Cluster Computing
https://doi.org/10.1007/s10586-021-03324-1

An analysis and evaluation of lightweight hash functions for blockchain-based IoT devices

Sa'ed Abed[1] · Reem Jaffal[1] · Bassam J. Mohd[2] · Mohammad Al-Shayeji[1]

Received: 7 December 2020 / Revised: 26 April 2021 / Accepted: 31 May 2021

Abstract
Blockchain is among the most promising new technologies due to its unique features, encompassing security, privacy, data integrity, and immutability. Blockchain applications include cryptocurrencies such as Bitcoin. Recently, many other applications have begun to deploy blockchain in their systems. These applications include internet of things (IoT) environments. Although deploying blockchain in IoT architecture has yielded numerous advantages, issues and challenges have arisen that require further research. Most IoT devices and platforms have limited storage capacity, low battery power, and limited hardware resources for computation and network communication. Thus, energy efficiency is a critical factor in these devices. On the other hand, blockchain requires extensive resources and high computational capabilities for mining and communication processes. Balancing computation complexity and IoT resources is a fundamental design challenge in implementing blockchain functions, including the hash function, which is crucial to blockchain design for the mining process. In this study, we present a literature review on the common hash functions used in blockchain-based applications, in addition to the lightweight hash functions available in literature. We evaluate and test the common lightweight hash functions (SPONGENT, PHOTON, and QUARK) on FPGA platforms to determine which is most suitable for blockchain-IoT devices. Moreover, we assess lightweight hash functions in terms of area, power, energy, security, and throughput. The results show tradeoffs between these hash functions. SPONGENT performs best on security and throughput. QUARK consumes the least power and energy but has the lowest security parameters. PHOTON utilizes less area and offers a balance between multiple performance metrics (area, energy, and security), rendering it the most suitable lightweight hash function.

Keywords Blockchain · IoT · Resource-constrained devices · Lightweight hash function · Power · Energy · Security

1 Introduction

Blockchain is one of the most promising technologies that have emerged in recent years. The blockchain concept was initially introduced in 2008 by Nakamoto in [1] and was initially proposed for the Bitcoin cryptocurrency system. Bitcoin is a peer-to-peer electronic cash system that facilitates secure electronic cash transactions and online payment without requiring third-party control through a bank or financial institution; instead, it uses cryptography technology to validate transactions and build trust. Blockchain is a public ledger technology that depends on distributed computing and decentralized data-sharing environments to record transactions across numerous users [2]. Another cryptocurrency, the Ethereum platform, was introduced in

✉ Sa'ed Abed
s.abed@ku.edu.kw

Reem Jaffal
reem.jaffal@ku.edu.kw

Bassam J. Mohd
bassam@hu.edu.jo

Mohammad Al-Shayeji
m.shayeji@ku.edu.kw

[1] Computer Engineering Department, College of Engineering and Petroleum, Kuwait University, P. O. Box 5969, Safat 13060, Kuwait

[2] Computer Engineering Department, College of Engineering, The Hashemite University, P.O. Box 330127, Zarqa 13133, Jordan

Published online: 06 June 2021

2013 by Vitalik Buterin in [3], as a transaction-based state machine to program a more generalized blockchain technology [4]. It allows blockchain to accomplish diverse types of transactions called smart contracts [5, 6]. It has many unique specifications such as secure, private, fast, decentralized, and immutable transactions, along with data integrity, a reliable platform, adaptability, and fault tolerance; these allow for blockchain's application in many other areas outside cryptocurrencies. These areas include the internet of things (IoT) industry, medical applications, identity management, and government [2, 7].

Due to its unique features, blockchain technology has recently gained attention for its possible integration into IoT environments. The IoT is a highly distributed network that connects numerous devices and objects over the Internet [8]. These devices are equipped with sensors that collect data to be stored, transmitted, and analyzed through networking nodes, which can be either servers or interconnected computers [9]. Huge amounts of data are captured and generated every day [10]. IoT deployment is expanding and being implemented in fields and applications such as smart homes, city development, smart healthcare, smart transportation, and smart industry [11]. Security and privacy issues have emerged as devices are connected through the Internet and as the computing infrastructure becomes increasingly complicated, which renders this environment more vulnerable to attacks than other end devices such as PCs and cell phones [12–14]. Moreover, IoT network devices have constraints such as limited computing power resources, storage capacity, and network bandwidth. These resource constraints favor less-complex security arrangements, which makes IoT devices vulnerable to malicious attacks. Energy efficiency is another major concern with IoT end-node devices for enabling long-lasting nodes, as they depend on resource-constrained hardware supplied by limited battery power [15]. Energy supply is one of the most critical resources in constrained-resource devices [16].

It is projected that there will be more than 75 billion IoT-connected devices in use by 2025. This is approximately three times the IoT installed base in 2019 [17]. Deploying blockchain in the IoT architecture may foster improvement by enhancing security and privacy, improving speed, reducing costs, improving traceability and reliability, and eliminating single points of failure [18, 19]. However, blockchain requires extensive resources and high computational capabilities for communication, validation, and mining processes, which constitutes a major challenge for deploying blockchain in IoT architecture. The poor connectivity of IoT network nodes (due to low battery capabilities) is another challenge, as blockchain is designed for stable network connections [18].

The mining process—the process of generating and validating the block in the blockchain—involves the use of consensus protocols, which also include hash functions. Cryptographic techniques (consisting of asymmetric cryptography algorithms and hash functions) play a vital role in blockchain transaction signature and validation processes, thereby ensuring security, privacy, and data integrity. The hash function is used to compute the Merkle root value and generate the unique block hash. Thus, it is used in the block generation process.

The computing requirements and high power consumption of the secure cryptography schemes used in blockchain are a challenge for resource-constrained IoT devices [20]. Asymmetric encryption algorithms are essential for providing the required level of security and privacy in blockchain, although the Rivest Shamir Adleman (RSA), for example, is not suitable for IoT devices because it is power-consuming and insufficiently fast [21]. Furthermore, the key size of the RSA algorithm that is considered to be secure (2048-bit) is not appropriate for implementation in such constrained devices. Currently, the ECC has proven to be a lighter alternative and better than the RSA in terms of throughput and energy consumption [22].

Consensus protocols such as Proof of Work (PoW) are not suitable for resource-constrained IoT devices, because heavy computational work to compute the hash is required to select the miner (the mining process) [23]. Hash functions are essential in blockchain-based systems for the transaction signature process (the digital signature), transaction validation, and for verifying data integrity; however, they also consume energy and power heavily. SHA-256 (Secure Hash Algorithm 256), a member of the SHA-2 hash functions family, is commonly used by Bitcoin. Many researchers have evaluated the performance of SHA-256 on various IoT devices [24, 25] and concluded that SHA-2 hash functions are not suitable for IoT applications due to size and power constraints and the use of low-power secure communication such as AES (instead of hash functions) [26]. Simon (as a hash) [27] and Lesamnta-LW [28] are more efficient. Many alternatives to hash functions used by blockchain-based applications have been marketed as being faster and shortening the energy mining process, such as scrypt [29], X11 [30], Blake-256 [31], and Myriad [32]. However, they are either software-based or unsuitable for IoT devices. Nevertheless, further analysis and performance evaluation is needed for cryptographic hash algorithms to be implemented in IoT devices [15]. A latency issue also appears during the mining process when the proper hash function is not implemented, which delays updates in the block process. Thus, the performance of the hash function affects the availability of the network [33].

To solve blockchain-based IoT devices issues, many lightweight solutions have been suggested. These focus on establishing lightweight architectures [34–36], consensus protocols [36–38], or authentication mechanisms [39]. Alternatively, they propose energy-aware methods (including [40–42]) without considering the selection or evaluation of lightweight hash function alternatives that are suitable for blockchain-based IoT devices, or the energy efficiency issue of the hash functions used. These hash functions should be sufficiently secure, which means that the output size should be at least 256-bit [33] and should not generate collisions [43]; concurrently, they should be energy efficient—as they directly affect the consensus protocols (and therefore the mining process)—and should be compatible with low-power computing constraints and the available resources in IoT devices. Energy efficiency is critical for enabling a long-lasting node deployment. Other performance metrics should also be considered, such as the utilized resources, speed, and throughput.

The hardware mining process has evolved from a CPU-based system to GPU, FPGA, and ASICs [44]. The authors in [45] prove that ASIC Miners-based systems are more efficient than CPU, GPU, and FPGA in terms of area and energy consumption for blockchain-based applications. However, the FPGA platform is more efficient for IoT devices because it offers greater flexibility, re-configurability features, low-cost designs [46], and efficient hardware and resource utilization architecture [47]. Most of the hash functions evaluations are performed with ASIC implementation [26, 28]. No research is available on the implementation and examination of lightweight hash functions for IoT applications on FPGA while considering the energy performance metric.

In this study, we present a survey on the lightweight hash functions investigated in the literature, in addition to the common hash functions used in blockchain-based applications. We thoroughly evaluate lightweight hash functions to find the most suitable candidates for use with IoT devices, based on blockchain-based IoT environment requirements. To the best of our knowledge, this is the first work to assess alternatives to lightweight hash functions for blockchain-IoT applications on the FPGA platform. We focus on the best lightweight hash function in terms of energy, which is the most critical factor in IoT environments. Most of the studies we review do not include energy evaluation in their criteria when implementing lightweight hash functions. High energy consumption affects the performance of consensus protocols (the mining process), which is crucial in blockchain-based systems. We also examine other performance metrics, including utilized area, power, throughput, and security. We determine the optimum lightweight hash function, which balances all of these metrics, although we focus primarily on energy and area as the dominant factors.

In this research, we direct our efforts to delivering the following contributions, as depicted in Fig. 1:

- Overview the lightweight and common hash functions available in the literature.
- Determine the most suitable lightweight hash function for blockchain-based IoT devices.
- Implement designs and techniques of lightweight hash functions in Hardware Description Language (HDL) and realize it in the FPGA platform.
- Determine the performance metrics of the lightweight hash functions implemented, including speed, power, area, energy, throughput, and security.
- Compare the results of existing lightweight hash function techniques to determine the most suitable candidate.
- Prepare the necessary background for further research on blockchain technology.

Finally, our research adds significant value in the area of hardware security. The results of this study may help provide solutions for energy issues with IoT devices regarding lightweight hash functions, in addition to related issues such as area, power, and throughput. This could enhance the security of blockchain-based IoT devices.

