:

$$\mathbf{E}[A] = \sum_{\omega\in\Omega} A(\omega)\mathbf{P}\big[\{\omega\}\big] = \sum_{a\in\mathcal{A}} a\mathbf{P}[A = a].$$

Note. The expected value of a matrix is the matrix containing the expected value of its elements:

$$\mathbf{E}[\mathbb{A}] = \begin{bmatrix} \mathbf{E}[\mathbb{A}_{0,0}] & \mathbf{E}[\mathbb{A}_{0,1}] & \dots & \mathbf{E}[\mathbb{A}_{0,m-1}] \\ \mathbf{E}[\mathbb{A}_{1,0}] & \mathbf{E}[\mathbb{A}_{1,1}] & \dots & \mathbf{E}[\mathbb{A}_{1,m-1}] \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{E}[\mathbb{A}_{n-1,0}] & \mathbf{E}[\mathbb{A}_{n-1,1}] & \dots & \mathbf{E}[\mathbb{A}_{n-1,m-1}] \end{bmatrix}.$$

Covariance and Correlation

The covariance between two random variables is a measure of similarity.

Definition A.14. (Covariance) The covariance is a function, $\mathbf{Cov} : \{\Omega \to \mathbb{R}^n\} \times \{\Omega \to \mathbb{R}^m\} \to \mathbb{R}^{n\times m}$, working on pairs of random variables (scalars or vectors):

$$\mathbf{Cov}[A, C] = \mathbf{E}\Big[\big(A - \mathbf{E}[A]\big)\big(C - \mathbf{E}[C]\big)^{\mathsf{T}}\Big],$$

for any two random variables (scalars or vectors): $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^n$, $C : \Omega \to \mathcal{C} \subseteq \mathbb{R}^m$.

The correlation is related to the covariance.

Definition A.15. (Correlation) The correlation is a function, $\mathbf{Cor} : \{\Omega \to \mathbb{R}^n\} \times \{\Omega \to \mathbb{R}^m\} \to \mathbb{R}^{n \times m}$, working on pairs of random variables (scalars or vectors):

$$\mathbf{Cor}[A, C] = \mathbf{E}\big[AC^\intercal\big] = \mathbf{Cov}[A, C] + \mathbf{E}[A]\mathbf{E}^\intercal[C],$$

for any two random variables (scalars or vectors): $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^n$, $C : \Omega \to \mathcal{C} \subseteq \mathbb{R}^m$.

Variance

The variance of a random variable is the covariance of the random variable with itself.

Definition A.16. (Variance) The variance is a function, $\mathbf{Var} : \{\Omega \to \mathbb{R}^n\} \to \mathbb{R}^{n \times n}$, working on random variables (scalars or vectors):

$$\mathbf{Var}[A] = \mathbf{E}\Big[\big(A - \mathbf{E}[A]\big)\big(A - \mathbf{E}[A]\big)^\intercal\Big] = \mathbf{Cov}[A, A],$$

for any random variable (scalar or vector): $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^n$.

A.3 Random Process

A random process describes a collection of random variables to the same measurable space, indexed by some index set.

Definition A.17. (Random process) An $\mathcal{A}$-valued random process, $\big\{A(t)\big\}_{t \in T}$ or $A(\omega, t)$, is a function from the sample space, Ω, and the index set, T, to a measurable space: $(\mathcal{A}, \Sigma_A)$, $\Omega \times T \to \mathcal{A}$.

Note. In this thesis, the index set of interest is a subset of the real line.

A realization of a random process can be regarded as a function from the index set to this measurable space.

Definition A.18. (Realization of a random process) A realization of an $\mathcal{A}$-valued random process, $\big\{A(t)\big\}_{t \in T}$, for a random outcome, ω, is a function: $a : T \to \mathcal{A}$ by $a(t) = A(\omega, t)$.

Evaluating the random process at a given index, yields a random variable.

Definition A.19. (Evaluation of a random process) An evaluation of an $\mathcal{A}$-valued random process, $\{A(t)\}_{t\in T}$, at an index, $t_i \in T$, yields an $\mathcal{A}$-valued random variable: $A(t_i) : \Omega \to \mathcal{A}$.

A.3.1 Expectation

A mean function is defined, assigning a mean value for the random process to each index in the index set.

Definition A.20. (Mean function) Given an $\mathcal{A}$-valued random process: $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^{n\times m}$, the mean function is defined as $\mu_A : T \to \mathbb{R}^{n\times m}$ by $\mu_A(t) = \mathbf{E}[A(t)]$.

For two real-valued random processes, the Cross-Correlation Function (CCF) describes the correlation between the evaluations of these processes.

Definition A.21. (CCF) Given two real-valued (scalar or vector) random processes: $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^n$ and $\{C(u)\}_{u\in U} : \Omega \times U \to \mathcal{C} \subseteq \mathbb{R}^m$, the CCF is defined as

$$R_{AC} : T \times U \to \mathbb{R}^{n\times m} \text{ by } R_{AC}(t,u) = \mathbf{Cor}[A(t), C(u)] = \mathbf{E}[A(t)C^{\intercal}(u)].$$

A cross-covariance function can also be defined.

Definition A.22. (Cross-covariance function) Given two real-valued (scalar or vector) random processes: $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^n$ and $\{C(u)\}_{u\in U} : \Omega \times U \to \mathcal{C} \subseteq \mathbb{R}^m$, the cross-covariance function is defined as

$$\begin{aligned} K_{AC} : T \times U \to \mathbb{R}^{n\times m} \text{ by } K_{AC}(t,u) &= \mathbf{Cov}[A(t), C(u)] \\ &= \mathbf{E}\Big[\big(A(t) - \mu_A(t)\big)\big(C(u) - \mu_C(u)\big)^{\intercal}\Big] \\ &= R_{AC}(t,u) - \mu_A(t)\mu_C^{\intercal}(u). \end{aligned}$$

The Auto-Correlation Function (ACF) equals the CCF of the process with itself.

Definition A.23. (ACF) Given a real-valued (scalar or vector) random process: $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^n$, the ACF is defined as

$$R_A : T^2 \to \mathbb{R}^{n\times n} \text{ by } R_A(t_i,t_j) = R_{AA}(t_i,t_j) = \mathbf{E}[A(t_i)A^{\intercal}(t_j)].$$

Similarly, an auto-covariance function can be defined as well.

Definition A.24. (Auto-covariance function) Given a real-valued (scalar or vector) random process: $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^n$, the auto-covariance function is defined as

$$\begin{aligned} K_A : T^2 \to \mathbb{R}^{n\times n} \text{ by } K_A(t_i,t_j) = K_{AA}(t_i,t_j) &= \mathbf{Cov}\big[A(t_i),A(t_j)\big] \\ &= \mathbf{E}\Big[\big(A(t_i)-\mu_A(t_i)\big)\big(A(t_j)-\mu_A(t_j)\big)^{\intercal}\Big] \\ &= R_A(t_i,t_j)-\mu_A(t_i)\mu_A^{\intercal}(t_j). \end{aligned}$$

A.3.2 Stationarity

A random process with a subset of the real coordinate space, $\mathbb{R}^n$, as its index set is strict-sense stationary when its properties do not change with an index shift.

