Security for Approximate Computing, Approximate Computing for Security

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

(Invited Paper)

Abstract—Approximate computing, an advanced computational technique which returns inaccurate but acceptable results instead of exact results, has emerged as a new preferable paradigm over traditional computing architectures for energy efficient system designs. It is crucial for nanoscale integrated circuits (ICs) to achieve high speed and low power, where some intrinsic errors are acceptable, such as (deep-) machine learning, image processing, communication and other error-tolerant and cognitive applications. However, approximate computing also introduces security vulnerabilities mainly due to the uncertainty and unpredictability of intrinsic errors during approximate execution which may be indistinguishable using malicious modification of the accurate result. On the other hand, interestingly, approximate computing can also provide new approaches for security. Existing literature in approximate computing covers threat models, countermeasures, and evaluations, but lacks a framework for analysis and comparison. In this paper, we provide a classification of the state of the art in this research field, including threat models in approximate computing and promising security approaches using approximate computing.

Index Terms—Approximate computing, hardware security, cryptography

I. Introduction

IN the last decade, various advanced computing systems, including supercomputers, ubiquitous computing centers, and servers, have been developed and widely deployed. Unfortunately, Moore's law is approaching its limitation [1], and conventional computing techniques are not able to provide higher computing performance under the restriction of power consumption. Therefore, new nanoscale computing paradigms are urgently required for low power and high performance computing systems. Hence, appropriate reduction of the computational accuracy could effectively improve the performance of computing systems without sacrificing functionality and perception.

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 or redundant information is tolerable for the computation. It has attracted a significant amount of interest in academia [2], [3], [4]. It is crucial for energy efficient systems and some of approximate computing techniques has been adopted in high speed and low power nanoscale integrated circuit (IC) designs. For example, Google's deep learning (DL) chip, the tensor processing unit (TPU), achieves a significant improvement in processing performance using common approximate computing techniques, such as precision scaling [5]. IBM research has pioneered to build on-chip AI accelerators with approximate computing techniques [6]. It utilized multiple approximate computing techniques, such as precision scaling and training compression approaches, and achieved 4-200x speedup over existing methods [7].

Previous research has made efforts to efficiently improve performance with acceptable loss of accuracy [3], [8]. However, approximate computing also introduces security vulnerabilities mainly because of the uncertainty and unpredictability of intrinsic errors during approximate execution which may be indistinguishable from malicious modification of the accurate result [9]. It also pointed out that approximate computing is well-suitable for security tasks. However, if approximate computing have security vulnerabilities, applications that related to will undoubtedly be affected. Interestingly, approximate computing can also provide or even enhance security solutions. For example, approximate circuits, based on simplified circuits which can reduce area and power consumption, have been proposed for information hiding [10]. 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 resources. Approximate computing opens up both challenges and opportunities for security.

Some initial survey and tutorials have been presented. [11] presented potential security vulnerabilities that will affect the integrity and security of approximate computing systems. The security threats overviewed in [11] mainly focused on approximate circuits and storage, including approximate DRAM, phase change memory and SRAM. [9] discussed security threats of approximate computing in a perspective of hardware-related primitives, for example, side-channel analysis (SCA), reverse engineering, cloning/counterfeiting and active attacks. [12] reviewed approximate computing based hardware security applications and also proposed some future research directions. These papers provide an initial introduction and discussion for this emerging field.

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 Maire 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 Electrical and Computer Engineering (ECE), University of Maryland, College Park, 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, email: lombardi@ece.neu.edu.

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

However, the existing literature lacks of a comprehensive and systematized analysis/comparison of threat models, countermeasures, and evaluations. We first provide a classification of significant contemporary challenges, including threat models in approximate circuits, hardware security circuits and approximate storage. We classify hardware security threats, including power leakage, reverse engineering, hardware Trojans, SCA, in approximate systems, such as approximate storage and approximate circuits. We also classify the application of approximate computing on building security primitives, such as approximate computing for cryptography, hardware security, approximate algorithms and biometric systems.

The rest of this paper is organized as follows. Section II provides background to approximate computing, including approximate computing strategies and techniques. Section IV presents a systematization of the security threats in approximate computing, which includes approximate circuits and approximate storage. A classification of approximate computing for security is discussed in Section V, which describes how to use approximate computing for security, such as cryptography and hardware security. Section VI describes future research directions. Conclusions are drawn in Section VII.

II. Approximate Computing

Approximate computing [8], [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 of approximate computing, including the relationship of performance, power and accuracy of an approximate computing design will be introduced. Depending on approximate level and behaviour determinism, approximate computing can be classified into three different categories [8], [13].

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 a system, there are many parameters, for example, delay, execution time and complexity, to affect performance, power and area consumption. The 2 dimension (2D) design space shows the relation ship between performance and power/area consumption. The 3 dimension (3D) design space presents the relationship between performance, computation accuracy and power/area consumption, which has one more dimension, computation accuracy, than the 2D accurate computing. The more accurate the computation, the slower the performance and the higher the power and area consumption. The more errors introduced the computation, the faster the performance and the lower the power and area consumption. This is a tradeoff needed to be considered when designing a system involving approximate computing. To achieve good performance and consume less power and area, the introduced errors should be also acceptable.

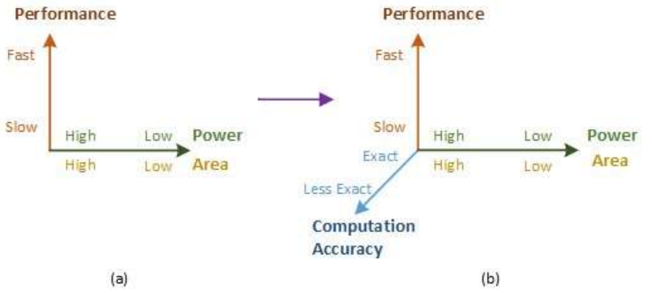


Figure 1: A design space (a) of performance and power for accurate computing (2D) and (b) of performance, power and accuracy for approximate computing (3D).

B. Classification

1) Approximate Level: 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 as shown in Fig. 2 is summarized as follows.

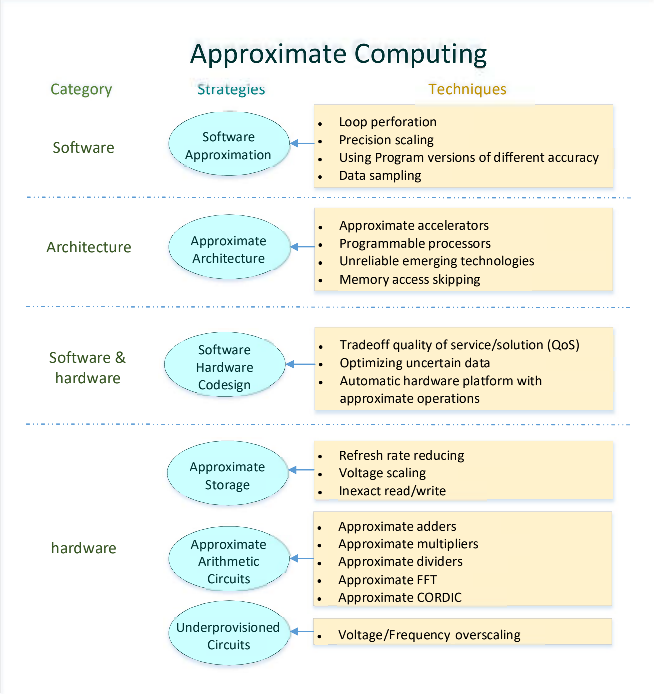


Figure 2: Approximate computing strategies and techniques.

- *Software Approximation*: Power consumption is reduced using simplified functions or data in programs. For example, loop perforation [14], precision scaling [15], [16],

[17], using program versions of different accuracy [18], and data sampling [19],

- *Approximate Architectures*: Approximate errors can be detected or optimized in approximate accelerators [20] or programmable processors [21]. Other techniques include memory access skipping [22], lossy compression [23], [24], and unreliable emerging technologies [25].
- *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 [26], voltage scaling [27] and inexact read/write [28].
- *Software/Hardware Codesign*: Most approximate research is mainly focused on a single layer. Software and hardware coordinated designs have also been presented to achieve efficient, high performance and dedicated outputs using approximate approaches. For example, a technique using trade-off quality of service or solution (QoS) was presented in [14]. [29] presented an approximate technique which optimizes uncertain data to achieve better performance. An automatic hardware platform with approximate operations was demonstrated in [30]. The incremental network approximation (INA) method has also been proposed to cooperate approximate circuits with deep neural networks (DNNs) algorithm with little loss of accuracy [31].
- *Approximate Arithmetic Circuits*: simplify circuit designs to achieve an approximate operation of the desired function, such as addition, multiplication and division. The main approximate arithmetic units include approximate adder [32], [33], approximate multipliers [34], [35], [36] and approximate dividers [37] that have been proposed. Other approximate circuits have approximate fast fourier transform (FFT) [38] and approximate CORDIC [39].
- *Underprovisioned Circuits*: Circuits, adjusted to operate at extreme conditions, such as power boundaries, which can easily trigger errors, can achieve lower power consumption. Relevant techniques include voltage overscaling [40] and frequency overscaling [41].