The remainder of the paper is organized as follows. Section 2 reviews the literature on existing hash functions, including the general hash functions used by blockchain applications, lightweight hash functions, and the selected hash functions. Section 3 gives background on the sponge construction and the chosen hash functions. Hardware implementation and the evaluation for the chosen hash functions are discussed in Sect. 4. Section 5 summarizes the outcomes of the paper and gives recommendations.

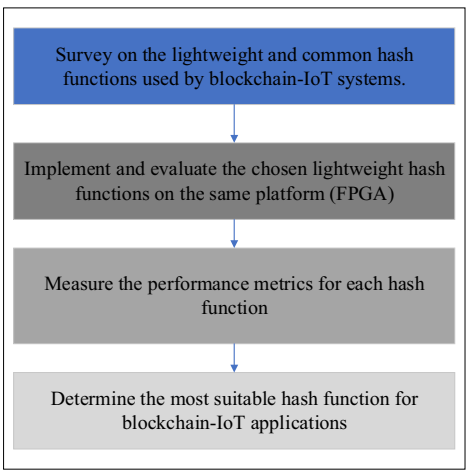

Fig. 1 Research steps

2 Hash functions presented in the literature

In this section, we review the hash functions evaluated in the literature.

2.1 Hash functions of blockchain-based applications

Hash functions are used in various blockchain-based applications, especially cryptocurrencies. Bitcoin [1], the first and the most widely used cryptocurrency, involves implementing the PoW consensus protocol, which in turn uses the SHA-256 [48] hash function. PoW is known to heavily consume power and computation. SHA-256 is used by other cryptocurrencies as well, including peercoin, Titcoin, and Bitcoin cash. The Ethereum platform [3]—another popular example of a blockchain-based application and the first to facilitate the use of smart contracts—implements the proof of stake (PoS) instead of PoW, because PoS is considered to be more secure and energy-efficient than PoW [33]. Coins that use Ethereum implement the Keccak [49] hash function through the Ethash [50] function. Keccak is an SHA-3 candidate and a family of hash functions that is based on the sponge structure [51]. Scrypt is another common hash function used by many cryptocurrencies [52], although it has a lower hash rate than the SHA-256 hash function [33].

A comprehensive survey on blockchain consensus algorithms and the hash functions used by them is presented in [53]. These hash functions include X11, Quark, CryptoNight, scrypt and its variants, and EquiHash. The advantages and limitations of the blockchain consensus protocols and their hash functions are reviewed and explained [53]. Consensus protocols are evaluated in terms of incentives, energy consumption, scalability, security (with respect to adversary tolerance), and ASIC-resistance. However, hash functions are neither implemented nor evaluated on the same platform. The hash algorithms used by various blockchain-based coins are presented in [54]. The top 10 hash algorithms are scrypt [52], X11 [30], SHA-256 [48], Quark [55], X13 [56], CryptoNight [57], EtHash [50], NeoScrypt [58], EquiHash [59], and Xevan [60].

A lightweight blockchain consensus protocol is presented in [38] for low-power IoT devices. This consensus protocol is considered for both general computer hardware and energy-efficient implementation. Various block hashing techniques are presented to determine which is most suitable for blockchain systems. However, there are no real implementations or evaluations of these techniques in terms of energy, power, area, or throughput.

In [35], the authors focus on enhancing the security and performance of encryption and decryption processes by implementing the Elliptic curve ElGamal (EC-ElGamal) algorithm. Additionally, they introduce a high-performance hashing using SHA-384 with the Genetic Algorithm. However, energy evaluation was not included in their evaluation criteria.

2.2 Lightweight hash functions

In [33], the lightweight hash functions QUARK, PHOTON, and SPONGENT are implemented and evaluated based on area, throughput, and power consumption. Cryptographic security is ensured in these hash functions and can be implemented using a small area; they consume less power than existing hash functions such as SHA-256 and are thus appropriate for resource-constrained devices. The selection of these hash functions is based on two factors: the security level and the implemented area. For security, the output hash size should not be less than 256-bit; if it is less than that, the security strength will be insufficient. However, the security of hash functions is based on several factors: output size, number of rounds, and word size. The area required for applying the lightweight hash function should be less than 5000 GEs. A list of recently proposed lightweight hash functions is presented in [61] by Cryptolu; these hash functions were selected from this list. The results show that QUARK performs better in terms of throughput, followed by PHOTON and SPONGENT. In the evaluation of security in terms of preimage resistance, second preimage resistance, and collision resistance, SPONGENT shows the best security, followed by PHOTON and QUARK. Moreover, the consensus protocols that use these hash functions are expected to perform lightly. However, there was no real evaluation and implementation for these light hash functions as far as energy and power on the same platform.

An energy-efficient authentication method is presented for resource-constrained IoT devices by proposing a customized BLAKE2b (c-BLAKE2b) hashing algorithm with a modified elliptic curve digital signature scheme (ECDSA) in [62]. The hardware implementation is conducted using the Raspberry Pi-3 platform under a client–server model. A significant improvement is shown by the proposed system, in contrast to traditional BLAKE2b. The proposed method showed a 0.7–1.91% enhancement in the signature generation time, which is considered a slight enhancement; it showed a 7.67–9.13% enhancement in signature verification time. Thus, it is more efficient and consumes less power in IoT devices. However, this study performed no real-time power and energy evaluation or estimation for the proposed system or hash function. More rigorous security analysis must be conducted, as stated by the authors.

In [27], a modified version of the Simon Cipher 4-block key expansion is presented for use as a low-cost hash function for resource-constrained systems, instead of the SHA-X hash function, which is unsuitable for RFID and Internet of Things (IoT) applications. The reason for choosing the Simon cipher is its high configurability, which enables different hash sizes, depending on the application. The SIMON128/256 configuration is considered to be ideal since it results in very few collisions because of its large output. This study does not include a performance evaluation on energy, power, and area.

An ASIC low-power architecture for the common hash function SHA-256 is shown to be comparable to the smallest available AES implementation in [26]. A comparison with other hash functions, such as SHA-1, MD5, and MD4, is also conducted. The results indicate that the AES cipher is much better than the SHA-256 hash function in both power and chip area. The authors conclude that hash functions like SHA-1, MD5, and MD4 are also less convenient than the AES for use in RFID tags. Hash functions are simpler than block cipher because no keys are used; thus, an evaluation of other lightweight hash functions is still needed.

A new lightweight 256-bit hash function, which is primary targeted for 8-bit CPUs, is proposed in [28]. This is the Lesamnta-LW, with security levels of at least 2^{120} in terms of collision, preimage, and second preimage attacks. This lightweight hash function is based on AES block cipher and is compact, fast, and optimized for lightweight applications. Lesamnta-LW achieved a smaller area implementation on ASIC than most other hash functions, such as MAME and SHA-256. Moreover, it is faster than SHA-256.

In [63], an implementation of various hash functions on an ATMEL AVR ATtiny45 8-bit microcontroller is presented, along with a performance evaluation. All implementations were optimized for code size and memory utilization, using a common interface. The implementations include well-known hash functions such as SHA-256 and SHA-3 candidates, as well as lightweight hash functions such as Quark (S and Q versions), PHOTON (160/36/36 and 256/32/32 versions), SPONGENT (160/160/80 and 256/256/128 versions), and AES-256. The results show that BLAKE performs best, followed by Grostl, Keccak, Skein, and JH. Among the lightweight hash functions, SPONGENT (256-bit) performs best in code size; in terms of RAM use, S- Quark (256-bit) performs best. For cycle count, Keccak (256-bit) has better results, followed by PHOTON (256-bit). This study does not address IoT device constraints. Furthermore, it does not consider energy evaluation, area, power consumption, or throughput, which are important metrics for measuring hash function performance.

Recent lightweight cryptography algorithms, including block ciphers and hash functions, are evaluated from an energy perspective in [64]. This work highlights the role of energy in the profiles of ciphers and differentiates between energy and power concepts. It also discusses the primary sources and sinks of the energy flow in lightweight cryptography applications. It presents the pre-computing technique in improving energy efficiency and the latency of cryptography algorithms. The lowest energy consumption is achieved by the Photon-160 fast hash function, while the highest energy consumption is recorded by Spongent-160 compact. However, different technologies and security levels are used in this work.

The importance of designing lightweight security schemes for IoT systems while considering IoT device constraints (such as limited memory size, power, and computational capabilities) is discussed in [65]. Furthermore, it presents a comparison and evaluation of 9 lightweight hash functions, in addition to 21 lightweight block ciphers, 19 lightweight stream ciphers, and 5 alternatives to elliptic curve cryptography (ECC). Ciphers were evaluated for hardware and software efficiency, area, power and energy, throughput, latency, and figure of merit (FoM). The results demonstrate that AES and ECC are the most suitable lightweight cryptographic algorithms. However, hash functions were evaluated for security and chip area only. SPONGENT and Lesamnta-LW, followed by Quark, recorded the highest security levels. KECCAK, Spongent, L-Hash, and Hash-one had the smallest chip areas. This work also presents the results from other papers, although the comparisons are not based on real implementations with the same environmental constraints.

In [66], the most popular lightweight cryptography techniques are presented and analyzed for resource-limited devices as replacements for conventional cryptography. The trends in designing lightweight techniques are also discussed. An illustration and explanation of the new lightweight hash functions (SPONGENT, PHOTON, Quark, and Lesamnta-LW) are presented without real evaluation and implementation.

A comprehensive review of all lightweight symmetric cryptographic algorithms, including lightweight block ciphers and lightweight hash functions in hardware and software platforms, is presented in [67]. The review analyzes and compares these lightweight symmetric cryptographic algorithms in terms of area, throughput, and power. However, this survey collects data from different papers and shows the comparisons based on different technologies and platforms. Furthermore, the energy performance metric is not considered.

In [68], lightweight stream ciphers and the lightweight hash functions are implemented and evaluated using FPGA and ASIC. The lightweight hash functions considered are

SPONGENT-160, D-QUARK, PHOTON-160/36/36, and Keccak. The number of cycles (time) and area are evaluated on FPGA, while power consumption results are shown on ASIC. Power consumption was evaluated for SPONGENT-160 and Keccak on ASIC, while for the others, it was evaluated by an equation they provided—not based on real implementation. The FPGA power results are not presented, and the energy evaluation is not considered.

In [69], an FPGA implementation is presented for the lightweight hash functions KECCAK-200 and KECCAK-400, PHOTON, and SPONGENT. These were analyzed in terms of size, throughput, and the ratio between throughput and slices. The results show that SPONGENT has the highest throughput-per-area ratio. KECCAK has the best throughput but consumes the most area. While PHOTON performs better in area, it has the lowest throughput. Power and energy analyses were not presented.