Definition A.25. (Strict-sense stationary) Given a random process (scalar or vector): $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^m$, with as an index set a subset of the real coordinate space, $T \subseteq \mathbb{R}^n$. This random process is strict-sense stationary if

$$\begin{aligned} &\forall k \in \mathbb{N}, \forall t_0,t_1,\ldots,t_k \in T, \\ &\forall \tau \in \mathbb{R}^n \mid t_0+\tau, t_1+\tau,\ldots,t_k+\tau \in T, \forall x_0,x_1,\ldots,x_k \in \mathbb{R}^m : \\ &\quad F_{A(t_0),A(t_1),\ldots,A(t_k)}(x_0,x_1,\ldots,x_k) \\ &\qquad = F_{A(t_0+\tau),A(t_1+\tau),\ldots,A(t_k+\tau)}(x_0,x_1,\ldots,x_k). \end{aligned}$$

Less stringent requirements on the random process lead to the Wide-Sense Stationary (WSS) property.

Definition A.26. (WSS) Given a random process (scalar or vector): $\{A(t)\}_{t\in T} : \Omega \times T \to \mathcal{A} \subseteq \mathbb{R}^m$, with as an index set a subset of the real coordinate space, $T \subseteq \mathbb{R}^n$. This random process is WSS if

$$\begin{gathered} \forall t \in T, \forall \tau \in \mathbb{R}^n \mid t+\tau \in T : \mu_A(t) = \mu_A(t+\tau), \\ \forall t_i,t_j \in T, \forall \tau \in \mathbb{R}^n \mid t_i+\tau, t_j+\tau \in T : R_A(t_i,t_j) = R_A(t_i+\tau,t_j+\tau), \\ \forall t \in T : \mathbf{E}\big[|A(t)|^2\big] < \infty. \end{gathered}$$

Note. When a random process is WSS, the mean function is constant and the ACF only depends on the time shift: $R_A(t_i,t_j) = R_A(0,t_j-t_i)$. Whenever $t_j - t_i \notin T$, the domain for the ACF is extended and this relation defines the value of the ACF at that index. The notation: $R_A(t_j-t_i) = R_A(0,t_j-t_i)$, is used.

A.3.3 Spectrum

The Power Spectral Density (PSD) is a density function through the Fourier frequency, representing the power density carried by a specific spectral component of a random process. When integrated in a frequency band, the power dissipation by the process after a band-pass filter is obtained.

Definition A.27. (PSD) The auto-PSD (or just PSD), for a continuous random process (scalar), $\{A(t)\}_{t\in T\subseteq\mathbb{R}} : \Omega\times T \to \mathcal{A}\subseteq\mathbb{R}$ and corresponding windowed FT random process: $\{C_\Delta(f)\}_{f\in F\subseteq\mathbb{R}} : \Omega\times F\to\mathbb{C}$ by $C_\Delta(\omega,f)=\mathcal{F}\{w_\Delta(t)A(\omega,t)\}$, is a positive-valued function of the Fourier frequency: $S_A : F\to\mathbb{R}_{\geq 0}$, defined as

$$\forall f\in F : S_A(f)=\lim_{\Delta\to\infty}\frac{1}{\Delta}\mathbf{E}\Big[\big|C_\Delta(f)\big|^2\Big],$$

When $\{A(t)\}_{t\in T}$ is a WSS process and its PSD exists, then this process' ACF and PSD form an FT pair: $\forall f\in\mathbb{R} : S_A(f)=\mathcal{F}\{R_A(\tau)\}$. This relation is known as the Wiener-Khinchin theorem.

A.4 Conditionality

Conditional probability mathematics provides a method to revise knowledge of a random variable or event, incorporating additional information beyond previously available prior knowledge.

A.4.1 Conditional Expectation

The conditional expectation yields the expected value, given the knowledge that a specified event occurred with certainty.

Conditioned on an Event

The conditional expectation of a random variable, given an event with non-zero probability generates a deterministic element with identical dimension as the original random variable.

Definition A.28. (Expectation conditioned on an event) Given a random variable (scalar or vector): $A : \Omega\to\mathcal{A}\subseteq\mathbb{R}^n$ and an event with non-zero

probability: $E \in \mathcal{F}$, $\mathbf{P}[E] \neq 0$, the conditional expectation of A, given E is a deterministic element (scalar or vector), $\mathbf{E}[A \mid E] \in \mathbb{R}^n$, defined as

$$\mathbf{E}[A \mid E] = \frac{\int_E A \mathrm{d}\mathbf{P}}{\mathbf{P}[E]}.$$

Note. For a random variable, $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^n$, and another discrete random variable, $C : \Omega \to \mathcal{C} \subset \mathbb{R}^m$, the conditional expectation, given the event: $C^{-1}(c)$ (C has a realization equal to $c \in \mathcal{C}$, with non-zero probability) equals

$$\mathbf{E}[A \mid C^{-1}(c)] = \frac{\int_{\omega \in C^{-1}(c)} A(\omega) \mathbf{P}[\mathrm{d}\omega]}{\mathbf{P}[C = c]} = \frac{\int_{a \in S_c} a f_{A,C}(a, c) \mathrm{d}a}{f_C(c)}, \tag{A.3}$$

with $S_c = A\big(C^{-1}(c)\big) = \big\{A(\omega) \mid \omega \in C^{-1}(c)\big\}$ and $f_{A,C} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}_{\geq 0}$, the joint mixed distribution function for A and C. A shorthand notation: $\mathbf{E}[A \mid C = c] = \mathbf{E}[A \mid C^{-1}(c)]$, is used.

Note. The conditional expectation, $\mathbf{E}[A \mid C = \cdot]$, can be regarded as a function: $\mu_{A|C} : \mathcal{C} \to \mathbb{R}^n$ by $\mu_{A|C}(c) = \mathbf{E}[A \mid C = c]$.

Conditioned on a Realization

The result from eq. (A.3) cannot be used when conditioning on the realization of a continuous random variable, C, as the event: $C^{-1}(c)$, has zero probability: $\mathbf{P}[C = c] = 0$. Instead, the conditional expectation, given a realization of another random variable is defined using the PDF/PMF.

Definition A.29. (Expectation conditioned on a realization) Given two random variables (scalar or vector): $A : \Omega \to \mathcal{A} \subseteq \mathbb{R}^n$ and $C : \Omega \to \mathcal{C} \subseteq \mathbb{R}^m$, with PDF/PMF: f_C and joint PDF/PMF: $f_{A,C}$. The conditional expectation of A, given a realization: $c \in \mathcal{C}$, is a deterministic element (scalar or vector), $\mathbf{E}[A \mid C = c] \in \mathbb{R}^n$, defined as

$$\mathbf{E}[A \mid C = c] = \frac{\int_{a \in \mathcal{A}} a f_{A,C}(a, c) \mathrm{d}a}{f_C(c)},$$

when $f_C(c) \in \mathbb{R}_{>0}$.

Note. When C is a discrete random variable, definition A.29 becomes equal to eq. (A.3), as $f_{A,C}(a, c) = 0$ for $a \notin S_c$.

A.4.2 Conditional Probability

The conditional probability can be expressed in terms of the conditional expectation, by making use of the indicator function for an event, $1_E : \Omega \to$

$\{0, 1\}$. The indicator function acts as a discrete real-valued random variable, producing a one if the event: E occurred and zero otherwise. The following relations hold: $\mathbf{P}[E] = \mathbf{P}[1_E = 1] = f_{1_E}(1) = \mathbf{E}[1_E]$.

Conditioned on an Event

The conditional probability of an event, given another event with non-zero probability occurred, generates a new probability.

