A High-Throughput FPGA-Based Elliptic Curve Digital Signature Core for IoT Edge Platforms

Cuong Pham-Quoc[1,2(✉)] and Pham Le Song Ngan[1,2]

[1] Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam
[2] Vietnam National University - Ho Chi Minh City (VNU-HCM), Thu Duc, Ho Chi Minh City, Vietnam
cuongpham@hcmut.edu.vn

Abstract. Information security is significant in many aspects, especially in IoT applications such as healthcare or monitoring. Therefore, cryptography algorithms are usually deployed on IoT edge platforms to ensure the integrity and safety of information. As one of the most attractive and efficient methods for implementing digital signature algorithms (DSA), elliptic curve cryptography (ECC) can be used for many security applications. In this work, we design and build an FPGA-based DSA hardware computing core with the ECC algorithm, called ECDSA, to accelerate the processing throughput of IoT edge platforms. We deploy the proposed system on the Kria KV260 edge computing platform with a Xilinx Zynq UltraScale+ FPGA device. Experimental results with test vectors provided by the National Institute of Standards and Technology (NIST) show that our edge computing platform can generate up to 3,361 signatures per second, with a processing throughput of up to 2.46 Mbps.

Keywords: FPGA · Hardware accelerator · ECC · ECDSA · Edge Computing · IoTs

1 Introduction

Statistics project that by 2025, people will interconnect over 75 billion IoT devices worldwide [20]. Furthermore, global spending on IoT is expected to exceed 1.1 trillion USD by 2023. As the number of IoT devices and financial investments increases, many industries are benefiting from IoT applications. In the healthcare sector [10], wearable IoT devices are increasingly utilized for patient monitoring, encompassing functionalities such as blood pressure tracking, connected inhalers, surgical robots, and intelligent hearing aids. Similarly, the smart home concept relies heavily on IoT devices for functionalities like smart door locks, lighting control, video surveillance, and home appliances [13]. Notably, the proliferation of IoT platforms geared towards smart cities has significantly increased, facilitating innovations such as intelligent streetlights, traffic management systems, and air quality monitoring solutions [5].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2024
O. Gervasi et al. (Eds.): ICCSA 2024, LNCS 14820, pp. 31–43, 2024.
https://doi.org/10.1007/978-3-031-65285-1_3

Despite the widespread adoption of IoT across various industries, challenges persist due to resource limitations and energy constraints inherent in IoT devices and platforms. Additionally, there's a pressing need for enhanced security measures to mitigate risks and minimize operational expenses associated with IoT deployments. Consequently, a growing demand for integrating IoT devices with robust security solutions is increasing [11]. However, effectively accomplishing this task requires addressing the inherent limitations of IoT ecosystems' hardware, networking, and software components.

In recent years, to overcome the limitations in terms of hardware components and processing performance, FPGA-based computing platforms have been exploited for IoT edge computing [4]. However, in terms of data security and protection, there are not many studies in the literature integrating digital signature algorithms to FPGA-based IoT computing platforms for protecting data integrity, confidentiality, and authentication. Therefore, this paper introduces our design and implementation of an ECC-based digital signature algorithm (ECDSA) for FPGA-based edge computing. The proposed system can increase data security levels and processing throughput and performance due to the high parallelism of FPGA devices. We build the proposed system on the Xilinx Kria KV260 edge computing platform as our prototype and test with a dataset provided by NIST [14]. The experimental results show that our platform outperforms state-of-the-art implementation regarding throughput for EDCSA. Our system can offer up to 3,361 signatures per second with a throughput of 2.46 Mbps.

The main contributions of our paper can be summarized in three folds.

1. We introduced our efficient FPGA-based ECC digital signature hardware core for accelerating the processing performance of IoT platforms.
2. We illustrate our proposed system with the prototype on Xilinx Kria KV260 edge computing platform to evaluate performance and throughput.
3. We analyze our experimental results based on the prototype platform and dataset provided by NIST to reference other future studies.

The rest of the paper is organized as follows. Section 2 shows the preliminaries of ECDSA and related work. We introduce our proposed FPGA-based ECDSA core in Sect. 3. Section 4 depicts our prototype with Xilinx Kria KV260 edge computing. Experiments with dataset provided by NIST and the prototype platform are analyzed in Sect. 5. Finally, Sect. 6 concludes our paper.