An FPGA hardware implementation of two lightweight hash functions (SPONGENT-88 and LHash-96) is presented in [70], in order to use them as building blocks for security schemes in IoT applications. The evaluation performance metrics considered were resource usage, latency, power, and energy. The results indicate that SPONGENT has better resource utilization and consumes less dynamic power than LHash. However, LHash performs better in throughput and energy. However, the output sizes and security levels for these lightweight hash functions were set to low values.

In [71], different optimized FPGA hardware implementations for LED and PHOTON lightweight hash functions are described and evaluated in terms of area, frequency, throughput, and efficiency. The implementations include round-based, serialized, and proposed serialized architecture based on SRL16s. Comparisons to other hash functions show that the presented implementations utilize less area. However, these comparisons were not conducted in the same environment and under the same constraints. Other critical performance metrics such as power and energy were not evaluated.

A presentation and comparison of PHOTON and Quark implementations in terms of design, security, and performance is given in [72]. The software implementation results are presented with comparisons to other lightweight hash functions, such as SPONGENT and Keccak. The hardware implementation results for PHOTON and Quark are presented only for energy and area. The authors conclude that PHOTON utilizes less area and consumes less energy. However, all of the presented performance results were collected from other studies and were not implemented under the same constrains and platforms.

2.3 Blockchain-IoT device constraints

The IoT environment is a distributed network that includes many heterogeneous devices, with limitations such as low computing power resources, limited capacity, and low energy requirements. Several constraints, listed below, must be considered to successfully deploy blockchain technology in the IoT environment:

1. Computational resources: The mining process, which is key in blockchain technology, uses consensus algorithms such as PoW that involve computing the hash value to select the miner [23]. This process requires high computational power, which is not suitable for IoT devices. Although PoS is a lighter alternative to PoW, the mining process remains a primary challenge with IoT resource-constrained devices [33]. Therefore, applying lightweight hashing techniques can help provide more efficient consensus protocols for IoT devices.
2. Energy efficiency: Energy supply is an important factor in constrained-resource devices [16] to maintain the availability of the node. Charging is not available for some deployed IoT devices that rely on their limited energy to perform their functionality [73]. However, blockchain is a power-hungry technology. The consensus protocol, which involves computing the hash, is a primary source of power and energy consumption. Another reason for high energy use is P2P communications at edge devices, which requires a continuous power supply [74, 75].
3. Latency: A complexity of consensus protocols and the hashing technique used can affect the mining process latency. Litecoin [76], for example, preferably uses the scrypt hash function because it is slightly faster than the SHA-256 hash function. This latency is due to the transaction validation process associated with the blockchain nodes, which can take up to 30 min in the case of bitcoin [33]. For IoT applications, the confirmation process should not exceed several minutes. Thus, considering the proper and fast hash function is essential for maintaining network availability.
4. Security level: Blockchain is known to be a secure technology; it brings numerous advantages to the IoT environment, including a high level of security. However, implementing cryptographic algorithms with high security standards in IoT resource-constrained devices remains problematic because it can drain resources. The heavyweight cryptographic algorithms (for example, SHA-256 and X11) applied by blockchain are not compatible with IoT devices [73]. Therefore, the lightweight hash functions considered in blockchain-IoT devices should have an output size

of at least 256 bits to provide a sufficient level of security [33].

2.4 Selection of lightweight hash functions and summary

Various surveys and studies had been conducted to explore blockchain deployment issues in IoT environments. Many articles focus on designing lightweight architectures for blockchain-based IoT systems. Others concentrate on applying lightweight consensus protocols, as they are dominant factors for energy and power consumption. Some studies examine and optimize the asymmetric cryptography algorithms. Several address the lightweight hash functions targeted for blockchain-based IoT environments, which are used through consensus protocols and a mining process consecutively.

Research concerned with consensus algorithms and hash functions can be found in [38] and [53]. Studies and implementations focused on evaluating and implementing hash functions are presented in [26–28, 33, 61, 62, 64, 65, 68–71] Most are ASIC implementations, such as [26, 28, 68, 72]. Some were implemented in FPGA, including [69–71]. Software implementations are presented in [28, 63], and [64]. Others have no real hardware implementations or evaluations, including [33, 38, 53, 63, 64, 66, 67].

Table 1 summarizes the lightweight hash functions presented in the literature, the platform used in each study, and the feedback. QUARK, PHOTON, SPONGENT, and KECCAK are the most popular of these and are commonly targeted for hardware implementation. However, evaluations of energy, power, and area are not presented in these studies. Other lightweight hash functions, such as GLUON, L-Hash, Hash-One, and NEEVA, have small output sizes and are not our targets for blockchain technology applications. While Lesamnta-LW shows good performance, it was designed primarily for a software platform. KECCAK is a popular hash function, although it is not considered in this study because it has a larger area than QUARK, PHOTON, or SPONGENT [69, 77, 92]. Therefore, we focus on QUARK, PHOTON, and SPONGENT for evaluation. The output size considered is 256-bit, as the main target is blockchain-based applications. We believe that our study is the first to implement and evaluate these lightweight hash functions on the FPGA platform while performing power and energy evaluations, in addition to area and throughput metrics. We consider a composite metric (the energy-area metric) that is considered to be the most critical metric in IoT constrained devices targeted for blockchain. We analyze these hash functions and determine the best hash function in each performance metric.

The energy consumption issue in the blockchain-IoT environment is discussed in the literature through reviews or overviews. The recent work in [40] presents an overview of energy management solutions, highlighting effective techniques for reducing energy and power consumption in the IoT field. Additionally, the existing studies on energy management solutions are categorized using a proposed taxonomy. The challenges and issues of efficiently implementing low power and energy consumption methods were also presented. Another review [42] focuses on the integration of blockchain-as-a-service (BaaS) models into IoT environments and discusses the BaaS role in managing energy in various blockchain-IoT fields. A recent review [41], based on a Systematic Literature Review (SLR), presents the existing green energy methods for blockchain-IoT technology in mobile crowd sensing. Principles and technical aspects of green energy approaches were classified and evaluated. Hash function evaluation was not included in these studies, although it is an important factor that affects blockchain-IoT energy consumption through the mining process. Thus, we devote greater attention to this area.

3 Background of selected lightweight hash functions

In this section, we provide a brief background on the sponge function and the three hash functions chosen (SPONGENT, PHOTON, and QUARK).

3.1 Sponge construction

The sponge construction function, introduced by Bertoni et al. [78], is the main component in designing hash functions. It is favored for lightweight hash functions over the traditional Merkle Damgard construction used in general hash functions (e.g., SHA-1 and SHA-2) [72]. It helps achieve a balance between security and memory requirements by reducing the second preimage security parameter for the same internal state size [79]. Sponge construction is based on a fixed, unkeyed permutation function. The input message m has a variable length while the produced output is of arbitrary length. Sponge construction is considered to be a flexible function because its parameters are wide-ranging and because new hash functions can be instantiated by altering these parameters [72]. Sponge construction, as depicted in Fig. 2, consists of the following parameters:

(1) Input message $m = \{m_0, m_1, m_2, m_3\}$: for example, as shown in Fig. 2, each m_i is a block of r-bit/
(2) Internal state b, where $b = c + r$, c is capacity, and r is rate; b is also known as *width*. The capacity

Table 1 Lightweight hash functions presented in the literature

Paper	Hash Functions	Platform	Results and feedback
[33]	QUARK, PHOTON, SPONGENT	No real implementation	In terms of throughput, QUARK performed best, followed by PHOTON and SPONGENT SPONGENT had the best security, followed by PHOTON and QUARK These lightweight hash functions were not implemented or evaluated for power and energy
[62]	c-BLAKE2b	Raspberry Pi-3	Improvement in signature verification time No real-time power and energy evaluation More security analysis is required
[27]	SIMON128/256 modified configuration	Hardware implementation	Considered to be secure because of larger output than other configurations No evaluation in terms of energy, power, and area
[26]	Low power SHA-256	ASIC	The AES cipher was considered to be better than the proposed SHA-256 Ciphers are more complex and power-consuming than hash functions
[28]	Lesamnta-LW	CPU and ASIC	Compact, fast, and optimized for lightweight applications Targeted for software implementation Had a small area of implementation
[63]	Quark (S and Q versions), PHOTON (160/36/36 and 256/32/32 versions), SPONGENT (160/160/80 and 256/256/128 versions), and KECCAK	ATMEL AVR ATtiny45 8-bit microcontroller	SPONGENT (256-bit) was the best in terms of code size S-Quark (256-bit) performed best on RAM usage KECCAK (256-bit) performed best on cycle count, followed by PHOTON (256-bit) No evaluation of energy, power, and area
[64]	Photon-160, Spongent-160	Different Software and hardware platforms	Photon-160 fast hash function achieved the lowest energy consumption Spongent-160 had the highest energy consumption Different technologies and security levels were used in this work
[65]	Quark, Lesamanta-LW, KECCAK, PHOTON, SPONGENT, GLUON, L-Hash, Hash-One, NEEVA	No real implementation	SPONGENT and Lesamanta-LW, followed by Quark, recorded the highest security levels KECCAK, Spongent, L-Hash, and Hash-one were the lightest in terms of chip area Evaluation of security and chip area were presented only
[66]	SPONGENT, PHOTON, Quark, Lesamnta-LW	No real implementation	Explanation of the lightweight hash functions was presented without real evaluation and implementation
[53]	X11, QUARK, CRYPTONIGHT, SCRYPT and its variants, EQUIHASH	No real implementation	Hash functions were neither implemented nor evaluated on the same platform

Table 1 (continued)

Paper	Hash Functions	Platform	Results and feedback
[67]	DM-PRESENT, PHOTON, KECCAK, QUARK, GLUON, SPONGENT, Neeva ARMADILLO	Different hardware implementation technologies	Comparisons are based on different technologies and platforms The energy performance metric was not considered
[68]	SPONGENT-160, D-QUARK, PHOTON-160/36/36, and Keccak	FPGA and ASIC	Not based on real implementation Neither FPGA power results nor an energy evaluation were presented
[69]	KECCAK-200, KECCAK-400, PHOTON, and SPONGENT	FPGA	SPONGENT had the highest throughput-per-area ratio KECCAK had the best throughput but consumed more area PHOTON performed better in area, but had the lowest throughput Power and energy analysis were not presented
[70]	SPONGENT-88, LHash-96	FPGA	SPONGENT had better resource utilization and consumed less dynamic power than LHash LHash performed better on throughput and energy The output sizes and the security levels for these lightweight hash functions were set to low levels
[71]	LED PHOTON	FPGA	The presented implementations utilized less area than others The comparisons were not conducted in the same environment Power and energy were not evaluated
[72]	PHOTON QUARK	ASIC	PHOTON utilized less area and consumed less energy Results were collected from other studies and were not implemented under the same constraints and platforms

(c) has a direct impact on the security level of the hash function [72].

