Security in Approximate Computing and Approximate Computing for Security: Challenges and Opportunities

Security in Approximate Computing and Approximate Computing for Security: Challenges and Opportunities

Weiqiang Liu, *Senior Member, IEEE,* Chongyan Gu, *Member, IEEE,* Máire O'Neill, *Senior Member, IEEE,* Gang Qu, *Senior Member, IEEE,* Paolo Montuschi, *Fellow, IEEE,* and Fabrizio Lombardi, *Fellow, IEEE*

(Invited Paper)

***Abstract*—Approximate computing is an advanced computational technique that trades the accuracy of computation results for better utilization of system resources. It has emerged as a new preferable paradigm over traditional computing architectures for many applications where inaccurate results are acceptable. However, approximate computing also introduces security vulnerabilities mainly due to the fact that the uncertain and unpredictable intrinsic errors during approximate execution may be indistinguishable from malicious modification of the input data, the execution process, and the results. On the other hand, interestingly, approximate computing present new opportunities to secure the system and the computation. Existing work on the security of approximate computing covers threat models, countermeasures, and evaluations, but lacks a framework for analysis and comparison. In this article, we provide a classification of the state-of-the-art works in this research field, including threat models in approximate computing and promising security approaches using approximate computing. Open questions and potential future research directions are also discussed.**

***Index Terms*—Approximate computing, hardware security, cryptography**

I. Introduction

In the last couple of decades, various advanced computing systems, including supercomputers, ubiquitous computing centers, and servers have been developed and widely deployed. Unfortunately, Moore's law is approaching its limit [1], and conventional computing techniques are not effective in providing further performance enhancement under physical restrictions such as power consumption. These new and emerging limitations have opened to many new challenges and some interesting solutions. One of the most promising directions to explore is to focus on a suitable reduction of the computational accuracy without sacrificing functionality and "perception". This approach is known as "approximate computing".

Inspired by the fault tolerance capability of the human brain, approximate computing can accept errors in calculation without affecting the results of certain human perception and recognition related computation, such as artificial intelligence (AI), (deep-) machine learning (ML), signal processing and communication, *etc.*, in which noisy data, redundant information, and inaccurate results are tolerable for the computation. Research in this area has attracted a large amount of interests from both academia and industry [2], [3], [4]. Approximate computing techniques are crucial for energy efficient systems and are being considered for high speed and low power nanoscale integrated circuit (IC) designs [5], [6], [7]. For example, a probabilistic design method was proposed to reduce the energy for multimedia applications by deliberately dropping the decoding of certain frames as long as the introduced errors cannot be noticed by human [8]. Another example is Google's deep learning (DL) chip, where the tensor processing unit (TPU) achieves a significant improvement in processing performance using common approximate computing techniques, such as precision scaling [5]. Recently, IBM research has launched the project of building on-chip AI accelerators with approximate computing techniques [6]. They utilize multiple approximate computing techniques, including precision scaling and approximate arithmetic units with various precision, and achieve a 4x-200x speedup over existing methods [7].

Most of the current research focus on using approximate computing to improve system performance with acceptable loss of accuracy [3], [9]. However, approximate computing also introduces security vulnerabilities [10], mainly because of the uncertainty and unpredictability of the intrinsic errors during approximate execution, which may be indistinguishable from malicious modification of the accurate result. If approximate computing can have security vulnerabilities, applications using such techniques will undoubtedly be affected. It is also pointed out [10] that approximate computing is well-suited for security applications. For example, approximate circuits, based on simplified circuits which can reduce area and power consumption, have been proposed for information hiding [11]. Compared to conventional security solutions based on exact circuits, approximate circuit-based security strategies not only provide the same security level but also save hardware

Weiqiang Liu is with College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics (NUAA), China, 211106, email: liuweiqiang@nuaa.edu.cn.

Chongyan Gu and Máire O'Neill are with Centre for Secure Information Technologies (CSIT), Institute of Electronics, Communications & Information Technology (ECIT), Queen's University Belfast (QUB), U.K., BT3 9DT, email: cgu01@qub.ac.uk, m.oneill@ecit.qub.ac.uk.

Gang Qu is with the Department of Electrical and Computer Engineering (ECE) and Institute for Systems Research (ISR), University of Maryland, College Park, MD, USA, email: gangqu@umd.edu.

Paolo Montuschi is with the Department of Control and Computer Engineering, Politecnico di Torino, Italy, email: paolo.montuschi@polito.it.

Fabrizio Lombardi is with Electrical and Computer Engineering (ECT), Northeastern University, Boston, USA, email: lombardi@ece.neu.edu.

Manuscript received xx xx, 2020; revised xx xx, 2020.

resources. Approximate computing opens up both challenges, *security for approximate computing*, and opportunities, *approximate computing for security*.

Several recent works have studied approximate computing and security. Some potential security vulnerabilities that may affect the integrity and security of approximate computing systems are reported in [12] with focus on approximate circuits and storage, including approximate DRAM, phase change memory and SRAM. Security threats of approximate computing from the perspective of hardware, namely, Side-Channel Analysis (SCA), reverse engineering, cloning/counterfeiting and active attacks are discussed in [10]. Case studies in approximate computing-based hardware security applications and future research directions are reported in [13]. These papers provide an initial introduction and discussion for this emerging field.

However, to date the existing works lack a comprehensive and systematized analysis and comparison of threat models, countermeasures, and evaluations. This article aims to filling this gap. More specifically, the main scientific contributions are as follows.

- We provide a classification of the challenges in approximate arithmetic circuits and approximate storage.
- We present a classification of approximate computing applications in building security schemes, such as approximate computing for cryptography, hardware security and machine learning based-schemes.
- We propose open questions and potential future research directions in the area of *security in approximate computing* and *approximate computing for security*.

This article is organized as follows. Section II provides the background treatment to approximate computing, including approximate computing design objectives and classifications. Section III introduces some advanced cryptographic schemes which are treated in more details in later sections. Section IV presents a classification of the security threats in approximate arithmetic circuits and approximate storage. A classification of approximate computing for security is discussed in Section V. The use of approximate computing for security, such as cryptography, hardware security and machine learning based security approaches is presented. Section VI describes future research directions. Conclusions are drawn in Section VII.

II. Approximate Computing

Approximate computing [9], [2], [3], [4] is driven by applications that are related to human perception and inherent error resilience, such as digital signal processing (DSP), communication, multimedia, machine learning and pattern recognition. It can be applied to these applications due to the large and redundant data sets that contain significant noise; therefore, numerical exactness can be relaxed. In this section, the design objectives associated with approximate computing, including the relationship between speed performance, power and accuracy of an approximate computing design are introduced. In [14], approximate computing is classified based on the levels on which approximate computing is applied, namely algorithm, application, architecture and circuits. In this article, approximate computing is classified based on the approximate level (Section II-B1) and the behavior determinism (Section II-B2) [9], [14].

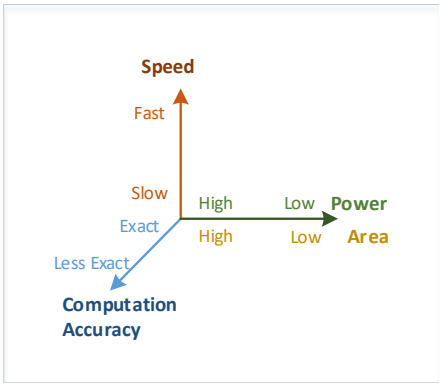
A. Design Objectives

Approximate computing can reduce power consumption and improve system performance by introducing acceptable errors. As such, computation accuracy has been introduced as a third design parameter in addition to delay and power/area consumption as shown in Fig. 1. In Fig. 1(a), the 3-dimension (3D) design space presents the design objectives: performance, power/area consumption and an additional dimension, computation accuracy. Conceptually, the more accurate the computation is, the slower/poorer the performance is and the more power and area the system consumes. The process of designing a system with approximate computing is to balance these design objectives in the 3D design space. More specifically, as shown in Fig. 1(b) on the right, the baseline design S_0 is the traditional design that provides exact results and optimized for speed and power. With approximate computing, both speed and power could be improved. For example, design S_1 has errors within the "Preferred" range, but provides both speedup and power saving. If more computation error is acceptable, design S_2 can further improve system performance. Any design beyond the right of the vertical line S_2 will be considered a failure because it has unacceptable errors.

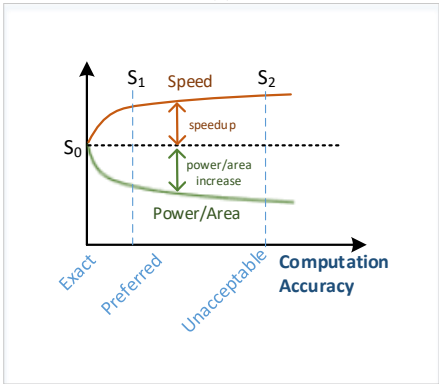
B. Classification

1) Abstraction Level of the Approximation: Approximate computing can be applied to different categories, in hardware and software and in different layers of systems. A classification of approximate computing techniques based on approximate level is summarized as follows.

- *Software Approximation*: Power consumption is reduced by using simplified functions or data in programs. For example, loop perforation [15], precision scaling [16], [17], [18], using program versions of different accuracy [19], and data sampling [20],
- *Approximate Architectures*: Approximate errors can be detected or optimized in approximate accelerators [21] or programmable processors [22]. Other techniques include memory access skipping [23], lossy compression [24], [25], and unreliable emerging technologies [26].
- *Approximate Storage*: Approximate storage is emerging as an efficient technique to reduce a significant portion of system power consumption. The techniques include reducing refresh rate for DRAM [27], voltage scaling [28] and inexact read/write [29].
- *Software/Hardware Codesign*: Most research on approximate computing focuses on either software or hardware. Software and hardware coordinated designs have also been presented to achieve efficient, high performance and dedicated outputs using approximate approaches. A hardware/software codesign method was proposed for approximate semi-supervised K-means clustering [30]. It reduced the power consumption while only a small loss of accuracy is introduced. An automatic hardware platform



(a)



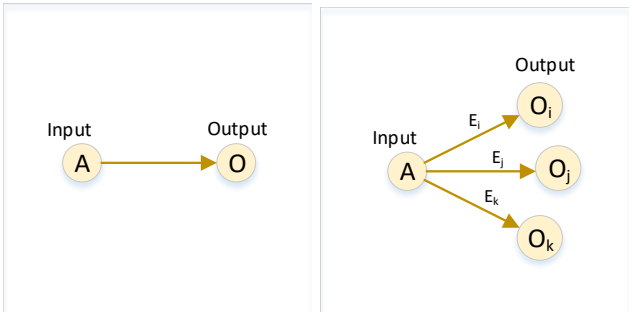
(b)

Figure 1: Approximate computing design space: (a) 3-D with performance, power and computation accuracy, (b) the trade-off between system performance (speed and power/area) and computation accuracy (error).

with approximate algorithm operations was demonstrated in [31]. The incremental network approximation (INA) method has also been proposed to integrate approximate circuits with deep neural network (DNN) algorithms with little loss of accuracy [32]. A similar work [33] also proposed an approximate multiplier to improve accuracy and hardware efficiency of neural networks (NNs).

- *Approximate Arithmetic Circuits*: this involves simplifying circuit designs to achieve an approximate operation of the desired function, such as addition, multiplication and division. The main approximate arithmetic units proposed to date include approximate adders [34], [35], approximate multipliers [36], [37], [38] and approximate dividers [39]. Other approximate arithmetic circuits proposed include approximate fast fourier transform (FFT) [40] and approximate CORDIC circuits [41].
- *Underprovisioned Circuits*: the circuits, which are adjusted to operate at extreme conditions, such as power boundaries, can easily trigger errors and achieve lower power consumption. Relevant techniques include voltage overscaling [42] and frequency overscaling [43].

