Designing, Implementing, and Testing Hardware for Cybersecurity

James Brown

A thesis submitted in partial fulfilment of the requirements of Liverpool John Moores University for the degree of Doctor of Philosophy

October 2020

Acknowledgment

First, I would like to thank Prof. Jianfu Zhang and Prof. Zhigang Ji who have both been my director of studies and 2nd supervisor at different times. Their continuous support, guidance, and encouragement have been invaluable during my PhD research project.

Secondly, I would like to thank my 3rd supervisor Dr Bo Zhou. Dr Zhou's insights into the application side of my research has been tremendously helpful and broadened the scope of my work beyond what it could otherwise have been. I would also like to thank Dr Brahim Benbakhti for reviewing my transfer report and providing me with solid advice throughout my whole PhD project.

I also wish to thank everyone else who I worked with as part of the LJMU microelectronics reliability and characterization group. They are Dr Rui Gao, Dr Zheng Chai, Dr Firas Hatem, Miss Mehzabeen Mehedi, Mr Pedro Freitas, Mr Chengda Shen, Mr Kean Tok, and Mr Dale Hodgkinson. Dr Gao greatly helped me to improve my characterisation skills and was always available to help with any difficulties I faced when using lab equipment.

Next, I thank the other support staff of the university. This includes everyone in the engineering administration office, particularly Natasha Walden-Jones, who were always available to help with any issues that I had. I would also like to thank Mr Stephen Gotts for his advice and support with any lab related issues through my entire time at LJMU.

Finally, I would like to express my deepest thanks and love to all of my friends and family.

Abstract

Cybersecurity is one of the key issues facing the world today. With an ever-increasing number of devices connected across the internet, the need to secure all these different devices against potential attackers is an endless effort. This thesis is focussed on the most promising new developments in the hardware aspect of this battle for security. The first section of the thesis looks at what is the current state of the art when it comes to hardware security primitives, with a focus on random number generators and Physically Unclonable Functions (PUF). The strengths and weakness of the current implementations of these systems are analysed so that the areas which are most in need of improvement can be highlighted.

The second major section of this thesis is looking to improve how random numbers are generated, which is essential for many current security systems. True random number generators have been presented as a potential solution to this problem but improvements in output bit rate, power consumption, and design complexity must be made. In this work we present a novel and experimentally verified true random number generator that exclusively uses conventional CMOS technology as well as offering key improvements over previous designs in complexity, output bit rate, and power consumption. It uses the inherent randomness of telegraph noise in the channel current of a single CMOS transistor as an entropy source. For the first time, multi-level and abnormal telegraph noise can be utilised, which greatly reduces device selectivity and offers much greater bit rates. The design is verified using a breadboard and FPGA proof of concept circuit and passes all 15 of the NIST randomness tests without any need for post-processing of the generated bitstream. The design also shows resilience against machine learning attacks performed by an LSTM neural network.

The third major section describes the development of a novel PUF concept, which offers a new approach to authentication, allowing low power devices to be included in existing networks without compromising overall security. The new PUF concept introduces time dependence to vastly increase the efficiency of entropy source usage, when compared with a traditional PUF. This new PUF also introduces a probability-based model which greatly reduces the required server memory for Challenge Response Pair (CRP) storage when large numbers of CRPs are used. The concept is verified experimentally on nano-scale CMOS technology as well as through simulation and a proof-of-concept circuit. These combined benefits bring the PUF concept much closer to being a viable solution for widespread cybersecurity applications.

List of Abbreviations

Abbreviation	Definition
ACRTN	Alternating current random telegraph noise measurement technique
BTI	Bias temperature instability
CMOS	complementary metal-oxide-semiconductor
CPUF	Controlled physically unclonable function
CRP	Challenge response pair
FET	Field effect transistor
FF	FinFet
FIFO	First in first out buffer
FPGA	Field programable gate array
GPIB	General Purpose Interface Bus
HC	Hot carrier
HCI	hot carrier injection
HMM	Hidden Markov model
IC	Integrated circuit
IV	Drain current (Id)~Gate voltage (Vg)
LRPUF	Logically reconfigurable physically unclonable function
LSTM	Long Short-Term Memory
MOS(FET)	Metal Oxide Semiconductor (Field Effect Transistor)
MRAM	Magnetic random access memory
NIST	The National Institute of Standards and Technology
OPSO	Overlapping-pairs-sparse-occupancy
OQSO	Overlapping-quadruples-sparse-occupancy
PDF	Power density function
PPUF	Public physically unclonable function
PRNG	Pseudo random number generator
PUF	Physically unclonable function
RAM	Random access memory
RFID	Radio-frequency identification
RNG	Random number generator
RO	Ring oscillator
RPUF	Reconfigurable physically unclonable function
RRAM	Resistive random access memory
RSA	Rivest–Shamir–Adleman algorithm
RTN	Random telegraph noise
SIMPL	simulation is possible but laborious
SMU	Source measurement unit
SNR	Signal to noise ratio
SRAM	Static random-access memory
TDDB	Time-dependent dialectic breakdown

TDDS	Time-Dependent Defect Spectroscopy
TRNG	True random number generator
UART	Universal Asynchronous Receiver/Transmitter
VHDL	Very High Speed Integrated Circuit Hardware Description Language
WGFMU	Waveform generator fast measurement unit

List of Symbols

Symbol	Description	Unit
Fac	AC switching frequency for ACRTN measurement	Hz
$Fosc_{min}$	Minimum oscillator frequency	Hz
fsp	Sampling frequency	Hz
gm	Transconductance of an Id-Vg curve	S
H(X)	Information Entropy	
Id	Drain Current	A
Idcc	Constant Id value for Vth extraction	A
Ig	Gate leakage current	A
k	maximum entropy possible from a system	
n	number of possible output states from a system	
T	Temperature	
Tox	Oxide thickness	m
Vd	Applied drain voltage	V
Vg	Applied gate voltage	V
VgH	High voltage value for ACRTN measurements	V
VgL	Low voltage value for ACRTN measurements	V
Vinj	Trap charging voltage for TDDS	V
Vth	Threshold voltage	V
Xt	Trap position in vertical direction	
ΔVth	Threshold voltage shift	V
τ0	Crossover point of τc and τe	s
τc	Capture time of RTN defect	s
τe	Emission time of RTN defect	s

List of Figures

List of Tables

Contents

Chapter 1 - Motivation of the project

Cybersecurity is a huge and growing industry with cyberattacks being listed as a tier 1 threat to the UK in the government's 2011-2016 and 2016-2021 "Cyber Security Strategy Reports" [1], [2]. The Internet of Things (IoT) is a new and rapidly growing market. It is expected to reach a value of $28.9 Billion by 2020 [3] and cybersecurity is one of the key parts that must be improved in order to unlock the potential of the IoT [4], [5]. At the heart of many modern cybersecurity systems is the random number generator [6]. The random number generator is the root for the current cryptographic systems such as encryption, authentication, Secure session links, car keys, and much more, but it is an area which is facing a huge problem [7].

Currently, the vast majority of security systems use pseudo-random number generators (PRNG). PRNGs use mathematical algorithms to create pseudo-random number streams and are incapable of creating truly random sequences [8]. As modern computers are becoming faster and more powerful, we are starting to see instances where PRNGs have been cracked or exploited for malicious activity. As this trend continues and becomes worse, we are facing the potential collapse of many of our cyber-security systems [5], [9]–[12]. To try and solve this problem new methodologies for creating random numbers have been explored. A leading new methodology is the use of "true random number generators" (TRNGs). TRNGs use an inherently random real-world phenomenon as the seed for the random number generation. Because the real-world source is not derived from mathematics, it cannot be predicted mathematically, no matter how much previous information an attacker has on it. They are naturally resistive to the cyber-attacks we are starting to see [13]. Despite the clear advantages of using TRNGs for cybersecurity purposes, TRNGs have not yet gained a major role within the industry. The primary reasons for this are deficiencies with one or more of the following in current designs: reliability, output bit rate, efficiency, power consumption, vulnerability to attack, or design footprint

[13]–[17]. The first aim of this project is to create a TRNG that can eliminate or improve upon these factors and meet the need of cyber-security systems, with a strong emphasis on application into IoT devices.

Another key task for current cybersecurity systems is authentication. Authentication is the process of verifying the identity of each entity in communication. Classical examples are the use of a password, pin number, or security questions. It has been known for many years that these classic authentication methods are not fully secure [18]–[23] and can be exploited by attackers [24]. Efforts have been made, but no solution has presented itself as the clear answer to all the problems. Biometrics offer some improvements for the security but still have limitations, such as recoverability if someone's biometric data is stolen, usability for people with disabilities that prevent them from using certain biometrics, limited entropy of biometric data making it cloneable, or inapplicability in cases such as IoT sensor nodes.

Physically/Physical Unclonable Functions (PUFs) have emerged as a possible solution to the weaknesses of current authentication systems. PUFs make use of the unique physical properties of a system embedded within them to create a unique set of responses to a given challenge input from the authenticating server. This unique set of challenge-response pairs are stored on the authentication server. When authentication with an external party is requested, the challenge is issued. If the party requesting authentication can reply with the correct response, authentication is successful. PUFs are particularly attractive for use within the IoT as they can potentially be much more efficient in terms of power and design footprint, when compared with traditional authentication methods [25], [26]. Despite the advantages of PUFs they are not yet widely used in real-world applications, because of key deficiencies in areas such as, reliability, clonability, key distribution, power consumption, design complexity, or design footprint [26]–[31]. The second aim of this project is to overcome or improve upon as many of these deficiencies as possible by developing a new PUF design and protocol tailored for use within the IoT.

Chapter 2 - Literature Review

In this chapter, the literature of the previously proposed designs for both TRNGs and PUFs is reviewed to outline the key concepts behind their operation as well as what level of performance they can achieve. Section 2.1 covers how the performance of TRNGs is accessed in terms of the randomness of the output. Section 2.2 focussed on the existing designs and what they can achieve. In section 2.3, attention will be paid to the various types of attacks being used against TRNGs and the protection methods against these attacks. Section 2.4 consists of an overview and analysis of the existing PUF designs, focussing on the most prolific type of PUF design, their working principles, as well as their drawbacks. The final section 2.5 covers the basic information of random telegraph noise (RTN) which will be used as the entropy source for both the TRNG and PUF developed in this project.

2.1 True Random Number Generators

2.1.1 Verification of Randomness

The first important concept to understand when analysing TRNGs is that of entropy. For TRNGs the type of entropy one is interested in is the Shannon entropy which gives the average rate at which information is created by a stochastic data source. TRNGs typically generate one bit at a time giving us two possible outputs, either '0' or '1'. Utilising equation 1, where 'n' is the number of possible outcomes, it can be calculated that the maximum entropy possible (k) to get from each output bit of a TRNG is equal to 1.

$$log_2(n) = k \tag{1}$$

If several bits are generated, one can calculate the information entropy of the TRNG using equation 2, where ‘H(X)’ is the information entropy of the system, ‘p’ is the probability of one of the two possible results and ‘q’ is the probability of the other.

$$H(X) = -(p \times log_2(p)) - (q \times log_2(q)) \tag{2}$$

Further exploring equation 2, one can see that if the probability of each outcome is equal the entropy is equal to the maximum possible value of ’1’ meaning there is no bias towards one result in the output. Conversely, if the probability of one of the outputs is certain (probability = 1) then the entropy calculated will be ‘0’ meaning the system is completely biased to one result and no randomness is produced.

For TRNGs to work and be used in the real world, it is essential to prove that the random numbers that it creates are truly random. Truly random processes, by definition, do not have memory, making it impossible for past outcomes to affect future outcomes. In this way, the true random number is impossible to predict, giving a system the highest security level. However, it is difficult to test if a number sequence is truly random. Several tools have been suggested for the test. The most widely used tests are discussed in this section.

Although the information entropy can give us a good quick impression of the randomness of a TRNG, it is not enough by itself to show that the output is truly random as it simply shows the probability of each result. For example, the information entropy of a TRNG that produces a bitstream of ‘1’ then ‘0’ repeatedly will have an entropy of ‘1’, but it is obviously not random as anyone can predict the next result with certainty. As a result of this several different test suits have been developed that can mathematically analyse a given bitstream to identify not just the entropy but also if there are any predictable patterns or other non-random events.

The first set of tests discussed in this section is the DIEHARD battery (of tests) [32], which was developed in 1995 by George Marsaglia. It consists of 12 different tests for randomness that must be passed for the output to be verified as random. These tests are birthday spacings, overlapping permutations, rank of matrices, monkey tests, count the 1s, parking lot test, min distance test, random spheres test, the squeeze test, overlapping sums test, runs test, and the craps test. The DIEHARD battery was used to verify many of the early TRNG designs but has now been replaced by the NIST test suite, which is to be discussed in the following section.

The second suite of randomness tests is the NIST test suite [33]. It was developed by the National Institute of Standards and Technology in 2010 and has now become the *de-facto* standard test suite to test random number generators and has been verified independently in works such as Sadique's 2012 review [34]. The NIST test suite is based on null hypothesis theory, where the null hypothesis is that the sequence is random. The tests are designed to predict the probability that this hypothesis is correct or not. Generally, the tests should be selected to approve the hypothesis if the tested sequences have less than 0.01% of disagreeing with it. The NIST suite consists of 15 tests: monobit frequency test, block frequency test, runs test, longest run of 1s in a block test, matrix rank test, discrete Fourier transform test, non-overlapping template test, overlapping template test, Maurer's universal statistical test, linear complexity test, serial test, approximate entropy test, cumulative sums test, random excursions test, and the random excursions variant test. The test suite is made to run on Linux systems and will be used to verify any random number generators developed in this project.

More recently, machine learning has also become a key part of both attacking and testing TRNGs [16], [33], [35]–[40]. Machine learning offers a new way of trying to identify patterns and non-random elements within a bitstream that would previously have been missed by conventional testing. The idea of using machine learning is to provide the machine learning algorithm with a series of previously generated bits and allow it to try and predict future generated bits. A common machine learning algorithm to use is a long short-term memory (LSTM) algorithm such as the one used in [38]. When looking at the results from machine learning testing, the ideal result is a 50% prediction rate as this is what one would expect when the result is a pure guess between two equally probably outcomes. Any deviation from 50% shows the machine learning was able to find some part of the bitstream that was not random and able to predict it.

2.1.2 Existing TRNG Designs

This section reviews the existing TRNG designs. A solid understanding of the existing designs is essential for this project both as a guide for further development of the design and as a benchmark for the performance that any new design must achieve in order to be commercially competitive. The most important designs for each type of TRNG are covered in detail and then other less notable designs are summarised in Table 1.

One of the earliest and most influential TRNG designs was by Intel in 1999 [7]. The block diagram is shown in Figure 1. This Intel design makes use of Johnson thermal noise across a resistor as the source of randomness. It is then amplified and used to control the frequency of a voltage-controlled oscillator. The rising edge of the voltage-controlled oscillator is used to trigger sampling of another much faster frequency oscillator. The current state of the high frequency oscillator determines whether the TRNG output is ‘0’ or ‘1’. To increase the entropy of the output bits and avoid chains of 0s or 1s, a Von Neumann post-processing circuit is used. Von Neumann post-processing is a technique which improves the entropy of the output bitstream in many earlier TRNG designs. Its basic principle is shown in Figure 2. Essentially, the Von Neumann technique eliminates long chains of ‘0’ or ‘1’ from the output bitstream as it only gives an output when there is a transition from one to the other. Although this allows the TRNG to have higher output entropy, it does come at the cost of efficiency.

Von Neumann processing has a high overhead in terms of design complexity, design footprint, and bit efficiency in terms of generated bits to final output bits. The inefficiencies of Von Neumann processing make most modern designs avoid it. The output bits were tested with the DIEHARD tests and shown to be random. No information on the bit rate, design area, or power consumption were given.

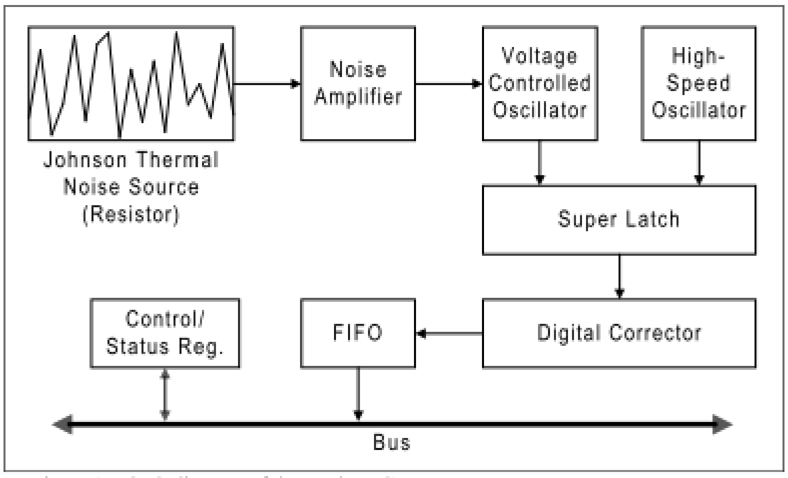
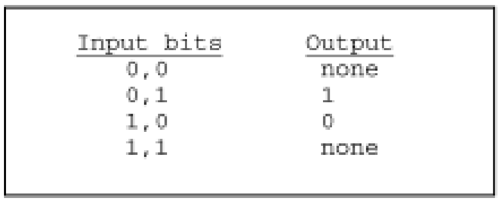


Figure 1, Block Diagram of Intel 1999 RNG[7].



Input bits	Output
0,0	none
0,1	1
1,0	0
1,1	none

Figure 2, Vonn Neumann Corrector Operation[7].

Metastability is a phenomenon that is frequently used for TRNG designs. One of the first influential designs is the 2008 Holleman et al. design [41]. As shown in Figure 3, the metastability is harvested from a pair of cross coupled latches. This design works by having two evenly matched latches triggered by the same clock signal. Depending on the current state of the noise source, either the left latch or the right latch will settle at 'high' and the other at 'low'. This circuit can be reset and triggered again at a fast rate in order to get a random bitstream. The main difficulty with this type of design is the balancing of the left and right latches so that the chance of a ‘0’ or ‘1’ output is 50%. The Holleman design uses a combination of circuits to remove correlated events from the noise source itself, including DC nulling on the latches to compensate for device mismatch and a post-processor based on non-linear feedback registers to remove any bias form the final bitstream. The Holleman design can achieve a bit rate of 50Kbit/s with a power consumption of 180uW and a design area of 1.49mm^2. The output randomness is verified by the NIST tests.

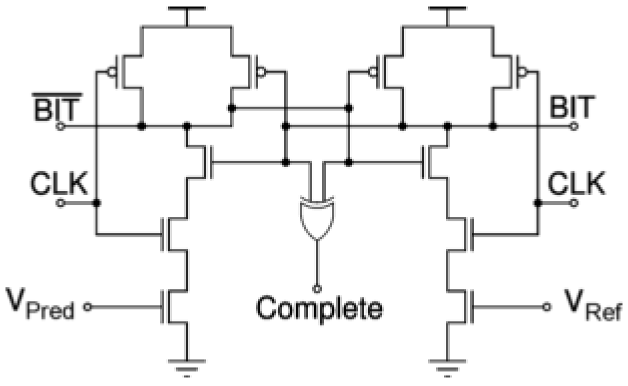


Figure 3, Cross coupled latch circuit used for metastability based TRNGs such as the 2008 Holleman design [41].

One of the most powerful existing TRNG designs is the 2012 Sanu et al. design developed at Intel [42]. This design is also based upon the metastability of cross coupled inverters. Figure 4 shows the random source circuit for this design. To harvest the randomness in this circuit, the Sanu design waits for the metastability to be resolved and then checks to see if the outputs ‘a’ and ‘b’ are ‘0’ and ‘1’ or ‘1’ and ‘0’ respectively. The key innovation is the tuning circuits that are used. As shown in Fig. 5, the Sanu design has 3 different tuning circuits, one for ‘coarse’ tuning, one for ‘fine’ tuning, and a self-calibrating feedback loop to select the tuning values. The coarse tuning is achieved through an array of selectable resistors on each side of the inverter that can be used to change the balance and account for device mismatch. The fine tuning is achieved through delay units before the input for each side of the inverter that can be adjusted to account for any small changes in environmental conditions. Both tuning systems are controlled by the feedback loop which extracts the probability of a ‘0’ or a ‘1’ output in a sample of bits and adjusting the tuning circuits to achieve a 50% probability of each. The output bits are tested and verified by the NIST statistical tests. All of this combines to create a TRNG with an output bit rate of 2.4Gb/s which is by far the fastest generation rate achieved by any design. However, at this generation rate the power consumption is 7mW and the design area is 4004um^2, making it power hungry, large, and not suitable for low power compact applications such as IoT edge devices.

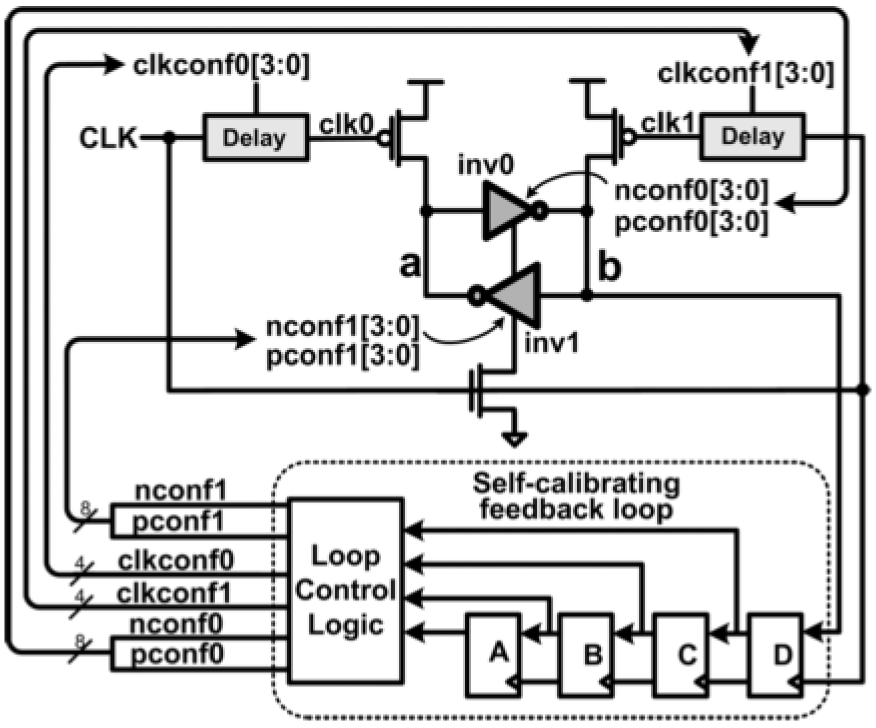


Figure 4, Random source circuit for 2012 Sanu TRNG [42].

The next key type of TRNG design is based on time-dependent dielectric breakdown (TDDB), first proposed by Liu et al. in 2011 [43]. Its block diagram and basic principles are shown in Figure 5. In this design the TDDB in an array of MOS capacitors is used. TDDB is a process where, after some stress time being applied to an MOS device, the oxide will break down and allow a current to flow through it. A counter is left running until a breakdown is detected. When a breakdown is detected the counter is stopped and will contain a number which is represented by the ‘Q’ output of each ‘count’ element in Figure 5. ‘N’ number of the most significant bits are then removed, because they are the least likely to change over multiple measurements. The remaining bits stored in the counter become the output of the TRNG. The number of most significant bits which are removed for each measurement (N) is a variable which can be adjusted. The effect of ‘N’ on the output bitstream’s ability to pass the NIST tests is shown in Figure 6. Each capacitor is estimated to be able to be tested ~5 billion times until the breakdown becomes permanent. This means that with the current array size, the TRNG will run out of devices after ~3-10 years. A major benefit of this design is that it requires no post-processing circuits to pass all the NIST tests. This design can generate random bits at 11kbits/s with a power consumption of 2mW and a design area of 0.0012mm^2. This makes the design smaller than most comparable designs but also slower and more power hungry.

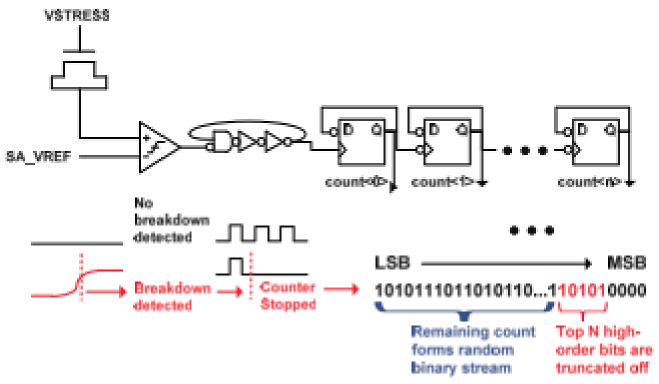


Figure 5, System diagram of 2011 Liu TDDB TRNG design [43].

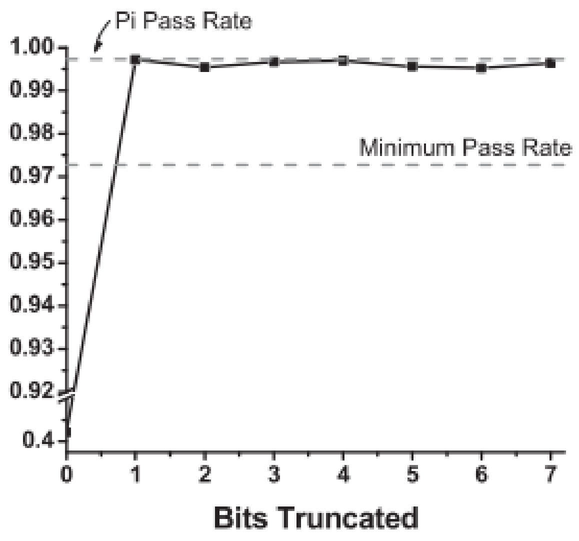


Figure 6, Effect of number of removed bits (N) on the pass rate of generated bitstreams when tested with the NIST test suite [43].

Oscillator jitter is another phenomenon that is frequently used in TRNGs to generate random bits. The 2014 design by Yang et al. is one of the most prominent designs based on oscillator jitter. As shown in Figure 7, it uses a ring oscillator that is modified with 3 input edges. These 3 edges propagate around the ring oscillator and each edge captures timing jitter, shifting the phase difference between the edges until two edges converge and the edge collapses. It is this time until collapse that is harvested to extract a random bit stream from the timing jitter. A second ring oscillator is used under normal operation to act as a reference so that a phase detector can capture the edge collapse event. Once a collapse is detected the number stored in the counter is extracted to give the random bits. The random bits generated by this design were verified by the NIST test suite. This design can achieve a bit rate of 23Mbits/s with a design area of 375um^2 and a power consumption of 0.54mW. This makes it faster and smaller than most comparable designs, although they only state "TRNG Core Area" which could be misleading. It also consumes lower power than other comparable designs, apart from the Random Telegraph Noise based TRNGs.

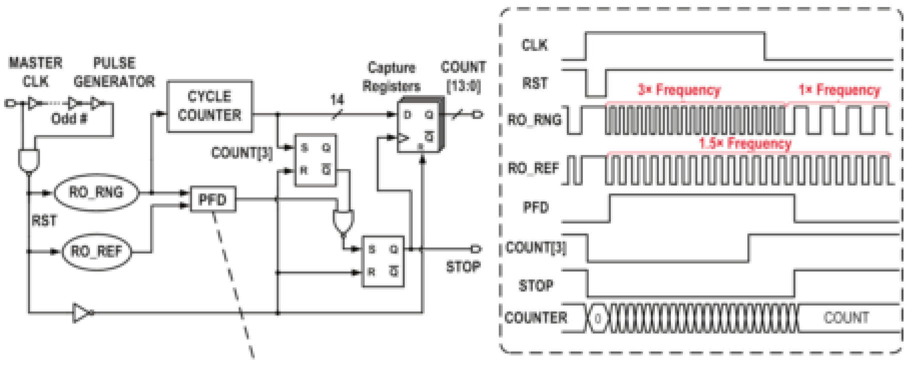


Figure 7, Block diagram of the 2014 Yang Oscillator Jitter based TRNG [44].

An emerging type of TRNG uses new memristor technology. The designs by Rai et al. [17] and Hao et al. [45] are two examples of this. Both designs make use of the static and dynamic stochastic behaviour of the time it takes to set and reset memristors to generate a random bitstream. The basic schematic of a memristor-based TRNG is shown in Figure 8. The working principle of the memristor-based design is that a write voltage is sent to a memristor and the time taken for the memristor to set is where the entropy is extracted from, as outlined in Figure 9. For Rai's work the time taken for the memristor to set itself is measured by an oscillator-controlled counter which is shown in Figure 8. When the memristor is set, the voltage across it exceeds the reference voltage and a comparator is used to signal the oscillator-controlled counter to stop counting. The value recorded within the counter is then used as the random numbers. There are still some issues with the current memristor designs. The first is the control, the reference voltage that is used to determine when the memristor is at the set voltage. The voltage used as reference must be set very accurately which increases design complexity as well as the fact that exact voltage required varies by devices so it must be adjustable. The second major issue is reliability of memristor devices. Currently memristor technology is not mature and the number of read and writes that each device can perform without breakdown is not enough for widespread use [46], [47]. The immaturity of memristor technology is also an issue in terms of availability and cost, as novel technology is much less attractive for low cost, high volume applications when compared with conventional CMOS technology. The final issue for the memristor designs is that neither of these two examples can pass the NIST statistical tests for randomness with the recommended level of significance (α) of 0.01 [33]. Due to the failure to pass the NIST tests with the recommended settings, these TRNGs are not suitable for real world implementation until further work is done.

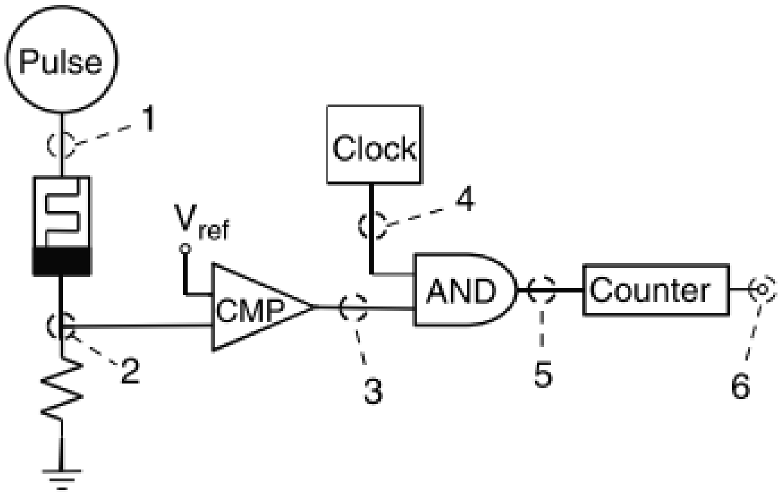


Figure 8, Basic circuit schematic of a memristor based TRNG [45].

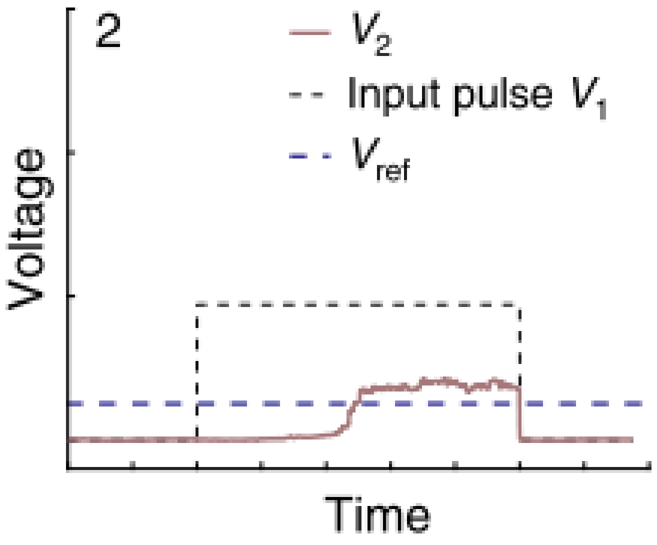


Figure 9, Operating principle of memristor based TRNGs [45].

An early TRNG design, and also a very important one for this work as it is based on the same Random Telegraph Noise (RTN) phenomenon that our design is based upon, is the 2006 Brederlow et al. design [48]. The Brederlow et al. design is the first to propose the use of RTN as a source for randomness, Figure 10 shows the block diagram for the design. The 2006 Brederlow et al. design amplifies the RTN signal (solid black signal in Figure 11) from a single transistor and then samples this RTN signal based on the timing of rising edges of a sampling clock (dashed black line in Figure 11) in order to obtain the random number (random numbers in Figure 11). This design also uses a Von Neumann correction post- processing circuit. This design is verified using the NIST randomness tests. One drawback of this design is that an RTN signal of the required speed and amplitude is very rare. Because of this, a very large transistor array (up to several million devices) is used so that a device with the required properties can be found. Another drawback is that it cannot extract entropy from multi-level RTN signals which have greater than 2 current levels. This design can produce random numbers in the order of ~200kbits/s with a power consumption of 50uW and a design area of 9000um^2. It is not fast compared with other designs of a comparable size, but it has low power thanks to the single transistor nature of the RTN signal.

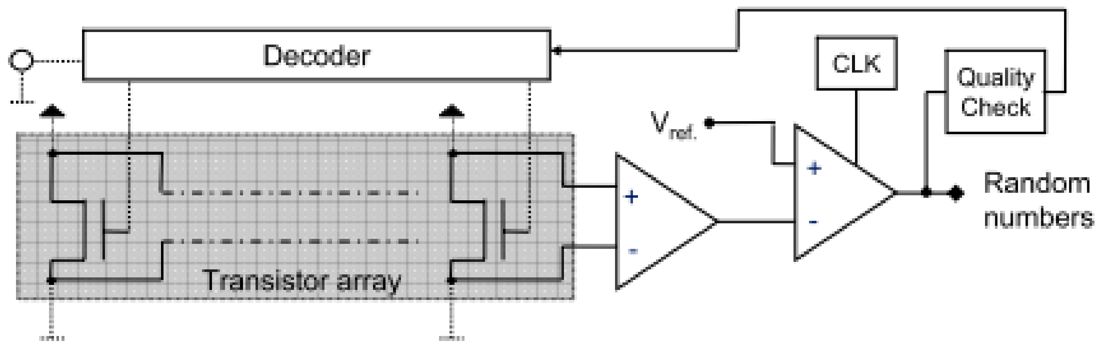


Figure 10, Block diagram of 2006 Brederlow TRNG design[48].

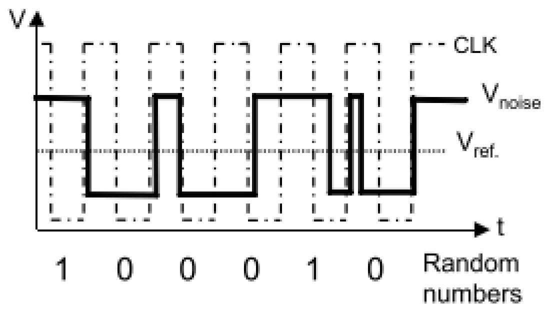


Figure 11, Amplified RTN, sampling clock signal, and example number outputs for Brederlow TRNG [48].

In 2017 another RTN based TRNG was proposed by Abinash Mohanty et al. [37]. It harvests the RTN as a random source in much the same way as that proposed by Brederlow [48]. However, it offers two key improvements over Brederlow’s original design. The first one is in the post-processing technique that avoids the need for a complex Von Neumann post-processor. The new post-processing technique is outlined in Figure 12. The figure shows that the output signal is simply toggled every time a rising edge is detected in the amplified and digitised RTN signal. This makes use of the fact that the average time between two rising edges is constant so the ideal 50% of a final ‘1’ or ‘0’ is achieved without any complex post-processing circuits [17].

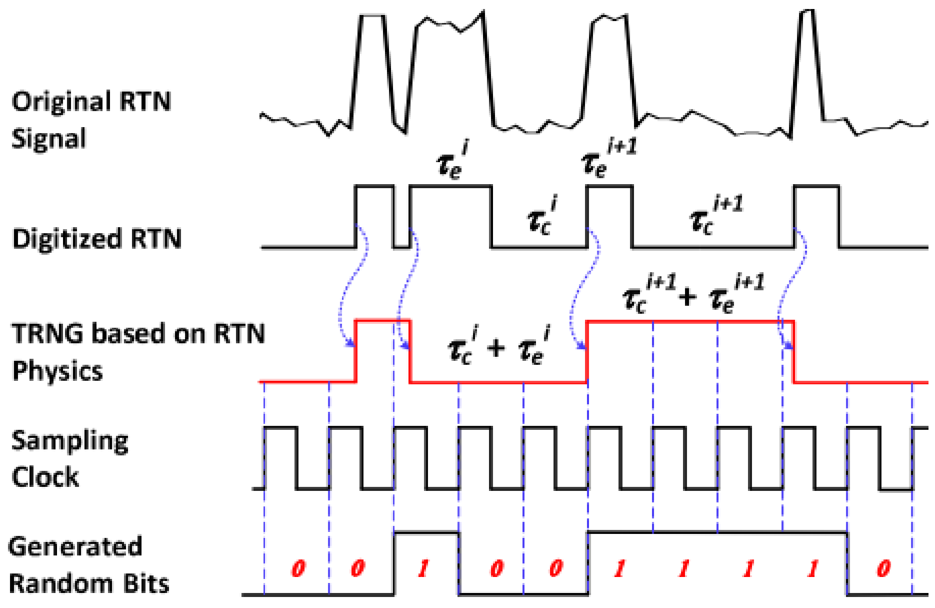


Figure 12, New post-processing method proposed by Mohanty [37].

The new post-processing method simplifies the TRNG system, making it viable for low power applications. It, however, does introduce some new issues. With traditional Von Neumann post-processing, long chains of ‘0’ or ‘1’ caused by oversampling of the RTN source were eliminated from the output bitstream. Without it, the sampling clock must be closely matched with the RTN signal to avoid oversampling. This oversampling issue adds design complexity as a frequency tuning system must be implemented. It also makes the design susceptible to failure if an unstable or abnormal RTN is present, which makes the time at the high and low states of the amplified RTN signal diverge from the typical distribution.

The second key innovation that Mohanty’s paper introduces is a thresholding technique that allows the TRNG to extract entropy from sources that exhibit multi-level RTN. The basic principles of this thresholding technique are shown in Figure 13. The idea is to select a threshold just above the minimum value that the RTN signal will reach and state that anything below this value will be counted as ‘0’ and anything above it will be counted as ‘1’. This allows multi-level RTN to be used. It means, however, that the entropy of all traps apart from the slowest one is ignored. As only the entropy from the slowest traps is extracted, this design suffers from slow operational speeds of only around 3 bits per second.

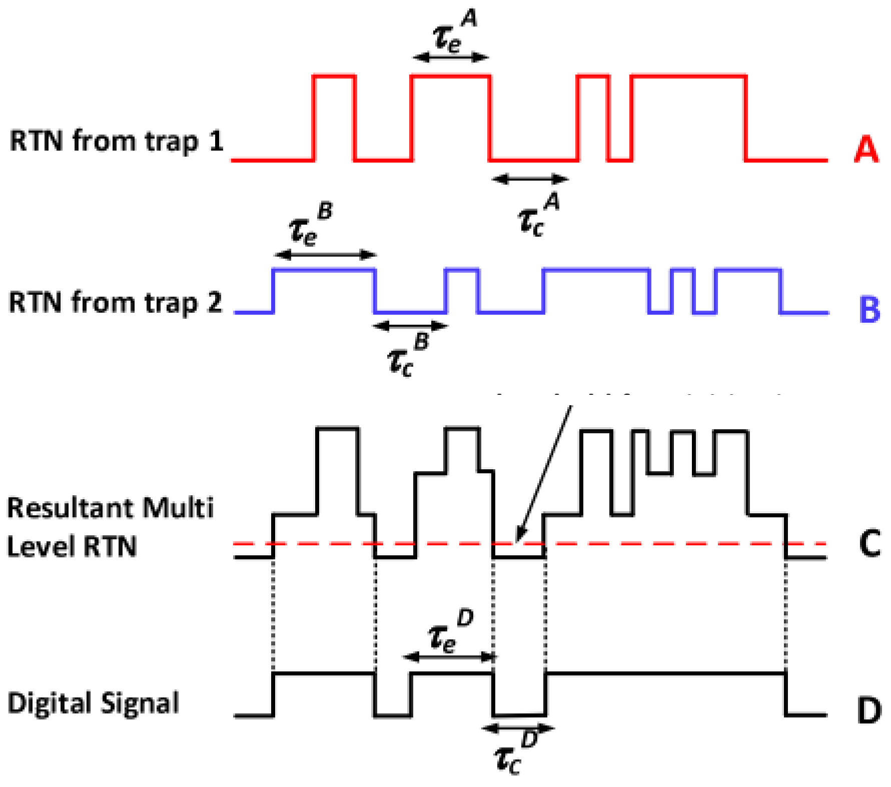


Figure 13, Thresholding technique for multi-level RTN entropy sources as proposed by Mohanty [37].

Finally, for clarity, the key TRNGs designed in the last few years are summarized in Table 1.

Table 1, Key points of TRNG designs

Authors	Type of random source	Generation rate	Design area	Power consumption	Verification
Sunar et al. 2007 [49]	Ring Oscillator Jitter	2.5 Mbit/s	N/A	N/A	N/A
Tokunaga et al. 2008 [50]	Metastability of inverters	200 kbit/s	$0.036mm^2$	1mW	Passes NIST at 20%-bit efficiency
Rahman et al. 2014 [51]	Power Supply Noise	N/A	N/A	N/A	Passes NIST
Sanu et al. 2015 [52]	Metastability of inverters	162.5 Mbit/s	$1088um^2$	1.5mW	Passes NIST
Figliolia et al. 2015 [53]	RTN with sigma delta converter	1-25 Mbit/s	$2200um^2$	432nW per MHz	N/A
Wieczorek et al. 2016 [54]	Metastability of inverters	1/0.16 Mbit/s	271 slices/199 cells	90mW	Passes NIST
Gimenez et al. 2017 [55]	Ring Oscillator Jitter	N/A	N/A	N/A	FIPS 140-1 passed
Kim 2017 et al. [56]	Ring Oscillator Jitter	9.9 Mbit/s	$920um^2$	0.418mW	Passes NIST with < 4-bit

					simultaneous output
Kim 2017 et al. [40]	RTN in ReRAM	N/A	N/A	N/A	Passes NIST
Matsumoto et al. 2008 [57]	Tunnelling in SiN device	0.3 Mbit/s	1200um^2	N/A	Passes FIPS 140-2
Pareschi et al. 2006 [58]	Chaotic Map Based	40 Mbit/s	0.52mm^2	30mW	Passes NIST
Jun et al. 1999[7]	Thermal Noise in Ring Oscillator	N/A	N/A	N/A	Passes DIEHARD
Brederlow et al. 2006 [48]	Random Telegraph Noise	200 kbits/s	0.009mm^2	50uW	Passes NIST
Holleman et al. 2008 [41]	Metastability of inverters	50kbits/s	1.49mm^2	180uW	Passes NIST
Liu et al. 2011 [43]	Time Dependent Dielectric Breakdown	110 kbits/s	0.0012mm^2	2mW	Passes NIST
Soucarros et al. 2013 [59]	Ring Oscillator Jitter	N/A	N/A	N/A	Passes NIST
Yang et al. 2014 [44]	Ring Oscillator Jitter	23 Mbits/s	0.000375mm^2	0.54mW	Passes NIST
Chen et al. 2015 [60]	Random Telegraph Noise	0.25-5 Mbits/s	N/A	N/A	Passes NIST
Jiang et al. 2017 [45]	Memristor Write Time	0.006mm^2	N/A	N/A	Passes NIST with reduced α value
Liu et al. 2017 [61]	Ring Oscillator Jitter	100 Mbits/s	N/A	N/A	Passes NIST
Mohanty et al. 2017 [37]	Random Telegraph Noise	3 bits/s	N/A	N/A	Passes NIST
Rai et al. 2017 [17]	Memristor Write Time	N/A	N/A	N/A	Fails NIST
Pamula et al. 2018 [62]	Metastability of Inverters	86 Mbits/s	0.01mm^2	523uW	Passes NIST
Yang et al. 2018 [63]	MRAM Write Time	66 Mbits/s	0.00018mm^2	N/A	Passes NIST
Chai et al. 2019 [38]	Ovonic Threshold Switching in RRAM	~100Mbits/s	N/A	N/A	Passes NIST with reduced α

2.1.3 Attacks on TRNGs

Attacks on TRNGs can be split into two main categories, invasive and non-invasive. Non-invasive attacks use an external influence such as temperature, voltage supply, or strong magnetic fields to affect the randomness of the TRNG. Invasive attacks are where the circuit of the TRNG is directly tampered with, such as the destruction of transistors with lasers or tapping of new connections to the system [49]. The effect of attacks can also be put into two main categories, degraded performance, and compromised distribution of events.

Gimenez et al.'s 2017 paper [55] covers several types of attack, some are exclusive to ring oscillator based TRNGs and some are for all TRNG designs. Figure 14 shows a block diagram of the most prevalent types of attack on TRNGs. The first type of attack covered is applicable to all TRNG designs and it is known as overclocking. In overclocking the time period of the clock is reduced in order to introduce timing violations into the system, which allows the attacker to manipulate the output bits. The best proposed solution to this type of attack is to use an internal clock signal for control of the circuit which makes it harder for attackers to gain any access to the clock system. The second type of attack covered is oversampling, where an attacker changes the ratio between the sampling circuit frequency and the system clock frequency. This results in a reduction of the entropy of the output bits. Again, the best way to solve this problem is to use internal signals for sampling and clocking. Fault injection attacks are also covered by Gimenez in his paper, which are specific to ring oscillator designs. The examples are forcing the oscillator into the incorrect mode of operation, such as burst-mode, phase covering mode, or bottleneck mode. When the oscillator is in an incorrect operation mode, the correct behaviour of the TRNG can no longer be guaranteed. Gimenez's paper also outlined two methods on how these incorrect operation modes can be caused. The first of these is the token/bubble injection attack where the number of events propagating around the oscillator is modified which can force the oscillator into one of the previously mentioned incorrect modes and therefore remove the output randomness. The second method is called delay modification and is achieved through creating an artificial delay in some nodes of the oscillator which can cause the system to enter bottleneck mode. The paper recommends using a prime number of nodes in the oscillator as well as an event counter and continuous output monitoring in order to prevent these attacks. This recommendation is supported by the fact that both the FIPS and AIS31 standards require the output bits to be monitored and evaluated continuously anyway.

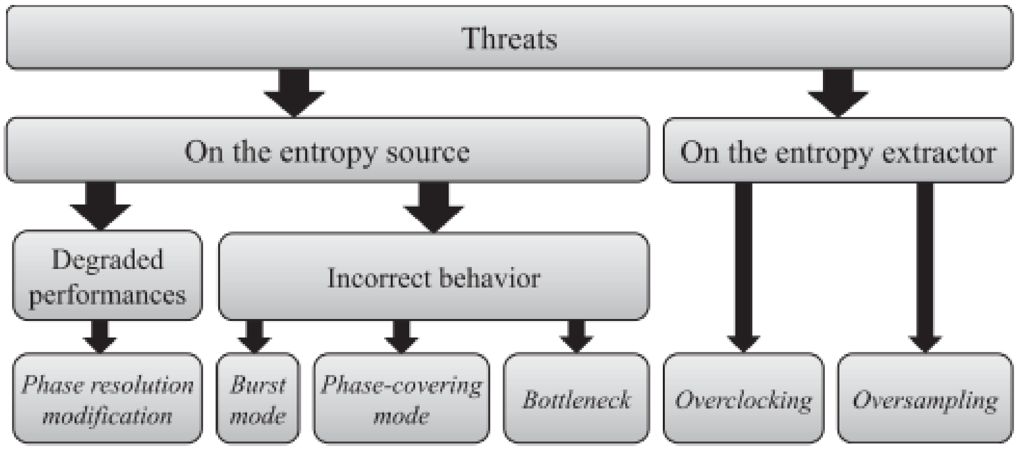


Figure 14, Common Attack Types on TRNGs[55].

Kim et al.’s 2017 paper [56] has a strong focus on power-injection attacks on ring oscillator based TRNGs. Power-injection attacks consist of applying an AC signal on to the power supply to the TRNG. The aim is for the applied AC signal to mask the random noise that is being used as a source and replace it with a deterministic signal. If the source for the TRNG becomes deterministic then the output will also be deterministic and the TRNG has failed. Kim et al.’s design makes use of additional feedback resistors in order to remove any bias in the system which makes it more robust against power-injection. The results, however, show the attack can still cause the TRNG to fail when the AC signal is at the correct frequency.

2.2 Physically/Physical Unclonable Functions

There are multiple types of PUF, and they are not limited to solely electronics-based designs. Even the name itself has inconsistency as the first design that could be described as a PUF was coined a Physical One-Way Function in the work by Pappu et al.[64]. Soon after that the term Physical random function was coined by Gassend et al.[65], which is then updated to Physical Unclonable Function to avoid having the same acronym as the Pseudo-Random Functions. Since then, others have used the term Physically Unclonable Function to describe designs that fit the same concept. To avoid confusion in this work, all designs that fit the same concept will be described under the common label of a PUF. The most basic principle of a PUF that all designs have in common is that they are a system that utilises some physical parameter of itself to generate a set of challenge and response pairs (CRPs). In this CRP system, a challenge can be issued to the PUF by an external entity, then the PUF will itself generate a unique response to that specific challenge that can be used to identify that individual PUF. Outside of this basic principle the variation between PUF designs can be significant. In the next section we will identify common characteristics and features of PUFs that have previously been developed and use this to focus on the specific sub-classes of PUF that we are interested in for this project.

To help with evaluation of PUF designs Maes et al [64] proposed a list of key properties that are desirable for PUF designs to possess and make comparison between different PUF designs more structured and easily digestible. The strictness of these parameters is dependent upon the limitations of the intended application of a PUF. The parameters are as follows:

Constructability – For useful PUF designs, they must be constructible. This means that it must be physically possible to manufacture it. It is desirable for a design to be more constructible which means easier to manufacture. It is important to note, however, that this means it should be easy to construct an example of the given PUF design with undefined CRPs and not that it should be easy to make a PUF with a defined set of CRPs.

Evaluability – For useful PUF designs, it must be possible to evaluate the responses it gives so that the CRP can be verified. Generally, the easier the evaluation of responses, the better the PUF design is. In practice a PUF can be considered to have good evaluability if the response can be evaluated within the given restrictions of the application that a PUF is used for.

Reproducibility - This refers to the ability of an individual PUF to reproduce a consistent response to the same challenge, regardless of external influencing factors. Reproducibility is hugely important for PUF designs as the response to a given challenge must match what is stored in the CRP or the verification of the PUF will fail. Reproducibility is generally quantified by the Intra hamming distance of the PUF, which will be covered in more detail later in this section.

Uniqueness – This refers to the uniqueness of PUFs that are made of the same design. This uniqueness is a result of the physical stochastic property that is utilised to generate the CRP pairs of each PUF. Each PUF should have a set of CRPs that are different to every other PUF, making it unique. This is typically quantified by the Inter hamming distance between different PUFs of the same design, which is covered in greater detail later in this section.

Identifiability – If a PUF has both reproducibility and uniqueness then it is considered to have good identifiability. Identifiability can be expressed as the distance between the Intra and Inter hamming distances of a PUF design. The greater the distance between these parameters the better the identifiability of the PUF. The practical implications of a PUF with good identifiability is a PUF design where each individual PUF can consistently generate responses to a given challenge that match the originally generated CRPs while different PUFs of the same design have completely different CRP pairs.

Physical Unclonability – For a PUF it is desirable that it cannot be copied to create another identical PUF with the same set of CRPs, this is called Unclonability. Physical clonability refers to a potential attacker creating a physical copy of the PUF, as opposed to a mathematical model. Ideally, a PUF should be so physically unclonable that even the original manufacturer could not create an identical copy of a PUF. This means that not even the manufacturer needs to be a trusted party, further increasing the security of the PUF. Combining unclonability with constructability makes it easy to create a PUF with random CRPs, but hard to create a PUF with specific CRPs.

Unpredictability – It is essential for a PUF to be unpredictable. Specifically, for PUFs, this means that there should be no correlation between different responses that would allow a potential attacker to guess future responses, having gathered a subset of previous responses. This is true for both multiple challenges and responses to a single PUF and across multiple different PUFs of the same design. Inter hamming distance can also be used to interpret the unpredictability of a PUF but rather than just looking at the distance between one CRP across multiple PUF instances, as was done for measuring uniqueness, the inter hamming distance here includes the distance between different CRPs on each individual PUF. Ideally, each CRP is completely independent of any other. However, this is generally not possible for every PUF. For PUFs where the independence of each CRP is not provable, the best practice is to prove that it cannot be predicted by the best-known type of attack. In this case the unpredictability of the PUF can be expressed in terms of how accurately the attacker could predict the future CRPs of the PUF. The specific types of attack used to predict PUF outputs will be discussed in greater detail later in this section.

Mathematical Unclonability – Mathematical Unclonability differs from physical Unclonability. For mathematical Unclonability, one assumes that the attacker has total access the PUF system and can gather as many CRPs as they like or even gain some insight into the entropy source. With this extended access the attacker can build a mathematical model of the whole PUF or create a table of all possible CRPs. For a PUF to be mathematically unclonable, this must be unfeasibly difficult to do through either having an unmanageably large number of CRPs or having an entropy source too complicated to be modelled. Hence, mathematical Unclonability is an extension to unpredictability where the attacker has total access to the PUF. If a PUF is mathematically unclonable, it must be unpredictable.

True Unclonability – Unclonability has been discussed so far from two different perspectives. Physical Unclonability refers to an attacker's ability to create a physical copy of the PUF that has the same set of CRPs and mathematical Unclonability refers to an attacker's ability to predict future CRPs by either modelling the source or building a table of all possible CRPs. For a PUF to be truly unclonable it must fit both definitions of Unclonability simultaneously.

One-Wayness – One Wayness for PUFs is similar to the original definition given for physical one-way functions proposed by Pappu [64]. This definition states that there should be no efficient way to find a challenge that will give a specified response. This is very close to the classical definition of a cryptographic one-way function. For PUFs, however, an extra consideration must be made. Classically, the definition only refers to the difficulty of inverting a single specified response to find the corresponding challenge. For PUFs this can be extended to include the total number of possible CRPs. The larger this number is, the more responses a potential attacker would have to invert to have a statistically significant chance of knowing enough possible CRPs to have pre-calculated a CRP that will be used for identification.

Tamper Evidence – As PUFs are used as a security device, there is always an increased chance that someone may attempt to tamper with them, to modify its operation in an unauthorised or harmful manner. Direct tampering is used to attempt to compromise security systems, this may be bypassing protection circuits or extracting important information about the internal workings of the system that could be used to compromise security. With this in mind, it is important that a security critical system, such as a PUF, should have some kind of mitigation of these risks. This usually involves the detection of such tampering and then performing an appropriate response to reduce security risks. In order to detect tampering, the system must have tamper evidence. Having temper evidence means that any tampering will have an unavoidable and detectable change on the system. For PUFs an example of tamper evidence would be a significant change in the challenge-response behaviour generated if any tampering has occurred. In the ideal case, a PUF would change its challenge-response behaviour so much from tampering that it would become undiscernible from being completely replaced with a different PUF instance. A PUF with this level of tamper evidence will have a natural protection from malicious tampering as the change in challenge and response behaviour would make the CRPs stored by the authenticating party no longer match those generated by the PUF, thereby locking it out of the system.

Maiti et al. [66] introduces a further two key concepts that will be used in this work to evaluate the performance of a PUF. The first of these concepts is that of uniformity. Maiti et al. define uniformity as an estimate of the proportion of ‘0’ to ‘1’ responses from the response bits of a PUF. If a PUF shows ideal uniformity, then the proportion of ‘0’ responses to ‘1’ responses should be 50% as this shows there is no bias in the responses to one result or the other. To calculate uniformity the Hamming Weight (HW) of an n-bit identifier response is calculated. The second metric proposed by Maiti et al. that will be used in this work is that of bit-aliasing. Bit-aliasing refers to the likelihood of different PUF units producing nearly identical responses. To estimate bit-aliasing Maiti et al. proposes using the Hamming Weight of the l-th bit of the response across a number(k) of PUF devices. The ideal value for the HW across response bits for bit-aliasing is again 50% as this indicates that there is no correlation between PUFs that could cause near identical responses to be generated.