Definition A.30. (Probability conditioned on an event) Given two events: $E \in \mathcal{F}$ and $F \in \mathcal{F}$, with $\mathbf{P}[F] \neq 0$, the conditional probability of E, given the occurrence of the event F, is a deterministic probability, $\mathbf{P}[E \mid F] \in [0, 1]$, defined as

$$\mathbf{P}[E \mid F] = \mathbf{E}[1_E \mid F] = \frac{\int_F 1_E \mathrm{d}\mathbf{P}}{\mathbf{P}[F]} = \frac{\mathbf{P}[E \cap F]}{\mathbf{P}[F]}.$$

Note. For an event: $E \in \mathcal{F}$ and a discrete random variable: $C : \Omega \to \mathcal{C} \subset \mathbb{R}^m$, the conditional probability of E, given the event: $C^{-1}(c)$ (C has a realization equal to $c \in \mathcal{C}$, with non-zero probability) equals

$$\mathbf{P}[E \mid C^{-1}(c)] = \frac{\int_{\omega \in C^{-1}(c)} 1_E(\omega) \mathbf{P}[\mathrm{d}\omega]}{f_C(c)} = \frac{\mathbf{P}\big[E \cap C^{-1}(c)\big]}{f_C(c)}, \tag{A.4}$$

with f_C the PMF for C.

Note. When the event, E, represents a realization of another discrete random variable: $E = A^{-1}(a)$, with $A : \Omega \to \mathcal{A} \subset \mathbb{R}^n$, the conditional probability becomes

$$\mathbf{P}[A^{-1}(a) \mid C^{-1}(c)] = \frac{\mathbf{P}\big[A^{-1}(a) \cap C^{-1}(c)\big]}{\mathbf{P}[C = c]} = \frac{f_{A,C}(a, c)}{f_C(c)}, \tag{A.5}$$

with $f_{A,C}$ the joint PMF for A and C. The notation: $\mathbf{P}[A = a \mid C = c]$ is used to denote this probability.

Conditioned on a Realization

Again, eq. (A.4) cannot be used when conditioning on the realization of a continuous random variable: C. Instead, the conditional probability, given a realization of a random variable is defined using the mixed joint probability distribution function.

Definition A.31. (Probability conditioned on a realization) Given an event: $E \in \mathcal{F}$ and a random variable: $C : \Omega \to \mathcal{C} \subseteq \mathbb{R}^m$, with PDF/PMF: f_C and mixed

joint probability distribution function: $f_{1_E,C}$. The conditional probability of E, given a realization: $c \in \mathcal{C}$, is a deterministic probability: $\mathbf{P}[E \mid C = c] \in [0, 1]$, defined as

$$\mathbf{P}[E \mid C = c] = \mathbf{E}[1_E \mid C = c] = \frac{f_{1_E,C}(1, c)}{f_C(c)} = \frac{f_{1_E,C}(1, c)}{f_{1_E,C}(0, c) + f_{1_E,C}(1, c)},$$

when $f_C(c) \in \mathbb{R}_{>0}$.

Note. When the event, E, represents the realization of a discrete random variable: $E = A^{-1}(a)$, with $A : \Omega \to \mathcal{A} \subset \mathbb{R}^n$ and $a \in \mathcal{A}$, definition A.31 becomes similar to eq. (A.5).

A.4.3 Conditional Distribution

The conditional distribution function is a scaled version of the joint distribution function.

Conditioned on an Event

The PDF/PMF for a random variable, given the occurrence of a non-zero probability event changes into a conditional PDF/PMF.

Definition A.32. (Distribution conditioned on an event) Given an $\mathcal{A}$-valued random variable: $A : \Omega \to \mathcal{A}$ and an event: $E \in \mathcal{F}$ with non-zero probability, $\mathbf{P}[E] \neq 0$. The conditional PDF/PMF for A, given the occurrence of E is $f_{A|E} : \mathcal{A} \to \mathbb{R}_{\geq 0}$, defined as

$$f_{A|E}(a) = \frac{f_{A,1_E}(a, 1)}{\mathbf{P}[E]},$$

with $f_{A,1_E}$, the joint distribution for A and 1_E, the indicator function for E.

Conditioned on a Realization

The PDF/PMF for a random variable, given the realization of a different random variable is defined using the joint PDF/PMF.

Definition A.33. (Distribution conditioned on a realization) Given two random variables: $A : \Omega \to \mathcal{A}$ and $C : \Omega \to \mathcal{C}$, with joint PDF/PMF: $f_{A,C}$ and marginal

PDF/PMF: f_C for C. The conditional PDF/PMF for A, given a realization of C: $c \in \mathcal{C}$, is $f_{A|C}(\cdot \mid c) : \mathcal{A} \to \mathbb{R}_{\geq 0}$, defined as

$$f_{A|C}(a \mid c) = \frac{f_{A,C}(a,c)}{f_C(c)},$$

when $f_C(c) \in \mathbb{R}_{>0}$.

Note. Definition A.32 equals definition A.33 for an event: $E = C^{-1}(c)$ and C a discrete random variable.

Note. A shorthand notation regarding conditional distribution functions is used. The notation: $A \mid C = c \sim \mathcal{X}(c)$ actually denotes the following relation: $\forall a \in \mathcal{D} : f_{A|C}(a \mid c) = f_D(a)$, with $D : \Omega \to \mathcal{D} \subseteq \mathcal{A}$, a dummy random variable following a parameterized distribution with parameter c: $D \sim \mathcal{X}(c)$. Both $A : \Omega \to \mathcal{A}$ and $C : \Omega \to \mathcal{C}$ are random variables, with $c \in \mathcal{C}$, a possible realization for C.

A.5 Entropy

This section defines the flowing concepts: information content, entropy and conditional entropy used in this thesis. All definitions in this section are only valid for discrete random variables.

A.5.1 Information Content

The information content of an event provides a way to quantify the amount of information gained upon observing the event.

Definition A.34. (Information content) The information content (surprisal) is a function from the set of events to the non-negative real line, $\mathbf{I} : \mathcal{F} \to \mathbb{R}_{\geq 0}$, which equals

$$\mathbf{I}[E] = -k \ln\big(\mathbf{P}[E]\big),$$

for any $E \in \mathcal{F}$ and with the factor: k, determining the units.

Note. For binary units (bit), use $k = \frac{1}{\ln(2)}$.

Note. Given a discrete random variable: $A : \Omega \to \mathcal{A}$, the event: $A^{-1}(a)$, obtained by observing the realization $a \in \mathcal{A}$, has the following information content: $\mathbf{I}\big[A^{-1}(a)\big]$.

Definition A.35. (Information content by a random variable) Given a discrete random variable: $A : \Omega \to \mathcal{A}$, the information content function by A: $I_A : \Omega \to \mathbb{R}_{\geq 0}$, is a random variable, defined as

$$I_A(\omega) = \mathbf{I}\Big[A^{-1}\Big(\big\{A(\omega)\big\}\Big)\Big] = \mathbf{I}\Big[\big\{\omega' \in \Omega \mid A(\omega') = A(\omega)\big\}\Big].$$

Similarly, a joint information content is defined.

Definition A.36. (Joint information content by random variables) Given two discrete random variables: $A : \Omega \to \mathcal{A}$ and $C : \Omega \to \mathcal{C}$, the joint information content function by A and C, $I_{A,C} : \Omega \to \mathbb{R}_{\geq 0}$, is a random variable, defined as

$$I_{A,C}(\omega) = \mathbf{I}\Big[\big\{\omega' \in \Omega \mid A(\omega') = A(\omega), C(\omega') = C(\omega)\big\}\Big].$$

A.5.2 Entropy for a Random Variable

From the information content, an entropy function is derived.