2) Deterministic and Non-deterministic: The classification of deterministic and non-deterministic for approximate computing depends on the output of the approximated design [42]. A deterministic design repeatedly returns the same output when given the same input as shown in Fig. 3(a). In contrast, Fig. 3(b) presents a non-deterministic design which has a rarely repeated output 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. To ensure that the errors, E_i,E_j,E_k, are acceptable for the underlying system, an error threshold θ is necessary to be utilised for evaluation. However, it is not necessary for a deterministic approximate design. Therefore, non-deterministic approximate designs have limited reproducibility.

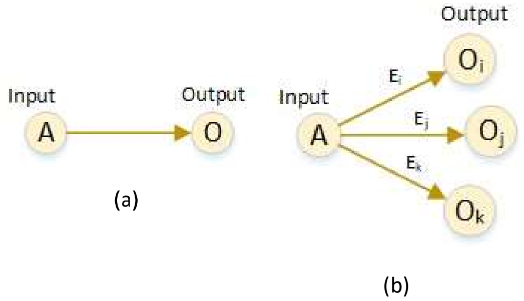


Figure 3: Approximate computing classification based on reproducibility: (a) non-deterministic, (b) deterministic.

The examples of deterministic approximate computing techniques in the above mentioned publications are [14], [16], [18], [15], [17], [19], [23], [24], [29], [30], [32], [33], [34], [35], [36], [34], [35], [36]. The non-deterministic approximate computing techniques of the above mentioned research include [20], [21], [25], [26], [27], [28], [40], [41].

In principle, a system which includes 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 been neglected and it is difficult to guarantee the security of operations that are approximated. Adversaries can target some components of an approximate computing system, for example, software programs, processors, accelerators, memories and circuits. The expected cost will go higher when the approximation level goes to architecture and hardware circuit designs since system developers, engineers, and circuit designers all may be involved. When security vulnerabilities exist in these approximate designs/systems, the test, detection and modification process must be more complicated than software debugging. According to the deterministic and nondeterministic approximate methods, attacking techniques for both should be different. In the subsequent sections, the vulnerabilities, attacking techniques and potential countermeasures for approximate computation will be discussed.

III. Security and Cryptographic Primitives

In this section, we will introduce some widely known security and attacking techniques which may affect the security of approximate computing designs. A summary of these concepts is shown in Table I.

A. Hardware Security

1) Side Channel Analysis (SCA): SCA reveals the implementation of security/cryptographic schemes by observing the behaviour of the operation to obtain additional information. During the operation, the adversary observes the power consumption of hardware implementation to calculate the cipher key or reveal details of the execution/data in the scheme. SCA can be classified into two groups: one is *invasive*

Table I: List of Frequently Used Hardware Security & Cryptographic Primitives

Category	Concept	Description
Hardware Security	Side channel analysis (SCA)	The adversary observes the power consumption of hardware during the computation.
	Reverse engineering	The adversary deconstructs an IC to reveal the design, architecture or extract knowledge from the hardware circuit.
	Hardware Trojan (HT)	A malicious alteration to the original design of an IC during design or fabrication.
	Physical unclonable function (PUF)	A circuit that uses manufacturing process variations to generate a unique unclonable digital fingerprint.
	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.
Cryptography	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.
	Homomorphic encryption (HE)	A cryptographic scheme allows arbitrary arithmetic function on encrypted data without the need of decryption.
	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.

and *noninvasive* and the other one is *passive* and *active*. Invasive SCA requires to depackage/break the device before the behaviour observation. In contrast, non-invasive SCA doesn't need to open the device during the attack. Obviously, the invasive SCA involves other pre-processing requirements and probably not applicable for the chip not acceptable to be opened. Passive SCA only observes the behaviour of the device's implementation. Active SCA can deliberately manipulate the inputs of the device, for example, carrying out fault injections, at the same time observing the behaviour. SCA has been discussed to be potentially harmful to approximate circuits, which will be introduced in details in the next section.

2) Reverse Engineering: In semiconductor industry, the technical information and patent-related information of a product are the most valuable and essential components for manufacturing companies. However, an adversary can deconstruct an IC to reveal the design, architecture or extract knowledge from a hardware circuit [43]. This process is commonly named as reverse engineering.

3) Hardware Trojan (HT): Resulting from the globalization of the semiconductor supply chain, the design and fabrication of ICs are now distributed worldwide. It brings great benefit to IC companies, leading to a lower design cost and a shorter time-to-market window [44]. However, it also raises serious concern about IC trustworthiness triggered by the use of thirdparty vendors. As a result, it is becoming very difficult to ensure the integrity and authenticity of devices. A hardware trojan (HT) can be inserted into IC products at any untrusted phase of the IC production chain by third-party vendors or adversaries with an ulterior motive [45].

4) Physical Unclonable Function (PUF): A PUF is a security primitive which utilizes the inherent process variations present during manufacturing in order to generate a unique digital fingerprint that is intrinsic to the device itself. As this natural variation between silicon dies is out of the manufacturer's control, they are inherently difficult to clone, as well as providing additional tamper-evident properties [46], [47], [48], [49]. PUF architectures can be broadly classified into Weak PUF and Strong PUF (SPUF) as discussed in [50]. SPUFs have a large number of possible challenge response pairs (CRPs), whereby a large number of random challenges will return a random response unique to each challenge, as well as the physical device. By design, this implies the requirement for a much larger entropy pool such that related challenges should not lead to related 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 Strategies (ES) based ML methods have been widely utilized to attack PUFs [51], [52], [53].

5) Logic Obfuscation: Logic obfuscation involves hiding important information, for example, functionality and implementation, related to a circuit design by inserting additional logic components into the original design so that reverse engineering will not work without authorization. In order to execute its valid functionality to generate correct outputs, a secret key is input into the logic obfuscated circuit. If a wrong key is applied, the functionality will be incorrect and wrong outputs are generated by the obfuscated circuit. Logic obfuscation techniques have been utilized to protect intellectual protection (IP) and evaluate the trust of hardware [54].

B. Cryptography

1) Post-Quantum Cryptography (PQC): In the near future, quantum computers will 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 [55], [56].

2) 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. Except conventional encryption and signatures, LBC can be flexibly applied to other constructions, such as identity based encryption and attribute based encryption and fully homomorphic encryption.

3) 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 [57].

IV. Security Threats in Approximate Computing

In this section, we will introduce and discuss the security threats, including both existing confirmed and potential attack models, in approximate computing. We will emphasize the operation and effectiveness of these threats to approximate computing systems. A comprehensive classification of security threats in approximate computing is shown in Fig. 4. In Fig. 4, the labeled *confirmed attach technique* represents the attacking method used to break the target approximate object. The *potential attack technique* refers to the attacking approach which has been stated to be potentially harmful to some approximate applications but no concrete results yet. The *confirmed affected application* is the approximate application presented to be vulnerable to specific attack techniques, which may be also effective to the *potential affected application*. It reminds that designers should be also aware of both the *confirmed attach techniques* and *potential attach techniques* for the *potential affected applications*. The affected applications are categorized into two groups, approximate circuits and approximate storage. The affected applications and relevant attack techniques will be discussed in details in the following subsections.

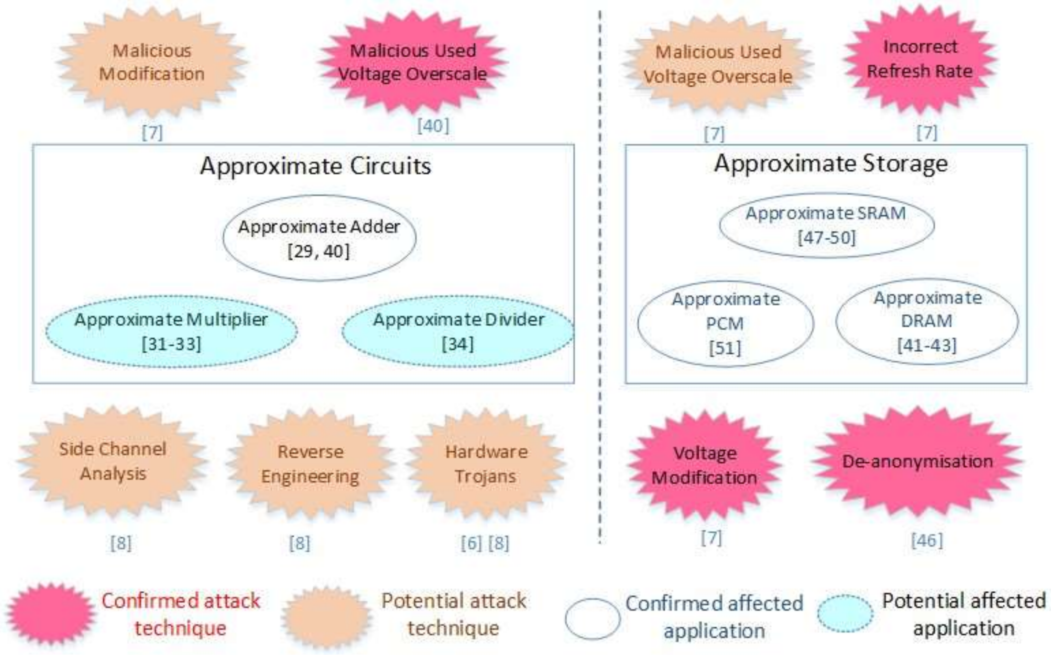


Figure 4: Classification of security threats in approximate computing.