2.2.1 Intra Hamming Distance

The hamming distance used for PUF evaluation refers to the proportion of bits that are different between two responses. In this work fractional hamming distance will be used when referring to hamming distance. For example, in two responses that are each 10 bits long, 8 bits are the same across the two responses and two are different then the fractional hamming distance would be 0.2. As reproducibility is a key characteristic for PUFs it is important to find a clear way to evaluate this parameter. The most common way of doing this is by evaluating the hamming distance between multiple responses given by a specific PUF instance to a given challenge. This is referred to as the intra hamming distance. For a PUF instance with perfect reproducibility we would expect the response to a given challenge to be the same every time which would result in an intra hamming distance of 0. If the response from the PUF instance to the given challenge changes at all this shows a reduction in reproducibility which will be represented as in increase in the intra hamming distance. For the intra hamming distance to give the most accurate representation of the reproducibility of the PUF each CRP selected for testing should be tested as many times as is feasible and under different operating conditions. To help in visualising the distribution of intra hamming distances for a PUF instance, a histogram is usually used, and one example is shown in Figure 15. Interpreting Figure 15, it can be seen that 65% of PUF responses have 0 bits in error, ~30% of responses have 2% of bits in error and ~4% of responses have 4% of bits in error. Other useful information can be gained by calculating the mean intra hamming distance and its standard deviation, which can then be compared to other designs or application specific requirements.

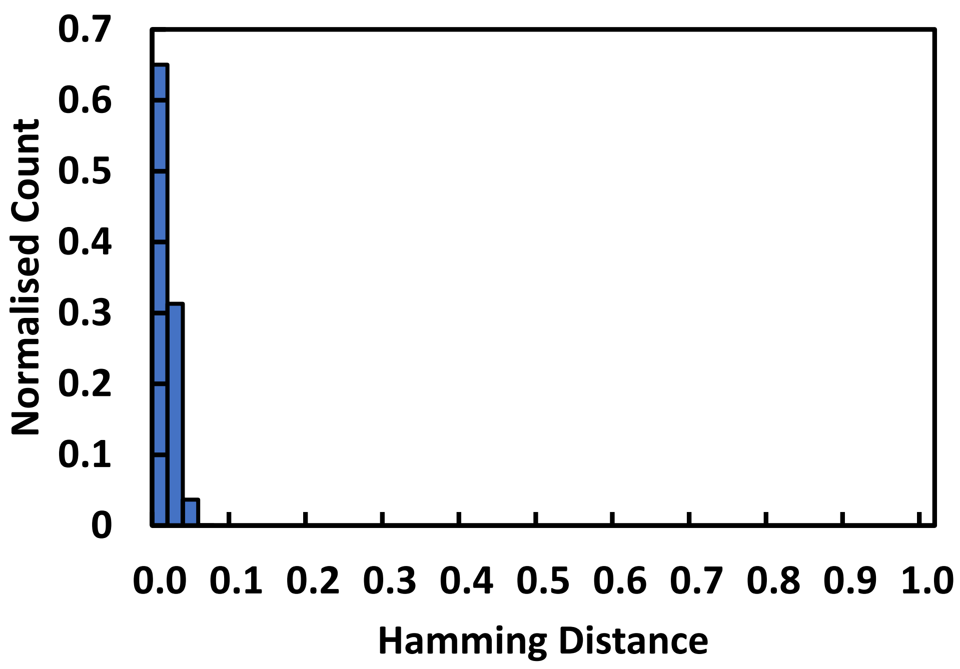


Figure 15, Example of histogram showing intra hamming distance of a PUF instance.

2.2.2 Inter Hamming Distance

Uniqueness is a key characteristic for a PUF to have and inter hamming distance can be used to evaluate it. The inter hamming distance is calculated by finding the difference between two responses, as is done when finding the intra hamming distance. However, to evaluate uniqueness we compare the responses from different PUF instances to the same challenge. This is known as the inter hamming distance. Inter hamming distance can also be evaluated using a histogram, and example of this is shown in Figure 16. To understand the histogram, we must consider what ideal uniqueness for a PUF would manifest in terms of hamming distance. If each PUF was completely unique then each bit of each response would have no correlation with the same bit of any other PUF instance's response. There would be a 50% chance of any selected bit being the same as the corresponding bit from another PUF instance, therefore. If each bit in the sampled responses has a 50% chance of matching the same bit in the other responses, then the ideal inter hamming distance for a PUF is 0.5. Referring now to Figure 16, we can see that that distribution of the inter hamming distance is around a mean of 50%

which is the ideal value. There are, however, some variations from this ideal value, as some PUF responses are not 50% the same as the corresponding response from other PUF instances of the same design. This variation can be quantified by calculating the standard deviation of the hamming distances. In the case of Figure 16, the standard deviation is 0.027. The deviation of the mean away from the ideal value of 50% and the size of the standard deviation for a given PUF can be used to evaluate whether a specific PUF design is suitable for a specific application, in combination with other limiting factors such as size, power consumption, and speed.

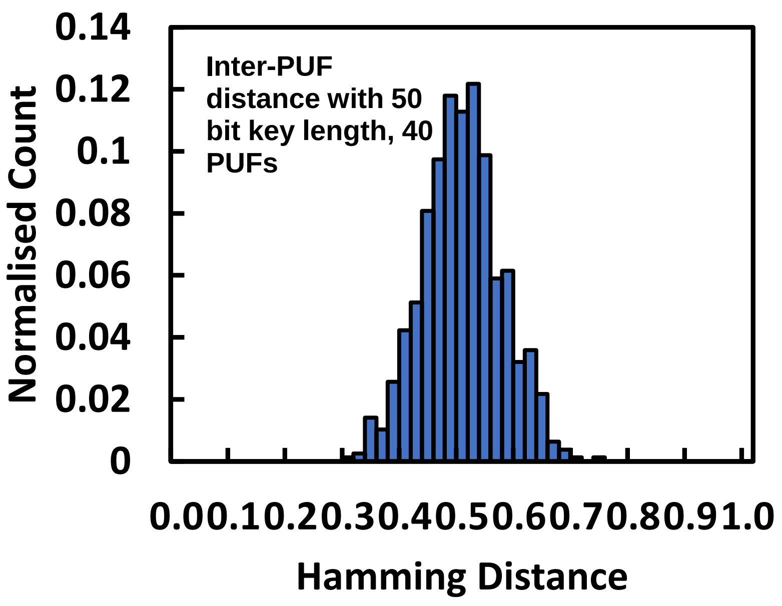


Figure 16, Example of histogram showing inter hamming distance of a PUF instance.

2.2.3 Electric and Non-Electric PUFs

One of the most obvious and important distinctions between PUF designs is whether the nature of their PUF behaviour is based upon inherently electric or non-electric principles. Non-electric PUFs are PUFs whose operation is based upon any physical phenomena that is not electrical in nature. This, however, only refers to the PUF behaviour and not how responses are processed or stored. Some examples of non-electric PUFs include the original PUF concept design by Pappu [67] which is based upon an optical principle of observing the light that passes through a substrate that has a unique pattern deposited onto it. Another example would be the paper-based PUF concept, where the surface of a sheet a paper is used as the source of entropy [68]. The surface of the paper is measured optically or texturally to create a unique “fingerprint” for each piece.

Electric PUFs on the other hand have their basic operating principles built around the observation of some electronic principle, such as resistance, capacitance, or some other electrical phenomenon. An example of this would be the metal coating-based PUF proposed by Tuyls et al. [69], where metal is specially coated to make the capacitance of the metal sheet have a large variance across it. The capacitance is then measured in multiple different locations across the sheet, which is used to create the unique “fingerprint” of each PUF. Within the electric PUF category there is also a large subcategory of silicon based PUFs that includes many of the more prolific and recent designs. Silicon based PUF are based upon electrical principles but also must be capable of being an integrated circuit so that they can be embedded onto silicon chips. The first example of this was proposed by Gassend et al. [65] and was based upon the difference in delay time between supposedly identical self-oscillating circuits. As the focus of this work is on security primitives that can be used for IoT applications, it is vital that they can be compact and used in integrated circuits. Because of this, the focus will be on silicon based PUFs.

2.2.4 Intrinsic and Non-Intrinsic PUFs

Another key distinction between different types of PUF design is if they are intrinsic or non-intrinsic PUFs. The idea of an intrinsic PUF was first proposed by Guajardo et al. [70] and then further refined by Maes et al. [64]. There are two key conditions for a design to be defined as an intrinsic PUF. The first key property is that it must perform all its evaluations internally. Internal evaluation means that the PUF must perform all measurements and data processing required to generate a response within the PUF structure itself. This gives a couple of advantages over external evaluation. The first is the practicality. If a PUF has internal evaluation, then it can work without having to consider any external limitations or restrictions while also offering greater potential reliability as there is less opportunity for external influences. The second advantage of internal evaluation is that it increases security potential. If the evaluation is performed purely internally and no response is openly broadcast to the outside world, the response can be considered as an internal secret. This internal secrecy makes any snooping harder to perform for a potential attacker. The embedded system is also harder to tamper with than external evaluation systems.

The most obvious disadvantage of internal evaluations is that it is harder to verify the operation of internal measurements as compared to external measurement. The worst possibility is internal measurements not working as intended and adding some unwanted correlations to the responses that could compromise security. However, it is more likely that if internal measurement is not working as expected the PUF will simply not function and will need to be replaced as a complete unit.

The second key characteristic that a PUF must have to be considered intrinsic is that the random variations extracted must be from an implicit source rather than an explicit one. An implicit random variations source comes as an artefact of uncontrollable variations in the manufacturing procedure that arise naturally. Alternatively, explicit random variations are added intentionally to be evaluated and used specifically as a PUF source. Implicit PUFs have some advantages over explicit PUFs. The first of these is that implicit variations occur naturally without any further effort during the manufacturing process. The removal of having to add a manufacturing step to create random variations can result in better efficiency in terms of both time and cost when bulk manufacturing is used. The second key advantage of implicit random variations is a potential increase of security potential. As these random variations are usually an undesirable side effect that reduces the yield, manufacturers spend a lot of time and effort trying to remove or control them. Despite these great efforts the random variations are still present, proving the difficulty any potential attacker would have to go through to try and impact the randomness of the PUF source.

Due to the potential benefits, particularly the self-contained and improved efficiency of manufacture, intrinsic PUFs are more attractive when considering the context of low-power IoT applications. The self-contained nature makes it far more feasible for use in applications such as IoT nodes or even potentially smaller applications such as RFID chips. These types of devices also tend to be low cost and high-volume devices, so that a reduction in manufacturing time and costs will be a huge advantage. Due to these considerations implicit PUFs will be the focus of this project and all the PUF designs examined in more detail will be in this category.

2.2.5 Weak and Strong PUFs

The final major classification is weak and strong PUFs. These classifications were first coined by Guajardo et al. [70] and further worked on by Ruhrmair et al. [71]. The strength of the PUF refers to the security characteristics of the challenge and response behaviour. For a PUF to be considered to be "strong", it must be able to continue to have a high probability of the next CRP being selected to be unknown to a potential attacker, even if they have a prolonged period to gather past CRPs. For this to be true Maes et al. [64] suggest that, at least, the following two conditions must be true. Firstly, the CRP set of the PUF must be extremely large (ideally infinite) to stop a potential attacker from simply querying all possible CRPs so there is none unknown left. Secondly, it should be infeasible for a potential attacker to build an accurate model of the PUF by observing previous CRPs, which was previously defined as the unpredictability. Any PUF that does not meet these specific requirements is considered to be a "weak" PUF. It is clear that the security potential of strong PUFs is greater than that of weak PUFs, thereby increasing their potential applications and impact. However, the practicality of building a strong PUF has proved to be very difficult as it is difficult to find a balance of design size, complexity, and number of CRPs. It is also difficult to prove the infeasibility of an attacker being able to simply learn all of the possible CRPs. Due to these difficulties, it can still be considered an open question as to whether a truly strong PUF is possible. It may be better to consider the strength of a PUF as a sliding scale that should be compared against a given application's specific requirements.

2.2.6 Existing Intrinsic PUF Designs

In this section the existing intrinsic PUF designs are reviewed. Currently all of these designs are electric and silicon-based designs. The random variation source in all of these designs comes from process variations introduced during the manufacturing of devices within the PUF. The basic working principles of these existing PUFs fit into 3 major categories. The first category is the PUFs whose working principles are based upon delays within digital circuitry where the time of specific delays derives from random process variation. These PUFs are labelled as delay based PUFs. The second major PUF category is the PUFs whose working principles are based on memory elements. These memory based PUFs use a random variation known as device mismatch which appears in bi-stable memory and affects the write time and balance between components. The final major group of intrinsic PUF designs is the PUFs whose random variation is extracted from mixed signal analysis. In these mixed signal PUFs some embedded analogue measurement is performed and then quantised through analogue-to-digital conversion to produce a response.

Arbiter PUFs

Arbiter PUFs are a delay-based silicon PUF whose basic principles are to introduce a race condition between signals travelling along two different paths through an IC circuit and was first proposed by Lee at al. [72]. At the end of the race an arbiter circuit is used to determine which path propagated the input signal faster. In these arbiter designs the race paths should be designed so that the delay times should be almost identical so that the fastest path is determined by random process variation and cannot be predicted simply by observing or modelling the design layout. However, due to the short time difference between the delay of the two paths there is a realistic chance that both signals arrive at the arbiter at a close enough time to cause the arbiter itself to enter a metastable state. If the arbiter enters a metastable state, then the eventual output will be dictated by the internal components of the arbiter and from the race condition. The metastability resolution is one of the major causes of error and unreliability in arbiter-based designs.

In the Lee et al. arbiter PUF [72], the race condition is created using a series of connected switch units that either allow the input signals to propagate along the one path or switch which path the two signals are propagating along. The challenge bits are used to determine which switch blocks will switch the signal path and which will not and consist of two cross coupled 2-to-1 multiplexers. At the end of the switch paths an SR latch is used as the arbiter circuit to determine which signal propagated faster, thereby giving a single response bit. This system is illustrated in Figure 17. In this design each switch block must be controlled by an individual challenge bit.

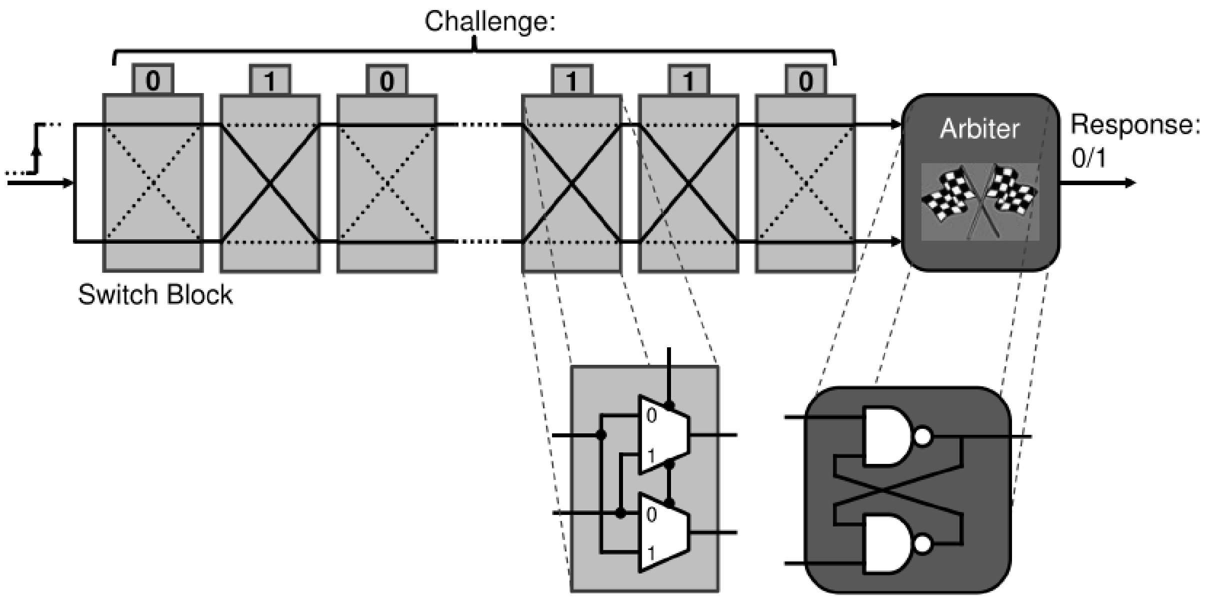


Figure 17, Working Principles of the Lee arbiter PUF[72] *as illustrated by Maes et al.*[64].

The arbiter-based design of PUF suffers from several key difficulties that have limited their usefulness and it is still yet to be seen if it is possible for them to be solved. The first fundamental problem is that each response is not independent of the others. This is due to the nature of the switch blocks where each one has a fixed underlying delay parameter which can be found and modelled by an attacker if they observe even a relatively small subset of past challenge and response pairs. This type of modelling attack is very powerful, especially against delay based PUFs, and will be discussed in greater detail later in this section. Another difficulty for arbiter PUFs is that of tuning and selecting the components of both the switches and the arbiter circuit itself. For the switches, the design should be that each path should nominally be perfectly symmetrical so that the variance in propagation time comes purely from device to device variation. If this is not the case, then the output of the PUF will have some bias which will cause different PUF instances to be more similar to each other resulting in poor uniqueness and identifiability. For the arbiter circuit, it should be balanced so that there is no bias towards one input or the other. If this is not the case the system will suffer the same deficiencies as with unbalanced switching components. These limitations are the reason why early arbiter-based designs such as the ones tested by Gassend et al. [73] have poor Inter hamming results of 1.05% indicating different PUF instances are not unique. Using more balanced components, Gassend's team were able to achieve a PUF with an intra hamming distance of 0.7% and inter hamming distance of 23% under ideal conditions. Although this PUF now shows high reliability and somewhat improved uniqueness Lim et al.[74] showed that it was possible to perform an accurate modelling attack after observing only 5,000 CRPs.

In an attempt to prevent these modelling attacks Lee at al.[72] proposed the feed forward PUF which introduces extra arbiters that are inserted along the race paths and are used to feed forward their responses to following switches in order to replace some 'known' challenge bits with unknown internal responses. This feed forward technique attempted to introduce non-linearity to the responses to make modelling more difficult. However, it also introduces more noise due to the metastability of some of the arbiters which increase the intra hamming distance to 9.84%, making the PUF fairly unreliable. Further, Rührmair et al.[75] showed that using more advanced modelling techniques could still build accurate models of feed forward arbiter PUFs, although the complexity of this task was correlated with the delay path length and number of feed forward arbiters. The need to have large, complicated designs for arbiter PUFs to be resistive to modelling attacks, as well as the far from ideal hamming distances achieved in these designs, have made them unsuitable for widespread adoption.

More recently arbiter based PUFs utilising emerging technologies, such as the design proposed by Govindaraj et al.[76], which uses the write time of resistive RAM in place of the traditional switch blocks, can offer some benefits in terms of design efficiency but there are still big questions to be answered in terms of their resistance to modelling attacks. Designs that use emerging technology also have a more difficult time trying to enter mainstream usage due to unreliability of the devices as well as cost of manufacturing when compared to conventional CMOS technologies. This is a particularly large obstacle when the focus of this project is on low cost, IoT applications.

Ring Oscillator PUFs

Ring oscillator PUFs are another major group of PUFs that have been prevalent in research. The first ring oscillator PUF was proposed by Gassend et al.[73] and utilised the same switch block based system as his arbiter PUF. However, negative feedback was introduced to induce oscillation in the system. After constructing several oscillators of identical design, the difference between the oscillation frequencies can be used as a PUF random variation source. One problem Gassend greatly improved upon is the external influence on PUF results. Gassend proposed using a differential function when comparing different oscillators to reduce external interference. If all the PUF’s oscillators are on the same chip in close proximity to each other, then the external influences such as temperature will be close to consistent across all of them. If the output oscillation frequencies are compared as a ratio between two compared oscillators, the effects of external factors are suppressed and the majority of observed variations will be from implicit device variation, thereby increasing PUF reliability. Although Gassend's design offers some good ideas for improving reliability, this ring oscillator design is vulnerable to the same modelling-based attacks as his arbiter design due to the switch block system. For a more robust ring oscillator PUF, the design proposed by Suh and Devadas[77] will be reviewed. Figure 18 shows a block diagram depicting the working principle of the ring oscillator PUF design proposed by Suh and Devadas. A large number of identically designed oscillators are connected to a pair of counters which in turn are connected to a comparison circuit. In this system the challenge is used to select which pairs of oscillators should be compared to find which is operating at a higher frequency. Each tested pair of oscillators will give a single response bit. As each oscillator is identically designed the primary source of variation between frequencies should be device-to-device variation introduced implicitly during manufacturing.

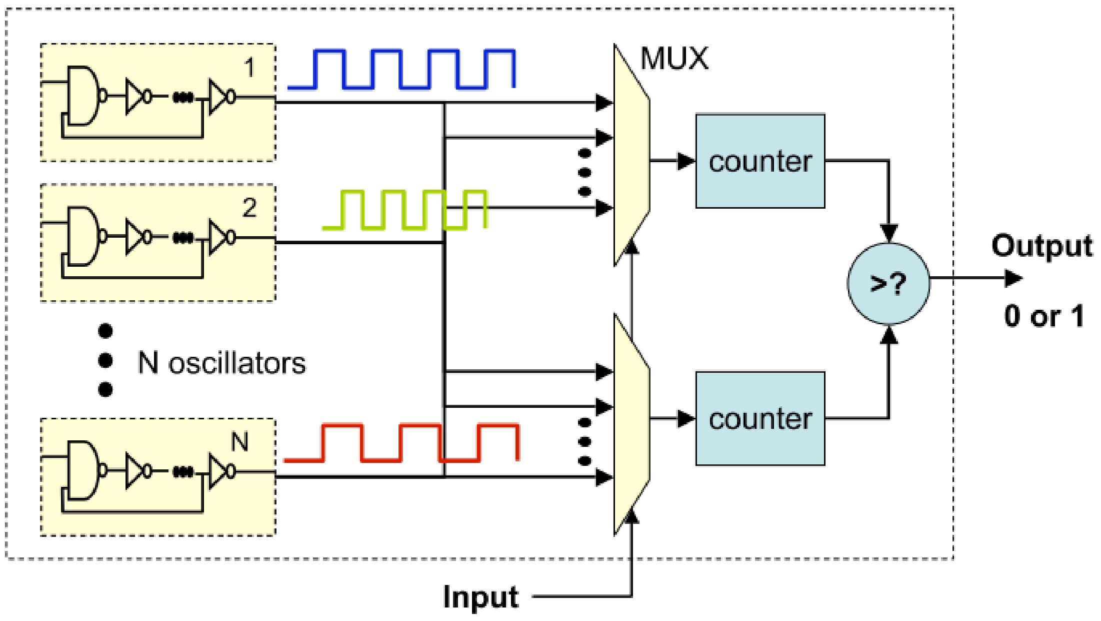


Figure 18, Diagram of Ring Oscillator PUF as proposed by Suh and Devadas [77].

To avoid dependence between the output responses in this design some pre-selection of comparable oscillators has to be performed. In Suh and Devadas' original proposal they group the oscillators into groups of 'k' size and find the pair of oscillators with the greatest frequency difference, then pre-select these as the pair to be compared. This gives two clear benefits. Firstly, it removes any dependence between responses, and secondly makes the PUF more reliable as external influencing factors will have to be large to 'flip' the PUF response, as the largest possible difference was pre-selected. Using this technique Suh and Devadas were able to achieve an intra hamming distance of 0.48% and an inter hamming distance of 46.15%, which shows a clear improvement over the arbiter PUF results. The major drawback to this method is that it is inefficient in terms of design complexity and footprint as the number of oscillators (n) has to be very large to give a reasonable number of CRPs, as the maximum number of CRPs is given by n/k. The need for pre-calculating the pairs of oscillators also adds a great deal of effort when manufacturing these PUFs, as each device must be independently measured and then a sorting system used to match the best possible pairs.

Several improvements on the ring oscillator PUF design have been proposed. Two of the most important ones are suggested firstly by Yin and Qu [78] and then by Maiti and Schaumont [79]. The first key improvement suggested by Yin and Qu is to the oscillator pairing scheme, where they ignore far fewer oscillators but maintain a similar reliability. To achieve this, they make use of a new pairing algorithm which can more efficiently group oscillator pairs. The second key improvement Yin and Qu proposed is a new architecture for arranging the oscillators and frequency counters so that multiple oscillators can share a single counter. The proposed architecture can better find an optimal value for response speed and resource demand of the PUF. Maiti and Schaumont also proposed two key improvements to the ring oscillator PUF's design. The first of these is a set of guidelines that should be followed when implementing ring oscillator PUFs in order to minimise any systematic bias. They looked in detail at the effect of each guideline on the reliability and uniqueness of the implemented

PUF design. The second key improvement that they proposed is an alteration on the oscillator paring scheme used. In the improved pairing scheme Maiti and Schaumont replaced the idea of finding the most reliable pairs and ignoring the others and instead utilised the configurability of the oscillators and found the most reliable configuration for each pair of oscillators. By doing this they can achieve equal or improved reliability while also not wasting oscillators. These improvements greatly increase the efficiency of ring oscillator PUFs. However, they still require significant resources to implement the large number of oscillators required for a useful number of CRPs. The process of pairing oscillators pre-installation is also still a large hurdle to overcome when considering this type of PUF for use in low cost IoT applications.

Glitch PUF

Another prominent type of PUF that has been proposed is the Glitch PUF. These PUFs extract their random variation from the glitch effect that can occur in combinational logic circuits. Combinational circuits have no internal states, they only have a steady-state output which is determined by the input signals applied. However, due to internal delays within the circuit, caused by random process variation, the output of the circuit can be unstable for some time before settling at the steady-state value. The value observed on the circuit output during this time of instability is the glitches. Because the instability is caused by delays resulting from process variation, they are consistent for each circuit and are unique in each component. This reliable and unique glitch behaviour is what is utilised to create a PUF system. The first glitch PUF design was proposed by Anderson et al. [80], in this design Anderson proposes a circuit that either creates a single or no glitch at its output, dependent on the random variation-based delay within itself. A flip-flop is then used to sample the output to see if a glitch has occurred or not. The output of this flip-flop is used as the response bit. When many of these circuits are copied across a chip a PUF can be created where the challenge is which circuits to test and the response is the presence or lack of a glitch within each challenged circuit. Using this technique

Anderson et al. achieved a PUF with an inter hamming distance of 48.3% and an intra hamming distance of 3.6%.

The glitch PUF principle was expanded upon by Shimizu et al. [81], [82]. The main improvement of Shimizu's design is the replacement of the specific combinational circuit in Anderson's design with the output from any combinational circuit. Shimizu's more recently proposed design uses a toggle flip-flop on the output of any combinational circuit to sample how many glitches occur on that output before a steady state is achieved. The response behaviour is then extracted from finding if the number of glitches that occurred was odd or even. This concept is illustrated in Figure 19.

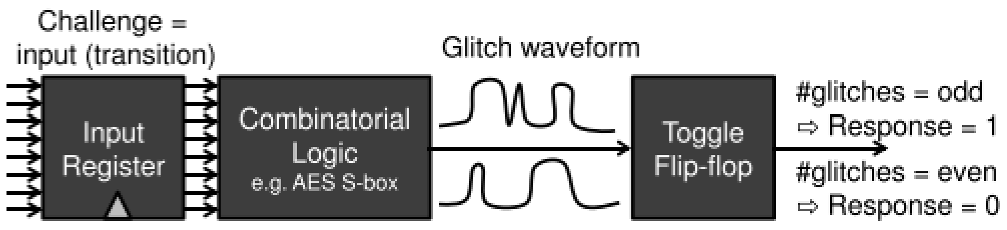


Figure 19, Glitch PUF as proposed by Shimizu et a[81]*l., illustrated by Maes et al*[64].

To improve the reliability of the proposed glitch PUF, Shimizu proposes a 'bit-masking' technique. In this technique, every PUF instance is measured before being used. The purpose of this measurement is to find which glitch responses are stable and which are not. The responses that are found to be stable are selected as actual responses for the PUF. The responses that are unstable are 'masked' which in this case simply means that they are ignored for normal PUF operation. The bits which are and are not used is information that must be distributed with the PUF and stored with the CRPs. This process of bit masking does result in some extra overhead in production and CRP storage but also greatly increases the reliability of the PUF. With bit masking Shimizu's PUF design can achieve an inter hamming distance of 38%, intra hamming of 1.3% with 38% of total bits being masked, while operating under ideal conditions. This design, however, is very sensitive to temperature and supply voltage variation with the intra hamming distance increasing to 15% at 85˚C and 5% supply voltage increase.

Glitch PUFs offer some complexity improvements over previous PUF designs in terms of circuit footprint. However, they require a lot of pre-installation tuning and adjustment to make them at least semi-reliable under nominal working conditions. Even with the potential reduced footprint, when compared to ring oscillator PUFs, a glitch PUF with a large number of CRPs requires a large number of combinational circuits to be implemented which may not be feasible for small scale applications. There has also not been any major work performed testing the resilience of glitch based PUFs to modelling or other types of attack.

SRAM Based PUFs

Static Random-Access Memory (SRAM) is a digital volatile memory technology built from bi-stable circuits. The typical CMOS implementation is built using cross coupled inverters, which is the same circuit used in the metastability based TRNGs, such as the design proposed by Sanu et al. [42], that were discussed earlier. SRAM PUFs make use of the same metastability resolution as the TRNGs using this technology. The key difference between the two applications is that the TRNG design wants the metastability resolution state to have a 50% probability of going either way, whereas the PUF design wants the metastability resolution state to be consistent so it can be used as a basis for PUF operation. The basic working principle of the design is that a set of ‘challenged’ SRAM units selected from a large array will be powered up into a state of metastability, the resolution of the metastable state for each selected SRAM circuit is used as the response for the PUF. As with the TRNGs using the same principle the SRAM units are designed to be identical and symmetrically balanced so that any bias in the resolution of the metastable state is a result of random device-to-device variation that occurs during fabrication. The first SRAM PUF designs were proposed by Guajardo et al.[70] and Holcomb et al.[83]

almost simultaneously. Guajardo's design was able to achieve an inter hamming distance of 49.97% and an intra hamming distance of 3.57% under ideal conditions and implanted on an FPGA. Holcomb's design was able to achieve similar results with a inter hamming distance of 43.16% and an intra hamming distance of 3.8% on 8 commercial SRAM chips and an inter hamming distance of 49.34% with an intra hamming distance of 6.5% on embedded memory in 3 microcontroller chips.

The main challenges for SRAM PUFs are tuning the SRAM circuits and reliability when using different technologies and when operating under different environmental conditions.

As the random variation source can be the same for both SRAM PUFs and metastability based TRNGs, Satpathy et al. [36] have proposed a combined design. In the combined design Satpathy uses the cross coupled inverts which have a strong bias to one stable state as a 'chip ID'. This chip ID is a PUF with only a single CRP, which is sometimes also known a physically obfuscated key. The cross coupled inverters that have the closest to a 50% probability of entering each possible stable state after resolving the metastability are used to create a TRNG following a design very similar to Sanu et al. [52] design, which was reviewed in detail earlier. This combined design offers key improvements in efficiency and security. Satpathy states that the combined design reduces the total area by 63% and the energy consumption by 20% as against implementing the same circuits individually. In this work the PUF part of the design is limited to a chip ID rather than a fully functional PUF which is one of the key areas where this idea could be built upon. The tuning required for this design is also extensive as the system has to find which cross coupled inverters should be used for each part of the design, as well as tuning the circuits, when selected to be biased appropriately to give a circuit with an acceptable reliability.

RTN PUF

After reviewing RTN-based TRNG design, the RTN-based PUF design is reviewed now. Chen et al. [84] proposed using the presence or lack of RTN in each device of an array of nFETs as the random variance source for a PUF design. To detect whether there is RTN in each device, a large voltage is applied to the gate to charge any traps within the device. After the charging, the gate voltage is dropped to a low value and the drain current is observed. With the low gate voltage any traps that were charged have a high chance to discharge causing a sudden rise in the drain current. If these current rises are detected, then the device is said to have RTN and the response will be '1'. If no rises are detected, then the device is said to have no RTN and the response will be '0'. A typical drain current response from a device showing RTN is shown in Figure 20. The challenge in this PUF design is selecting which devices within the array to test. Using this methodology Chen was able to achieve an inter hamming distance of ~42% and an intra hamming distance of ~5% under ideal conditions. One big advantage of this design is that only a single device is needed to obtain the random variation for each CRP rather than the complex circuits used in other designs, although the supply voltage and measurement circuits are still relatively complex. The main drawback of this design is the temperature dependence of RTN, which means some devices without RTN at one temperature in the given sampling time might show some RTN at another temperature, reducing reliability. Another disadvantage is that the drain current must be observed for a relatively long time (0.1s) before a response can be extracted for each device. To speed up the PUF behaviour multiple devices could be measured at once, however this would require more measurement circuits, increasing design area and power consumption.

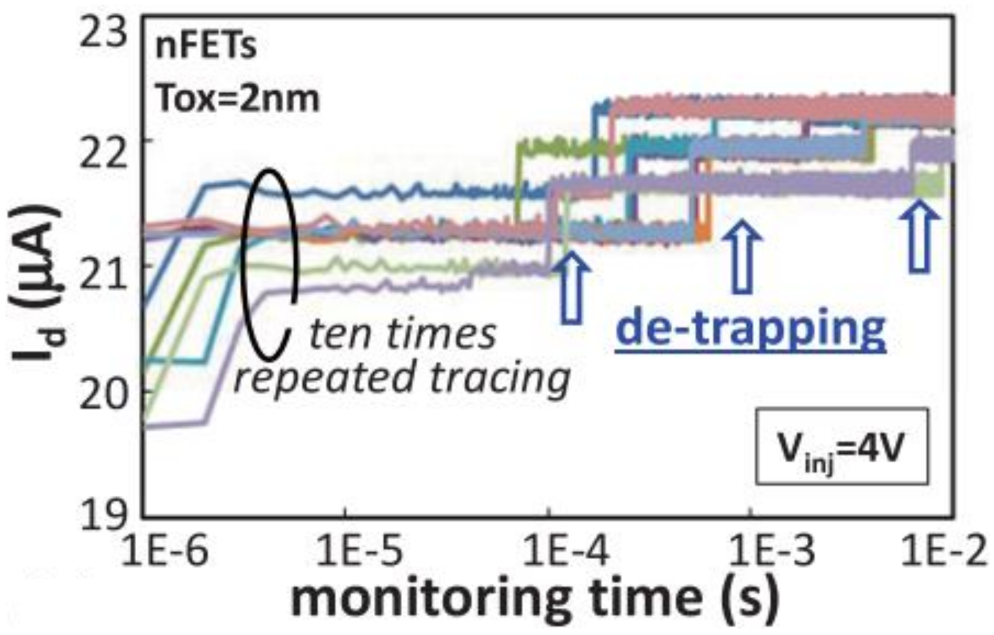


Figure 20, Typical drain current signal response for a device with RTN [84].

Other Notable PUF Designs

This subsection covers the basic principles and performances of some other notable PUF designs that are not as prolific as the previous ones covered in detail or are not as relevant to the research performed in this project.

Latch PUFs

Latch PUFs were first proposed by Su et al. [85] and work on a very similar principle to SRAM PUFs. Latch PUFs are based on the metastability resolution state of cross couples NOR-latches. The key difference between this PUF and SRAM PUFs is that Latch PUFs induce the metastability through resetting the NOR-latches while powered on rather than having to de-power and power up the SRAM circuits used in SRAM PUFs. This 'reset while powered' functionality gives the latch PUF some advantages over SRAM PUFs. Firstly, the response behaviour can be quickly re-invoked while the devices are still powered up. This means that the generated responses do not need to be stored in other memory blocks over the active time of the PUF and can be recreated at will. This fast re-creation time of responses allows the PUF to use more complex post-processing techniques, such as majority voting, to improve reliability. Using their proposed Latch PUF, Su was able to achieve a PUF with an inter hamming distance of 50.55% and an intra hamming distance of 3.04%. Latch PUFs still have many of the difficulties faced by SRAM PUFs, such as the difficulty of tuning the balance of the cross coupled devices which create the random variation source. Also, if more complex post-processing is needed to increase reliability to an acceptable level, the increase in design complexity, power consumption, and design footprint may not be suitable for many low power IoT applications.

Butterfly PUFs

The Butterfly PUF is an extension of the SRAM PUF principle and was proposed by Kumar et al. [86]. The Butterfly PUF was developed as a way to implement an SRAM PUF type onto general purpose FPGAs. This is required as the traditional SRAM PUF cannot be implemented onto most FPGAs as they have in-built systems that clear the values stored in the SRAM units after being powered up. As the SRAM units are automatically and unavoidably cleared after start-up, the metastability resolution state required for SRAM PUF operation is lost. The Butterfly PUF avoids this problem by using creating cross coupled transparent data latches rather than SRAM units. These transparent data latches create a bistable circuit that can be forced into a state of metastability through the use of preset / clear functionality. Using this methodology Kumar was able to achieve a PUF with an inter hamming distance of ~50% and an intra hamming distance of ~5% under high temperate conditions. As this design is made specifically for implementation onto FPGAs it may not be applicable for application onto many IoT devices such as remote sensors. The Butterfly PUF principle also has the difficulty of creating well matched random variation sources, as the discrete routing of most FPGAs can make it difficult to create balanced designs.

Buskeeper PUFs

Buskeeper PUFs are another PUF design that makes use of a bistable memory element and was first proposed by Simons et al.[87]. The Buskeeper PUF is based upon Buskeeper memory cells, which are components connected to bus lines within embedded systems. Buskeeper memory cells is a cross coupled inverter-based latch system with a weak drive-strength. The basic operating principles past this point are the same as with SRAM PUFs. Using this design Simons was able to achieve a PUF with an inter hamming distance of 49.9% and an intra hamming distance of 7% under ideal conditions. The main advantage of Buskeeper PUFs over SRAM PUFs is that the Buskeeper memory units are slightly more efficient in terms of design footprint when compared to SRAM units; other than this all of the challenges present for SRAM PUFs are still present for Buskeeper-based designs.

Table 2, Key points of PUF Designs gives a quick reference to compare the key statistics of the some of the major electronic PUF designs that have been proposed previously.

Table 2, Key points of PUF Designs

Authors	Entropy Source	Mean Intra Hamming (%)	Mean Inter Hamming (%)
Lee et al. [72]	Arbiter	0.7	23
Suh et al. [77]	Ring Oscillator	0.48	46.15
Maiti et al. [79]	Ring Oscillator	Not provided	47.31
Anderson et al. [80]	Glitch	3.6	48.3
Shimizu et al. [81], [82]	Glitch	1.3	38
Sanu et al. [42][88]	Metastability SRAM	2.8	50

Guajardo et al.[70]	Metastability SRAM	3.57	49.97
Holcomb et al.[83]	Metastability SRAM	3.8	43.16
Chen et al. [84]	RTN	~5	~42
Su et al. [85]	SRAM Latch	3.04	50.55
Kumar et al. [86]	SRAM Butterfly	~5	~50
Simons et al.[87]	Buskeeper	7	49.9

2.2.7 Variations on the PUF Principle

This section looks at ideas that are variations of the core operation of the PUF principle. These concepts take the basic CRP-based approach of PUFs but add extra elements to it to try and make the PUF principle more secure, more practical, or both.

Controlled PUFs (CPUFs)

Controlled PUFs are a variation of the PUF principle, where a normal PUF system is taken but it is only accessible through another cryptographic primitive. In an ideal case this link should be made so that if the systems do become separated it should destroy the functionality of the PUF. This idea was first proposed by Gassend et al. [89] and he outlined four examples of how this idea could be useful. The first proposed idea was to use a hash function to generate the challenges sent to the PUF in order to stop chosen challenge attacks. It has since been shown, however, that model building attacks work equally well for random challenges or chosen challenges. The second proposed way of creating a CPUF was to add error correction algorithms to the output of the PUF to increase reliability. This is a technique that many subsequent PUFs have adopted, although most have not defined themselves as CPUFs. The third suggested implementation of a CPUF is to add a hash function to the error corrected outputs of the previous CPUF suggestion. Adding a hash function breaks any connection between the observed responses and the underlying PUF behaviour, making model building attacks much more difficult. However, it is very important that the error corrected PUF output has zero error, as the hash function will create a completely different response even if a single bit is in error, causing a CRP mismatch. The final suggestion for CPUFs is to use a hash function with additional inputs so that a single PUF can have multiple 'personalities' so that each user connected to the PUF can have their own unique set of CRPs. This may be desirable where multiple parties can be connected to the PUF and not all parties trust each other.

Reconfigurable PUFs (RPUFs)

Reconfigurable PUFs (RPUFs) are an extension of the typical PUF concept that was first proposed by Kursawe et al. [90]. In Kursawe's proposal, RPUFs are defined as a PUF whose CRP behaviour can be reconfigured. This reconfiguring can be a partial or complete change of the CRP behaviour, which is random and preferably irreversible, resulting in a new PUF behaviour. Kursawe proposed two ways in which this type of behaviour for a PUF could be achieved. The first is an optical PUF where the optical medium used to give the PUF behaviour can be melted by a laser, thereby changing the scatter pattern and the PUF identity. The second possible implementation is to use phase change memory. With this phase change memory, the phase of each cell can be changed from crystalline to amorphous or some intermediary states. The resistance of the phase memory cell is then measured to give the PUF behaviour. Re-writing the phase array cells will change the resistance in a random way thereby reconfiguring the PUF identity. Kursawe did not test these implementations in detail so that its performance is not known.

Katzenbeisser et al. [91] proposes a specific type of RPUF which they coin as a logically reconfigurable PUF or LRPUF. Katzenbeisser defines the LRPUF as a typical PUF that has a state-update mechanism which can transform the input and output to change the CRP behaviour. The state-update mechanism consists of a non-volatile memory system that stores a state and state dependent input and output transformation. LRPUFs do have some drawbacks compared to the original RPUF as proposed by Kursawe. Firstly, the transformations that LRPUFs perform are not irreversible as each transformation possible is available within the state-update circuit, which potentially reduces security. The second drawback of LRPUFs over RPUFs is that the state-update circuit requires complex anti-tampering circuitry to ensure security of the system. With this being said, LRPUFs use well matured technology so that they can practically be implemented currently, whereas the technology used in the proposed RPUFs is not commonplace, making their implementation difficult.

Public PUFs (PPUFs)

PPUFs were first proposed by Rührmair et al. [92] where he outlines a new type of PUF scheme based on a public key structure. This public key system is based on the idea that Rührmair terms as SIMPL, which stands for "SIMulation Possible, but Laborious". In this system, a PUF is possible to model to find a CRP but this modelling takes much longer than it takes the actual PUF instance to generate the same CRP. Using this SIMPL behaviour, the challenging party can pre-compute a CRP using the public model of the PUF then issue the challenge. When the PUF instance receives the challenge, it generates the appropriate response and replies. When the challenging party receives the response, it checks to see how long the response took to generate. As only the specific PUF instance can generate the response quickly, the challenging party can determine whether the response was generated by the PUF instance or not. Rührmair's design was based on an optical PUF principle, however. The first silicon based PPUF was proposed by Beckmann et al. [93]. This design is based upon a PUF system built from XOR gates that have different delay times for signal propagation through each gate caused by manufacturing variation. This PUF can then be used in a PPUF scheme where the delays of each device are distributed as the public model and a challenge can be built from this model. However, these early PPUF designs had quite a few drawbacks. The first is the very precise time measurements that have to be performed. It is possible to measure time very accurately, but precise measurement requires complicated and expensive equipment that might not be available or appropriate for most applications. While keeping the time measurement accuracy requirements to a minimum, the PUF must also have speed, an advantage over attackers trying to clone the system. This advantage must be at least several seconds [29], making these PPUF systems significantly slower than traditional public key cryptographic protocols. These types of PPUF are also potentially vulnerable to mathematical cloning or machine learning attacks if the number of available CRPs is not sufficiently large.

Matched PUFs are a more recent innovation within the PPUF concept. First proposed by Meguerdichian et al. [94], this PUF concept is based upon the idea of creating two identical PUFs but making the creation of any more impossible. Meguerdichian suggests creating these two identical PUFs through simultaneous device aging through stress to the two PUF instances; after this stressing, any parts which cannot be matched across the two PUFs are removed through either gating or software mechanisms. When two identical PUFs have been created they can be distributed so that the authenticating party has one and the external party the other. The key advantage of this mechanism over the previous PPUF designs is that it removes the need for the authenticating party to simulate the PUF instance from a public model, which requires significant effort by design. Despite offering this potential improvement, matched PUFs have several unanswered questions. The first is how difficult it really is to create a 3rd copy of the PUF instance either through physical or mathematical cloning. Secondly, an authenticating server may have to be connected to a large number of external PUF instances. It may not be practical to have another full PUF instance within the server structure to match every external PUF. This is further exacerbated if more parties than the two need to be able to communicate using this PUF then creating more matched PUF becomes increasingly difficult. Finally, reliability of each PUF could be a problem for this design as any change in either PUF instance will result in them no longer being matched.

Quantised PUFs are the final type of PPUF to be reviewed here. First proposed by Meguerdichian et al. [95], quantised PUFs are an extension of the matched PUF, which solves the issue of matched PUF only being able to secure communication between two parties. Quantised PUFs are created by aging PUF instances separately and then matching elements later and de-activating all elements which do not match across the PUF that will need to communicate. To get a reasonable number of PUF components to match across multiple instances the responses from each element are quantised to a level of accuracy that makes finding matched elements acceptable likely. This PUF idea however does still have issues with reliability as well as potential weakness to model building attacks, either physically or mathematically.

Chapter 3 - Measurement and Characterisation Methodology

3.1 Introduction

This chapter presents the equipment, samples, measurement techniques, and characterisation methods used in this project. Firstly, the equipment required to perform the experiments is described. Next, the samples used for testing are listed and described. Thirdly, the details of each measurement technique are explained.

3.2 Equipment Used

This section details the equipment required to perform the experiments. Table 3 shows a summary of the equipment. The main piece of equipment used is the Keysight B1500 semiconductor analyser. It is equipped with 4 source measure units (SMU) and 2 waveform generator/fast measurement units (WGFMU), although only 2 SMU and 2 WGFMU units are used in this project. The B1500 was controlled by an external PC through a general-purpose interface bus (GPIB), with control programs written in Visual Basic. Devices were probed on a Cascade MPS150 probe station which was also fitted with a controllable temperature hot plate.

For the proof of concept and design testing circuits a Keysight 81150A pulse generator was used to provide gate voltage to the sample wafer which was probed on a Cascade Summit Probe station. A Keysight E3632A DC power supply was used to supply drain voltage. A Femto DHPCA-100 amplifier was used to amplify the drain current. An Agilent Infiniium MSO8104A oscilloscope was used to record the required signals. A Xilinx Spartan 3 FPGA was used as the basis for the proof of concept circuit development.

Table 3, Measurement Equipment Usage.

Test Equipment	Tests that require this equipment	Specific Requirements
Keysight B1500	Stepped IV	SMU
	DC RTN	SMU and WGFMU
	ACRTN	SMU and WGFMU
	HC Stress and Recover	SMU and WGFMU
	TDDS	SMU and WGFMU
Cascade MPS150	Stepped IV	
	DC RTN	
	ACRTN	
	HC Stress and Recover	High temperature
	TDDS	
Cascade Summit	Proof of concept circuit	
Keysight 81150A	Proof of concept circuit	
Keysight E3632A	Proof of concept circuit	
Femto DHPCA-100 amplifier	Proof of concept circuit	
Agilent Infiniium MSO8104A	Proof of concept circuit	
Xilinx Spartan 3 FPGA	Proof of concept circuit	

3.3 Samples Used in this Work

Several different types of devices were used in this work to verify the functionality of the designs across different processes. Table 4 summarises the key details of the samples used in this work. Each process was available in both nMOSFETs and pMOSFETs. The samples were sourced from either the Interuniversity Microelectronics Centre (IMEC) or CSR plc (formerly Cambridge Silicon Radio acquired by Qualcomm in 2015). The samples from IMEC were developed for the Industrial consortium of Logic devices based at IMEC. The CSR wafer is a commercial grade product and was manufactured by the Taiwan Semiconductor Manufacturing Company (TSMC) using their 28 nm technology node. The majority of the work was performed on the 22 nm FinFET process with only trap timing information being extracted from the other IMEC samples. The CSR wafer was exclusively used for testing with the proof of concept circuit for the TRNG design.

Table 4, Test Sample Information.

Sample Name	**HK45**	**HK28**	**FF**	**CSR**
Gate Material	Metal	Metal	Metal	Metal
Dielectric	HK Stack	HK Stack	HK Stack	HK Stack
EoT (nm)	1.45	1.3	1	1.2
Technology Node (nm)	45	28	22	28
Structure	Planar	Planar	Planar	Planar
Supplier	IMEC	IMEC	IMEC	CSR

3.4 Measurement Methodologies

This section explains and discusses the different types of measurements performed in this work. The aim is to create an understanding of how each measurement was taken so that the results shown in later sections can more easily be interpreted, as well as providing key information for anyone trying to replicate any results from this work.

3.4.1 Stepped IV Measurement – Drain and Gate Current Measurement Against Gate Voltage

Drain current against gate voltage (IV) measurement is one of the simplest and also most important measurements performed in this work. IV measurement has several different uses. It is used to have a quick check whether the probing and basic operation of a selected device are working as expected. It is also used to quickly see if there are any clear RTN traps present, as the characteristic RTN switching behaviour will be evident in the IV plot. Using IV measurement for quickly checking for RTN is more powerful than checking at individual gate voltage as most RTN traps will only be present over a specific range of gate voltages and the IV measurement sweeps a wide range of gate voltages. The final use of the IV measurement is to find threshold voltage shift (ΔVth). ΔVth is a key parameter when discussing device aging and RTN characterisation, both of them are widely discussed in this work. For device aging, an increase in the magnitude of threshold voltage (Vth) is a key measurement of the device lifetime [96] and for RTN characterisation the ΔVth caused by trapping and de-trapping is a key measurement.

In this work, a stepped IV measurement is made by using the SMU units of the Keysight B1500 semiconductor analyser. Figure 21 shows the test waveforms for the stepped IV measurement.

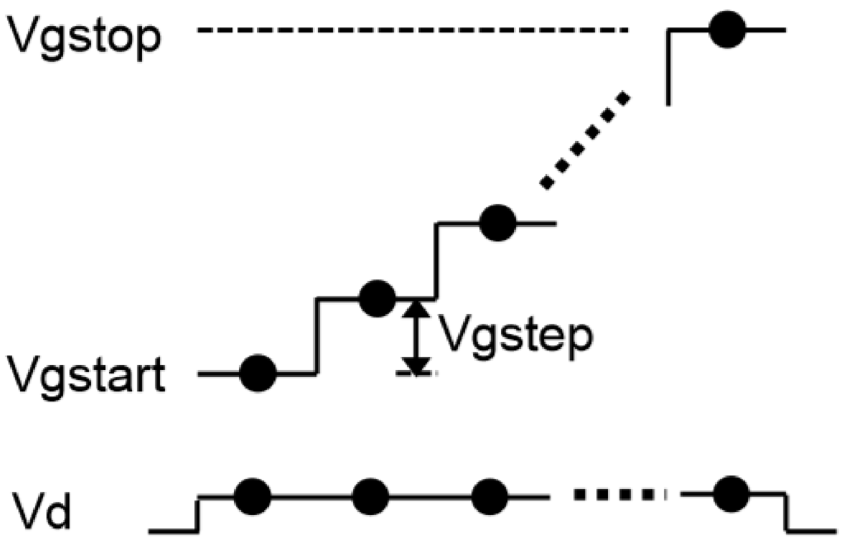


Figure 21, Test waveforms for a stepped IV measurement[97].

For the stepped IV measurement, one SMU is used to increase the gate voltage in steps, while another SMU supplies a constant drain voltage. At the level part of each voltage step the drain current is observed, also using the SMU. The time spent at each step as well as the size of each step are adjustable parameters. The typical time to measure the drain current at each step is from a few to a few hundred milliseconds and the step size is typically from a few hundred microvolts to a few millivolts. The typical time to complete an IV measurement is from 1 second to 20 seconds. The very high resolution of the SMU units allows for this IV measurement to be very accurate (sub-pA range) which is ideal for the experiments that are frequently operating around or below the threshold voltage. This technique also allows measuring the gate leakage current, which is significantly smaller than drain current under normal operation, using the same SMUs. Observing gate leakage can be important when looking at devices under a heavy stress to ensure that the device is not broken down. It also allows observing any RTN events visible in the gate current. Figure 22 shows a typical result from a stepped IV measurement on a fresh 22nm FinFET device which is functioning normally with a drain bias of Vd=0.1 V.

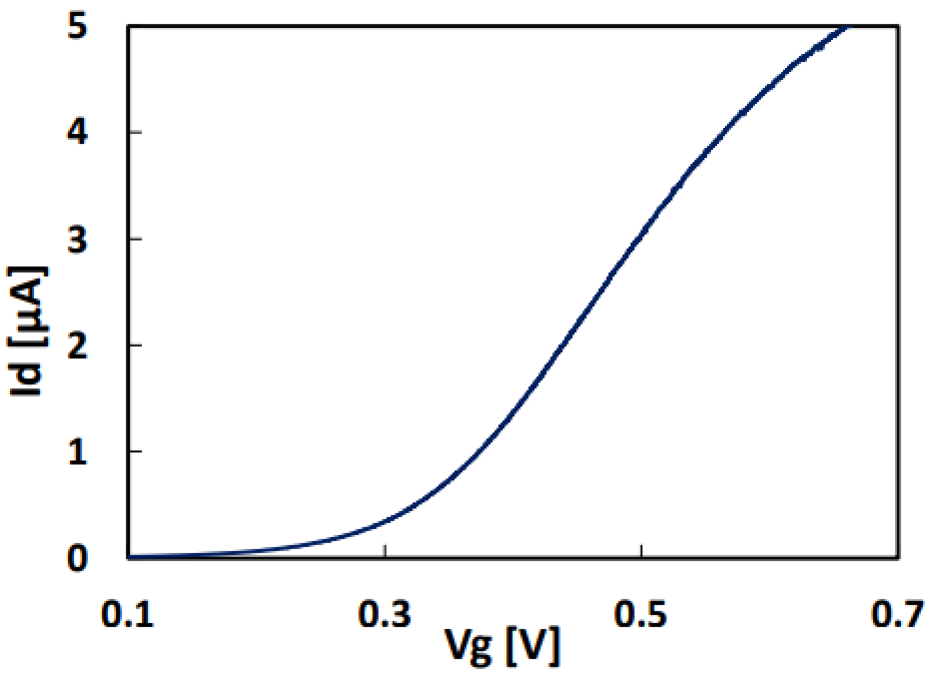


Figure 22, Typical stepped IV measurement result.

As mentioned, one of the main purposes for performing an IV measurement is to find the Vth of a device. The method for finding the Vth used in this work is the max-gm method which is illustrated in Figure 23. The first step is to find the transconductance (gm). Transconductance is the slope of the IV measurement so it can be calculated by finding the first derivative of the IV measurement curve (blue curve in Figure 23). The next step is to find the corresponding Vg value to the maximum gm (max-gm) value. Finally, a tangent line is drawn from the max-gm Vg value and its intersect with the Vg-axis is the Vth (blue square in Figure 23).

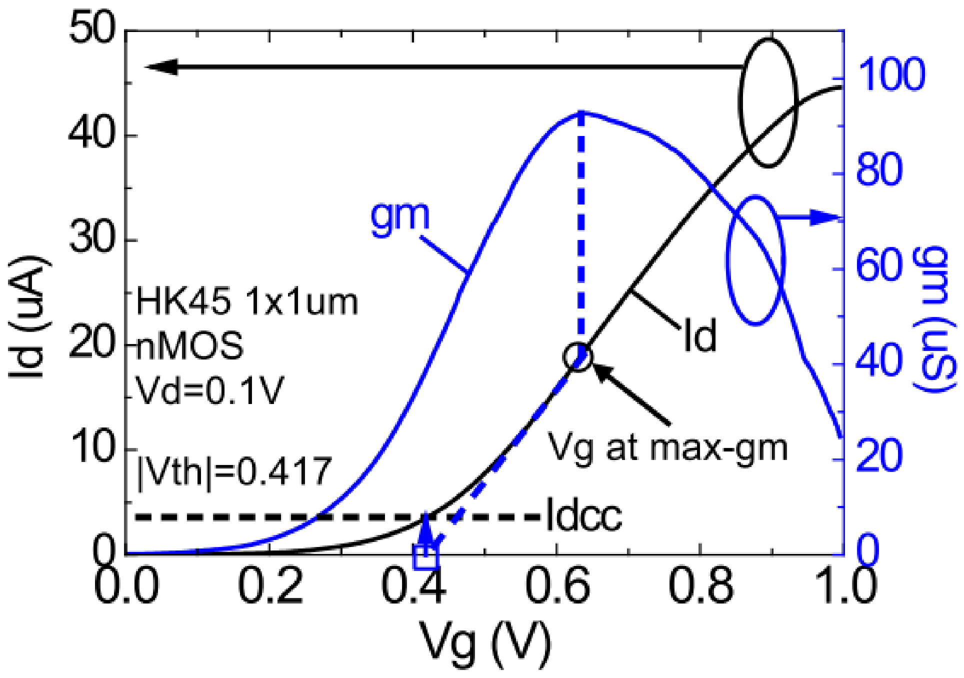


Figure 23, Illustration of max-gm method for finding Vth [97].