2) Deterministic and Non-deterministic: The classification of deterministic and non-deterministic for approximate computing depends on the output of the approximated design [44]. A deterministic design repeatedly returns the same output when given the same input as shown in Fig. 2(a). In contrast, Fig. 2(b) presents a non-deterministic design which may return different outputs for the same input. For a deterministic approximate design, a constant error E is generated when given the same input A. However, a non-deterministic approximate design generates different errors, E_i, E_j, E_k for the same input A, which leads to different outputs, O_i, O_j, O_k, respectively. To ensure that the errors, E_i, E_j, E_k, are acceptable for the underlying system, an error threshold θ is necessary for evaluation. However, it is not necessary for a deterministic approximate design. Therefore, non-deterministic approximate designs have limited reproducibility.



(a) (b)

Figure 2: Approximate computing classification based on reproducibility: (a) deterministic design, (b) non-deterministic design.

Examples of deterministic approximate computing techniques in the aforementioned publications are loop perforation [15], precision scaling [17], [16], [18], using program versions of different accuracy [19], data sampling [20], lossy compression [24], [25], optimizing uncertain data [45], the automatic hardware platform with approximate operations [31], approximate adders [34], [35] and approximate multipliers [36], [37], [38]. The non-deterministic approximate computing techniques of the above mentioned research include approximate accelerators [21], programmable processors [22], unreliable emerging technologies [26], refresh rate reduction [27], voltage scaling [28], inexact read/write [29], voltage overscaling [42] and frequency overscaling [43].

In principle, a system that enables approximate computing to trade off accuracy for delay/power/area should ensure the same security as its exact counterpart. However, to date the security issues of approximate computing have not been fully investigated and it is difficult to guarantee the security of operations that are approximated. Adversaries could target components of an approximate computing system, for example, software programs, processors, accelerators, memories and circuits. The expected cost of protecting will be higher when the approximation level is at the architecture layer and hardware circuit design layers since system developers, engineers, and circuit designers all may need to be involved. When security vulnerabilities exist in these approximate designs/systems, test, detection and modification processes are more complicated than for conventional computing. Attacks to the deterministic and non-deterministic approximate methods

are different. In the subsequent sections, the vulnerabilities, attacking techniques and potential countermeasures for approximate computation will be discussed.

III. Security and Cryptographic Schemes

In this section, we will introduce some cryptographic schemes, hardware security and attacking techniques, which are referred to in later sections, both in the context of how they can affect the security of approximate computing designs and how they can benefit approximate computing. A summary of these concepts is provided in Table I.

A. Hardware Security

1) Hardware Trojan (HT): A hardware Trojan (HT) is any modification or addition to a circuit for malicious purpose [46], [47], [48]. Common malicious goals of HT include leaking sensitive information, changing or controlling the functionality of the circuit, and reducing circuit reliability. An HT can be inserted into IC products at any untrusted phase of the IC production chain. The size of an HT varies from several logic gates to a large functional circuit. One important feature of HT is how to activate or trigger the HT for the malicious goals. Most of the HTs are triggered by rare event or signal such that they will not be discovered easily. HT detection and prevention is a very important and challenging problem for trusted IC design.

2) Logic Locking: Logic locking involves hiding important information, for example, functionality and implementation, related to a circuit by inserting additional logic into the original design. It aims to thwart IP piracy leakage, HTs, reverse engineering and IC overproduction [49], [50]. To execute its valid functionality and generate correct outputs, a secret key is provided as input to the logic locking circuit. If a wrong key is applied, the functionality is incorrect and erroneous outputs are generated by the locked circuit.

3) Physical Unclonable Function (PUF): A PUF is a security primitive, which utilizes the inherent process variations during manufacturing to generate a unique digital fingerprint that is intrinsic to the device itself. As such natural variations in silicon dies are out of the manufacturer's control, they are inherently difficult to clone, and can provide additional tamper-evident properties [51], [52], [53], [54]. PUF architectures can be broadly classified into Weak PUF and Strong PUF (SPUF) as discussed in [55]. SPUFs have a large number of possible challenge response pairs (CRPs), whereby each challenge will return a random response that is unique to the physical device. By design, this implies the requirement for a much larger entropy pool such that related challenges should not lead to the corresponding responses on the same device. Hence, SPUFs have been proposed for use in applications such as lightweight mutual authentication, *etc.* However, most SPUF architectures based on linear and additive functions have been shown to be vulnerable to ML attacks. To date, linear regression (LR), support vector machine (SVM), and evolutionary strategy (ES) based ML methods have been widely utilized to attack SPUFs [56], [57], [58].

4) Reverse Engineering: In the semiconductor industry, the patent related technical information of a product are the most valuable for the company who designs, manufactures, and owns the product. However, an adversary can deconstruct an IC to reveal the layout, netlist, architecture or extract knowledge from a hardware circuit [59]. This process is commonly referred to as reverse engineering.

5) Side Channel Analysis (SCA): SCA can reveal sensitive information from the implementation of security/cryptographic schemes by observing their electrical characteristics while operating. The adversary observes side channel leakages, such as the power consumption or eletromagnetic emanations, from the implementation to uncover the secret cipher key or reveal details of the execution/data in the scheme. SCA can be classified into as *invasive* and *non-invasive* or as *passive* and *active*. Invasive SCA requires the device to be de-packaged/reverse engineering to some extent before its behavior is observed. In contrast, non-invasive SCA does not need the device to be directly accessed during the attack. Passive SCA passively observes the behavior of the device's implementation while active SCA seeks to deliberately manipulate the inputs of the device, for example, carrying out fault injections, at the same time as observing its behavior. SCA is considered to be potentially harmful to approximate circuits, and this will be introduced in details in the next section.

B. Cryptography

1) Homomorphic Encryption (HE): Homomorphic encryption is a cryptographic approach that can perform calculations directly on encrypted data without needing to decrypt the data first. It allows a third party to analyze and apply functions on encrypted data without the risk of information/privacy leakage, which enables important applications, for example, securing data in the cloud and providing data analytics in regulated industries. A survey of various homomorphic encryption algorithms and schemes can be found in [60].

2) Post-Quantum Cryptography (PQC): In the near future, quantum computers may break today's most popular public-key cryptographic systems, including RSA, elliptic-curve cryptography, DSA, and ECDSA. PQC is a branch of cryptography that operates on today's classical computers but are based on mathematical problems that are not under threat from attacks by known quantum algorithms [61], [62].

3) Lattice-Based Cryptography (LBC): Lattice-based cryptography (LBC) is one of the most popular branches of PQC due to its versatility, its security hardness and the fact that it can be constructed efficiently on various computing platforms. In addition to conventional encryption and signatures, LBC can be flexibly applied to other constructions, such as identity based encryption, attributed based encryption and fully homomorphic encryption. The LWE problem is hard as several worst-case lattice problems; it is defined as: $As+e = b \, mod \, q$, so given (A, b), find s, where e is an error vector in a Gaussian distribution and q is a field modulus.

IV. Security Threats in Approximate Computing

In this section, we will review both existing attacks and potential security threats in approximate computing. We will

Table I: List of Hardware Security and Cryptographic Schemes

Category	Concept	Description
Hardware Security	Hardware Trojan (HT)	A malicious alteration to the original design of an IC during design or fabrication.
	Logic Obfuscation	A circuit includes logic encryption/locking and IC camouflaging techniques. It inserts additional gates to hide the correct functionality and gate-level implementation of a design.
	Physical Unclonable Function (PUF)	A circuit that uses manufacturing process variations to generate a unique unclonable digital fingerprint.
	Reverse engineering (RE)	The adversary deconstructs an IC to reveal the design, architecture or extract knowledge from the hardware circuit.
	Side channel analysis (SCA)	The adversary observes information such as power consumption during execution and sue such information to reveal the secret.
Cryptography	Homomorphic Encryption (HE)	A cryptographic scheme allows arbitrary arithmetic function on encrypted data without the need of decryption.
	Post-Quantum Cryptography (PQC)	Cryptographic algorithms that are invulnerable to known quantum algorithm attacks by a quantum computer.
	Lattice based Cryptography (LBC)	One of the most promising candidates for PQC, constructed using lattices.
	Learning With Errors (LWE) problem	Defined as $As + e = b \ mod \ q$, given (A, b), find s, where e is an error vector in a Gaussian distribution and q is a field modulus.

first introduce a general classification of these attacks and then elaborate on approximate hardware (*i.e.*, approximate arithmetic circuits and approximate memory).

A. Classification of Attacks

Security threats in approximate computing can be classified as either *confirmed* (successful attacks that have been demonstrated in the literature) or *potential* (harmful but no concrete evidence reported yet). Similarly, applications can be classified as *confirmed affected applications* and *potential affected applications*. Designers need to be aware of both the *confirmed* and *potential attack* techniques for the *potential affected applications*.

Confirmed attacks include those by maliciously using voltage scaling technique [63], changing approximate computing signals or voltage to create incorrect DRAM refresh rate [12] and de-anonymization [64]. The affected hardware units include approximate adders [34], [63], approximate DRAM [65], [66], [67], approximate SRAM [68], [69], [70], [71] and approximate PCM [24]. This impacts almost all the high level applications as long as they use the approximate computation units or memory.

Potential attacks could emerge based on reported vulnerabilities including malicious modification of data and control signals [12], insertion of HT [11], [10], facilitating reverse engineering [10], and performing SCA [10]. As most of the attacks have been demonstrated on approximate adders, it has been reported that approximate multiplier [36], [37], [38], approximate divider [39] and other logic units potentially can also be attacked.

B. Approximate Arithmetic Circuits

Arithmetic units including adders, multipliers and dividers are essential for computation because they not only produce the results but also determine and significantly affect the performance and power consumption of computation. For cognitive applications, such as image and pattern recognition, data analysis, and computer vision, certain level of errors can be tolerated and approximate computing principles have been adopted. The heavy usage of the arithmetic operations has motivated the approximate circuit implementations to carry out these operations. Most of the reported designs are based on logic reduction and pruning methods to implement approximate adders and multipliers for high performance and low power. Since security is not considered as a design objective, these approximate circuits may be vulnerable to security threats as we will elaborate next.

1) Malicious Modification of Registers: By deliberately manipulating the adder's inputs, the attacker can force the approximate adder to continuously generate erroneous outputs. As a result, the error correction code (ECC) or fault tolerant process will be activated more than usual. It is reported that with this malicious modification attack, the power consumption of the approximate adder when 50% errors are acceptable is higher than the same adder when only 25% errors are acceptable [12]. This violates the principle of approximate computing as shown in Fig. 1.

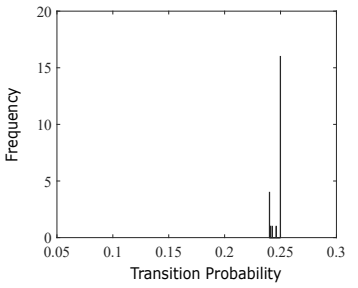
Fig. 3 shows the images of a potential malicious modification attack on an approximate logarithmic multiplier (ALM). The ALM works by truncating certain number of bits, which is controlled by the truncation parameter t, stored in a register. The proposed attack deliberately tampers the truncation parameter after it is read out from the register in order generate unexpected output values. In this example, Fig. 3(a) represents the exact result of the ALM calculation with both 8-bit input and 8-bit output. Fig. 3(b) to Fig. 3(e) are the results of the same image generated by the ALM with different truncation parameter t ($t = 6, 4, 3$ and 2, respectively). An attacker may deliberately change t to 2, say from 6, and produce image Fig. 3(e) that may not be acceptable to the user. In circuits protected by ECC/FT mechanisms, this fools the user to believe that the image with $t = 6$ is unacceptable. Consequently, the user will either not use approximate computing or have to activate the ECC for fault tolerant process and waste power. The attacker achieves this without modifying the value of inputs.