(3) The fixed permutation (P) function operates on a fixed b internal state.

(4) Output hash $z = \{z_0, z_1\}$: for example, as presented in Fig. 2, Z_i is a part of the hash value, which is of n-bit length (the digest size).

This type of construction involves the following three primary phases [80]:

1. Initialization phase: The message m is first padded by a single bit 1 followed by a specific number of 0 bits up to a multiple of r bits; it is then divided into m_i blocks of r bits. The padding function is necessary to protect the hash function against length extension attacks [69].
2. Absorbing phase: Each m_i message block of r-bit is XORed with the first r-bit of state b, then executed with applications of permutation P.
3. Squeezing phase: After absorbing all message blocks, the squeezing phase begins by producing one or more hash chunks $Z_i$with an r-bit length (the returned rate of the state) until n-bit hash is returned.

The security level of the sponge construction depends on the capacity c and output length n; it is estimated as follows:

1. Collision-resistance: This means that it is difficult to find the identical hash value for two different messages m_0 and m_1 (i.e., $H(m0) = H(m1)$); attacks involve the following amount of computation: min ($2^{n/2}$, $2^{c/2}$).

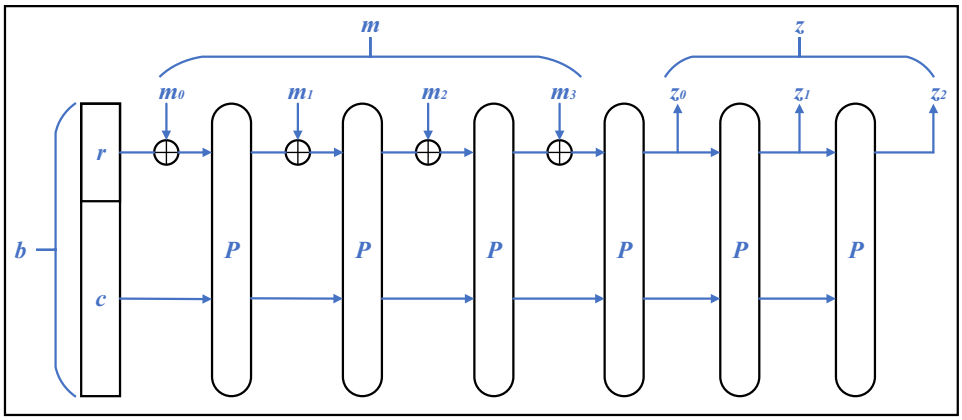

Fig. 2 Sponge construction

2. Preimage resistance: This means that it is difficult to find the message m from its hash value (i.e., $H(m)$), which requires the following amount of computation: min (2^n, $2^{c/2}$).
3. Second preimage resistance: Given m_0, it is challenging to find input m_1 (where $m0 \neq m1$) that has the same hash of input m_0 ((i.e., $H(m0) = H(m1)$). This requires the following amount of computation to attack: min (2^n, $2^{c/2}$).

3.2 SPONGENT

SPONGENT [80] is a lightweight hash function built on a sponge construction that relies on building blocks of the PRESENT block cipher [81]. It is represented as SPONGENT-*n/c/r*, where n is the output hash size, c is the capacity, and r is the rate. The round permutation in this hash function is denoted as π_b; the size of the internal state is b, where b is the sum of c and r ($b = c + r$), as stated previously. The design of SPONGENT offers 13 versions with five digest lengths (88,128,160, 224, and 256 bits) and five different security levels, where the capacity ranges from 80 to 512 and the rate ranges between 8 and 256. The number of rounds R is different for each version; it begins with 45 for the first version and ends at 385 for the last version. In this work, we focus on the largest version, which offers the highest level of security (i.e., SPONGENT- 256/512/256).

The initial value in SPONGENT for b-bit is 0 before the absorbing phase. In all versions of SPONGENT, the hash length n is equal to either c or $2c$, except in the first version. The message blocks are XORed into the r rightmost-bit locations of the state. The same r-bits locations compose parts of the hash output.

The PRESNET-type permutation π_b (shown in algorithm 1) is an R-round transform of the b-bit input state. The permutation round function is iterated R times, where R depends on the SPONGENT version used. It consists of three main functions: $\text{lCounter}_b(i)$, $\text{sBoxLayer}_b(\text{state})$, and $\text{pLayer}_b(\text{state})$. $\text{lCounter}_b(i)$ is the state of an LFSR defined by Primitive Polynomial; a round constant is computed each i round and is added to the rightmost bits of the state. $\text{lCounter}_b(i)_{\text{reversed bit-order}}$ is the same value as $\text{lCounter}_b(i)$, while its bits are order-reversed and are combined to the leftmost bits of the state. In the sBoxLayer_b, the state is divided into 4-bit blocks, with each block mapped to another block of the same size of 4-bit. It uses the same mapping function used by PRESENT S-box [81] and is executed b/4 times in parallel. In FPGA implementation, it should be implemented using look-up tables. pLayer_b is an expansion of the (inverse) present bit-permutation.

Algorithm 1 Pseudo-code of SPONGENT

- *for i = 1 to R do*
  state ← $\text{lCounter}_b(i)_{\text{reversed bit-order}} \oplus \text{state} \oplus \text{lCounter}_b(i)$
  state ← $\text{sBoxLayer}_b(\text{state})$
  state ← $\text{pLayer}_b(\text{state})$
- *end for*

3.3 PHOTON

Guo et al. [82] designed the PHOTON lightweight hash function, which is built on a sponge construction and uses an AES-like permutation. It is represented as PHOTON-*n/r/r'*, where n is the output size ($64 \leq n \leq 256$) and r and r' are the input and output bit rate, respectively. The round permutation in this hash function is denoted as P_t, where t is the size of the internal state and is the sum of the c-bit capacity and the r-bit rate ($t = c + r$); t has 5 different values (100, 144, 196, 256, and 288 bits) depending on the hash output size. Consequently, there are 5 versions of PHOTON, and the internal permutation P_t is defined for each internal t-bit state size with the N_r number of rounds. The number of rounds (N_r) is the same for each version. In

this work, we focus on the largest version, which has the highest level of security (i.e., PHOTON-256/32/32).

For the initialization phase, the t-bit of the internal state is first initialized with a fixed value, and the message m is padded and divided into r-bit blocks. These *r*-bit message blocks are XORed with the bit rate part of the internal state *t*-bit and executed by P_t round permutation in the absorbing phase. The *r*-bit of the bit-rate part of the internal state is produced in the squeezing phase to construct the *n*-bit hash output value.

The permutation P_t involves implementing the following four functions as illustrated in Algorithm 2: AddConstants, SubCells, ShiftRows, and MixColumnsSerial. In the PHOTON, the internal state is represented as a matrix. The process begins by inserting a fixed value in the cells of the matrix (internal state), using the AddConstants function. Next, an *s*-bit Sbox is applied to each cell by the SubCells function, while the positions of the cells in each row are rotated by ShiftRows. Finally, all of the columns are linearly mixed independently via the MixColumnsSerial method.

Algorithm 2 Pseudo-code of PHOTON

- *for i = 1 to R do*
  state ← AddConstants(state)
  state ← SubCells(state)
  state ← ShiftRows(state)
  state ← MixColumnsSerial(state)
- *end for*

3.4 QUARK

Quark, developed by Aumasson et al. [55] in 2010, is based on sponge construction and uses a b-bit permutation function (P_b). The permutation P_b is built on the stream ciphers Grain [83] and the block cipher KATAN [84]. It can be represented as Quark-*n/c/r*, where *n* is the hash length, *c* is the capacity, and *r* is the rate. The size of the internal state is *b*-bits, where *b* is the sum of the capacity and the rate ($b = c + r$). Three different versions are offered by this hash function, with three security levels: U-Quark (64-bit security), D-Quark (80-bit security), and S-Quark (112-bit security). In this study, we are concerned with the highest level of security, offered by the S-Quark version (Quark-*256/224/32*).

In the initialization phase, the message is padded and divided into r-bits blocks. The padded blocks are XORed with the last r-bits of the internal state and then executed by Pb permutation in the absorbing phase. The squeezing phase begins with producing the *r*-bits until n-bits hash output is returned at the end.

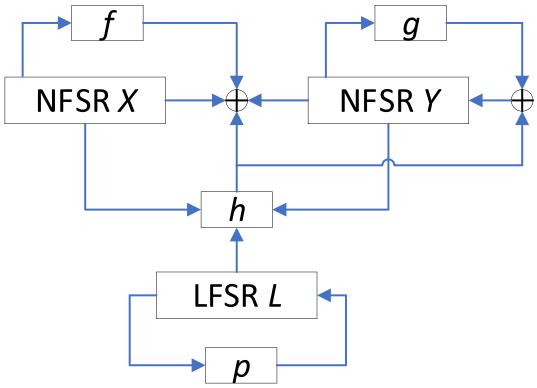

Fig. 3 Quark permutation function

The permutation round function is depicted in Fig. 3. It consists of three feedback shift registers (FSRs): two nonlinear FSRs (NFSRs)—which are f, g (similar to the Grain), and h; —and a one linear FSR (LFSR), which is p. The process of Pb undergoes three phases: initialization, state update, and computation of output. The nonlinear functions f, g, and h are different for each version.

4 Hardware implementation and evaluation results

4.1 Hardware architecture implementation

The hardware implementations offered by these hash functions are parallel and serial architecture implementations. Our focus in this work is the parallel architecture, which implements one round per cycle; the serialized architecture involves implementing a larger number of rounds, as a fraction of round function is performed each cycle [85]. Although the serialized implementation has less area and power, it involves more delay and consumes more energy. Therefore, parallel implementations are more suitable for IoT constrained devices. Additionally, the operation time of parallel architecture is much better than the serialized architecture, which restricts the use of the RFID tag [68]. Therefore, the parallel architecture is more appropriate for these hash functions, targeted for low-resource devices.