3.4.2 DC RTN Measurement

The basic principle of DC RTN measurement is to observe the drain current of a device under constant gate and drain voltage. For this work the first purpose is to detect visible RTN within the drain current. Measuring the drain current in this way allows extracting several key characteristics of RTN. Firstly, it gives the timing of the trapping and de-trapping, usually in terms of capture time (τc) and emission time (τe), which are the average time taken to observe a capture or emission event, respectively. Secondly, DC RTN measurement allows finding the magnitude or impact of a trap, either in terms of change in drain current (ΔId) or change in threshold voltage (ΔVth) if an IV measurement is also available.

The applied voltage waveforms for performing a DC RTN measurement are shown in Figure 24. A Vg voltage is supplied by an SMU unit and its value is guided by the regions of an IV measurement that shows clear signs of RTN. The Vd voltage is fixed at a constant level and supplied by an SMU unit. The Vd is fully adjustable but 0.1V is the typical value used initially. It is then expanded to more values when looking at a device in detail. The Id current is observed using a WGFMU unit, this is used as it can perform measurements more quickly than an SMU unit, which is preferable as it can capture fast RTN traps that can have capture and emission events within the microsecond range. The time window of a DC RTN measurement is also fully adjustable. Typically, the duration of the test should be long enough to observe at least several RTN events if the goal is simply scoping or long enough to observe at least 100 RTN events if the goal is to extract the characteristics of an RTN trap. The sampling rate selected is again, fully adjustable. The sampling rate should be selected so that it is at least 10 times faster than the speed of the fastest RTN event switching if the characterisation results are to be of an acceptable accuracy.

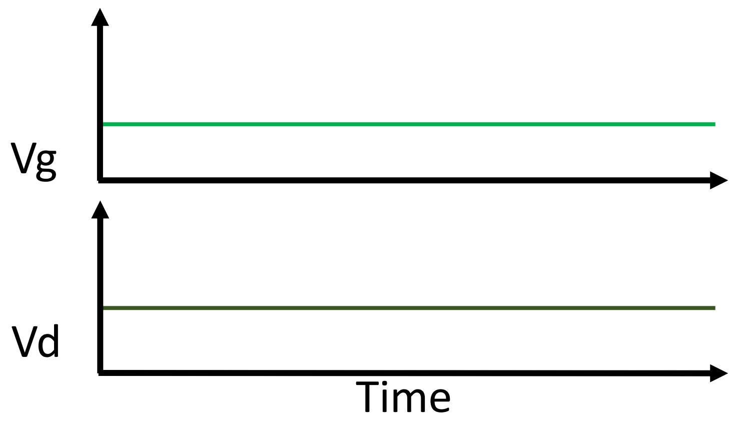


Figure 24, Applied voltage waveform for DC RTN measurement.

Figure 25 shows a typical result of a DC RTN measurement when performed on a device exhibiting clear single trap RTN behaviour. Figure 25 clearly shows the RTN switching events and the number of data points is significantly higher than 10 per switching event. The number of events visible is around

30, which is good for showing the type of RTN present in this device. If full characterisation of the trap was to be performed, however, a longer time should be used so that at least 100 RTN events are captured (run for around 100ms to capture a statistically significant number of events).

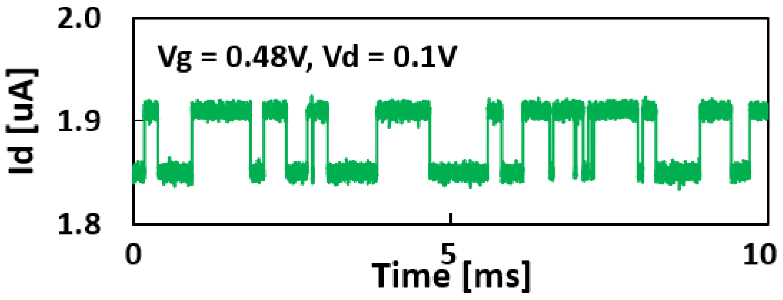


Figure 25, Typical type of result for a DC RTN measurement of a device with clear, single dominating trap RTN.

After a DC RTN measurement has been used to capture enough RTN events, the key parameters of that RTN trap can be extracted. To improve the accuracy of the parameters and to make extraction of them easier the raw Id data containing the RTN signal is fitted using a hidden Markov model (HMM) [98]. Figure 26a shows both the raw Id data and the fitting result. To extract the amplitude of the trap, ΔId, it is a simple case of subtracting the Id value of the captured state from that of the emitted state of the trap. The amplitude can also be expressed in other ways. To find the amplitude as a percentage of total current flow or ΔId/Id0, ΔId is divided by the value of the current in the emitted state, Id0. The amplitude in terms of ΔVth can be obtained by ΔVth=ΔId/gm, where gm can be obtained from the IV measurement as shown in Figure 23.

After fitting a DC RTN measurement result with the HMM technique, the values of τc and τe can be extracted. To find τc, all of the times that the RTN signal spends in the emitted state before transitioning to the captured state are extracted. This gives all of the tc values. The τc is the mean of the tc values. Finding the τe value uses the same method by extracting all of the times the RTN signal takes to transition from a captured state to an emitted state, giving the te values. Figure 26d shows the distribution of tc and te. If the extraction is accurate then the tc and te should follow an exponential distribution as they do in Figure 26d.

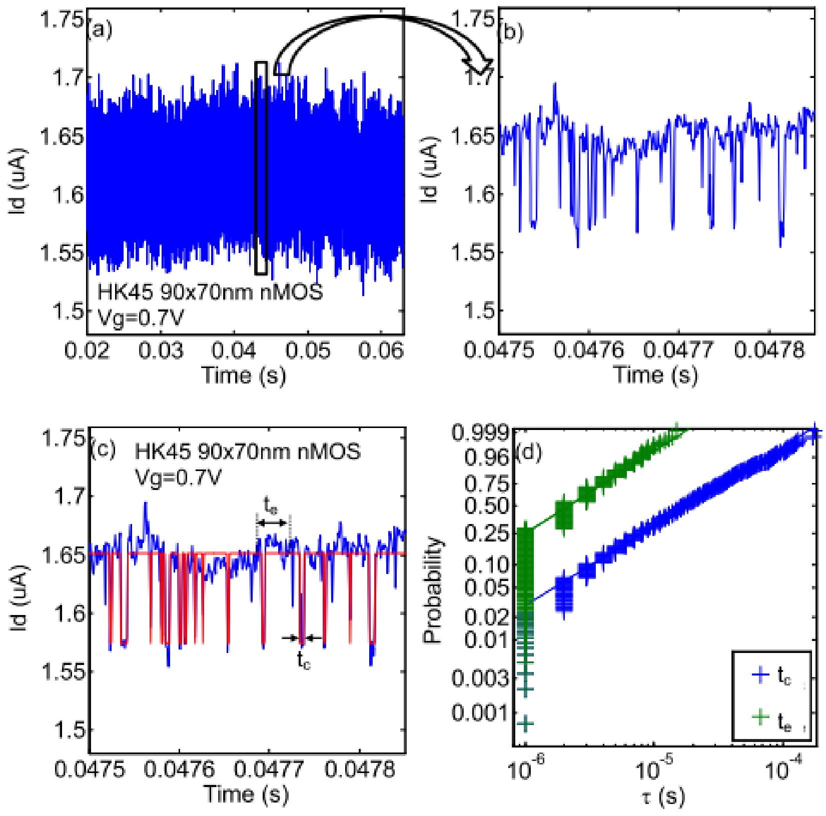


Figure 26, (a) Raw Id data with many RTN events (b) Zoomed in snapshot showing RTN events in Id data (c) Raw Id data in blue with HMM Fitting in red (d) distribution of tc and te as a Weibull plot. [97].

It is important to note that these characterisation methods are for single RTN traps. This methodology for RTN characterising is most effective when employed upon devices which show one clear dominating RTN trap. Extraction of multiple traps is possible using the HMM fitting technique, although adding more traps makes the extraction significantly more difficult and time consuming.

3.4.3 AC RTN Measurement

This work also utilises an AC RTN measurement scheme, to accelerate the events of an RTN trap. Using an AC gate voltage when performing an RTN measurement was first proposed by Zou et al. [99] as a way of better understanding RTN behaviour under digital circuit operations.

The basic principle of AC RTN measurement is to replace the constant Vg and Vd used in DC RTN measurement with two level voltages which alternate inversely at a pre-determined rate, as shown in Figure 27. The voltages are applied by a pair of SMUs. All voltage levels in this test are fully adjustable. As a general guide, VgH should be set to the highest voltage possible without overstressing the device, VgL should be the lowest voltage possible while still being able to accurately measure the Id current, VdH should be the highest value possible without overstressing the device, and VdL should be 0V. The frequency of switching between voltage levels (Fac) is also fully adjustable but as a general guide it should be as fast as possible with the equipment used.

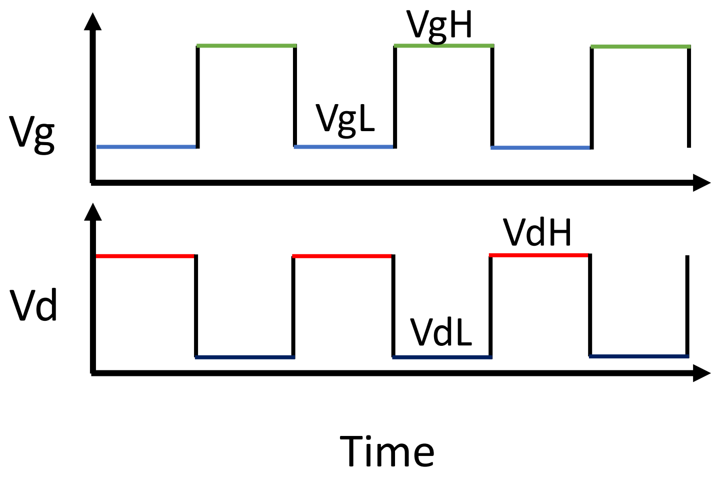


Figure 27, Applied voltage waveforms for AC RTN measurement.

In this test the Id current is measured only at VgL and not observed at VgH. This is done mainly for reasons of efficiency and design complexity. The Id current is measured using a WGFMU and the sampling rate should be selected to be at least 10 time faster than both Fac and the switching frequency of the fastest observed RTN trap.

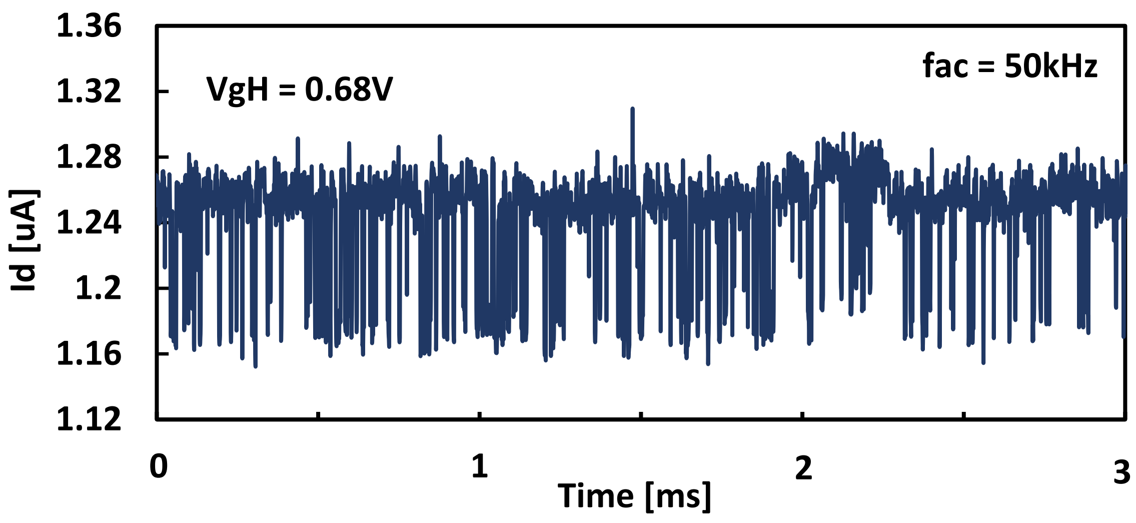


Figure 28, Typical result type from an AC RTN measurement of a device with clear, single dominating trap RTN.

Figure 28 shows a typical result of an AC RTN measurement when a single dominating trap is present within the measured device. After a result similar to Figure 28 has been obtained then the RTN trap can be characterised. The methodology used for characterisation of RTN traps observed by AC RTN is identical to that used in the DC RTN section. The only important difference is that the τc extracted is not the true τc value for that trap. As Id is measured only at VgL for the AC RTN, the vast majority of capture events will occur under VgH levels and will therefore not be observed until after switching to VgL. This results in some correlation between Fac and observed τc.

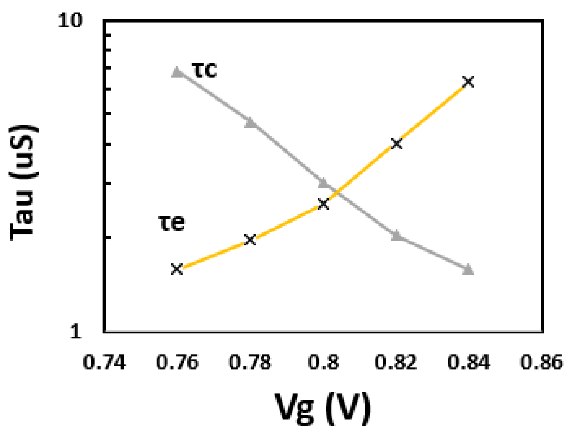


Figure 29,Experimental RTN timing information used to build FAC analysis model.

To further explore the effect of Fac on the observed τ values a number of simulations were run. The RTN model used in these simulations is built on a first-order Markov chain process. The full details of the model used were published by Toledano et al. [100]. To find Fac's effect, an RTN trap where τc(VgH) and τe(VgL) are equal to 2μs is simulated. These parameters are based on the timing analysis shown in Figure 29 with applied VgL = 0.78V and VgH = 0.82V. Figure 30 shows the results from two of these simulations where the Fac values were 1MHz (Figure 30a) and 40KHz (Figure 30b).

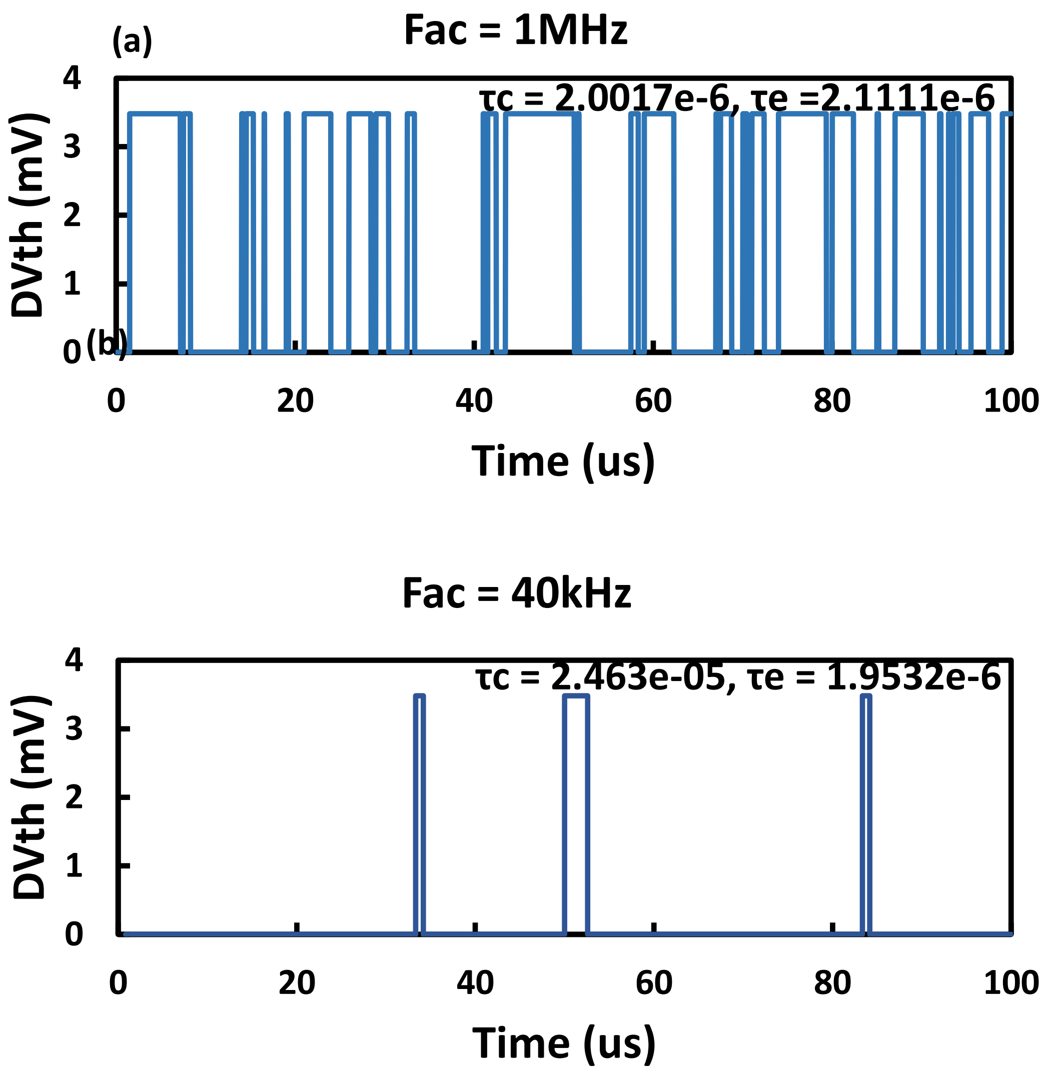


Figure 30. Two simulated AC RTNs under two different fac, 1MHz (a) and 40kHz (b), respectively. The time to capture/emission used in the simulation, τc and τe are set to be equal (=2μs).

In Figure 30 it can be seen that under the lower frequency, there are far fewer RTN events, indicating that the pre-set *τc* and *τe* are different from its apparent values, τc' and τe', which is extracted from the simulated RTN trace, showing the "shadowing" behaviour caused by AC RTN measurements. To find an appropriate minimum value for Fac (Fac_min), this simulation was run for a very large range of Fac and the resulting observed values for τc and τe are shown in Figure 31.

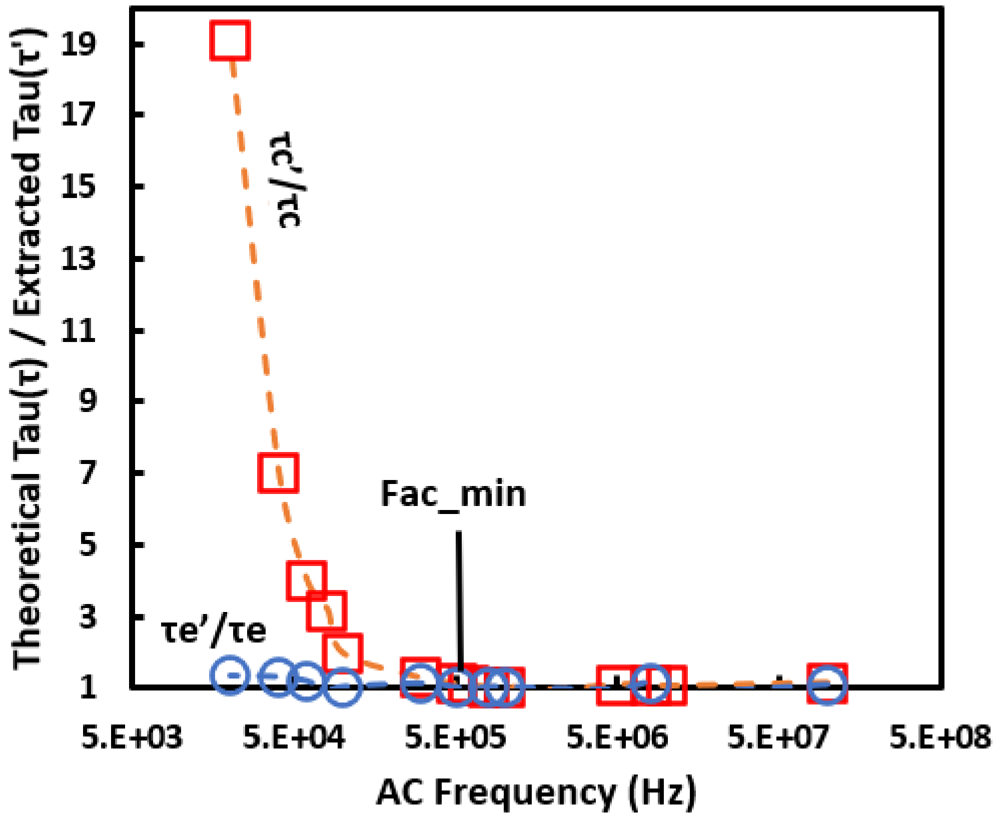


Figure 31, Impact of Fac on the time to capture/emission, represented by the ratio between the simulation parameters (τc & τe) and the apparent values extracted from the simulated AC RTN traces (τc' & τe').

In Figure 31, it can clearly be seen that Fac has a huge effect on the τc'/τc ratio but hardly any on the τe'/τe. In Figure 31, Fac_min is defined as the minimum frequency required to achieve a unity ratio between τc' and τc. This can be explained by the following, when the gate voltage is switched from VgH to VgL, the trapped charge can be emitted in a very short time, which, on average, is equal to the *τe* of this trap. Under the low Fac, the time under VgL is much longer than *τe.* Therefore, *τc* is controlled by time at VgL, which is much larger than the *τc* under VgH. This suggests that the Fac frequency must be higher than Fac_min to obtain accurate τc when performing an AC RTN measurement.

The final piece of exploration performed on this subject is looking at the relationship between Fac_min, τc and τe. To do this a wide range of τ values was randomly generated and the corresponding Fac_min was found by following the procedure laid out in Figure 31. The results of this process are shown in Figure 32.

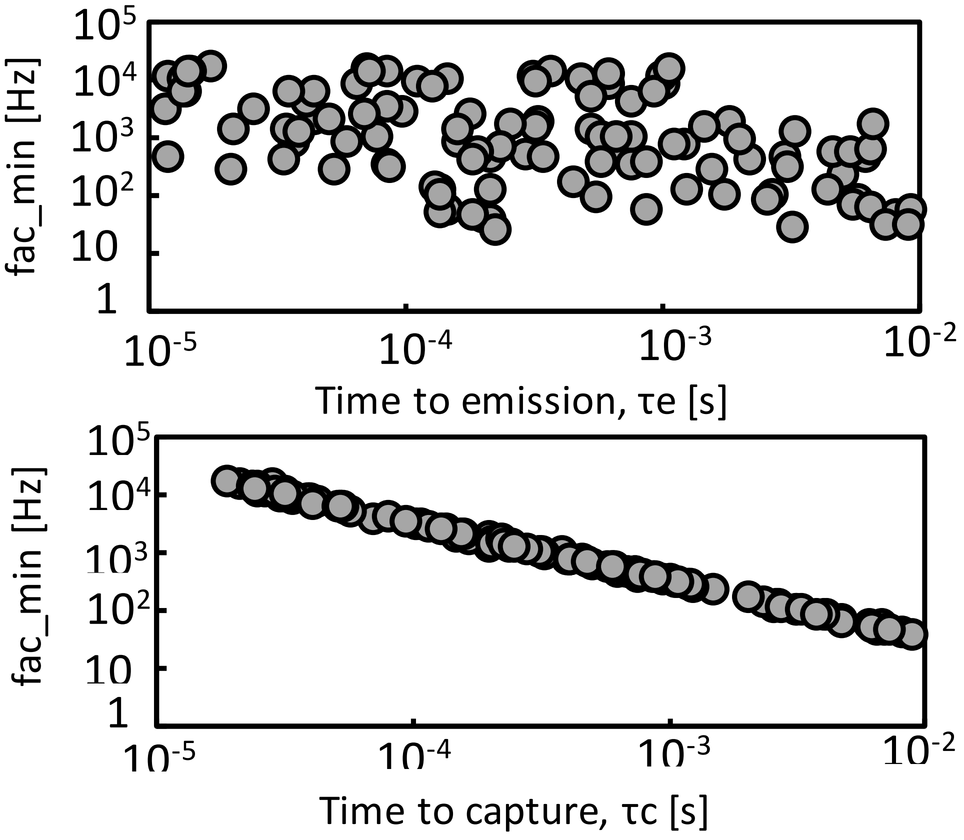


Figure 32, The correlation between the critical frequency, fac_min, and the time to capture and emission pre-set in the simulation (τc_th and τe_th), respectively.

Figure 32 shows that τc has a much stronger correlation with Fac_min than τe. The relationship between τc and Fac_min follows a strong power law correlation. This relationship can be used to find a goof value for Fac when performing an AC RTN measurement to ensure that the “shadowing” effect does not impact upon the potential speed of observed RTN events.

3.4.4 Hot Carrier or Bias Temperature Instability Device Stressing and Recovery

In this work defects are generated in devices that originally do not show any clear RTN in order to increase the yield of our TRNG designs. To do this, devices were stressed thermally and electrically. Chen, Puglisi and Brown et al. [101]–[103] showed the effectiveness of using heavy hot carrier (HC) or bias temperature instability (BTI) stresses when trying to generate defects. After applying heavy stresses, some recovery should be allowed to remove any temporary or unstable defects. B1500a control program can be used to apply both of these stresses and the recovery voltages. Figure 33 shows the voltage waveforms for stressing and recovery of devices.

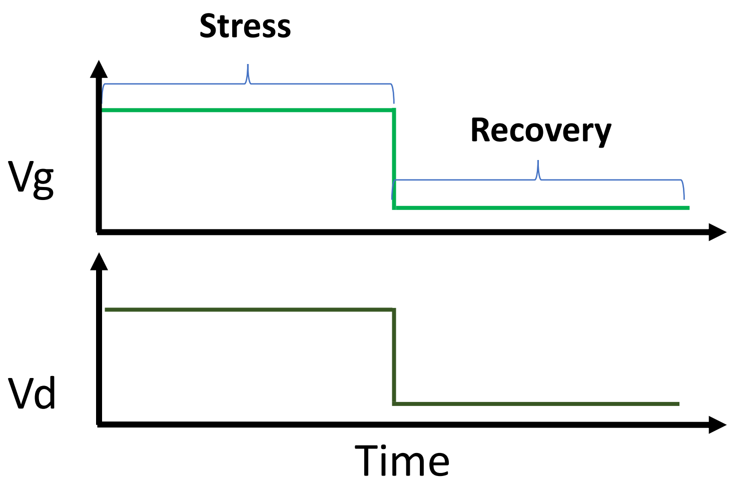


Figure 33, Device stressing and recovery voltage waveforms.

The voltages for stressing and recovery are supplied by 2 SMUs. Both stressing and recovery are best performed at a high temperature of 120˚C controlled by the hot plate on the cascade MPS150 probe station. For HC stress the stress voltages for Vg and Vd should be equal and can range from around

1.7V to 2V for the samples used in this work. For BTI stress the Vd voltage supplied should be 0V and the Vg stress should be between 1.7V-2V for the samples used in this work. For both HC and BTI the stress time is fully adjustable, typically ranged from 10s to 100s in this work. For the recovery, Vd is equal to 0V and Vg was typically -0.6V. The recovery time is also fully adjustable but typically ranged from 10s to 100s depending upon the specific test performed. Figure 34 shows a typical IV measurement result from a device when it is fresh, after HC stressing, and after recovery. The specific conditions for the example in Figure 34 were 1.9V HC stress voltages for 100s and -0.6V recovery voltage for 100s applied to one of the 22nm FinFET samples. In Figure 34 it can be seen that after stressing, the IV has significantly degraded and clear RTN switching appears. After recovery, the IV are partially recovered, but the RTN switching is still present.

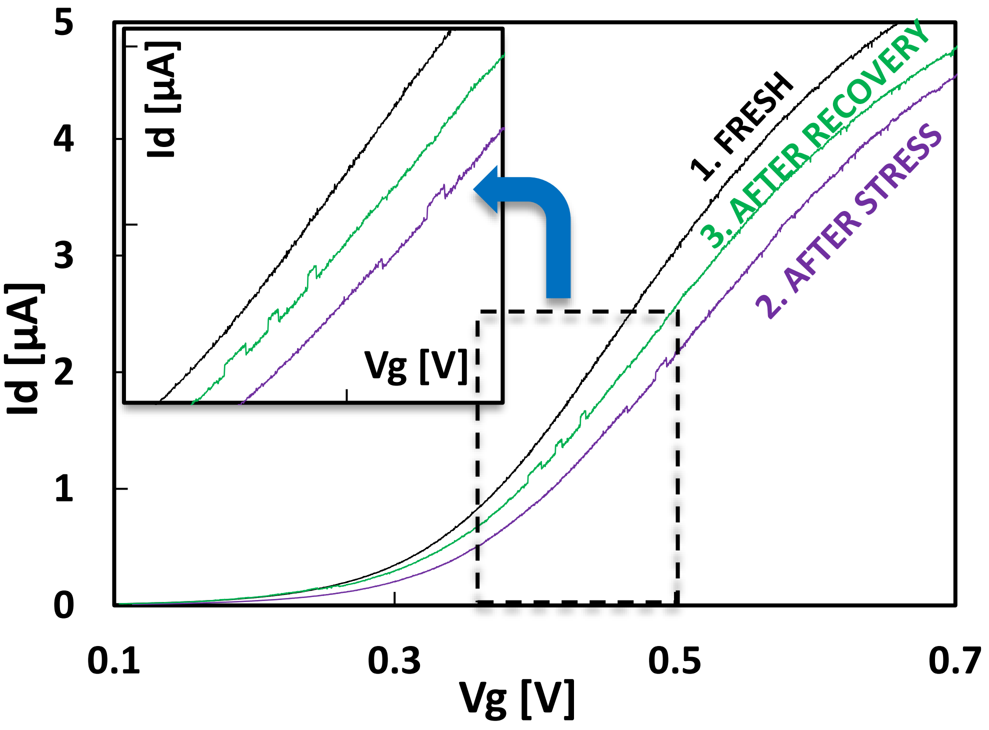


Figure 34, Typical IV measurement results before and after stressing and recovery on a 22nm FinFET device.

3.4.5 Time Dependent Defect Spectroscopy (TDDS) Measurement

TDDS measurement is the basis of the PUF section of this work. The TDDS measurement technique was first proposed by Grasser et al.[104] as a technique for characterising BTI within a device. The basic principle of TDDS measurement is to charge all of the defects within a given device with a high gate voltage and then observe all of the de-trapping events when this gate voltage is dropped to a low voltage.

Figure 35 shows the voltage waveforms of a TDDS measurement. It begins with a stabilisation period where both the applied Vg and Vd voltages are 0V, the duration of this time is adjustable. The typical stabilisation time used in this work is 0.1s. After stabilisation, the Vg voltage is increased to the VgH level which is used to charge as many traps as possible within the device. The typical value for VgH used in this work is 1.2V, which is high enough to give rapid trap charging but not high enough to overstress the device. The Vd voltage is left at 0V during this trap charging stage in order to minimise device stress. The time spent charging traps is adjustable but typically done for 0.1s in this work. After trap charging, the Vg voltage is dropped to the Vgmeasure level. In the original proposal for TDDS, Vgmeasure was selected as Vth [104] but in this work we use a Vg slightly lower than Vth such as 0.36V for the HK45 devices, whose Vth is ~0.4V. During this measurement stage the Vd voltage is stepped up so that the Id current can be measured, because we are using a very low Vg during this stage, the Vd is slightly increased compared to other tests in this work to a value of 0.3V which gives a measurable Id current. In this test the only measured parameter is the Id current during the Vgmeasure phase of the test. The Id current is measured using a WGFMU allowing for fast measurement with a sampling rate of 100ksa/s. The measurement phase typically lasts for 0.1s in this work. For our PUF application this TDDS measurement is usually repeated many times on each device between 100 and 1,000 times.

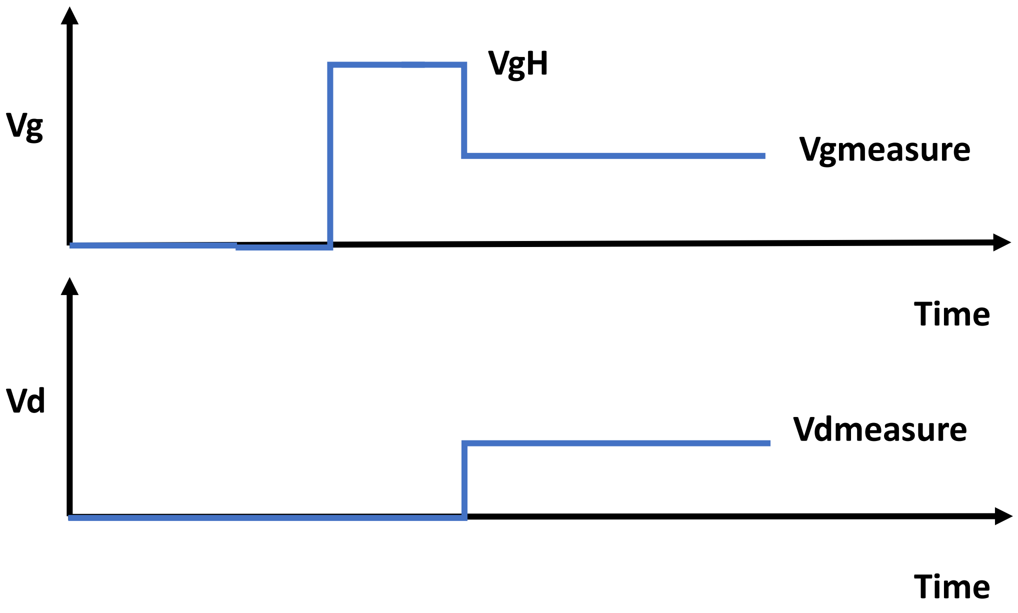


Figure 35, Applied voltages for TDDS measurement.

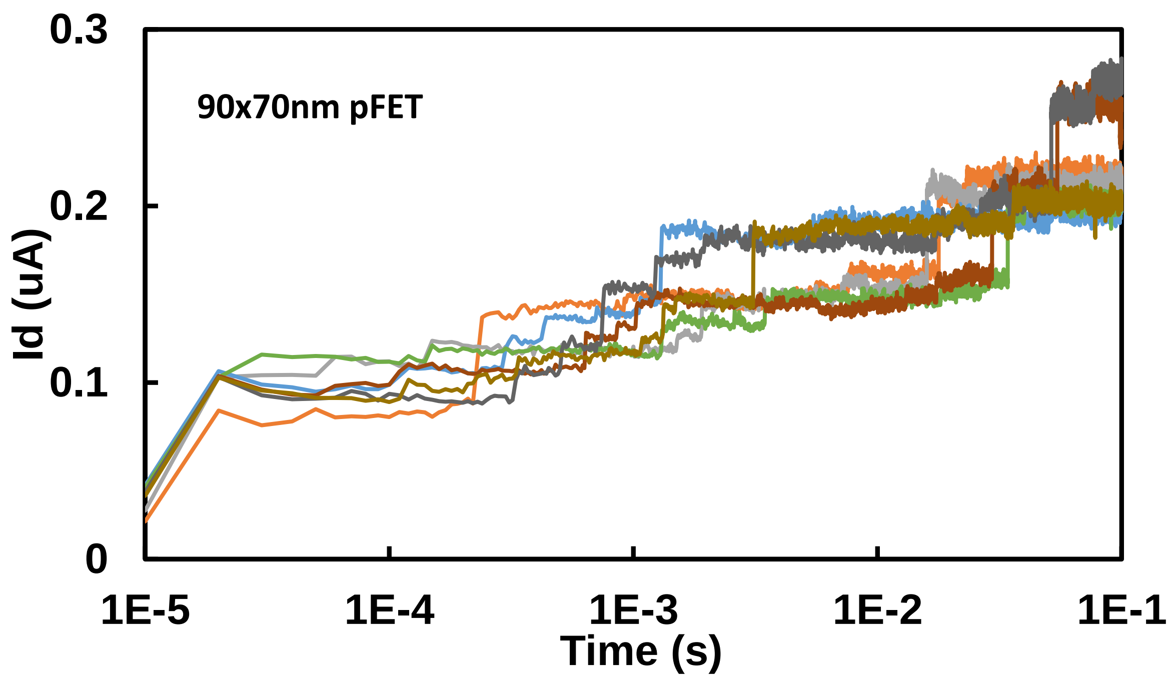


Figure 36, Typical Id measurement results from 6 TDDS measurements on a HK45 device.

Figure 36 shows a typical result from observing the Id current in a HK45 device, after applying the TDDS measurement technique 6 times. In Figure 36 the de-trapping events can clearly be observed when the Id current jumps up a level very quickly.

Chapter 4 - A novel Random Telegraph Noise (RTN) Based TRNG

4.1 Introduction

As discussed in section 2.1.2, RTN is a phenomenon that has been observed in transistors for decades but is now becoming a key issue with many research papers focussed on it ([97], [102], [113], [105]–[112] are examples of recent papers). RTN is caused by defects within a device which have a chance to capture or emit charge carriers flowing through the device channel [114]. The effect of these captures and emissions is a characteristic square wave effect in the drain and/or gate current of the device [101]. RTN is a growing issue in the nano-electronics community because it creates device reliability problems as the current fluctuations can result in reduced device lifetime and unwanted noise. RTN is gaining more attention from researchers because it has been observed that the amplitude of the RTN signal is inversely proportional to device size, meaning that as devices are getting smaller every year, the impact of RTN on the device performance is larger[113], [115], [116].

As described in section 2.1.2, RTN has been proposed as an entropy source for TRNGs in several papers and shows key benefits over other types of noise when it comes to resistance to noise manipulation attacks[37], [48], [53], [60], [117].

Figure 37 shows the conventional design for an RTN-based TRNG, where the Id current in the device is harvested and sampled to gain a random sequence. Some post-processing is then applied to gain the random bitstream. However, all of the previously suggested designs have had issues with either low speed or poor device selectivity. Some recent research into RTN has given some potential improvements to these problems that are investigated in this work.

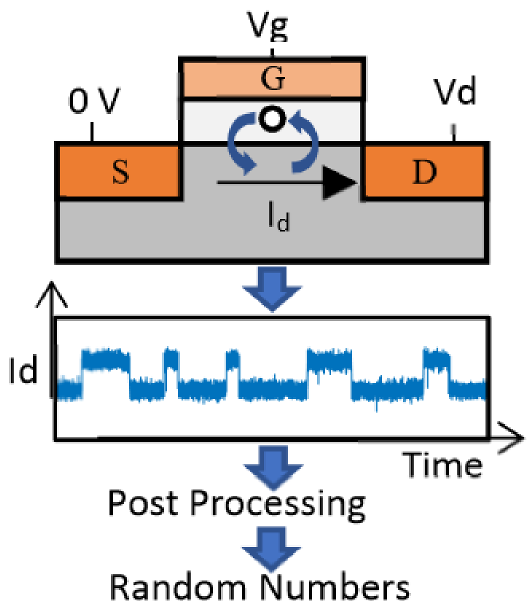


Figure 37, Conventional RTN based TRNG design.

To begin, improvements to the utilisation of the individual device as the entropy source in isolation are examined. The first major improvement is how the chance of finding clear RTN can be increased. In Chen et al.'s 2011 paper [101] and Puglisi et al.'s 2016 paper [102] it is shown that the RTN phenomenon can be generated in devices that do not initially show any, through heavy electrical and thermal stress. In this work how stressing can be used to reliably create RTN are investigated, to improve the selectivity.

The second Improvement over the traditional RTN-based TRNG scheme is the rate at which any given RTN trap can generate random numbers. The use of an AC signal on the gate allows the average capture and emission times to be controlled independently, whereas with a DC signal they have an inverse relationship with Vg, so one will increase as the other decreases, which limits the maximum generation speed. In this work the effect of using an AC signal is further investigated to see how much the RTN signal can be accelerated, in the hope of solving the problem of slow number generation rate with traditional RTN-based TRNGs.

After looking at how to improve the efficiency of the entropy source in isolation, improvement in circuit design and components of the TRNG are studied. This includes how the RTN signal is extracted, manipulated, and the methodology used for converting the entropy into random numbers. In the circuit design section, research is carried out on how multi-level and abnormal RTN can be extracted as well as how the security and efficiency of the circuit can be improved.

Finally, the overall performance of the proposed TRNG design is compared with other state-of-the-art TRNG designs. To do this the two main randomness tests suites, the NIST and the DIEHARD tests are used. The emerging technique of neural networks is used to test the resistance of the proposed TRNG against cracking. The key stats and considerations of the proposed design for the specific applications are discussed against other TRNG options.

4.2 - Generating RTN traps

In order to improve upon device selectivity, which has been a problem for previous RTN-based TRNGs, this work explores the use of electrical and thermal stress to generate traps into any device. By stressing the transistors with a voltage higher than operating condition, new traps can be generated through either breaking the hydrogen bond [102] or converting some kind of precursors [109]. These generated traps can also induce RTN, like pre-existing traps [102]. This provides a pathway to insert RTN into nano-scaled devices. This work explores how to optimise this stressing technique for generating traps so that they can be used as an entropy source for a TRNG. These generated traps are analysed, and their key parameters are characterized, including the types of traps generated, the number of traps generated per device, their amplitude, location, and timing characteristics.

4.2.1 Optimising Trap Generation Procedure

The first step in this process was to find the optimum parameters for generating traps in a given device. First, we look at what type of stress gives the best chance of generating traps as well as at what temperature. After finding what stress type is most effective, we look at what are the most efficient stress voltages and times to generate traps reliably and efficiently, which will be performed a single time during manufacture. We also look at the stability of these generated traps to make sure they are stable and will not disappear after some time. To accelerate the stability testing and to eliminate temporary defects we explore the use of a recovery voltage. The specific voltage used for recovery as well as recovery time and temperature are explored. This parameter optimisation work was mainly performed on the 22nm FinFET samples as they are the most numerous, allowing us to achieve good statistical confidence in the results. After the optimal values were found, several HK45 and HK28 devices were also tested to show the procedure can work with devices from different fabrication processes.

To make sure we are only looking at generated traps for this section we first scoped all tested devices to see if there was any pre-existing RTN. The procedure for this probing is illustrated in Figure 38. The probing starts with an IV measurement to see if there is any clear RTN on the IV. The IV measurement should range from Vg=0.1V to 1.1V in steps of 2mV with a Vd of 0.1V. If there are clear RTN steps on the IV, then we can confidently say there is clear pre-existing RTN without looking in further detail. If the IV measurement does not show any clear RTN then we look more closely at the DC RTN measurements. For the DC RTN tests, we repeat the measurement 36 times sweeping Vg values from 0.4V to 1.1V, in steps of 0.02V with a constant Vd of 0.1V with a measurement duration of 0.1s. The DC RTN measurement is first done with a sampling frequency of 100kSa/s. If no extractable RTN can be observed in any of these measurements we repeat the DC RTN measurement with a sampling frequency of 1MSa/s, and if this has no extractable RTN we repeat again with a sampling frequency of 1kSa/s. Only if all of these measurements do not show any extractable RTN do we consider the device to have no pre-existing traps. Figure 39 shows the results from this scoping for the 90nm x 28nm devices of the 22nm FinFET wafer, where we can see that 38% of tested devices show no signs of any RTN traps being present. It is important to note that even though 58% of devices showed some signs of RTN it does not mean that this RTN would be useful as an entropy source for a TRNG as the current processing circuits require clear edges and plateaus in order to extract the timing information. We only aim to look at generated traps in this section so devices that showed any signs of RTN are avoided in later testing.

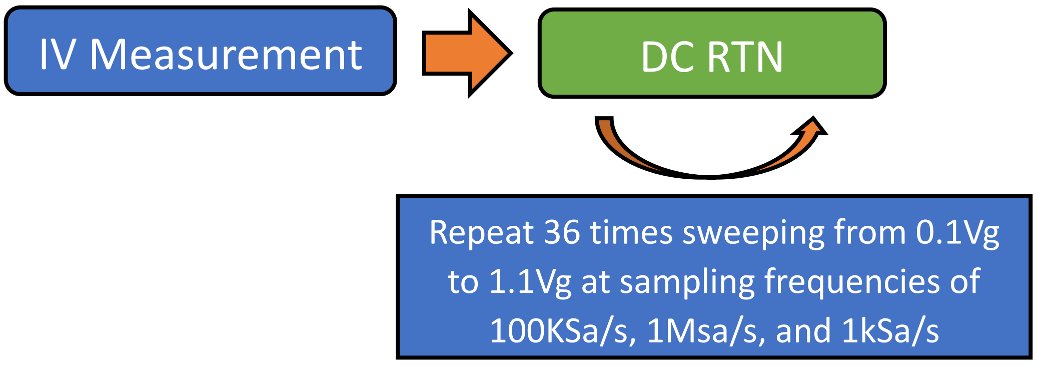

IV Measurement
DC RTN
Repeat 36 times sweeping from 0.1Vg to 1.1Vg at sampling frequencies of 100KSa/s, 1Msa/s, and 1kSa/s

Figure 38, Scoping procedure for finding pre-existing RTN.

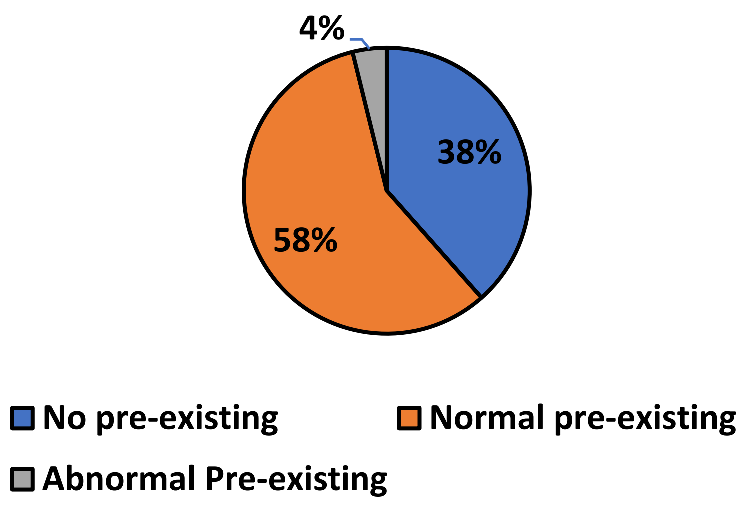


Figure 39, Rates of pre-existing traps in 175 9nm x 28nm n-type 22nm FinFET devices.

The next step was to find the stress conditions which gave the highest chance of generating stable RTN in our devices. This stress will be performed a single time during manufacturing. Several stress modes can introduce trap generation, such as Hot Carrier Injection (HCI) and Bias Temperature instability (BTI). BTI is a uniform stress and therefore independent of device geometry [101]. HCI stress is non-uniform and can be enhanced with shorter channel length [108]. It has been reported that, for some advanced CMOS technologies, HCI can be more effective than BTI in generating new defects [107]. To verify the idea that HC can be more effective than BTI and to find the optimal stress conditions, scoping tests using both BTI and HC stress with a range of voltages, temperatures, and stress times were performed. Figure 40 shows the procedure used to scope each device. 5 devices for each BTI and HC were tested for each stress condition.

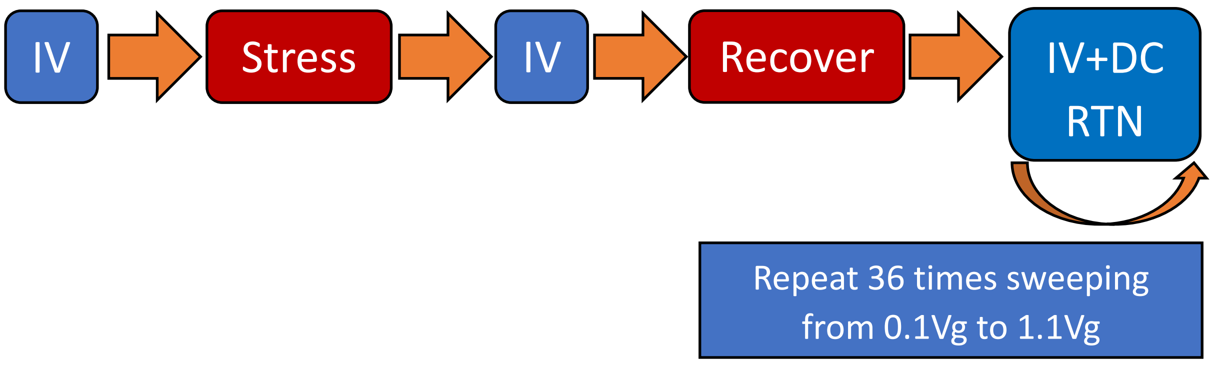


Figure 40, Procedure for optimal stress condition scoping on devices showing no pre-existing RTN. stress can be BTI or HC.

The scoping tests ranged from stress voltages of 1.5V (Vg=1.5V, Vd=0V for BTI and Vg=Vd=1.5V for HC) to stress voltages of 2.1V (Vg=2.1V, Vd=0V for BTI and Vg=Vd=2.1V for HC). These voltages were applied for 10 seconds at 30˚C and 125˚C for each stress type. Table 5 shows the results from these scoping tests. From the scoping tests a HC at 125˚C and 1.9V stress voltages was found to be optimal. The higher stress of 2.1V initially shows a very good generation rate, but devices have a high chance of breaking down soon after this stress is applied, which is taken into account in the results of Table 5. With the stress voltages and temperature now selected the stress time was further explored. Stress times of 1s, 5s, 10s, 30s, and 50s were applied. The results from this testing are shown in Figure 41. We found that increasing the stress time increases the chance of generating RTN, however this increased chance follows a power law that gives diminishing returns in terms of generation chance. From these tests we selected 50s as the final stress time as it provided a good generation chance of 85% and increasing the generation chance any higher than this would require significant stress times that would be impractical for real world use.

Table 5, RTN trap generation rates under different stress conditions. Optimal results are highlighted.

Stress Type	**Stress Temperature (˚C)**	**Stress Voltage (V)**	**Generation Rate (%)**
BTI	30	1.5	0
		1.7	0
		1.9	0
		2.1	0
	125	1.5	0
		1.7	0
		1.9	0
		2.1	20
HC	30	1.5	0
		1.7	0
		1.9	0
		2.1	0
	125	1.5	0
		1.7	20
		1.9	**80**
		2.1	20

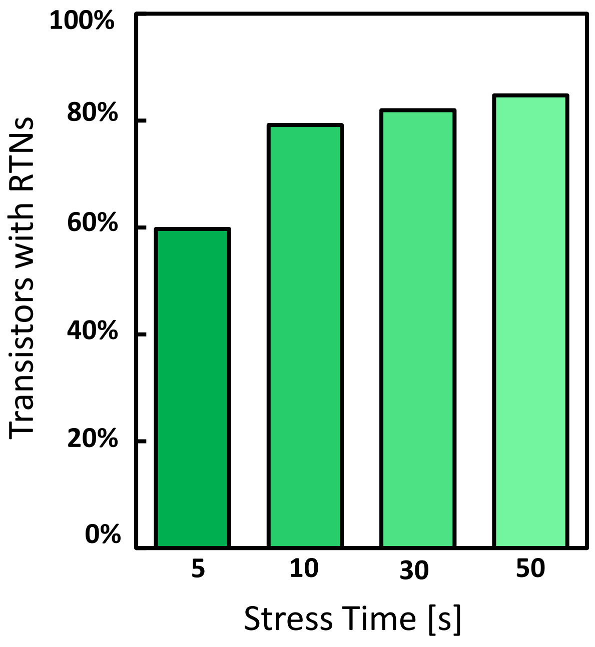


Figure 41, Percentage of devices showing clear RTN after given stress times.

Our findings fit with the widely accepted idea that the HCI of nano-scaled devices is driven by carrier energy and carrier–carrier interaction [112]. As a consequence, the most effective trap generation occurs under Vg = Vd with the other two terminals grounded. This supports our choice to use Vg=Vd=1.9V at 125° C to accelerate the trap generation during preliminary stressing. Others have also suggested that high temperature can further accelerate the generation of traps [102].

Figure 42 shows a typical result for the IV curves measured when a fresh device is stressed and recovered following the optimal procedure explored in this section. The fresh IV shows no signs of RTN switching whereas both the stressed and recovered IV curves show clear RTN switching. The stressed IV curve shows clear degradation compared to the fresh IV. After recovery, the IV curve is shifted closer to the fresh IV. Figure 43 shows typical DC RTN measurements of a device that has been stressed and recovered following the procedure outlined in this section. Normally a DC RTN measurement is not performed between the stress and recovery steps, however one has been included here to show the effects of the process in full. The fresh DC RTN measurement has no signs of RTN switching. After both stress and recovery, clear RTN is present showing that the stressing procedure has been successful.

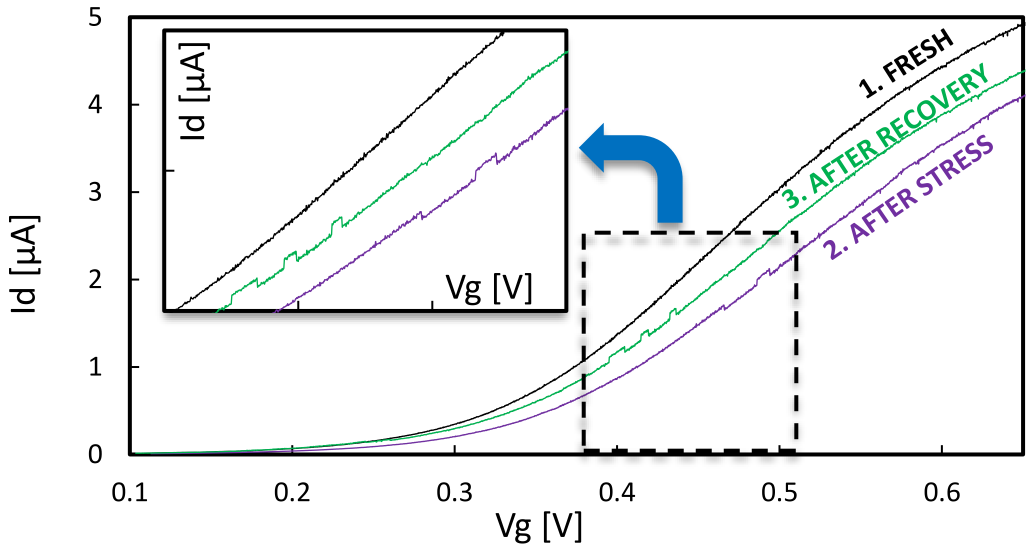


Figure 42. I-V curves showing generation of an RTN trap after stress and after recovery.

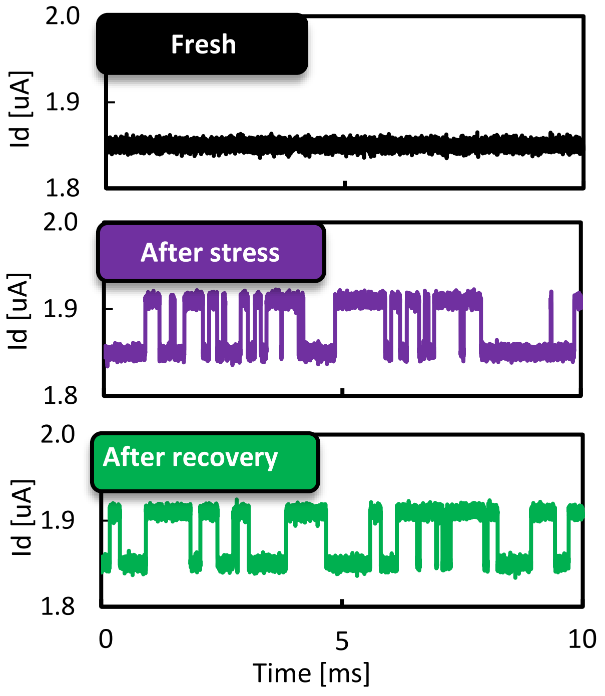


Figure 43, RTN measurements on (a) fresh (b) stresses (c) recovered device following the procedure outlined in section 4.2.1.