A. Approximate Circuits

Computing arithmetic units including adders, multipliers and dividers are essential for processors, and significantly affect the performance and power consumption of the whole computing system. For cognitive applications, such as recognition, data analysis and computer vision, the aim is to achieve higher speed and lower power consumption as well as satisfied error tolerance. This has motivated the fast development of approximate arithmetic circuit designs. Most of the approximate computing circuits proposed to date are based on logic reduction and pruning methods. In cognitive computing applications, for example, image recognition, machine learning and pattern recognition, the key arithmetic units include adders and multipliers. Therefore, high performance and low power adders and multipliers have been extensively studied. However, these may be vulnerable to security threats.

1) Malicious Modification of Inputs or Registers: [11] introduced a potential attack, *i.e.* malicious modification, of an approximate adder by deliberately manipulating the adder's inputs to continuously generate erroneous outputs to activate error correction code (ECC) or fault tolerant process more than usual. It has been shown that the correlation between the output and power for the adder with 50% errors is higher than that with 25% errors.

Fig. 5 presents a potential malicious modification of an ALM by deliberately tampering with the truncation parameter t, which is normally stored in a memory's register on board. A truncation parameter, read out from the register, can be maliciously manipulated to provide an unexpected value. As an example, the original picture as shown in Fig. 5(a) represents the exact result of the ALM calculation with both 8-bit input and output. Fig. 5(b) to Fig. 5(e) illustrate the results generated by different malicious modifications of the truncation parameter t (t = 6,4,3 and 2, respectively). For example, an attacker may deliberately manipulate to change t directly to 2, producing an unacceptable image. There is no need to modify the value of inputs. However, an adversary can hack/change the value of the truncation parameter in register. Finally, the large number of erroneous outputs will also activate ECC or fault tolerant process more than usual. It will also increase the power consumption since obviously more schemes are frequently activated.

2) Hardware Trojan: [9] and [11] discussed the potential security threats introduced by hardware Trojans. Approximate devices might require extra hardware components to control the level of approximation, which provides opportunities for hardware Trojan insertion.

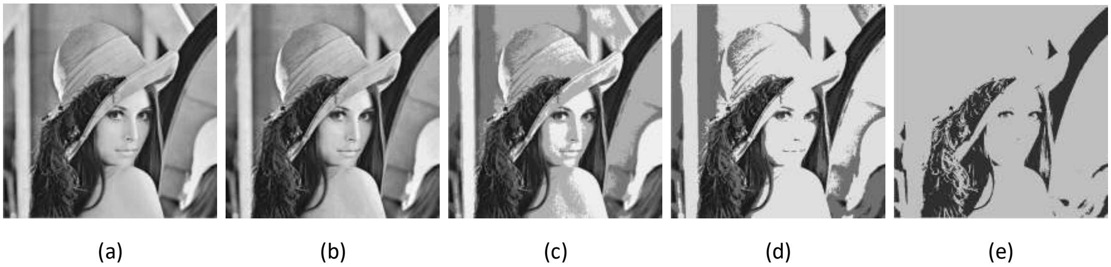


Figure 5: Potential malicious modifications on the truncation parameter (t) of approximate logarithmic multiplier (approximate logarithmic multiplier (ALM)): (a) original result with 8-bit input, (b) t = 6, (c) t = 4, (d) t = 3 and (e) t = 2.

Moreover, the approximate circuits also introduce possibilities for hardware Trojan insertion compared to exact circuits. At the design stage of IC, the transition probability of a circuit is a key feature for HT insertion by an adversary. Normally, a HT is inserted into a circuit with low transition probability since it is easy to hide the HT circuits. The transition probability distributions for both an exact 8-bit adder and an approximate 8-bit adder are presented in Fig. 6(a) and Fig. 6(b), respectively. The transition probability distribution of the exact adder is distributed close to the high transition probability area, which is near to the highest probability value 0.25. There is no transition probability smaller than 0.20. However, for the approximate adder, the transition probability presents a random distribution in the range [0.05, 0.25]. With a spread of low transition probability, this means that HTs have a higher possibility of being added into approximate adders than the exact adders and hence, approximate circuits are more vulnerable to such hardware Trojan attacks. This is an initial result for a specific approximate adder and used as an example. It is interesting to investigate the security of other approximate circuits, for example, other approximate adders, multipliers and dividers, which will be further discussed in the future work section.

3) Voltage Scaling and Reverse Engineering: [9] discussed that approximate circuits may leak information at some operating points using voltage scaling techniques. [58] utilized voltage over-scaling based approximate computing method to result in different clock period which may lead to the delay difference of a signal propagation in a circuit. Due to the process variation, different chips will output different delays for the critical path. This is similar to the principle of a PUF in hardware security. Hence, the erroneous outputs can utilize as an identity or possibly leak privacy information. Voltage scaling techniques have also been utilized for approximate storage which will be discussed in Section IV-B.

Reverse-engineering can also affect approximate circuits as analyzed in [9] since it is difficult to apply approximate circuits to implement cryptographic algorithms, which can then be differentiated from other blocks implemented as approximate circuits. Moreover, applying reverse-engineering techniques to reveal and reconstruct an approximate circuit is easier than for an original exact circuit.

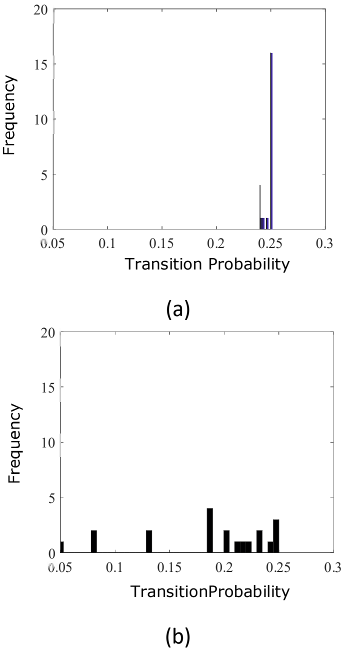


Figure 6: The transition probability distributions for (a) 8-bit exact adder and (b) 8-bit approximate adder.

B. Approximate Storage

Storage is another important aspect in approximate computing. Memory access is extensive in many error-tolerant and cognitive applications including machine learning, computer vision, graphics, *etc.* The error resilience ability enables these applications to produce acceptable results even if inputs are noisy or erroneous. This has led to the rise of designing approximate memories/storage to achieve large savings in power consumption.

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 and leaks charge, which results in a significant power waste. Approximate computing provides many possibilities for substantial energy savings. [59] first presented an approach which splits an application into critical and non-critical parts and allocates this data separate parts of memory. Different refresh rates are utilized for both parts to save energy for the non-critical data. [60] proposed a hardware based approximating method to refresh the most important bits (MSBs) of operands at a higher refresh rate and alternatively the least important bits (LSBs) of operands at a lower rate. Software-based approaches have also been proposed. For example, [61] proposed a method based on software modification and DRAM controller changes to improve energy quality and can be applied to commercial off-the-shelf (COTS) devices. DRAM has also been developed for hardware security, for example, DRAM PUF for identification and authentication [62] and for true random number generation [63].

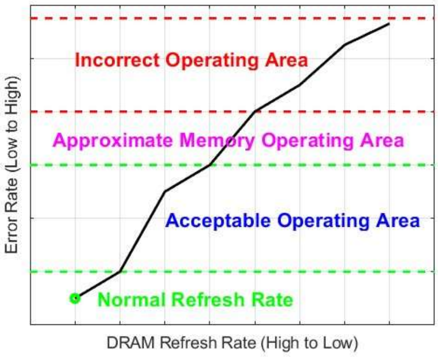


Figure 7: The relationship between DRAM refresh rate and error rate leads to different operating conditions.

Fig. 7 presents the relationship between DRAM refresh rate and error rate which leads to different operating conditions. The lower the refresh rate for DRAM, the higher the bit error rate. Normal refresh rates result in the lowest error rate but higher power consumption. 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 usually operate at the border of the acceptable operating area. If the refresh counter is manipulated by an adversary, the precise DRAM could be refreshed at an incorrect rate. Then, the data stored in the precise DRAM would be approximated and generate unacceptable errors.

[11] demonstrated another example of tampering the memory counter of DRAM to initiate DRAM with an incorrect refresh frequency. To save energy consumption, a memory composed of precise and approximate DRAM cells can be refreshed in different refresh rates. The refresh commands are generated under a control logic unit inside the DRAM module. A counter in the control unit is utilized to calculate the address of the next refresh event. An adversary may only need to manipulate the configuration signal to damage the stored data in the precise DRAM. As shown in Fig. 8, when the configuration signal is deliberately modified to go high one clock in advance, the counter for the number of MSBs calculation starts earlier to increase. Therefore, it will deactivate the DRAM refresh enable signal one clock earlier than normal. The precise DRAM is refreshed in an abnormal condition, which may result in key data lost or unexpected errors.