The basic round-based hardware architecture is described in Fig. 4. The implemented hash functions (SPONGENT, QUARK, and PHOTON) are all based on sponge construction and follow the same process. However, the permutation round function for each is different, as described in Sect. 3.

The process begins with the initialization phase, where the message $\boldsymbol{m}$ is padded to reach r-bits blocks' length, with each block marked as $\boldsymbol{m_i}$. Additionally, the internal

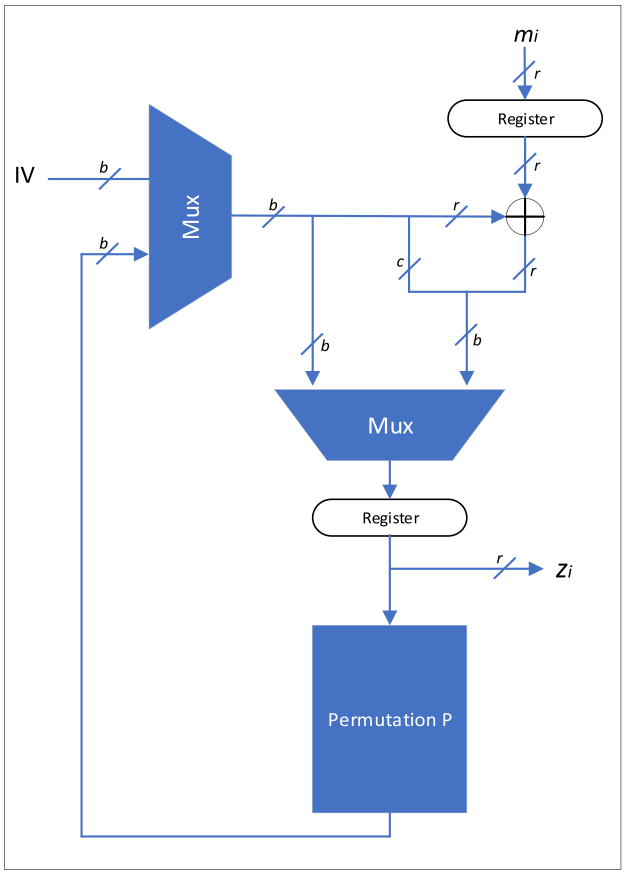

Fig. 4 Round-based architecture

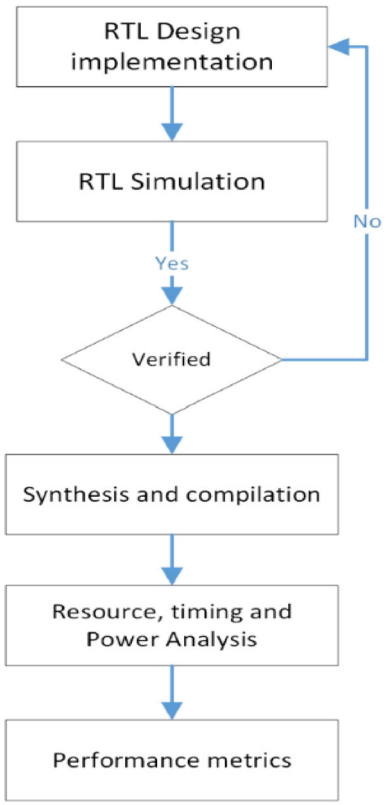

Fig. 5 Design flow

state of b-bits (t-bits in PHOTON) is initialized to 0 (as in SPONGENT) or a specific value (as in PHOTON and QUARK). The initial value is denoted as IV. A multiplexer of 2 × 1 is used to feed in either the IV in the initialization phase or as a feedback to the internal state in the absorbing and squeezing phases. The second phase is the absorbing phase, in which each message block m_i is loaded into a register and XORed with r-bits of the internal state and executed by the permutation round function. This process is repeated until all message blocks are absorbed by the permutation. Another multiplexer of 2 × 1 is used to feed in either the internal state that is XORed with the message blocks (which is the case with the absorbing phase) or the internal state only (which is the case with the squeezing phase) by storing it into the register. Next, the squeezing phase begins by extracting the r-bits of the internal state (i.e., z_i) until the n-bits hash output is returned at the end. This permutation round function is repeated until the maximum number of rounds, as specified by each hash function, is reached.

The FPGA hardware implementation is achieved by following the same procedure used previously in [86–88]. To obtain the evaluation results presented in the next section, these steps are illustrated in the design flow in Fig. 5.

The hash functions were first applied using the register transfer level (RTL) in Verilog™. ModelSim™ was then used to verify the Verilog implementation, which is a dynamic simulation tool. It was used to provide the wave-files that capture the node activity, in order to further estimate the power consumed by the hardware design. Next, the hash functions were compiled and synthesized using the Altera software package Quartus-II. A 50-MHz timing constraint was used during the compilation process. Quartus-II analysis provides the following reports to capture the results:

- Timing analysis reports, which report the maximum frequency of the design.
- Resource utilization analysis, which presents the number of logic elements (LEs) utilized by the hash function.
- Power analysis, which calculates the average dynamic power of the design. The node activities' power is reported from the value change dump (VCD) files produced by ModelSim simulation.

Finally, the performance metrics were presented and computed according to the implementation results. These performance metrics include area, speed/throughput, power, and energy. The most critical of these—the energy-area metric—is computed. We determine the best hash

function that balances these metrics (especially the energy-area metric) while providing the best security level.

4.2 Hardware performance evaluation

To enable fair comparisons and evaluations among the chosen hash functions, we consider the parameters presented in Table 2. We chose the largest version of each hash function, which offers the highest security parameters. The hash output size for each is 256-bit. The evaluation setup is shown in Table 3.

The performance metrics considered in this study are maximum frequency, utilized resources, and power and energy consumption. Table 4 summarizes the implementation results for all hash functions.

4.2.1 Speed and throughput

Figures 6 and 7 show the maximum frequency and throughput for the implemented hash functions, respectively. PHOTON records the highest frequency, which indicates the fastest design, followed by QUARK and SPONGENT. SPONGENT has the lowest frequency because it requires more cycles per block (number of rounds). In terms of throughput, which is computed by calculating the encrypted bits per second, SPONGENT performs best because it has the largest block size, followed by QUARK and PHOTON.

4.2.2 Resource utilization

The area utilization is measured by the number of LEs, which is the sum of combinational LUTs and registers. Figure 8 depicts the total number of LEs for each hash function, along with its components (LUTs and registers number). SPONGENT has the largest number of LEs, followed by QUARK and PHOTON. Thus, PHOTON utilizes less area than the others by approximately 50%; it has the lowest number of registers and combinational LUTs. Power.

Table 3 Evaluation setup

Parameters	Value
FPGA device	Startix-IV
Frequency	50 MHz
I/O timing setup	Half clock cycle
Number of input vectors (for power)	10

4.2.3 Power

The estimated power is the total dynamic power, which is based on three components: combinational logic, clock enable, and register power. Clock enable and register power are the sequential logic. Figure 9 presents the power consumption for all hash functions, including the power components. SPONGENT clearly consumes the largest amount of power; this is expected because it has the largest number of LEs. Its power consumption is higher than that of PHOTON and QUARK by 3 and 5 times, respectively. The reason behind this is noted by the sequential power column (marked as the register in Fig. 9). SPONGENT has high sequential activities for flip flops, which drives up the sequential power. On the other hand, QUARK reports the lowest power consumption because the logic executed in one cycle is less than in PHOTON. Glitch power increases when more logic is required for execution in one clock cycle, which explains the higher power value of PHOTON over QUARK, even though it has the lowest number of LEs, as described in Sect. 4.2.2.

4.2.4 Energy

The energy consumption value is computed for each hash function based on the reported power for each block, using Eq. (1):

$$\begin{aligned} Energy(pJ/block) &= PowerXT_{block} \\ &= PowerX\left(T_{cycle}X\#cycles\right) \end{aligned} \tag{1}$$

Table 2 Hash functions parameters and security levels

Hash function	Hash output size (n)	Data path size (b)	Capacity (c)	Rate (r)	Cycles per block (Number of rounds)	Security		
						Pre	2nd pre	Col
SPONGENT [80]	256	768	512	256	385	256	256	128
PHOTON [82]	256	288	256	32	156	224	128	128
QUARK [55]	256	256	224	32	64	224	112	112

Table 4 Implementation results

Hash function	Fmax (MHz)	LEs	Power (mW)	Energy (pJ/block)
SPONGENT [80]	56.37	1569	28.53	219.681
PHOTON [82]	68.47	783	10.56	35.4816
QUARK [55]	59.9	1205	5.2	8.892

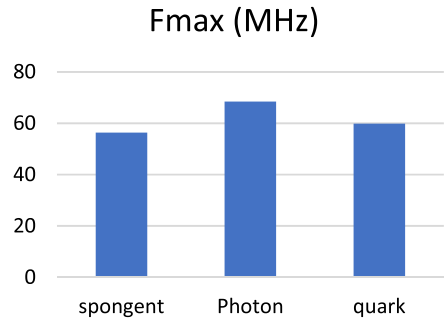

Fig. 6 Maximum frequency of hash functions

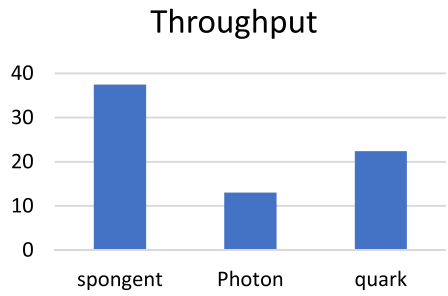

Fig. 7 Throughput of hash functions

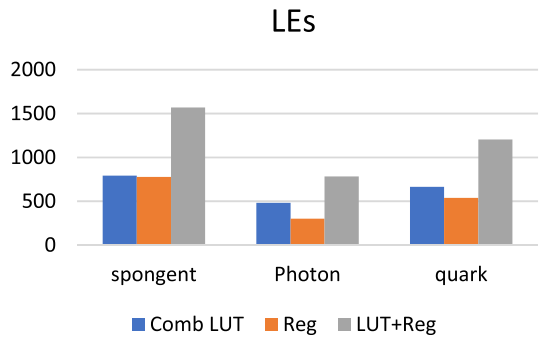

Fig. 8 Resource utilization of hash functions

where T_{block} is the time to process one block and T_{cycle} is the cycle time.