As discussed, when introducing the recovery process, electrical stress can not only generate new defects but will also trigger unstable traps. Although both types of trap introduce RTN when immediately measured after the stress, it has been reported that the latter will recover and disappear after floating for some time. To make sure that our recovery procedure was sufficient for eliminating temporary traps we measured the time constants of the generated traps both immediately after generation and one month later. Figure 44 shows the results from the time constant measurement of the RTN immediately after stress and after 1 month on 20 devices. It can be seen that the time constant is very stable after 1 month for all of the tested devices. As the time constant is stable on all of the tested devices, we can say with some confidence that the recovery procedure is effective at removing unstable traps and that our generation procedure is effective at creating long term RTN traps.

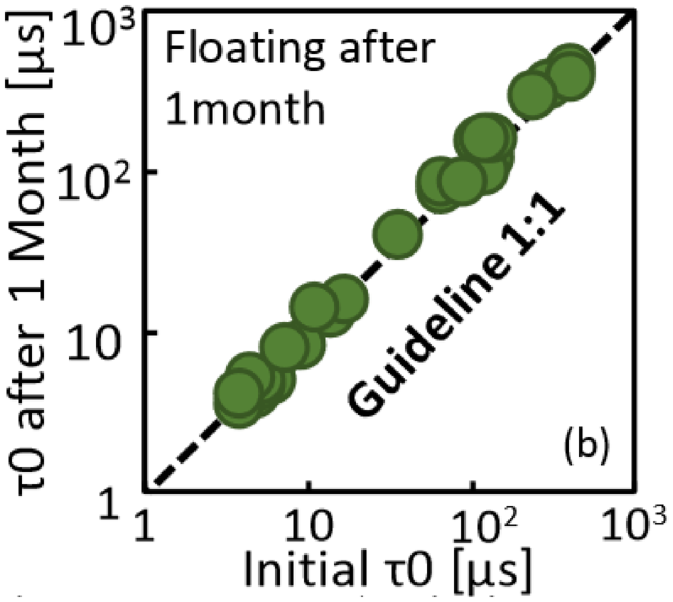


Figure 44, Time constant of generated RTN signal in 20 devices immediately after generation and after 1 month.

The final test for the generation procedure was the effect of the stressing on the gate current (Ig). If the stress level is too harsh the gate current of the device increases, because the damage in the gate oxide results in larger leakage current. This greater leakage current would lead to decreased efficiency of the final design and also could indicate the lifetime of the stressed devices has been reduced too significantly. To check the leakage current through the gate, Ig versus Vg measurements were performed. These measurements follow the same procedure as the IV sweeps, but the Ig is recorded rather than the Id. The Vd for these measurements was fixed at 0.1V. Figure 45 shows the results of a typical Ig, Vg measurement. The results from the Ig, Vg measurements show that the increase in the Ig current is not significant after stressing (still in the pA range) which shows that our stress level is not causing too much damage to the oxide of the devices.

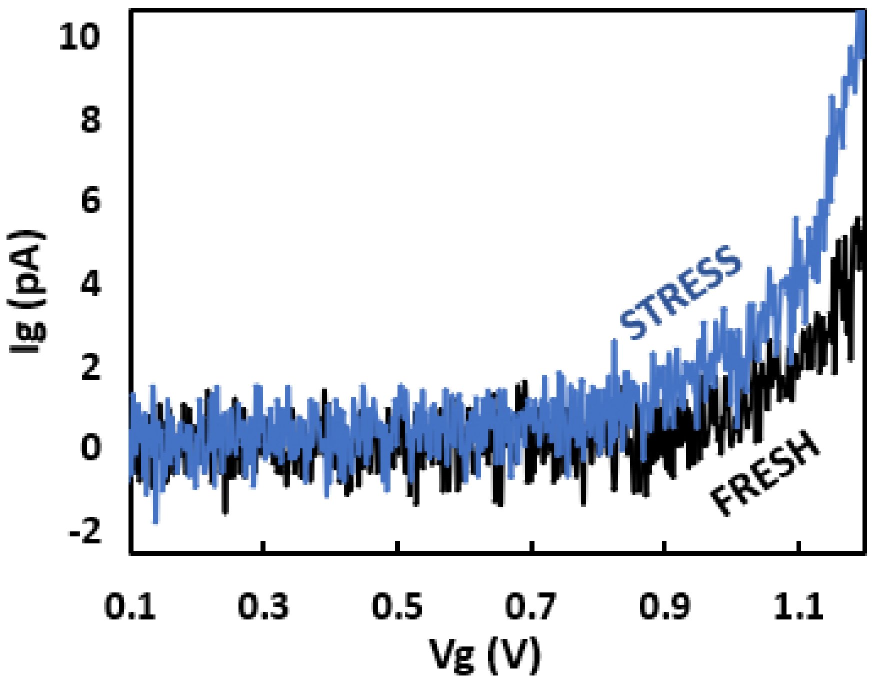


Figure 45, typical Ig, Vg measurement of a device before and after optimal stressing procedure.

With all of the key parameters for the generation of RTN traps found the final procedure could be set. In the final design it is not necessary to separate pre-existing and generated traps so the initial scoping and measurement to eliminate pre-existing traps does not need to be performed. It is still important to do an IV measurement after recovery to test the device is still working correctly and also to do a DC RTN measurement to find which devices have clear RTN so they can be used in the TRNG circuit. The final procedure is shown in Figure 46, with the stress and recovery times being 50s and the IV and DC RTN measurements matching the parameters used in the scoping tests from this section.

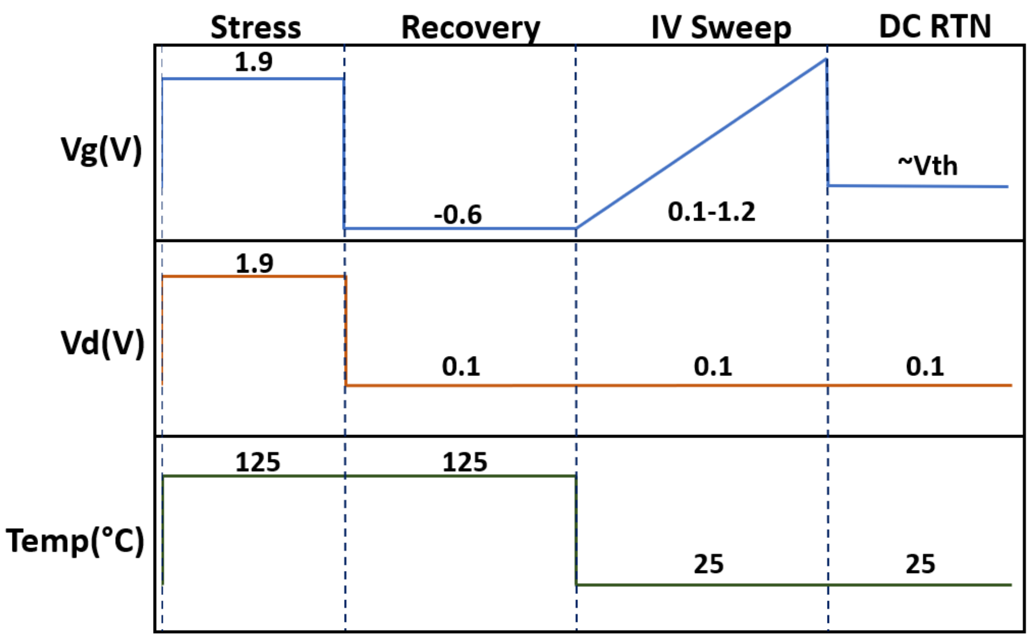


Figure 46, Generation procedure to be used in final design.

4.2.2 Characterisation of Generated RTN Traps

In order to better understand the properties of the generated RTN traps, so that they can be utilised to their maximum extent as an entropy source of a TRNG, a number of different characterisation techniques were used. The first aspect of the generated RTN to be characterised was the Vg dependence of the capture and emission times. This was done to find what Vg values would give the optimal τ values for TRNG operation. Following the Vg dependence, we look at the Vd dependence of the RTN traps to see how this affects the τ values extracted from the traps. After fully exploring the voltage dependencies of the τ times we look into the location of the RTN traps within the stressed devices. We first look at the location of the traps in the vertical plane (distance from substrate and gate). The location of the traps caused by generation can give us some insight into how they differ from pre-existing traps and if using generated traps can have some advantages over pre-existing traps. Finally, for the characterisation techniques we look at the amplitude of generated traps over a range of Vg values to find what effect Vg has on single trap impact and if this can be used to improve performance of the final TRNG design.

To explore the Vg dependence of the capture and emission times a large number of RTN traces were required. Following the procedure finalised in section 4.2.1 we generated RTN traps in 33 devices and performed multiple DC RTN measurements on each device sweeping all of the Vg values where RTN could be observed. After the DC RTN measurements were collected the capture and emission events were extracted from the RTN traps present in the Id. The RTN events were extracted using the HMM fitting technique discussed in section 3.4.2. Figure 47 shows the extracted tc and te values from a typical DC RTN measurement. It shows that the tc and te for generated traps follow an exponential distribution in the same way as pre-existing traps do. The average of the tc and te values is then taken to give τc and τe.

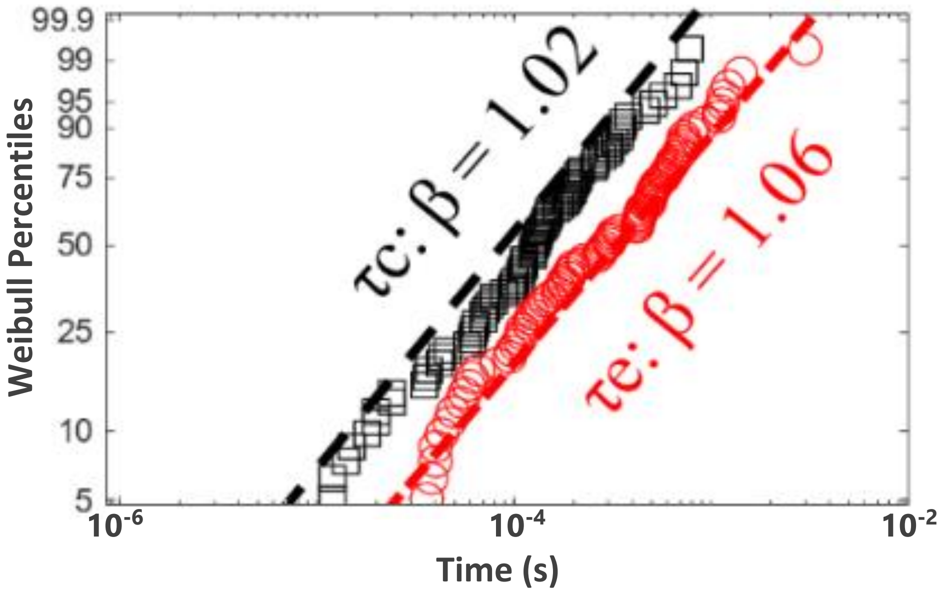


Figure 47, tc and te extraction using HMM method from DC RTN measurement with constant Vg with exponential fitting.

By performing the same test procedure across the multiple DC RTN measurements, sweeping Vg values for each trap, the typical voltage dependence for τc, τe and τc/τe can be extracted. Figure 48 shows the most common type of Vg dependence that was found (this type of dependence will be referred to as type 1 dependence) and exhibits a similar trend to that of the RTN signals from pre-existing traps discussed in previous works [99], [100], [118]. This type of RTN accounts for 81% of extract dependencies. In this type 1 dependency, the log(τe) is proportional to Vg and log(τc) is inversely proportional to Vg.

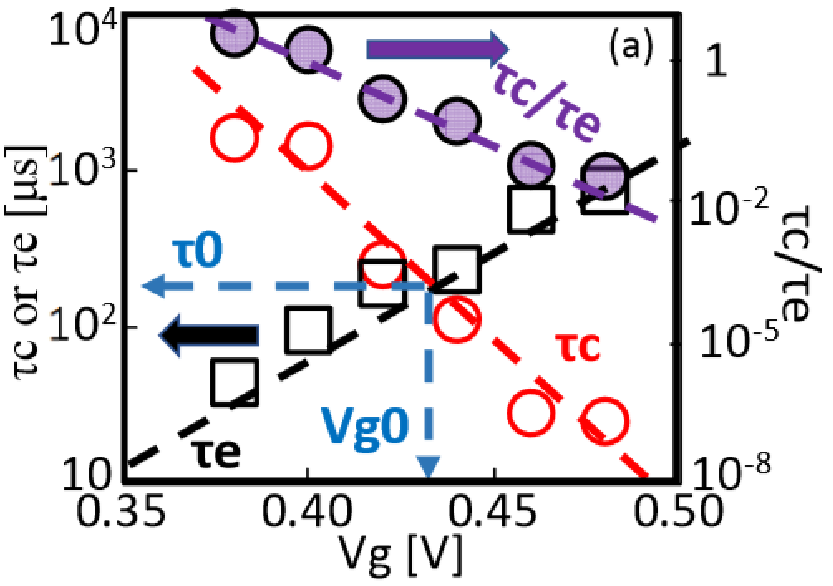


Figure 48, Example of most common type (type 1) of Vg dependence of tc, te, and tc/te for a generated RTN trap.

We also found a second type of Vg dependence (type 2) that accounts for 19% of the extracted dependencies. Figure 49 shows that log(τc) is inversely proportional to Vg, whereas log(τe) has little dependence on Vg. For both types of traps τc is typically more sensitive to Vg than τe, which can be seen from the downward trend of τc/τe when Vg is increased. In traditional RTN based TRNG designs the optimal Vg to use for operation would be as high as possible to take advantage of the reduced τc/τe at higher Vg values. The lower the τc/τe the more RTN events will be observed in any given time resulting in a faster entropy source and therefore faster number generation.

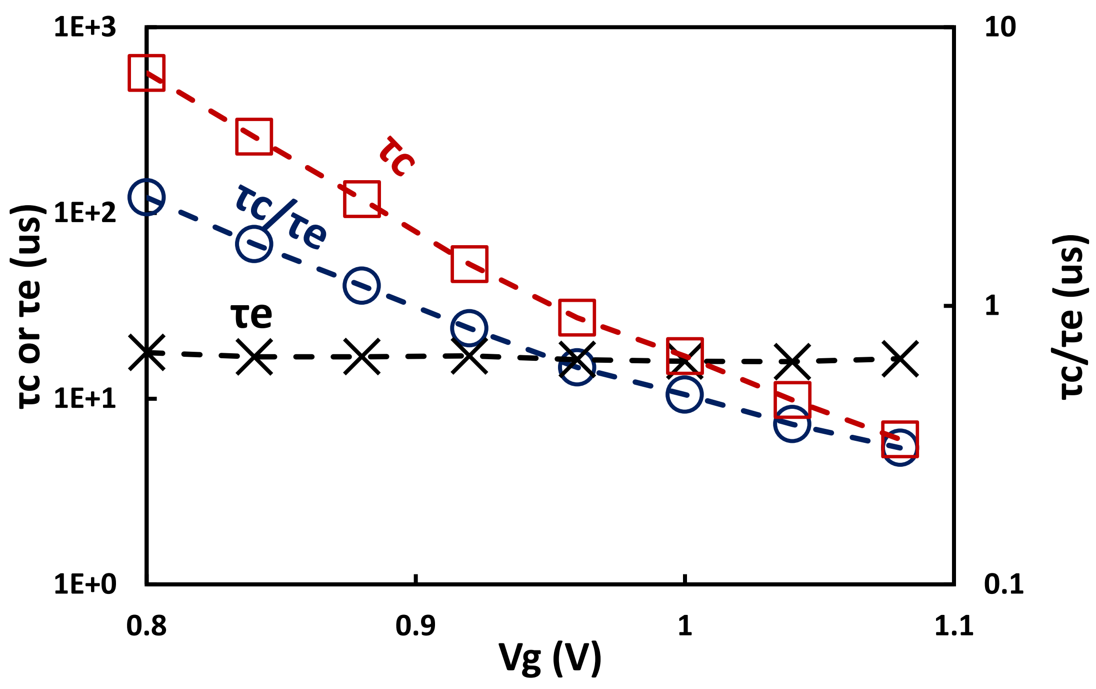


Figure 49, Example of type 2 trap's Vg dependence for tc, te, and tc/te for a generated RTN trap.

The next thing that we look at is how the capture and emission times of generated traps respond to different applied Vd voltages. To do this we applied the same methodology as that for finding Vg dependence and we repeated the measurements over a range of Vd values from 0.1V to 0.3V. After extracting all of the τ values for an RTN trap we plotted the Vg dependence of τc and τe against Vg for each Vd voltage that was applied. Figure 50 shows a typical response of a generated trap to different applied Vd values. It can be seen that changing the Vd applied to a generated trap has the ability to shift the τ values to the left or right for any given Vg value. This effect can be quite significant. If we look at a Vg value of 0.35 in Figure 50, the τe value is over 1 order of magnitude shorter under 0.3Vd than it is at 0.1Vd. Across the devices we tested to see if the slope of τc and τe with regards to applied Vg remains constant under different applied Vd values. To find the average effect of changing Vd on generated RTN defects, this measurement was repeated across 12 devices, varying the applied Vd from 0.1 to 0.5V. Once the τ values were extracted for each trap at every voltage level, the average shift in the Vg required to give τ0 was extracted. Figure 51 illustrates how the Vg at τo are shifted when Vd is increased.

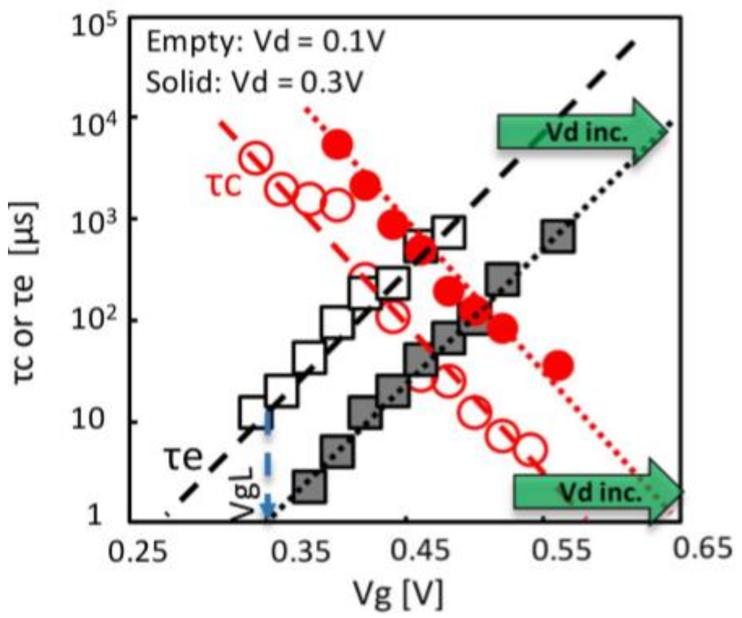


Figure 50, Typical Response of τ values to a change in Vd for a generated RTN trap.

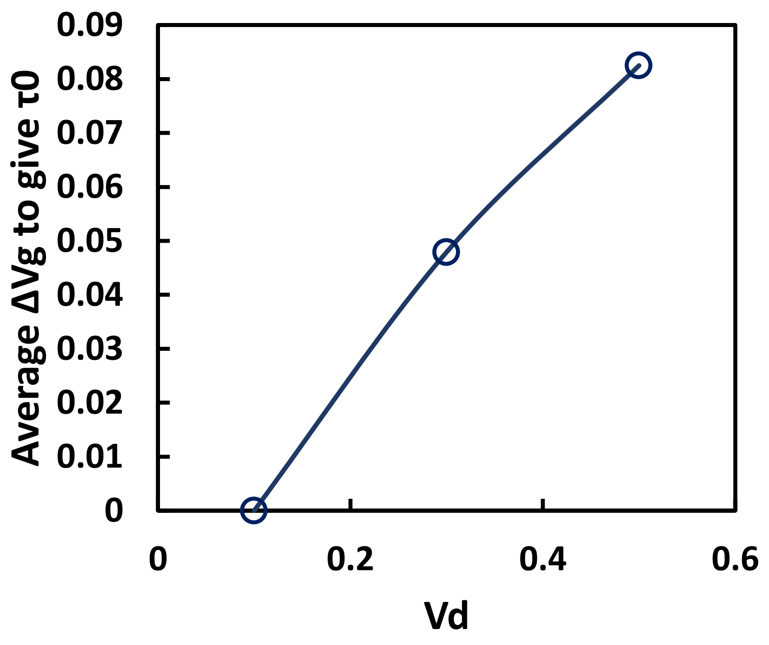


Figure 51, Average shift of Vg required to give τ0 when increasing applied Vd from 0.1V(baseline) to 0.5V across 12 generated RTN defects.

Next, we look at the location of the generated defects within the devices. First, the spatial location in terms of distance from the gate and substrate of 20 generated traps was extracted using equation 3 which is derived in detail in [119], the results of this process are shown in Figure 52.

$$\frac{Tox-Xt}{Tox} = \frac{kT}{q}\frac{\partial \ln\left(\frac{\tau c}{\tau e}\right)}{\partial Vg}, \tag{3}$$

where:
Tox = oxide thickness
Xt = trap position in vertical direction
k = Boltzmann constant
T = temperature
q= magnitude of the electrical charge on the electron

Looking at the results in Figure 52 we can see that the location of the majority of the traps is away from the substrate interface and also away from the gate interface distributed in a close to normal distribution, slightly biased towards the substrate. As the traps are away from the interface it is more difficult for charge carriers to be captured from or emitted to the channel, because of this these traps are likely to be slower than the traps at the substrate interface. This impact is, however, not significant as we observe many traps which are of useful switching speeds.

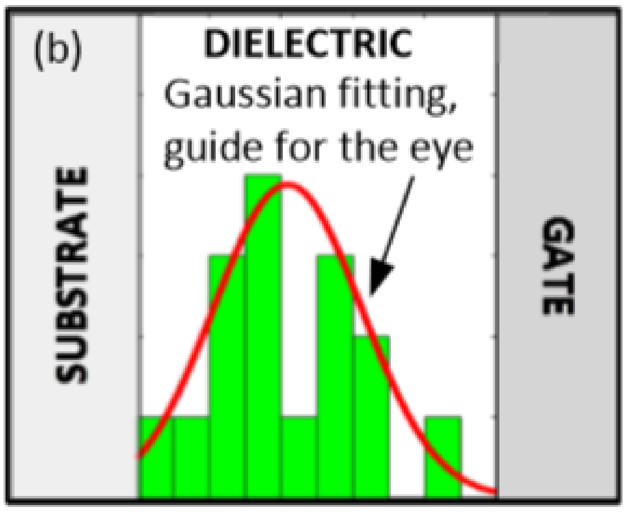


Figure 52, Spatial location of generated traps in the vertical plane.

The final characterisation was to look at the magnitude of the generated traps. To do this the RTN signals were observed, and the top and bottom current levels of the signal were extracted to find the ΔId. Id0 was estimated by extracting the uncaptured level of the RTN signal. The ΔId/Id0 for each trap was found at a range of different voltage levels. Figure 53 shows the results from 4 typical generated RTN defects, where the RTN fluctuation can be as much as 30% of the total current flow, or as little as 6%. The clear trend of all the RTN signals in Figure 53 is that the single trap impact increases as the Vg is reduced, this is not surprising when looking at the impact in terms of ΔId/Id0 as the total current flow through the device is reduced with lower Vg values, but the traps themselves can have the same size impact on the ΔId. There are some practical implications of this Vg dependence of the single trap impact. Firstly, if the RTN signal is extracted at a lower Vg the signal to noise ratio (SNR) will be larger making it easier to extract the signal against other environmental noises that could affect the entropy of the signal while also making it harder for a potential attacker to insert a malicious noise signal to the system. Secondly, it makes the amount of amplification required to extract the signal smaller. This, however, is less important as it will not remove the need for amplification, so that the efficiency saving is not significant.

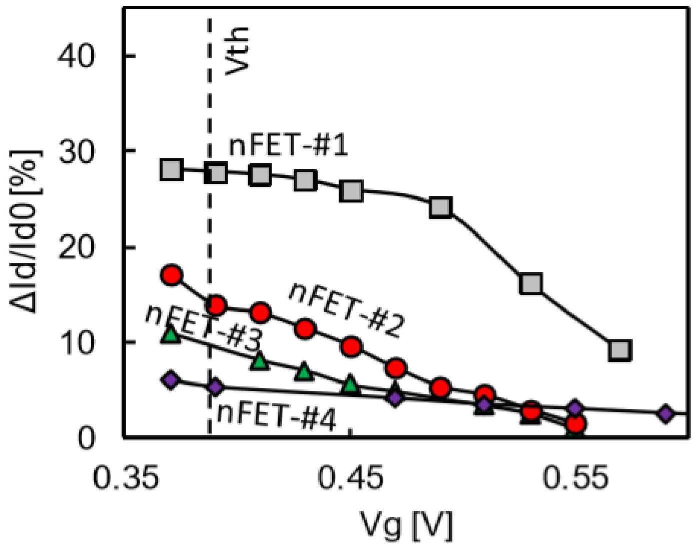


Figure 53, Magnitude of 4 generated RTN traps across a range of Vg values.

4.2.3 Design Implications of Using Generated RTN

This subsection looks at the practical implications of using generated RTN over purely pre-existing RTN as an entropy source for a TRNG. Firstly, we look at how this affects the requirements for the array size used to ensure a useful entropy source is present. After that, we discuss the extra requirements for using generated RTN, when compared with using pre-existing RTN. Some other potential ideas are also explored.

The primary purpose of using generated RTN was to decrease the total area required for the TRNG circuit, achieved by reducing the array size required to guarantee the presence of a good entropy source. Figure 54 shows an analysis of the probability of having a good entropy source, as array size is increased with smaller probabilities of clear RTN being present. In Figure 54 we can see that with our generated RTN rate of 68%, there is a standard normal cumulative distribution (SNCD) (maps percentages to a distribution that has a mean of 0 and a standard deviation of 1) of 6 (equal to a probability of ~99.999998%) when the array size is only 10 devices. If the probability RTN is reduced to 3%, which is closer to the amount of RTN in high quality commercial wafers, then the number of devices required for a similar SNCD is many hundreds of devices.

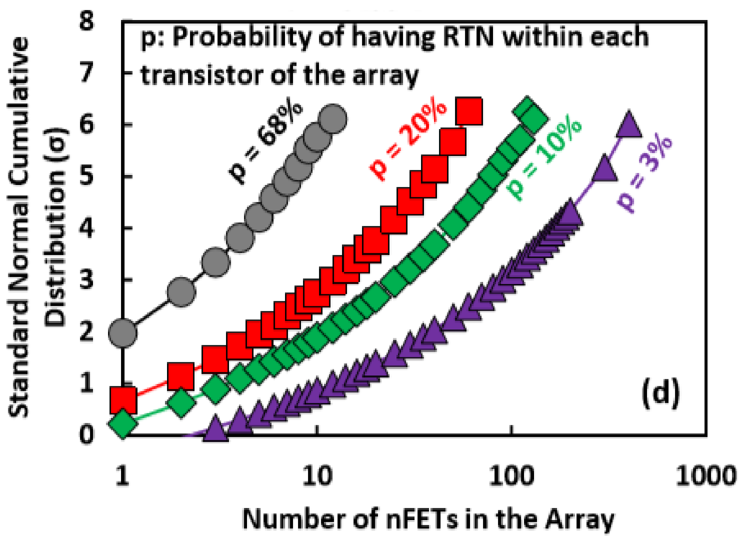


Figure 54, Analysis of required array size the give a high chance of a good entropy source being present with different probabilities of RTN existing within source devices.

These results show that the amount of space required for devices within the transistor array can be reduced by ~97.5% which is significant when the target market for this design is super low power devices which can be 0.05 x 0.05 millimetres or smaller[120]. From this we can say that we have improved upon the issue of device selectivity that faced previous RTN based TRNG designs. In addition, there are more improvements that could be made to further increase this efficiency. For example, entropy source efficiency could be further increased by making multi-level and abnormal RTN usable as entropy sources which we will look at in more detail in section 4.4.2. The main downside to using generated RTN is that it adds an extra step where devices in the array are stressed using the procedure outlined in section 4.2.1. The stressing step will make the manufacture more costly and time consuming and the benefits in the area saving have to be weighed and compared to these extra costs when considering potential applications.

Using heavy stresses may not be the only way to introduce or enhance RTN in devices reliably. It may also be possible to introduce RTN with specific modification to the device design itself, either in the layout of each device or the materials used. For example Lin et al.[121] reduce RTN amplitude by changing the fin height direction in FinFET transistors, we could follow the same process but with the opposite goal of increasing RTN impact by varying the fin height direction. Unfortunately, we do not currently have the required resources to fully explore these ideas and whether they are practical.

In conclusion for this section, the results show that we can greatly increase the chance of there being clear RTN in a device through the use of our generation procedure. We also characterise the behaviour and properties of generated RTN so that it can be fully utilised as a TRNG entropy source. As we can now have a much greater chance of RTN being present we have greatly improved the previous limitation of poor device selectivity that was a problem for previously proposed TRNG designs.

However, this innovation does not solve the issue of poor random number generation speed, so another key innovation had to be made.

4.3 Using AC RTN for TRNGs

As mentioned, the rate of random number generation has been a key limiting factor for previous RTN based TRNGs. In this section we discuss our novel solution to this problem. The basis of this new technique is to use AC RTN technique, inspired by the work of Zou et al. [99], [118], where an AC signal is used on the gate of a device to accelerate the observed τc and τe of an RTN trap. By accelerating the observed τ values we can extract entropy from the RTN at a faster rate, allowing for random numbers to be generated more quickly. The details of the AC RTN measurement procedure were introduced in section 3.4.3, so this section will focus on how using AC RTN measurement can improve the efficiency and performance of an RTN based TRNG.

Traditionally RTN based TRNGs have used a DC measurement technique where constant Vg is used [37], [48], [53], [60]. However, using this measurement technique, the maximum output bit rate is limited and is one of the key weaknesses of this type of design. The reason for this was illustrated in Figure 48, it can be seen that as the value for Vg is adjusted to reduce either τc or τe then the other will increase meaning that one of them will always be the bottleneck for the speed. To solve this bottleneck, this work proposes using AC RTN which allows us to control τc and τe independently. For AC RTN, a square wave signal is applied as Vg rather than the traditional constant voltage signal. This allows us to control the τe with the low part of the AC signal (VgL) and the τc with the high part of the AC signal (VgH).

Figure 55 shows a comparison in how our AC technique and the traditional DC technique interact differently with the τc and τe of an RTN trap. With DC RTN measurement, only a single Vg value is used, which means that the observed τc and τe are linked, introducing the problem of them having inverse relationships with Vg. The AC RTN measurement allows us to use VgL and VgH to separate the observed τc and τe so long as the Fac frequency is sufficiently high (see section 3.4.3). The results in Figure 55 show that the difference in these observed values can be many orders of magnitude when switching from DC to AC techniques. The degree to which an RTN trap can be accelerated using this method is proportional to the slopes of τc and τe against Vg. Ideally, the gradient of τe should be as high as possible and the gradient of τc should be as negative as possible for the most acceleration.

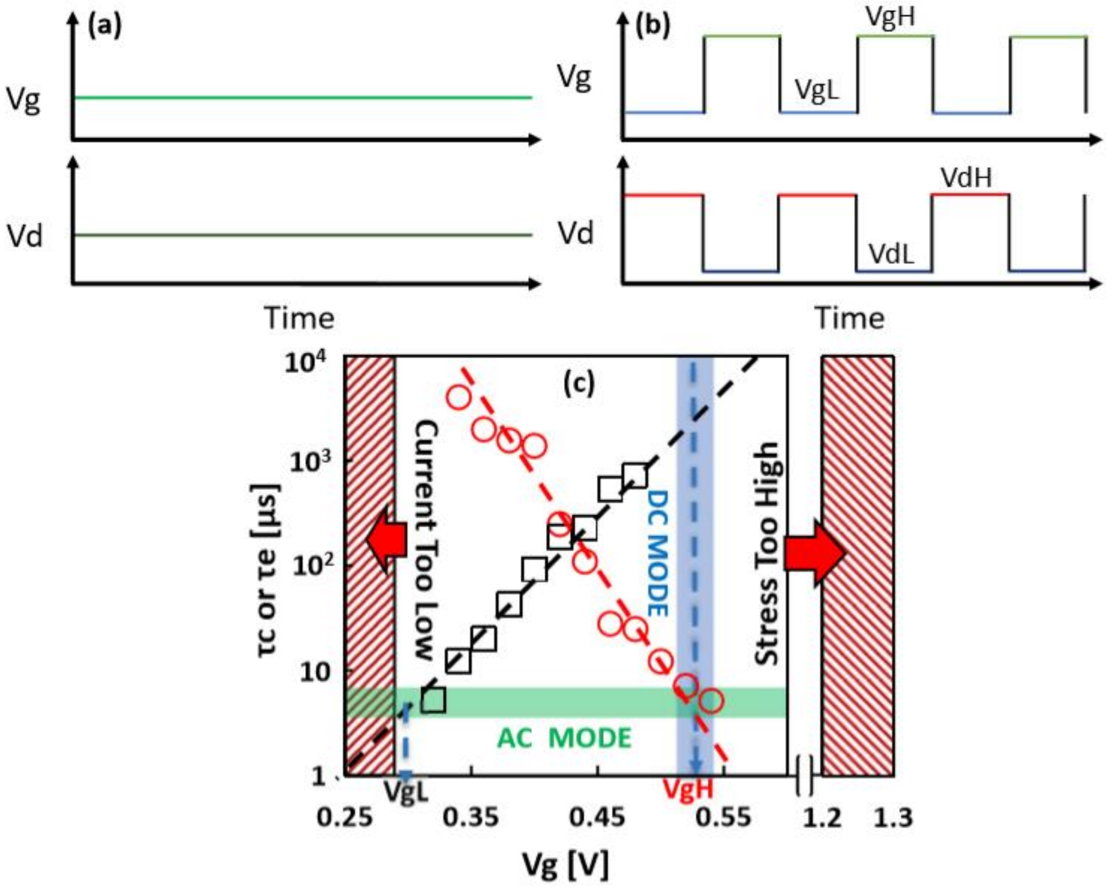


Figure 55, Diagram of (a) DC RTN measurement technique (b) AC RTN measurement technique and (c) a comparison of DC and AC measurement techniques' effect on observed τ values.

To further illustrate the increased rate at which RTN events can be observed using the AC RTN, by increasing VgH, one trap's observed τ values were extracted under different gate voltage conditions, the results of this process are shown in Figure 56. The results clearly show that when using the AC RTN technique, the greater the difference between VgL and VgH the smaller τ will be.

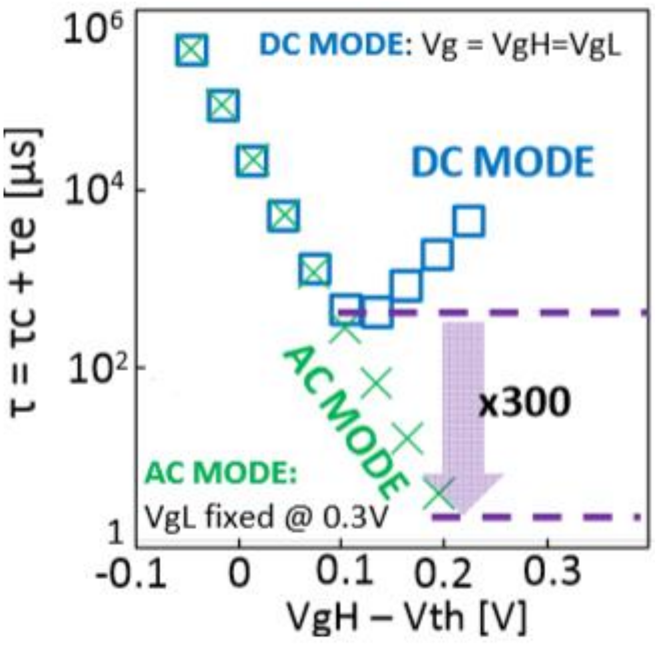


Figure 56, Illustration of the reduction in observed τ values when using AC measurement method compared to DC.

To explore this concept in more detail, some of the raw RTN signals were plotted with a range of Vg values for the DC RTN results and a range of VgH values, with fixed VgL, for the AC RTN results. Figure 57 clearly shows that for the DC results with 0.53Vg τc is longer than τe. As the Vg is increased τc becomes shorter but τe becomes longer. Looking at the AC results, 0.53VgH has a longer τc than τe but when VgH is increased τc becomes smaller but τe remains constant, which shows how the τ values have been de-coupled. It is clear that the AC result with a 0.68VgH has many more RTN events than any of the DC results, clearly demonstrating the rate of acceleration of using an AC RTN signal.

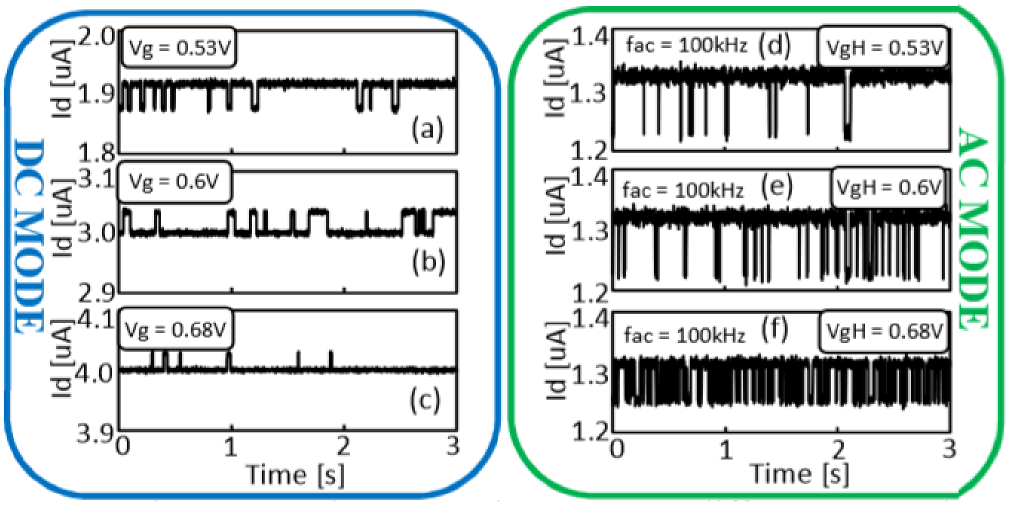


Figure 57, DC and AC comparison with different VgH and AC frequencies and a fixed VgL of 0.28V.

One major issue with using AC RTN measurement is that half of the time the Id signal is not measured. This is because measurement is only performed at VgL and not VgH. While this does offer some benefits, such as lower power consumption and taking advantage of the larger single trap impact at lower Vg values, losing half the signal needs to be considered. One possible solution is to use two devices as entropy sources and apply the AC RTN technique to each, but 180˚ out of phase with each other so that when VgH is applied to one, VgL is applied to the other. Using these 2 inverse entropy sources allows for continuous Id signal to be used for random number generation.

One further improvement that can be added to the AC RTN measurement is to utilise the Vd dependence of RTN traps that was discussed in section 4.2.2. In that section it was shown that increasing Vd can shift the τ cross so that the observed τe value will be lower at any given Vg value than for the same Vg with a lower Vd. This concept is illustrated in Figure 58. If this Vd effect is combined with the Vg effect so that Vd is increased at VgL and decreased at VgH, it is possible to further reduce the observed τ values significantly. Using the trap shown in Figure 58 as an example, both τc and τe can be accelerated by an order of magnitude by utilising an AC Vd and Vg signal. The specific voltage levels for VdH and VdL can be selected so that VdH is the highest voltage possible without over stressing the device (0.3V usually for this work) and VdL is 0V so that power consumption is minimised, and acceleration maximised.

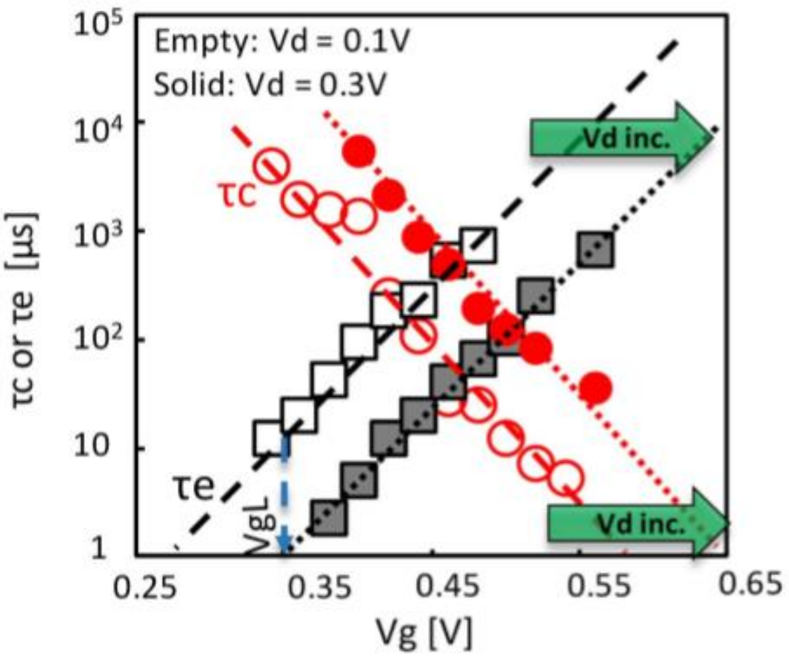


Figure 58, Illustration of how adjusting Vd can be used to decrease observed τ values.

In conclusion for this section, the results have shown that our second key innovation of using a new AC RTN technique has allowed us to greatly increase the speed at which RTN events can be observed, thereby increasing the speed at which entropy can be extracted to generate random numbers. It has also been shown how using AC signals for both Vg and Vd can further accelerate the speed.

4.4 - TRNG Circuit Design

In this section we look at the circuits that are used to convert the entropy from RTN into a random number sequence. We first look at a basic circuit built up from conventional RTN TRNG designs and analyse the randomness of the output bits, the performance, and limitations of the circuit. We then research into a novel circuit design for extracting entropy from RTN and converting it to random numbers, to improve TRNG performance and efficiency at which entropy is extracted from a transistor with RTN. As with the first circuit we also evaluate the performance and limitations of the new TRNG circuit and analyse the randomness of the output bitstream. To analyse the randomness of these circuits we utilise both the NIST test suite and the DIEHARD battery of tests as well as the emerging technique of using neural networks to try and crack TRNGs.

4.4.1 Basic TRNG Circuit

Here we look at the first circuit design that was created for the TRNG developed in this project. It uses a lot of elements from existing TRNGs together with our generated and AC accelerated RTN as the entropy source. Figure 59 shows that this circuit follows the traditional principles of amplification, digitisation, de-biasing, and sampling, to convert the raw RTN entropy to a random bitstream. In this circuit design we use two entropy sources in parallel with two AC voltage supplies 180˚ out of phase in order to avoid the problem of AC RTN entropy sources only measuring half of the AC cycle. The entropy will be extracted from the transistor currently supplied with VgH and VdL.

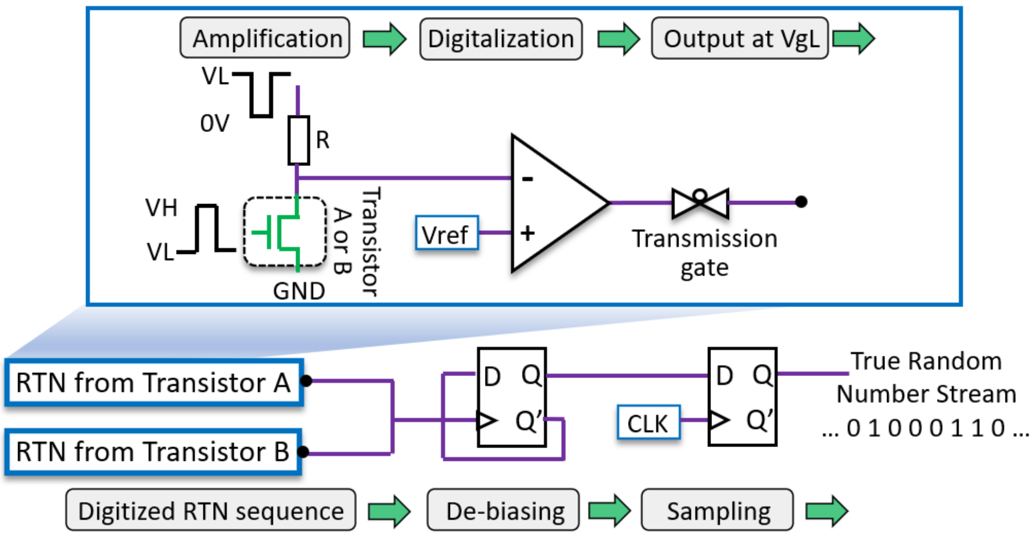


Figure 59, Basic RTN based TRNG circuit design.

The performance of this design was verified by simulation. Using simulation allowed for results to be gained quickly with limited facilities and time available for this project. The first stage of our simulations was to import RTN data from our measurements. A simulated RTN signal was also created with parameters extracted from RTN measurement, rather than directly taken from measurement. This was done because it was necessary to have RTN data over exceptionally long time periods (>100s) to generate enough random numbers to verify the randomness. The memory and processing time required to use raw measurement data over time periods this long makes it unfeasible when using a simulation-based processing circuit. This limitation is present because we need to maintain a high sampling rate of the Id current to be able to extract the RTN signal and this high sampling rate can only be maintained for a few seconds before filling the memory of the B1500A. As the processing circuit is simulated, all of the RTN data needs to be saved and then imported to the circuit simulation. For the randomness analysis that only requires a small number of bits, the raw data is used, for the tests that require very large numbers of bits, the simulated RTN signal is used.

The RTN simulation followed the technique proposed by Puglisi et al. [105]. The key parameters of τc, τe, and amplitude were extracted from the results of AC RTN measurements. The extracted parameters were then used as inputs for the simulations to give an RTN signal which has the same timing and amplitude characteristics as the measured signal but over a much longer period. Figure 60 shows a comparison of our simulated RTN signal, as well as the amplified and digitised version of an RTN measurement using a simulation of the circuit shown in Figure 59. Figure 60 shows that we can accurately simulate the timing characteristics of a measured RTN signal.

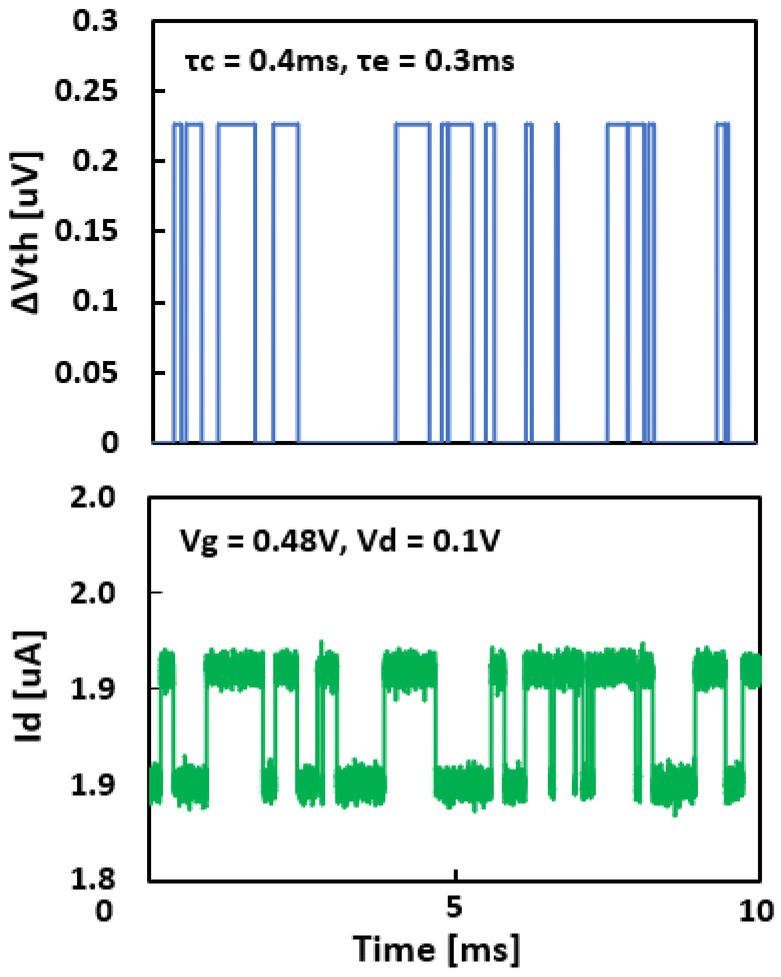


Figure 60, Comparison of simulated (top) and measured (bottom) RTN signals with the same time constants.

With the simulated RTN signal now accurately representing the amplified RTN signal, the next stage is the de-biasing section of the circuit. De-biasing is an essential step to ensure that the average time spent at ‘high’ and ‘low’ is the same. It is important that the average time spent at ‘high’ and ‘low’ is the same so that when the signal is sampled at regular intervals, the chances of the output being a ‘1’ or a ‘0’ are 50%, which is a key feature for any random sequence. Traditionally de-biasing is done using a post-processing circuit, such as a Von-Neumann corrector, as explored in section 2.1.2, however Chen et al. [60] proposed a new de-biasing technique which can be much more efficient. The principles of Chen’s proposed technique are shown in Figure 61. Chen’s method was implemented into our simulations to achieve the desired output signal for the de-biasing section of the circuit.

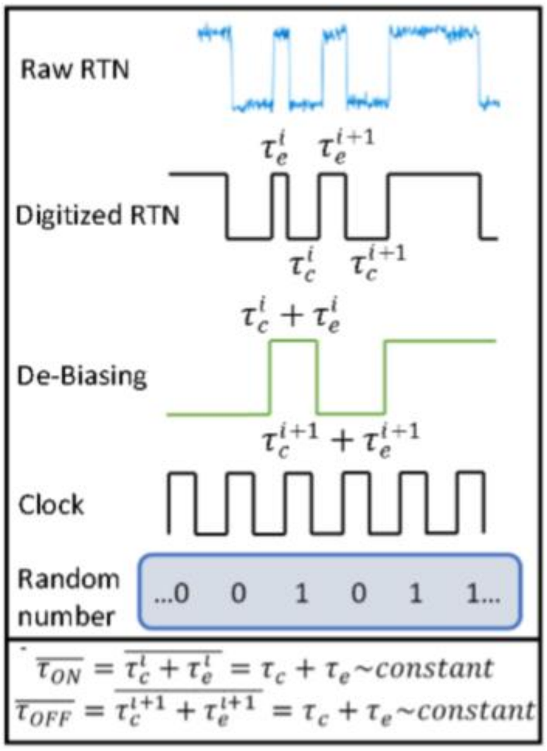
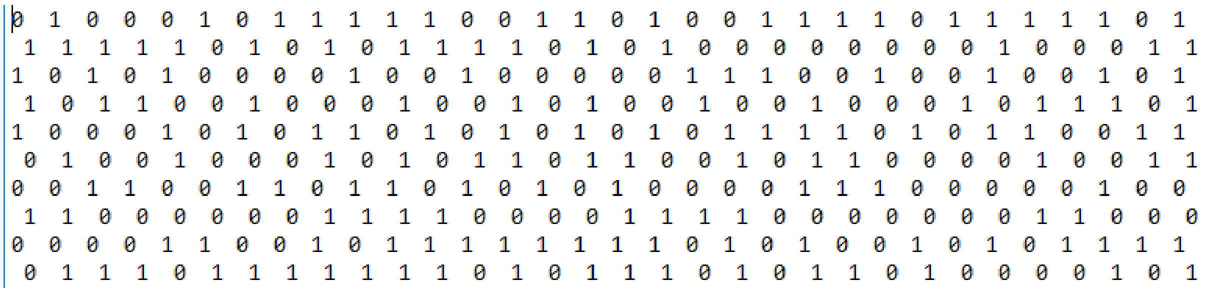

Raw RTN
τ_e^i
τ_e^{i+1}
Digitized RTN
τ_c^i
τ_c^{i+1}
$\tau_c^i + \tau_e^i$
De-Biasing
$\tau_c^{i+1} + \tau_e^{i+1}$
Clock
Random number
...0 0 1 0 1 1...
$\overline{\tau_{ON}} = \overline{\tau_c^i + \tau_e^i} = \tau_c + \tau_e \sim constant$
$\overline{\tau_{OFF}} = \overline{\tau_c^{i+1} + \tau_e^{i+1}} = \tau_c + \tau_e \sim constant$

Figure 61, Chen's proposed de-biasing method [60].

The basic principle of Chen’s de-biasing technique is to use the digitised RTN signal as the input and to toggle the output between high and low every time a rising edge is detected on the input. This principle works because an imbalance between time at ‘high’ and time at ‘low’ is caused by a difference between τc and τe. By only toggling the output on a rising edge from the input, the time average between switching events becomes the sum of τc and τe which will then balance the time at 'high' and 'low'. The obvious downside to this de-biasing scheme is that the time between switching events becomes longer, therefore reducing the rate at which the signal can be sampled without having long chains of a single bit. Another downside to this de-biasing scheme is that the sampling rate has to be closely matched to the output de-biased signal to avoid oversampling while maintaining the highest bit rate possible, which requires some tuning circuits. The advantages it offers over traditional post-processing circuits, however, are significant. Firstly, it is a much simpler circuit, only requiring a single flip-flop, massively saving on design footprint and efficiency. This de-biasing also has much better bit-efficiency than Von Neumann post-processing as every sampled bit can be used in the output bitstream, whereas Von Neumann has to discard many generated bits to maintain high entropy.

With the de-biased RTN waveform gained the last step is to sample the waveform to obtain the '1' and '0' output bits. A sampling clock in the form of the 'clock' signal in Figure 61 is used to sample the de-biased RTN signal on every rising edge of the clock signal. The sampling rate has to be carefully selected as a sampling rate that is too fast will cause long chains of '1s' or '0s' which reduces the output entropy and causes the bitstream to fail the randomness tests. Figure 62 shows a sample of some random bits that were generated with a sampling rate of 2Msa/s.



```
0 1 0 0 0 1 0 1 1 1 1 1 0 0 1 1 0 1 0 0 1 1 1 1 0 1 1 1 1 1 0 1
 1 1 1 1 1 0 1 0 1 0 1 1 1 1 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 1
1 0 1 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 0 0 1 0 0 1 0 0 1 0 1
 1 0 1 1 0 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1
1 0 0 0 1 0 1 0 1 1 0 1 0 1 0 1 0 1 0 1 1 1 1 0 1 0 1 1 0 0 1 1
 0 1 0 0 1 0 0 0 1 0 1 0 1 1 0 1 1 0 0 1 0 1 1 0 0 0 0 1 0 0 1 1
0 0 1 1 0 0 1 1 0 1 1 0 1 0 1 0 1 0 0 0 0 1 1 1 0 0 0 0 0 1 0 0
 1 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 0 0 0 1 1 0 0 0
0 0 0 0 1 1 0 0 1 0 1 1 1 1 1 1 1 1 0 1 0 1 0 0 1 0 1 0 1 1 1 1
 0 1 1 1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 0 1 1 0 1 0 0 0 0 1 0 1
```

Figure 62, Sample of random bitstream, sampled at a rate of 2MSa/s.