Shannon Entropy

The Shannon (average) entropy for a discrete random variable, introduced in [78], quantifies the expected information gained when observing a realization of this random variable.

Definition A.37. (Shannon entropy) Given a discrete random variable: $A : \Omega \to \mathcal{A}$, with PMF: f_A, the Shannon entropy, $\mathbf{H} : \{\Omega \to \mathcal{A}\} \to \mathbb{R}_{\geq 0}$, is defined as

$$\mathbf{H}[A] = \mathbf{E}[I_A] = \int_{\Omega} I_A(\omega)\mathbf{P}[\mathrm{d}\omega] = -k \sum_{a \in \mathcal{A}} f_A(a) \ln\big(f_A(a)\big).$$

Min-Entropy

The min-entropy for a random variable equals the smallest amount of information gained when observing a realization of this random variable.

Definition A.38. (Min-entropy) Given a discrete random variable: $A : \Omega \to \mathcal{A}$, with PMF: $f_A(a)$, the min-entropy, $\mathbf{H_m} : \{\Omega \to \mathcal{A}\} \to \mathbb{R}_{\geq 0}$, is defined as

$$\mathbf{H_m}[A] = \min_{\omega \in \Omega} I_A(\omega) = -k \ln\big(\max_{a \in \mathcal{A}} f_A(a)\big).$$

Note. Any outcome: $a_m \in \operatorname{argmax}_{a\in\mathcal{A}} \mathbf{P}[A = a]$ would be used when guessing a future realization of A. The probability: $\mathbf{P}[A = a_m] = \mathrm{e}^{-\frac{\mathbf{H_m}}{k}}$ equals the success rate of the optimal strategy for guessing A.

Joint Entropy

A joint entropy is similarly defined.

Definition A.39. (Joint entropy) Given two discrete random variables: $A : \Omega \to \mathcal{A}$ and $C : \Omega \to \mathcal{C}$, with joint PMF: $f_{A,C}$. The joint Shannon entropy and joint min-entropy for A and C: $\mathbf{H}, \mathbf{H_m} : \{\Omega \to \mathcal{A}\} \times \{\Omega \to \mathcal{C}\} \to \mathbb{R}_{\geq 0}$, are defined as

$$\mathbf{H}[A, C] = \mathbf{E}[I_{A,C}] = \int_\Omega I_{A,C}(\omega)\mathbf{P}[\mathrm{d}\omega] = -k \sum_{a\in\mathcal{A}, c\in\mathcal{C}} f_{A,C}(a, c) \ln\big(f_{A,C}(a, c)\big),$$

$$\mathbf{H_m}[A, C] = \min_{\omega\in\Omega} I_{A,C}(\omega) = -k \ln\big(\max_{a\in\mathcal{A}, c\in\mathcal{C}} f_{A,C}(a, c)\big).$$

A.5.3 Conditional Entropy

The conditional entropy allows to express the expected or worst case information content, given additional knowledge.

Conditioned on an Event

The Shannon entropy for a random variable, given the occurrence of an event equals the average information content of a realization, given an occurrence of the event, minus the information content of the event itself. The min-entropy is accordingly defined.

Definition A.40. (Entropy conditioned on an event) Given a discrete random variable: $A : \Omega \to \mathcal{A}$ and a non-zero probability event: $E \in \mathcal{F}$, $\mathbf{P}[E] \neq 0$. The Shannon entropy and min-entropy for A, given E are equal to

$$\begin{aligned}\mathbf{H}[A \mid E] &= -k \sum_{a\in\mathcal{A}} f_{A|E}(a) \ln\big(f_{A|E}(a)\big) \\ &= \sum_{a\in\mathcal{A}} f_{A|E}(a)\mathbf{I}[A^{-1}(a) \cap E] - \mathbf{I}[E],\end{aligned}$$

$$\mathbf{H_m}[A \mid E] = -k \ln\big(\max_{a\in\mathcal{A}} f_{A|E}(a)\big).$$

Note. When A is independent of E, then $f_{A|E}(a) = f_A(a)$ and therefore $\mathbf{H}[A \mid E] = \mathbf{H}[A]$.

Conditioned on a Realization

Considering the event when a certain realization for a random variable is observed, the Shannon- and min-entropy for a random variable conditioned on the realization of a different random variable are defined.

Definition A.41. (Entropy conditioned on a realization) Given two discrete random variables: $A : \Omega \to \mathcal{A}$ and $C : \Omega \to \mathcal{C}$, with PMF: f_C and joint PMF: $f_{A,C}$. The Shannon entropy and min-entropy for A, given a realization of C: $c \in \mathcal{C}$, with non-zero probability, $\mathbf{P}[C^{-1}(c)] \neq 0$, are equal to

$$\mathbf{H}[A \mid C = c] = \mathbf{H}[A \mid C^{-1}(c)] = -k \sum_{a \in \mathcal{A}} f_{A|C}(a \mid c) \ln\big(f_{A|C}(a \mid c)\big)$$

$$= \sum_{a \in \mathcal{A}} f_{A|C}(a \mid c) \mathbf{I}[A^{-1}(a) \cap C^{-1}(c)] - \mathbf{I}[C^{-1}(c)],$$

$$\mathbf{H}_{\mathbf{m}}[A \mid C = c] = -k \ln\big(\max_{a \in \mathcal{A}} f_{A|C}(a, c)\big).$$

Conditioned on a Random Variable

The Shannon entropy for a random variable, conditioned on a different random variable equals the average Shannon entropy, given a realization of the second random variable. The min-entropy is similarly defined.

Definition A.42. (Entropy conditioned on a random variable) Given two random variables: $A : \Omega \to \mathcal{A}$ and $C : \Omega \to \mathcal{C}$, with PMF: f_C and joint PMF: $f_{A,C}$. The Shannon entropy and min-entropy for A, given C are equal to

$$\mathbf{H}[A \mid C] = \sum_{c \in \mathcal{C}} f_C(c) \mathbf{H}[A \mid C = c] = \mathbf{H}[A, C] - \mathbf{H}[C],$$

$$\mathbf{H}_{\mathbf{m}}[A \mid C] = \sum_{c \in \mathcal{C}} f_C(c) \mathbf{H}_{\mathbf{m}}[A \mid C = c].$$

A.6 Key Probability Distributions

This section lists several probability distributions encountered in this thesis.

A.6.1 Multivariate Normal Distribution

Definition A.43. (Multivariate normal distribution) An $n \times 1$ random vector, $\vec{\boldsymbol{A}}$, that follows an n-dimensional multivariate normal distribution is noted as: $\vec{\boldsymbol{A}} \sim \mathcal{N}_n\big(\vec{\boldsymbol{\mu}}, \boldsymbol{\Sigma}\big)$, with mean vector: $\vec{\boldsymbol{\mu}}$ and covariance matrix: $\boldsymbol{\Sigma}$.