DRAM PUF based on the decay characteristics of DRAM cells was proposed to provide a lightweight security approach to devices for key generation or authentication. [64] presented how to characterize a DRAM PUF by exploring the decay characteristics of DRAM cells in approximate DRAM since it could not achieve 100% reliability of stored data due to unique errors that can be used as an identification/fingerprint. However, attackers can use the memory fingerprint to identify other approximate outputs from the same system since similar patterns of errors in results coming from the same chip. Hence, future hardware based approximate designs should be aware of design anonymity and avoid to expose privacy sensitive data.

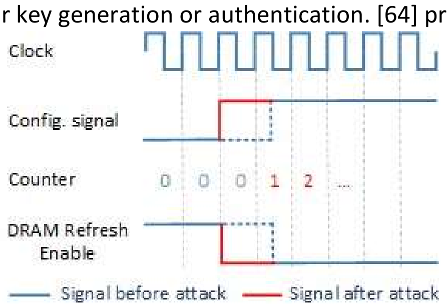


Figure 8: Timing diagram of DRAM with incorrect refresh frequency due to tampered memory counter [11].

2) Approximate SRAM: Supply voltage scaling, which can reduce the power consumption of each memory access, is a preferable technique for SRAM array for image processing and multimedia applications although it leads to a higher bit error rate. [65] proposed a dynamically reconfigurable SRAM array which implements a lower voltage for cells storing the LSBs and a nominal voltage for cells storing the MSBs. The error rates can be modified in run-time by reconfiguring the number of bits in the lower voltage mode. [66] also proposed a voltage scalable architecture to save power dissipation by storing different "quality" data in various "quantity" of SRAM bit-cells. The principle is to save the most sensitive data in video applications in higher order 8T bit-cells while the lower order bits are stored in 6T bit-cells. The supply voltage in the 8T SRAM memory remains normal to ensure the output video is of sufficient quality. The supply voltage scaling technique can be utilized for the less important lower order bits, stored in 6T bit-cells, since errors/failures are acceptable in some applications, for example, video processing. The research in [67] improved the voltage scaling idea by proposing an efficient sizing algorithm to reduce the computation time needed to select the SRAM bit-cell sizes. Such techniques still have drawbacks. Since the bit error rate (BER) in the MSBs is fixed at design time, it is impossible to derive a dynamic energyquality trade-off at run time, which has been achieved by [68] and [27].

However, approximate SRAMs may also be vulnerable to security threats. [11] discussed a potential attack for the above mentioned approximate SRAM by introducing more errors than that can be tolerant. It will overburden the ECC scheme in the memory. As an example, Fig. 9 presents attacks on approximate SRAM based on a maliciously manipulation of voltage scaling techniques. A typical supply voltage scaling technique for SRAM [65] is shown in Fig. 9(a), where a lower voltage is applied to the LSBs and a nominal voltage is executed for the MSBs. However, an adversary can manipulate and introduce errors to the MSBs through the supply voltage scaling technique, as in Fig. 9(b).

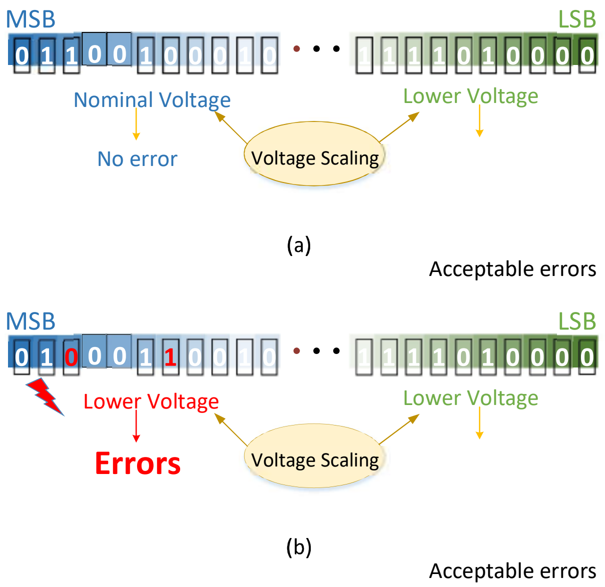


Figure 9: An example of attacks on approximate SRAM: (a) normal voltage scaling technique for SRAM to generate acceptable errors on the LSB and no errors on the MSB, (b) maliciously apply voltage scaling technique to the MSB to introduce unacceptable errors.

3) Approximate Phase-Change Memory (PCM): The inevitable "scaling limits", a limit to how small a flash or DRAM process can be shrunk, determines the number of electrons that can be stored on a memory cell and forces memory manufacturers to adopt alternative technologies, such as PCM. PCM is a type of non-volatile memory (NVM) and can be considered as a replacement for disk, flash and potentially DRAM, to solve some of their disadvantages, *i.e.* DRAM's scaling woes and vastly outpace flash solid-state drivers (SSDs). Although PCM also has some drawbacks, such as low speed, power hungry and finite lifetime, *etc.*, approximate computing techniques can address these. [69] proposed an approximate storage technique based on PCM to make efficient data storage. Although [69] achieved an improved performance of speed over precise PCM, it opens up new attack vectors for approximate PCM. [11] discussed potential security vulnerabilities along the writing flow of approximate PCM as shown in Fig. 10. The threshold T_{approx} defines the margin between accurate and approximate PCM memory blocks. If the threshold T_{approx} is altered to an incorrect value, the critical data stored in the accurate PCM memory will be affected. During the writing operation, the writing voltage v is gradually increased in each iteration. $N(u_r{,}^{\sigma}{}_{r}{}^{2})$ represents the noise function, where u_r and σ_r are the mean and standard deviation of the error effect. The writing operation may be failed when the voltage step is maliciously compromised by underestimating or overestimating the noise function. Since the number of writing iterations depends on the voltage difference. If the sensing circuit is compromised, for example, adding a voltage offset, 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 accurate PCM memory.

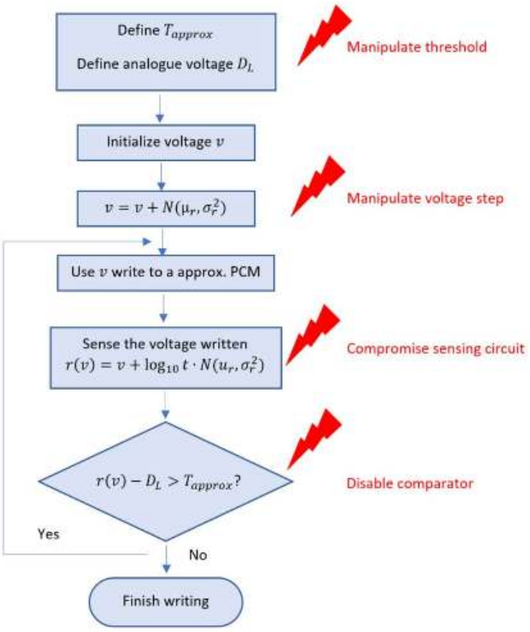

Define T_{approx}
Define analogue voltage D_L
Manipulate threshold
Initialize voltage v
$v = v + N(\mu_r, \sigma_r^2)$
Manipulate voltage step
Use v write to a approx. PCM
Sense the voltage written
$r(v) = v + \log_{10} t \cdot N(u_r, \sigma_r^2)$
Compromise sensing circuit
$r(v) - D_L > T_{approx}$?
Disable comparator
Yes
No
Finish writing

Figure 10: Flowchart of approximate PCM writing and potential security vulnerabilities [11].

V. Approximate Computing for Security

In the previous section, we discussed previous research on potential security threats for approximate computing. Actually, a dimension of approximate computing, can bring security to address some essential vulnerabilities, as discussed in this section. A comprehensive classification of security solutions for approximate computing is shown in Fig. 11. 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

1) Post Quantum Cryptography (PQC): Discrete Gaussian sampling is a critical constituent of many LBC based schemes [70]. The sampler often becomes the bottleneck of schemes requiring high performance and its implementation has been successfully attacked by SCA [71], [72].

Rejection sampling, shown in Fig. 12, is a common method employed to execute discrete Gaussian sampling in lattice based cryptography [73]. 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 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. [74] presented the performance, resource consumption and quality of six conditions of the implemented comparator-based Gaussian sampler for different tail cuts and statistical distances.

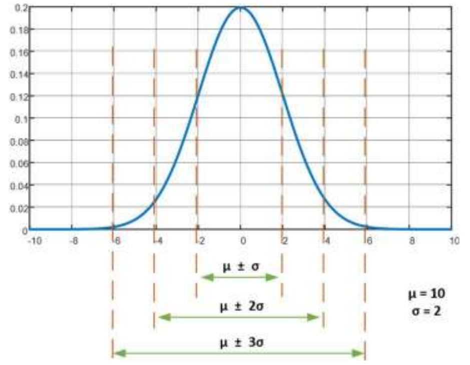


Figure 12: The tail-cut of Gaussian sampling.