2) Hardware Trojans: Approximate circuits may be more vulnerable for HT insertion compared to exact circuits [10]. First, approximate circuits are controlled normally by additional signal. As this is not part of the original design, it could be exploited to trigger the HT. Second, if ECC or other error detection circuits are accompanied with the approximate circuits, they become another target for HT insertion. Third, signals in an approximate circuit and its corresponding exact circuit may have very different transition probability [72]. For example, Fig. 4 depicts the transition probability distributions for an exact 8-bit adder (Fig. 4(a)) and an approximate 8-bit adder (Fig. 4(b)). They are quite different and might affect how an HT will be triggered. Finally, approximate circuits can be used to facilitate HT detection, which we will elaborate in the future work section.

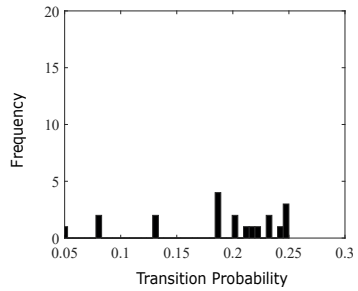
3) Voltage Scaling and Reverse Engineering: [10] discussed how approximate circuits may leak information at some operating points using voltage scaling techniques. [63]

(a) (b) (c) (d) (e)

Figure 3: Potential malicious modifications on the truncation parameter (t) of approximate logarithmic multiplier (ALM): (a) original result with 8-bit input, (b) $t = 6$, (c) $t = 4$, (d) $t = 3$ and (e) $t = 2$.



(a)



(b)

Figure 4: The transition probability distributions for (a) 8-bit exact adder and (b) 8-bit approximate adder.

utilized a voltage over-scaling based approximate computing method to slow down the signal propagation in a circuit and force errors. Due to process variation, the errors will occur on different paths for different chips. This is similar as the principle of a PUF in hardware security. Hence, the erroneous outputs can be utilized as an identity to authenticate the chip [63], but it could leak information about the chip and the data it is processing. Voltage scaling techniques have also been utilized for approximate storage which will be discussed in the next section.

Reverse Engineering (RE) can be affected by approximate circuits in multiple ways [10]. On one hand, because approximate circuits normally are accompanied by additional control and ECC units, this can help an RE attacker to identify the approximate parts and partition the circuit. On the other hand, some approximate circuits may have different structures from the exact circuits and will complicate the RE process. Moreover, the concept of approximate circuit allows RE attackers to reveal and reconstruct an approximate circuit instead of one identical to the original exact circuit [10].

C. Approximate Storage

Storage is another important system component in approximate computing. Memory access is extensive in many error-tolerant applications. This has led to the rise of designing approximate memories or storage to achieve large power savings.

1) Approximate DRAM: Due to its low cost, longevity and high density, DRAM is still the main option for memory in most embedded systems. However, data stored in DRAM must be periodically refreshed, which results in significant power wastage. The concept of approximate computing has been used to solve this problem. Data in an application can be split into critical and non-critical parts and allocated to different parts of memory with different refresh rates, where low refresh rate is utilized to save energy on non-critical data despite of some errors [65]. A hardware based approximation method is proposed to refresh the most important bits of operands at a higher refresh rate and alternatively the least important bits of operands at a lower rate [66]. Software-based approaches have also been proposed to modify software and change DRAM controller to improve energy quality [67]. We now use two examples to demonstrate the security vulnerabilities of these approaches.

Fig. 5 presents the relationship between DRAM refresh rate and error rate, where a low fresh rate leads to high bit error rate. Fault tolerant mechanisms such as ECC are used to correct the errors. However, when the refresh rate drops below an acceptable range, error correcting codes are no longer feasible, and the DRAM will not function correctly. Approximate DRAMs expands the acceptable operating area due to the error tolerant nature of approximate computing. However, if the refresh counter is manipulated by an adversary, DRAM could be refreshed at a lower rate, causing malicious errors in the stored data. This can also be achieved by manipulating DRAM's configuration signal [12].

2) Approximate SRAM: Supply voltage scaling, which can reduce the power consumption on memory accesses, is a preferred for SRAM array in image processing and multimedia applications although it leads to high bit error rates. A dynamically reconfigurable SRAM array is proposed to use a low voltage for cells storing the least important bits (LSBs) and the nominal voltage for cells storing the most important bits (MSBs). The error rates can be controlled at run-time by reconfiguring the number of bits in the lower voltage mode [68]. A voltage scalable architecture is built to

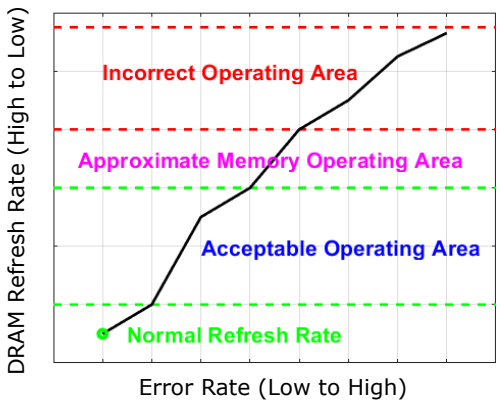


Figure 5: The relationship between DRAM refresh rate and error rate leads to different operating conditions.

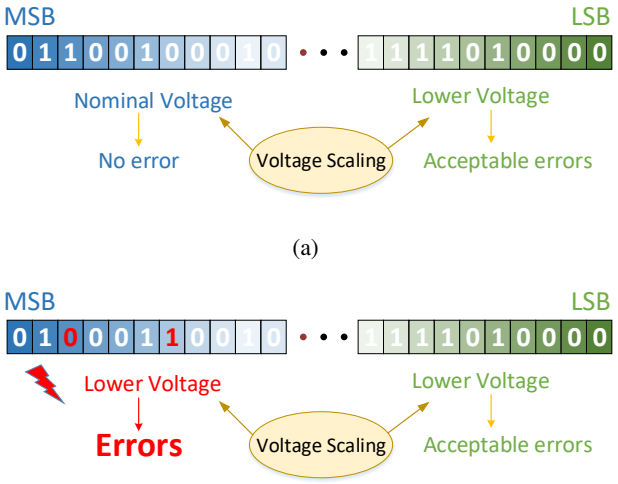


Figure 6: An example of attacks on approximate SRAM: (a) normal voltage scaling technique for SRAM to generate acceptable errors on the LSB and no errors on the MSB, (b) maliciously applying voltage scaling to the MSB to introduce unacceptable errors [68].

save power by storing different 'quality' data in SRAM bit-cells of different 'quantity' [69]. The principle is to save the most sensitive data in video applications in higher order 8T bit-cells supported by nominal voltage while the lower order bits are stored in 6T bit-cells with the supply voltage scaling technique. The errors/failures caused by low voltage in the 6T bit-cells are acceptable in error-tolerant applications such as video processing.

These approximate SRAMs may also be vulnerable to security attacks. For example, errors more than what can be tolerated can be introduced to overburden the ECC unit in the memory [12] as illustrated in Fig. 6. A typical supply voltage scaling technique for SRAM [68] is shown in Fig. 6(a), where a low voltage is applied to the LSBs and the nominal voltage is executed for the MSBs. However, an adversary can manipulate the voltage scaling technique and introduce errors to the MSBs as in Fig. 6(b).

3) Approximate Phase-Change Memory (PCM): PCM is one type of non-volatile memory (NVM) that can be considered as a replacement for disk, flash and potentially DRAM, to solve some of their disadvantages, such as, DRAM's scaling woes and the slow performance of flash solid state drivers. But PCM has its own drawbacks such as low speed, high power consumption, and limited lifetime, *etc.* Approximate computing techniques have been proposed to address some of these drawbacks. An approximate storage technique was proposed to improve the performance (in terms of speed) over precise PCM [73].Various security vulnerabilities exist along the writing flow of such approximate PCM as presented in [12]. For instance, a threshold defines the margin between precise and approximate PCM memory blocks. If it is altered to an incorrect value, the critical data stored in the precise PCM memory might be affected. During the writing operation, the writing voltage is gradually increased in each iteration guided by a noise function. The writing operation may fail when the voltage step is maliciously compromised by underestimating or overestimating the noise function. The number of writing iterations depends on the voltage difference, if the sensing circuit is compromised and a voltage offset is added, the data in the PCM will be modified. Finally, if the voltage comparator is disabled, the attacker can directly overwrite critical data stored in the precise PCM memory.

V. Approximate Computing for Security

In the previous section, we discussed research on potential security threats in approximate computing. On the other hand, the additional dimension of approximate computing can benefit security as well. A comprehensive classification of approximate computing based security solutions is shown in Fig. 7. The effective approaches are categorized into two main groups, cryptography and hardware security, and these will be discussed in details in this section.

A. Approximate Computing for Cryptography

This subsection discusses how approximate computing can benefit both traditional cryptographic primitives, *e.g.*, hash functions, and advanced cryptographic schemes. Since advanced schemes are typically more complex than traditional approaches, those that are based on 'learning with errors' mathematical problems can take advantage of the performance benefits that approximate computing can offer.

1) Post Quantum Cryptography (PQC): Discrete Gaussian sampling is a critical constituent of many LBC based schemes [74]. The sampler is often the bottleneck of schemes requiring high performance and its implementation has been successfully attacked by SCA [75], [76].

Rejection sampling, shown in Fig. 8, is a common method employed to execute discrete Gaussian sampling in lattice based cryptography [77]. An integer $x \in \{-\tau\sigma, \cdots, \tau\sigma\}$, where τ is the 'tail-cut' factor, is chosen from a uniform distribution depending on the security parameters. The larger the tail-cut, the higher the precision for each discrete value of the distribution and consequently the higher the security

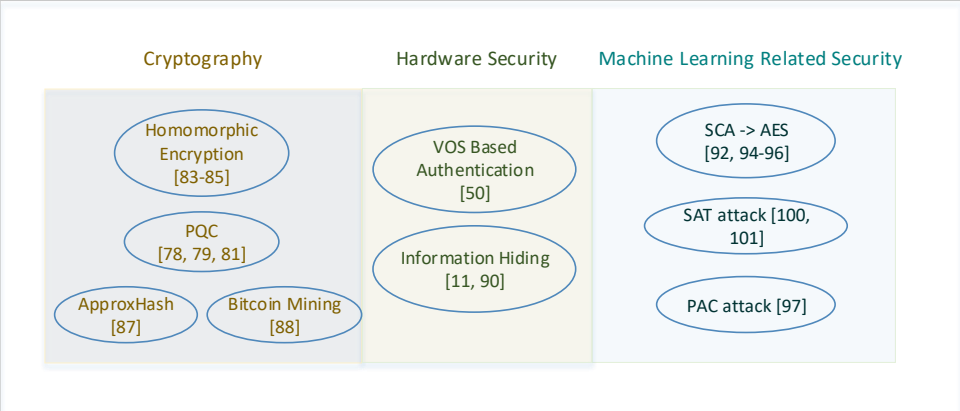


Figure 7: Classification for the applications of approximate computing in cryptography, hardware security and machine learning related security.

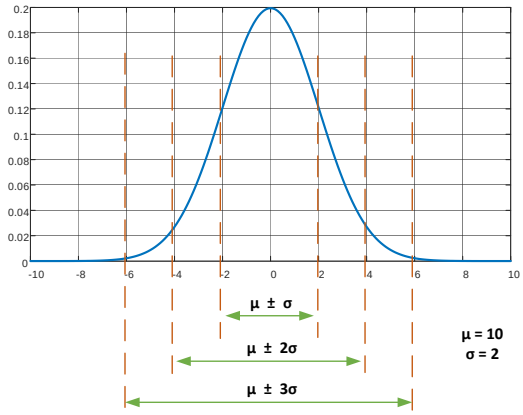


Figure 8: The tail-cut of Gaussian sampling.

achieved; however, the implementation cost is also higher. Hence, there is a trade-off between hardware resource consumption and security level. For ring learning with errors (RLWE), the probability of decryption error is mainly determined by the tail-cut and the standard deviation (STD) of the Gaussian distribution. [78] presented the performance, resource consumption and quality of six conditions of the implemented comparator-based Gaussian sampler for different tail cuts and statistical distances.