The PDF for an n-dimensional multivariate normal distribution, noted as $\phi_{\mathcal{N}_n}(\vec{\boldsymbol{a}}; \vec{\boldsymbol{\mu}}, \boldsymbol{\Sigma})$, is equal to

$$f_{\vec{\boldsymbol{A}}}(\vec{\boldsymbol{a}}) = \phi_{\mathcal{N}_n}(\vec{\boldsymbol{a}}; \vec{\boldsymbol{\mu}}, \boldsymbol{\Sigma}) = \frac{\exp\Big(-\frac{1}{2}(\vec{\boldsymbol{a}} - \vec{\boldsymbol{\mu}})^{\intercal}\boldsymbol{\Sigma}^{-1}(\vec{\boldsymbol{a}} - \vec{\boldsymbol{\mu}})\Big)}{\sqrt{(2\pi)^n|\boldsymbol{\Sigma}|}},$$

with $|\cdot|$, the matrix determinant.

Note. The CDF for a multivariate normal distribution is denoted as $\Phi_{\mathcal{N}_n}(\vec{\boldsymbol{a}}; \vec{\boldsymbol{\mu}}, \boldsymbol{\Sigma})$.

Note. When the dimension, n, exceeds one, there is no closed form available for the CDF. One must resort to numerical integration methods to determine the CDF value.

Note. When the dimension n equals one, the dimension subscript is omitted: $\mathcal{N}$ is used instead of $\mathcal{N}_1$.

Note. The PDF and CDF of a standard normal distributed variable (one-dimensional, normally distributed with mean $\vec{\boldsymbol{\mu}} = 0$ and variance $\boldsymbol{\Sigma} = 1$) are denoted as ϕ_s and Φ_s, respectively.

A.6.2 Gaussian Process

A Gaussian process is a random process that, when sampled at any finite collection of time instances, generates a multivariate normal distribution.

Definition A.44. (Gaussian process) A random process: $\big\{A(t)\big\}_{t\in T}$, is a Gaussian process if and only if (iff)

$$\forall n \in \mathbb{N}_{\neq 0}, \forall t_0, t_1, \ldots, t_{n-1} \in T : \vec{\boldsymbol{A}} = \big[A(t_0), A(t_1), \ldots, A(t_{n-1})\big]^{\intercal} \sim \mathcal{N}_n.$$

Note. The Gaussian process, $\big\{A(t)\big\}_{t\in T}$, is completely described by a mean function: $\mu_A(t) = \mathbf{E}\big[A(t)\big]$ and an ACF: $R_A(t_i, t_j) = \mathbf{E}\big[A(t_i)A(t_j)\big]$.

Note. As only metrics of at most second degree are required to fully describe the Gaussian process, it is stationary in the strict sense iff it is WSS.

Note. A Wiener process with drift μ, and infinitesimal variance σ^2, is a Gaussian process with mean function: $\forall t \in \mathbb{R}_{\geq 0} : \mu_W(t) = \mu t$, and ACF: $\forall t_i, t_j \in \mathbb{R}_{\geq 0} \mid t_i \leq t_j : R_W(t_i, t_j) = \mu_W(t_i)\mu_W(t_j) + \sigma^2 t_i$.

A.6.3 Inverse-Gaussian Distribution

The inverse-Gaussian distribution is associated with the distribution of the time it takes for a Brownian motion process to reach a fixed threshold (hitting time).

Definition A.45. (Inverse-Gaussian distribution) A random variable, A, that follows an inverse-Gaussian distribution is noted as: $A \sim \mathcal{IG}(\mu, \lambda)$, with mean: $\mu \in \mathbb{R}_{>0}$ and shape: $\lambda \in \mathbb{R}_{>0}$.

The PDF for an inverse-Gaussian distribution, noted as $\phi_{\mathcal{IG}}(a; \mu, \lambda)$, is equal to

$$\forall a \in \mathbb{R}_{>0} : f_A(a) = \phi_{\mathcal{IG}}(a; \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi a^3}} \exp\Big(-\frac{\lambda(a-\mu)^2}{2\mu^2 a}\Big).$$

A.6.4 Degenerate Distribution

A random variable following a degenerate distribution centered around a constant, has a realization equal to that constant with a probability of one.

Definition A.46. (Degenerate distribution) A random variable, A, that follows a degenerate distribution is noted as: $A \sim \mu$, with mean: μ.

The PDF for a degenerate distribution is equal to a shifted Dirac delta distribution: $\forall a \in \mathbb{R} : f_A(a) = \delta(a - \mu)$. We have $\mathbf{P}[A = \mu] = 1$ and $\forall a \in \mathbb{R}_{\neq \mu} : \mathbf{P}[A = a] = 0$.

A.6.5 Uniform Distribution

A random variable following a uniform distribution on the interval $[a_i, a_j]$, with $a_i < a_j$, has an equal probability for each realization in that interval.

Definition A.47. (Uniform distribution) A random variable, A, that follows a uniform distribution on the interval $[a_i, a_j]$, with $a_i < a_j$, is noted as: $A \sim \mathcal{U}(a_i, a_j)$.

The PDF for a uniform distribution is equal to a constant: $\forall a \in [a_i, a_j] : f_A(a) = \frac{1}{a_j - a_i}$.

Note. For a continuous random variable, the boundaries of the interval can be open or closed without affecting the PDF. This is because the endpoints have zero probability: $\mathbf{P}[A = a_i] = \mathbf{P}[A = a_j] = 0$.

Bibliography

[1] A.A. Abidi. "Phase Noise and Jitter in CMOS Ring Oscillators". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 41, no. 8, pp. 1803–1816, 2006.

[2] Muayad J. Aljafar, Zain Ul Abideen, Adriaan Peetermans, et al. "SCALLER: Standard Cell Assembled and Local Layout Effect-Based Ring Oscillators". In: *IEEE Embedded Systems Letters*, p. 4, 2024.

[3] D.W. Allan. "Statistics of Atomic Frequency Standards". In: *Proceedings of the IEEE*, vol. 54, no. 2, pp. 221–230, 1966.

[4] Takehiko Amaki, Masanori Hashimoto, Yukio Mitsuyama, et al. "A Design Procedure for Oscillator-Based Hardware Random Number Generator with Stochastic Behavior Modeling". In: *Information Security Applications*, Aug. 2010.

[5] Josep Balasch, Florent Bernard, Viktor Fischer, et al. "Design and Testing Methodologies for True Random Number Generators towards Industry Certification". In: *IEEE European Test Symposium (ETS)*, May 2018.

[6] Mathieu Baudet, David Lubicz, Julien Micolod, et al. "On the Security of Oscillator-Based Random Number Generators". In: *Journal of Cryptology*, vol. 24, no. 2, pp. 398–425, 2011.

[7] L. Benea, M. Carmona, F. Pebay-Peyroula, et al. "On the Characterization of Jitter in Ring Oscillators Using Allan Variance for True Random Number Generator Applications". In: *Euromicro Conference on Digital System Design (DSD)*, Aug. 2022.

[8] Licinius Benea, Mikael Carmona, Viktor Fischer, et al. "Impact of the Flicker Noise on the Ring Oscillator-based TRNGs". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2024, no. 2, pp. 870–889, 2024.

[9] Florent Bernard, Viktor Fischer, and Boyan Valtchanov. "Mathematical Model of Physical RNGs Based on Coherent Sampling". In: *Tatra Mountains Mathematical Publications*, vol. 45, no. 1, pp. 1–14, 2010.

[10] Nathalie Bochard, Florent Bernard, and Viktor Fischer. "Observing the Randomness in RO-Based TRNG". In: *International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, Dec. 2009.

[11] Yang Cao, Vladimir Rožić, Bohan Yang, et al. "Exploring Active Manipulation Attacks on the TERO Random Number Generator". In: *IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*, Oct. 2016.