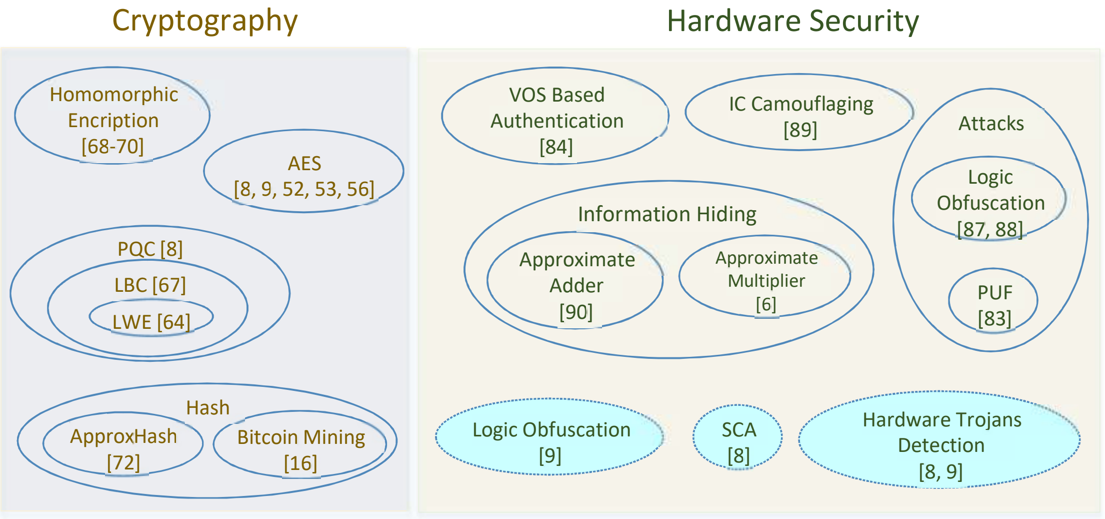
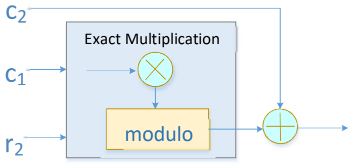
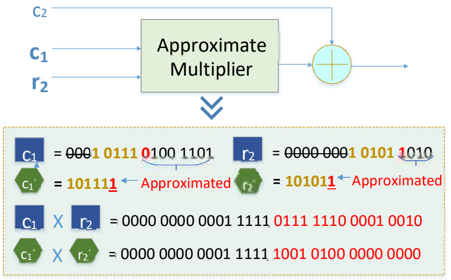


Figure 11: Classification of approximate computing for hardware security and cryptography.

Apart from the Gaussian sampling, the modular polynomial multiplication in a RLWE algorithm is the main bottleneck in the realization of a practical resource-constrained design for embedded Internet of things (IoT) devices. Exploiting the inherent approximate nature of RLWE problem, [75], [75] presented an approach utilizing approximate computing for RLWE based applications as shown in Fig. 13. Fig. 13(a) presents an accurate multiplication for the hardware architecture of RLWE decryption. An optimized dynamic range multipliers (DRUM) approximate multiplier, as shown in Fig. 13(b), has been proposed by [75] to improve the speed, reduce the area usage and power consumption for RLWE decryption hardware only.



(a)


c2
c1
r2
Approximate Multiplier
c1 = ~~000~~1 0111 0100 1101
r2 = ~~0000 000~~1 0101 1010
c1′ = 101111 ← Approximated
r2′ = 101011 ← Approximated
c1 X r2 = 0000 0000 0001 1111 0111 1110 0001 0010
c1′ X r2′ = 0000 0000 0001 1111 1001 0100 0000 0000

(b)

Figure 13: Hardware architecture of RLWE decryption, where (a) includes an exact mulitplication [76] and (b) demonstrates

an approximated multiplication using DRUM approximate multiplier.

Later on [77] proposed a design of an area/power efficient approximate modular multiplier (so called AxMM) for complete RLWE hardware, by exploiting the statistics of Gaussian noise in addition to the technique of [78]; transforming the unsigned Gaussian data to signed format. Fig. 14 presents the design of AxMM, comprising of approximate multiplier (AxMult) followed by an approximated 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 rather than applied at the end to get the required result for a negative number. Compared to the smallest exact RLWE multiplier design [78], the AxMM is able to reduce the area by over 35% and power consumption by over 23% with slight reduction in STD of Gaussian distribution as well as the security level.

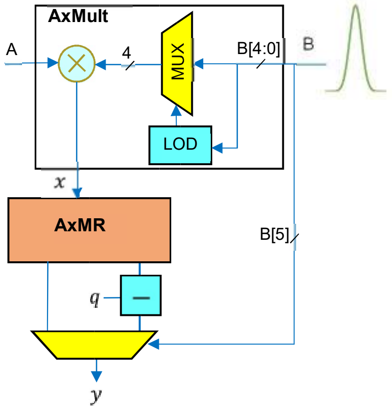
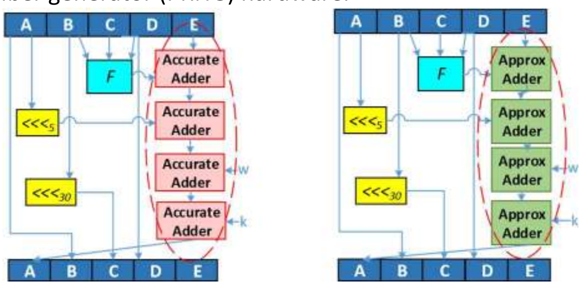


Figure 14: Approximate modular multiplier (AxMM) [77]

2) Homomorphic Encryption: [79] 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. [79] achieved linear complexity in the growth of the cipher-text modulus compared to other work with exponential complexity growth. Subsequent work by the authors [80] presented an approximate bootstrapping operation for homomorphic decryption. Also, [81] utilized the approximate computing techniques proposed in [75] 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. 15(a)), hash functions have been significantly developed and utilized in many security primitives [82]. Approximate implementations of Secure Hash Algorithm-1 (SHA-1) as shown in Fig. 15(b), have been proposed [83] to optimize the delay, power and area consumption for cryptographic applications. 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 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 ApproxHash can be utilized in error tolerant applications and pseudo random number generator (PRNG) hardware.



(a) (b)

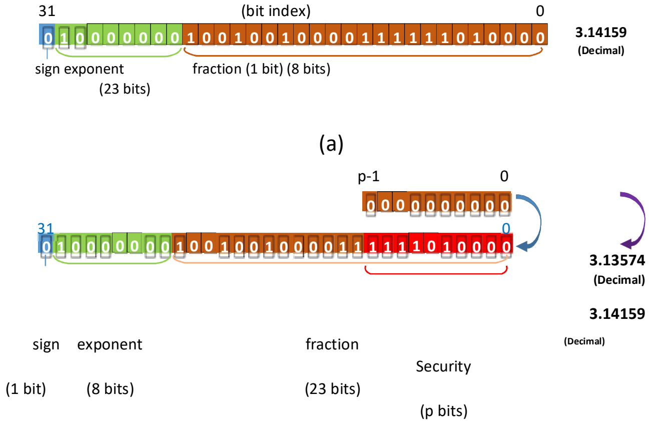
Figure 15: Approximate adders applied to Hash functions, where (a) and (b) are basic building blocks of a conventional SHA-1 algorithm using accurate adders and an approximate SHA-1 algorithm using approximate adders, respectively.

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 reduce government restrictions. Bitcoin mining is a process of creating and adding transactions to the Bitcoin ledger, called Blockchain. Bitcoin mining, based on complex computation process, is inherently error tolerant. With this property, approximate computing can be applied to Bitcoin mining as proposed by [84]. 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 [84]. 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 [84] 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 Meltdown and Spectre vulnerabilities on processors demonstrated examples of hardware based attacks. [85] 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: The ubiquitous of IoT will revolutionize our lives but also opens up new attack vectors for criminal hackers. 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 [10], based on basic arithmetic operations executed by approximate function units, enabling embedded information for authentication and other security related applications. The principle is presented in Fig. 16, 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.



(a)

(b)

Figure 16: The application of approximate computing to extract security: (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, we will show two examples of hiding information into approximate computing, one is using an approximate adder and the other one is using and approximate multiplier.

[86] presented an information hiding strategy using an approximate adder based on an accuracy configurable adder [87]. A short message M can be deliberately hidden in the operation of an approximate adder to detect incorrect results which can be used as a proof that the adder carries hidden information.

Fig. 17 shows the process and an example of applying an approximate multiplier for information hiding [10]. Two real numbers A and B can be written as $A = A^0 \oplus K_A$ and $B = B^0 \oplus K_B$ using the approximate format, where A^0 and B^0 are the numbers A and B in the approximate format with the last p bits are 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, $A^0 \times B^0 = 3.1413574 \times 12.30957 = 38.6687588$ is calculated for the approximate computation with $p = 10$. The final result with security information embedded (K_r) is computed

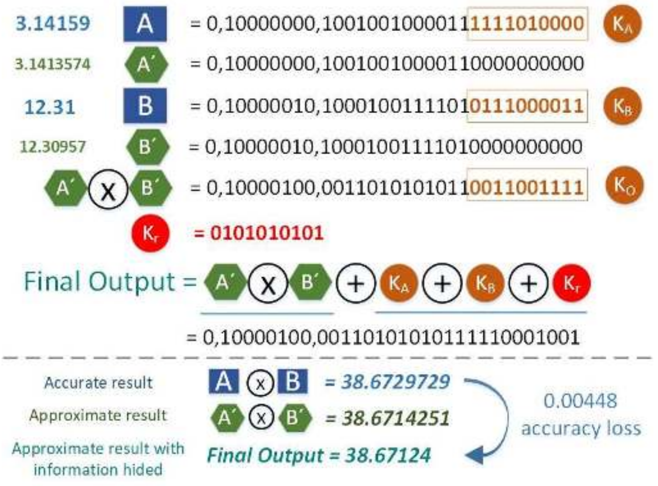


Figure 17: An example of the application of an approximate multiplier for information embedding.

as $A^0{\times}B^0{\oplus}K_A{\oplus}K_B{\oplus}K_r = 38.67124$, 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.