In addition to the Gaussian sampling, the modular polynomial multiplication in a RLWE algorithm is also a significant bottleneck in the realization of a practical resource-constrained design for embedded Internet of Things (IoT) devices. Exploiting the inherent approximate nature of the RLWE problem, [79] presented an approach utilizing approximate computing for RLWE based applications as shown in Fig. 9. Fig. 9(a) presents an accurate multiplication for the hardware architecture of RLWE decryption. An optimized dynamic range multipliers (DRUM) approximate multiplier, as shown in Fig. 9(b), has been proposed by [79] to improve the speed, and reduce the area usage and power consumption for RLWE decryption hardware designs.

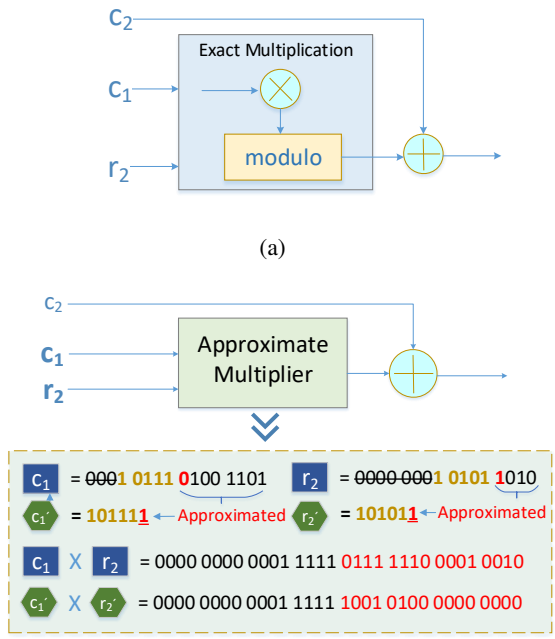


Figure 9: Hardware architecture of RLWE decryption, where (a) includes an exact mulitplication [80] and (b) demonstrates an approximated multiplication using DRUM approximate multiplier.

Later, [81] proposed a design of an area/power efficient approximate modular multiplier (referred to as AxMM) for a RLWE hardware design, by exploiting the statistics of Gaussian noise in addition to the technique proposed in [82]; transforming the unsigned Gaussian data to a signed format. Fig. 10 presents the design of AxMM, comprising an approximate multiplier (AxMult) followed by an approximate modular reduction circuitry (AxMR). The leading one detector (LOD) of AxMult performs a single bit truncation on the Gaussian data (B) there by reducing its width from 6-bit to 4-bit for modulus $q = 7,681$, whereas MSB signed bit (b[5]) is

not utilized during the modular multiplication but it is applied at the end to generate the required result for a negative number. Compared to the smallest exact RLWE multiplier design [82], the AxMM can reduce the area by over 35% and power consumption by over 23% with a slight reduction in STD of the Gaussian distribution as well as the security level.

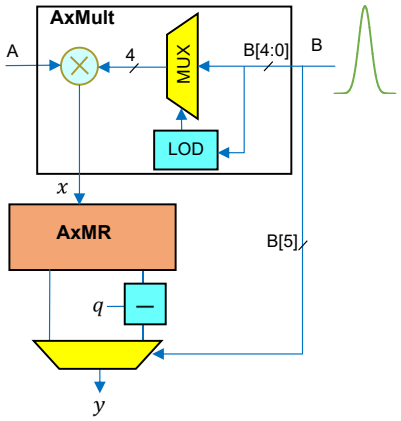
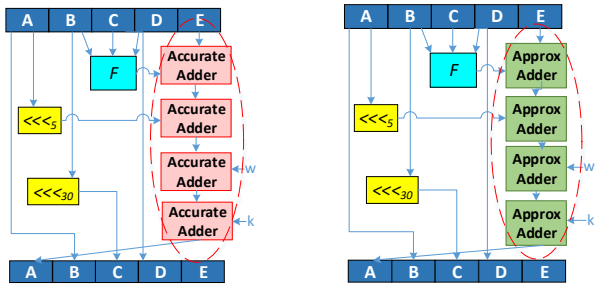


Figure 10: Approximate modular multiplier (AxMM) [81]

2) Homomorphic Encryption: [83] proposed a homomorphic encryption scheme using approximate arithmetic based on the RLWE. It utilized encryption noise as a form of error involving approximate computations. Modular reduction is an important operation in homomorphic decryption. [83] achieved linear complexity in the growth of the cipher-text modulus compared to other work with exponential complexity growth. Subsequent work by the authors [84] presented an approximate bootstrapping operation for homomorphic decryption. Also, [85] utilized the approximate computing techniques proposed in [79] to improve the efficiency of homomorphic decryption. It also proposed a theoretical model to examine the error behavior of secure inference and presented parameters that can achieve smaller ciphertext size.

3) ApproxHash: As a basic building block (see Fig. 11(a)), hash functions have been significantly developed and utilized in many security primitives [86]. An approximate implementation of the Secure Hash Algorithm-1 (SHA-1) as shown in Fig. 11(b), have been proposed to optimize the delay, power and area consumption for cryptographic applications [87]. Approximate modular-32 adders, specifically approximate mirror adders (AMAs), have been utilized to replace accurate modular-32 adders at 80 out of N stages of the conventional SHA-1 to improve the delay, power and area metrics at the cost of degradation in its classical security strength. Hence, one can select appropriate ApproxSHA-1 with N stages of approximation according to the security strength as required by the application. Such an ApproxHash could be utilized in error tolerant applications and pseudo random number generator (PRNG) hardware.

4) Bitcoin Mining: Bitcoin is a crypto-currency, mainly created to simplify transaction processes without needing a third-party, increase the speed of cross-border transactions, and to be independent of government regulations. Bitcoin mining is a process of creating and adding transactions to



(a) (b)

Figure 11: Approximate adders applied to Hash functions [87], where (a) and (b) are basic building blocks of a conventional SHA-1 algorithm using accurate adders and an approximate SHA-1 algorithm using approximate adders, respectively.

the Bitcoin ledger, called Blockchain. Bitcoin mining, based on complex computation, is inherently error tolerant. Since the cost (*e.g.*electricity) for bitcoin mining is very high; hence, low power strategies are important for bitcoin mining. To address this, approximate computing has been considered to improve the efficiency of Bitcoin mining [88]. Approximate circuits can be built to reduce delay and area consumption but trading off reliability. Two forms of approximation, functional approximation and operational approximation, have been proposed in [88]. For functional approximation, approximate circuits have been utilized to replace original circuits to reduce area and delay. Operational approximation, carried out by running the circuits at different timings, such as executing circuits at a higher frequency, accepts Better-than-Worst-Case operation. However, Bitcoin mining utilizes a hashcash based proof-of-work, which can apply approximate circuits for the hardware implementation. For other distributed ledgers, it is unknown if the approximation approach [88] is applicable.

B. Approximate Computing for Hardware Security

Cryptographic algorithms and protocols depend on hardware implementation to achieve real-time performance and more inherent security than software implementation. However, the recent microarchitectural attacks, such as Meltdown and Spectre, on processors demonstrated examples of hardware based attacks. [89] shows that hardware security threats have spread to every corner of the semiconductor supply chain. In this subsection, we introduce countermeasures and potential research directions for hardware security using approximate computing.

1) Information Hiding for Approximate Computing: Providing security to IoT devices is a major challenge as small devices tend to be limited in terms of resources and power. Conventional security approaches, based on computationally complex cryptographic algorithms, are typically too resource intensive for implementation on these devices. To reduce the power consumption for IoT devices and simultaneously provide a practical security solution, Gao *et al.* proposed an intrinsic security strategy [11], based on arithmetic operations

executed by approximate function units, enabling embedded information for authentication and other security related applications. The principle is presented in Fig. 12, where the floating-point based approximate arithmetic computing has 1 sign bit, 8 exponent bits and 23 fraction bits. The left component is the MSB, and the right p bits in the fraction, and the LSB, have little impact on the value. Hence, they can be directly used as *security* bits to hide information without affecting the other $32 - p$ bits. The error introduced to the precision value is 0.0074, which means the last p bits introduce less than 2^{p-24} error compared to the precision format.

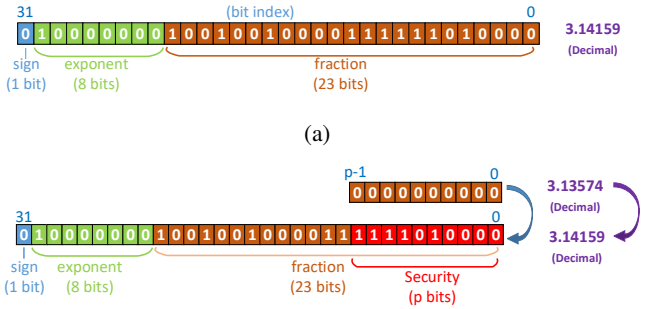


Figure 12: The application of approximate computing to extract security [11]: (a) IEEE 754 single-precision floating-point format for 32-bit data, (b) approximate format with security extraction. The last p LSB bits can be used as security bits to embed information.

With this in mind, two examples of hiding information into approximate computing are shown, one using an approximate adder and the other using and approximate multiplier.

[90] presented an information hiding strategy using an approximate adder based on an accurate configurable adder [91]. A short message M can be deliberately hidden in the operation of the approximate adder and M can be retrieved to detect incorrect results.

Fig. 13 shows the process and an example of applying an approximate multiplier for information hiding [11]. Two real numbers A and B can be written as $A = A' \oplus K_A$ and $B = B' \oplus K_B$ using the approximate format, where A' and B' are the numbers A and B in the approximate format with the last p bits replaced by 0s; K_A and K_B are the last p bits of A and B. $\oplus$ is an XOR operation.

As an example, assume the numbers A and B are 3.14159 and 12.31, respectively. $A \times B = 3.14159 \times 12.31 = 38.6729729$ is obtained for the precise computation, $A' \times B' = 3.1413574 \times 12.30957 = 38.6687588$ is calculated for the approximate computation with $p = 10$. The final result with security information embedded (K_r) is computed as $A' \times B' \oplus K_A \oplus K_B \oplus K_r = 38.67124$, with only a 0.00448 percentage accuracy loss over the accurate result. Hence, compared to direct approximate computing, this approach achieves approximate computing and information hiding at the same time, which can significantly reduce power and hardware resource consumption.

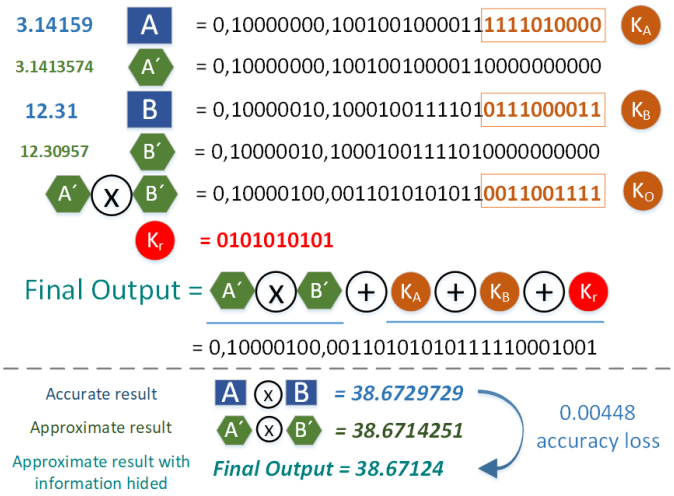


Figure 13: An example of the application of an approximate multiplier for information embedding.