[12] Abdelkarim Cherkaoui, Viktor Fischer, Alain Aubert, et al. "A Self-Timed Ring Based True Random Number Generator". In: *IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, May 2013.

[13] Abdelkarim Cherkaoui, Viktor Fischer, Laurent Fesquet, et al. "A Very High Speed True Random Number Generator with Entropy Assessment". In: *Cryptographic Hardware and Embedded Systems (CHES)*, Aug. 2013.

[14] Andrew Chi, Brandon Enright, and David McGrew. "Detecting Weak Keys in Manufacturing Certificates: A Case Study". In: *Annual Computer Security Applications Conference (ACSAC)*, Dec. 2023.

[15] Mathieu Coustans, Abdelkarim Cherkaoui, Laurent Fesquet, et al. "Subthreshold Logic for Low-Area and Energy Efficient True Random Number Generator". In: *IEEE Symposium in Low-Power and High-Speed Chips (COOLCHIPS)*, Apr. 2018.

[16] Nicola Da Dalt, and Ali Sheikholeslami. *Understanding Jitter and Phase Noise: A Circuits and Systems Perspective*. Cambridge University Press, 2018.

[17] J.-L. Danger, S. Guilley, and P. Hoogvorst. "High Speed True Random Number Generator Based on Open Loop Structures in FPGAs". In: *Microelectronics Journal*, vol. 40, no. 11, pp. 1650–1656, 2009.

[18] Markus Dichtl. "A Closer Look at a Recent Pipelined True Random Number Generator Design". In: *IACR Cryptology ePrint Archive*, vol. 2022, no. 1597, p. 7, 2022.

[19] Markus Dichtl. "How to Predict the Output of a Hardware Random Number Generator". In: *Cryptographic Hardware and Embedded Systems (CHES)*, Sept. 2003.

[20] Donhee Ham, and A. Hajimiri. "Virtual Damping and Einstein Relation in Oscillators". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 38, no. 3, pp. 407–418, 2003.

[21] Viktor Fischer, and David Lubicz. "Embedded Evaluation of Randomness in Oscillator Based Elementary TRNG". In: *Cryptographic Hardware and Embedded Systems (CHES)*, Sept. 2014.

[22] G. Ghibaudo, O. Roux, Ch. Nguyen-Duc, et al. "Improved Analysis of Low Frequency Noise in Field-Effect MOS Transistors". In: *Physica Status Solidi (a)*, vol. 124, no. 2, pp. 571–581, 1991.

[23] Oded Goldreich, and Dana Ron. "On Testing Expansion in Bounded-Degree Graphs". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation.* Springer Berlin Heidelberg, 2011.

[24] Miloš Grujić, and Ingrid Verbauwhede. "TROT: A Three-Edge Ring Oscillator Based True Random Number Generator With Time-to-Digital Conversion". In: *IEEE Transactions on Circuits and Systems I (TCAS I)*, vol. 69, no. 6, pp. 2435–2448, 2022.

[25] Z. Gutterman, B. Pinkas, and T. Reinman. "Analysis of the Linux Random Number Generator". In: *IEEE Symposium on Security and Privacy (S&P)*, May 2006.

[26] Patrick Haddad, Viktor Fischer, Florent Bernard, et al. "A Physical Approach for Stochastic Modeling of TERO-Based TRNG". In: *Cryptographic Hardware and Embedded Systems (CHES)*, Sept. 2015.

[27] Patrick Haddad, Yannick Teglia, Florent Bernard, et al. "On the Assumption of Mutual Independence of Jitter Realizations in P-TRNG Stochastic Models". In: *Design Automation and Test in Europe Conference & Exhibition (DATE)*, Mar. 2014.

[28] A. Hajimiri, and T.H. Lee. "A General Theory of Phase Noise in Electrical Oscillators". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 33, no. 2, pp. 179–194, Feb./1998.

[29] A. Hajimiri, S. Limotyrakis, and T.H. Lee. "Jitter and Phase Noise in Ring Oscillators". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 34, no. 6, pp. 790–804, 1999.

[30] Nadia Heninger, Zakir Durumeric, Eric Wustrow, et al. "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices". In: *USENIX Security Symposium*, Aug. 2012.

[31] F. Herzel, and B. Razavi. "A Study of Oscillator Jitter Due to Supply and Substrate Noise". In: *IEEE Transactions on Circuits and Systems II (TCAS II)*, vol. 46, no. 1, pp. 56–62, Jan./1999.

[32] D.A. Howe, D.U. Allan, and J.A. Barnes. "Properties of Signal Sources and Measurement Methods". In: *Annual Frequency Control Symposium*, May 1981.

[33] S. M. Ishraqul Huq, Oli Lowna Baroi, Sk. Aqiluzzaman Shihab, et al. "Comparative Study and Design of Current Starved Ring Oscillators in 16 Nm Technology". In: *IEEE Transactions on Circuits and Systems II (TCAS II)*, vol. 68, no. 4, pp. 1098–1102, 2021.

[34] ISO/IEC JTC 1/SC 27. *Test and Analysis Methods for Random Bit Generators within ISO/IEC 19790 and ISO/IEC 15408*. International Organization for Standardization (ISO), 2019.

[35] C. Jakobson, I. Bloom, and Y. Nemirovsky. "1/f Noise in CMOS Transistors for Analog Applications from Subthreshold to Saturation". In: *Solid-State Electronics*, vol. 42, no. 10, pp. 1807–1817, 1998.

[36] Olav Kallenberg. *Foundations of Modern Probability*. Springer International Publishing, 2021.

[37] Auguste Kerckhoffs. "La Cryptographie Militaire". In: *Journal des Sciences Militaires*, vol. 9, pp. 161–191, 1883.

[38] Wolfgang Killmann, and Werner Schindler. *A Proposal for: Functionality Classes and Evaluation Methodology for True (Physical) Random Number Generators*. Bundesamt für Sicherheit in der Informationstechnik (BSI), 2001.

[39] Wolfgang Killmann, and Werner Schindler. *A Proposal for: Functionality Classes for Random Number Generators*. Bundesamt für Sicherheit in der Informationstechnik (BSI), 2011.

[40] Eunhwan Kim, Minah Lee, and Jae-Joon Kim. "8.2 8Mb/s 28Mb/mJ Robust True-Random-Number Generator in 65nm CMOS Based on Differential Ring Oscillator with Feedback Resistors". In: *IEEE International Solid- State Circuits Conference (ISSCC)*, Feb. 2017.

[41] M.J. Kirton, and M.J. Uren. "Noise in Solid-State Microstructures: A New Perspective on Individual Defects, Interface States and Low-Frequency ($1/f$) Noise". In: *Advances in Physics*, vol. 38, no. 4, pp. 367–468, 1989.

[42] Netanel Klein, Eyal Harel, and Itamar Levi. "The Cost of a True Random Bit—On the Electronic Cost Gain of ASIC Time-Domain-Based TRNGs". In: *Cryptography*, vol. 5, no. 3, p. 25, 2021.

[43] Paul Kohlbrenner, and Kris Gaj. "An Embedded True Random Number Generator for FPGAs". In: *ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA)*, Feb. 2004.

[44] Yrjo Koyen, Adriaan Peetermans, Vladimir Rožić, et al. "Attacking Hardware Random Number Generators in a Multi-Tenant Scenario". In: *Workshop on Fault Detection and Tolerance in Cryptography (FDTC)*, Sept. 2020.