In order to verify that the generated bits have a high degree of randomness they were tested using the NIST test suite, which was discussed in section 2.1.1. For the NIST tests at least 5 million bits should be generated for the results of the tests to be reliable. Using the settings defined in the user guide [33] for the parameters of each NIST test is important to gain reliable results. The settings for the NIST test suite used in this work are as follows:

NIST Test Parameters

Significance value (α) = 0.01

Frequency monobit test – sequence length (n) = 100, number of sequences = 100

Frequency within a block test – sequence length (n) = 100, block size (M) = 20, number of sequences = 100

Runs test - sequence length (n) = 100, number of sequences = 100

Test for the Longest Run of Ones in a Block – sequence length (n) = 128, length of each block (M) = 8, number of sequences = 100

Binary matrix rank test – sequence length (n) = 38,912, number of sequences = 100

Discrete Fourier Transform (Spectral) Test – sequence length (n) = 1,000, number of sequences = 50

Non-overlapping Template Matching Test - bits of each template (m) = 9, length of the entire bit string under test (n) = 1,000

Overlapping Template Matching Test - sequence length (n) = 1,000,000, number of sequences = 10

Maurer's "Universal Statistical" Test - sequence length (n) = 387,840, number of sequences = 10

Linear Complexity Test - sequence length (n) = 1,000,000, block size (M) = 500

Serial Test - sequence length (n) = 1,000, length in bits of each block (m) = 7, number of sequences = 100

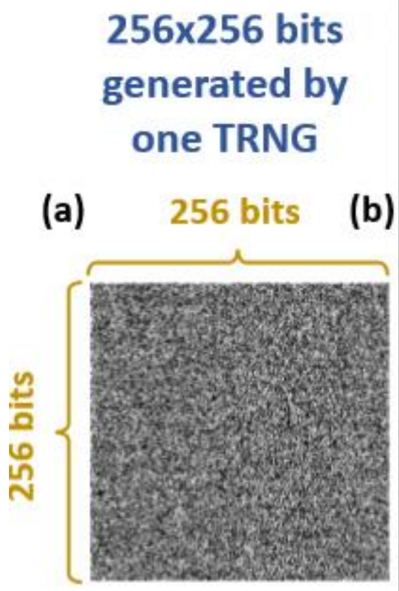
Approximate Entropy Test - sequence length (n) = 1,000, length in bits of each block (m) = 4, number of sequences = 100

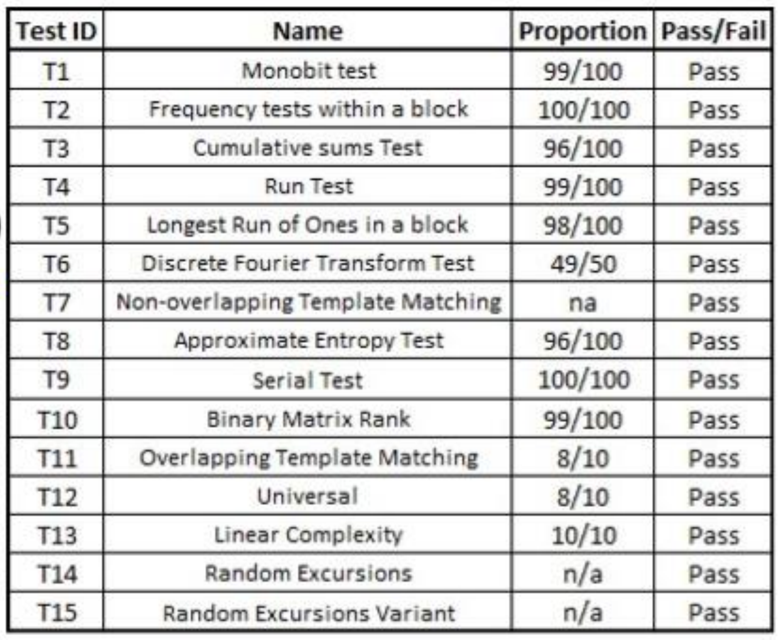
Cumulative Sums (Cumsum) Test - sequence length (n) = 1,000, number of sequences = 100

Random Excursions Test - sequence length (n) = 1,000,000

Random Excursions Variant Test - sequence length (n) = 1,000,000

Figure 63(a) shows a bitmap of some of the TRNG results from the measured RTN data. Bitmaps make it easier to visualise the randomness of the results, as no clear patterns or large blocks of a single colour can be seen. Figure 63(b) shows the results from the NIST tests which must be passed if the bitstream is to be declared as random. Tests T1 to T9 could be performed with measured RTN data, the other tests required simulated RTN to generate enough bits to complete the test. As the figure shows, the proposed TRNG design passes all of the NIST tests under the applied sampling rate of 2MSa/s. The tests which require many repeated tests show the proportion of bitstreams that pass as well as the overall pass/fail status. The tests that perform a single test on one bitstream only show the pass/fail result of that one test.




(b)

Test ID	Name	Proportion	Pass/Fail
T1	Monobit test	99/100	Pass
T2	Frequency tests within a block	100/100	Pass
T3	Cumulative sums Test	96/100	Pass
T4	Run Test	99/100	Pass
T5	Longest Run of Ones in a block	98/100	Pass
T6	Discrete Fourier Transform Test	49/50	Pass
T7	Non-overlapping Template Matching	na	Pass
T8	Approximate Entropy Test	96/100	Pass
T9	Serial Test	100/100	Pass
T10	Binary Matrix Rank	99/100	Pass
T11	Overlapping Template Matching	8/10	Pass
T12	Universal	8/10	Pass
T13	Linear Complexity	10/10	Pass
T14	Random Excursions	n/a	Pass
T15	Random Excursions Variant	n/a	Pass

Figure 63, (a)Bitmap of random bitstream created by proposed TRNG and (b)NIST test results of bitstream generated by proposed TRNG with a significance value of 0.01. Results from one TRNG instance operating with a sampling rate of 2MSa/s. Tests T1-T9 use measured RTN data, the others use simulated RTN data.

The sampling frequency(fsp) plays a huge role in the output entropy of the TRNG. Oversampling under high fsp lowers the randomness significantly as it will result in long chains of '0' or '1' if the RTN is sampled multiple times before changing state, as is discussed by Mohanty et al [37]. To explore the effects of fsp on the proposed TRNG design, we simulate an AC RTN entropy source with τc=τe=2μs at the fac larger than fac_min. Different fsp values are used to generate random numbers, which are evaluated through the NIST test suite. Figure 64 shows the results of this process. When fsp is higher than a specific value, which we term fsp_max, the TRNG fails some or all of the NIST tests. To explore the impact of fsp_max on different entropy sources, which have different trap properties, we randomly generate τc and τe values and then determine the fsp_max for each trap by performing the NIST tests on each result. Figure 65 shows the results from this experiment. In Figure 65 fsp_max is plotted against τc + τe, a clear power law correlation can be observed between these two parameters. For example, a maximum random bit generation speed of 1MHz can be achieved if the RTN entropy source has τc + τe = 2μs. For each entropy source, by selecting the smallest possible τc and τe with VgH, VgL and operating under the suitable fac and fsp, the maximum possible speed of the TRNG based on this entropy source can be achieved. Figure 65 shows that fsp_max can be estimated by using: fsp_max = 1/(τc+τe).

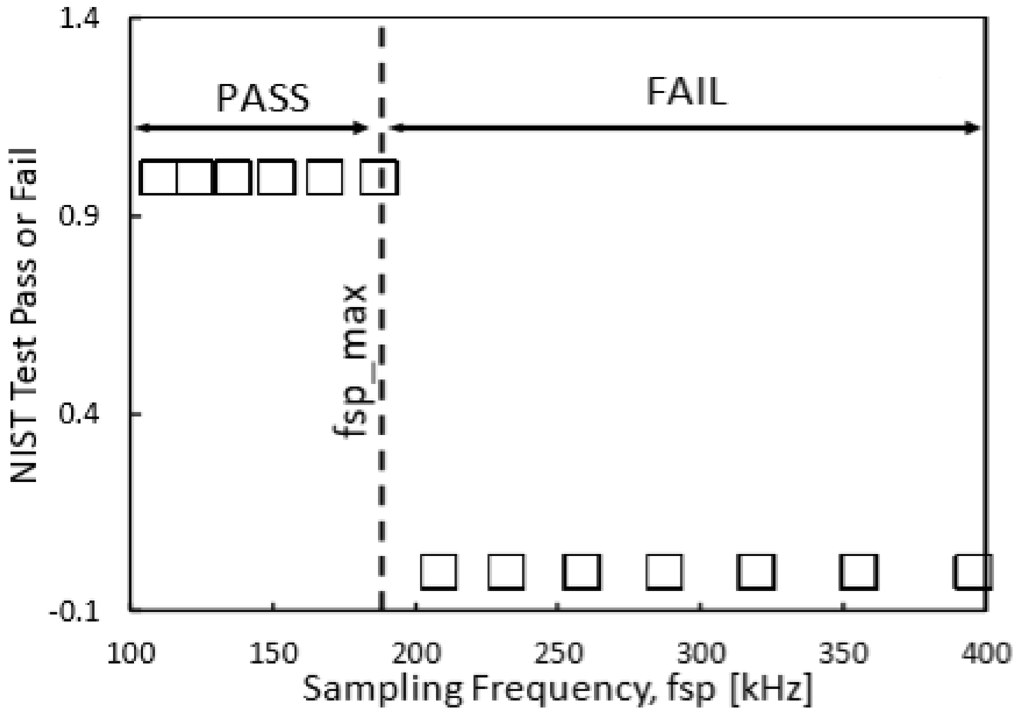


Figure 64, NIST test results on the generated bitstream of a single TRNG instance under different sampling frequencies (fsp).

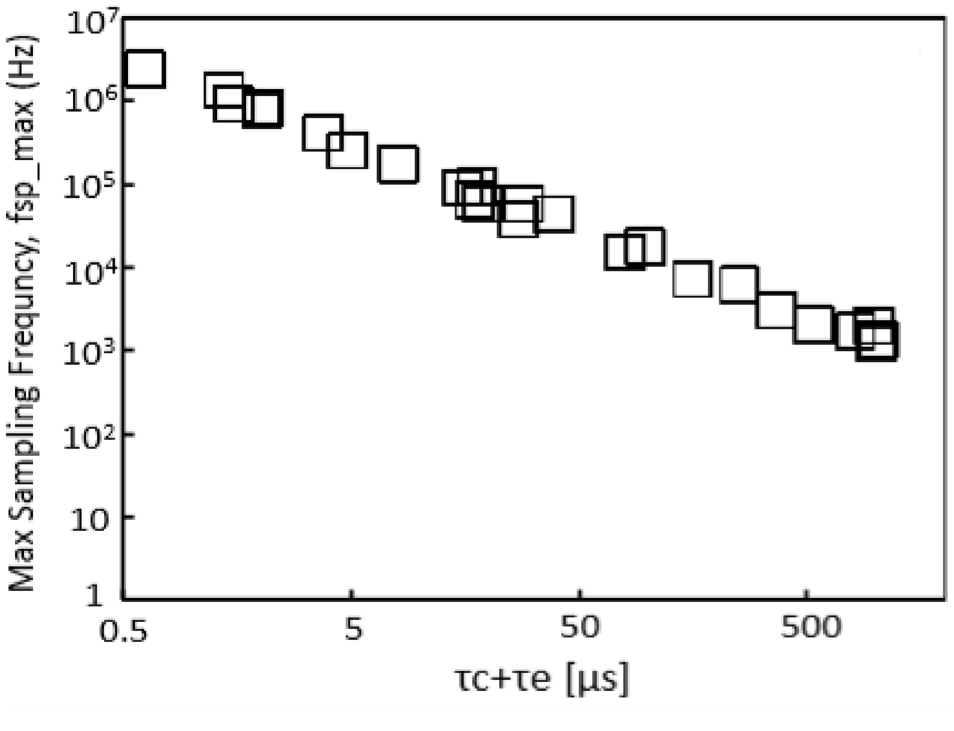


Figure 65, Fsp_max for TRNGs with RTN entropy sources of different τc+τe values.

To find the optimal sampling rate to be applied across multiple TRNGs, a yield analysis was performed. 20 TRNGs were simulated and a range of sampling frequencies was applied. The number of TRNGs that passed the NIST tests at each sampling value were then counted. Figure 66 shows the analysis of these 20 simulated TRNGs, based on different measured traps. The results show that currently just under 20% of devices will pass the NIST tests at a sampling rate of 5Mbps and 40% at 3Mbps, under the current limitations. 90% of devices can function under a sampling rate of 1Mbps which is more than what is needed by some low power applications such as, internet secure session links, car keys, or ID cards [45].

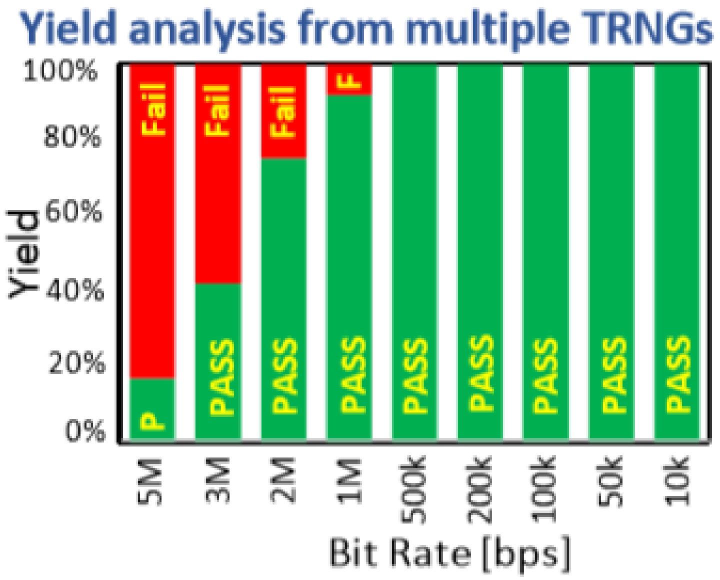


Figure 66, Percentage of devices (20 total) that pass the NIST tests at a given sampling frequency.

Next, we will discuss the limitations of this TRNG design and potential improvement. The first limitation is this design's ability to extract entropy from devices which have more than 1 active RTN trap, resulting in multi-level RTN. In this design a simple thresholding technique can be used to allow for multi-level RTN to be used as an entropy source. The thresholding technique was proposed by Mohanty et al. [37] and is illustrated in Figure 67. A threshold is selected and any value above it is considered as 'high' and any value below it is considered as 'low'. This allows for multi-level RTN to be used as an entropy source; but it limits the operation of the TRNG to the speed of the slowest RTN trap. Only being able to extract entropy from the slowest trap is clearly not ideal as a significant amount of entropy is lost which could be used to increase the operational speed of the TRNG.

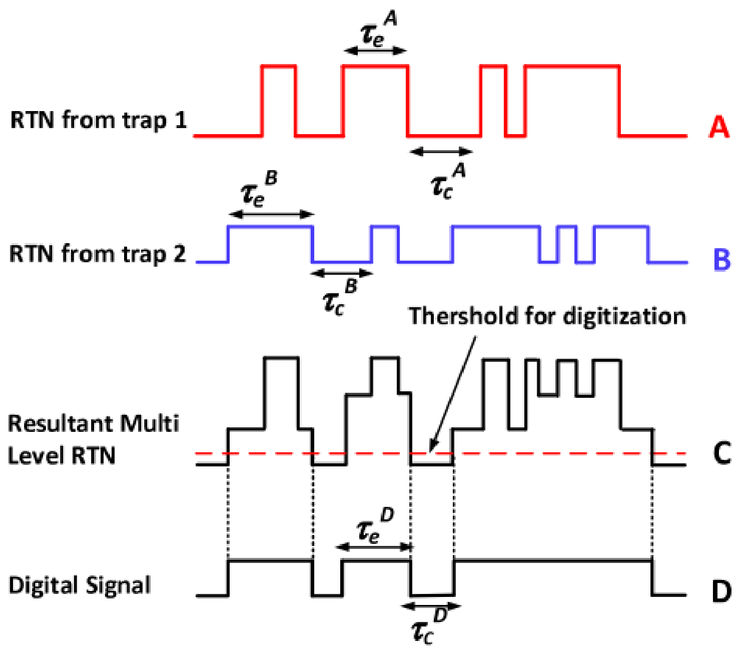


Figure 67, Threshold technique for extracting entropy from multi-level RTN, as proposed and illustrated by Mohanty et al [37].

Another key limitation of this TRNG design is its ability to deal with abnormal RTN where there can be ‘quiet’ phases, which is when it appears that the RTN has disappeared for some time and will then re-appear after a random delay. The obvious problem is that, even when the RTN temporarily disappears, the TRNG continues to sample the signal at the same rate, causing oversampling and a long chain of either ‘0’ or ‘1’. No previous RTN based TRNGs have considered this problem even though it could be significant as abnormal RTN is fairly common and could be mistaken for normal RTN if the time window for observing it is too small. As most RTN measurements are currently performed with short time windows it is highly possible that abnormal RTN is a bigger issue than currently known. Finally, this type of RTN based TRNG can be susceptible to oversampling attacks where an extra signal is injected into the sampling clock in order to effectively increase the sampling rate. If the sampling rate is increased through this attack it can cause long chains of a single output which reduce the output entropy and make the system easier to predict.

In conclusion for this section, a novel RTN based TRNG circuit has been proposed that improves upon the selectivity and generation rate. These improvements are achieved through a novel RTN generation scheme and a new use of AC RTN measurement. The proposed design's key parameters are outlined, and the randomness is verified using the industry standard NIST test suite. This section also discussed how the proposed TRNG still has limitations when trying to extract entropy from RTN sources which are not ideal two-level RTN, as well as issues with resistance to oversampling attacks.

4.4.2 Improved RTN Based TRNG Circuit

In this section we look at the details of the improved RTN based TRNG circuit. First, the problems that this new circuit hopes to solve are looked at in more detail. Next, we develop a circuit that converts all RTN edges to pulses which can be used as a sampling signal for generating random numbers. This is followed by looking at the FPGA-based circuit for converting the RTN edge pulses to random numbers. Finally, the performance of the improved TRNG circuit is compared with the most prominent existing TRNG designs.

In the previous RTN TRNG explored in this work, only two-level RTN could be extracted effectively. This type of RTN is shown in Figure 68a where a constant gate (Vg) and drain (Vd) voltage are applied and two clear levels can be observed in the measured drain current (Id). However, it is often the case that more than one trap is present in a given device, resulting in an RTN signal with more than 2 visible levels. This is referred to as multi-level RTN [122] and one example is given in Figure 68.b, where we can observe 4 clear levels in the Id. When 4 clear levels can be observed, there are 2 active traps influencing the channel current. It is possible to have more traps active, although in nano-scale devices it is rare to have more than a few clearly discernible traps. The number of traps present is proportional to device area and the single trap impact is inversely proportional to it [113], [116]. It is also observed that an RTN trap may only be active when a dominating trap is in a certain state [123]. This can result in Id measurements similar to Figure 68.c where there is clearly a slower trap and a faster trap, but the faster trap is only active when the slow trap is in the empty state. This third type of trap is what we refer to as abnormal RTN and shows the 'quiet' phases of the RTN signal which would cause issues for previously proposed RTN TRNGs.

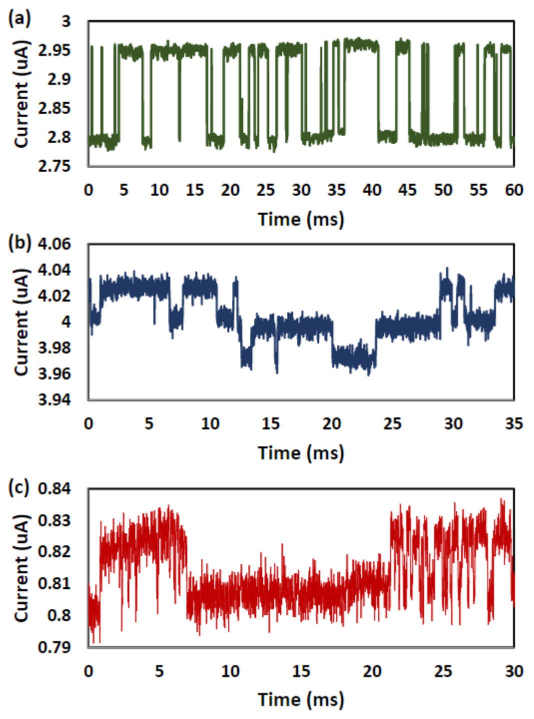


Figure 68, Major different types of RTN observed in measurement of individual transistors of the same design and size. (a) An example of two-level RTN, the most well studied type of RTN utilised by traditional RTN based TRNGs (b) An example of multi-level RTN, a more complex RTN signal which cannot be used by traditional RTN based TRNGs but can be used by our newly presented TRNG design. (c) An example of abnormal RTN, the most complex type of RTN discussed here. Abnormal RTN could not be used in traditional designs and could cause major issues such as oversampling.

It was found that only 81% of traps displayed single level RTN, with the rest showing multi-level RTN. 19% of traps having clear multi-level behaviour is a significant amount that would not be efficiently extracted using traditional techniques for RTN-based TRNGs. It is also possible that this is an underestimation of the number of traps that show multi-level behaviour. As a very high sampling rate has to be used by the WGFMUs to capture the RTN trapping, a very short time window (<1s) has to be used which means that it is possible that a slower trap could be missed which would destroy the performance of a traditional RTN TRNG. This is also the reason we do not show the percentage of traps that are abnormal as it is impossible to accurately predict this with the time windows available using the measurement techniques discussed in this work. To avoid this problem the improved circuit design can utilise any of the types of RTN.

TRNG design utilising multi-level RTN. The TRNG presented in this work has 3 major parts. The first part is the entropy source which is a single nano-scale transistor with a constant Vg and Vd applied to it, whose Id current is amplified so that it can be used as an input to the edge-to-pulse circuit. The second major section of the TRNG circuit is the edge-to-pulse circuit which is used to perform the analogue to digital conversion where the amplified RTN signal is converted to digital pulses that can be used as an input to the processing circuit. The final part of the circuit is the processing circuit, which converts the digital pulses to random numbers through the sampling of a high frequency oscillator. The processing circuit also acts as a bridge between the asynchronous random number generation and any synchronous circuit it is attached to.

Edge-to-Pulse circuit. The edge-to-pulse circuit takes an amplified RTN signal and converts the trapping and de-trapping events into short pulses. These short pulses can then be used by the processing circuit to generate random numbers based upon the timing interval of these events. The edge-to-pulse circuit consists of 3 sub-circuits.

The first of these sub-circuits is a differentiator that converts the RTN trapping and de-trapping events into positive and negative voltage spikes respectively as seen in Figure 69a-b. It is important that the differentiator circuit is biased so that no edges give a constant output voltage of 0. This 0V no edge output is set so that differentiated de-trapping events will give a negative voltage spike which is required for the next absolute value sub-circuit. In our proof of concept circuit, this 0V bias was achieved with a variable resistor connected to the non-inverting terminal supplied with -5V. In a commercial product this would be replaced with a fixed resistor as the expected Id current from the entropy source for the given Vg and Vd would be known.

The second sub-circuit is the absolute value circuit seen in Figure 69c-d. It converts the negative spikes from the differentiated de-trapping events into positive spikes so that they can be digitised by the same comparator. The gain of the absolute value circuit can be greater than 1 if required, but in our experimental case it is left at unity as no further amplification was required for the spikes to be of sufficient magnitude as input for the comparator.

The final sub-circuit shown in Figure 69e-f is a comparator which is used to convert the analogue spikes from the absolute value circuit to digital pulses which can be used as an input to the processing section of the TRNG. Figure 70 shows the output with a multi-level RTN input. The comparator is set to give a 0V or 3V output for low and high levels which is required by the I/O of the FPGA used for the processing section. The reference voltage for the comparator input is set to be slightly higher than the background noise of the absolute value circuit so that all RTN event spikes will be digitised. The complete breadboard implementation of the edge-to-pulse circuit is shown in Figure 71.

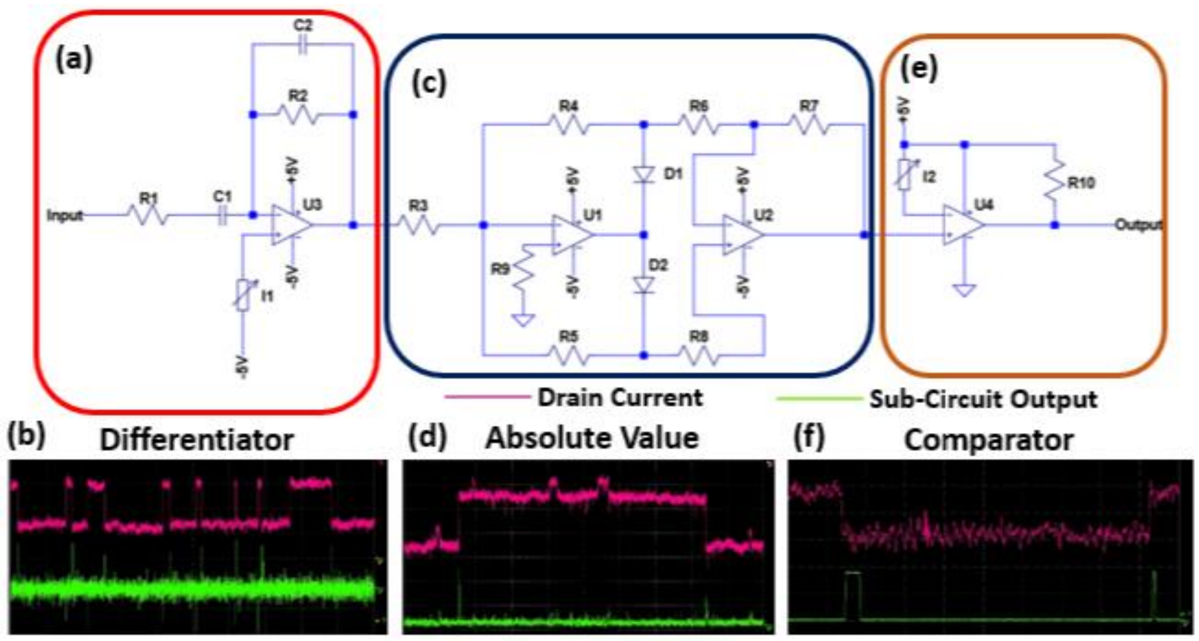


Figure 69, Edge-to-pulse circuit diagram and example outputs. (a) Circuit diagram of differentiator used to convert RTN edges to voltage spikes. (b) Example output signal from differentiator for the given input RTN signal, edges converted to positive and negative spikes. (c) Circuit diagram of absolute value circuit used to convert positive and negative differentiator spikes to all positive spikes. (d) Example output signal from absolute value circuit for the given RTN input. All voltage spikes are now positive. (e) Circuit diagram of comparator used to convert voltage spikes from absolute value circuit to digital pulses that can be used as input to the FPGA. (f) Example output from comparator circuit for given RTN input, spike has been converted to square pulse.

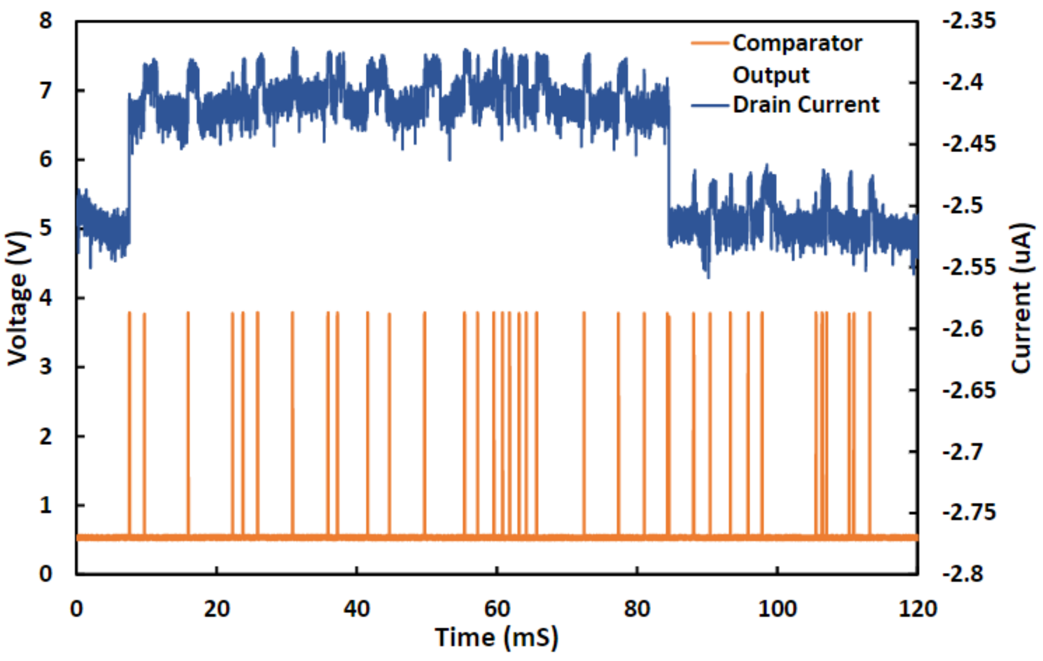


Figure 70, Output of comparator for given RTN input signal to edge-to-pulse module. Output of comparator has a magnitude of 3V making it a suitable input for the FPGA. The comparator output is triggered on both the edges of the large slower trap and the smaller faster trap, showing our design can make use of multi-level RTN. Some of the edges of the small trap can be missed, however, as the magnitude of the trap is close to the background noise.

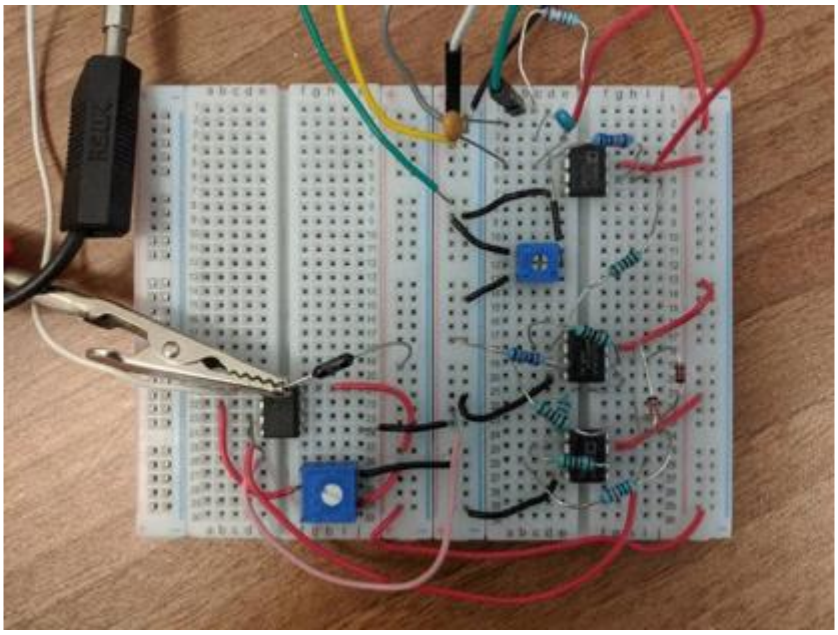


Figure 71, Photograph of Edge-to-pulse circuit, implemented on breadboard, including differentiator, absolute value, and comparator circuits. Amplified RTN signal is used as input to this circuit and the comparator output is sent to processing FPGA.

For the proof of concept circuit, supply voltages were applied with Keysight E3632A DC power supplies, Id current with RTN events was amplified using a Femto DHPCA-100 amplifier, and a Xilinx Spartan 3 FPGA development board was used. The entropy source device was probed on a sample silicon wafer using a Cascade Summit probe station. FPGA programming was performed in the ISE design suite using a combination of custom and conventional VHDL blocks assembled in the visual programming environment. Videos of the TRNG proof of concept circuit in operation are available.

Processing Circuit implemented on an FPGA. The second major part of our TRNG design is the processing circuit where the digitised pulses from the edge-to-pulse circuit are used to generate random numbers, the complete block diagram of this part of the TRNG is shown in Figure 72. The processing circuit itself can be split into 3 sub-circuits. The first two sub-circuits for sampling and data buffering are essential for random number generation. The third sub-circuit is used for transmission of the generated numbers to an external analysis PC which is used for result verification. To perform the data transmission, the RS232 UART communication on the FPGA development board was used. The requirements of the RS232 communication governed the selection of the 8-bit words of random numbers that were created in the first 2 sub-circuits, as well as the maximum data transfer rate of 19200 bps, which is limited by the maximum clock speed of the FPGA. Without the limitations of the RS232 communication these variables could be changed to fit the requirements of the given application.

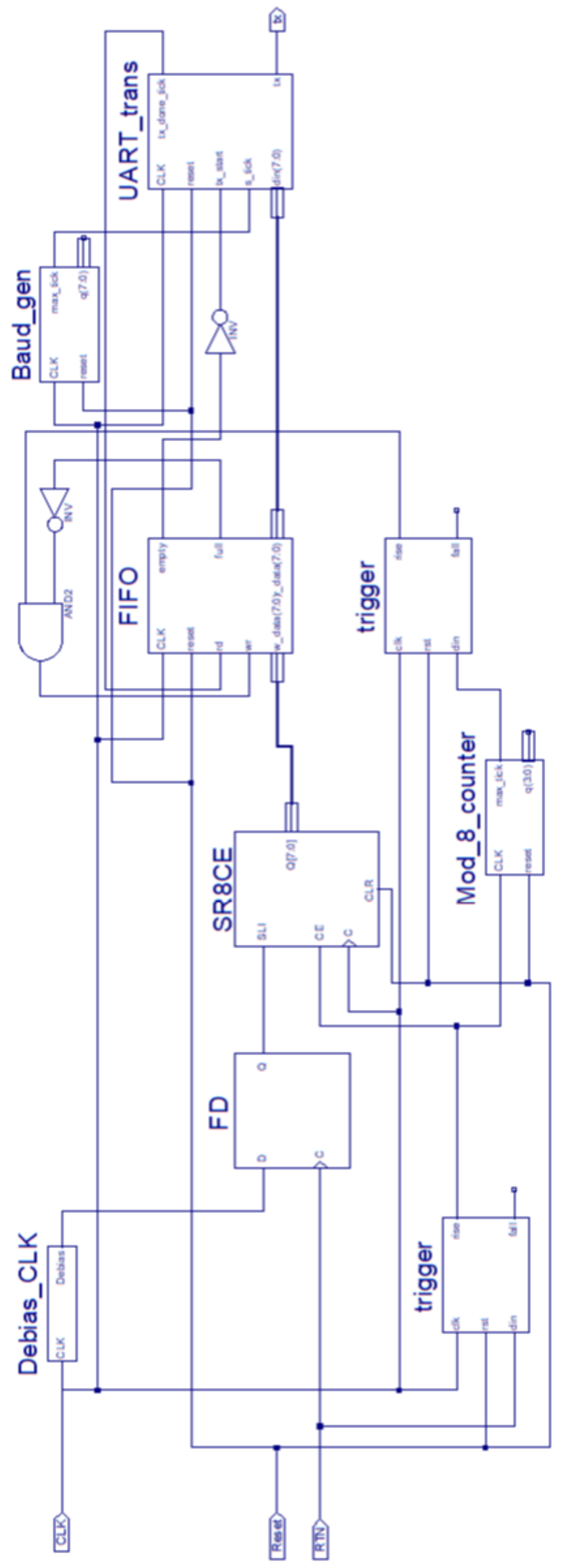


Figure 72, Block diagram of FPGA section of the proof of concept TRNG. This block diagram was used for the visual programming part of the FPGA implementation. Each block is created using VHDL programming, with the De-bias and Trigger modules being custom code and the other modules created using conventional designs. This block diagram shows all 3 key parts of the FPGA design, which are the sampling, buffering, and transmission circuits.

The sampling sub-circuit contains two key components, the first of these is a D-Flip-flop which is used to asynchronously sample an oscillator (in this case the clock of the FPGA) whenever a rising pulse edge is received from the edge-to-pulse circuit. To achieve the sampling, the oscillator is connected to the D-input of the flip-flop and the output of the edge-to-pulse circuit is connected to the clock input of the flip-flop. The output 'Q' will update to whatever value the oscillator was at when the rising edge of the pulse was received. The second key component in this section is a trigger module that is used to ensure each pulse from the edge-to-pulse circuit only results in one value being written into the data buffering section. This is achieved by the trigger module having both the FPGA clock and the edge-to-pulse output connected as inputs. Every time a pulse rising edge is received from the edge-to pulse circuit the trigger unit will output a pulse that has a duration of a single clock cycle of the FPGA clock. This single clock cycle pulse can then be used as an enable signal in other parts of the circuit that requires to be synchronous.

The data buffering sub-circuit is used to buffer the random numbers that are generated at an asynchronous non-constant rate so that they can be read at a constant synchronous rate. In our proof of concept circuit, it also converts the single generated random bits into 8-bit random words so they can be transmitted following the RS232 communication scheme. The first important module of the data buffering sub-circuit is the serial-to-parallel converter. For the proof of concept circuit this module converts 8 serial input random numbers to a single parallel 8-bit random word which can then be sent to the first-in-first-out (FIFO) buffer. The second key module is a counter which outputs a single pulse for every 8 input pulses from the trigger module of the sampling sub-circuit. This counter is used to control the transmission of data from the serial to parallel converter to the FIFO buffer. The data buffering sub-circuit also contains a trigger module that is the same as in the sampling sub-circuit. This second trigger module is used to ensure that the output from the counter is a pulse that only lasts for a single clock pulse of the FPGA. This is important to ensure that when data is transferred from the serial to parallel converter and then to the FIFO buffer, it is only transferred once per 8-bit word.

The final key module in the data buffering sub-circuit is the FIFO buffer, the main purpose of this module is to allow the random words to be read from the TRNG at a different rate from that which they are generated at. This concept is key as the random bits are generated at the rate of the RTN events whereas any application using these numbers can use them at a rate dictated by the application's requirements. The depth of the FIFO buffer can also be adjusted depending on the application's requirements. The FIFO buffer will write in the random number sent from the serial to parallel converter every time an enable pulse is received from the data buffering trigger module. When the FIFO is full, data will not be written in and instead will be discarded. The FIFO buffer will output a random 8-bit word every time an enable signal is received from the transmission sub-circuit unless the FIFO buffer is empty in which case nothing will be transmitted.

The transmission sub circuit is used to transmit the random 8-bit words from the data buffering sub-circuit to an external analysis PC. In practice this sub-circuit is not required for TRNG operation, but it is used here in our proof of concept circuit so analysis such as the NIST tests could be performed. The first key module in the data transmission circuit is the baud generator, which divides the FPGA clock signal down by 16 times to give a baud rate for the RS232 UART communication. The second key module is the UART transmitter, this is a module that follows the RS232 communication scheme and has a maximum data transfer rate of 19200 bps for the used FPGA. This maximum data transfer rate is the bottleneck for our proof of concept circuit. The TRNG itself can generate random bits at a much faster rate, the maximum of which is limited by the slew rate of the op-amps in the edge-to-pulse circuit.

Sampling Oscillator Parameters. The parameters for the oscillator are key for the proposed TRNG to work properly and output unbiased random numbers. The key parameters are the frequency of oscillation, and duty factor.

For the frequency of oscillation, the key point is that the frequency should be high enough that the signal is not sampled multiple times by the RTN before switching state. If the oscillator is too slow and sampled multiple times before switching, the output will contain long chains of the same bit which lowers entropy and causes the bitstream to be non-random. Figure 73 shows a simulation of a TRNG with multiple RTN traps with fixed capture times (τc) and emission times (τe) operating across a range of oscillator frequencies. The bit rate remains almost constant as this is controlled by the RTN event timing. When the oscillator frequency is reduced below $Fosc_{min}$, however, the generated bits start to fail the Runs Test for randomness. $Fosc_{min}$ is defined by equation 5 where n is the number of traps present in the entropy source. To act as a rule of thumb when choosing an oscillator frequency, it should be selected to be as high as possible without violating the other key parameters as any frequency above $Fosc_{min}$ will give the same results.

$$Fosc_{min} = \sum_{i=1}^{n} \frac{2}{\tau c_i + \tau e_i} \qquad (5)$$

For the duty factor of the oscillator the key point is that it should be kept as close to 50% as possible, this is because a duty factor significantly different to this will result in a bias of the output bitstream to either ‘0’ or ‘1’. Figure 74 shows the relationship between duty factor of oscillator and the Shannon entropy of the output bitstream. Shannon entropy is used to check if there is any bias towards one value. For a system with two possible outputs, the ideal result for Shannon entropy is ‘1’. Any reduction from this represents a reduction in the randomness of the results. To find the maximum allowable error in the duty factor away from 50% the NIST test suite was used to test bitstreams generated from a simulated TRNG over a range of duty factors. It was found that the maximum variation away from 50% that would still pass all 15 NIST tests was 6%. If the oscillator has a duty factor that is not ideal but has a consistent frequency greater than double $Fosc_{min}$ the duty factor can be improved by use of an extra de-bias module. This module takes the unbalanced oscillator as the input and toggles the output every time a rising edge is detected in the input. This follows the de-biasing scheme outlined in section 4.4.1 as proposed by Mohanty et al. [37].

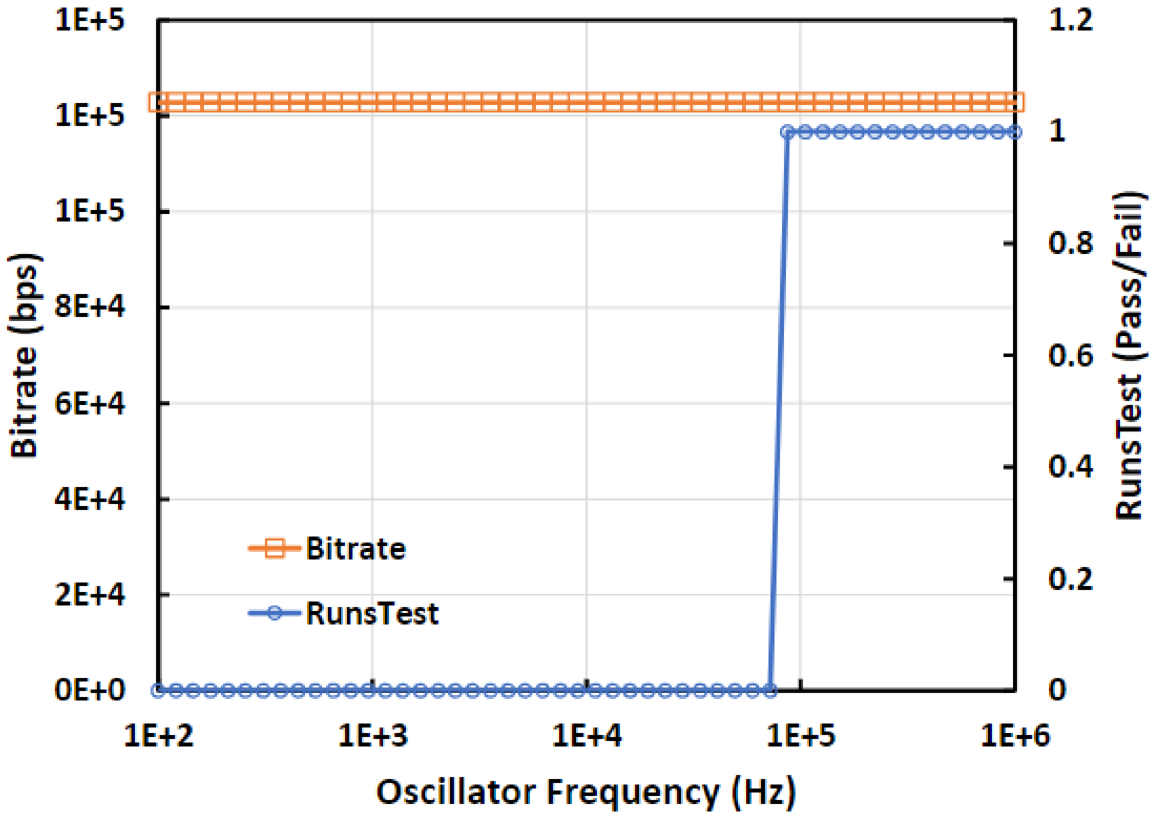


Figure 73, Output bit rate and Runs Test result against the frequency of the oscillator being sampled by the RTN edge pulses. The Runs Test is used to find runs of '1' or '0' and determines whether a run of the same result that number of times is acceptable in a bitstream of the given length, based on a null hypothesis theory with a significance value of 0.01. The Runs Test gives a result of '1' if it passes and a '0' if it fails, a failure indicates oversampling of the oscillator by the RTN signal causing long chains of the same result. The RTN signal simulated for this test uses 5 traps with τc and τe values extracted from characterisation of real traps. Both the rising and trailing edges of the RTN signal were used for sampling giving and $Fosc_{min}$ of 1.21E+5Hz. The minimum frequency that could pass the Runs Test was 8.6E+4Hz and the output bit rate was fixed at 122800bps.

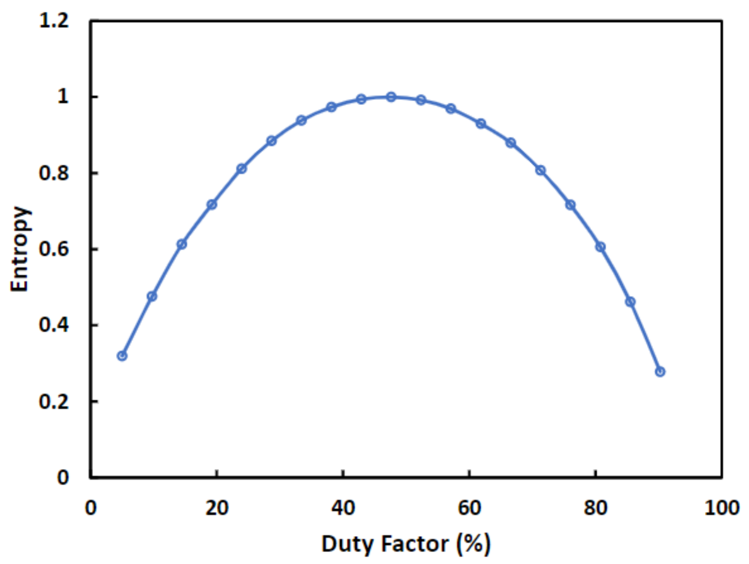


Figure 74, Shannon entropy of output bitstream when RTN signal is used to sample oscillators of different duty factors. Shannon entropy is used to measure the ratio of '0' bits to '1' bits. A Shannon entropy of '1' indicates a 50% of any selected bit being '0' or '1'. A Shannon entropy less than 1 is caused by a bias in the bitstream to either '0' or '1', indicating that the bitstream is not random. A 100kb bitstream was generated at each data point to be tested. 9 NIST tests were done on these bitstreams to find the largest error from 50% duty factor that could pass. When the largest error that could pass 9 NIST tests was found, a 3Mb bitstream was generated with that error to verify the results with all 15 NIST tests.

Bitstream analysis and NIST test results. To verify the randomness of the generated bitstreams, they were transmitted to an external analysis PC. On the analysis PC, bitstreams were divided up into blocks whose sizes are defined by the guidelines provided in the instruction manual for the NIST test suite[33] (see section 4.4.1 for full details) and all 15 NIST tests were run. The significance value for our NIST tests was set to 0.01 which is the recommended value from NIST. For the tests excluding the non-overlapping template matching, random excursions, and random excursions variant, multiple blocks of random numbers are tested, and a proportion defined by NIST must pass the test for the bitstream to be considered random. For the three tests that do not require multiple blocks tested, the individual P-value is used to determine success or failure. The bitstreams from our TRNG have passed all 15 NIST tests without post-processing as shown in Table 6.

Our generated bitstream was also tested by using the DIEHARD test suite [124]. As the original version of the DIEHARD test suite is not readily available any more, we used the Dieharder test suite constructed by Robert G. Brown [124] for Duke University, which contains accurate reconstructions of each of the original DIEHARD tests. To perform the DIEHARD tests the binary bitstreams used for the NIST test were converted to a stream of 32-bit unsigned integers, as required by the “202” generator setting in the Dieharder test suite. 8Mb of data was used for testing in the Dieharder suite. The lengths of the bitstreams used for each test were determined by the guidelines provided with the Dieharder test suite [124]. The random excursions and random excursions variant tests were not included in the Dieharder test suite, however the tests are identical to the tests with the same name in the NIST test suite, so the NIST test results are shown. The Monkey tests (OPSO, OQSO and DNA) are listed as being unreliable so they were not used for the DIEHARD testing. The DIEHARD tests work on the same null-hypothesis theorem as the NIST tests, so that we kept the same significance value of 0.01 for determining P-value success or failure. The results of the DIEHARD tests are shown in Table 7. Our design was able to pass all of the available DIEHARD tests, giving a great confidence in the truly random nature of our generated bitstream. To further illustrate the randomness of our generated bitstream, a sample set of 146k bits was used to create a bitmap in Figure 75, where no clear patterns can be seen, confirming the randomness of the bitstream.

Table 6, NIST Test suite results for bitstream generated by our TRNG design with a significance value of 0.01.

Test ID	Name	Proportion	Pass/Fail
T1	Monobit test	99/100	Pass
T2	Frequency tests within a block	100/100	Pass
T3	Cumulative sums Test	96/100	Pass
T4	Run Test	99/100	Pass
T5	Longest Run of Ones in a block	98/100	Pass
T6	Discrete Fourier Transform Test	49/50	Pass
T7	Non-overlapping Template Matching	n/a	Pass
T8	Approximate Entropy Test	96/100	Pass
T9	Serial Test	100/100	Pass
T10	Binary Matrix Rank	99/100	Pass
T11	Overlapping Template Matching	8/10	Pass
T12	Universal	8/10	Pass
T13	Linear Complexity	10/10	Pass
T14	Random Excursions	n/a	Pass
T15	Random Excursions Variant	n/a	Pass

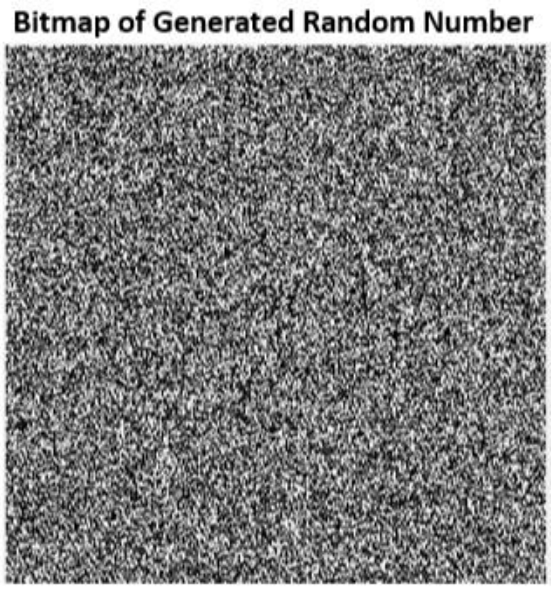
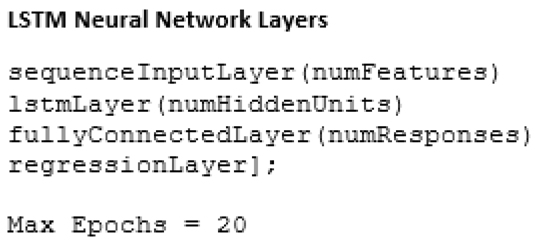


Figure 75, Bitmap of 146k output random bits from the improved RTN based TRNG circuit.

Table 7. DIEHARD Test Results

Test ID	Name	P-Value	Pass/Fail
T1	Birthday spacings test	0.87125	Pass
T2	Overlapping permutations	0.98398	Pass
T3	Rank of matrices	0.56833	Pass
T4	Bitstream test	0.72978	Pass
T5	Count-the-1's stream test	0.36753	Pass
T6	Count-the-1's byte test	0.22222	Pass
T7	Parking lot test	0.94059	Pass
T8	Minimum distance test	0.12461	Pass
T9	3D spheres test	0.12013	Pass
T10	Squeeze test	0.55082	Pass
T11	Overlapping sums test	0.16278	Pass
T12	Runs test	0.95088	Pass
T13	Craps Test	0.35721	Pass
T14	Random Excursions	n/a	Pass
T15	Random Excursions Variant	n/a	Pass

Resistance to Machine Learning Attacks. To test how our TRNG would resist machine learning attacks and further enhance confidence in the security of the TRNG, we tested some of our generated bitstreams with an LSTM neural network. LSTM networks have been shown to be able to predict some random number generator outputs, even if the RNG passes all of the NIST tests [125]. We used 1 million generated bits for testing with the LSTM network and tried to predict future bits that will be generated by the TRNG. The construction of the LSTM neural network was guided by the earlier works that used machine learning to try to crack random number generators [38], [125], [126]. Based on the principles outlined in these previous works, our LSTM network was constructed with an LSTM layer of 256 hidden units and a single hidden layer, connected to a fully connected layer. Figure 76 shows the format and connections of the layers within the LSTM network. The training was performed with 1 million bits over 20 epochs. 100,000 predicted bits were generated and tested against 100,000 bits from the TRNG and the hamming distance between them was calculated. The hamming distance achieved was 0.4981. The result of this process is that after training, the LSTM network was only able to predict the bits of our TRNG output with an accuracy of 49.81%, which is very close to the ideal value of 50% that could be achieved through random guessing. This shows that our design is resilient against machine learning based attacks and further increases confidence in the true random nature of the output.



LSTM Neural Network Layers

```
sequenceInputLayer(numFeatures)
lstmLayer(numHiddenUnits)
fullyConnectedLayer(numResponses)
regressionLayer];

Max Epochs = 20
```

Figure 76, Layout and connections of layers within the LSTM network used for testing the TRNG design.

Next, we will discuss the various benefits and limitations of this newly proposed design, as well as comparing it to previously proposed designs. We also discuss the tuning of various parameters within the design to optimise it for different applications. Previously proposed RTN based TRNGs could not extract the full entropy from multi-level or abnormal RTN. They rely upon finding a device from a large array that fits the ideal requirements of the TRNG[48]. These ideal devices can be exceedingly rare, resulting in a large TRNG footprint as many devices have to be included in the array as well as complicated tuning and selection circuits to identify and select the ideal device. In our new design the ability to use devices that have multi-level or abnormal RTN vastly increases the chances that a device will be suitable for use in a TRNG. This means that the size of array required for potential entropy sources can be reduced to fewer than 10 devices. There is further benefit to using multi-level RTN in terms of output bit rate. In traditional designs where multi-level RTN might be present, the device is still usable if one trap is significantly larger than the other. In this case, only the entropy from this largest trap, however, was extracted. With our design, we can extract entropy from all of the traps. This allows us to utilise many more RTN events in a given time window, when compared to a traditional design.

The new processing scheme proposed in this work gives us several key advantages over previous works. The first one is the removal of the complicated tuning of a sampling clock frequency. In traditional designs, the sampling clock frequency must be heavily correlated with the capture and emission times of the device being used as an entropy source in order to maintain a high output bit rate, while also avoiding oversampling of the entropy source. As the capture and emission times of a given device are unknown and can have a large device to device variation, the selection of a sampling clock frequency is complicated as it has to match a specific device before manufacture, or the TRNG must have a complex self-tuning mechanism that will greatly increase design complexity and power consumption. With our proposed TRNG design the RTN events are used to do the sampling, so that no tuning is needed as long as the oscillator being sampled is running at a higher frequency than the RTN events. This sampling technique also makes our design immune to over-sampling attacks which has been known to be an issue for previous TRNGs [127], where a higher frequency signal is imposed on top of the sampling clock to oversample the RTN signal and give long chains of '1' or '0'. It is important to note that our design would be susceptible to under-sampling attacks which would give the same effect as oversampling for a traditional design. It is more difficult, however, to remove edges from the oscillator signal, without having direct access to the oscillator itself, than to add extra edges to a signal by simply applying an additional signal on top of it.

Another key advantage of the new processing scheme is that it removes the need for complicated post-processing. In traditional designs, Von Neumann or ORing post-processing were the most popular techniques to remove bias from the output[128]. These techniques require large complex circuits to implement and reduce TRNG throughput. More recently new post-processing techniques have been proposed that use a toggling output for each input rising edge from the entropy source[37]. This greatly reduces the circuit complexity, but it also greatly reduces the bit rate of the design. With our new technique no post-processing is required, as long as the sampling oscillator has a duty factor between 44% of 56%.

The current bit rate of our TRNG is limited to 19,200 bps by the requirements of the RS232 communication. For practical applications, this transmission circuit is not required so this limitation on the maximum bit rate would be removed. With the data transmission bit rate limit removed the next limiting factor is the slew rate of the op-amps used in the edge-to-pulse module. With the currently selected components the slew rate allows us to achieve a bit rate of several Mbps. If the edge-to-pulse model was designed from the transistor level for this purpose, higher slew rates should be achieved, further increasing bit rate into the range of at least tens of Mbps.