2) VOS based Authentication: Due to the ubiquitous nature of IoT devices, lightweight authentication of an entity is one of the most fundamental problems in providing IoT security. A novel voltage over-scaling (VOS) based lightweight authentication approach is presented in [50] to address this challenge. VOS commonly uses approximate computing to reduce power consumption and can extract information through exacerbating the effects of process variation. Digital circuits and systems normally operate under a nominal voltage to guarantee correct outputs. Properly reducing the operating voltage under the prescribed margin can save considerable amount of power consumption. However, process variation is effected by scaling voltage, which can generate timing errors and thus affect the output precision. Hence, a two-factor authentication scheme that uses passwords and hardware properties was proposed to achieve lightweight authentication for IoT applications in [50], where the authors introduced an example of voltage over-scaling based computation as shown in Fig. 14. An image processing technique, superimposition, is applied to images (a) and (b) to generate a new image (c) by using an accurate ripple-carry adder under regular voltage. Then voltage is over-scaled to a low value which causes errors in the ripple-carry adder. Such errors depend on process variations and will be device dependent. Images (d) and (e) are the superimposed images created by two devices with the same ripple-carry adder. Their patterns of error from the original image (c) are shown in (f) and (g), respectively. The difference between the two error patterns is shown in image (h). Hence, it can be used for digital fingerprint generation and applied to authentication.

C. Approximate Computing for Machine Learning Based Security Approaches

1) Side Channel Analysis (SCA) of Cryptographic Algorithms: In recent years, machine learning techniques have been used to improve SCA attacks. A relatively new approach to SCA profiling attacks involves the application of machine

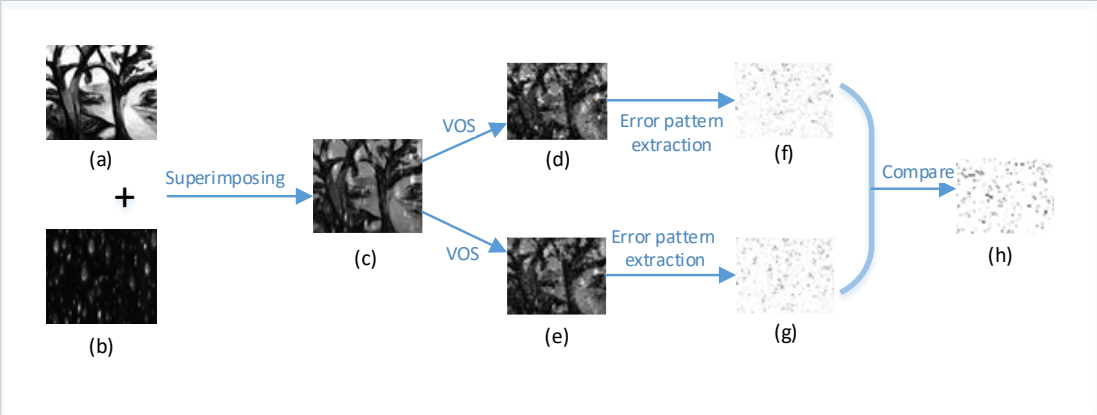


Figure 14: An example of the effect of process variations in voltage over-scaling based computation [50]. Two images (a) trees and (b) snowflakes are superimposed to generate (c) snowfall. When the computation is under voltage over-scaling technique and two adders are identical except the process variations of the hardware, (d) and (e) images are different with the error patterns (f) and (g), respectively, which are the deviations of each adder from the correct image (c). (h) presents the difference between the two error patterns (f) and (g).

learning techniques to improve their efficiency and success rate. It has been shown that these attacks can be even more powerful than the more traditional template attacks in practice, as less assumptions are required on the distribution of the underlying trace data [92], [93]. Much of the research to date has centered on the use of SVMs [94], [95] and random forests [92]. Research by Lerman *et al.* [92] showed how such approaches can be used to uncover the key of a (masked) advanced encryption standard (AES) implementation designed to be resilient against power analysis.

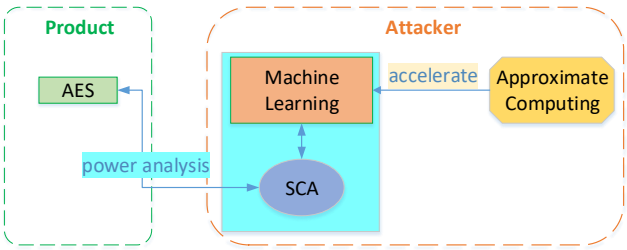


Figure 15: An example of the application of machine learning to SCA. Approximate computing can be used to accelerate the machine learning process and improve the attack efficiency.

An illustration of this idea is shown in Fig. 15. Gilmore *et al.* in [96] built on this research by investigating the novel application of an NN-based attack (that can be accelerated by approximate computing) against a masked AES design. This two-stage attack first uses an NN model to recover the mask, and then uses a second NN model to recover the masked secret data. Combining the knowledge recovered from both attacks allows subsequent key recovery with only a single trace. Similar work has shown how to recover the secret key with only a single model with no knowledge of the mask at a cost of additional traces in the attack stage [93].

2) PUF: The probably approximately correct (PAC) algorithm has been utilized to model $k-$XORed Arbiter PUFs

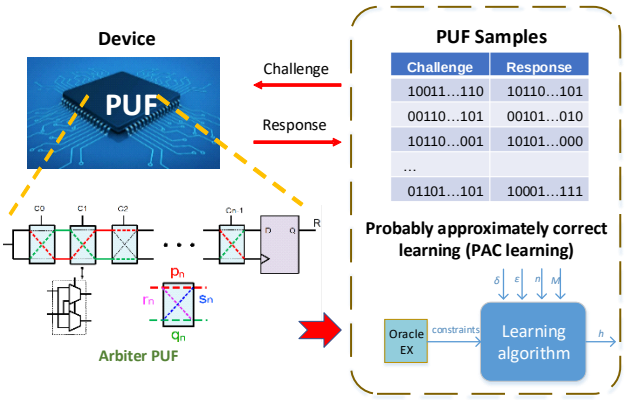


Figure 16: An example of the application of PAC to model an Arbiter PUF design [97].

(APUFs) suitable for $k < 4$ [97] as shown in Fig. 16. In order to prevent modelling attacks, SPUF designs have been enhanced by increasing their complexity. Since approximate computing can be used to significantly improve the performance of machine learning attacks, applying approximate computing based modelling attacks to break SPUF designs will improve efficiency and success rates.

3) Logic Obfuscation: Most traditional circuit obfuscation techniques have been proven to be vulnerable to a Boolean satisfiability (SAT) based attack [98]. The principle of a SAT attack is presented in [99], as shown in Fig. 17(a). The core idea of the SAT attack is to find the correct key using a number of distinguishing input/output (DIO) pairs, which can identify a subset of wrong key combinations (WK_i). In each iteration, a subset of WK_i will be found. Finally all wrong key combinations are identified, therefore the correct one is revealed. SAT resistant countermeasures have been proposed by exponentially increasing the minimum number of queries needed to eliminate all the wrong keys. However, an exact

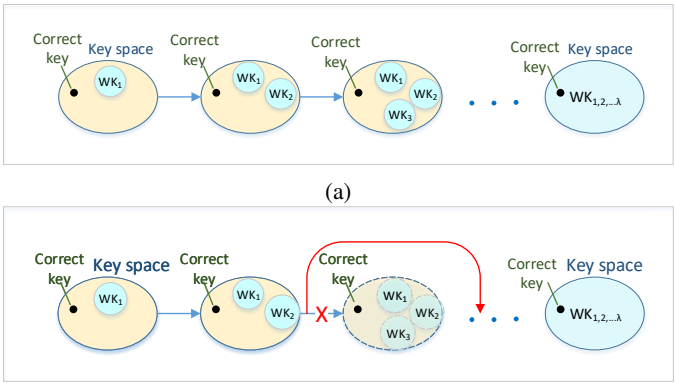


Figure 17: The application of approximate computing to SAT attacks on logic obfuscation: (a) illustration of the iterative SAT attack process [99], (b) an approximate deobfuscation algorithm based on SAT attacks and random testing [101].

deobfuscation accuracy is required for the countermeasure based on implicit assumptions. To address this, [100] and [101] proposed an approximate attack, AppSAT, as shown in Fig. 17(b). It is based on approximate computing algorithm (at software level) to deobfuscate circuits by terminating the attack at an early stage. High corruptibility, or 'compound' schemes, have been proposed to prevent SAT attacks. [102] proposed an approximate SAT-based attack framework to enhance the efficiency of the attack using approximate techniques, which converts a compound SAT attack into a general SAT attack. These approximate strategies are only based on approximate computing algorithms to address the issues in hardware security.

VI. Future Research Directions

We discuss some open questions and potential research directions in the security of approximate computing and how approximate computing can help security.

A. Security in Approximate Computing

1) Error Manipulation: Perhaps the biggest difference between accurate computing and approximate computing is the introduced errors in the result. An accurate computing design is supposed to generate precise results and any error, if occurs, would be unintentional. In contrast, an approximate computing design, by definition, may introduce various errors and give many different results. This creates a severe security threat because adversaries can manipulate the computation process to introduce malicious errors.

As we have seen already, both the input data and the approximate computing control signals could be the target for attack. Furthermore, an adversary could also blatantly change the precise result from an accurate computing, and if caught, blame approximate computing as the source of the error.

Characterizing the errors generated by approximate designs could be helpful in identifying malicious errors inserted by an adversary. For example, a threshold value could be used for errors of approximate designs. Any error that goes beyond the threshold will be considered as a potential malicious attack. However, it is challenging to characterize errors as different approximate computing mechanisms most likely will generate errors with different characteristics. Some interesting research questions are: how to differentiate errors from approximate computing and malicious errors; how to model, analyze and control the errors, and how to set an appropriate error threshold value.

2) Testing: In Section II-B2, we mentioned that approximate computing designs can be classified as deterministic and non-deterministic designs. They have different approximate schemes and the associated error patterns, which open up opportunities for both attacks and new security applications. For example, the example of maliciously using voltage overscaling in Section IV-B3 is only applicable for non-deterministic approximate computing designs. This also brings challenges for testing.

In addition to the objectives for normal testing, testing techniques for approximate computing need to consider whether the approximate mechanism is deterministic or non-deterministic, and whether it is static or dynamic. In general, non-deterministic and dynamic methods would be more challenging to test. Conventional testing techniques may be not applicable to approximate computing designs because acceptable errors in approximate designs may be mistakenly detected as failures using conventional testing methods. Therefore, security and testing need to be developed and evaluated together for approximate computing designs.

3) Hardware Trojans: As we have seen in the example of an approximate adder (Fig. 4 in Section IV-B2), approximate computing could change the transition probability of certain signals dramatically. Such change could make the trigger signal of a hardware Trojan easier to be detected, it could also introduce rare signal values that can be used as the trigger. It would be interesting to study how the change of signal transition probability is related to the errors of approximate circuits and how it affects the activation of hardware Trojans. It is also worth investigating the impact of the types of approximate circuits (adder, multiplier or divider) on transition probability and the possibility of hardware Trojan insertion. Furthermore, as mentioned above, the complicated errors introduced by approximate computing may also provide another opportunity for hardware Trojan.

4) Countermeasures to Attacks on Approximate Computing: Most of the existing work on the vulnerability of and attacks to approximate computing have countermeasures as well. However, their main contribution is on the discovery of new vulnerability and attack. The countermeasures are normally simple and straightforward. Therefore, non-trivial approaches on how to design and evaluate new countermeasures that are effective and robust against attacks will be needed. Since the advantages of approximate computing is the reduction in energy consumption and/or improvements in speed, the countermeasures must be low-cost and efficient.