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 [54] to address this challenge. VOS technique commonly uses approximate computing method 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 considerably save power consumption. However, process variation is effected by scaling voltage, which can generate timing errors and thus sacrifice the output quality. Hence, a two-factor authentication scheme that uses passwords and hardware properties was proposed to achieve lightweight authentication for IoT applications [54]. [54] introduced an example of the effect of process variations in voltage overscaling based computation as shown in Fig. 18. An image processing technique, superimposition, is applied to images (a) and (b) to generate a new image (c)without voltage overscaling technique. The process is carried out using an accurate adder. When two voltage over-scaled ripple-carry adders with process variations are applied, images (d) and (e) with the error patterns (f) and (g), respectively, are received. The difference between two error patterns is shown in image (h). Hence, it can be used for digital fingerprint generation and then applied to authentication. However, it may also have the same deanonymization issues as mentioned in Section IV-B1.

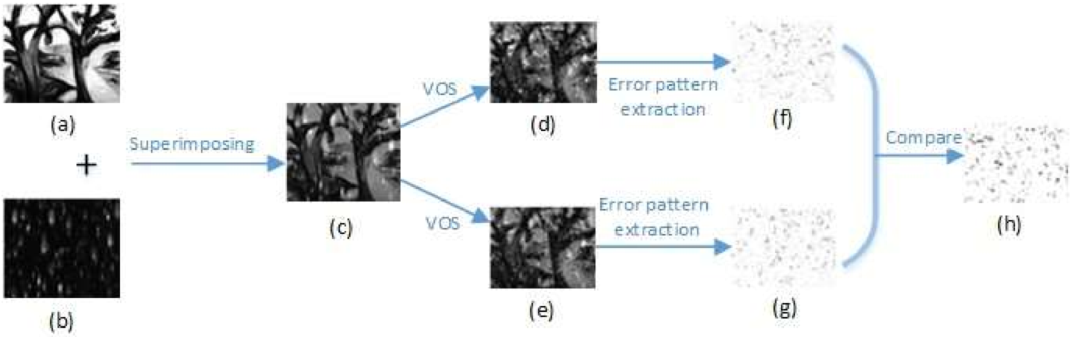


Figure 18: An example of the effect of process variations in voltage over-scaling based computation. Two images (a) trees and (b) snowflakes are superimposed to generate (c) snowfall. When the computation is under voltage over-scaling technique and two adders are identical except the process variations 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

An adversary could deanonymize the approximate circuit based on the physical variations by analyzing the error patterns of a the two error patterns (f) and (g). voltage over-scaled circuit.

C. Approximate Computing for Machine Learning Based Security

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 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 [88], [89]. Much of the research to date has centered on the use of SVMs [90], [91] and random forests [88]. Research by Lerman *et al.* [88] showed how such approaches can be used to uncover the key of a (masked) advanced encryption standard (AES) implementation, that include protection against attacks, such as power analysis.

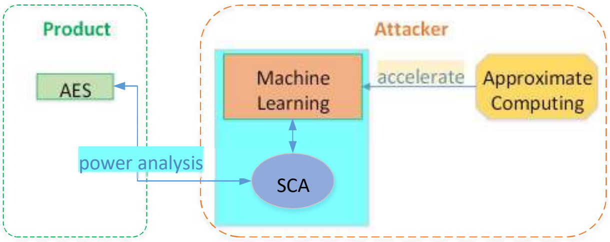


Figure 19: An example of the application of machine learning to SCA. Approximate computing can be used to accelerate the machine learning process and improve the attack efficiency.

An illustration of this idea is shown in Fig. 19. Gilmore *et al.* in [92] built on this research by investigating the novel application of a neural network (NN)-based attack (that can be accelerated by approximate computing) against a masked AES design. This two stage attack first uses a NN model to recover the mask, with a second NN model built to recover the masked secret data. Combining the knowledge recovered from both attacks allows subsequent key recovery with only a single trace. Parallel 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 [89].

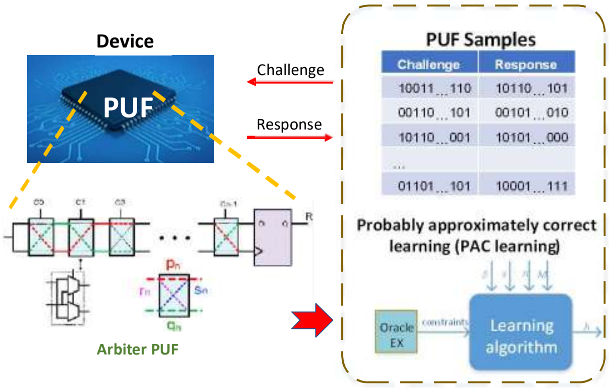
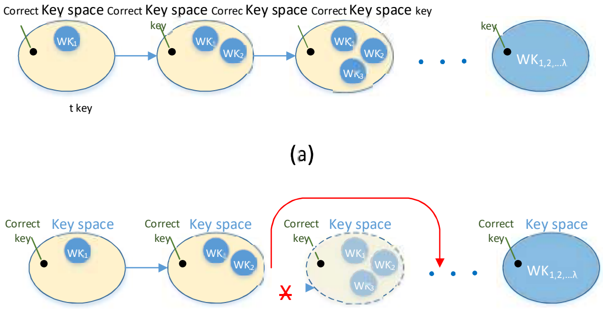


Figure 20: An example of the application of PAC to model an Arbiter PUF design [93].

2) PUF: The probably approximately correct (PAC) algorithm has been utilized to model k−XORed Arbiter PUFs (APUFs) suitable for $k < 4$ [93] as shown in Fig. 20. In order to prevent modeling 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 modeling 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 [94]. The principle of a SAT attack is presented in [95], as shown in Fig. 21(a). SAT resistant countermeasures have been proposed by exponentially increasing the minimum number of queries needed for addressing the problem. However, an exact deobfuscation accuracy is required for the countermeasure based on implicit assumptions. To address this, [96] and [97] proposed an approximate attack, AppSAT, as shown in Fig. 21(b) to deobfuscate circuits by terminating the attack at an early stage.



(a)

(b)

Figure 21: The application of approximate computing to SAT attacks on logic obfuscation: (a) illustration of the iterative SAT attack process [95], (b) an approximate deobfuscation algorithm based on SAT attacks and random testing [97].

High corruptibility, or ‘compound’ schemes, have been proposed to prevent SAT attacks. [98] proposed an approximate SAT-based attack framework to enhance the efficiency of the attack using approximate techniques to convert a compound SAT attack to a general SAT attack.

4) Hardware Trojan Detection: Deep Learning (DL) is a data driven approach, where the goal is to ensure the learning algorithm is agnostic to the problem at hand, only the data changes [99]. This type of approach is often based on NN type architectures with multiple hidden layers. With advances in training algorithms and computational power, it is now possible to train vast amounts of data leading to today’s rapid advancements and adoption. As mentioned in Section IV-A2, a HT is a type of malicious modification to circuits by an attacker. Recently, Hasegawa *et al.* [100] proposed a Trojan classification method for gate-level netlists using SVMs. By analyzing the netlists from the Trust-HUB benchmark suite [101], they identify several features strongly related to HTs. Trained by these features their SVM approach results in high true positive rates, but relatively poor true negative rates when applied to the benchmark suite. They also proposed the first use of DL in HT detection for gate-level netlists. Fig. 22 shows an approach using approximate computing to accelerate DL algorithms for HT detection. According to the effectiveness of the approximate circuit and algorithm development, the efficiency of the HT detection will be significantly improved.

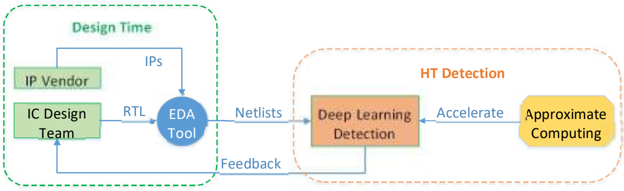


Figure 22: The application of approximate computing to accelerate the detection of HTs.

VI. Future Research Directions

The area of *security in approximate computing* and *approximate computing for security* is not mature. There are a number of open questions needed to be addressed for practical applications, which lead to the future research directions.