Figure 10 shows the energy consumption for the implemented hash functions. SPONGENT has the highest energy consumption because the energy depends on the average computed power and the number of cycles, which is highest for SPONGENT, followed by PHOTON and QUARK. QUARK has the lowest power and number of cycles for processing one block; thus, it has the lowest

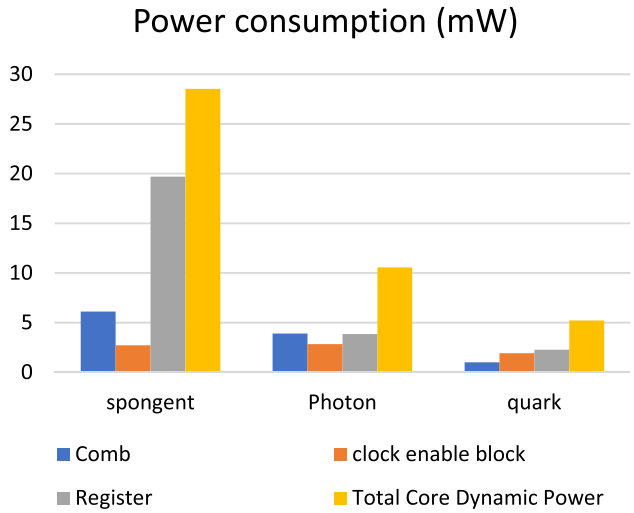

Fig. 9 Power consumption of the hash functions

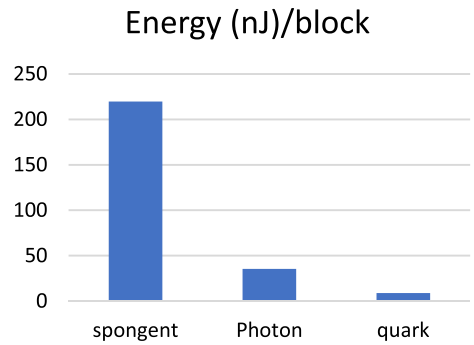

Fig. 10 Energy consumption

energy. It consumes only 4% and 25% as much as SPONGENT and PHOTON, respectively.

4.2.5 The optimum design

To determine the optimum hash function for resource-constrained devices targeted for blockchain-based applications, we assess two primary factors: area and energy. These hash functions should have the lowest number of LEs (utilizing less area) and provide a long battery life by consuming less energy. Therefore, Eq. (2) is used to measure the optimum hash function [87]:

$$Optimum\,Metric = EXLE \tag{2}$$

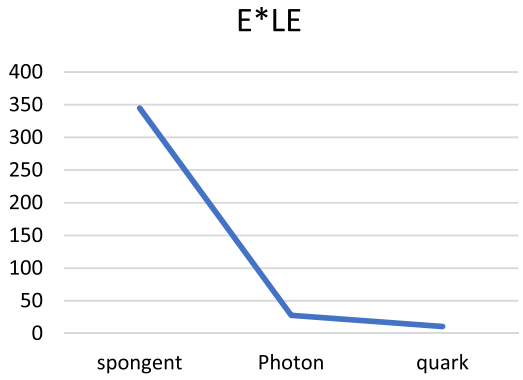

Fig. 11 Optimum design

Figure 11 plots the optimum design equation trend. QUARK has the lowest value—it consumes less energy and has the lowest number of LEs. It is followed by PHOTON, then SPONGENT. However, we must examine the security side. As stated previously, blockchain-based applications should offer a high level of security. Thus, the optimum hash function should balance between optimum metric Eq. (2) and the security level. Table 2 shows that QUARK has the lowest collision and second preimage parameters among the chosen hash functions. Thus, it cannot be considered an optimum design. The second choice is PHOTON, which recorded a higher value than QUARK by 62%. Moreover, it offers better security than QUARK, with higher second preimage and collision parameters. Additionally, it utilized only 65% of QUARK's area and had the highest frequency value. SPONGENT has the highest value for Eq. (2); it also consumes more energy and utilizes more area than the others by an average of 22 times. Although SPONGENT has the highest security parameters—and its throughput is 2-times better than the other hash functions—it is considered a bad choice for low-resource IoT devices. Hence, PHOTON is the optimal hash function; it balances the most critical factors (energy and area) while providing a suitable level of security for these devices. Moreover, it achieves better area utilization than SPONGENT and QUARK by 50% and 35%, respectively.

4.3 Research results, and blockchain-based IoT applications and challenges

4.3.1 Research results and blockchain-based IoT applications

The results of our research are valuable for blockchain-based IoT applications in which timing and area constraints are critical. There are numerous such applications:

- Healthcare applications: Recently, IoT-blockchain infrastructures have been developed to secure data in healthcare systems, including [89, 90]. This type of data is called electronic medical records (EMRs) or electronic health records (EHRs) [91].
- Internet of vehicles (IoV) technology: IoV is a new era of IoT, in which the smart things are vehicles [92]. Vehicle communication can be vehicle-to-vehicle, vehicle-to-human, or vehicle-to-everything [93]. Blockchain-IoV systems have been developed to enhance the security and privacy of IoV communications [94, 95].
- IoT in the 5G era: 5G technology is predicted to have more than 1.7 billion connections worldwide by 2025 [96]. Privacy concerns are emerging along with this advancement. Some researchers have created blockchain-based systems to address this issue [97].
- Cloud and fog computing: Cloud and fog computing platforms are built on top of IoT infrastructure to ease communication and storage between users and data centers. The introduction of fog computing technology has accelerated processing for data and services between clients and IoT devices, without relying on cloud services [98]. Blockchain technology can be incorporated into such systems to manage and secure communications between IoT and fog devices [99].
- Crowd-sensing applications: Recently, crowd-sensing applications have gained popularity. These applications rely on IoT technology and involve security and privacy concerns due to the sensitive nature of the user data. The work in [100] integrates blockchain technology in such systems to overcome this issue.

4.3.2 Research challenges and open issues

Despite the implementation of lightweight hash functions in blockchain-based IoT environment, still, there are challenges and issues need to be considered. Depending on the lightweight hash function implemented in authentication protocols for providing security might not be the best solution to avoid attacks in this environment. Rather, implementing a robust security mechanism should be considered. The authors in [101] showed that the C2MP and ULAB authentication protocols which implement the lightweight hash functions are vulnerable to attacks in RFID based IoT systems. Additionally, they designed blockchain-enabled protocols, named CLAB and OLAB and validated the robustness against the compromised reader threats and had the security requirements of RFID systems. Hence, considering the secure mechanism while implementing the lightweight hash functions is a challenging task.

It is known that IoT scale is growing constantly, which affects the performance of blockchain technology. For example, The PoW consensus protocol involves many issues to the blockchain-IoT environment including throughput, delay and high computing recourses consumption, which affect the efficiency of IoT management [102]. Hence, it is essential to consider the proper consensus protocol when implementing the lightweight hash functions. The consensus protocol used should be equipped with the strict networking and computing constraints of IoT devices as well [103]. Many recent researches focus on implementing adaptive and effective consensus protocol in blockchain-based IoT such as improved PBFT [104] and AEchain [105].

In order to successfully implement a secure blockchain-IoT environment and communication protocols, energy and storage requirements of resource-constrained devices should be considered and managed [106]. Hence, designing lightweight communication protocols or implementing energy managements techniques are challenging issues in blockchain-based IoT environment.

5 Conclusion

In this work, we consider the common lightweight hash functions that are targeted for blockchain-based IoT applications. We review the hash functions used by blockchain applications and the lightweight hash functions considered in previous research. We select a lightweight hash list from these lightweight hash functions to determine which is most suitable for blockchain-based IoT applications. IoT devices are known to have numerous constraints, including low area and power supply and long battery life requirements. Therefore, hash functions implemented on IoT devices should be secure and consume energy efficiently where possible. Energy is the most critical factor for consideration when evaluating hash functions, as it affects the mining process in blockchain-based applications. In this work, we focus on evaluating lightweight hash algorithms in terms of energy because few researchers have examined this area. We also consider other performance metrics in our evaluation criteria, including speed, throughput, resource utilization, and power. Moreover, an optimum metric assessing energy and utilized area is presented to determine the most suitable hash function for the targeted application. The evaluated results show that SPONGENT is the most secure and has the highest throughput—approximately 2-times more than PHOTON and QUARK. However, it consumes more area, power, and energy—approximately 22-times more than QUARK and PHOTON. QUARK has the lowest values for power and energy, performs the best on the optimum metric, and has better throughput than PHOTON by 42%. On the other hand, it has the lowest security parameters, which is unsuitable for blockchain-IoT applications. PHOTON achieves better area utilization than SPONGENT and QUARK by 50% and 35%, respectively. Additionally, it consumes only 37% of SPONGENT's power and 16% of its energy. Thus, PHOTON is the most suitable hash function. It balances the most critical factors (energy and area) and provides a suitable level of security for these IoT constrained devices. Future work should examine the significance and impact of our results using data mining and meta-heuristic algorithms.

Author contributions SA proposed the methodology of the research, analysed and interpreted the results and was a major contributor in writing the manuscript. RJ did the literature review, was a major contributor in the experimental results and in writing the manuscript. BJM revised the work, analysed and interpreted the results. MA revised the work and analysed the results. All authors read and approved the final manuscript.

Funding The authors declare that they have no sources of funding.

Data availability The datasets used and/or analysed during the current study are available from the corresponding author on reasonable request.

Code availability The authors declare that the code is available from the corresponding author on reasonable request.

Declarations

Conflicts of interest The authors declare that they have no competing interests.