The speed of the RTN events also governs the bit rate of the TRNG. At the current bit rate limit of 19,200 bps it is extremely common to have an RTN which is capable of generating bits at this rate. As the bit rate increases to a level that the rate of the RTN events becomes the limiting factor, there are several options available to increase the TRNG throughput. The first option is to find a device with faster RTN events. As mentioned earlier RTN event rate has a large device to device variation. Some devices may be capable of generating TRNGs with a bit rate in the Mbps range[129] while others can only achieve a few kbps. In practice all TRNGs could be tested after manufacture and have their throughput rated accordingly so a consumer can select a TRNG with a suitable speed for their given application. Another option for increasing throughput is to use an AC RTN scheme for the voltages applied to the entropy source device. An AC RTN scheme has been shown to be capable of increasing the observed rate of RTN events by thousands of times [103]. The downside to using this AC RTN scheme is an increase in design complexity. The final option for increasing throughput is to have multiple devices used in parallel as the entropy source, doing this could massively increase the bit rate. As for the AC RTN, having multiple devices as the entropy source increases design complexity so it should be avoided if not necessary for a given application.

Another factor to consider for TRNG operation is the temperature effect on the TRNG. The key effect of temperature on the proposed TRNG is its effect upon the rate of RTN events. The effect of temperature on the capture and emission times of RTN traps has been studied in some detail [130]. It is known that typically an increase in temperature will decrease both the capture and emission times of an RTN trap. For our TRNG, higher temperature will increase bit rate. In terms of security, these variations will not affect the randomness of the generated bitstream thanks to the RTN sampling oscillator scheme. This was verified by heating the entropy source device in the design to 125°C and recording the output bits generated. With the entropy source device heated to 125°C, a high entropy of 0.9996 was maintained and no change in output bitrate was observed due to the 19,200 bps limit still set by the RS232 communication. The only time that security could be compromised is if the RTN is accelerated to such an extent that it surpasses $Fosc_{min}$ and the RTN starts oversampling the oscillator. To avoid this, Fosc should be selected to be sufficiently above the maximum possible rate at which RTN could be accelerated by raising temperature. Conversely, decreasing the temperature will increase the capture and emission times. Increased time constants will slow down the generation rate of the random numbers, but it will not affect the security of the system as the RTN will simply sample the clock signal less frequently but will retain the same entropy.

A further consideration for the parameters related to the entropy source is the voltage applied to it, namely Vg and Vd. The effect of Vg and Vd on RTN traps has also been studied previously[107], [123]. Typically, an increase in Vg will decrease the capture time and increase the emission time with a given Vd and the exact relationship being highly device dependent. For Vd variation with a given Vg, an increase in Vd will cause a decrease in emission time and an increase in capture time[131], again with the exact relationship being highly device dependent. For low power applications it is advisable to use the lowest possible value for Vg and Vd as this will give the lowest current flow through the device, thereby reducing power consumption. It has also been reported that a low Vd will give the largest single trap impact on the Id within the device, making extraction of the RTN signal easier. Figure 77 shows the effect on the output bitstream entropy by altering the Vg supplied to the entropy source device. Figure 77 shows that changing the supply voltage between 0.35-0.8 V has less than 0.4% effect upon the output entropy showing that the design has a high tolerance for supply voltage variation.

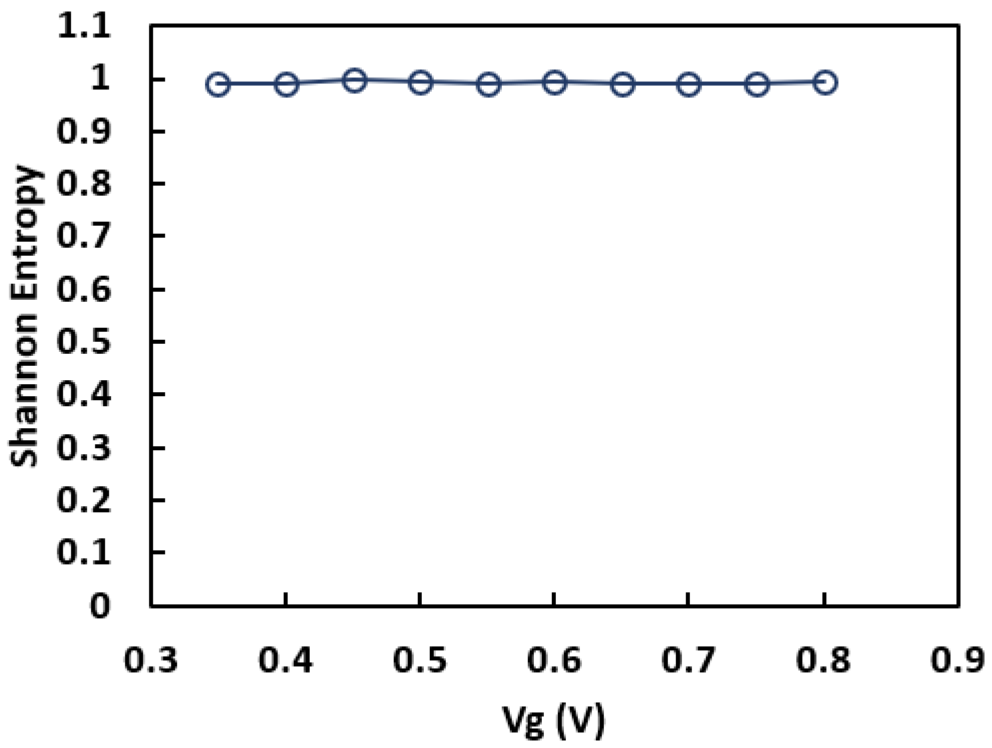


Figure 77, Effect of Altering Vg applied to entropy source device on the entropy of output bitstream. Vd fixed at 0.3V with temperature at 30°C.

The last key consideration for the proposed TRNG design is the depth of the FIFO buffer. The depth of the FIFO buffer is a parameter that can be adjusted to make the proposed design fit a wider range of applications. For applications that require consistent random numbers at a low rate then the depth of the FIFO buffer does not need to be significant as the TRNG will be able to fill the buffer at a faster rate than the numbers are read from it. For applications that require fast number generation in bursts then the depth of the FIFO buffer can be increased as the TRNG can consistently fill the buffer while no numbers are being read from it and then output them at a faster rate when required.

We now compare our design with some of the prominent TRNG designs previously proposed. The key figures for the discussed designs are shown in Table 8. Firstly, we look at the original RTN based TRNG proposed by Brederlow et al. [48] The key improvements over this design include removing the post-processing circuit and thereby increasing design efficiency. As we use a much broader range of RTN types, it reduces the required size of device array.

The design of Figliolia et al.[53] has a good generation rate of 25MHz (which we could potentially match with the UART limitation removed as the Figliolia design is also RTN based), but is significantly more complex and requires a lot more design area when implemented onto an IC. Also, Figliolia's design was not tested using the DIEHARD or NIST test suite, so the randomness of the generated bitstream is not verified.

Chen et al.[60] proposed a design that can either offer a comparable or better bit rate than our design depending upon the type of device used and the level of confidence in the security that is required. At a similar security confidence, the bit rate will be similar to ours, if the RS232 limitation is removed and a single entropy source is utilised. Our design offers improvements over Chen's design in areas of design complexity and efficiency of entropy extraction as we utilise more types of RTN. Chen's design is only tested in a simulated environment, rather than with a practical circuit, so that there is more uncertainty in the real-world performance of their design when compared to ours.

The design by Mohanty et al.[37] improves the efficiency when compared to other previous RTN based designs. Our proposed design further improves the efficiency. Mohanty removes the need for post-processing circuits through a balancing scheme. We also remove the post-processing circuits by using RTN to sample an oscillator. For multi-level RTN, Mohanty's proposed design extracts the entropy from the slowest trap, whereas our design extracts the entropy from all RTN events. This improved entropy extraction is the primary reason why our design generates bits at a significantly higher rate as compared with Mohanty's design.

Next, we compare our design with some of the most prominent non-RTN based TRNGs. The metastability based TRNG proposed by Sanu et al. [52] can achieve bit rates higher than any of the RTN based TRNGs, but it is significantly more complex and requires tuning and processing circuits, making it suit high performance, high power applications. On the other hand, the simplicity and low power of our design fits better for applications such as remote IoT sensors or ID cards.

Matsumoto et al. [57] proposed a design based on tunnelling in SiN devices. Its bit rate is lower than the bit rate of our design without the RS232 limitation and its area is even larger than that of Sanu et al.[52]. Matsumoto also mentions that the proposed design only works well with a specific type of device, whereas our CMOS-based design does not have this limitation.

The design proposed by Jiang et al.[45] is based on the write time of memristors. Jiang's design can achieve a bit rate that is comparable with our design when limited by the RS232 and would be significantly slower than our design with that limitation removed. Their design has a similar level of complexity as ours. Our design has the advantage that it is based on CMOS technologies, making it readily implementable.

*Table 8. Comparison with Other TRNGs. *speed when limited by RS232 communication.*

TRNG	Entropy Source	Generation rate	Design area	Post-Processing	Verification
This Work	**Random Telegraph Noise**	**19.2 kbits/s***	**N/A**	**None**	**Passes NIST and ML attack**
Sanu et al. 2015 [52]	Metastability of inverters	162.5 Mbit/s	1088um^2	XOR decorrelator and BIW extractor	Passes NIST
Figliolia et al. 2015 [53]	RTN with sigma delta converter	1-25 Mbit/s	2200um^2	None	N/A
Matsumoto et al. 2008 [57]	Tunnelling in SiN device	0.3 Mbit/s	1200um^2	None	Passes FIPS 140-2
Brederlow et al. 2006 [48]	Random Telegraph Noise	200 kbits/s	0.009mm^2	Von Neumann	Passes NIST
Chen et al. 2015 [60]	Random Telegraph Noise	0.25-5 Mbits/s	N/A	Von Neumann or bit truncation	Passes NIST
Jiang et al. 2017 [45]	Memristor Write Time	6 kbits/s	N/A	none	Passes NIST with reduced α value
Mohanty et al. 2017 [37]	Random Telegraph Noise	3 bits/s	N/A	none	Passes NIST

In conclusion, we present an experimentally verified TRNG design, based on the trap event timing of multi-level RTN, using standard CMOS technology. The proposed design makes use of a new edge-to-pulse entropy extraction technique, which enables us to utilise multi-level RTN, abnormal RTN as well as traditional two level RTN, thereby improving on device selectivity and output bit rate. The design also introduces a new processing technique, which removes the need for any post-processing, the need for complicated tuning circuits for sampling frequency, as well as increasing resistance to oversampling attacks. The randomness of the proposed TRNG is verified by the 15 NIST Special Publication 800-22 randomness tests as well as the DIEHARD randomness tests. The proposed design also shows resilience against machine learning attacks. Selection of key parameters is discussed for optimising TRNG performance in terms of output bit rate and design complexity for different user requirements. The novel improvements presented in this work pave the way for a TRNG suitable for the low power, low area, and low-cost applications that are key requirements of IoT edge units.

Chapter 5 - A Novel Time-Dependent PUF

5.1 Introduction

As mentioned in section 2.2, the need for more secure methods of authentication has been evident for a long time [5]. One of the promising techniques for authentication is the Physically Unclonable Function (PUF). The PUF has been around for many years now [77], [84], [85], [132], but has not broken through as a mainstream cybersecurity solution [133]. The main reasons for this are: limited number of available challenge and response pairs (CRP), long-term reliability of PUF circuits, slow speed for authentication, difficulties in storing and distributing CRPs to the server, and importantly, vulnerability to attacks [28],[134]. The majority of recent research into PUFs has used the same basic principle of finding some repeatable physical property and using this to create a set of fixed CRPs, which are defined at the creation of PUF and later distributed to a server that controls authentication. The PUFs built in this way are generally referred to as "weak PUFs", as their low number of available CRPs limits their applications in the cyber security domain [64]. A strong PUF, which has an infinite number of CRPs, is difficult to develop as the circuits are too complex to justify their use over existing solutions such as public key infrastructures and digital certificates. A radical new approach is needed if the PUF is to become a mainstream solution for cybersecurity applications.

In this chapter we propose a novel approach to the PUF scheme. The new PUF design utilises the de-trapping phenomena of defects within nanoscale MOSFETs in order to add a time-dependent element to the PUF design. This aim of creating this new concept is to increase the number of CRP pairs for a given number of entropy sources, without increasing the circuit footprint of the design. The second major aim is to use this new design to create a novel PUF protocol that can be used for authentication in applications such as local networks.

We first look at the operating principles of the newly proposed PUF scheme and explain in detail how each section works as well as how the PUF's parameters can be adjusted to tune the PUF for different situations. We then look at the performance of the PUF in terms of the key criteria that were discussed in section 5.4. This evaluation includes analysis of the hamming distances as well as a point by point evaluation of the key PUF characteristics as defined by Maes et al. [64].

5.2 The Time-Dependent PUF Concept

In this section we discuss the operating principles of a time dependent PUF. We first outline the basic principles of a time-dependent PUF in terms of the CRP behaviour, created by use of a PUF module and associated probability model. Next, we look at how the probability model can be created from observing the de-trapping behaviour of defects within nanoscale MOSFETs. We then explore how specific PUF, and probability model parameters can be adjusted to optimise performance. We then go into more detail on the construction of the PUF module through the use of a proof of concept circuit, built from discrete components and an FPGA module. This proof of concept circuit is used to verify that the proposed PUF module is feasible to construct and operate in the real world. Finally, we analyse three novel PUF protocols that can be used with the proposed time-dependent PUF to create a full authentication scheme. Like the previously proposed PUFs discussed in this work, the security of the response length is multiplicative rather than exponential. The multiplicative nature of the PUF is one of the key deficiencies of the concept that has been present since its first inception and has not confidently been solved. While it does mean the security of these PUFs is not optimal there are still viable applications for them in areas such as local network authentication.

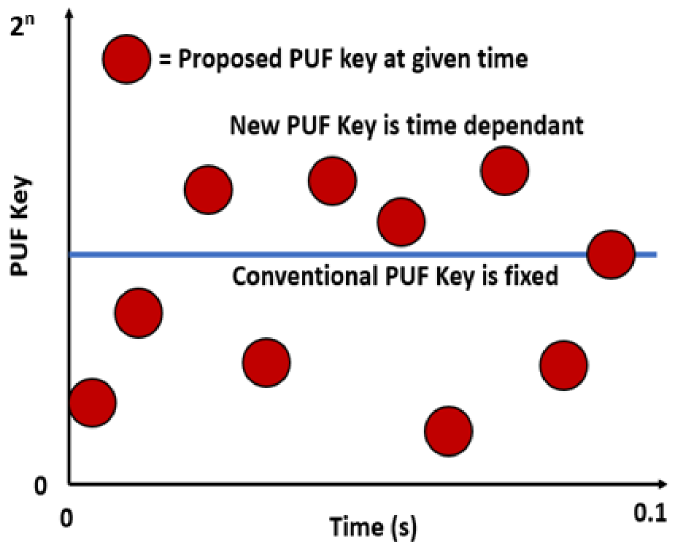


Figure 78, Comparison of response time dependence between proposed PUF and conventional PUF schemes.

The basic principle of a time-dependent PUF is that the response to a given challenge is dependent upon time. This principle is illustrated in Figure 78 where a conventional PUF design has a response that is fixed, the proposed time-dependent PUF scheme has responses that can change to any value between 0 and 2^n when the challenge time is changed, where n is the number of entropy sources (equal to response length) within the PUF.

The proposed time-dependent PUF consists of two main elements, the first of which is the PUF module. The PUF module is where the CRP behaviour, which enables authentication with the PUF, originates. For the proposed time-dependent PUF, the PUF module is a system that contains an array of entropy sources which have a known probability to generate a specific response when measured for a specific amount of time. The response for each entropy source is either ‘1’ or ‘0’ depending upon whether a specified event is observed or not. The second key element is the probability model. The probability model is a model of the PUF module which contains the probabilities of each entropy source exhibiting the specified event at a given time. This probability information can be expressed as a probability curve as shown in Figure 79. The figure shows how the probability of observing the event varies between 0 and 1 over the measurement window. To get a CRP from this, the challenge can be defined as any specific time within the measurement window. The response is whether the specified event occurs or not, the probability curve for an entropy source can be used to determine what the expected response should be at any given time. If the probability of observing the event is '1' then the expected response is '1', if the probability of observing the event is '0' then the expected response is '0'. The expected response for times where the probability is between these values is unknown. Each bit of the CRP is created by finding the expected response from each entropy source within the PUF module at that specified challenge time. If the expected response from an entropy source is unknown for the specified challenge time, the response from that entropy source is ignored when authenticating. The other option for uncertain responses is the authenticating server removing that entropy source from the challenge, instead selecting only entropy sources that have a certain response at the specified challenge time. Choosing between these two options is covered in section 5.3. The length of the response is therefore determined by how many entropy sources are measured for each challenge. If the number of available entropy sources is greater than the required response length, different combinations of entropy sources can be used to give multiple CRPs for each challenge time, increasing the total number of CRPs for each PUF module.

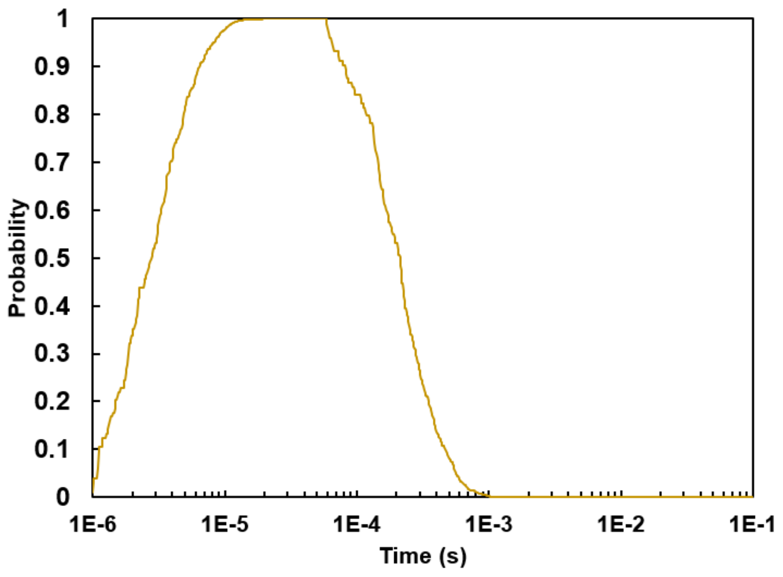


Figure 79, Example of a probability curve for a single entropy source which shows the probability of finding a specific event at each point in time over the measurement time window.

5.2.1 Creating and optimising the PUF Probability Model

The probability model is the key feature which enables the functionality of this PUF. In this work we measure de-trapping events of defects within nano scale MOSFETs as our entropy source. The basic principle is that a number of de-trapping events inside of nanoscale MOSFETs have a certain probability to be observed within a given amount of time. If the probability of either not observing or observing a certain number of de-trapping events at a given time is almost certain, we can define these outcomes as ‘0’ and ‘1’ respectively, to create a single bit challenge/response. To observe the de-trapping events, we use the TDDS methodology outlined in section 3.4.5. Figure 80 shows the waveforms and voltages used for the TDDS measurements. The typical time for the measurement is 0.1s of charging and then 0.1s of measurement.

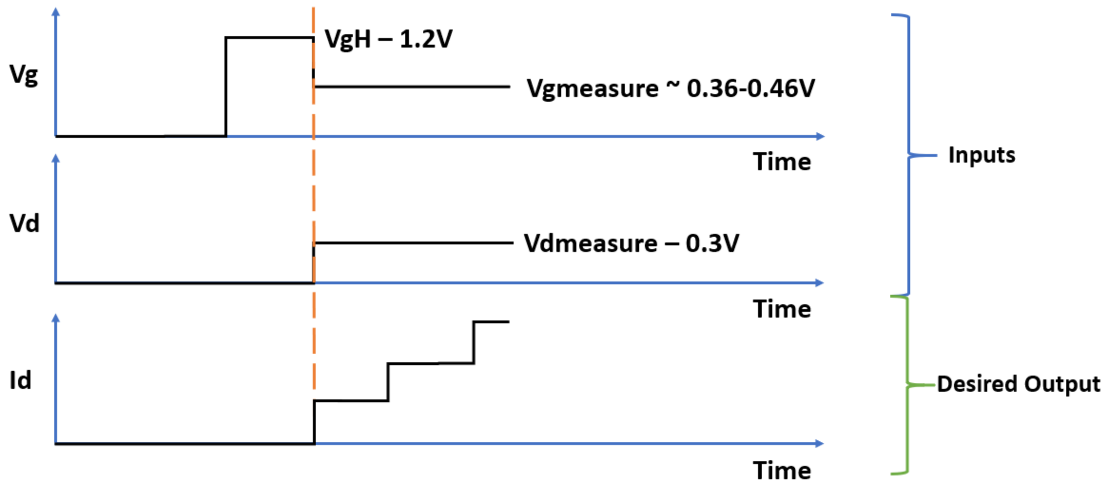


Figure 80, Waveforms and parameters of TDDS measurements used for proposed PUF concept.

The TDDS measurement principle works because of the typical dependence of defects time constants on applied Vg. Figure 81 shows how τe is typically shorter at low Vg and τc is typically shorter at high Vg. This dependence is the same as that utilised for the AC measurement procedure used in section 4.3. Here the high Vg is used to charge traps within the device quickly. After the charging step, the Vg is dropped to the low value to allow for the traps to discharge quickly. It is this discharging, which is observed as steps in the measured Id, with the timing of the discharges being dependent on the distribution of te values for each trap.

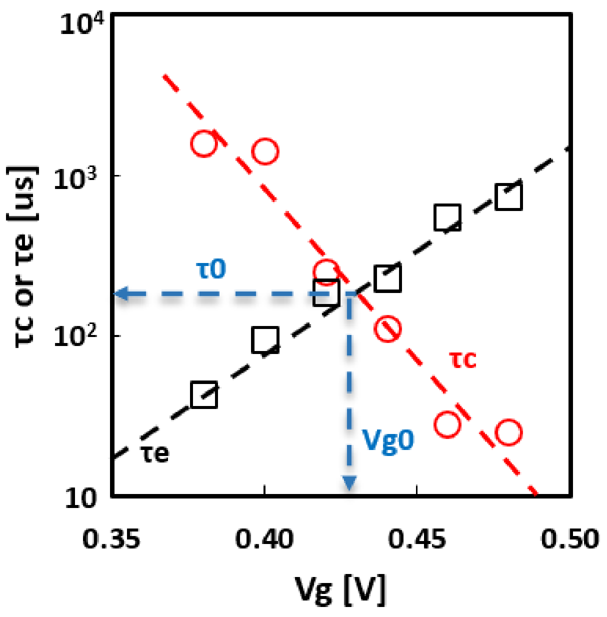


Figure 81, Typical dependence of τc and τe to applied Vg.

For this part of the work, we used 90x70nm pFET devices of the HK45 wafer. 91% of the tested devices show clear steps in TDDS measurement. This is already a very good yield for fresh devices although it could potentially be improved even further by using the stressing scheme proposed in section 4.2, if required. We have also shown in section 4.2.2 that the timing characteristics of these traps is stable.

To create the probability model, each response device within the PUF is measured using the TDDS procedure 500 times. We then define a specific number of traps that we are looking for (selection of this number will be discussed in more detail later in this section) and calculate the probability of having that number of traps at any given time within the measurement window. Figure 82 illustrates this process of finding the de-trapping times and then how that relates to the probability model for that device. It can be seen how the probability of finding the correct number of de-trapping events starts at 0% and increases (ideally to 100%) and then decreases again (ideally to 0%). When the probability is ~0% the expected response from the PUF should be that the correct number of de-trapping events is not found and for ~100% the correct number of de-trapping events is found. Between ~0% and

~100% the result is uncertain. To create the probability model of the whole PUF each individual bit probability model is combined. The bottom part of Figure 82 shows a PUF probability model that has 10 CRP bits. It can be seen how each bit's probability curve is unique and varies between ~0% and ~100% across the measurement time window. It is this uniqueness of each bit's probability curve that enables PUF operation. The distribution of possible challenge times should be logarithmic as the de-trapping events occur across the time window in logarithmic scale more evenly than with a linear scale. The probability model of a full PUF is the combination of the individual probability models of the de-trapping exhibited in individual 'bit' devices. In practice, a PUF module would be created by measuring the de-trapping events in each bit device and then be distributed to the authentication server.

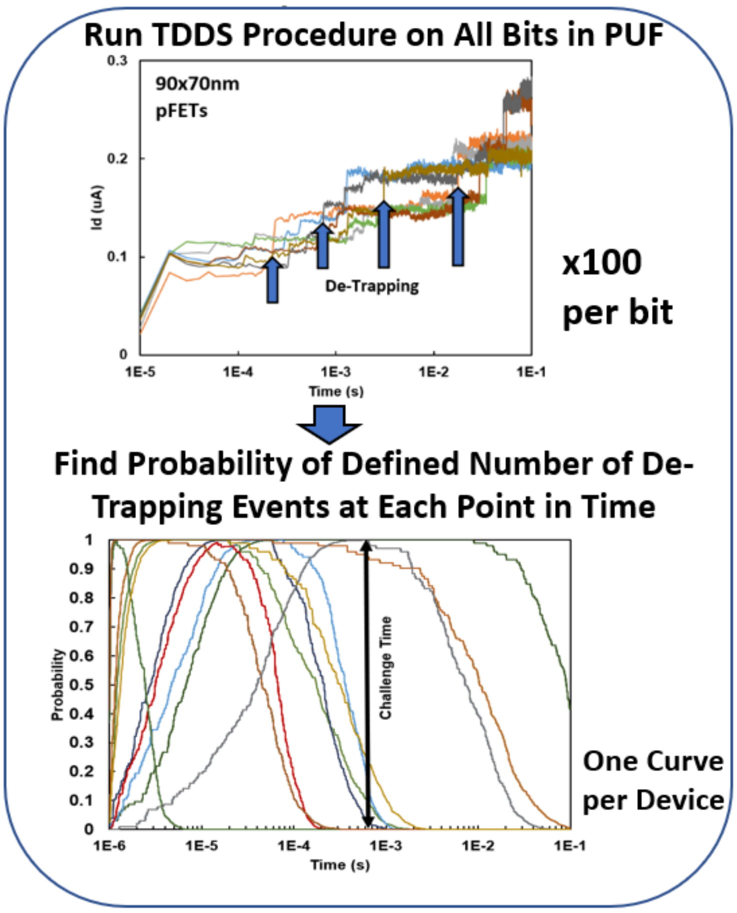


Figure 82, Methodology for creating the probability model of defect de-trapping events. One model is created for each CRP 'bit' within the PUF. The models are then stored on the authenticating server. Bottom plot shows the probability models from 10 devices (bits) that have had this methodology applied. 'Defined number of de-trapping events' is the number of observed events whose probability of having is found.

Here, we look at how some of the parameters of the probability model can be selected and adjusted to improve the performance of PUF. The three key parameters are the number of grouped traps, the number of ORed remeasurements, and the starting trap. The number of grouped traps refers to the number of traps we find the probability of observing. The number of grouped traps is usually a range and its purpose is to increase the probability of finding the correct number and to tune the bit bias of the PUF. The number of Ored measurements refers to the number of times the TDDS measurement is repeated on each challenge bit. The main aim of repeating the measurement is to increase the proportion of possible challenges that will be certain, reducing the need for extra devices to give a guaranteed number of certain responses. The starting trap refers to the trap number at which the group range starts. For example, a starting trap of ‘1’ would mean that the first detected trap would be the start of the number of grouped traps range, starting trap value of ‘2’ would that the first detected trap is ignored and counting starts on the second. The aim of adjusting the starting trap value is to better balance the distribution of time at ‘1’ and time at ‘0’ across the measurement window.

Figure 83 shows how changing the number of grouped traps effects the probability of a response device within a PUF. By increasing the number of grouped traps, we allow for a greater range of possible results from the TDDS measurement that will result in a positive ‘1’ outcome. As we increase this range, the maximum probability that the probability model can achieve increases. In Figure 83 we can see that when we are only looking for a single trap the maximum probability is around 0.65, whereas when we are looking for between 1 and 4 traps the maximum has increased to ~100%. Having a maximum probability of around 100% is essential for the PUF operation as this is the only time when a ‘1’ result can be achieved as the response. We can also see that increasing the number of grouped traps widens the distribution of probabilities against time. It is this widening effect that can be used to control the bit bias of the PUF. Figure 84 shows the bit bias of across all possible challenges on 60 PUF bits with various different grouping parameters. To simplify the operation of the PUF it is desirable to have a constant number of grouped traps across all devices. Through the experimental testing we found that 3 grouped traps gave the best balance between a good maximum probability and a 50% bit bias of responses, when the sample times are distributed on a logarithmic scale. It is important to note that this ideal number of grouped traps will only be accurate for the specific device size and design that we used here. Different devices will require different tuning of this parameter as they can have different average numbers of traps per device.

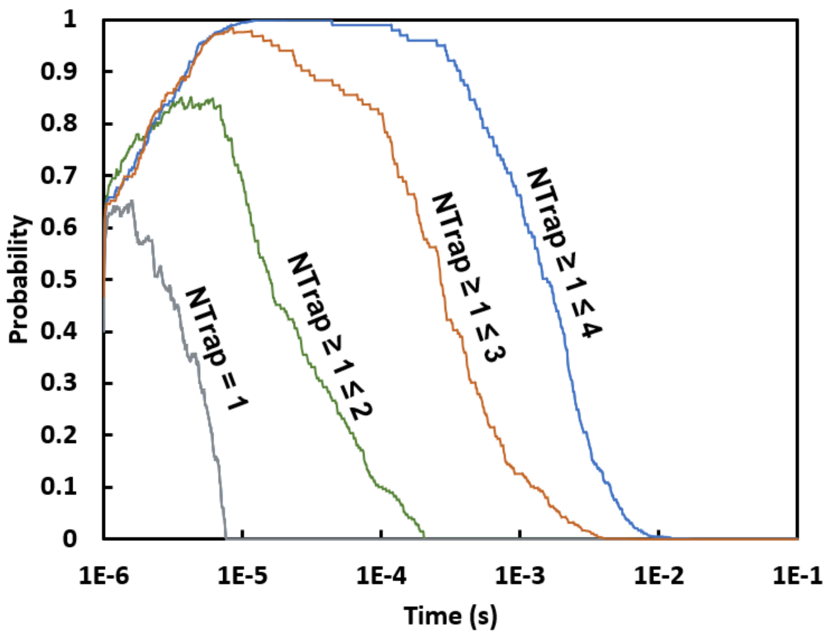


Figure 83, Effect of changing the number of grouped traps parameter on a PUF probability model.

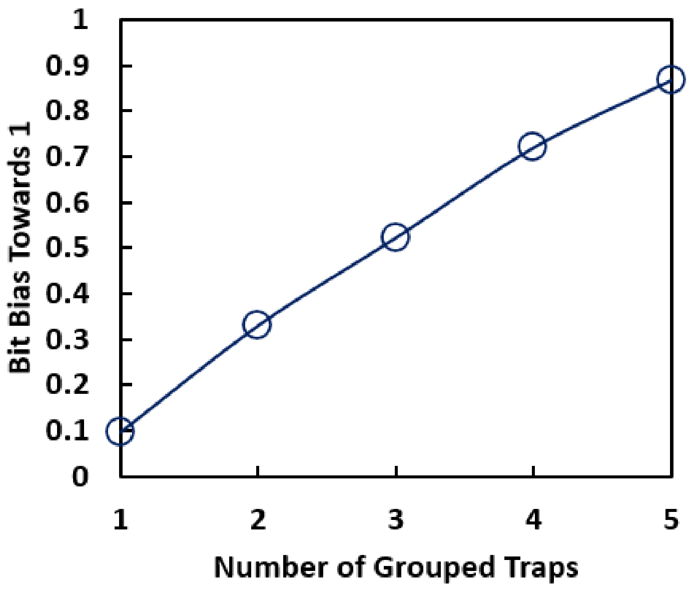


Figure 84, Effect of the number of grouped traps on the bit bias of 60 PUF bits across all challenge times.

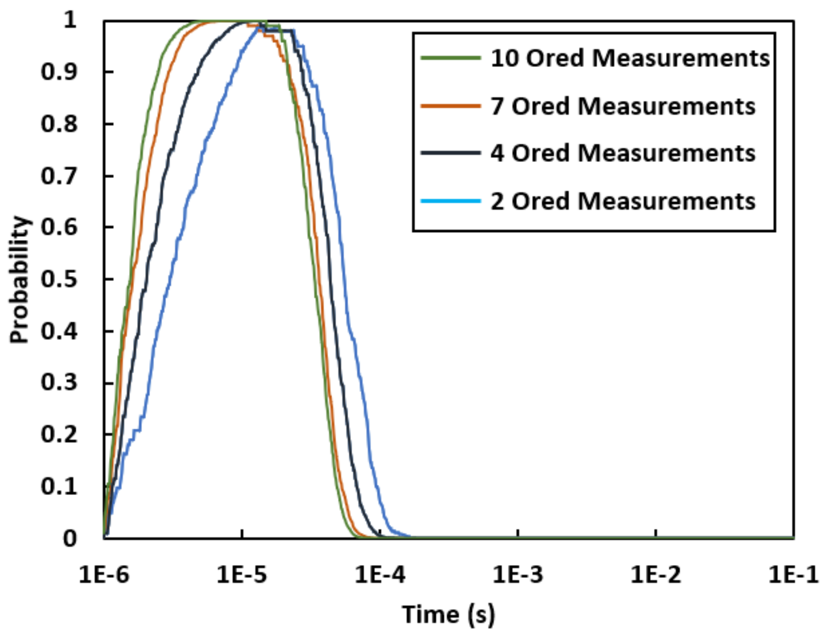


Figure 85, Effect of changing the number of the ORed measurements parameter on a PUF probability model.

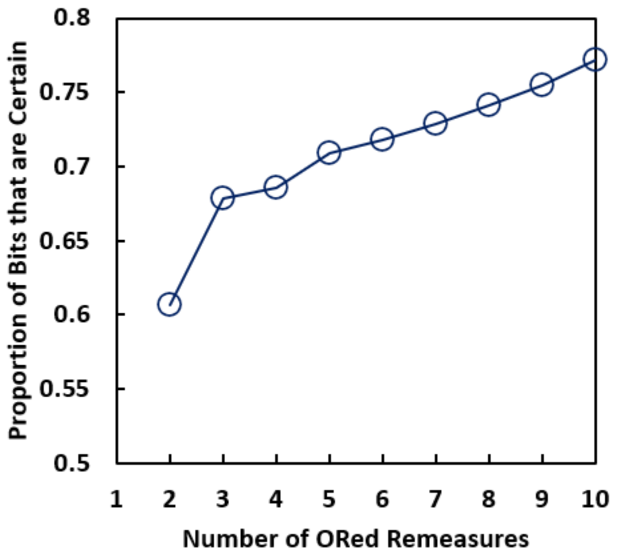


Figure 86, Effect of changing the number of ORed measurements on the proportion of 60 response bits that are certain across all challenge times.

Figure 85 shows the effect of changing the number of repeated measurements that are Ored together when creating the probability model or generating a response from the PUF. By allowing any of the repeated measurements to give a positive result when looking for the specified number of traps we can have a couple of positive effects on the PUF operation. Firstly, we can slightly further increase the maximum probability achieved in the probability model, making the chance of creating a '1' response slightly higher. Secondly, and more significantly, we can increase the gradient of the edges of the probability model. Increasing the gradient of these edges is highly desirable as it reduces the amount of time the probability of the PUF response will be in the unsure state between a ‘0’ and ‘1’ response. Figure 86 shows how increasing the number of Ored measurements from 1 to 10 can increase average number of certain bits at any challenge time by over 17%. The obvious downside to this ORing scheme is that repeating the measurements takes longer time as each individual measurement takes at least 0.2s to charge and measure. A balance has to be found for each application between the reliability, and efficiency of the PUF against the speed of the response. For a highly secure and important application it is probably best to have many ORed measurements as the time taken to gain the result is less important than the security. For a less important communication system it may better to reduce the number of Ored measurements in order to increase the communication speed.

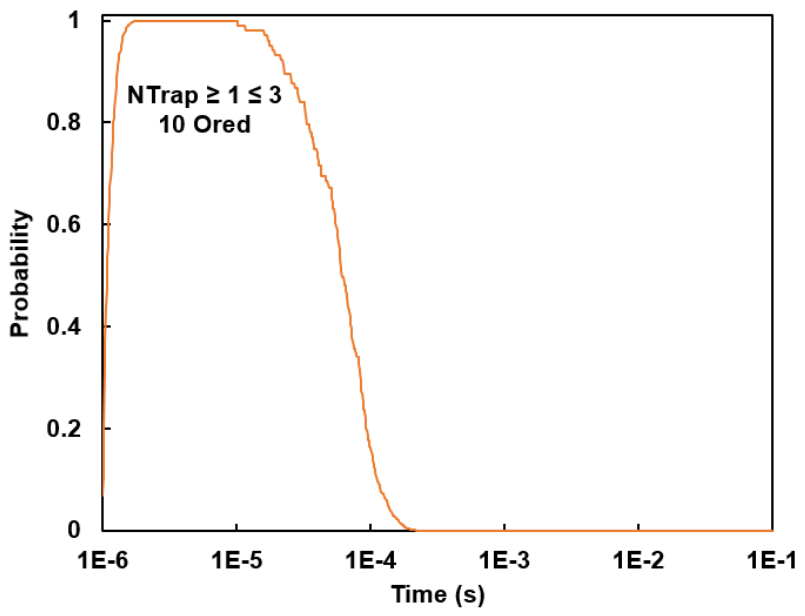


Figure 87, Device probability curve when 5 ORed measurements are used and the desired number of traps is between 1 and 3 traps.

Figure 87 shows the result of applying both the grouped traps and ORed measurements schemes to one device. We can see that the probability curve for this device has a maximum probability of ~100% and spends a reasonable amount of time at that value. We can also see the minimum probability returns to ~0% after some time which is desirable as different devices will do this at different times adding to the uniqueness of the PUF as a whole. The edges of the probability curve are quite steep with not a significant amount of time spent in the unsure region between 0% and 100%. For our chosen devices, these parameters of between 1 and 3 desired traps and 10 ORed measurements results in probability curves that have good maximum and minimum values and a good distribution of times at these values.

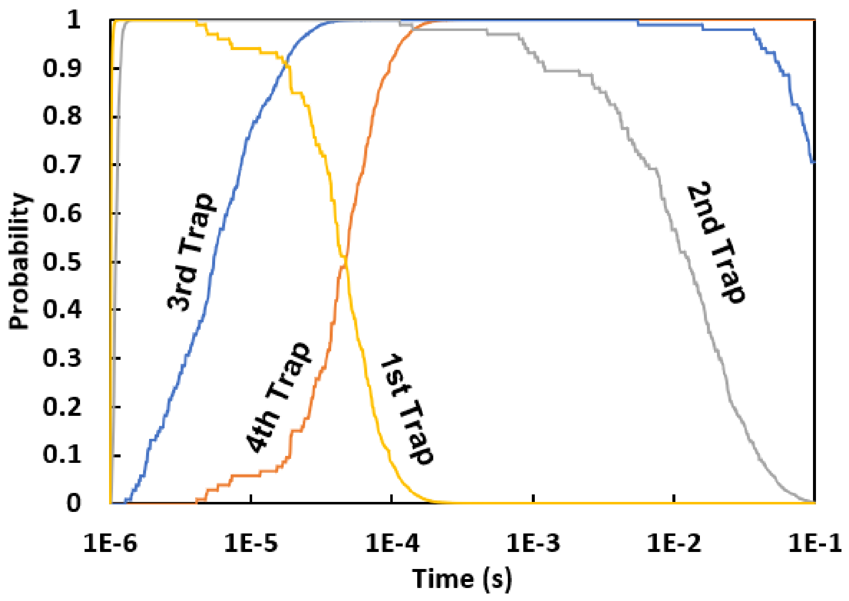


Figure 88, Effect of changing the starting trap number on a single device probability curve.

Figure 88 shows the effect of changing the starting trap number that the grouped traps are counted from. In the figure it can be seen that increasing the starting trap number shifts the probability curve to the right as a longer time passes before the number of traps observed is within the defined range. Tuning this parameter allows for shifting the distribution of probability curves within the measurement time window. The aim of adjusting the distribution is to have a more even balance of '1' and '0' results across the whole time window. Adjusting this parameter can also help reduce the amount of correlation between time and the expected result as the chance of having a specific result at the start and end of the measurement window can be changed by altering the starting trap number. In the rest of this work the starting trap number is selected as 1 as this gives a good distribution of '1' and '0' probability across the measurement window with the typical de-trapping events' timing observed.

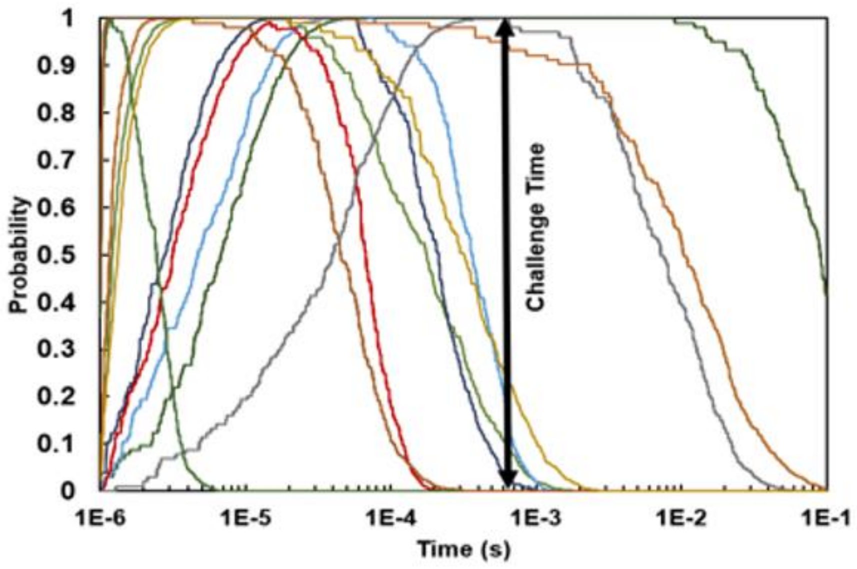


Figure 89, 10 device PUF probability model using both grouped traps and ORing schemes.

Figure 89 shows the probability curves of 10 devices that were created using the previously discussed parameters of 1 to 3 grouped traps, 10 ORed remeasures, and a starting trap value of 1. From these 10 probability curves we can see that all of the devices have regions that are either ~100% or ~0%, at different times. This means that every one of these devices can provide different responses to a challenge at different times, demonstrating the time dependence of the proposed PUF scheme. It also shows that the setting of the grouping and ORing parameters are well set for these given devices. We can also see that each device's curve is unique with the times at high and low probability being distributed at different times for different curves, demonstrating some degree of uniqueness of each. Having these key properties of the PUF scheme shown mean that the basic principles are solid. Next, we discuss a proof of concept circuit that is used to show the practicality of implementing this system in the real world.

5.2.2 PUF Node Proof of Concept Circuit

Here we outline and discuss one potential circuit that could be used to implement the proposed PUF module. This circuit is explored using a proof of concept circuit which has a single entropy source. The purpose of the proof of concept circuit is to show that it is feasible to create the whole PUF module but also do this in a way that can be done with limited equipment. The proof of concept circuit can be created using discrete components in a breadboard. The entropy is provided to the proof of concept circuit by probing a single MOSFET using a probe station, and analysis is performed through experimentally informed simulation, oscilloscopes, and an external analysis PC.

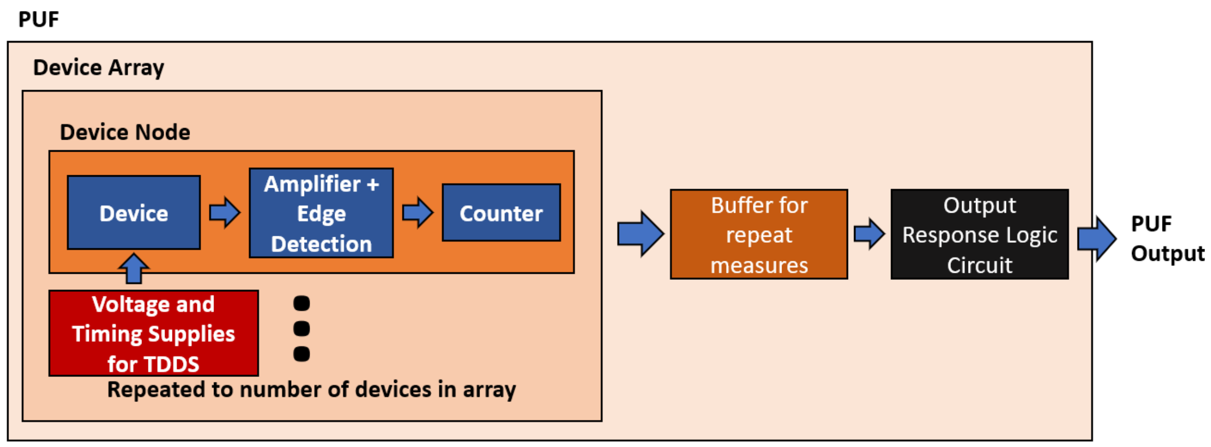

PUF
Device Array
Device Node
Device
Amplifier + Edge Detection
Counter
Voltage and Timing Supplies for TDDS
Repeated to number of devices in array
Buffer for repeat measures
Output Response Logic Circuit
PUF Output

Figure 90, Block diagram of key components required for proposed PUF module.

Figure 90 shows a block diagram of all the components that are required for basic PUF module operation, using the proposed time-dependent PUF concept. In Figure 90, the Device node block contains all of the components that are necessary to extract the number of de-trapping events observed in a single entropy source. Within the Device node block there is the Device block, the Amplifier and Edge Detection block, and the Counter block. The Device block refers to the MOSFET that is used as the entropy source for a single CRP bit. The Amplification and Edge Detection block contains the circuits that are required to detect the de-trapping events' timing, as well as digitise that information. The counter block is used to count how many de-trapping events are observed within the measurement time window.

The Voltage and Timing Supplies for TDDS block contains the circuits that are required to apply the TDDS voltages which are described in Figure 80 and the timing that is used to control how long the current should be measured for to have the correct challenge time. The Buffer for Repeat

Measurements is used to store the multiple repeat TDDS measurements that are required to implement the ORed remeasures methodology. The Output Response Logic block is used to determine whether the TDDS results stored in the Buffer for Repeat Measurements block should result in a ‘1’ or a ‘0’ response for each response bit.

In the proof of concept circuit, the Device block is created using one of the 28nm devices of the CSR wafer that was probed using the Cascade Summit probe station. Timing and voltages were applied to the Device block using Keysight 81150A and Keysight E3632A power supplies and signal generators allowing for accurate TDDS measurement. In the final circuit these could be replaced by low power adjustable voltage supply circuits, such as the one proposed by Guo et al. [135] .The amplification part of The Amplification and Edge Detection block was achieved using a Femto DHPCA-100 amplifier. The edge detection and digitisation requirements for the Amplification and Edge Detection block use circuits based on the same principle as was used in section 4.4.2 for the edge-to-pulse module. Figure 91 shows the proposed circuits for the edge detection and digitisation sections of the Edge Detection block. The edge detection is performed by a differentiator which converts all the edges present in the amplified Id signal into short pulses. For the PUF application we are only interested in rising edges caused by de-trapping, therefore the absolute value that is present for the TRNG design is not required here. A comparator is then used to convert the pulses from the differentiator into square pulses of the appropriate magnitude to be used as an input to the digital section of the circuit (square wave signal between 0-5V).

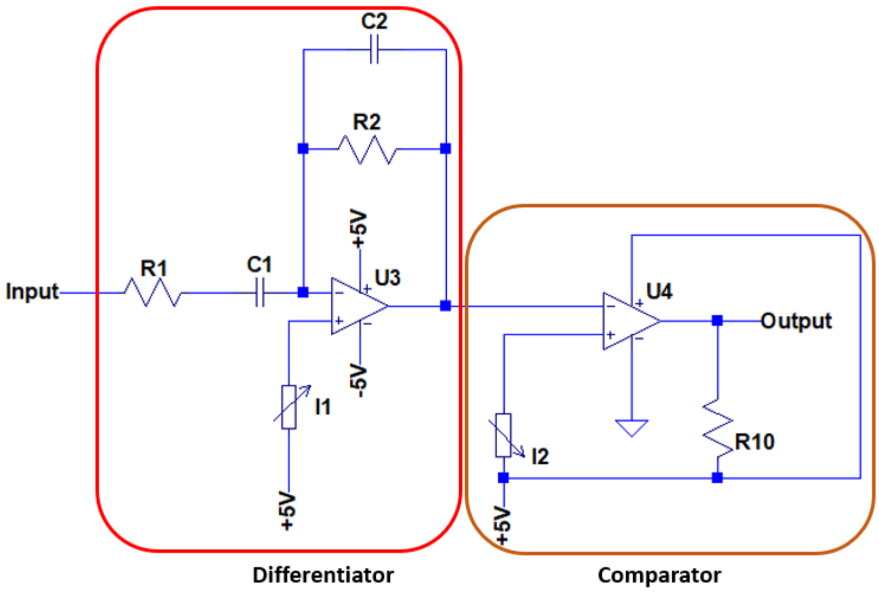


Figure 91,Edge detection and digitisation circuits within Amplification and Edge Detection block of PUF proof of concept circuit.

The functionality of the circuit shown in Figure 91 was verified using the Proteus 8 Professional simulation software to be used with simulation of the digital section. The circuit in Figure 91 was also verified experimentally with a breadboard circuit. The results from a single TDDS measurement for the Figure 91 circuit are shown in Figure 92.

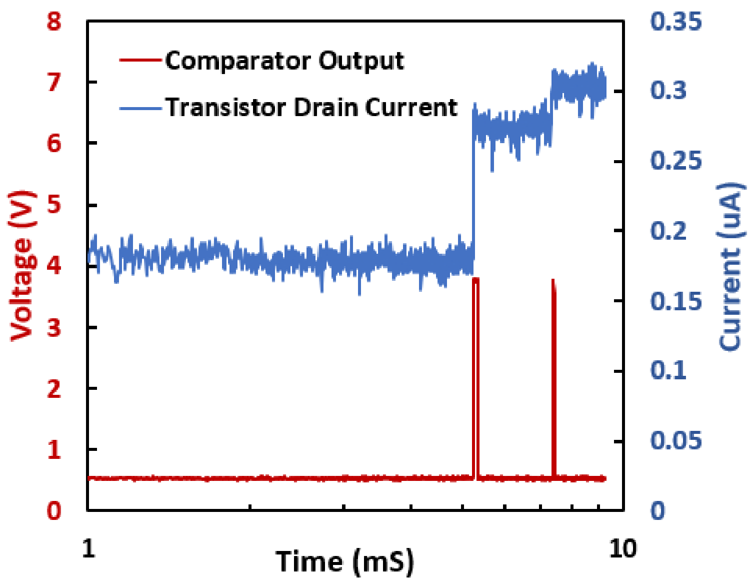


Figure 92, Results from a single TDDS measurement of the breadboard version of the edge detection block of the PUF circuit

The simulated version of the circuit was created in Proteus and a signal that is representative of an amplified Id signal from a TDDS measurement was used as an input. The amplified Id signal and the corresponding output signals from the differentiator and comparator are shown in Figure 93. In the figure the clear de-trapping events can be seen as the large steps in the amplified Id current signal shown in Figure 93(a). The output of the differentiator signal can be seen to have a clear negative pulse at the same time as each de-trapping event. The pulse is negative due to the inverting nature of the differentiator. the magnitude of each differentiator signal pulse is correlated to the size of the step in the amplified Id current, although all pulses have magnitude greater than several hundred mV. The output of the comparator is set so that every time there is a negative pulse in the differentiator output, the comparator outputs a positive 5V pulse. The rising edge of the 5V comparator pulses can be used as an input to the FPGA section of the proof of concept circuit. It can also be clearly seen that timing of the comparator pulses is almost identical to the timing of the de-trapping events, with the error of this timing being mostly attributed to the slew rates of the operational amplifiers.

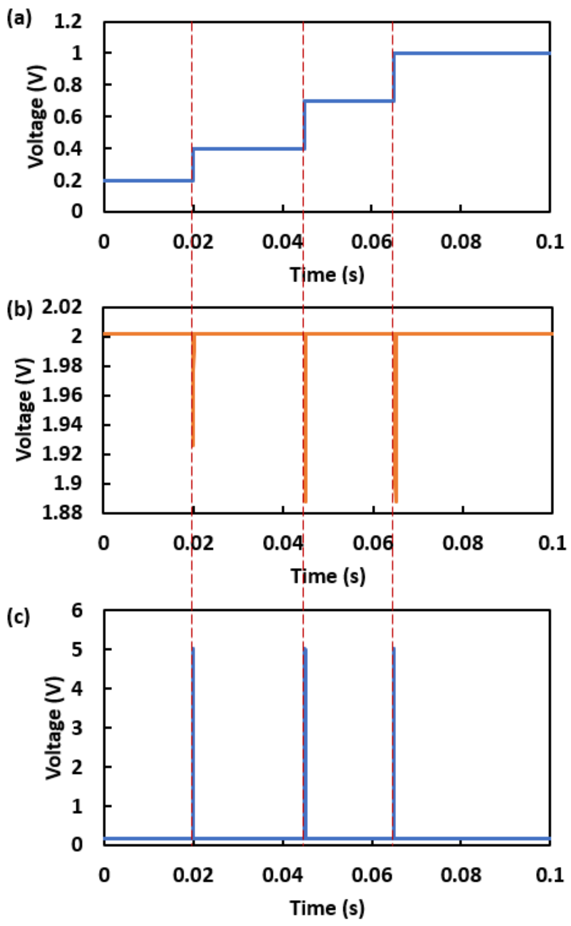


Figure 93, (a) Input amplified de-trapping signal, (b) output differentiator signal, and (c) output comparator signal of PUF analogue to digital circuit section of Amplification and Edge Detection block.

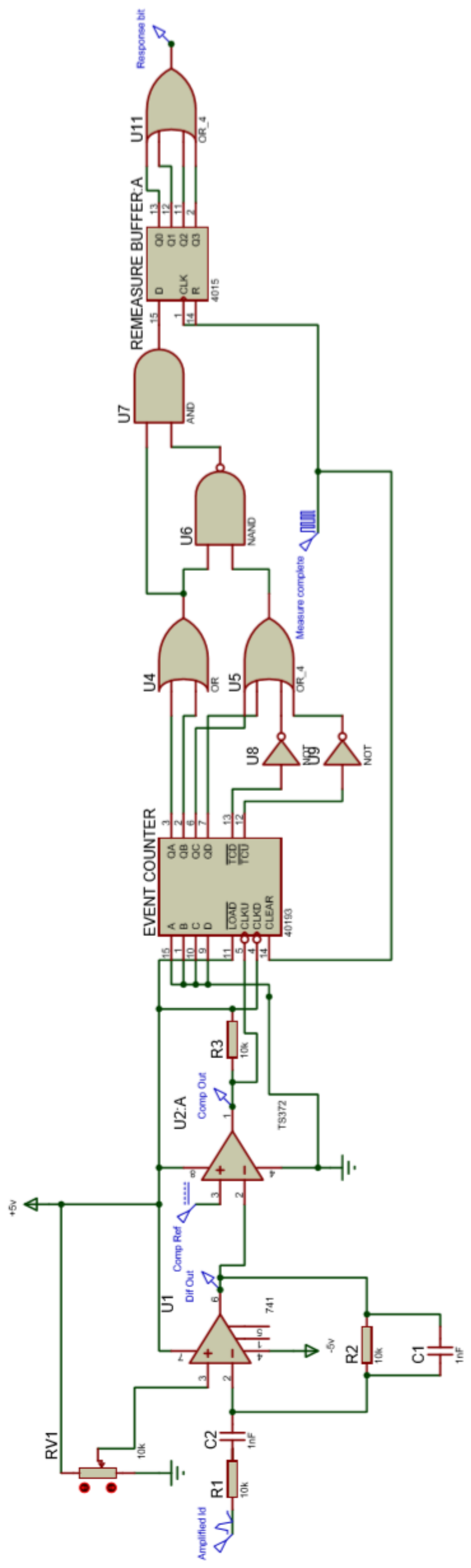


Figure 94, Circuit Diagram for PUF Proof of Concept Circuit. This circuit looks for 1-3 de-trapping events and 4 Ored remeasures.