A. Security in Approximate Computing

1) The Impact of Error: As shown in Fig. 2, one of the biggest differences between accurate computing and approximate computing is the appearance of errors. An accurate computing design generates exact results without any errors. In contrast, an approximate computing design, acceptable to errors, may have many possible outputs. An interesting phenomenon is adversaries can also introduce errors to both accurate and approximate designs. Based on this, some challenges or open questions need to be discussed.

- For the accurate computing design, the malicious actions/errors can be easily detected. However, it is more difficult for the approximate computing design to differentiate normal errors generated by the approximate design and malicious errors deliberately manipulated by adversaries. Moreover, the impact of the security of the approximate computing is still unknown.
- A threshold value can be set for the errors within the reasonable range of approximate designs. Any error beyond the threshold can be considered as potential malicious attacks. However, the selection or setting of the threshold is also challenging.
- A normal test and approximate test have different yield and security challenges. Future test techniques for approximate computing need to consider how to effectively evaluate security vulnerabilities of an approximate system/design.
- Previously, we mentioned that approximate computing designs can be also classified as deterministic and nondeterministic approximate designs. The error patterns and approximate schemes of both the deterministic and nondeterministic approximate designs are different, which may open up various opportunities for new security attacks. Relatively, the testing techniques for both the deterministic and non-deterministic approximate designs should also be different.
- The error characteristics of approximate design and malicious circuits are also important, which has different impact on the results. Need to model, analyze and control the error.
- In Section IV-A2, the potential threats introduced by hardware Trojans has been discussed. An initial result presented that the approximate adder circuit is probably more vulnerable than the exact adder since the approximate adder has a lower transition probability distribution compared to the exact adder. It is interesting to investigate whether the low transition probability is related to the errors of approximate circuits. For example, whether the more approximate the lower transition probability? It is also worth to investigate that the impact of the types of approximates (adder, multiplier or divider) to the possibility of hardware Trojan insertion.

2) *Countermeasures for Attacks:* The final objective is to securely apply approximate computing techniques to the practical scenarios. The feasibility and effectiveness of conventional cryptographic and security approaches need a concrete evaluation when utilized to approximate computing system/designs. New countermeasures will be probably required if the conventional methods are less effective. How to design and evaluate new countermeasures for the attacks on approximate computing system/design is a new question. Since the most outstanding properties of approximate computing techniques are less energy consumption and faster speed, the countermeasures for such techniques should be also low-cost and more general.

3) *Cross-Layer Approximate Computing:* Is it necessary to perform cross-layer security analysis of approximate computing? Cross-layer is required due to the personalising and customization. Over the years, previous research on approximate computing has spanned from devices to systems. However, most of the current research has mainly focused on a single layer. For example, the research of an approximate computer arithmetic circuits is only applied to simple fault-tolerant applications with a small amount of arithmetic operations, while current approximate algorithms mostly run on precise hardware. In the case where an approximate algorithm is executed on approximate hardware, the current research has not fully considered how to make the approximate hardware and the approximate algorithm mutually compensate for and tolerate errors to realize synergy and achieve the best '3D' (precision, performance and power consumption) trade-off. Some initial work has been presented in [31]. However, security threats associated with such multi-level approximate designs have been neglected. It is essential to investigate and evaluate the security vulnerabilities in these designs and consider countermeasures for securing multi-level co-designs. For example, approximate computing is promising for dealing with DNN. An approximate multiplier is a good candidate in the hardware implementation for the construction of flexible NNs. The application of approximate multiplication for the hardware implementation of an approximate NN is illustrated in Fig. 23.

B. Approximate Computing for Security

1) *The Impact of Approximate Computing for Security:* It is necessary to know how the security of a system/design will be affected by introducing approximate computing to an existed system. In this situation, the existed system may already have a protection scheme. The introduction or replacement of approximate computing components may reduce the security level while improve the energy efficiency. It may also bring in unknown vectors for attackers. As previously mentioned, there is a tradeoff between the security and performance for approximate computing based designs. It is worth to think

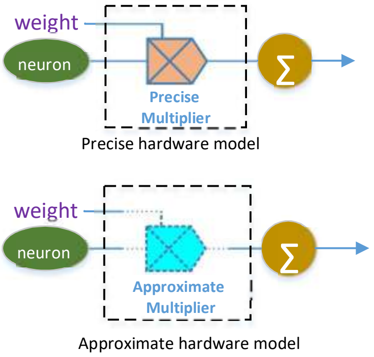


Figure 23: The application of an approximate multiplier for the hardware implementation of an approximate NN.

about the balance of both security and performance before developing an approximate design.

2) Benefits By Applying Approximate Computing: In this paper, we have discussed that approximate computing techniques have been utilized in cryptography or security, including the following aspects:

- Efficiency Improvement for cryptography: the PQC in Section V-A1, the Homomorphic encryption in Section V-A2, the approxHash in Section V-A3 and the bitcoin mining in Section V-A4.
- Acceleration for attacking techniques: the SCA attack acceleration in Section V-C1, the PAC attack on PUF in Section V-C2, the SAT attack acceleration on logic obfuscation in Section V-C3.
- Security enhancement: providing a low-cost information hiding scheme in Section V-B1, a lightweight authentication method in Section V-B2.

These raise up another concern, the impact of approximate computing for the above strategies. For example, how much efficiency the cryptography can be improved, how machine learning based SCA is affected and how much information can be hided when different degrees of approximate computing are applied. Except these, other promising research areas can be also further investigated. In the following content, we will introduce three new topics using approximate computing techniques for hardware security.

a) Approximate Circuit Based PUF: In Section IV-B1, an approximate DRAM-based PUF [64] was presented for identification/authentication. The DRAM memory-based PUF, is only one of a range of types of PUF designs. Other types include circuit based Arbiter PUFs [49] and ring oscillator (RO) PUFs [102]. Moreover, in Section V-C2, an approximate algorithm, PAC, was utilised to attack PUF designs [93]. However, there has been no research to date on approximate circuitbased PUF design, which is another promising approach.

b) Logic Obfuscation: In circuits logic obfuscation, an attacker can decipher the key by sensitizing the key values to the output or isolating the key related gates since the logic obfuscation circuit, can be removed from the original circuit [103]. To counter this, Fig. 24 shows a potential application of approximate arithmetic circuits in logic obfuscation. If the underlying design to be obfuscated is an approximate arithmetic circuit, logic obfuscation can be applied to the MSB or LSB of the circuit to ensure it only be used correctly by applying the key required for the logic obfuscation circuit. Otherwise, the computation results will be too erroneous to be used.

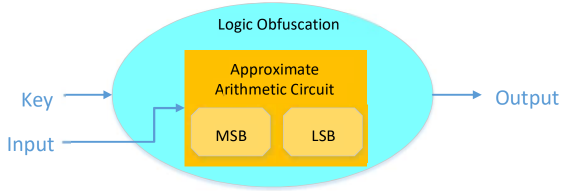


Figure 24: A potential application of an approximate arithmetic circuit for logic obfuscation.

3) Other Security Applications Adopt Approximate Computing: We use biometric security as an example to show the potential security applications can adopt approximate computing. Due to the high probability of hacking threats in traditional security strategies based on user-names and passwords, biometric approaches are often considered for security applications, such as finger printing, iris scanners and face recognition. They have been widely adopted in several scenarios: smartphones, banking, borders, and so on. There are two main reasons for the widespread application of these biometric approaches. One is that they have a large data pool and the other is that the data has high uniqueness. A certain portion of errors is tolerable for an iris encoding. For example, the possibility that two different individuals have 25% of the same hamming distance is only 1 in 13 billion. However, the data processing speed is a challenge in iris recognition systems which includes many computational components, for example, focus assessment, image segmentation, normalization and data encoding [104]. To accelerate the response time of iris scanning from image collection to iris encoding, [104] recently proposed a new method to speed up the iris recognition process using approximate computing at both the software and hardware level. Compared to a pure software implementation, the approximate computing based software and hardware codesign achieved a speedup of 378× while maintaining an acceptable accuracy.

4) Emerging Approximate Techniques: A significant growth on approximate computing technologies will revolutionize modern computing systems and platforms. It may bring in new security vulnerabilities or require security approaches for protection. For example, emerging nonvolatile memory technologies, such as resistive random access memory (RRAM) and magnetic random-access memory (MRAM), which combine the advantages of conventional memory technologies, have become very attractive for future memory hierarchies. [105] shows that RRAM-based analog approximate computing systems are very energy efficient; however, their accuracy is difficult to control and quantify. Research outputs [106], [107], [108] have presented new achievements which can be applied to memory technologies. An approximate multilevel cell (MLC) spin transfer torque-random access memory (STT-RAM) [109] has been proposed to eliminate the ECC requirement and significantly improve memory utilization with negligible image quality degradation. A scaled STT-RAM [110] has also been proposed for approximate computing to reduce power consumption and area usage. A comprehensive evaluation of using STT-RAM to replace DRAM technology is investigated in [111]. It shows that STT-RAM achieved comparable performance compared to DRAM and is a very promising memory technology. An approximate MLC STTRAM [112] has also been shown to achieve high energy efficiency without degrading the image quality requirement of applications. This research demonstrates that approximate computing has been applied to emerging memory technologies. However, the security of these approximate technologies is unknown and needs to be investigated.