[45] L. H. A. Leunissen, W. Zhang, W. Wu, et al. "Impact of Line Edge Roughness on Copper Interconnects". In: *Journal of Vacuum Science & Technology B*, vol. 24, no. 4, pp. 1859–1862, 2006.

[46] David Lubicz, and Nathalie Bochard. "Towards an Oscillator Based TRNG with a Certified Entropy Rate". In: *IEEE Transactions on Computers*, vol. 64, no. 4, pp. 1191–1200, 2015.

[47] David Lubicz, and Viktor Fischer. "Entropy Computation for Oscillator-based Physical Random Number Generators". In: *Journal of Cryptology*, vol. 37, no. 2, p. 13, 2024.

[48] Willy M. C. Sansen. *Analog Design Essentials*. Springer US, 2006.

[49] Yuan Ma, Jingqiang Lin, Tianyu Chen, et al. "Entropy Evaluation for Oscillator-Based True Random Number Generators". In: *Cryptographic Hardware and Embedded Systems (CHES)*, Sept. 2014.

[50] Abhranil Maiti, and Patrick Schaumont. "Improving the Quality of a Physical Unclonable Function Using Configurable Ring Oscillators". In: *International Conference on Field Programmable Logic and Applications (FPL)*, Aug. 2009.

[51] Mehrdad Majzoobi, Farinaz Koushanfar, and Srinivas Devadas. "FPGA-Based True Random Number Generation Using Circuit Metastability with Adaptive Feedback Control". In: *Cryptographic Hardware and Embedded Systems (CHES)*, Sept. 2011.

[52] B. Mandelbrot. "Some Noises with 1/f Spectrum, a Bridge between Direct Current and White Noise". In: *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 289–298, 1967.

[53] George Marsaglia. *Diehard Battery of Tests of Randomness*. Florida State University, 1995. URL: https://web.archive.org/web/20160125103112/http://stat.fsu.edu/pub/diehard/ (visited on 05/06/2024).

[54] Sanu K. Mathew, Suresh Srinivasan, Mark A. Anders, et al. "2.4 Gbps, 7 mW All-Digital PVT-Variation Tolerant True Random Number Generator for 45 Nm CMOS High-Performance Microprocessors". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 47, no. 11, pp. 2807–2821, 2012.

[55] J.A. McNeill. "Jitter in Ring Oscillators". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 32, no. 6, pp. 870–879, 1997.

[56] Arturo Mollinedo Garay, Florent Bernard, Viktor Fischer, et al. “An Evaluation Procedure for Comparing Clock Jitter Measurement Methods”. In: *Smart Card Research and Advanced Applications (CARDIS)*, Nov. 2022.

[57] Anindo Mukherjee, and Kevin Skadron. *Measuring Parameter Variation on an FPGA Using Ring Oscillators*. University of Virginia, Department of Computer Science, 2006.

[58] NIST FIPS 140-2. *Security Requirements for Cryptographic Modules*. National Institute of Standards and Technology (NIST), 2001.

[59] Elie Noumon Allini, Maciej Skórski, Oto Petura, et al. “Evaluation and Monitoring of Free Running Oscillators Serving as Source of Randomness”. In: *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2018, no. 3, pp. 214–242, 2018.

[60] Venkata Rajesh Pamula, Xun Sun, Sung Min Kim, et al. “A 65-Nm CMOS 3.2-to-86 Mb/s 2.58 pJ/Bit Highly Digital True-Random-Number Generator With Integrated De-Correlation and Bias Correction”. In: *IEEE Solid-State Circuits Letters*, vol. 1, no. 12, pp. 237–240, 2018.

[61] Jieun Park, Yong Ki Lee, Karpinskyy Bohdan, et al. “A 60Mb/s TRNG with PVT-Variation-Tolerant Design Based on STR in 4nm”. In: *IEEE International Solid-State Circuits Conference (ISSCC)*, Feb. 2024.

[62] Adriaan Peetermans, Miloš Grujić, Vladimir Rožić, et al. “A Self-Calibrating True Random Number Generator”. In: *International Conference on Field Programmable Logic and Applications (FPL)*, Sept. 2019.

[63] Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede. “A Highly-Portable True Random Number Generator Based on Coherent Sampling”. In: *International Conference on Field Programmable Logic and Applications (FPL)*, Sept. 2019.

[64] Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede. “Design and Analysis of Configurable Ring Oscillators for True Random Number Generation Based on Coherent Sampling”. In: *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 14, no. 2, pp. 1–20, 2021.

[65] Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede. “Random Number Generator”. U.S. Patent 10761809B1, Mar. 12, 2020; South Korea Patent 102170985B1, Oct. 29, 2020; Japan Patent 6960697B2, Nov. 5, 2021.

[66] Adriaan Peetermans, and Ingrid Verbauwhede. "An Energy and Area Efficient, All Digital Entropy Source Compatible with Modern Standards Based on Jitter Pipelining". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2022, no. 4, pp. 88–109, 2022.

[67] Adriaan Peetermans, and Ingrid Verbauwhede. "Characterization of Oscillator Phase Noise Arising From Multiple Sources for ASIC True Random Number Generation". In: *IEEE Transactions on Circuits and Systems I (TCAS I)*, vol. 71, no. 3, pp. 1144–1157, 2024.

[68] Adriaan Peetermans, and Ingrid Verbauwhede. "TRNG Entropy Model in the Presence of Flicker FM Noise". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2024, no. 4, pp. 285–306, 2024.

[69] Matthias Peter, and Werner Schindler. *A Proposal for: Functionality Classes for Random Number Generators*. Bundesamt für Sicherheit in der Informationstechnik (BSI), 2022.

[70] Oto Petura, Ugo Mureddu, Nathalie Bochard, et al. "A Survey of AIS-20/31 Compliant TRNG Cores Suitable for FPGA Devices". In: *International Conference on Field Programmable Logic and Applications (FPL)*, Aug. 2016.

[71] Tao Pi, and Patrick J. Crotty. "FPGA Lookup Table with Transmission Gate Structure for Reliable Low-Voltage Operation". U.S. Patent 6667635B1, Dec. 23, 2003.

[72] Gudlavalleti Rajahari, Yashu Anand Varshney, and Subash Chandra Bose. "A Novel Design Methodology for High Tuning Linearity and Wide Tuning Range Ring Voltage Controlled Oscillator", July 2013.

[73] Vladimir Rožić, Bohan Yang, Wim Dehaene, et al. "Highly Efficient Entropy Extraction for True Random Number Generators on FPGAs". In: *Annual Design Automation Conference (DAC)*, June 2015.

[74] Andrew Rukhin, Juan Soto, James Nechvatal, et al. *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*. National Institute of Standards and Technology (NIST), 2010.

[75] Markku-Juhani O. Saarinen. "On Entropy and Bit Patterns of Ring Oscillator Jitter". In: *Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, Dec. 2021.

[76] Y. Savaria, D. Chtchvyrkov, and J.F. Currie. "A Fast CMOS Voltage-Controlled Ring Oscillator". In: *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 1994.

[77] Werner Schindler, and Wolfgang Killmann. “Evaluation Criteria for True (Physical) Random Number Generators Used in Cryptographic Applications”. In: *Cryptographic Hardware and Embedded Systems (CHES)*, Aug. 2003.

[78] C. E. Shannon. “A Mathematical Theory of Communication”. In: *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[79] Mathilde Soucarros, Cecile Canovas-Dumas, Jessy Clediere, et al. “Influence of the Temperature on True Random Number Generators”. In: *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST)*, June 2011.