Figure 94 shows the circuit diagram for the proof of concept circuit. This proof of concept circuit is the most basic implementation of the proposed PUF module, with one entropy source and fixed values for the number of grouped traps and ORed remeasures. In addition to the circuit shown in Figure 93, the circuit shown in Figure 94 contains circuits to create the counter, buffer for repeated measurements, and output logic blocks of Figure 90. The output rising edges from the comparator are used as an input to a the 'CLKU' pin of a 40193 counter, by doing this the counter will increment the stored value by one every time a de-trapping event is observed. Some basic logic circuits are also attached to the counter to determine whether the number of observed traps is within the defined range or not. In this example the circuit is configured to look for between 1 and 3 traps. If the number of traps is within this range then the logic will output '1', if not it will output '0'. For the real PUF design this logic can be created in hardware so that the defined number of traps is fixed, or this logic can be created using firmware such as an FPGA unit so the defined number of traps can be adjusted after manufacture. It is also possible to replicate this section using a software-based approach if a microcontroller is available for use with the PUF circuit. The buffer for repeat measures is implemented using a 4015 serial to parallel register. To use this register, we also generate a pulse every time a TDDS measurement is completed. This measurement completion pulse is then used to write the output of the previous logic stage into the register. The measurement completion pulse is also used to clear the counter from the previous measurement so only the observed de-trapping events from the current measurement are counted. The result from each measurement is then written to consecutive output bits of the register. In this example there are 4 parallel bits in the register, so the maximum number of measurements is 4. To increase the number of possible measurements to be ORed together, the number of parallel output bits can be increased. To determine the final response output of the PUF module a simple OR gate is used as any measurement stored in the remeasure buffer containing the correct number of de-trapping events will be considered a '1' response. A '0' response is only achieved if none of the TDDS measurements had the correct number of de-trapping events. After all measurements are completed the final response bit results can be sent to the device's transmission module to be sent to the server.

Figure 95 shows an example of a PUF reading where the generated response is '1'. In this example 2 TDDS measurements are ORed together and the defined number of de-trapping events to look for is between 1 and 3. The challenge time is set to be the full measurement window of 0.1s. In Figure 95(a) it can be seen that the first TDDS measurement has 3 de-trapping events and the second one has 4. In Figure 95(b) it can be seen that as soon as a measurement with the correct number of de-trapping events is completed the response bit output changes to its high value, giving a response of '1'.

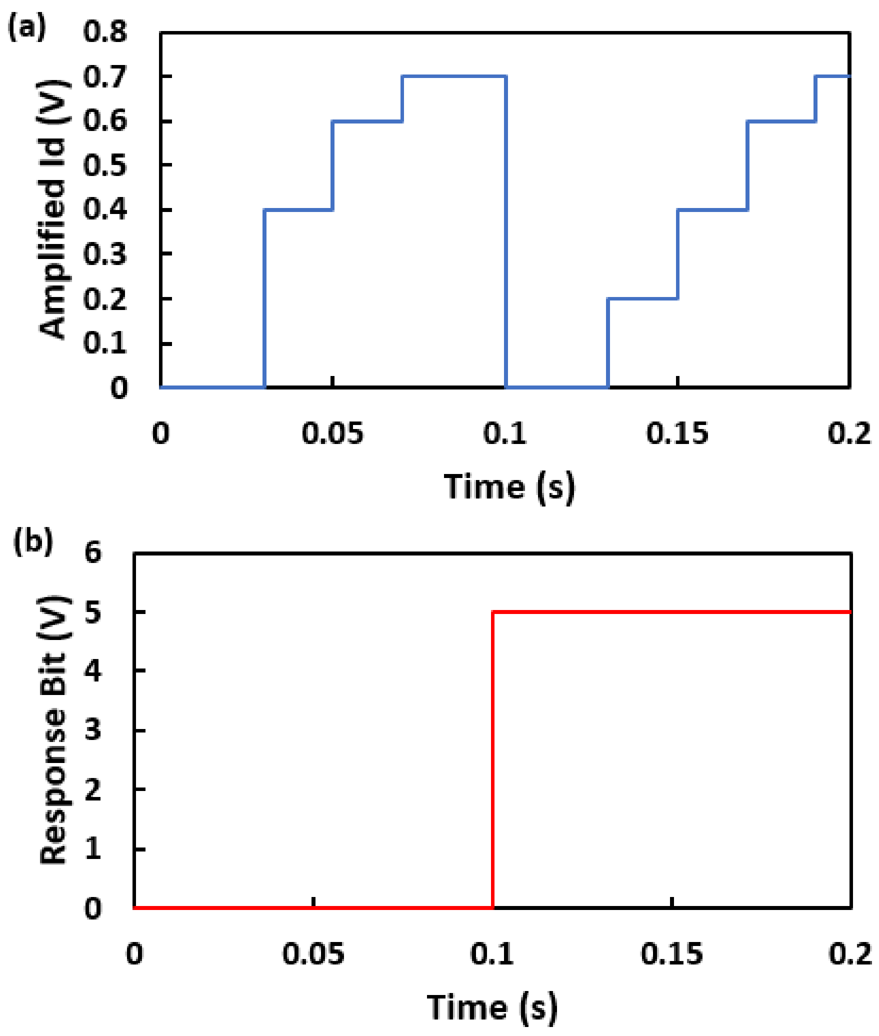


Figure 95, An example of a '1' response when (a) Amplified Id signal of 2 repeat TDDS measurements is used and (b) response bit output voltage when defined number of traps is 1-3 and two TDDS measurements are Ored.

Figure 96 shows an example of a PUF reading where the generated response is ‘0’. This example uses the same settings as the example in Figure 95 but with different emission times for traps within the entropy source. This time it can be seen in Figure 96(a) that neither TDDS measurement has an observed number of de-trapping events within the defined range of 1-3. The result of neither TDDS measurement having the correct number of events can be seen in Figure 96(b) where the response bit output remains at its low value giving a ‘0’ response.

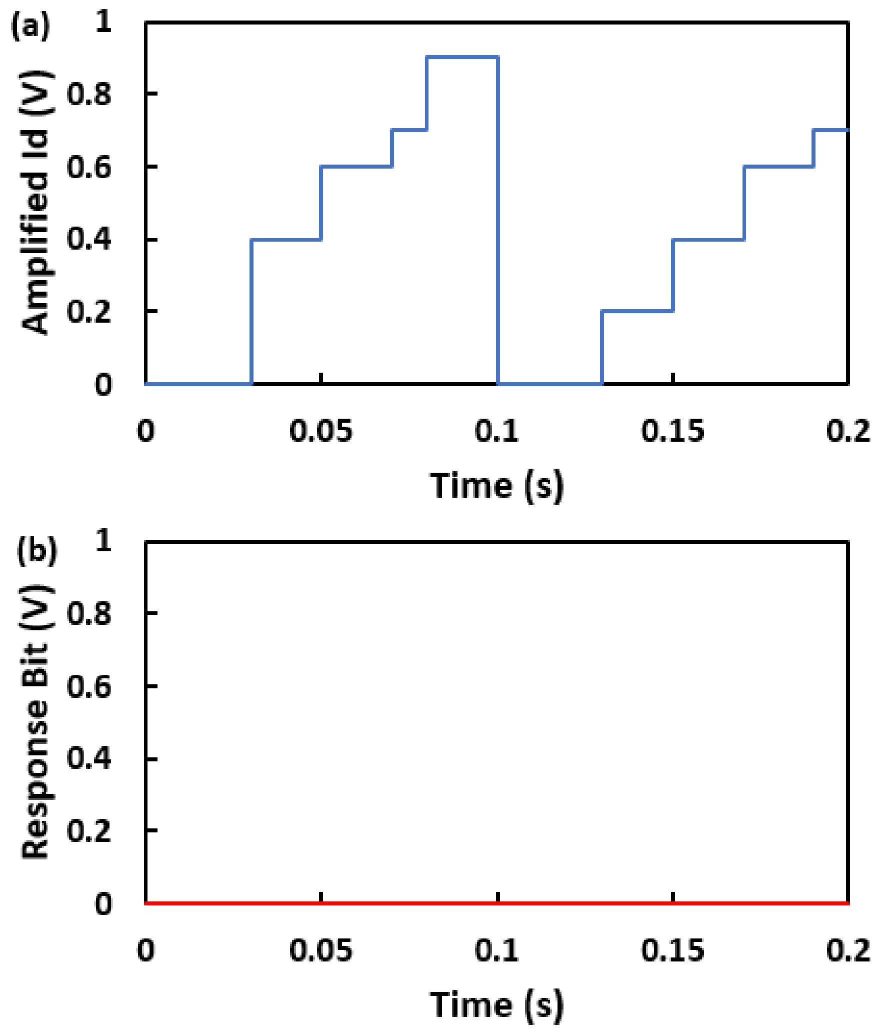


Figure 96, An example of a ‘0’ response when (a) Amplified Id signal of 2 repeat TDDS measurements is used and (b) response bit output voltage when defined number of traps is 1-3 and two TDDS measurements are Ored.

The two examples in Figure 95 and Figure 96 show that the proposed circuit for the PUF module is working as intended.

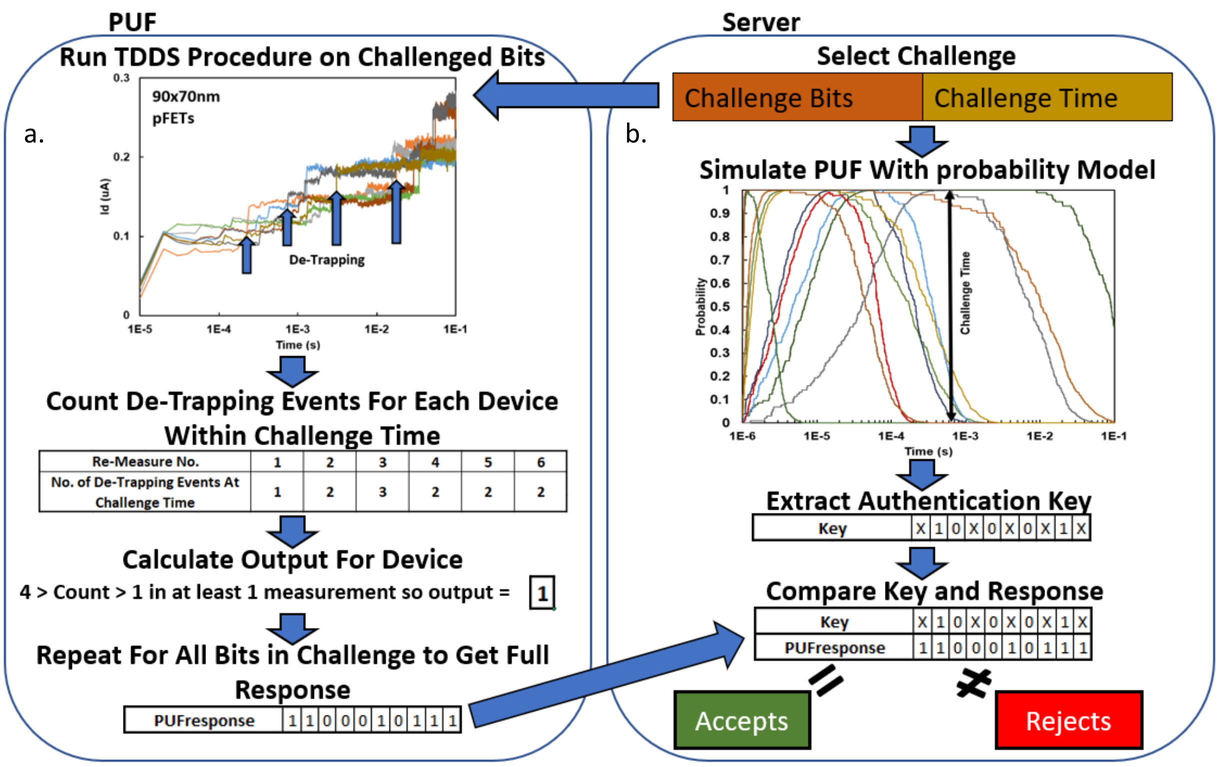
To create a full PUF with multiple entropy sources and response bits the circuit shown in Figure 94 can simply be repeated to give the desired number of response bits. It is also possible to create a full PUF with certain parts in parallel by repeating the PUF core components but then replacing the following sections with components that have bus input and outputs where each PUF core is connected to a different bus bit of the inputs and outputs of the counter, register and logic circuits.

5.3 PUF Based Authentication Protocol

In this section we discuss a possible authentication protocol that could be used with our proposed PUF. The proposed protocol aims to take advantage of the unique properties of both PUFs in general but also the new time-dependent element introduced by our newly proposed design. The protocol only requires the basic PUF module and transmission circuits to be available.

5.3.1 Basic Time-Dependent PUF Authentication Protocol

The proposed protocol only requires the basic PUF node to operate. Figure 97 shows the proposed interaction between the authenticating server, with the PUF probability model, and the PUF module itself, that will facilitate authentication of the PUF unit. The authentication may be initiated by either the PUF or the server. The following step is for the server to create a CRP, this is done by selecting a set of challenge bits to be used and a challenge time. These parameters can be selected randomly or chosen specifically if there is a requirement to do so. It is important to note that each CRP that is created should be unique and only used once to minimise the risk of any “man in the middle” attacks where a hostile party observes the communication and records previously used CRPs. After the CRP is created, the challenge section is sent to the PUF module and the server awaits the response. The response of the PUF module to the challenge is illustrated in Figure 97.a. When the PUF module receives the challenge, it performs TDDS measurements on the challenged devices. These TDDS measurements may be repeated several times if the ORing scheme is used. The PUF module then counts how many de-trapping events were observed in each TDDS measurement within the challenge time. If the number of de-trapping events observed in any of the measurements of challenge ‘bit’ device match the specified amount, the response is ‘1’, otherwise the response is ‘0’. When a response for every challenged bit is found they are all sent back to the authenticating server. The actions of the authenticating server that are performed while the PUF module is generating a response, and then how the server verifies if the response is valid or not are illustrated in Figure 97.b. Immediately after the challenge is issued to the PUF module the PUF server uses the probability models stored on it to calculate the expected response for each challenge bit sent. To do this the server loads the model for each bit and then checks the probability of finding the specified number of traps at the given challenge time. If the probability is certain then the expected response will be either ‘1’ or ‘0’ depending on whether the probability is ~100% or ~0%. If the probability is anywhere in the middle of these values, then the expected response will be set to uncertain. When this has been performed for all challenged devices then an authentication key is created containing all of the expected results. The server then waits to receive the response bits from the PUF module. When the server receives the response from the PUF module, it compares the expected results to the received results. For the certain bits (expected ‘1’ and ‘0’ results), the expected and the received results must match for authentication to be successful. For the uncertain bits (probability not ~0% or 100%), the received response does not need to match the expected result. Essentially, the uncertain bits are ignored by the authenticating server.


PUF
Run TDDS Procedure on Challenged Bits
a.
90x70nm
pFETs
Id (uA)
De-Trapping
Time (s)
Count De-Trapping Events For Each Device Within Challenge Time
Calculate Output For Device
4 > Count > 1 in at least 1 measurement so output = 1
Repeat For All Bits in Challenge to Get Full Response
Server
Select Challenge
Challenge Bits
Challenge Time
b.
Simulate PUF With probability Model
Probability
Challenge Time
Time (s)
Extract Authentication Key
Compare Key and Response
=
≠
Accepts
Rejects

Re-Measure No.	1	2	3	4	5	6
No. of De-Trapping Events At Challenge Time	1	2	3	2	2	2

Key	X	1	0	X	0	X	0	X	1	X
PUFresponse	1	1	0	0	0	1	0	1	1	1

Figure 97, Illustration of authentication protocol using the proposed PUF methodology. (a) The steps that the PUF module runs during an authentication cycle. (b) the steps the authentication server performs during an authentication cycle. It is assumed that the complete PUF probability model has been distributed to the authentication server before authentication begins. For this example, the desired number of observed de-trapping events is considered to be <4 but >1. It is also set so that each bit is remeasured 6 times so that the ORing scheme can be used.

Using our experimental sampling frequency of 100KHz, we have 10,000 different time points within the 0.1s TDDS measurement time window that could potentially be used as the challenge time. There are some limitations on this that reduce that number due to correlations between some time values, as well as limitations of the measurement hardware available for PUF modules. For example, the first few measurement points have a much higher chance of giving a ‘0’ response than the later time points as de-trapping events take some time to occur. To try and avoid some of the correlations issues the challenge times possible can be limited to the range where the bit bias is not close to 0 or 1. To further remove visible correlations the challenge times should not directly correlate to the real measurement time through the use of some scrambling or hashing function that would make it much harder for a potential attacker to find each challenge bit’s time dependence through observation. Figure 98 shows how the bit bias towards ‘1’ varies across the TDDS measurement window across 60 PUF bits. From

Figure 98 it is clear that at least the first ~5 measurement points should be avoided as possible challenge times as the bits are very strongly biased towards ‘0’. Figure 98 also shows how there is some correlation between time and the expected response, which supports the suggestion to obfuscate the real measurement time from the specified challenge time through the use of some hashing type of scheme.

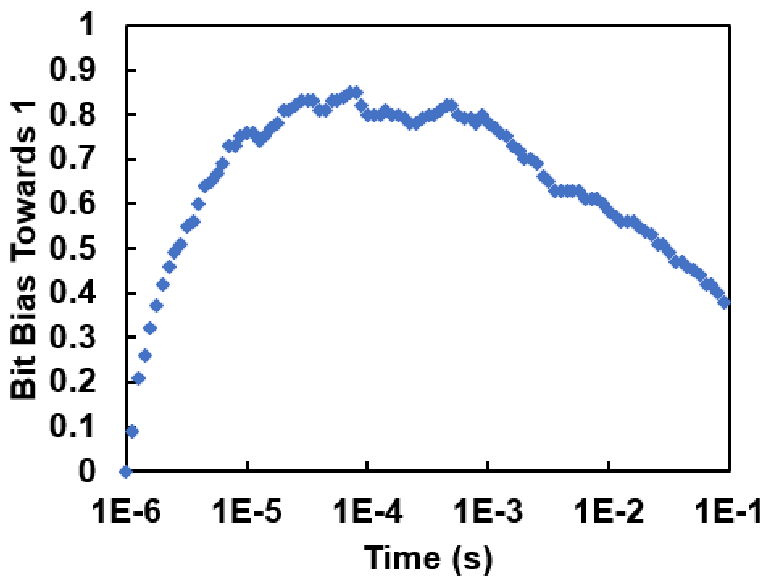


Figure 98, Bit bias towards ‘1’ across 60 PUF bits over the whole measurement time window.

There are also two other limitations of this implementation, the first of which is related to the uncertain bits. As the uncertain bits are not used for authentication these bits would not have to be guessed correctly by any attacker trying to guess the response. This effectively has the effect of shortening the key length by the number of uncertain bits. To compensate for this the overall key used should be long enough that the key length minus the number of uncertain bits is at least as long as the minimum required key length. From the number of Ored measurements analysis shown in Figure 86, each PUF response that is used should be at least 30% longer (when using 5 remeasures) than the minimum required key length to ensure security against guessing attacks.

The other limitation of this protocol is that the responses are transmitted in plain text. This means that the PUF module and device that it is embedded into does not need any complicated mathematical or cryptographic functions. However, it also means that any potential attackers can collect previously used CRPs with relative ease. To avoid this causing a serious reduction in security each entropy source can only be used at each possible challenge time once. If this is not done, then any attacker who has previously observed this response bit will know what the expected response is at the previously used challenge time. This would mean that even if the combination of challenge bits is unique, a potential attacker could already know the response from many of the selected bits if they have been sampled at the given challenge time previously. To avoid this each challenge bit should only be used at any given challenge time once. This means that in order to expand the number of CRPs beyond the number of unique challenge times the number of challenge bits has to be increased by the minimum response length which will give a new unique set of challenge bits at every available challenge time. Figure 99 shows this relationship between the number of available response bits and the number of available CRPs. This relationship makes increasing the number of CRPs available to this PUF quite inefficient compared to the other proposed protocols that are discussed later in this section.

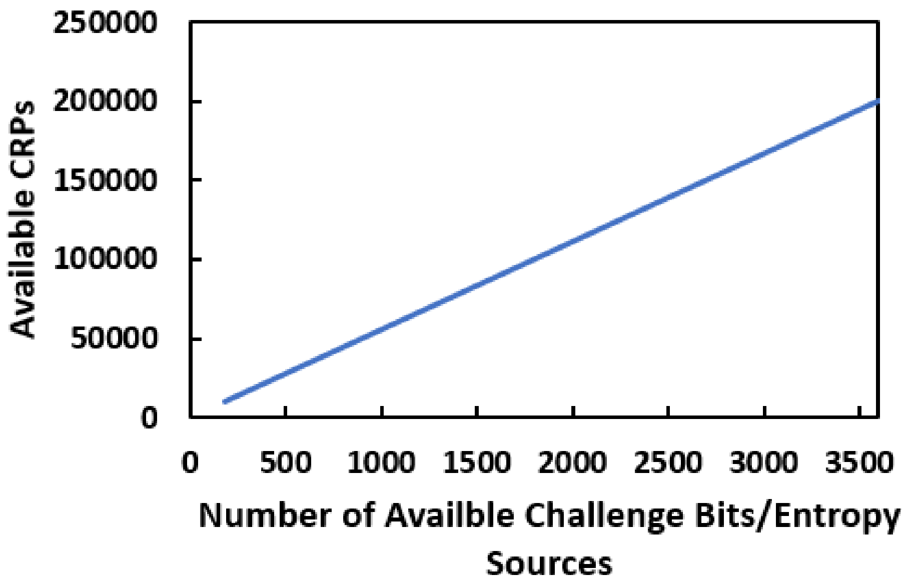


Figure 99, Number of CRPs available with a given number of challenge bits/entropy sources using the Basic Time-Dependent PUF Authentication Protocol. Sampling rate of 100kSa/s for 0.1s TDDS measurements and 128-bit key length (180 bits including uncertain bits).

In conclusion for this protocol, it should only be used when the device within which the PUF is embedded has very limited available power and operations as this PUF protocol is very simple. It should also only be used when the required number of CRPs is small (several thousand), and the security of the device is desired but not critical. The best use for this proposed protocol is for local network applications where the likelihood of there being malicious parties is much lower, but authentication of different entities is still desirable.

5.4 Performance Analysis of Proposed PUF Scheme

5.4.1 Analysis of Key PUF Characteristics

In this section we look at the performance of our proposed PUF in terms of the hamming distances as well as evaluating the design against the key points outlined in section 2.2, which include constructability, evaluability, reproducibility, uniqueness, identifiability, physical unclonability, unpredictability, mathematical unclonability, one wayness, and tamper evidence. We discuss each of these characteristics individually as well as look at key results for specific characteristics.

Constructability: this characteristic describes the feasibility of physically manufacturing the proposed PUF. This parameter is hard for us to evaluate quantitatively due to limited facilities available for circuit design. Section 5.2.2 coverers the design and analysis of the PUF node's circuit requirements. Figure 90 shows the complete block diagram of required components to create the full PUF module. The proof of concept circuit shows how the PUF module could be created. In the proof of concept circuit, the voltage supply and timing signals were provided by external sources. In a real implemented PUF module the voltage and timing would have to be provided either by the PUF itself or by the device it is embedded within. For the voltage supply a linear power supply with adjustable output voltage is required. Voltage supplies that can provide the required values of 0.4-1.2V have been proposed previously [135] and could be used to provide the voltage required for the TDDS measurements performed in this work. The timing requirements of the circuit can be met by utilising any clock signal that is available in the device within which the PUF is embedded. The clock signal can be divided and counted to give intervals that give the required timing of the TDDS measurements. The timing interval of the TDDS measurements is directly correlated to how many different points in time can be used as challenge times. In this work 100KSa's was used which requires a timing signal with a frequency of 100KHz. 100KHz is a slow clock speed for digital circuits so is highly achievable for this PUF design.

The proof of concept circuit designed in this work only has a single response bit (device node). However, the device array is created by copying many device nodes to create a large set of possible response bits and thereby increase key length. As the proof of concept circuit has shown that the device node is constructible then the device array is also constructible and the efficiency depends upon the size of each device node, as well as the efficiency of circuits created to select specific nodes within the array for testing. Also, it is likely that a single voltage supply for TDDS block can be used to supply all of the Device Nodes so this should not have to be copied across the Device Array. The Buffer for Repeated Measures Block is used so that the ORing multiple remeasures scheme can be achieved. If this scheme is not used, then this block can be removed.

Finally, we discuss the feasibility of creating the probability model that is stored on the authentication server. To create this model, it must be possible for the manufacturer (or potentially user) to measure the counter output of the CRP device against the time when TDDS measurement is applied. This should be repeated at least 100 times for every CRP device within the device array. The probability model can then be constructed on an analysis PC and distributed to the authenticating server. All of these steps are very feasible to achieve, although the ability to directly read the output counter of the CRP device should be removed in some way before the PUF is deployed to stop any potential attackers from easily building their own model (further discussed in the unclonable sections).

In conclusion, all components of the PUF are constructible. The only limitations may be in the size and efficiency of the circuit when a large number of CRP bits are required as repeating the Device and Amplification and Detection blocks may be expensive in terms of design area and cost if it cannot be parallelised in some way.

Evaluability is the characteristic that refers to the PUF scheme's ability to generate and evaluate CRPs. In section 5.3 we discussed three proposed schemes that can be used to generate and evaluate CRPs using the proposed PUF circuit and probability-based model of that PUF circuit. The proposed schemes clearly demonstrate how an authenticating server can evaluate the response to a given challenge in order to authenticate the identity of a given PUF circuit. The proposed PUF design therefore has a high degree of evaluability.

Reproducibility: this characteristic refers to the ability of a single PUF instance to provide consistent responses to a given challenge. The key form of analysis for this characteristic is the intra hamming distance between responses to the same challenges. For our specific proposed PUF this is slightly more complex than with traditional PUFs. This is because our proposed design is probability based, so we can have both certain and uncertain bits in the response, where the certain bits are used for authentication and the uncertain bits are used as 'salt' to obscure the true CRP to any observing attacker. This obviously means that the certainty that is required for a bit result to be certain has an impact on the intra hamming distance of the PUF. Figure 100 shows how changing the tolerance of the certainty changes the mean of the intra hamming distance between certain PUF responses. It can be seen that decreasing the certainty exponentially increases the intra hamming distance away from the ideal value of 0%. The downside to increasing the certainty is that more bits will be considered uncertain and therefore the number of bits actually used for authentication will decrease, potentially decreasing security and bit efficiency. For the following results we use a certainty tolerance of 1% as this value should maintain a low intra hamming distance while also allowing for more bits to be considered certain for authentication. The proportion of certain to uncertain bits for a given challenge across 40 PUFs with these parameters has already been shown in **Error! Reference source not found.**. It can be seen that the proportion of uncertain bits is quite significant for most PUFs, so it is in the interest of efficiency and performance to keep the tolerance of the certainty at ~1%.

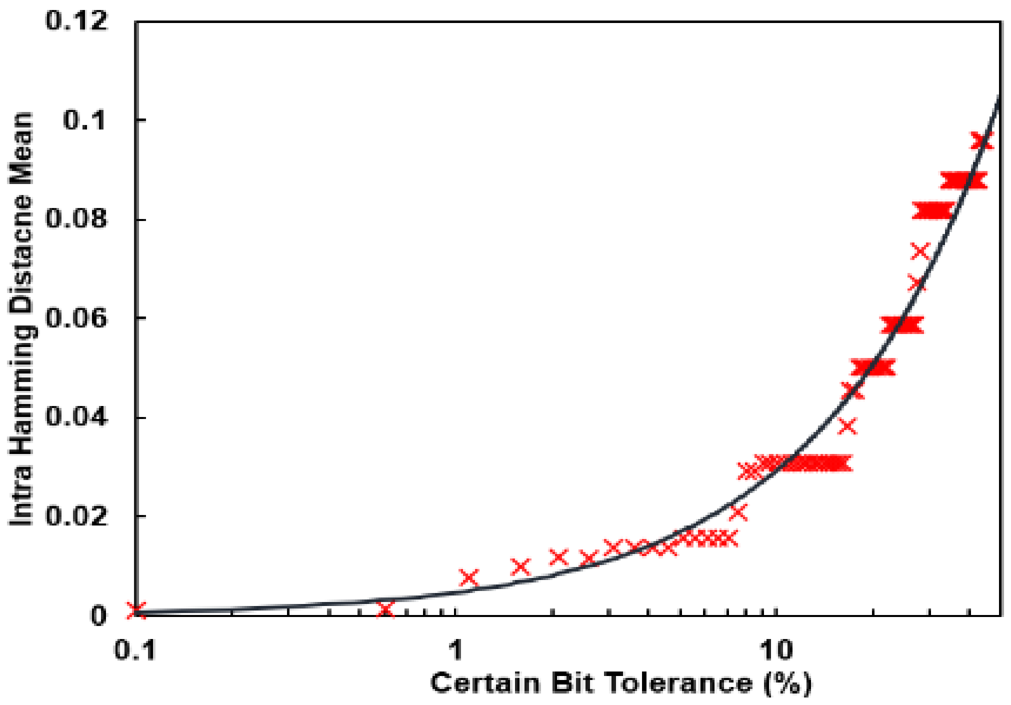


Figure 100, Experimental results showing dependence of the mean of the intra hamming distance, of only the certain bits, on the certainty tolerance of certain bits.

Figure 101 shows the full distribution of the intra hamming distances between responses of the proposed PUF using the parameters that have been selected earlier in this section. Figure 101 shows both the intra hamming distance for just the certain bits, and for the uncertain with certain bits together. The effect of including the uncertain bits is clear, the mean of the intra hamming distance is significantly increased when they are included. With only the certain bits, the hamming distance has an average of less than 2%, which is comparable to the contemporary designs shown in Table 10. A low certain bit intra hamming distance is very positive as these bits are the ones that are used for authentication. With the uncertain bits included the mean increases to ~18%, which shows how some of these bits are changing with repeated measurements. The distribution of the hamming distance with only certain bits is also much narrower than the one with uncertain bits included.

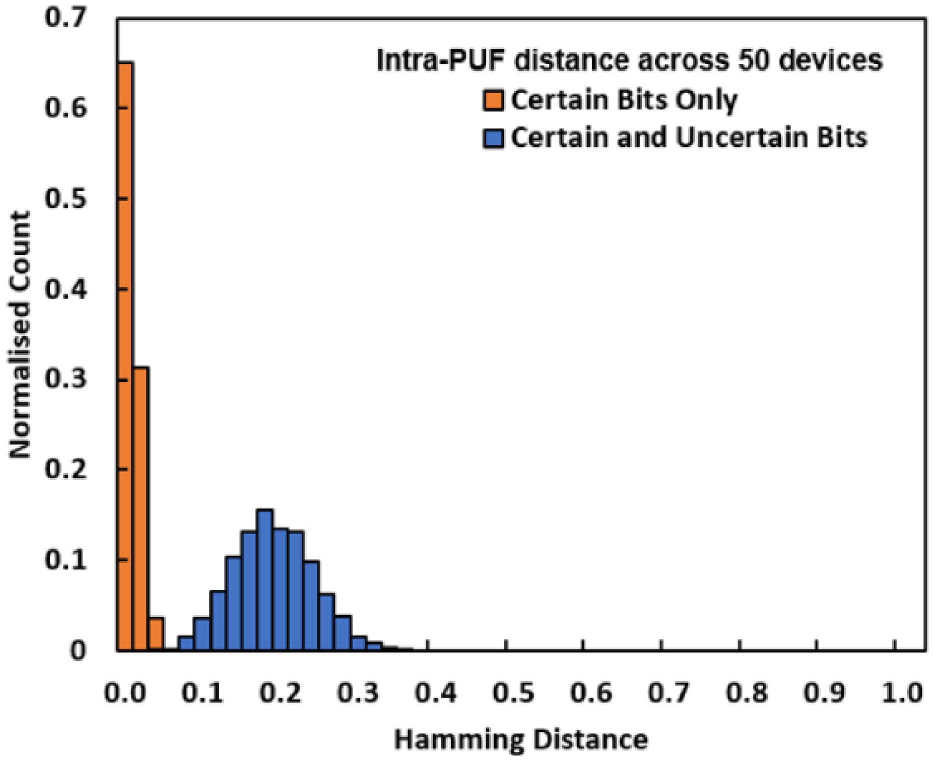


Figure 101, Experimental results for intra hamming distance of certain bits used for authentication and the intra hamming distance including uncertain bits which are not used for authentication but obfuscate which bits are used for authentication if observed by any 3rd party.

The main source of error, that increases the intra hamming distance of certain bits, is RTN traps that overlap in timing with other traps as well as traps which can still have capture events at the Vg measure voltage, causing multiple de-trapping events to be observed for that one trap. Another cause of unreliability in the reproductivity is other sources of noise within the Id of the device (such as within device fluctuation and white noise) which are more prevalent in the research samples that we are using. The other noises are lesser in commercial grade wafers, as can be seen in the results of the TDDS-based PUF as reported by Chen et al. [84]. Therefore, we can expect to see some improvement of our PUF's performance when it is implemented using commercial grade devices. The primary reason for this would be a significant improvement of the signal to noise ratio, resulting in fewer de-trapping events being missed or other noise sources being counted as de-trapping.

Even with the current limitations, our proposed PUF has comparable reproducibility to contemporary PUF designs as shown later in Table 10. With these findings we can say that our proposed PUF has an acceptable level of reproducibility, although further improvements could be made. Currently, some error correcting scheme should be employed with our proposed design in order to avoid the ~1% of bits in error causing authentication to fail.

Uniqueness is the next key characteristic that a PUF must have. Uniqueness for a PUF is defined as no two PUF instances having the same challenge. The typical measurement of this characteristic is the inter hamming distance between different PUF responses to a given challenge. Figure 102 shows one example of the simulated TDDS measurements on which this section is based. The τc, τe, and magnitude of traps as well as the number of traps per device are selected randomly within ranges that are based upon the typical observed values from our experimental results. The timing distributions of the capture and emission events are based on the same hidden Markov model as used for simulated RTN events in section 4.4.

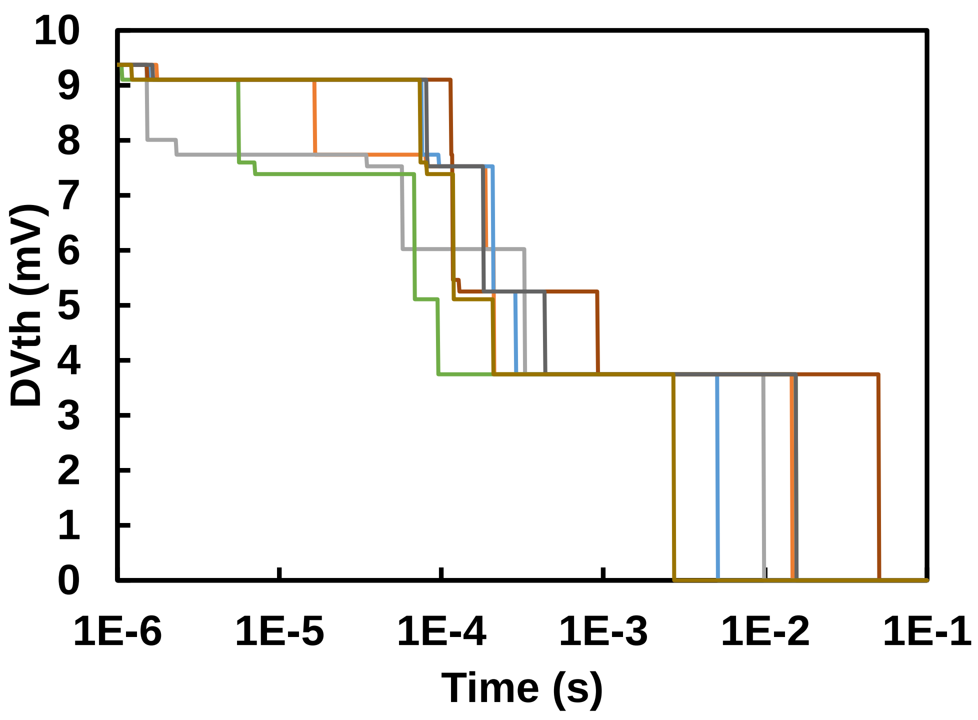


Figure 102, Simulation of Id current during the measure Vg (0.4V) phase of 6 TDDS measurements on a single device.

To find the inter hamming distance between different PUFs we simulated 40 PUFs, each with 50 response bits. A random challenge was selected and the response from every PUF was calculated. The hamming distance between each response was then found to give the inter hamming distance. Figure 103 shows the inter hamming distance between all response bits (certain and uncertain) (as used in the basic and public key-based proposed protocols) and we can see that the mean of the hamming distances is close to the ideal value of 50% (50.08%), showing that each PUF response bit has on average a 50% chance as being the same as the corresponding response bit from another PUF. The distribution is also fitted with a binomial power density function (PDF), as outlined by Daugman [136]. This binomial fitting has 52 degrees of freedom. The data suggest that it is highly unlikely that two different PUF instances will provide responses differ in less than about a third of their information. If an attacker was observing the communication between the PUF and the server, these are the bits observed when using the basic or public key proposed protocols. As these bits are very close to ideal inter hamming distances for a PUF, it would be very difficult for an attacker to predict another PUF instance's responses, even if one PUF unit is compromised. Figure 104 shows the inter hamming distance between only the certain bits of the different PUF instances. The inter hamming distance between only the certain bits is not as ideal as with all the bits, only being 40%.

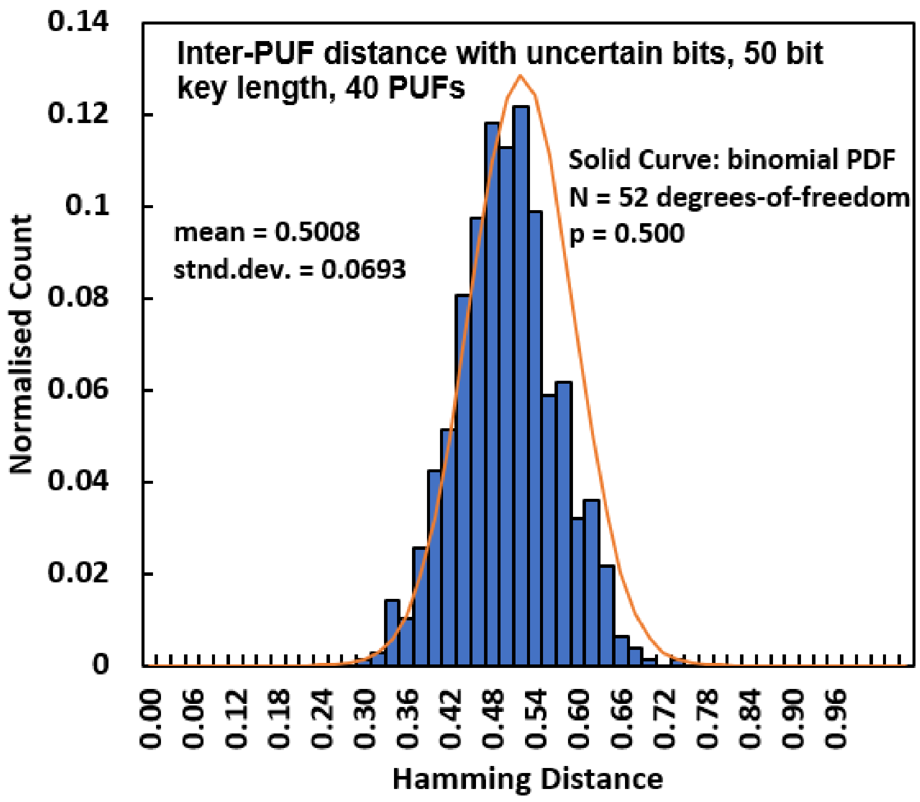


Figure 103, Simulation results of the inter hamming distance between different PUFs tested with the same challenged bits at the same challenge time including certain and uncertain response bits. The histogram fits very well with a binomial distribution of P=0.5 and N=52 degrees of freedom, as shown by the solid curve.

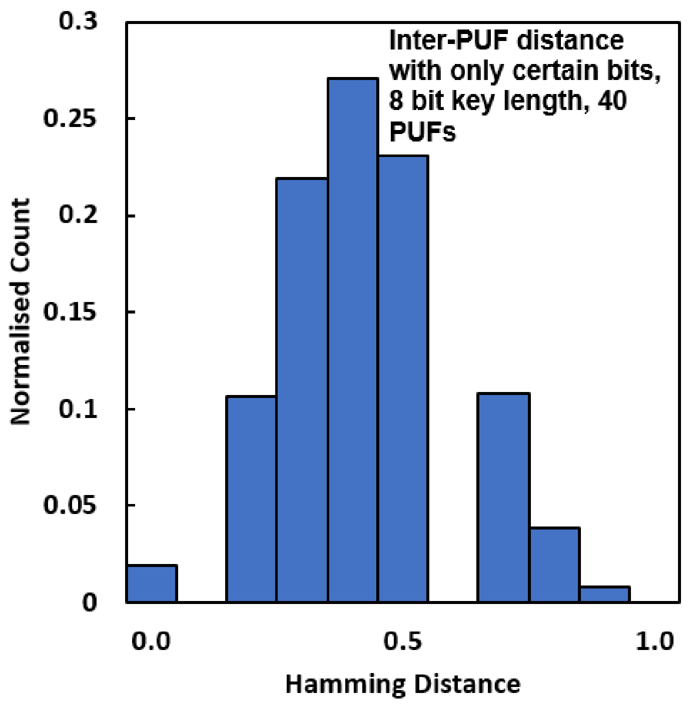


Figure 104, Simulation results of the inter hamming distance between PUFs tested with the same challenge bits at the same challenge time with only the certain response bits.

Overall, the inter hamming distance results from these simulations show that there is a high degree of uniqueness in the responses from different PUF instances. From these finding we can say that the proposed PUF has acceptable uniqueness and the inter hamming distance for all bits is comparable with, or in excess of, contemporary PUF designs as shown later in Table 10.

Identifiability is another key parameter of a good PUF and is defined as a PUF that has both good reproducibility of responses for each instance and uniqueness of responses from other PUF instances. The most common way to measure this is the distance between the intra hamming distance and inter hamming distance of a PUF design, with the ideal value being 50%. The distance between the mean of the hamming distances of our PUF is ~49% which is very close to the ideal value. To further illustrate the proposed designs identifiability, the binomial fitting shown in Figure 103 was used to predict the chance of a false positive result during authentication when various Hamming Distances are used as the criteria to validate each response. The results from this false positive estimation are shown in Table 9 and show that when the Hamming Distance criteria is set to below ~0.15 the chance of a false positive is extremely low. These results in combination with the already discussed reproducibility and uniqueness show the proposed design has a good degree of identifiability.

Table 9, Binomial fitting in Figure 103 used to estimate chance of a false match occurring when HD is used for verifying response at various HD criteria.

HD Criterion	Odds of false match
0.11	1 in 233,967,864
0.15	1 in 6,294,219
0.2	1 in 77,760
0.25	1 in 7355
0.3	1 in 447

Physical Unclonability is an important characteristic property of a good PUF. It should be impossible for anybody to create a PUF instance that has the same CRPs as any other PUF instance, this is to avoid attacks where a hostile party could use the copy of a PUF to impersonate another. Our proposed PUF has a strong case when displaying physical unclonability as it is based on the timing of de-trapping events of defects within nano-scale devices. This timing is based on the same principle as that for the entropy source for our proposed TRNG in Chapter 4 - , where we showed that these de-trapping events' timing is random and unique. As this timing is randomly distributed and cannot be controlled then it is unfeasible for anyone to make a physical clone of a PUF constructed using this concept.

Unpredictability of the PUF responses is also important as it should not be possible for a potential attacker to guess a response to a given challenge, even if they know many previously used CRPs. The most common way to analyse the ability of the PUF to be unpredictable is to look at the inter hamming distance between the responses from a single PUF instance. If the PUF responses are uncorrelated and therefore unpredictable then the mean of the inter hamming distance between them will be 50%. Figure 105 shows the inter hamming distances between 39,231 responses from one simulated PUF using our proposed design which has 16 selectable challenge bits and each CRP being 8 bits long. 10 different challenge times are available for each challenge bit and these times are distributed logarithmically between 10us and 0.1s. The mean of the inter hamming distances for this PUF is 50.14%, which is very close to the idea value of 50%. This almost ideal mean combined with the distribution shown in Figure 105, which shows that the vast majority of responses are different to any other response, shows that our PUF has a high degree of unpredictability. The fact that some responses are the same is not unexpected with only an 8-bit key, resulting in 256 possible unique results, and the total possible number of unique challenge combinations being 39,231.

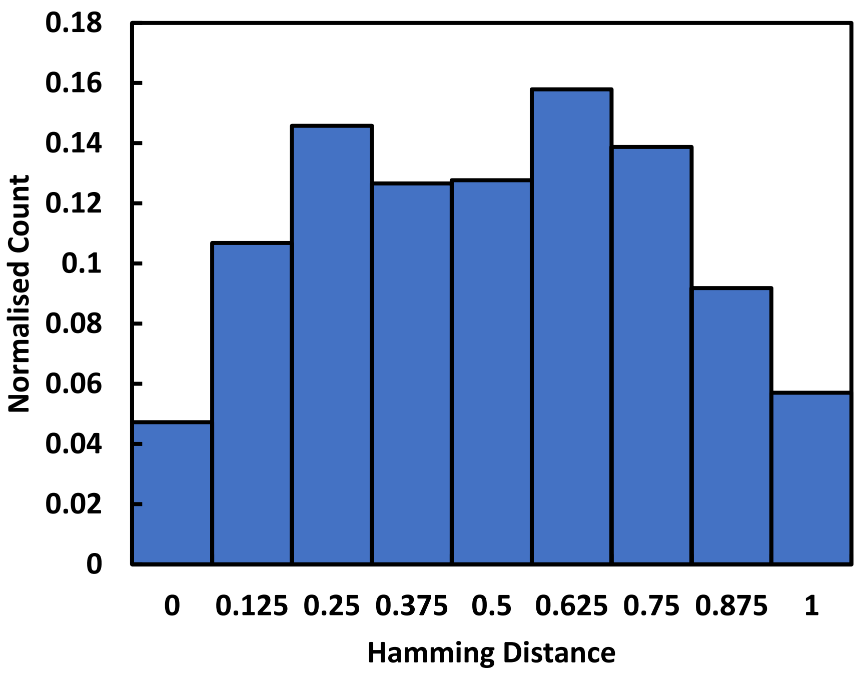


Figure 105, 500,001 Inter hamming distances between responses from a 16-challenge bit PUF with a key length of 8 bits.

Mathematical Unclonability is a characteristic that refers to an attacker’s ability to create a mathematical model of the PUF behaviour when they have total access to the PUF’s CRPs or even the entropy source itself. The best way to increase mathematical unclonability is to increase the total number of CRPs so that it is unfeasible to analyse them and to have an entropy source that is too complicated to analyse. This is a property that no PUF has been able to show in practice and is very difficult to achieve. For our proposed design, we do offer some improvements as we increase the potential number of CRPs for any given number of challenge bits, thereby making the modelling more difficult. As our entropy source is probability based it would also take many measurements for an attacker to be able to construct their own model. Overall, our proposed PUF cannot be said to be fully mathematically unclonable, however it is not worse than previously proposed PUFs and may offer some improvement if the proposed public key protocol is used, as the PUF can only be activated if the correct digital signature is used.

One Wayness is the characteristic that refers to an attacker's ability to know what specific challenge should be used to give a known response. For PUFs One Wayness can be improved by increasing the total number of CRPs or by making it very difficult to find what challenge gives a certain response. Our proposed PUF offers some key improvements on this characteristic. The proposed design increases the number of potential CRPs for a fixed number of challenge bits, thereby increasing One Wayness of the PUF, as defined by Maes et al. [64]. If the challenge time is also hashed so that it does not correspond to the real measurement time this also makes it more difficult for an attacker to try and guess a specific challenge to give a known response. This is because any correlation between response and measurement time is obfuscated through this hashing. In section 5.4.2 we show how the probability model-based storage of CRPs makes it much more feasible for servers to store a very large number of CRPs, further increasing One Wayness as the number of CRPs that can be used is increased.

Tamper evidence is the final characteristic that is desirable for a PUF to have. This refers to it being clear if there has been an attempt to modify the behaviour of the PUF through physical tampering. For our proposed PUF, it is impossible to create entropy sources with any specified parameters so it would be obvious if any of the entropy sources are altered as this would cause a mismatch between the probability model stored on the server and the responses from the PUF causing the PUF to be locked out. The real issue for our proposed design would be if an attacker tampered with the PUF so that they could measure many responses from each entropy source within the PUF to create their own probability model. To avoid this a tamper evidence-based system could be employed that could show the server if the entropy sources have been measured without proper communication with the server. Another way this problem could be minimised is to have enough entropy sources so that it is unfeasible for an attacker to measure enough of them to have a reasonable chance of predicting a future CRP. A good way to do this would be to physically limit how may entropy sources can be measured simultaneously after manufacture.

In conclusion, our proposed PUF shows solid evidence that it displays each of the key characteristics that are required for PUF operation. This means the overall concept of our PUF is solid and we can move on to look at how our PUF compares to previously proposed designs.

5.4.2 Comparison with Other Notable PUF designs

Here we compare the performance and characteristics of our proposed PUF concept with designs previously proposed by others, such as the hamming distances, attack resistance, and the efficiency of extraction of entropy from each response bit. Table 10 shows the comparison.

*Table 10, Comparison of key characteristics of notable PUF designs, * Results are taken from my estimation of a figure rather than a value stated by the author, ** Value is directly correlated to the resolution of the measurement of time during TDDS measurements.*

PUF Design	**J. Chen et al. 2015** [84]	**Govindaraj et al. 2016** [76]	**P. Chen et al. 2015** [137]	**Yanambaka et al. 2016**[138]	**This Work**
Source Type	**TDDS**	**Arbiter**	**RRAM Variation**	**RO**	**TDDS**
CRPs per Challenge Bit Combination	**1**	**1**	**1**	**1**	**Many****
Server CRP Storage	**all CRPs**	**all CRPs**	**all CRPs**	**all CRPs**	**probability model**
Mean of Intra Hamming	**~0.1***	**0.0013**	**0.02***	**0.0079**	**0.0227**
Mean of Inter Hamming	**~0.5***	**0.513**	**0.499***	**0.509**	**0.5008**

The first thing we look at is the number of CRPs that can be created for every possible combination of challenge bits. For conventional PUF designs, each challenge bit combination will only give a single fixed response. In our design we can utilise different sampling/challenge times to achieve variable responses from a single challenge bit combination. As shown in Figure 78 the response from the proposed PUF can have 2^n unique response possibilities (with n being the number of response bits) over the time window, for any given combination of challenge bits. The number of unique CRPs that can be extracted using this relationship is governed by the resolution that time can be measured at. The value in the table of 10,000 different possible CRPs for each challenge bit combination comes from our experimental results. In our experiments we used a 100kSa/s sampling rate over a 0.1s time window giving up to 10,000 possible time points to sample. When implemented in a practical PUF circuit these time points should be distributed logarithmically to better match the distribution of de-trapping events. It may also be necessary to remove some sampling points from the beginning and end of the time window as most bits will be the same in these regions, causing a reduced entropy in those regions. Even with these limitations it is clear that our PUF design offers a significant increase in the number of possible CRPs over previous designs, thereby increasing the design efficiency.

Next, we look at the difference in how traditional PUFs and our PUF store the CRP behaviour on the authentication server. In traditional PUFs, every CRP must be stored separately on the server which means PUFs with many possible challenge and response bits can have many millions of possible combinations which must all be stored taking up a significant amount of memory (especially if the server is connected to many PUFs). A PUF with only 16 challenge bits already has 65,536 unique possible CRPs and this number will increase exponentially as the number of challenge bits is increased. In our new proposed design, we only store the data for one probability curve (can be expressed as a look-up table or as a fitted equation) per response bit. For the same example as above, a PUF with 16 challenge bits and an 8-bit length key would require the 16 look-up tables or curve equations to be stored on the server to be able to calculate all possible CRPs. For very large and secure PUFs these savings in server storage could be significant. Figure 106 shows a comparison between the number of bits required to store CRP data using both the traditional method and our new proposed method. Both PUFs operate with a response length of 128 bits because this is the minimum value for key length that is considered secure for the Advanced Encryption Standard [139]. For these results, the memory required was calculated in two ways. For the traditional method, each challenge was represented as a binary number with the fewest required number of bits and the response was taken as a binary number of 128-bit length. The total required memory was the total number of bits used to have every possible CRP combination. For our newly proposed method, the total memory required was calculated by representing each entropy source's probability model look-up table as 10,000 32-bit floating point numbers. We then calculated how many unique CRP combinations a PUF with the given number of entropy sources could create when using a fixed response length of 128 bits. It can be seen that initially with a small number of CRPs (10,000) (which is the minimum amount for our proposed PUF with the 100kSa/s sampling rate during 0.1s of TDDS measurement) the traditional method is more memory efficient. However, when the number of CRPs is increased the traditional method requires significantly more memory to store the data as compared to our proposed method. For example, with our proposed method, 1x10^9 CRPs can be stored with a few 10s of Mb whereas the traditional method would require close to 1Tb of memory. This shows a very significant improvement for CRP storage and make it very feasible for a server to be connected to many PUFs with many CRPs without the memory requirements for storage becoming unfeasible.

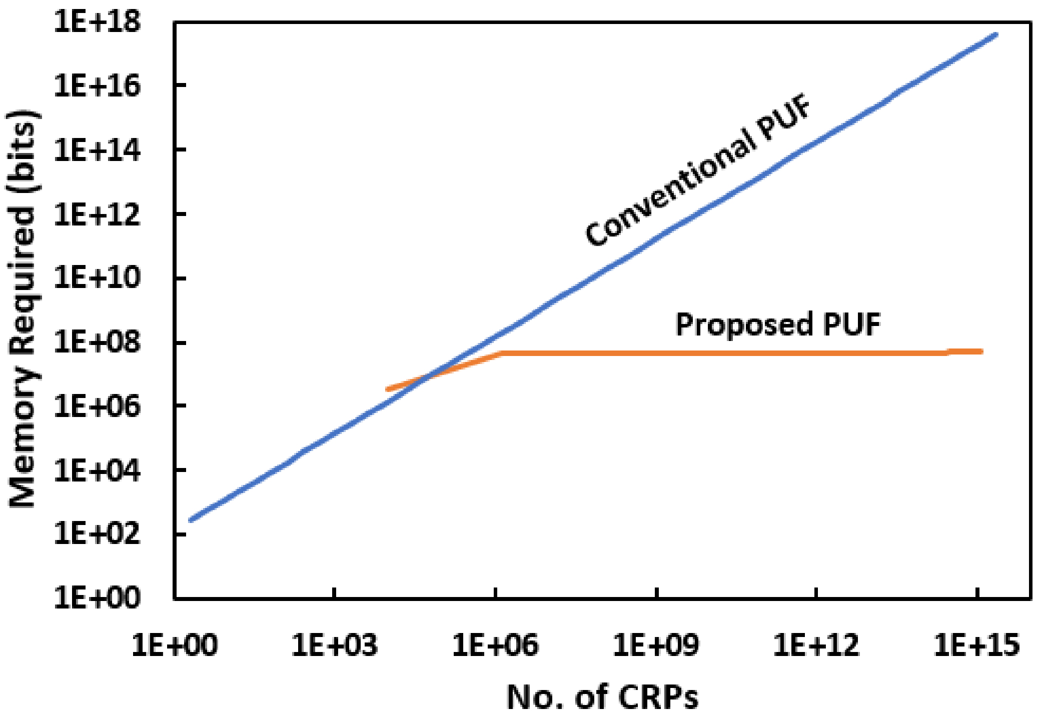


Figure 106, Comparison between memory bits required to store all CRP data from a traditional PUF vs our proposed PUF. Both PUFs use a 128bit response length. For the traditional PUF, the challenge number is represented with as few bits as possible. For out proposed PUF, each data point for the probability curve is a 32-bit floating-point number with a 100kSa/s sampling rate used during TDDS measurement.

Figure 107 shows how many challenge bits would be required to give the number of CRPs shown in Figure 106. Figure 107 shows that the number of additional challenge bits required to give very large numbers of CRPs is not very large, for example 1x10^9 CRPs can be achieved with only ~10 additional challenge bits to the 128 required to give the 128 bit response length. To find the number of possible challenge bit combinations in Figure 107 we first found all the possible numbers that could be represented with that many bits, we then found how many of those numbers had the correct number of '1' bits in them (for this example 128 '1s' to give 128 response values). By counting how many numbers have the correct numbers of '1' bits we can find the number of possible challenge bit combinations for any number of entropy sources. Due to the massive amount of numbers possible for PUFs with many additional bits (greater than 20), data after that value is based on estimation from fitting the previous data with a 4th order polynomial line of best fit, and extrapolating.