5) Security in Cross-Layer Approximate Computing: In the future, it will be necessary to perform cross-layer security analysis for approximate computing. Research to date on

approximate computing has spanned from devices to systems, while most has focused on a single level. For example, research on approximate arithmetic circuits is independent of research on approximate algorithms and software. In the case where an approximate algorithm is executed on approximate hardware, the research to date has not fully considered how to make the approximate hardware and the approximate algorithm mutually compensate for errors to realize synergy and achieve the best '3D' (precision, performance and power consumption, in Fig. 1) trade-off. For example, it is difficult to apply highly approximate arithmetic to DNNs due to the effect of error accumulation and the convergence problem in the retraining phase. To address this problem, a hardware-software co-design method is proposed to provide fault tolerance for DNNs and offered additional trade-offs between accuracy and hardware consumption [32]. Another example is the application of approximate computing at both the software and hardware levels to achieve a speedup of $378\times$ in iris recognition process while maintaining an acceptable accuracy. Such multi-level approximate designs lead to new attack surfaces. It is important to investigate and evaluate their security vulnerabilities and develop appropriate countermeasures.

6) Security of Approximate Memory: In Section IV-C, the security vulnerabilities of approximate DRAM, SRAM and PCM have been thoroughly discussed. [12], [64] showed how they could be successfully attacked using incorrect refresh rate, voltage modification and de-anonymisation. [12] discussed how the malicious use of voltage overscaling might be harmful to approximate SRAM memory. The growth of approximate computing technologies is revolutionizing the design of modern computing systems in particular the memory of systems. Emerging non-volatile memory technologies, such as resistive random access memory (RRAM) and magnetic random-access memory (MRAM), have become attractive for future memory hierarchies. Approximate multi-level cell (MLC) has been utilised to these new memory technologies to achieve high energy efficiency and low power consumption. We expect to see more research on security of new emerging approximate techniques. For example, existing attacks on current non-volatile caches and traditional memory systems may be applicable to approximate memories with new security threats from approximate errors. For each of these attacks, it is more important to design the corresponding countermeasures. Ultimately, security may become one of the major design objectives for approximate memory system design.

B. Approximate Computing for Security

1) Advanced Cryptography: Section V-A discussed the application of approximate computing in lattice-based encryption/decryption and homomorphic encryption designs with Gaussian sampling and/or based on the RLWE algorithm, which are inherently approximate in nature. There have been a number of recently proposed techniques that are also build on these components such as lattice-based identity based encryption [103] and lattice-based attribute-based encryption [104]. As we are in the very early stage of this emerging field, there will be many challenges and opportunities. For instance, the complexity of the above schemes is high, it will be interesting to investigate whether and how approximate computing can help to solve this problem. Another example is to study how to balance the security strength, energy efficiency, and performance (speed) with approximate computing components and techniques for cryptographic designs, in particular for IoT and embedded applications.

2) Approximate PUF: Approximate DRAM-based PUF [64], [105] is only one example of PUF designs based on approximate computing. [106] proposed an intrinsic processor PUF by exploiting the fact that a given instruction may fail under different frequency points across different chips. Other approximate processors have also been developed [25], [22]. It is worth investigating the feasibility of building approximate processor based intrinsic PUFs for specific system or architecture level applications. For example, will these behaviors, such as instruction fails in an approximate processor, be distinct enough across different chips to provide sufficient entropy required by the challenge response pair of a PUF? Also, as discussed in Section V-C2, an approximate algorithm, PAC, was utilised to improve the efficiency of the attack on PUF designs [97]. The effectiveness of approximate attacks compared to other attacking techniques, *e.g.*, machine learning based modelling attacks, can be another interesting research topic.

3) Hardware Trojan Detection: A hardware Trojan (HT) is any type of malicious modification to circuits. In Section IV-B2, we have discussed how approximate computing can affect HT insertion and detection. Recently, machine learning based approaches such as NNs have been proposed for HT detection [107], [48]. However, the computational complexity of ML approaches and the time it takes to collect data samples and for a model to converge are among the biggest drawbacks of such HT detection methods. Fig. 18 illustrates a potential approach using approximate computing to accelerate this learning process. Because of the effectiveness of approximate circuit and algorithm in other applications, the efficiency of such learning based HT detection could be significantly improved. However, it will be a challenging question to assess the accuracy of the HT detection due to the error introduced by approximate computing. Here once again we see one of the core challenge in approximate computing is how to balance security, performance and errors.

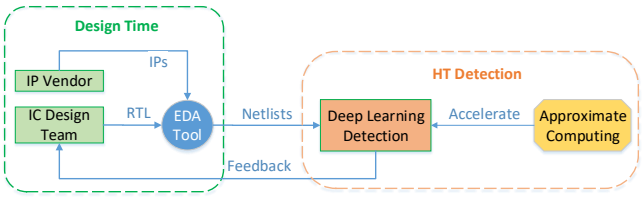


Figure 18: The application of approximate computing to accelerate the detection of HTs.

VII. Conclusion

Due to a high demand for low power but high performance computing systems, approximate computing, which outper-

forms traditional computing architectures, is being rapidly developed and applied to practical systems. It is beneficial for many applications, such as AI, machine learning, image processing, *etc.*, where accurate results are not essential and intrinsic errors are tolerable for the calculation. However, research on security related challenges and opportunities for approximate computing have been neglected to some extent. In this paper, approximate computing circuit designs, multi-layer codesign, state-of-the-art security threats in approximate computing and approaches using approximate computing for both security and cryptography, have been comprehensively reviewed. A classification of the state-of-the-art in this research area, including threat models, existing and potential approaches, has been presented. We hope the classification and review can give researchers a clear understanding of this research area. Currently, security in/for approximate computing has not been widely studied. In particular, the utilisation of approximate computing to enhance security/cryptographic primitives has a promising future.

ACKNOWLEDGMENT

This work is supported by grants from the National Natural Science Foundation of China (62022041 and 61871216), the Fundamental Research Funds for the Central Universities China (NE2019102), the Six Talent Peaks Project in Jiangsu Province (2018XYDXX-009) and the Engineering and Physical Sciences Research Council (EPSRC) (EP/N508664/-CSIT2). We acknowledge Dr. Dur-E-Shahwar Kundi for revising the subsection of approximate computing for PQC.

REFERENCES

[1] J. Hruska, "Nvidia's CEO Declares Moore's Law Dead," 2017.

[2] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Proc. 18th IEEE European Test Symposium (ETS)*, May 2013, pp. 1–6.

[3] Q. Xu, T. Mytkowicz, and N. Kim, "Approximate computing: A survey," *IEEE Design & Test*, vol. 33, no. 1, pp. 8–22, Feb 2016.

[4] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification, and comparative evaluation of approximate arithmetic circuits," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 13, no. 4, pp. 60:1–60:34, 2017.

[5] N. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, and A. Borchers, "In-datacenter performance analysis of a tensor processing unit," in *Proc. 44th Annual International Symposium on Computer Architecture (ISCA)*, 2017, pp. 1–12.

[6] "Unlocking the promise of approximate computing for on-chip ai acceleration," Last accessed 1 July 2020. [Online]. Available: https://www.ibm.com/blogs/research/2018/06/approximate-computing-ai-acceleration/

[7] B. Fleischer, S. Shukla, M. Ziegler, J. Silberman, J. Oh, V. Srinivasan, J. Choi, S. Mueller, A. Agrawal, T. Babinsky, N. Cao, C. Chen, P. Chuang, T. Fox, G. Gristede, M. Guillorn, H. Haynie, M. Klaiber, D. Lee, S. Lo, G. Maier, M. Scheuermann, S. Venkataramani, C. Vezyrtzis, N. Wang, F. Yee, C. Zhou, P. Lu, B. Curran, L. Chang, and K. Gopalakrishnan, "A scalable multi- teraops deep learning processor core for AI trainina and inference," in *IEEE Symposium on VLSI Circuits*, 2018, pp. 35–36.

[8] S. Hua, G. Qu, and S. S. Bhattacharyya, "Energy reduction techniques for multimedia applications with tolerance to deadline misses," in *Proc. 40th annual Design Automation Conference (DAC)*, 2003, pp. 131–136.

[9] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Survey*, vol. 48, no. 4, pp. 62:1–62:33, 2016.

[10] F. Regazzoni, C. Alippi, and I. Polian, "Security: The dark side of approximate computing?" in *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2018, pp. 1–6.

[11] M. Gao, Q. Wang, M. T. Arafin, Y. Lyu, and G. Qu, "Approximate computing for low power and security in the internet of things," *Computer*, vol. 50, no. 6, pp. 27–34, 2017.

[12] P. Yellu, N. Boskov, M. A. Kinsy, and Q. Yu, "Security threats in approximate computing systems," in *Proc. Great Lakes Symposium on VLSI (GLSVLSI)*, 2019, pp. 387–392.

[13] W. Liu, C. Gu, G. Qu, and M. O'Neill, *Approximate Computing and Its Application to Hardware Security*. Springer, 2018, pp. 43–67.

[14] M. Ammar Ben Khadra, "An introduction to approximate computing," *arXiv*, pp. arXiv–1711, 2017.

[15] W. Baek and T. M. Chilimbi, "Green: a framework for supporting energy-conscious programming using controlled approximation," in *ACM Sigplan Notices*, vol. 45, no. 6, 2010, pp. 198–209.

[16] M. A. Anam, P. N. Whatmough, and Y. Andreopoulos, "Precision-energy-throughput scaling of generic matrix multiplication and discrete convolution kernels via linear projections," in *Proc. 11th IEEE Symposium on Embedded Systems for Real-time Multimedia*, 2013, pp. 21–30.

[17] V. Chippa, S. Chakradhar, K. Roy, and A. Raghunathan, "Analysis and characterization of inherent application resilience for approximate computing," in *Proc. 50th Annual Design Automation Conference (DAC)*, 2013, pp. 113–118.

[18] V. K. Chippa, D. Mohapatra, K. Roy, S. T. Chakradhar, and A. Raghunathan, "Scalable effort hardware design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 9, pp. 2004–2016, Sep. 2014.

[19] J. Ansel, Y. L. Wong, C. Chan, M. Olszewski, A. Edelman, and S. Amarasinghe, "Language and compiler support for auto-tuning variable-accuracy algorithms," in *Proc. 9th Annual IEEE/ACM International Symposium on Code Generation and Optimization*, 2011, pp. 85–96.

[20] I. Goiri, R. Bianchini, S. Nagarakatte, and T. Nguyen, "Approxhadoop: Bringing approximations to Mapreduce frameworks," in *Proc. ACM SIGARCH Computer Architecture News*, vol. 43, 2015, pp. 383–397.

[21] D. S. Khudia, B. Zamirai, M. Samadi, and S. Mahlke, "Rumba: An online quality management system for approximate computing," in *Proc. ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, June 2015, pp. 554–566.

[22] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Quality programmable vector processors for approximate computing," in *Proc. 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2013, pp. 1–12.

[23] A. Yazdanbakhsh, G. Pekhimenko, B. Thwaites, H. Esmaeilzadeh, O. Mutlu, and T. C. Mowry, "RFVP: Rollback-free value prediction with safe-to-approximate loads," *ACM Transactions on Architecture and Code Optimization*, vol. 12, no. 4, pp. 62:1–62:26, 2016.

[24] M. Samadi, J. Lee, D. A. Jamshidi, A. Hormati, and S. Mahlke, "SAGE: Self-tuning approximation for graphics engines," in *Proc. 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Dec 2013, pp. 13–24.

[25] Y. Yetim, M. Martonosi, and S. Malik, "Extracting useful computation from error-prone processors for streaming applications," in *Proc. Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2013, pp. 202–207.

[26] H. Cho, L. Leem, and S. Mitra, "ERSA: Error resilient system architecture for probabilistic applications," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 31, no. 4, pp. 546–558, April 2012.

[27] K. Cho, Y. Lee, Y. H. Oh, G. Hwang, and J. W. Lee, "eDRAM-based tiered-reliability memory with applications to low-power frame buffers," in *Proc. IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, Aug 2014, pp. 333–338.