VII. Conclusion

Due to a high demand for low power but high performance computing systems, approximate computing, which outperforms 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, 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 (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).

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 *2018 IEEE Symposium on VLSI Circuits*, 2018, pp. 35–36.

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

[9] 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.

[10] 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.

[11] 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.

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

[13] M. Ammar Ben Khadra, “An introduction to approximate computing,” *arXiv*, pp. arXiv–1711, 2017.

[14] 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.

[15] M. A. Anam, P. N. Whatmough, and Y. Andreopoulos, “Precisionenergy-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.

[16] 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.

[17] 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.

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

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

[24] 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.

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

[26] K. Cho, Y. Lee, Y. H. Oh, G. Hwang, and J. W. Lee, “eDRAMbased 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.

[27] F. Frustaci, D. Blaauw, D. Sylvester, and M. Alioto, “Better-thanvoltage 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] 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.

[38] 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.

[39] 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.

[40] 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.

[41] 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 *2013 26th International Conference on VLSI Design and 2013 12th International Conference on Embedded Systems*, 2013, pp. 109–114.

[42] 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.

[43] 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.

[44] A. Kulkarni, Y. Pino, and T. Mohsenin, “SVM-based real-time hardware trojan detection for many-core platform,” in *Proc. 17th International Symposium on Quality Electronic Design (ISQED)*, March 2016, pp. 362–367.

[45] X. Xie, Y. Sun, H. Chen, and Y. Ding, “Hardware trojans classification based on controllability and observability in gate-level netlist,” *IEICE Electronics Express*, vol. 14, no. 18, pp. 20170682–20170682, 2017.

[46] 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.

[47] C. Gu, N. Hanley, and M. O’Neill, “Improved reliability of FPGAbased PUF identification generator design,” *ACM Transactions on Reconfigurable Technology and Systems*, vol. 10, no. 3, pp. 20:1–20:23, 2017.

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

[49] 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.

[50] 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.

[51] U. Ruhrmair, F. Sehnke, J. S¨ olter, G. Dror, S. Devadas, and J. Schmid-¨ huber, “Modeling attacks on physical unclonable functions,” in *Proc. 17th ACM Conference on Computer and Communications Security(CCS)*, 2010, pp. 237–249.

[52] 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. [53] “On the pitfalls of using arbiter-PUFs as building blocks.”

[54] 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.

[55] “Post-quantumcryptography,” Last accessed 16 January 2018.
[Online]. Available: https://pqcrypto.org/

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

[57] 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.

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

[59] 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.

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

[61] 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.

[62] 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.

[63] F. Tehranipoor, W. Yan, and J. A. Chandy, “Robust hardware true random number generators using DRAM remanence effects,” in *Proc. IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, May 2016, pp. 79–84.

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

[65] 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.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] L. Groot Bruinderink, A. Hulsing, 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.

[72] 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.

[73] J. Von Neumann, "Various techniques used in connection with random digits," *NBS Applied Mathematics Series*, vol. 12, 1961.

[74] T. Poppelmann and T. G¨ uneysu, "Towards practical lattice-based¨ public-key encryption on reconfigurable hardware," in *Proc. Selected Areas in Cryptography (SAC)*, T. Lange, K. Lauter, and P. Lisonek,ˇ Eds. Springer Berlin Heidelberg, 2014, pp. 68–85.

[75] 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.

[76] 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.

[77] Y. Zhang, C. Wang, D. E. S. Kundi, A. Khalid, M. O'Neill, and W. Liu, "An efficient and parallel r-lwe cryptoprocessor," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 5, pp. 886–890, 2020.

[78] 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.

[79] 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.

[80] 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.

[81] S. Bian, M. Hiromoto, and T. Sato, "Darl: Dynamic parameter adjustment for lwe-based secure inference," in *2019 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2019, pp. 1739–1744.

[82] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2014.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

[87] 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.

[88] 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.

[89] 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.

[90] A. Heuser and M. Zohner, "Intelligent machine homicide," in *Proc. International Workshop on Constructive Side-Channel Analysis and Secure Design*, 2012, pp. 249–264.

[91] 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.

[92] 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.

[93] 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.

[94] 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.

[95] 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.

[96] K. Shamsi, M. Li, T. Meade, Z. Zhao, D. Z. Pan, and Y. Jin, "AppSAT: approximately deobfuscating integrated circuits."

[97] 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.

[98] 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.

[99] V. Sze, Y. Chen, T. Yang, and J. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec 2017.

[100] K. Hasegawa, M. Oya, M. Yanagisawa, and N. Togawa, "Hardware trojans classification for gate-level netlists based on machine learning," in *Proc. IEEE 22nd International Symposium on On-Line Testing and Robust System Design (IOLTS)*, 2016, pp. 203–206.

[101] TrustHub, "Trusthub.org," Last accessed 12 January 2018. [Online]. Available: http://trust-hub.org/

[102] W. Liu, L. Zhang, Z. Zhang, C. Gu, C. Wang, M. O'Neill, and F. Lombardi, "XOR-based low-cost reconfigurable PUFs for IoT security," *ACM Transactions on Embedded Computing Systems*, vol. 18, no. 3, pp. 25:1–25:21, 2019.

[103] J. Rajendran, Y. Pino, O. Sinanoglu, and R. Karri, "Security analysis of logic obfuscation," in *Proc. Design Automation Conference (DAC)*, June 2012, pp. 83–89.

[104] S. Hashemi, H. Tann, F. Buttafuoco, and S. Reda, "Approximate computing for biometric security systems: A case study on iris scanning," in *Proc. Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2018, pp. 319–324.

[105] B. Li, P. Gu, Y. Wang, and H. Yang, "Exploring the precision limitation for rram-based analog approximate computing," *IEEE Design & Test*, vol. 33, no. 1, pp. 51–58, 2015.

[106] M. Wang, W. Cai, K. Cao, J. Zhou, J. Wrona, S. Peng, H. Yang, J. Wei, W. Kang, Y. Zhang *et al.*, "Current-induced magnetization switching in atom-thick tungsten engineered perpendicular magnetic tunnel junctions with large tunnel magnetoresistance," *Nature communications*, vol. 9, no. 1, pp. 1–7, 2018.

[107] M. Wang, W. Cai, D. Zhu, Z. Wang, J. Kan, Z. Zhao, K. Cao, Z. Wang, Y. Zhang, T. Zhang *et al.*, "Field-free switching of a perpendicular magnetic tunnel junction through the interplay of spin–orbit and spintransfer torques," *Nature electronics*, vol. 1, no. 11, pp. 582–588, 2018.

[108] X. Lin, W. Yang, K. L. Wang, and W. Zhao, "Two-dimensional spintronics for low-power electronics," *Nature Electronics*, vol. 2, no. 7, pp. 274–283, 2019.

[109] Z. Liu, T. Liu, J. Guo, N. Wu, and W. Wen, "An ECC-free MLC STTRAM based approximate memory design for multimedia applications," in *Proc. IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, July 2018, pp. 142–147.

[110] B. Zeinali, D. Karsinos, and F. Moradi, "Progressive scaled STTRAM for approximate computing in multimedia applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 7, pp. 938–942, July 2018.

[111] E. Kult¨ ursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu, "Eval-¨ uating STT-RAM as an energy-efficient main memory alternative,"

in *Proc. IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, April 2013, pp. 256–267.

[112] H. Zhao, L. Xue, P. Chi, and J. Zhao, “Approximate image storage with multi-level cell STT-MRAM main memory,” in *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, Nov 2017, pp. 268–275.

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 90 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 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 (2018.12019.12). 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 (M’16–S’14) received the Ph.D. degree from Queen’s University Belfast, Belfast, U.K., in 2016. She received the M.Sc. degree with distinction in data communications from The University of Sheffield, Sheffield, U.K., in 2006. She is currently an Assistant Professor in the Center for Secure Information Technologies (CSIT), Queen’s University Belfast, U.K.. Before joining Queen’s University Belfast, she was an electronic engineer in vehicle security and communication system design of GAC Mitsubishi Corporation, China. Her current research interests include hardware security and trust, physical unclonable functions (PUFs), approximate computing for hardware security, true random number generator (TRNGs), hardware Trojan detection, logic obfuscation circuit and machine learning attacks.

Maire O’Neill´ (M’03-SM’11) is currently Director of the UK Research Institute in Secure Hardware and Embedded Systems (RISE). She is Chair of Information Security and is Research Director of Data Security Systems at the Centre for Secure Information Technologies (CSIT), Queen’s University Belfast. She also leads the EU H2020 SAFEcrypto (Secure architectures for Future Emerging Cryptography) project (www.safecrypto.eu). She previously held an EPSRC Leadership Fellowship (2008-2014) and was a former holder of a Royal Academy of Engineering research fellowship (2003-2008). She has received numerous awards for her research work which include a 2014 Royal Academy of Engineering Silver Medal and British Female Inventor of the Year 2007. She has authored two research books and has over 130 peer-reviewed conference and journal publications. She is an Associate Editor for IEEE TC and IEEE TETC and is an IEEE Circuits and Systems for Communications Technical committee member. She is a Fellow of 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, postquantum cryptography and quantum-dot cellular automata circuit design.

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.