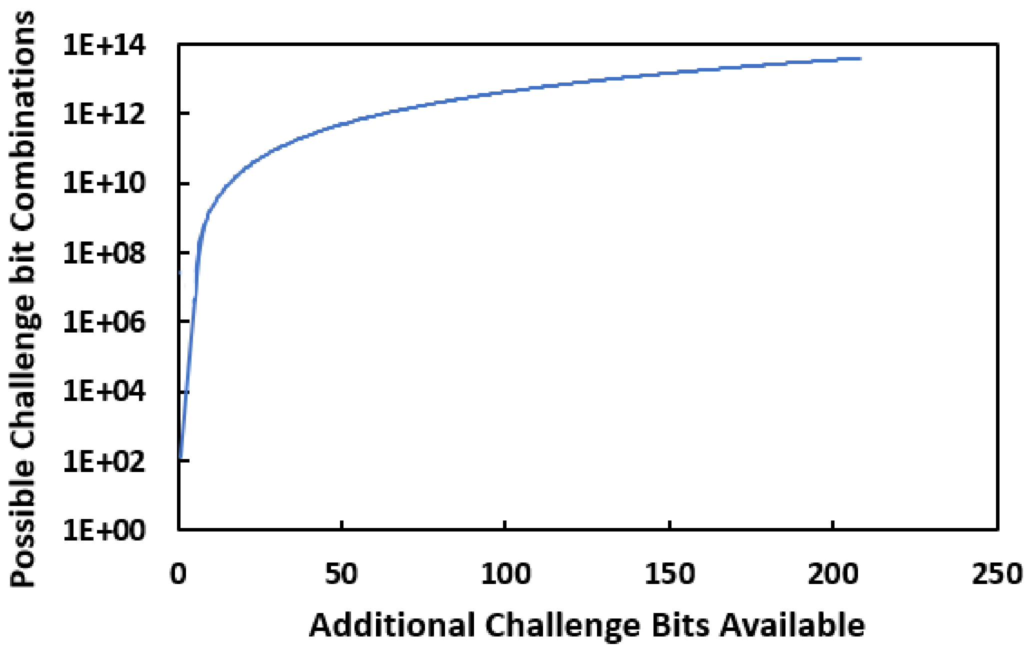


Figure 107, Number of possible challenge bit combinations for the proposed PUF when using a fixed response length of 128 bits. Additional challenge bits are added to the base 128 to allow for extra unique challenge combinations.

Looking at the mean intra hamming distance between responses to a single challenge for each PUF gives us good insight into the reliability of each design. Our design has a very similar mean intra hamming distance to the designs by J. Chen et al. [84] and P. Chen et al. [137], although it is noticeably higher than that for the other designs in Table 10, which suggests that they are more reliable. However, all of the designs would still need some error correction scheme for fully reliable communication so that the difference may not be as significant as it first appears. Also, we mentioned previously that using improved quality devices has a good chance of further improving the reliability of our proposed design. Overall, the slightly higher mean intra hamming distance is not ideal, but it is not a serious disadvantage against our proposed design.

Next, we discuss the mean inter hamming distance between responses from different PUFs to the same challenge. This is a good measure of the uniqueness of each PUF instance. Our PUF has a very good mean inter hamming distance and this is comparable with the other existing PUF designs shown in Table 10. There is no significant difference between the means of the inter hamming distances, showing that all the PUF designs have a good degree of uniqueness. For our PUF this means that we have reached the benchmark for this parameter and it is hard to show any major improvement over previous designs as this characteristic was already well displayed.

One important characteristic that we have shown is the mean inter hamming distance between different CRPs on one PUF instance. This value is a good representation of the correlation between different responses of a PUF instance. This is very important as modelling (most commonly machine learning) attacks are a big problem for PUFs that has previously not been fully addressed [74], [75], [134], [140], [141]. Our proposed PUF shows a very good mean inter hamming distance for this of 49.57% which is a good indicator that our PUF has very little correlation between responses that could be used as a vector of attack on the PUF. As a further extension to this, Table 10 shows measures that are taken to increase the attack resistance of the PUFs. None of the previous PUF designs explicitly take any measure to avoid these modelling attacks. Our PUF makes use of the uncertain bits resulting from the probability-based approach to obfuscate the actual used response bits from any observing attacker. This adds another level to the difficulty of building a machine learning-based model to attack the security of our PUF.

In conclusion for this comparison, our PUF has similar performance to previously proposed PUFs in some categories and exceeds them in other categories. The reliability (intra hamming distance) of our PUF is its weakest attribute when compared to some contemporary designs. In terms of entropy source use efficiency, server storage space efficiency, our PUF offers significant improvements and brings the concept of the PUF much closer to mainstream reality.

Chapter 6 - Conclusions and Future Work

The main goal of this project is to improve upon existing security primitives so that they could be realistically implemented into IoT devices, to secure them against the ever-growing threat of cyberattacks. In this chapter we conclude each major section of the undertaken project. This includes what has been achieved and how that contributes to the specific areas of research. We also discuss the potential future work that could be undertaken to continue the work of this project. We begin by looking at the work performed on the RTN-based TRNG and then move onto the proposed time-dependent PUF. We finish this chapter outlining a new potential security primitive which combines the TRNG and PUF concept into a single system.

6.1 Conclusions and Future Work for RTN-Based TRNG

6.1.1 Conclusions

Our aim for the TRNG is to improve it in areas that would make it more appropriate for use in IoT devices as existing designs all had issues that made them difficult to use for such applications. The first major issue is device selectivity. This was primarily a problem for existing RTN-based TRNGs where the RTN that was suitable for use as an entropy source was rare, resulting in the use of very large device arrays which occupy the majority of the TRNGs' design area. To improve upon this area, we made several novel advances. The first of these advances was the use of HC stressing to generate RTN traps reliably in devices that would be used as an entropy source. This could generate traps in over 80% of devices, meaning array sizes could be massively reduced. We also introduced a new entropy extraction scheme using a differentiator to allow us to effectively extract entropy from both multi-level and abnormal RTN traps which would have caused issues for previous designs. Being able to use more types of RTN as the entropy source further reduced device selectivity issues by ~30%. We also introduced a new sampling scheme where the RTN signal is used to sample an oscillator signal. This scheme simplified the complex post-processing circuits used in most previous RTN TRNGs. These key improvements reduced the area required for the TRNG circuit, making it a more appealing solution for generating random numbers in low power and space restricted IoT devices.

The second major limiting factor of previous RTN-based TRNG designs is the low bit rate. We have presented major improvements in this area. Firstly, we showed how using an AC RTN scheme can massively accelerate the observed RTN events, so that they can generate entropy at a faster rate. In the AC RTN scheme, we showed how using AC signals for both Vg and Vd can accelerate RTN events by potentially thousands of times. The new differentiator-based RTN sampling technique also helps improve TRNG output bit rate. By allowing extraction of entropy from multi-level RTN, we can have more RTN events extracted within a given time window, increasing the output bit rate.

We experimentally verified the performance of these improvements by creating a proof of concept circuit that extracted the entropy from the RTN in a single MOSFET device and applied the new processing and sampling schemes to give a random output bitstream. This bitstream was analysed using both the NIST test suite and the DIEHARD battery of tests as well as using an LSTM neural network to guess future bits. All the testing showed that the bitstream was random and suitable for use as a security device within IoT applications. All of these together show that we have achieved the goals for this section of the project and created a new RTN-based TRNG that is more suitable to IoT applications.

6.1.2 Future work

The obvious next step for this work is to create a fully integrated circuit (IC) prototype that could be used to fully assess the capabilities of the proposed design. With an IC prototype we could assess the actual design area, as well as important factors such as power consumption and the real maximum bit rate.

Another interesting avenue to explore would be to see if multiple devices could be connected in parallel and extract the RTN entropy from all of them simultaneously. This parallelisation has the potential to greatly increase the output bit rate. With enough devices in parallel this could even potentially give us the opportunity to scale up the TRNG designs with many entropy sources to give us output bit rates that could compete with the fastest TRNG designs.

Finally, one could explore increasing the chances of having clear RTN within devices during the manufacturing process without the need for stressing. Some possible things to explore are the doping levels used in each area of the device as well as the specific materials and design used for the device. If it is found that the amount of RTN within a device can be somewhat controlled through one of these mechanisms, then it would have to be calculated what is the most efficient, in terms of time and cost, way of inserting RTN into entropy source devices for TRNGs.

6.2 Conclusions and Future Work for Time-Dependent PUF

6.2.1 Conclusions

The second major aim of the project was to create a new PUF scheme that could improve upon as many of the major deficiencies of previous PUF designs as possible. The PUF concept has been around for many years but has not managed to break through into mainstream usage, especially for IoT devices. The main reasons for this are deficiencies in reliability, clonability, key distribution, power consumption, design complexity, or design footprint. We proposed a new time-dependent PUF scheme with the aim of improving upon these issues.

In terms of reliability the currently proposed PUF circuit does not offer huge improvements over existing PUF designs as the intra hamming distance is comparable with most existing designs. We do offer some encouragement for the long-term reliability by showing that the timing characteristics for the RTN traps, which are used as the source of the PUF behaviour, are stable over long periods of time.

For the clonability (including modelling) characteristic we show some key improvements over previous designs. As the probability-based approach is based on the random de-trapping times of defects within MOSFETs it also makes it impossible to make a physical clone of the PUF using the same technique. We have also shown that there is very little correlation between the responses of each PUF which further increases the difficulty of cloning the PUF. The time dependence also increases the difficulty of cloning the PUF because it increases the possible number of CRPs that each PUF module can have, which increases the number of CRPs that have to be modelled to clone the PUF.

For key distribution, we show some major possible improvements for the memory required on the server to store the CRPs of each PUF. By utilising the proposed probability model, we make it so each response bit of the PUF can have all of its CRP information stored as a single curve. This allows the server to create any possible CRP behaviour from these models rather than having to store every possible CRP. When as PUF instance has many hundreds of response bits (or more) the number of possible challenge combinations is huge. Our new scheme allows for a reduction in the required space by several orders of magnitude in this case.

For the final three deficiencies (power consumption, design complexity, and design footprint) we cannot comment yet on how our proposed PUF design will perform as we have not developed an IC design which can be used to compare our performance with that of existing PUF designs. This IC design would be guided by the findings of the proof of concept circuit created to verify the PUF functionality.

Overall, we have shown some key improvements over traditional PUF designs without reducing performance in any of the other key areas. However, the full potential of our PUF design has not been fully explored due to the time and facility limitations of this project.

6.2.2 Future Work

From our conclusions it is clear that there is a lot of further work that could be done to improve the proposed PUF design. The first and most basic task would be to repeat the experiments with high quality commercial devices to see if that can improve the reliability of the results and the PUF performance. When this is done it would also be better to use a much higher number of devices so inter hamming distances between different PUFs can be verified experimentally as well as with more simulations.

The next step would be to try and create a full IC design based upon the proposed proof of concept circuit that could be developed to find what the overall performance of the PUF could be. This would give much more insight into whether the PUF concept, in its current form, is suitable for real-world applications. This process would give essential information on the overall size of the circuit, as well as power consumption and cost. This would also give us more accurate results for the hamming distances for the PUF as modules with much higher numbers of response bits could be created that are not feasible to simulate due to the time required to simulate the TDDS measurements. Having a large number of CRPs also gives a good opportunity to test the proposed PUF's resistance to modelling attacks as there would be a large amount of data available to use as training data for a machine learning-based attack.

Another part of the proposed PUF principle that could be further explored is the possibility of using different challenge times for different devices within a single challenge to the PUF to further reduce any correlations between different response bits, making modelling attacks even more difficult. Another possibility is to use multiple challenge times for response bits in a single CRP, this would allow for longer authentication keys for a given number of response bits. Having longer keys could potentially increase the security of a PUF with a fixed number of response bits. However, this could introduce some correlations between bits within the response that would make predicting them easier so further analysis needs to be done.

Finally, it could be worth investigating the possibility of using the stored probability models in a public key-based system. In this system the probability models publicly available and the scheme would follow a similar authentication protocol as the one proposed by Rührmair et al.[92]. In this proposed system the CRPs are made available through the use of a public model of the PUF and then the PUF model itself can be authenticated if it can respond to a challenge faster than anyone could simulate the result using the public model. This solution solves a key problem for PUFs, which is the ability to easily distribute the CRPs to authenticating parties. The main issue that currently stops our PUF using a similar scheme is the slow speed of the TDDS measurements so it is unlikely that the real PUF module could generate the correct response more quickly than someone else with the probability model. To solve this our probability model-based approach could be used but with a different entropy source that is capable of creating responses at a significantly faster rate. A PUF utilising all of these concepts together would have a good chance of making a design that could solve the major challenges that are currently stopping PUFs from becoming a mainstream security primitive.

6.3 - Other Possible Security Primitives for Low Power IoT Applications

In this section we discuss a security primitive that could be developed using a combination of two designs which have been discussed in this project but did not have time to fully explore. This third possible design is a combined PUF and TRNG that both utilise the same entropy source. The concept of a combined entropy source was first proposed by Satpathy et al. [36], where they combined a TRNG and a Chip-ID, which is a basically a PUF with only a single CRP. This circuit has a lot of potential as there is often a requirement for a TRNG to be present as well as a PUF to secure a system and implementing them separately is much less efficient. By creating a unified entropy source for both significant area, power, and cost saving are possible, all of which are major considerations for IoT devices.

The entropy source for this new combined TRNG and PUF is again single MOSFETs with RTN defects present. The entropy is harvested from these devices using the same TDDS measurement as used in the PUF part of this project. The output current from each TDDS measurement is sent to an edge-to-pulse unit following the same design as the one used in the TRNG proof of concept circuit from section 4.4.2 and the PUF proof of concept circuit shown in section 5.2.2. It is these pulses that are used differently to create TRNG and PUF operation simultaneously. For the TRNG operation, the pulses are simply used to sample a clock signal, as the pulses in section 4.4.2 are used. This process will create a random bitstream where the timing of each bit is determined by the de-trapping times of the traps measured in the TDDS measurement. By using the timing of all of the de-trapping events across multiple entropy sources in parallel, there is potential to have a very high random number throughput while the TDDS measurements are being performed.

For the PUF section of the circuit, the pulses are sent to counters that count how many pulses have occurred within a certain timeframe. The counted pulses are the used following the same probability-based principle as used in section 5.2 of this work. Figure 108 shows a block diagram of this concept.

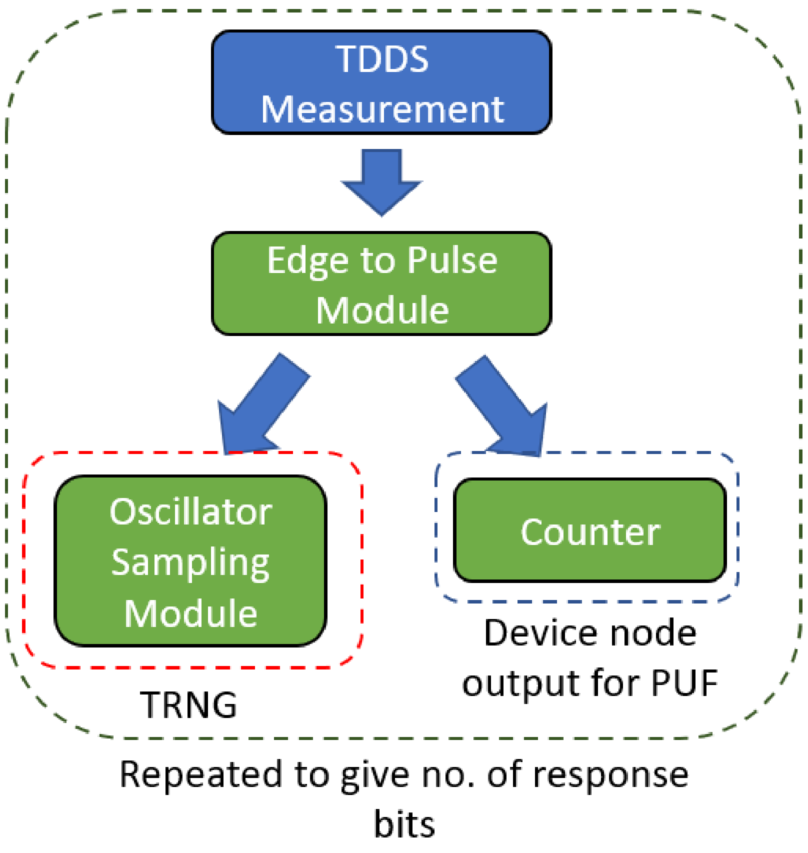

TDDS Measurement
Edge to Pulse Module
Oscillator Sampling Module
TRNG
Counter
Device node output for PUF
Repeated to give no. of response bits

Figure 108, Block diagram of proposes combined TRNG and PUF unit.

In this concept both units can operate by themselves if only one system is required at the time or both can be operated simultaneously so that the TRNG can generate a random bitstream which can be use with an encryption protocol to secure the PUF communication following the idea as proposed by Satpathy et al. [36].

This concept was not explored any further in this project, so how much this approach could improve design efficiency and/or security has not been quantified. However, this proposal does show that there is a clear next step for future work following on from this work and there is a lot of further potential in each of the proposed concepts.

References

[1] Cabinet Office, "The UK cyber security strategy," 2016. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/516331/UK_Cyber_Security_Strategy_Annual_Report_2016.pdf.

[2] HMG, "National Cyber Security Strategy," 2016. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/567242/national_cyber_security_strategy_2016.pdf.

[3] J. Manyika *et al.*, "The Internet of Things: Mapping the value beyond the hype," 2015. doi: 10.1007/978-3-319-05029-4_7.

[4] J. Manyika *et al.*, "Unlocking the potential of the Internet of Things | McKinsey & Company," 2016. http://www.mckinsey.com/business-functions/business-technology/our-insights/the-internet-of-things-the-value-of-digitizing-the-physical-world%5Cnhttp://files/1357/Manyika%5Cnet%5Cnal.%5Cn-%5CnUnlocking%5Cnthe%5Cnpotential%5Cnof%5CnInternet%5CnThings%5Cn.p (accessed Jun. 01, 2018).

[5] A. Mohsen Nia and N. K. Jha, "A Comprehensive Study of Security of Internet-of-Things," *IEEE Trans. Emerg. Top. Comput.*, vol. 5, no. 4, pp. 586–602, 2016, doi: 10.1109/TETC.2016.2606384.

[6] D. Eastlake, J. Schiller, and S. Crocker, "Randomness Requirements for Security," 2005. doi: 10.1.1.175.7831.

[7] P. K. B. Jun, "THE INTEL® RANDOM NUMBER GENERATOR," 1999.

[8] B. Schneier, *Applied Cryptography Protocols, Algorithms, and Source Code in C*. New York: John Wiley & Sons, 1996.

[9] Z. Gutterman, B. Pinkas, and T. Reinman, "Analysis of the linux random number generator," *Proc. - IEEE Symp. Secur. Priv.*, vol. 2006, pp. 371–385, 2006, doi: 10.1109/SP.2006.5.

[10] MIT, "Side channel Attacks on RSA," 2014.

[11] E. Carmon, J. P. Seifert, and A. Wool, "Photonic side channel attacks against RSA," *Proc. 2017 IEEE Int. Symp. Hardw. Oriented Secur. Trust. HOST 2017*, pp. 74–78, 2017, doi: 10.1109/HST.2017.7951801.

[12] J. Kelsey, B. Schneier, D. Wagner, and C. Hall, "Cryptanalytic attacks on pseudorandom number generators," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 1372, pp. 168–188, 1998, doi: 10.1007/3-540-69710-1_12.

[13] I. Verbauwhede and K. Chuang, "Security and reliability – friend or foe," *Int. Electron Devices Meet. IEDM*, pp. 290–293, 2019.

[14] D. Fang, Y. Qian, and R. Q. Hu, "Security for 5G Mobile Wireless Networks," *IEEE Access*, vol. 6, pp. 4850–4874, 2017, doi: 10.1109/ACCESS.2017.2779146.

[15] E. Barker and A. Roginsky, "NIST Special Publication 800-133 Recommendation for Cryptographic Key Generation," 2012. doi: 10.6028/NIST.SP.800-133.

[16] A. T. Markettos and S. W. Moore, "The frequency injection attack on ring-oscillator-based true

random number generators," *Cryptogr. Hardw. Embed. Syst.*, vol. 5747 LNCS, pp. 317–331, 2009, doi: 10.1007/978-3-642-04138-9_23.

[17] V. K. Rai, S. Tripathy, and J. Mathew, "Memristor based Random Number Generator: Architectures and Evaluation," *Procedia Comput. Sci.*, vol. 125, pp. 576–583, 2017, doi: 10.1016/j.procs.2017.12.074.

[18] W. C. Ku and S. M. Chen, "Weaknesses and improvements of an efficient password based remote user authentication scheme using smart cards," *IEEE Trans. Consum. Electron.*, vol. 50, no. 1, pp. 204–207, 2004, doi: 10.1109/TCE.2004.1277863.

[19] W. C. Ku, "Weaknesses and drawbacks of a password authentication scheme using neural networks for multiserver architecture," *IEEE Trans. Neural Networks*, vol. 16, no. 4, pp. 1002–1005, 2005, doi: 10.1109/TNN.2005.849781.

[20] L. Gong, M. A. Lomas, R. M. Needham, and J. H. Saltzer, "Protecting Poorly Chosen Secrets from Guessing Attacks," *IEEE J. Sel. AREAS Commun.*, vol. 11, no. 5, pp. 648–656, 1993.

[21] V. Taneski, M. Hericko, and B. Brumen, "Password security - No change in 35 years?," *2014 37th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2014 - Proc.*, no. May, pp. 1360–1365, 2014, doi: 10.1109/MIPRO.2014.6859779.

[22] J. Heasuk, L. Yunho, K. Mijin, K. Seungjoo, and W. Dongho, "Off-line password-guessing attack to Yang's and Huang's authentication schemes for session initiation protocol," *NCM 2009 - 5th Int. Jt. Conf. INC, IMS, IDC*, pp. 618–621, 2009, doi: 10.1109/NCM.2009.251.

[23] S. Schechter, A. J. B. Brush, and S. Egelman, "It's no secret Measuring the security and reliability of authentication via 'secret' questions," *Proc. - IEEE Symp. Secur. Priv.*, pp. 375–390, 2009, doi: 10.1109/SP.2009.11.

[24] Ipsos MORI Social Reseach Institute, "UK Cyber Survey Key findings – General public," no. April, pp. 1–19, 2019.

[25] A. Braeken, "PUF based authentication protocol for IoT," *Symmetry (Basel).*, vol. 10, no. 8, pp. 1–15, 2018, doi: 10.3390/sym10080352.

[26] B. Halak, M. Zwolinski, and M. S. Mispan, "Overview of PUF-Based hardware security solutions for the internet of things," *Midwest Symp. Circuits Syst.*, no. October, pp. 16–19, 2017, doi: 10.1109/MWSCAS.2016.7870046.

[27] A. Chen *et al.*, "Using Emerging Technologies for Hardware Security Beyond PUFs," *Des. Autom. Test Eur. Conf. Exhib.*, vol. 2, no. 1, pp. 1544–1549, 2016.

[28] U. Ruhrmair and J. Solter, "PUF modeling attacks: An introduction and overview," *Des. Autom. Test Eur. Conf. Exhib. (DATE), 2014*, pp. 1–6, 2014, doi: 10.7873/DATE.2014.361.

[29] M. Potkonjak and V. Goudar, "Public physical unclonable functions," *Proc. IEEE*, vol. 102, no. 8, pp. 1142–1156, 2014, doi: 10.1109/JPROC.2014.2331553.

[30] I. Verbauwhede, "Physical(ly) Unclonable Functions An introduction to Intrinsic PUFs," 2011.

[31] C. K. H. Suresh and S. S. Ali, "a Comparative Security Analysis of Current and Emerging Technologies the Challenges of Cmos Technology Below the 20-Nm Level Have Led," *IEEE Comput. Soc.*, pp. 50–61, 2016.

[32] D. Bucerzan and M. Cr, "Stream Ciphers Analysis Methods," *Int. J. Comput. Commun. Control*,

vol. V, no. 4, pp. 483–489, 2010.

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

[34] J. Zaman and R. Ghosh, “A Review Study of NIST Statistical Test Suite: Development of an indigenous Computer Package,” *arXiv Prepr. arXiv1208.5740*, 2012, [Online]. Available: http://arxiv.org/abs/1208.5740.

[35] R. Sharma, R. Ullagaddimath, A. B. Roy, A. Halder, and V. Hegde, “Optical theremin based true Random Number Generation (TRNG) system,” *2015 Int. Conf. Adv. Comput. Commun. Informatics, ICACCI 2015*, pp. 571–575, 2015, doi: 10.1109/ICACCI.2015.7275670.

[36] S. Satpathy *et al.*, “An All-Digital Unified Static / Dynamic Entropy Generator Featuring Self-Calibrating Hierarchical Von Neumann Extraction for Secure Privacy-Preserving Mutual Authentication in IoT Mote Platforms,” *2018 Symp. VLSI Circuits Dig. Tech. Pap.*, pp. 158–159, 2018.

[37] A. Mohanty, K. B. Sutaria, H. Awano, T. Sato, and Y. Cao, “RTN in Scaled Transistors for On-Chip Random Seed Generation,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 25, no. 8, pp. 2248–2257, 2017, doi: 10.1109/TVLSI.2017.2687762.

[38] Z. Chai *et al.*, “GeSe-based Ovonic Threshold Switching Volatile True Random Number Generator,” *IEEE Electron Device Lett.*, vol. PP, no. c, p. 1, 2019, doi: 10.1109/LED.2019.2960947.

[39] J. Kim *et al.*, “Nano-Intrinsic True Random Number Generation: A Device to Data Study,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 66, no. 7, pp. 2615–2626, 2019, doi: 10.1109/TCSI.2019.2895045.

[40] J. Kim *et al.*, “Nano-Intrinsic True Random Number Generation,” pp. 1–9, 2017, [Online]. Available: http://arxiv.org/abs/1701.06020.

[41] J. Holleman, S. Bridges, B. P. Otis, and C. Diorio, “A 3 µW CMOS true random number generator with adaptive floating-gate offset cancellation,” *IEEE J. Solid-State Circuits*, vol. 43, no. 5, pp. 1324–1336, 2008, doi: 10.1109/JSSC.2008.920327.

[42] M. K. Sanu *et al.*, “2.4 Gbps, 7 mW all-digital PVT-variation tolerant true random number generator for 45 nm CMOS high-performance microprocessors,” *IEEE J. Solid-State Circuits*, vol. 47, no. 11, pp. 2807–2821, 2012, doi: 10.1109/JSSC.2012.2217631.

[43] N. Liu, N. Pinckney, S. Hanson, D. Sylvester, and D. Blaauw, “A True Random Number Generator using Time-Dependent Dielectric Breakdown,” *Symp. VLSI Circuits Dig. Tech. Pap.*, pp. 216–217, 2011.

[44] K. Yang, D. Fick, M. B. Henry, Y. Lee, D. Blaauw, and D. Sylvester, “A 23Mb/s 23pJ/b fully synthesized true-random-number generator in 28nm and 65nm CMOS,” *Dig. Tech. Pap. - IEEE Int. Solid-State Circuits Conf.*, vol. 57, pp. 280–281, 2014, doi: 10.1109/ISSCC.2014.6757434.

[45] H. Jiang *et al.*, “A novel true random number generator based on a stochastic diffusive memristor,” *Nat. Commun.*, vol. 8, no. 1, p. 882, 2017, doi: 10.1038/s41467-017-00869-x.

[46] P. Pouyan, E. Amat, and A. Rubio, “Insights to memristive memory cell from a reliability perspective,” *2015 Int. Conf. Memristive Syst. MEMRISYS 2015*, pp. 1–2, 2016, doi: 10.1109/MEMRISYS.2015.7378382.

[47] Y. Wang *et al.*, “Investigation of electrical performance and reliability of memristors by tuning compliance current during electroforming process,” *2019 IEEE Int. Conf. Electron Devices Solid-State Circuits, EDSSC 2019*, pp. 1–2, 2019, doi: 10.1109/EDSSC.2019.8753917.

[48] R. Brederlow, R. Prakash, C. Paulus, and R. Thewes, “A low-power true random number generator using random telegraph noise of single oxide-traps,” *IEEE Int. Solid-State Circuit Conf.*, pp. 1666–1675, 2006, doi: 10.1109/ISSCC.2006.1696222.

[49] D. S. B. Sunar, W. Martin, “A Provably Secure True Random Number Generator with Built-in Tolerance to Active Attacks,” *IEEE Trans. Comput.*, vol. 56, no. 1, pp. 109–119, 2007, doi: 10.1109/TC.2007.250627.

[50] C. Tokunaga, D. Blaauw, and T. Mudge, “True Random Number Generator With a Metastability-Based Quality Control,” *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 78–85, 2008, doi: 10.1109/JSSC.2007.910965.

[51] T. Rahman, K. Xiao, D. Forte, X. Zhang, J. Shi, and M. Tehranipoor, “TI-TRNG: Technology Independent True Random Number Generator Md.,” *Proc. 51st Annu. Des. Autom. Conf. Des. Autom. Conf. - DAC ’14*, pp. 1–6, 2014, doi: 10.1145/2593069.2593236.

[52] S. Mathew *et al.*, “µrNG: A 300-950mV 323Gbps/W all-digital full-entropy true random number generator in 14nm FinFET CMOS,” *Eur. Solid-State Circuits Conf.*, vol. 2015-Octob, pp. 116–119, 2015, doi: 10.1109/ESSCIRC.2015.7313842.

[53] T. Figliolia, P. Julian, G. Tognetti, and A. G. Andreou, “A true Random Number Generator using RTN noise and a sigma delta converter,” *IEEE Int. Symp. Circuits Syst.*, vol. 2, no. 1, pp. 17–20, 2016, doi: 10.1109/ISCAS.2016.7527159.

[54] P. Z. Wieczorek, “Lightweight TRNG based on multiphase timing of bistables,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 63, no. 7, pp. 1043–1054, 2016, doi: 10.1109/TCSI.2016.2555248.

[55] G. Gimenez, A. Cherkaoui, R. Frisch, and L. Fesquet, “Self-timed Ring based True Random Number Generator: Threat model and countermeasures,” *2017 2nd Int. Verif. Secur. Work. IVSW 2017*, pp. 31–38, 2017, doi: 10.1109/IVSW.2017.8031541.

[56] E. Kim, M. Lee, and J. J. Kim, “8Mb/s 28Mb/mJ robust true-random-number generator in 65nm CMOS based on differential ring oscillator with feedback resistors,” *Dig. Tech. Pap. - IEEE Int. Solid-State Circuits Conf.*, vol. 60, pp. 144–145, 2017, doi: 10.1109/ISSCC.2017.7870302.

[57] M. Matsumoto, R. Ohba, S. I. Yasuda, K. Uchida, T. Tanamoto, and S. Fujtta, “Non-stoichiometric SixN metal-oxide-semiconductor field-effect transistor for compact random number generator with 0.3 Mbit/s generation rate,” *Japanese J. Appl. Physics, Part 1 Regul. Pap. Short Notes Rev. Pap.*, vol. 47, no. 8 PART 1, pp. 6191–6195, 2008, doi: 10.1143/JJAP.47.6191.

[58] F. Pareschi, G. Setti, and R. Rovatti, “A fast chaos-based true random number generator for cryptographic applications,” *ESSCIRC 2006 - Proc. 32nd Eur. Solid-State Circuits Conf.*, no. 1, pp. 130–133, 2006, doi: 10.1109/ESSCIR.2006.307548.

[59] M. Soucarros, J. Clédière, C. Dumas, and P. Elbaz-Vincent, “Fault Analysis and Evaluation of a True Random Number Generator Embedded in a Processor,” *J. Electron. Test.*, vol. 29, no. 3, pp. 367–381, 2013, doi: 10.1007/s10836-013-5356-1.

[60] X. Chen *et al.*, “Modeling Random Telegraph Noise as a Randomness Source and Its Application

in True Random Number Generation," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 35, no. 9, pp. 1435–1448, 2015, doi: 10.1109/TCAD.2015.2511074.

[61] Y. Liu, R. C. C. Cheung, and H. Wong, "A Bias-Bounded Digital True Random Number Generator Architecture," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 64, no. 1, pp. 133–144, 2017, doi: 10.1109/TCSI.2016.2606353.

[62] V. R. Pamula, X. Sun, S. Kim, F. Rahman, B. Zhang, and V. S. Sathe, "An All-Digital True-Random-Number Generator with Integrated De-correlation and Bias Correction at 3 . 2-to-86 Mb / s , 2 . 58 pJ / bit in 65-nm CMOS," *2018 Symp. VLSI Circuits Dig. Tech. Pap.*, no. 2017, pp. 2017–2018, 2018.

[63] K. Yang *et al.*, "A 28nm Integrated True Random Number Generator Harvesting Entropy from MRAM," *2018 Symp. VLSI Circuits Dig. Tech. Pap.*, vol. 2, pp. 171–172, 2018.

[64] R. Maes, "Physically Unclonable Functions: Constructions, Properties and Applications (Fysisch onkloonbare functies: constructies, eigenschappen en toepassingen)," Katholieke Universiteit Leuven, 2012.

[65] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Silicon physical random functions," *Proc. ACM Conf. Comput. Commun. Secur.*, pp. 148–160, 2002, doi: 10.1145/586131.586132.

[66] A. Maiti, V. Gunreddy, and P. Schaumont, "A systematic method to evaluate and compare the performance of physical unclonable functions," *Embed. Syst. Des. with FPGAs*, vol. 9781461413, pp. 245–267, 2013, doi: 10.1007/978-1-4614-1362-2_11.

[67] R. Pappu, "Physical One-Way Functions. Massachusetts Ave," Massachusetts Institute of Technology, 2001.

[68] E. Toreini, S. F. Shahandashti, and F. Hao, "Texture to the Rescue: Practical Paper Fingerprinting based on Texture Patterns," *arXiv*, 2017.

[69] P. Tuyls, G. J. Schrijen, B. Škorić, J. Van Geloven, N. Verhaegh, and R. Wolters, "Read-proof hardware from protective coatings," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4249 LNCS, pp. 369–383, 2006, doi: 10.1007/11894063_29.

[70] J. Guajardo, S. S. Kumar, G. Schrijen, and P. Tuyls, "FPGA Intrinsic PUFs and Their Use for IP Protection," *Work. Cryptogr. Hardw. Embed. Syst. – CHES*, pp. 63–80, 2007.

[71] U. Rührmair and D. E. Holcomb, "PUFs at a glance," *Proc. -Design, Autom. Test Eur. DATE*, 2014, doi: 10.7873/DATE2014.360.

[72] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," *IEEE Symp. VLSI Circuits, Dig. Tech. Pap.*, no. CIRCUITS SYMP., pp. 176–179, 2004, doi: 10.1109/vlsic.2004.1346548.

[73] B. Gassend, "Physical Random Functions Blaise Gassend Masters Thesis 2003," Massachusetts Institute of Technology, 2003.

[74] D. Lim, "Extracting Secret Keys from Integrated Circuits," Massachusetts Institute of Technology, 2004.

[75] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, D. Srinivas, and J. Schmidhuber, "Modeling attacks

on physical unclonable functions," *ACM Conf. Comput. Commun. Secur. (CCS '10)*, 2010.

[76] R. Govindaraj, "A Strong Arbiter PUF using Resistive RAM," *Int. Conf. Embed. Comput. Syst. Archit. Model. Simul.*, pp. 275–280, 2016.

[77] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," *Proc. - Des. Autom. Conf.*, pp. 9–14, 2007, doi: 10.1109/DAC.2007.375043.

[78] C. E. D. Yin and G. Qu, "LISA: Maximizing RO PUF's secret extraction," *Proc. 2010 IEEE Int. Symp. Hardware-Oriented Secur. Trust. HOST 2010*, no. Ic, pp. 100–105, 2010, doi: 10.1109/HST.2010.5513105.

[79] A. Maiti and P. Schaumont, "Improved ring oscillator PUF: An FPGA-friendly secure primitive," *J. Cryptol.*, vol. 24, no. 2, pp. 375–397, 2011, doi: 10.1007/s00145-010-9088-4.

[80] J. H. Anderson, "A PUF design for secure FPGA-based embedded systems," *Proc. Asia South Pacific Des. Autom. Conf. ASP-DAC*, pp. 1–6, 2010, doi: 10.1109/ASPDAC.2010.5419927.

[81] K. SHIMIZU, D. SUZUKI, and T. KASUYA, "Glitch PUF: Extracting Information from Usually Unwanted Glitches," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E95.A, no. 1, pp. 223–233, 2012, doi: 10.1587/transfun.E95.A.223.

[82] D. Suzuki and K. Shimizu, "The glitch PUF: A new delay-PUF architecture exploiting glitch shapes," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6225 LNCS, pp. 366–382, 2010, doi: 10.1007/978-3-642-15031-9_25.

[83] D. E. Holcomb, W. P. Burleson, and K. Fu, "Power-Up SRAM state as an identifying fingerprint and source of true random numbers," *IEEE Trans. Comput.*, vol. 58, no. 9, pp. 1198–1210, 2009, doi: 10.1109/TC.2008.212.

[84] J. Chen, T. Tanamoto, H. Noguchi, and Y. Mitani, "a Novel Concept Physical Unclonable Function (PUF) with Robust Reliabilities T40 T41," *2015 Symp. VLSI Technol. (VLSI Technol.*, pp. T40–T41, 2015, doi: 10.1109/VLSIT.2015.7223695.

[85] Y. Su, J. Holleman, and B. Otis, "A 1.6pJ/bit 96% Stable Chip-ID Generating Circuit using Process Variations," *Int. Solid-State Circuits Conf. ISSCC*, pp. 406–408, 2007.

[86] S. S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "Extended abstract: The butterfly PUF protecting IP on every FPGA," *2008 IEEE Int. Work. Hardware-Oriented Secur. Trust*, no. 71369, pp. 67–70, 2008, doi: 0.1109/HST.2008.4559053.

[87] P. Simons, E. Van Der Sluis, and V. Van Der Leest, "Buskeeper PUFs, a promising alternative to D Flip-Flop PUFs," *Proc. 2012 IEEE Int. Symp. Hardware-Oriented Secur. Trust. HOST 2012*, pp. 7–12, 2012, doi: 10.1109/HST.2012.6224311.

[88] S. K. Satpathy *et al.*, "An All-Digital Unified Physically Unclonable Function and True Random Number Generator Featuring Self-Calibrating Hierarchical von Neumann Extraction in 14-nm Tri-gate CMOS," *IEEE J. Solid-State Circuits*, vol. 54, no. 4, pp. 1074–1085, 2019, doi: 10.1109/JSSC.2018.2886350.

[89] B. Gassend, D. Clarke, M. Van Dijk, and S. Devadas, "Controlled physical random functions," *Proc. - Annu. Comput. Secur. Appl. Conf. ACSAC*, vol. 2002-Janua, pp. 149–160, 2002, doi: 10.1109/CSAC.2002.1176287.

[90] K. Kursawe, A. R. Sadeghi, D. Schellekens, B. Škorić, and P. Tuyls, "Reconfigurable physical

unclonable functions - Enabling technology for tamper-resistant storage," *2009 IEEE Int. Work. Hardware-Oriented Secur. Trust. HOST 2009*, no. January, pp. 22–29, 2009, doi: 10.1109/HST.2009.5225058.

[91] S. Katzenbeisser, Ü. Kocabaş, V. van der Leest, A. R. Sadeghi, G. J. Schrijen, and C. Wachsmann, "Recyclable PUFs: Logically reconfigurable PUFs," *J. Cryptogr. Eng.*, vol. 1, no. 3, pp. 177–186, 2011, doi: 10.1007/s13389-011-0016-9.

[92] U. Rührmair, "SIMPL Systems: On a Public Key Variant of Physical Unclonable Functions.," *IACR Cryptol. ePrint Arch.*, pp. 1–16, 2009, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.215.7745&rep=rep1&type=pdf.

[93] N. Beckmann and M. Potkonjak, "Hardware-based public-key cryptography with public physically unclonable functions," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 5806 LNCS, pp. 206–220, 2009, doi: 10.1007/978-3-642-04431-1_15.

[94] S. Meguerdichian and M. Potkonjak, "Matched public PUF: Ultra low energy security platform," *Proc. Int. Symp. Low Power Electron. Des.*, pp. 45–50, 2011, doi: 10.1109/ISLPED.2011.5993602.

[95] S. Meguerdichian and M. Potkonjak, "Using Standardized Quantization for Multi-Party PPUF Matching: Foundations and Applications," *2012 Ieee/Acm Int. Conf. Comput. Des.*, pp. 577–584, 2012.

[96] W. Wondrak, "Physical limits and lifetime limitations of semiconductor devices at high temperatures," *Microelectron. Reliab.*, vol. 39, no. 6–7, pp. 1113–1120, 1999, doi: 10.1016/S0026-2714(99)00158-4.

[97] R. Gao, "Bias Temperature Instability Modelling and Lifetime Prediction on Nano-scale MOSFETs," Liverpool John Moores University, 2018.

[98] F. M. Puglisi and P. Pavan, "RTN analysis with FHMM as a tool for multi-trap characterization in HfOX RRAM," *2013 IEEE Int. Conf. Electron Devices Solid-State Circuits, EDSSC 2013*, pp. 3–4, 2013, doi: 10.1109/EDSSC.2013.6628059.

[99] J. Zou *et al.*, "New insights into AC RTN in scaled high-κ/ metal-gate MOSFETs under digital circuit operations," *Dig. Tech. Pap. - Symp. VLSI Technol.*, pp. 139–140, 2012, doi: 10.1109/VLSIT.2012.6242500.

[100] M. Toledano-Luque *et al.*, "Response of a single trap to AC negative bias temperature stress," *IEEE Int. Reliab. Phys. Symp. Proc.*, pp. 4A.2.1-4A.2.8, 2011, doi: 10.1109/IRPS.2011.5784501.

[101] C.-Y. Chen *et al.*, "Correlation of Id- and Ig-random telegraph noise to positive bias temperature instability in scaled high-k metal gate n-type MOSFETs," *Reliab. Phys. Symp. (IRPS), 2011 IEEE Int.*, pp. 3A.2.1-3A.2.6, 2011, doi: 10.1109/IRPS.2011.5784475.

[102] F. M. Puglisi, F. Costantini, B. Kaczer, L. Larcher, and P. Pavan, "Probing defects generation during stress in high-κ/metal gate FinFETs by random telegraph noise characterization," *Eur. Solid-State Device Res. Conf.*, vol. 2016-Octob, pp. 252–255, 2016, doi: 10.1109/ESSDERC.2016.7599633.

[103] J. Brown *et al.*, "A low-power and high-speed True Random Number Generator using generated RTN," *Dig. Tech. Pap. - Symp. VLSI Technol.*, vol. 2018-June, no. 2, pp. 95–96, 2018, doi: 10.1109/VLSIT.2018.8510671.

[104] T. Grasser, H. Reisinger, P. Wagner, and B. Kaczer, “The Time Dependent Defect Spectroscopy (TDDS) Technique for the Bias Temperature Instability,” *Reliab. Phys. Symp. (IRPS), 2010 IEEE Int.*, vol. 1835, pp. 9–10, 2010.

[105] F. M. Puglisi, N. Zagni, L. Larcher, and P. Pavan, “Random Telegraph Noise in Resistive Random Access Memories: Compact Modeling and Advanced Circuit Design,” *IEEE Trans. Electron Devices*, pp. 1–9, 2018, doi: 10.1109/TED.2018.2833208.

[106] S. Guo, R. Wang, D. Mao, Y. Wang, and R. Huang, “Anomalous random telegraph noise in nanoscale transistors as direct evidence of two metastable states of oxide traps,” *Sci. Rep.*, vol. 7, no. 1, pp. 1–6, 2017, doi: 10.1038/s41598-017-06467-7.

[107] R. Gao *et al.*, “Predictive As-grown-Generation (A-G) model for BTI-induced device/circuit level variations in nanoscale technology nodes,” *Tech. Dig. - Int. Electron Devices Meet. IEDM*, no. 5, pp. 31.4.1-31.4.4, 2017, doi: 10.1109/IEDM.2016.7838520.

[108] J. F. Zhang, M. Duan, Z. Ji, and W. Zhang, “Hot carrier aging of nano-meter devices,” *2016 13th IEEE Int. Conf. Solid-State Integr. Circuit Technol. ICSICT 2016 - Proc.*, pp. 432–435, 2017, doi: 10.1109/ICSICT.2016.7998943.

[109] F. M. Puglisi, A. Padovani, L. Larcher, and P. Pavan, “Random telegraph noise: Measurement, data analysis, and interpretation,” *IEEE 24th Int. Symp. Phys. Fail. Anal. Integr. Circuits*, pp. 1–9, 2017, doi: 10.1109/IPFA.2017.8060057.

[110] F. M. Puglisi and P. Pavan, “Guidelines for a Reliable Analysis of Random Telegraph Noise in Electronic Devices,” *IEEE Trans. Instrum. Meas.*, vol. 65, no. 6, pp. 1435–1442, 2016, doi: 10.1109/TIM.2016.2518880.

[111] S. Dongaonkar, M. D. Giles, A. Kornfeld, B. Grossnickle, and J. Yoon, “Random telegraph noise (RTN) in 14nm logic technology: High volume data extraction and analysis,” *Dig. Tech. Pap. - Symp. VLSI Technol.*, vol. 2016-Septe, no. c, pp. 1–2, 2016, doi: 10.1109/VLSIT.2016.7573424.

[112] J. F. Zhang, M. Duan, Z. Ji, and W. Zhang, “Defects for random telegraph noise and negative bias temperature instability,” *China Semicond. Technol. Int. Conf. 2016, CSTIC 2016*, vol. 1, pp. 1–3, 2016, doi: 10.1109/CSTIC.2016.7464070.

[113] Z. Zhang, S. Guo, X. Jiang, R. Wang, R. Huang, and J. Zou, “Investigation on the amplitude distribution of random telegraph noise (RTN) in nanoscale MOS devices,” *2016 IEEE Int. Nanoelectron. Conf.*, pp. 1–2, 2016, doi: 10.1109/INEC.2016.7589332.

[114] A. Konczakowska and B. Wilamowski, “Noise in Semiconductor Devices,” in *Fundamentals of Industrial Electronics*, 2010, pp. 11–1, 11–12.

[115] K. P. Cheung, J. P. Campbell, S. Potbhare, and A. Oates, “The amplitude of random telegraph noise: Scaling implications,” *IEEE Int. Reliab. Phys. Symp. Proc.*, pp. 2–4, 2012, doi: 10.1109/IRPS.2012.6241908.

[116] K. P. Cheung and J. P. Campbell, “On the magnitude of random telegraph noise in ultra-scaled MOSFETs,” *2011 IEEE Int. Conf. Integr. Circuit Des. Technol. ICICDT 2011*, pp. 9–12, 2011, doi: 10.1109/ICICDT.2011.5783191.

[117] M. L. Fan, V. P. H. Hu, Y. N. Chen, P. Su, and C. Te Chuang, “Analysis of Single-Trap-Induced Random Telegraph Noise on FinFET Devices, 6T SRAM Cell, and Logic Circuits,” *IEEE Int. Reliab. Phys. Symp. Proc.*, vol. 59, no. 8, pp. 2227–2234, 2012, doi: 10.1109/IRPS.2012.6241886.

[118] J. Zou *et al.*, “Deep Understanding of AC RTN in MuGFETs through New Characterization Method and Impacts on Logic Circuits,” *IEEE Symp. VLSI Technol.*, pp. 186–187, 2013.

[119] T. Nagumo, K. Takeuchi, T. Hase, and Y. Hayashi, “Statistical characterization of trap position, energy, amplitude and time constants by RTN measurement of multiple individual traps,” *Tech. Dig. - Int. Electron Devices Meet. IEDM*, pp. 628–631, 2010, doi: 10.1109/IEDM.2010.5703437.

[120] Hitachi Ltd, “Hitachi Develops a New RFID with Embedded Antenna µ-Chip,” *Hitachi.com*, 2003. http://www.hitachi.com/New/cnews/030902.html#:~:text=The µ-Chip%2C announced by,rewritten%2C thus guaranteeing its authenticity. (accessed Jun. 02, 2020).

[121] Z.-T. Lin and V. P.-H. Hu, “Reduced RTN Amplitude and Single Trap induced Variation for Ferroelectric FinFET by Substrate Doping Optimization,” *2019 Silicon Nanoelectron. Work.*, pp. 6–7, 2019, doi: 10.23919/SNW.2019.8782943.

[122] F. M. Puglisi and P. Pavan, “RTN analysis with FHMM as a tool for multi-trap characterization in HfOX RRAM,” *2013 IEEE Int. Conf. Electron Devices Solid-State Circuits, EDSSC 2013*, pp. 1–2, 2013, doi: 10.1109/EDSSC.2013.6628059.

[123] P. Ren *et al.*, “New observations on complex RTN in scaled high-κ/metal-gate MOSFETs - The role of defect coupling under DC/AC condition,” *Tech. Dig. - Int. Electron Devices Meet. IEDM*, pp. 31.4.1-31.4.4, 2013, doi: 10.1109/IEDM.2013.6724731.

[124] Robert G. Brown, “Dieharder: A Random Number Test Suite,” 2017. https://webhome.phy.duke.edu/~rgb/General/dieharder.php (accessed Feb. 28, 2020).

[125] N. D. Truong, J. Y. Haw, S. M. Assad, P. K. Lam, and O. Kavehei, “Machine Learning Cryptanalysis of a Quantum Random Number Generator,” *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 2, pp. 403–414, 2018, doi: 10.1109/TIFS.2018.2850770.

[126] J. Kim *et al.*, “Nano-Intrinsic True Random Number Generation: A Device to Data Study,” *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 66, no. 7, pp. 2615–2626, 2019, doi: 10.1109/TCSI.2019.2895045.

[127] B. Yang, V. Rozic, N. Mentens, and I. Verbauwhede, “On-the-fly tests for non-ideal true random number generators,” *Proc. - IEEE Int. Symp. Circuits Syst.*, vol. 2015-July, pp. 2017–2020, 2015, doi: 10.1109/ISCAS.2015.7169072.

[128] W. Killmann and W. Schindler, “A proposal for : Functionality classes for random number generators 1,” no. September, p. 113, 2011.

[129] K. Abe, A. Teramoto, S. Sugawa, and T. Ohmi, “Understanding of traps causing random telegraph noise based on experimentally extracted time constants and amplitude,” *IEEE Int. Reliab. Phys. Symp. Proc.*, vol. 2, pp. 4A.4.1-4A.4.6, 2011, doi: 10.1109/IRPS.2011.5784503.

[130] M. Toledano-Luque *et al.*, “Temperature and voltage dependences of the capture and emission times of individual traps in high-k dielectrics,” *Microelectron. Eng.*, vol. 88, no. 7, pp. 1243–1246, 2011, doi: 10.1016/j.mee.2011.03.097.

[131] X. Yang *et al.*, “Extraction of Position and Energy Level of Oxide Trap Generating Random Telegraph Noise in 65 nm NOR Flash Memory,” *Integr. Ferroelectr.*, vol. 164, no. 1, pp. 103–111, 2015, doi: 10.1080/10584587.2015.1044884.

[132] M. Li, J. Miao, K. Zhong, and D. Z. Pan, “Practical Public PUF Enabled by Solving Max-Flow Problem on Chip,” *Proc. 2016 53rd ACM/EDAC/IEEE Des. Autom. Conf.*, pp. 1–6, 2016, doi:

10.1145/2897937.2898067.

[133] A. Wild, G. T. Becker, and T. Guneysu, “On the problems of realizing reliable and efficient ring oscillator PUFs on FPGAs,” *Proc. 2016 IEEE Int. Symp. Hardw. Oriented Secur. Trust. HOST 2016*, pp. 103–108, 2016, doi: 10.1109/HST.2016.7495565.

[134] A. Vijayakumar, V. C. Patil, C. B. Prado, and S. Kundu, “Machine learning resistant strong PUF: Possible or a pipe dream?,” *Proc. 2016 IEEE Int. Symp. Hardw. Oriented Secur. Trust. HOST 2016*, pp. 19–24, 2016, doi: 10.1109/HST.2016.7495550.

[135] G. Jian, Z. Jie, and Z. Li, “Design of a DC Linear Power Supply with Adjustable Voltage,” *IERI Procedia*, vol. 3, pp. 73–80, 2012, doi: 10.1016/j.ieri.2012.09.013.

[136] J. Daugman, “The importance of being random: statistical principles of iris recognition,” *Pattern Recognit.*, vol. 36, no. 2, pp. 279–291, 2003, [Online]. Available: http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6V14-458WVY9-1&_user=10&_rdoc=1&_fmt=&_orig=search&_sort=d&_docanchor=&view=c&_searchStrId=1044316686&_rerunOrigin=google&_acct=C000050221&_version=1&_urlVersion=0&_userid=10&md5=c109b89698b3e7f6d6.

[137] P.-Y. Chen, R. Fang, R. Liu, C. Chakrabarti, Y. Cao, and S. Yu, “Exploiting resistive cross-point array for compact design of physical unclonable function,” *IEEE Int. Symp. Hardw. Oriented Secur. Trust*, pp. 26–31, 2015.

[138] V. P. Yanambaka, S. P. Mohanty, and E. Kougianos, “Novel FinFET based physical unclonable functions for efficient security integration in the IoT,” *Proc. - 2016 IEEE Int. Symp. Nanoelectron. Inf. Syst. iNIS 2016*, pp. 172–177, 2017, doi: 10.1109/iNIS.2016.047.

[139] B. Rothke, “A look at the Advanced Encryption Standard (AES),” 2007. doi: 10.1201/9781439833032.ch89.

[140] W. Yan, F. Tehranipoor, and J. A. Chandy, “PUF-Based Fuzzy Authentication Without Error Correcting Codes,” *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 36, no. 9, pp. 1445–1457, 2017, doi: 10.1109/TCAD.2016.2638445.

[141] Y. Pang, H. Wu, B. Gao, D. Wu, A. Chen, and H. Qian, “A novel PUF against machine learning attack: Implementation on a 16 Mb RRAM chip,” *Tech. Dig. - Int. Electron Devices Meet. IEDM*, pp. 12.2.1-12.2.4, 2018, doi: 10.1109/IEDM.2017.8268376.

List of Publications

Conferences

1. **James Brown**, Rui Gao, Zhigang Ji, Jiezhi Chen, Jixuan Wu, Jianfu Zhang, Bo Zhou, Qi Shi, Jacob Crawford, and Weidong Zhang., "A low-power and high-speed True Random Number Generator using generated RTN, "*Dig. Tech. Pap. - Symp. VLSI Technol.*, vol. 2018-June, no. 2, pp. 95–96, 2018.

2. Z. Ji, **James Brown**, and J. Zhang, "True Random Number Generator (TRNG) for Secure Communications in the Era of IoT," China Semicond. Technol. Int. Conf. 2020, CSTIC 2020, 2020, doi: 10.1109/CSTIC49141.2020.9282535.

Journals

3. **James Brown**, Jianfu Zhang, Bo Zhou, Mehzabeen Mehedi, Pedro Freitas, John Marsland, Zhigang Ji, "Random - telegraph - noise - enabled true random number generator for hardware security," *Sci. Rep.*, no. 0123456789, pp. 1–13, 2020.

4. **James Brown**, Xinsheng Wang, Zezhong Tu, Yongkang Xue, Jiezhi Chen, Zhigang Ji, Jianfu Zhang, Bo Zhou and Yue Gao, "Understanding Generated RTN as an Entropy Source for True Random Number Generators in IoT", *IEEE IoT* (Under Review 2020)

Letters

5. Zheng Chai, Wei Shao, Weidong Zhang, **James Brown**, Robin Degraeve, Flora D. Salim, Sergiu Clima, Firas Hatem, Jian Fu Zhang, Pedro Freitas, John Marsland, Andrea Fantini, Daniele Garbin, Ludovic Goux, and Gouri Sankar Kar, "GeSe-based Ovonic Threshold Switching Volatile True Random Number Generator," *IEEE Electron Device Lett.*, vol. PP, no. c, p. 1, 2019.

Patents

6. **J. Brown**, Z. Ji, and J. F. Zhang, "True random number generator using generated random telegraph noise," GB2571546, 2019.