[28] F. Frustaci, D. Blaauw, D. Sylvester, and M. Alioto, "Better-than-voltage scaling energy reduction in approximate SRAMs via bit dropping and bit reuse," *Proc. 25th International Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, pp. 132–139, 2015.

[29] Y. Fang, H. Li, and X. Li, "SoftPCM: Enhancing energy efficiency and lifetime of phase change memory in video applications via approximate write," in *Proc. IEEE 21st Asian Test Symposium (ATS)*, Nov 2012, pp. 131–136.

[30] P. Huang, C. Wang, R. Ma, W. Liu, and F. Lombardi, "A hardware/software co-design method for approximate semi-supervised k-means clustering," in *Proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2018, pp. 575–580.

[31] S. Misailovic, M. Carbin, S. Achour, Z. Qi, and M. C. Rinard, "Chisel: Reliability-and accuracy-aware optimization of approximate computational kernels," in *ACM SIGPLAN Notices*, vol. 49, no. 10, 2014, pp. 309–328.

[32] Z. Liu, K. Jia, W. Liu, W. Qi, F. Qiao, and H. Yang, "INA: Incremental network approximation method for limited precision deep neural networks," in *Proc. IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, 2019, pp. 1–6.

[33] M. S. Ansari, V. Mrazek, B. F. Cockburn, L. Sekanina, Z. Vasicek, and J. Han, "Improving the accuracy and hardware efficiency of neural networks using approximate multipliers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 2, pp. 317–328, 2020.

[34] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," vol. 32, no. 1, pp. 124–137, 2013.

[35] L. Chen, F. Lombardi, P. Montuschi, J. Han, and W. Liu, "Design of approximate high-radix dividers by inexact binary signed-digit addition," in *Proc. Great Lakes Symposium on VLSI (GLSVLSI)*, 2017, pp. 293–298.

[36] W. Liu, J. Xu, D. Wang, and F. Lombardi, "Design of approximate logarithmic multipliers," in *Proc. Great Lakes Symposium on VLSI (GLSVLSI)*, 2017, pp. 47–52.

[37] W. Liu, J. Xu, D. Wang, C. Wang, P. Montuschi, and F. Lombardi, "Design and evaluation of approximate logarithmic multipliers for low power error-tolerant applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 9, pp. 2856–2868, Sep. 2018.

[38] W. Liu, T. Cao, P. Yin, Y. Zhu, C. Wang, E. E. Swartzlander Jr., and F. Lombardi, "Design and analysis of approximate redundant binary multipliers," *IEEE Trans. Computers*, vol. 68, no. 6, pp. 804–819, 2019.

[39] L. Chen, J. Han, W. Liu, and F. Lombardi, "Design of approximate unsigned integer non-restoring divider for inexact computing," in *Proc. ACM 25th Edition on Great Lakes Symposium on VLSI (GLSVLSI)*, 2015, pp. 51–56.

[40] W. Liu, Q. Liao, F. Qiao, W. Xia, C. Wang, and F. Lombardi, "Approximate designs for fast fourier transform (FFT) with application to speech recognition," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 12, pp. 4727–4739, 2019.

[41] L. Chen, J. Han, W. Liu, and F. Lombardi, "Algorithm and design of a fully parallel approximate coordinate rotation digital computer (CORDIC)," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 3, pp. 139–151, 2017.

[42] R. Hegde and N. R. Shanbhag, "A voltage overscaled low-power digital filter IC," *IEEE Journal of Solid-State Circuits*, vol. 39, no. 2, pp. 388–391, Feb 2004.

[43] R. T. Uppu, R. K. Uppu, A. D. Singh, and A. Chatterjee, "A high throughput multiplier design exploiting input based statistical distribution in completion delays," in *Proc. 26th International Conference on VLSI Design and 2013 12th International Conference on Embedded Systems*, 2013, pp. 109–114.

[44] T. Moreau, J. San Miguel, M. Wyse, J. Bornholt, A. Alaghi, L. Ceze, N. Enright Jerger, and A. Sampson, "A taxonomy of general purpose approximate computing techniques," *IEEE Embedded Systems Letters*, vol. 10, no. 1, pp. 2–5, 2018.

[45] J. Bornholt, T. Mytkowicz, and K. S. McKinley, "Uncertain <T>: A first-order type for uncertain data," *SIGARCH Comput. Archit. News*, vol. 42, no. 1, pp. 51–66, 2014.

[46] M. Xue, C. Gu, W. Liu, S. Yu, and M. O'Neill, "Ten years of hardware Trojans: a survey from the attacker's perspective," *IET Computers & Digital Techniques*, August 2020.

[47] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware trojan attacks: Threat analysis and countermeasures," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229–1247, 2014.

[48] S. Yu, C. Gu, W. Liu, and M. O'Neill, "A novel feature extraction strategy for hardware Trojan detection," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[49] M. Yasin, B. Mazumdar, J. J. V. Rajendran, and O. Sinanoglu, "Sarlock: Sat attack resistant logic locking," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2016, pp. 236–241.

[50] M. Arafin, M. Gao, and G. Qu, "VOLtA: Voltage over-scaling based lightweight authentication for IoT applications," in *Proc. 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2017, pp. 336–341.

[51] C. Gu and M. O'Neill, "Ultra-compact and robust FPGA-based PUF identification generator," in *Proc. International Symposium on Circuits and Systems (ISCAS)*, 2015, pp. 934–937.

[52] C. Gu, N. Hanley, and M. O'Neill, "Improved reliability of FPGA-based PUF identification generator design," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 10, no. 3, pp. 20:1–20:23, 2017.

[53] C. Gu, Y. Cui, N. Hanley, and M. O'Neill, "Novel lightweight FF-APUF design for FPGA," in *Proc. 29th Int. Conf. on System-on-Chip (SOCC'16)*. Seattle, WA, USA: IEEE, Sep. 2016, pp. 75–80.

[54] C. Gu, W. Liu, Y. Cui, N. Hanley, M. O'Neill, and F. Lombardi, "A flip-flop based arbiter physical unclonable function (APUF) design with high entropy and uniqueness for FPGA implementation," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–12, 2019.

[55] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, *FPGA Intrinsic PUFs and Their Use for IP Protection*, P. Paillier and I. Verbauwhede, Eds., Vienna, Austria, Sep. 2007.

[56] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th ACM Conference on Computer and Communications Security(CCS)*, 2010, pp. 237–249.

[57] G. T. Becker, "The gap between promise and reality: On the insecurity of XOR arbiter PUFs," in *Proc. International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2015, pp. 535–555.

[58] "On the pitfalls of using arbiter-PUFs as building blocks."

[59] R. Torrance and D. James, "The state-of-the-art in ic reverse engineering," in *Cryptographic Hardware and Embedded Systems - CHES 2009*, C. Clavier and K. Gaj, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 363–381.

[60] P. V. Parmar, S. B. Padhar, S. N. Patel, N. I. Bhatt, and R. H. Jhaveri, "Survey of various homomorphic encryption algorithms and schemes," *International Journal of Computer Applications*, vol. 91, no. 8, 2014.

[61] "Post-quantum cryptography," Last accessed 16 January 2018. [Online]. Available: https://pqcrypto.org/

[62] D. Micciancio and O. Regev, *Lattice-based Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 147–191.

[63] S. Keshavarz and D. Holcomb, "Privacy leakages in approximate adders," *arXiv preprint arXiv:1802.08919*, 2018.

[64] A. Rahmati, M. Hicks, D. E. Holcomb, and K. Fu, "Probable cause: The deanonymizing effects of approximate DRAM," in *Proc. ACM/IEEE 42nd Annual International Symposium on Computer Architecture (ISCA)*, 2015, pp. 604–615.

[65] S. Liu, K. Pattabiraman, T. Moscibroda, and B. G. Zorn, "Flikker: Saving DRAM refresh-power through critical data partitioning," *SIGPLAN Not.*, vol. 46, no. 3, pp. 213–224, 2011.

[66] J. Lucas, M. Alvarez-Mesa, M. Andersch, and B. Juurlink, "Sparkk: Quality-scalable approximate storage in DRAM," in *Proc. Memory Forum*, 2014, pp. 1–9.

[67] A. Raha, S. Sutar, H. Jayakumar, and V. Raghunathan, "Quality configurable approximate DRAM," *IEEE Transactions on Computers*, vol. 66, no. 7, pp. 1172–1187, July 2017.

[68] M. Cho, J. Schlessman, W. Wolf, and S. Mukhopadhyay, "Reconfigurable SRAM architecture with spatial voltage scaling for low power mobile multimedia applications," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 1, pp. 161–165, Jan 2011.

[69] I. J. Chang, D. Mohapatra, and K. Roy, "A priority-based 6t/8t hybrid SRAM architecture for aggressive voltage scaling in video applications," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 2, pp. 101–112, Feb 2011.

[70] J. Kwon, I. J. Chang, I. Lee, H. Park, and J. Park, "Heterogeneous SRAM cell sizing for low-power H.264 applications," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 10, pp. 2275–2284, Oct 2012.

[71] F. Frustaci, M. Khayatzadeh, D. Blaauw, D. Sylvester, and M. Alioto, "13.8 a 32kb SRAM for error-free and error-tolerant applications with dynamic energy-quality management in 28nm CMOS," in *Proc. IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, Feb 2014, pp. 244–245.

[72] Y. Dou, S. Yu, C. Gu, M. O'Neill, C. Wang, and W. Liu, "Security analysis of hardware Trojans on approximate circuits," in *Proc. the 2020 on Great Lakes Symposium on VLSI*, ser. GLSVLSI '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 315–320. [Online]. Available: https://doi.org/10.1145/3386263.3407591

[73] A. Sampson, J. Nelson, K. Strauss, and L. Ceze, "Approximate storage in solid-state memories," in *Proc. 46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2013, pp. 25–36.

[74] C. Peikert, "An efficient and parallel gaussian sampler for lattices," in *Proc. Advances in Cryptology (CRYPTO)*, T. Rabin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 80–97.

[75] L. Groot Bruinderink, A. Hülsing, T. Lange, and Y. Yarom, "Flush, gauss, and reload – a cache attack on the BLISS lattice-based signature scheme," in *Proc. Cryptographic Hardware and Embedded Systems*