[80] Berk Sunar, William Martin, and Douglas Stinson. “A Provably Secure True Random Number Generator with Built-In Tolerance to Active Attacks”. In: *IEEE Transactions on Computers*, vol. 56, no. 1, pp. 109–119, 2007.

[81] Sachin Taneja, Viveka Konandur Rajanna, and Massimo Alioto. “36.1 Unified In-Memory Dynamic TRNG and Multi-Bit Static PUF Entropy Generation for Ubiquitous Hardware Security”. In: *IEEE International Solid- State Circuits Conference (ISSCC)*, Feb. 2021.

[82] Thomas E. Tkacik. “A Hardware Random Number Generator”. In: *Cryptographic Hardware and Embedded Systems (CHES)*, Aug. 2003.

[83] Meltem Sönmez Turan, Elaine Barker, John Kelsey, et al. *Recommendation for the Entropy Sources Used for Random Bit Generation.* National Institute of Standards and Technology (NIST), 2018.

[84] Boyan Valtchanov, Alain Aubert, Florent Bernard, et al. “Modeling and Observing the Jitter in Ring Oscillators Implemented in FPGAs”. In: *International Workshop on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, Apr. 2008.

[85] Boyan Valtchanov, Viktor Fischer, and Alain Aubert. “Enhanced TRNG Based on the Coherent Sampling”. In: *International Conference on Signals, Circuits and Systems (SCS)*, Nov. 2009.

[86] Michal Varchola, and Milos Drutarovsky. “New High Entropy Element for FPGA Based True Random Number Generators”. In: *Cryptographic Hardware and Embedded Systems (CHES)*, Aug. 2010.

[87] Ihor Vasyltsov, Eduard Hambardzumyan, Young-Sik Kim, et al. “Fast Digital TRNG Based on Metastable Ring Oscillator”. In: *Cryptographic Hardware and Embedded Systems (CHES)*, Aug. 2008.

[88] F Vernotte, and E Lantz. “Metrology and $1/f$ Noise: Linear Regressions and Confidence Intervals in Flicker Noise Context”. In: *Metrologia*, vol. 52, no. 2, pp. 222–237, 2015.

[89] J. Wang, P. Ghanta, and S. Vrudhula. "Stochastic Analysis of Interconnect Performance in the Presence of Process Variations". In: *International Conference on Computer Aided Design (ICCAD)*, Nov. 2004.

[90] T.C. Weigandt, Beomsup Kim, and P.R. Gray. "Analysis of Timing Jitter in CMOS Ring Oscillators". In: *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 1994.

[91] Piotr Zbigniew Wieczorek, and Krzysztof Golofit. "Dual-Metastability Time-Competitive True Random Number Generator". In: *IEEE Transactions on Circuits and Systems I (TCAS I)*, vol. 61, no. 1, pp. 134–145, 2014.

[92] Xilinx, AMD. *Vivado Design Suite User Guide Implementation*. Dec. 2018. URL: https://docs.xilinx.com/v/u/2018.3-English/ug904-vivado-implementation (visited on 11/28/2023).

[93] Bohan Yang, Vladimir Rožic, Miloš Grujić, et al. "ES-TRNG: A High-throughput, Low-area True Random Number Generator Based on Edge Sampling". In: *IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES)*, vol. 2018, no. 3, pp. 267–292, 2018.

[94] Bohan Yang, Vladimir Rožić, Miloš Grujić, et al. "On-Chip Jitter Measurement for True Random Number Generators". In: *Asian Hardware Oriented Security and Trust Symposium (AsianHOST)*, Oct. 2017.

[95] Jing Yang, Yuan Ma, Tianyu Chen, et al. "Extracting More Entropy for TRNGs Based on Coherent Sampling". In: *Security and Privacy in Communication Networks (SecureComm)*, Oct. 2016.

[96] Kaiyuan Yang, David Blaauw, and Dennis Sylvester. "An All-Digital Edge Racing True Random Number Generator Robust Against PVT Variations". In: *IEEE Journal of Solid-State Circuits (JSSC)*, vol. 51, no. 4, pp. 1022–1031, 2016.

[97] Kaiyuan Yang, David Fick, Michael B. Henry, et al. "16.3 A 23Mb/s 23pJ/b Fully Synthesized True-Random-Number Generator in 28nm and 65nm CMOS". In: *IEEE International Solid- State Circuits Conference (ISSCC)*, Feb. 2014.

[98] Yun-Hsueh Chuang, Sheng-Lyang Jang, Jian-Feng Lee, et al. "A Low Voltage 900MHz Voltage Controlled Ring Oscillator with Wide Taning Range". In: *IEEE Asia-Pacific Conference on Circuits and Systems*, Dec. 2004.

Curriculum Vitae

Adriaan Peetermans commenced his pursuit of a PhD degree in October 2018 with the COSIC research group at KU Leuven, supervised by Prof. dr. ir. Ingrid Verbauwhede. Supported by a personal grant from the Research Foundation - Flanders (FWO), his research focuses primarily on cryptographic hardware primitives, including True Random Number Generators and Physical Unclonable Functions.

List of Publications

International Journals

Design and Analysis of Configurable Ring Oscillators for True Random Number Generation Based on Coherent Sampling
Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
ACM Transactions on Reconfigurable Technology and Systems (TRETS), 2021

An Energy and Area Efficient, All Digital Entropy Source Compatible with Modern Standards Based on Jitter Pipelining
Adriaan Peetermans, and Ingrid Verbauwhede
IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES), 2022

Characterization of Oscillator Phase Noise Arising From Multiple Sources for ASIC True Random Number Generation
Adriaan Peetermans, and Ingrid Verbauwhede
IEEE Transactions on Circuits and Systems I (TCAS I), 2024

SCALLER: Standard Cell Assembled and Local Layout Effect-Based Ring Oscillators
Muayad J. Aljafar, Zain Ul Abideen, Adriaan Peetermans, Benedikt Gierlichs, and Samuel Pagliarini
IEEE Embedded Systems Letters, 2024

TRNG Entropy Model in the Presence of Flicker FM Noise
Adriaan Peetermans, and Ingrid Verbauwhede
IACR Transactions on Cryptographic Hardware and Embedded Systems (TCHES), 2024

International Conferences with Proceedings

A Highly-Portable True Random Number Generator Based on Coherent Sampling
Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
International Conference on Field Programmable Logic and Applications (FPL), Sept. 2019

Attacking Hardware Random Number Generators in a Multi-Tenant Scenario
Yrjo Koyen, Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
Workshop on Fault Detection and Tolerance in Cryptography (FDTC), Sept. 2020

Patents

Random Number Generator
Adriaan Peetermans, Vladimir Rožić, and Ingrid Verbauwhede
U.S. Patent 10761809B1, Mar. 12, 2020; South Korea Patent 102170985B1, Oct. 29, 2020; Japan Patent 6960697B2, Nov. 5, 2021

Demos

A Self-Calibrating True Random Number Generator
Adriaan Peetermans, Miloš Grujić, Vladimir Rožić, and Ingrid Verbauwhede
International Conference on Field Programmable Logic and Applications (FPL), Sept. 2019

FACULTY OF ENGINEERING SCIENCE
DEPARTMENT OF ELECTRICAL ENGINEERING
COSIC
Kasteelpark Arenberg 10, bus 2452
B-3001 Leuven
adriaan.peetermans@esat.kuleuven.be
https://www.esat.kuleuven.be/cosic/