References

1. Nakamoto, S., Bitcoin, A.: A peer-to-peer electronic cash system. *Bitcoin.* URL: https://bitcoin.org/bitcoin.pdf, pp. 1–9 (2008)
2. Ferrag, M.A., Derdour, M., Mukherjee, M., Derhab, A., Maglaras, L., Janicke, H.: Blockchain technologies for the internet of things: Research issues and challenges. IEEE Internet Things J. **6**(2), 2188–2204 (2018)
3. "Ethereum white paper." https://cryptorating.eu/whitepapers/Ethereum/Ethereum_white_paper.pdf
4. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper **151**(2014), 1–32 (2014)
5. Soulsby, M.: The benefits of the Ethereum Blockchain. (2018). https://medium.com/plutus-it/the-benefits-of-the-ethereum-blockchain-f332e62f7659
6. Kosba, A., Shi, M.E., Wen, Z., Papamanthou, C : 2016 Hawk: the blockchain model of cryptography and privacy-preserving smart contracts, In: IEEE symposium on security and privacy (SP), pp. 839–858 (2016)
7. Wu, M., Wang, K., Cai, X., Guo, S., Guo, M., Rong, C.: A comprehensive survey of blockchain: from theory to IoT

applications and beyond. IEEE Internet Things J. **6**(5), 8114–8154 (2019)
8. Xia, F., Yang, L.T., Wang, L., Vinel, A.: Internet of things. Int. J. Commun Syst **25**(9), 1101 (2012)
9. Stallings, W.: The internet of things: network and security architecture. Internet Protoc. J **18**(4), 2–24 (2015)
10. Ali, A., Latif, S., Qadir, J., Kanhere, S., Singh, J., Crowcroft, J.: Blockchain and the future of the internet: a comprehensive review. arXiv preprint arXiv:1904.00733, pp. 1–21, (2019)
11. Mehmood, Y., Ahmad, F., Yaqoob, I., Adnane, A., Imran, M., Guizani, S.: Internet-of-things-based smart cities: Recent advances and challenges. IEEE Commun. Mag. **55**(9), 16–24 (2017)
12. Sultan, A., Mushtaq, M.A, Abubakar, M.: IOT security issues via blockchain: a review paper. In: Proceedings of the 2019 International Conference on Blockchain Technology, pp. 60–65 (2019)
13. Ahmed, A.W., Ahmed, M.M., Khan, O.A., Shah, M.A.: A comprehensive analysis on the security threats and their countermeasures of IoT. Int. J. Adv. Comput. Sci. Appl. **8**(7), 489–501 (2017)
14. Oravec, J.A.: Emerging 'cyber hygiene' practices for the Internet of Things (IoT): professional issues in consulting clients and educating users on IoT privacy and security. In: IEEE Int Professional Communication Conference (ProComm), pp. 1–5 (2017)
15. Fernández-Caramés, T.M., Fraga-Lamas, P.: A review on the use of blockchain for the internet of things. IEEE Access **6**, 32979–33001 (2018)
16. Mohd, B.J., Hayajneh, T., Vasilakos, A.V.: A survey on lightweight block ciphers for low-resource devices: comparative study and open issues. J. Netw. Comput. Appl. **58**, 73–93 (2015)
17. Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025. https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. By 2015, forecasts suggest that there will be, the internet and can "communicate" with each other
18. Dai, H.-N., Zheng, Z., Zhang, Y.: Blockchain for internet of things: a survey. IEEE Internet Things J. **6**(5), 8076–8094 (2019)
19. "IoT and Blockchain Convergence: Benefits and Challenges." https://iot.ieee.org/newsletter/january-2017/iot-and-blockchain-convergence-benefits-and-challenges.html
20. Li, N., Liu, D., Nepal, S.: Lightweight mutual authentication for IoT and its applications. IEEE Trans. Sustain. Comput. **2**(4), 359–370 (2017)
21. Suárez-Albela, M., Fernández-Caramés, T.M., Fraga-Lamas, P., Castedo, L.: A practical evaluation of a high-security energy-efficient gateway for IoT fog computing applications. Sensors **17**(9), 1978 (2017)
22. Suárez-Albela, M., Fraga-Lamas, P., Fernández-Caramés, T.M.: A practical evaluation on RSA and ECC-based cipher suites for IoT high-security energy-efficient fog and mist computing devices. Sensors **18**(11), 3868 (2018)
23. Ramachandran, G.S., Krishnamachari, B.: Blockchain for the IoT: opportunities and challenges. arXiv preprint arXiv:1805.02818 (2018)
24. Ometov, A., et al.: Feasibility characterization of cryptographic primitives for constrained (wearable) IoT devices., In: IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops), pp. 1–6 (2016)
25. Lunardi, R.C., Michelin, R.A., Neu, C.V., Zorzo, A.F.: Distributed access control on iot ledger-based architecture. In: NOMS 2018–2018 IEEE/IFIP Network Operations and Management Symposium, pp. 1–7 (2018)
26. Feldhofer, M., Rechberger, C.: A case against currently used hash functions in RFID protocols. In: OTM Confederated International Conferences. On the Move to Meaningful Internet Systems, pp. 372–381 (2006)
27. Degnan, B., Rose, E., Durgin, G., Maeda, S.: A modified simon cipher 4-block key schedule as a hash. IEEE J. Radio Freq. Identif **1**(1), 85–89 (2017)
28. Hirose, S., Ideguchi, K., Kuwakado, H., Owada, T., Preneel, B., Yoshida, H.: An AES based 256-bit hash function for lightweight applications: Lesamnta-LW. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **95**(1), 89–99 (2012)
29. "Original Scrypt Function for Tarsnap." http://www.tarsnap.com/scrypt.html
30. "X11 Official Documentation for Dash," [Online]. Available: https://dashpay.atlassian.net/wiki/spaces/DOC/pages/1146918/X11
31. Aumasson, J.-P., Henzen, L., Meier, W., Phan, R.C.-W.: Sha-3 proposal blake. Submission to NIST **229**, 230 (2008)
32. "Myriad." http://myriadcoin.org/
33. Seok, B., Park, J., Park, J.H.: A lightweight hash-based blockchain architecture for industrial IoT. Appl. Sci. **9**(18), 3740 (2019)
34. Abdulqadder, I.H., Zhou, S., Zou, D., Aziz, I.T., Akber, S.M.A.: Bloc-sec: blockchain-based lightweight security architecture for 5G/B5G enabled SDN/NFV cloud of IoT. In: 2020 IEEE 20th International Conference on Communication Technology (ICCT), pp. 499–507 (2020)
35. Guruprakash, J., Koppu, S.: EC-ElGamal and Genetic algorithm-based enhancement for lightweight scalable blockchain in IoT domain. IEEE Access **8**, 141269–141281 (2020)
36. Liu, Y., Wang, K., Lin, Y., Xu, W.: $\mathsf LightChain $: A lightweight blockchain system for industrial internet of things. IEEE Trans. Industr. Inf. **15**(6), 3571–3581 (2019)
37. Puthal, D., Mohanty, S.P., Yanambaka, V. P., Kougianos, E.: Poah: a novel consensus algorithm for fast scalable private blockchain for large-scale iot frameworks. arXiv preprint arXiv:2001.07297, pp. 1–26 (2020)
38. Finlow-Bates, K.: A lightweight blockchain consensus protocol. In: Computer Security Resource Center. Available at https://www.chainfrog.com/wp-content/uploads/2017/08/consensus.pdf. Accessed 8 July 2018 (2017)
39. Khalid, U., Asim, M., Baker, T., Hung, P.C.K., Tariq, M.A., Rafferty, L.: A decentralized lightweight blockchain-based authentication mechanism for IoT systems. Clust. Comput. **15**, 1–21 (2020)
40. Wang, D., Zhong, D., Souri, A.: Energy management solutions in the internet of things applications: technical analysis and new research directions. Cognit Syst Res. **67**, 33–49 (2021)
41. Sisi, Z., Souri, A.: Blockchain technology for energy-aware mobile crowd sensing approaches in Internet of Things. In: Transactions on Emerging Telecommunications Technologies, p. e4217.
42. Li, D., Deng, L., Cai, Z., Souri, A.: Blockchain as a service models in the Internet of Things management: systematic review. In: Transactions on Emerging Telecommunications Technologies, p. e4139 (2020)
43. Rogaway, P., Shrimpton, T.: Cryptographic hash-function basics: definitions, implications, and separations for preimage resistance, second-preimage resistance, and collision resistance. In: International workshop on fast software encryption, pp. 371–388 (2004)
44. Taylor, M.B.: The evolution of bitcoin hardware. Computer **50**(9), 58–66 (2017)
45. Magaki, I., Khazraee, M., Gutierrez, L.V., Taylor, M.B.: Asic clouds: Specializing the datacentre. In: 2016 ACM/IEEE 43rd