(CHES), B. Gierlichs and A. Y. Poschmann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 323–345.
[76] P. Pessl, L. G. Bruinderink, and Y. Yarom, "To BLISS-B or not to be: Attacking strongswan's implementation of post-quantum signatures," in *Proc. ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017, pp. 1843–1855.
[77] J. Von Neumann, "Various techniques used in connection with random digits," *NBS Applied Mathematics Series*, vol. 12, 1961.
[78] T. Pöppelmann and T. Güneysu, "Towards practical lattice-based public-key encryption on reconfigurable hardware," in *Proc. Selected Areas in Cryptography (SAC)*, T. Lange, K. Lauter, and P. Lisoněk, Eds. Springer Berlin Heidelberg, 2014, pp. 68–85.
[79] S. Bian, M. Hiromoto, and T. Sato, "DWE: Decrypting learning with errors with errors," in *Proc. 55th Annual Design Automation Conference (DAC)*, 2018, pp. 3:1–3:6.
[80] S. Fan, W. Liu, J. Howe, A. Khalid, and M. O'Neill, "Lightweight hardware implementation of R-LWE lattice-based cryptography," in *Proc. IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2018, pp. 403–406.
[81] D.-E.-S. Kundi, S. Bian, A. Khalid, C. Wang, M. O'Neill, and W. Liu, "AxMM: Area and power efficient approximate modulo multiplier for R-LWE cryptosystem," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2020.
[82] W. Liu, S. Fan, A. Khalid, C. Rafferty, and M. O'Neill, "Optimized schoolbook polynomial multiplication for compact lattice-based cryptography on FPGA," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 10, pp. 2459–2463, Oct 2019.
[83] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. International Conference on the Theory and Application of Cryptology and Information Security*, 2017, pp. 409–437.
[84] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "Bootstrapping for approximate homomorphic encryption," in *Proc. Advances in Cryptology (EUROCRYPT)*, J. B. Nielsen and V. Rijmen, Eds. Springer International Publishing, 2018, pp. 360–384.
[85] S. Bian, M. Hiromoto, and T. Sato, "Darl: Dynamic parameter adjustment for lwe-based secure inference," in *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 1739–1744.
[86] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2014.
[87] S. Dutt, B. Paul, A. Chauhan, S. Nandi, and G. Trivedi, "ApproxHash: delay, power and area optimized approximate hash functions for cryptography applications," in *Proc. 10th International Conference on Security of Information and Networks*, 2017, pp. 291–294.
[88] M. Vilim, H. Duwe, and R. Kumar, "Approximate bitcoin mining," in *Proceedings of the 53rd Annual Design Automation Conference (DAC)*, 2016, pp. 97:1–97:6.
[89] M. Rostami, F. Koushanfar, and R. Karri, "A primer on hardware security: Models, methods, and metrics," *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1283–1295, Aug 2014.
[90] Y. Wang, Q. Xu, G. Qu, and J. Dong, "Information hiding behind approximate computation," in *Proc. Great Lakes Symposium on VLSI (GLSVLSI)*, 2019, pp. 405–410.
[91] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *Proc. 49th Annual Design Automation Conference (DAC)*, 2012, pp. 820–825.
[92] L. Lerman, G. Bontempi, and O. Markowitch, "A machine learning approach against a masked AES," *Journal of Cryptographic Engineering*, vol. 5, no. 2, pp. 123–139, 2015.
[93] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," in *Proc. International Conference on Security, Privacy, and Applied Cryptography Engineering*, 2016, pp. 3–26.
[94] A. Heuser and M. Zohner, "Intelligent machine homicide," in *Proc. International Workshop on Constructive Side-Channel Analysis and Secure Design*, 2012, pp. 249–264.
[95] G. Hospodar, B. Gierlichs, E. De Mulder, I. Verbauwhede, and J. Vandewalle, "Machine learning in side-channel analysis: a first study," *Journal of Cryptographic Engineering*, vol. 1, no. 4, p. 293, 2011.
[96] R. Gilmore, N. Hanley, and M. O'Neill, "Neural network based attack on a masked implementation of AES," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2015, pp. 106–111.
[97] F. Ganji, S. Tajik, and J.-P. Seifert, "PAC learning of arbiter PUFs," in *Journal of Cryptographic Engineering*, vol. 6, no. 3, 2016, pp. 249–258.
[98] P. Subramanyan, S. Ray, and S. Malik, "Evaluating the security of logic encryption algorithms," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, May 2015, pp. 137–143.
[99] Y. Xie and A. Srivastava, "Anti-SAT: Mitigating SAT attack on logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 2, pp. 199–207, Feb 2019.
[100] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: approximately deobfuscating integrated circuits."
[101] K. Shamsi, T. Meade, M. Li, D. Z. Pan, and Y. Jin, "On the approximation resiliency of logic locking and IC camouflaging schemes," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 347–359, Feb 2019.
[102] K. Shamsi, D. Z. Pan, and Y. Jin, "On the impossibility of approximation-resilient circuit locking," in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2019, pp. 161–170.
[103] S. McCarthy, N. Smyth, and E. O'Sullivan, "A practical implementation of identity-based encryption over ntru lattices," in *Proc. IMA International Conference on Cryptography and Coding*. Springer, 2017, pp. 227–246.
[104] L. Liu, S. Wang, B. He, and D. Zhang, "A keyword-searchable abe scheme from lattice in cloud storage environment," *IEEE Access*, vol. 7, pp. 109 038–109 053, 2019.
[105] F. Tehranipoor, N. Karimian, W. Yan, and J. A. Chandy, "DRAM-based intrinsic physically unclonable functions for system-level security and authentication," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 3, pp. 1085–1097, March 2017.
[106] A. Maiti and P. Schaumont, "A novel microprocessor-intrinsic physical unclonable function," in *Proc. 22nd International Conference on Field Programmable Logic and Applications (FPL)*, 2012, pp. 380–387.
[107] K. Hasegawa, M. Yanagisawa, and N. Togawa, "Hardware trojans classification for gate-level netlists using multi-layer neural networks," in *Proc. IEEE 23rd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2017, pp. 227–232.

Weiqiang Liu (M'12-SM'15) received the B.Sc. degree in Information Engineering from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China and the Ph.D. degree in Electronic Engineering from the Queen's University Belfast (QUB), Belfast, UK, in 2006 and 2012, respectively. In Dec. 2013, he joined the College of Electronic and Information Engineering, NUAA, where he is currently a Professor and the Vice Dean of the college. He has published one research book by Artech House and over 100 leading journal and conference papers. His paper was selected as the Feature Paper of IEEE TC in the 2017 December issue. He has two Best Paper Candidates in IEEE ISCAS 2011 and ACM GLSVLSI 2015. He has been awarded the prestigious Outstanding Young Scholar Award by National Natural Science Foundation of China in 2020. He serves as the Associate Editors for IEEE Transactions on Circuits and System I: Regular Papers (2020.1-2021.12), IEEE Transactions on Emerging Topics in Computing (2019.5-2021.4) and IEEE Transactions on Computers (2015.5-2019.4), an Steering Committee Member of IEEE Transactions on Multi-Scale Computing Systems (2018.1-2019.12). He is the program co-chair of IEEE ARITH 2020, and also technical program committee members for ARITH, DATE, ASAP, ISCAS, ASP-DAC, ISVLSI, GLSVLSI, SiPS, NANOARCH, AICAS and ICONIP. He is a member of CASCOM and VSA Technical Committee of IEEE Circuits and Systems Society. His research interests include approximate computing, hardware security and VLSI design for digital signal processing and cryptography.

Chongyan Gu (S'14–M'16) received the Ph.D. degree from Queen's University Belfast, Belfast, U.K., in 2016. She is currently a Lecturer (Assistant Professor) in the School of EEECS at Queen's University Belfast, and a member of Center for Secure Information Technologies (CSIT) with in the Institute of Electronics Communications and Information Technologies (ECIT). Dr. Gu is an expert in hardware security. Her research into physical unclonable function (PUF) has been utilised as part of a security architecture for electronic vehicle (EV) charging systems, licensed by LG-CNS, South Korea, and was also licensed for evaluation by Thales, U.K.. Her team was the overall winner of INVENT 2015, a competition to accelerate the commercialisation of innovative ideas. She has co-authored two research book chapters on the topics of "Lightweight Cryptographic Identity Solutions for the Internet of Things" published by IET in 2016 and "Approximate Computing and Its Application to Hardware Security" published by IET in 2016 and Springer in 2018, respectively. She has successfully organised two special session conferences (IEEE APCCAS in 2018 and ACM GLSVLSI in 2020). She was invited to give tutorial/talks to international conferences, such as, IEEE ASP-DAC 2020 on the topic of practical PUF design on FPGA. Her current research interests include physical unclonable functions (PUFs), security in/for approximate computing, true random number generator (TRNGs), hardware Trojan detection and machine learning attacks.

Máire O'Neill (M'03-SM'11) is Regius Professor of Electronics and Computer Engineering at Queen's University Belfast. She is Director of the Institute of Electronics Communications and Information Technologies (ECIT) and the Centre for Secure Information Technologies (CSIT) at Queen's. She is also Director of the £5M EPSRC/NCSC-funded Research Institute in Secure Hardware and Embedded Systems (RISE: www.ukrise.org) and recently led the €3.8M EU H2020 SAFEcrypto (Secure architectures for Future Emerging Cryptography: www.safecrypto.eu) project (2014-2018). She previously held a UK EPSRC Leadership Fellowship (2008-2014) and was a former holder of a UK Royal Academy of Engineering research fellowship (2003-2008). She has received numerous awards, which include a Blavatnik Engineering and Physical Sciences medal, 2019, a Royal Academy of Engineering Silver Medal, 2014 and British Female Inventor of the Year 2007. She has authored two research books, and over 160 peer-reviewed international conference/journal publications. She is an Associate Editor for IEEE TC and IEEE TETC and is secretary of the IEEE Circuits and Systems for Communications Technical committee. She is a member of the UK AI Council. She is a Fellow of the Royal Academy of Engineering, a member of the Royal Irish Academy and a Fellow of the Irish Academy of Engineering. Her research interests include hardware cryptographic architectures, lightweight cryptography, side channel analysis, physical unclonable functions, and post-quantum cryptography.

Gang Qu received his B.S. (with honor) and M.S. in Mathematics from the Univesity of Science and Technology of China and M.S. (with honor) and Ph.D. in Computer Science from the University of California, Los Angeles. He then joined the Department of Electrical and Computer Engineering in the University of Maryland, College Park where he is currently a professor and the director for Maryland Embedded Systems and Hardware Security Lab (MeshSec) and Wireless Sensor Laboratory. He is known for his work on dynamic voltage scaling for low power, VLSI design intellectual property (IP) protection and hardware security, as well as the sensor exposure and coverage problems in wireless sensor network. His recent research activities are on trusted integrated circuit design, design IP protection, nano-scale hardware security primitives, and their applications in the Internet of Things.

Paolo Montuschi (M'90-SM'07-F'14) is a Full Professor in the Department of Control and Computer Engineering and a Member of the Board of Governors at Politecnico di Torino, Italy. His research interests include computer arithmetic and architectures, computer graphics, electronic publications. He is an IEEE Fellow, and an IEEE Computer Society (CS) Golden Core member. He is currently serving as the 2017-20 IEEE Computer Society Awards Chair, as a Member-at-Large of the Publication Services and Products Board (PSPB) (2018-20), and as the Chair of its Strategic Planning Committee (2019-20). He is serving as the 2020-21 Chair of the IEEE TAB/ARC (TAB/Awards and Recognitions Committee), as a Member of the IEEE Awards Board, as a Member (2020) of the IEEE PRAC (Periodicals Review and Advisory Committee), and as a Vice Chair of the 2020 Computer Society Fellows Committee, Previously, he served, among all, as the Editor-in-Chief of the IEEE Transactions on Computers, and as . the 2019 Acting (interim) Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computing. He is a life member of the International Academy of Sciences of Turin and of Eta Kappa Nu (the Honor Society of IEEE). In March 2017 he co-founded the fifirst HKN Student Chapter in Italy and in Europe, Chapter. Contact him at paolo.montuschi@polito.it and visit http://staff.polito.it/paolo.montuschi.

Fabrizio Lombardi received the BSc (Hons.) degree in electronic engineering degree from the University of Essex, United Kingdom, in 1977, the master's degree in microwaves and modern optics in 1978 and the diploma degree in microwave engineering in 1978 from the Microwave Research Unit at the University College London, and the PhD degree from the University of London in 1982. In 1977, he joined the Microwave Research Unit at the University College London. He is currently the holder of the International Test Conference Endowed Chair Professorship at North-eastern University, Boston. He is the founding Editor-in-Chief (EiC) of IEEE Transactions on Emerging Topics in Computing and serves as the EiC for IEEE Transactions on Nanotechnology (2015-2019) and IEEE Transactions on Computers (2007-2010). He is an elected two-term member of the Board of Governors of the IEEE Computer Society (2012-2017); he is also a member of the Executive Board of the IEEE Nanotechnology Council and the Future Directions Committee of the IEEE. He is currently the Vice President of IEEE Computer Society and IEEE Nanotechnology Council. His research interests include bioinspired and nanomanufacturing/computing, VLSI design, testing, and fault/defect tolerance of digital systems. He has extensively published in these areas and coauthored/edited seven books. He is a fellow of the IEEE.