Annual International Symposium on Computer Architecture (ISCA), pp. 178–190 (2016)
46. Mohd, B.J., Hayajneh, T., Khalaf, Z.A., Yousef, K.M.A.: Modeling and optimization of the lightweight HIGHT block cipher design with FPGA implementation. Secur. Commun. Netw. **9**(13), 2200–2216 (2016)
47. Wollinger, T., Guajardo, J., Paar, C.: Security on FPGAs: state-of-the-art implementations and attacks. ACM Trans. Embed. Comput. Syst. (TECS) **3**(3), 534–574 (2004)
48. Kasgar, A.K., Agrawal, J., Shahu, S.: New modified 256-bit MD 5 algorithm with SHA compression function. Int. J. Comput. Appl. **42**(12), 15 (2012)
49. Van Assche, G., Van Keer, R., Bertoni, G., Daemen, J., Hoffert, S., Peeters, M., "Team Keccak." https://keccak.team/
50. "Ethereum. Ethash." https://github.com/ethereum/wiki/wiki/Ethash
51. Kavun, E.B., Yalcin, T.: A lightweight implementation of keccak hash function for radio-frequency identification applications. In: International Workshop on Radio Frequency Identification: Security and Privacy Issues, pp. 258–269 (2010)
52. Percival, C.: Stronger key derivation via sequential memory-hard functions. BSDCan, pp. 1–16, (2009)
53. Ferdous, M.S., Chowdhury, M.J.M., Hoque, M.A., Colman, A.: Blockchain consensus algorithms: a survey. pp. 1–39 (2020)
54. "Cryptocurrency Algorithms." https://cryptorival.com/algorithms
55. Aumasson, J.-P., Henzen, L., Meier, W., Naya-Plasencia, M.: Quark: a lightweight hash. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 1–15, (2010)
56. "X13". https://www.hashgains.com/algorithms/x13
57. Seigen, M.J., Nieminen, T.: Neocortex, and AM Juarez,'CryptoNight hash function. (2013)
58. Doering, J.: Neoscrypt, a strong memory intensive key derivation function. (2014)
59. Biryukov, A., Khovratovich, D.: Equihash: asymmetric proof-of-work based on the generalized birthday problem. Ledger **2**, 1–30 (2017)
60. "Xevan Algorithm" https://coinguides.org/xevan-coins/
61. "Lightweight Hash Functions" https://www.cryptolux.org/index.php/Lightweight_Hash_Functions#SPN-Hash
62. Rao, V., Prema, K.V.: Light-weight hashing method for user authentication in Internet-of-Things. Ad Hoc Netw. **89**, 97–106 (2019)
63. Balasch, J. et al.: Compact implementation and performance evaluation of hash functions in attiny devices. In: International Conference on Smart Card Research and Advanced Applications, pp. 158–172 (2012)
64. Patrick, C., Schaumont, P.: The role of energy in the lightweight cryptographic profile. (2016)
65. Dhanda, S.S., Singh, B., Jindal, P.: Lightweight cryptography: a solution to secure IoT. In: Wireless Personal Communications, pp. 1–34 (2020)
66. Buchanan, W.J., Li, S., Asif, R.: Lightweight cryptography methods. J. Cyber Secur. Technol. **1**(3–4), 187–201 (2017)
67. Hammad, B.T., Jamil, N., Rusli, M.E., Raba, M.R., Ahmed, I.T.: Implementation of lightweight cryptographic primitives. J. Theor. Appl. Inform. Technol. **95**(19), 5126–5141 (2017)
68. Mikami, S., Watanabe, D., Sakiyama, K.: A performance evaluation of cryptographic algorithms on FPGA and ASIC on RFID design flow. In: 2016 4th International Conference on Information and Communication Technology (ICoICT), pp. 1–6 (2016)
69. Jungk, B., Lima, L.R., Hiller, M.: A systematic study of lightweight hash functions on FPGAs. In: 2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14), pp. 1–6 (2014)
70. Lara-Nino, C.A., Morales-Sandoval, M., Diaz-Perez, A.: Small lightweight hash functions in FPGA. In: 2018 IEEE 9th Latin American Symposium on Circuits & Systems (LASCAS), pp. 1–4, (2018)
71. Anandakumar, N.N., Peyrin, T., Poschmann, A.: A very compact FPGA implementation of LED and PHOTON. In: International Conference on Cryptology in India, pp. 304–321 (2014)
72. Meuser, T., Schmidt, L., Wiesmaier, A.: Comparing lightweight hash functions–PHOTON & quark. (2015)
73. Yang, Y., Wu, L., Yin, G., Li, L., Zhao, H.: A survey on security and privacy issues in Internet-of-Things. IEEE Internet Things J. **4**(5), 1250–1258 (2017)
74. Zhou, Z. et al.: EEP2P: an energy-efficient and economy-efficient P2P network protocol. In: International Green Computing Conference, pp. 1–6 (2014)
75. Sharifi, L., Rameshan, N., Freitag, F., Veiga, L.: Energy efficiency dilemma: P2p-cloud vs. datacentre. In: 2014 IEEE 6th International Conference on Cloud Computing Technology and Science, pp. 611–619 (2014)
76. "Litecoins." https://litecoin.com/en/
77. Jungk, B.: FPGA-based evaluation of cryptographic algorithms. Goethe University Frankfurt am Main, pp. 1–280 (2016)
78. Bertoni, G., Daemen, J., Peeters, M., Van Assche, G.: On the in differentiability of the sponge construction. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 181–197 (2008)
79. Hammad, B.T., Jamil, N., Rusli, M.E., Reza, M.Z.: A survey of lightweight cryptographic hash function. Inter. J. Sci. Eng. Res **8**, 806–814 (2017)
80. Bogdanov, A., Knežević, M., Leander, G., Toz, D., Varıcı, K., Verbauwhede, I.: SPONGENT: a lightweight hash function. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 312–325 (2011)
81. Bogdanov, A. et al.: PRESENT: an ultra-lightweight block cipher. In: International workshop on cryptographic hardware and embedded systems, pp. 450–466 (2007)
82. Guo, J., Peyrin, T., Poschmann, A.: The PHOTON family of lightweight hash functions. In: Annual Cryptology Conference, pp. 222–239 (2011)
83. Hell, M., Johansson, T., Meier, W.: Grain: a stream cipher for constrained environments. Int. J. Wireless Mobile Comput. **2**(1), 86–93 (2007)
84. De Canniere, C., Dunkelman, O., Knežević, M.: KATAN and KTANTAN—a family of small and efficient hardware-oriented block ciphers. In: International Workshop on Cryptographic Hardware and Embedded Systems, pp. 272–288 (2009)
85. Mikami, S., Watanabe, D., Sakiyama, K.: A comparative study of stream ciphers and hash functions for RFID authentications. In: RFIDSec Asia, pp. 83–94 (2013)
86. Mohd, B.J., Hayajneh, T., Yousef, K.M.A., Khalaf, Z.A., Bhuiyan, M.Z.A.: Hardware design and modeling of lightweight block ciphers for secure communications. Futur. Gener. Comput. Syst. **83**, 510–521 (2018)
87. Abed, S., Jaffal, R., Mohd, B.J., Alshayeji, M.: FPGA modeling and optimization of a SIMON lightweight block cipher. Sensors (Switzerland) **19**(4), 1–28 (2019). https://doi.org/10.3390/s19040913
88. Abed, S., Jaffal, R., Mohd, B.J., Alshayeji, M.: Performance evaluation of the SM4 cipher based on field-programmable gate array implementation. IET Circuits Devices Syst. **15**, 121–135 (2021)
89. Guo, R., Shi, H., Zhao, Q., Zheng, D.: Secure attribute-based signature scheme with multiple authorities for blockchain in

electronic health records systems. IEEE Access **6**, 11676–11686 (2018)
90. Esposito, C., De. Santis, A., Tortora, G., Chang, H., Choo, K.-K.R.: Blockchain: a panacea for healthcare cloud-based data security and privacy? IEEE Cloud Comput. **5**(1), 31–37 (2018)
91. Tseng, L., Yao, X., Otoum, S., Aloqaily, M., Jararweh, Y.: Blockchain-based database in an IoT environment: challenges, opportunities, and analysis. Clust. Comput. **23**(3), 2151–2165 (2020)
92. Dandala, T.T., Krishnamurthy, V., Alwan, R.: Internet of Vehicles (IoV) for traffic management. In: 2017 International conference on computer, communication and signal processing (ICCCSP), pp. 1–4 (2017)
93. Butt, T.A., Iqbal, R., Salah, K., Aloqaily, M., Jararweh, Y.: Privacy management in social internet of vehicles: review, challenges and blockchain based solutions. IEEE Access **7**, 79694–79713 (2019)
94. Li, L., et al.: Creditcoin: a privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles. IEEE Trans. Intell. Transp. Syst. **19**(7), 2204–2220 (2018)
95. Huang, X., Xu, C., Wang, P., Liu, H.: LNSC: A security model for electric vehicle and charging pile management based on blockchain ecosystem. IEEE Access **6**, 13565–13574 (2018)
96. "Positive 5G Outlook Post COVID-19: What does it mean for avid gamers?"
97. Fan, K., Ren, Y., Wang, Y., Li, H., Yang, Y.: Blockchain-based efficient privacy preserving and data sharing scheme of content-centric network in 5G. IET Commun. **12**(5), 527–532 (2017)
98. Al-Ridhawi, I., Otoum, S., Aloqaily, M., Jararweh, Y., Baker, T.: Providing secure and reliable communication for next generation networks in smart cities. Sustain. Cities Soc. **56**, 102080 (2020)
99. Ashik, M.H., Maswood, M.M.S., Alharbi, A.G.: Designing a Fog-Cloud architecture using blockchain and analyzing security improvements. In: 2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), pp. 1–6 (2020)
100. Wang, J., Li, M., He, Y., Li, H., Xiao, K., Wang, C.: A blockchain based privacy-preserving incentive mechanism in crowdsensing applications. IEEE Access **6**, 17545–17556 (2018)
101. Aghili, S.F., Mala, H., Schindelhauer, C., Shojafar, M., Tafazolli, R.: Closed-loop and open-loop authentication protocols for blockchain-based IoT systems. Inform. Process.Manag. **58**(4), 102568 (2021)
102. Chen, F., Xiao, Z., Cui, L., Lin, Q., Li, J., Yu, S.: Blockchain for Internet of things applications: a review and open issues. J. Netw. Comput. Appl. **45**, 102839 (2020)
103. Bhushan, B., Sahoo, C., Sinha, P., Khamparia, A.: Unification of Blockchain and Internet of Things (BIoT): requirements, working model, challenges and future directions. Wireless Netw. **27**(1), 55–90 (2021)
104. Li, C., Zhang, J., Yang, X., Youlong, L.: "Lightweight blockchain consensus mechanism and storage optimization for resource-constrained IoT devices. Inform. Process. Manag. **58**(4), 102602 (2021)
105. Khan, S., Lee, W.-K., Hwang, S.O.: AEchain: a lightweight blockchain for IoT applications. IEEE Consumer Electron. Mag. (2021). https://doi.org/10.1109/MCE.2021.3060373
106. Khan, M.A., Salah, K.: IoT security: review, blockchain solutions, and open challenges. Futur. Gener. Comput. Syst. **82**, 395–411 (2018)

Sa'ed Abed received his B.Sc. and M.Sc. in Computer Engineering from Jordan University of Science and Technology, Jordan in 1994 and 1996, respectively. In 2008, he received his Ph.D. in Computer Engineering from Concordia University, Canada. He has previously worked at Hashemite University, Jordan, as an Assistant Professor from 2008–2014. Currently, Dr. Abed is an Associate professor in the Department of Computer Engineering at Kuwait University. His research interests include Formal Methods, VLSI Design and Image Processing. Dr. Abed also served as a reviewer for various international conferences and journals. He published over 90 papers in reputable journals and conferences.

Reem Jaffal received the B.S. degree in computer engineering from College of Engineering and Petroleum, Kuwait University, Kuwait, in 2014. Her M.Sc. degree in computer engineering was received in 2018 from College of Engineering and Petroleum, Kuwait University, Kuwait. She is currently a Research Assistant at Kuwait University. Her current research interests include hardware design, power/energy optimization and cryptography.

Bassam J. Mohd received his B. S. in Computer Engineering from the KFUPM of Dhahran-KSA, his M. S. in Computer Engineering from the University of Louisiana at Lafayette and his PhD from the University of Texas- Austin, 2008. He has worked for several semiconductor companies including Intel, SUN, Synopsys and Qualcomm. He is currently a full professor at Computer Engineering Department, Hashemite University, Jordan. His research interest includes DSP designs, steganographic processors, hardware security, encryption processors, image processing, speech recognition and power/energy optimization.

Mohammad Al-Shayeji received his B.Sc. (Computer Engineering), from the University of Miami, and M.S. (Computer Science) from University of Central Florida. He got his Ph.D. (Computer Science) from the University of Southern California. He is working as Associate Professor in Computer Engineering Department, College of Computing Sciences and Engineering, Kuwait University. He has several publications in different international Journals and Conferences. His research interest includes VOD, Video Servers, Multimedia DMS, Security and Distributed Systems.