2 Preliminaries and Related Work

In this section, we first present the background of digital signature in data security and the elliptic curve digital signature algorithm (ECDSA). We then analyzed related work in the literature targeting ECDSA on FPGA devices.

2.1 Digital Signature for Data Security

The Digital Signature Algorithm (DSA) is a widely used cryptographic algorithm that facilitates the creation and verification of digital signatures. DSA offers several critical advantages for digital signature applications, such as: (i) Security: DSA relies on mathematical properties that make it computationally infeasible to forge signatures without knowing the signer's private key. (ii) Non-repudiation: Once a signature is created, the signer cannot deny signing the message, providing a solid form of non-repudiation. (iii) Efficiency: DSA offers relatively fast signing and verification processes, making it suitable for real-time applications. DSA was approved as a digital signature standard (DSS) by NIST in 1994, and it includes four primary steps [6].

1. **Key generation**: The process begins with key generation. The signer generates a pair of cryptographic keys-a private key and a corresponding public key. The private key is kept confidential and used by the signer to create digital signatures, while the public key is shared with others for signature verification.
2. **Message hashing**: Before signing a message, the signer typically calculates a cryptographic hash of the message using a secure hashing algorithm such as SHA-256. This hash serves as a unique message representation and ensures message integrity.
3. **Signing**: The signer generates a digital signature for the message using their private key. This involves applying mathematical operations to the message hash and the private key to produce the signature, which is specific to both the message and the signer's private key.
4. **Verification**: Upon receiving the message and its associated signature, the recipient can verify the authenticity and integrity of the message. The recipient uses the sender's public key to verify the signature. This process involves applying mathematical operations to the message, the signature, and the sender's public key to determine whether the signature is valid. If the verification process succeeds, the recipient can be confident that the claimed sender indeed sent the message and that it has not been altered since it was signed.

2.2 Elliptic Curve Digital Signature Algorithm

ECDSA, approved by NIST in 2000, is a variant of DSA based on elliptic curve cryptography (ECC) for generating private and public keys. Like the original DSA, ECDSA includes four steps: key generation, hash, signing, and verification. During the key generation step, a base point on the curve is chosen to generate private and public keys based on this point. SHA-256 hash function is usually used to hash the plaintext before signing. In the signing step, the signature (r, s) value will be generated based on the point of the curve, the private key, and the hash message. Finally, the receiver will use the public key to validate the signature attached to the received message. Figure 1 depicts the generic DSA and also illustrates the ECDSA variant.

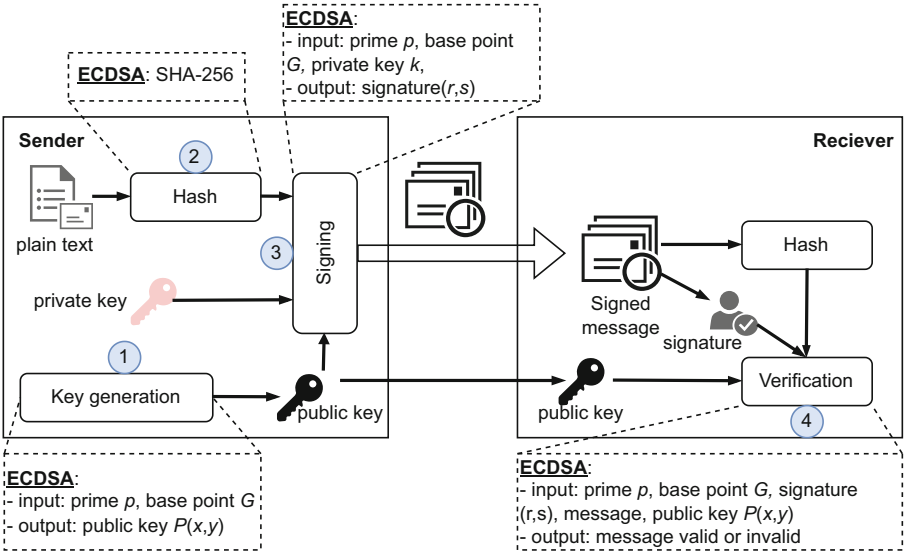

Fig. 1. The general Digital Signature Algorithm and the Elliptic Curve Digital Signature instance

2.3 Related Work

In recent years, many authors have proposed FPGA-based ECDSA systems in the literature. Benjamin Glas introduced a prime field ECDSA signature generating on FPGA [9]. The system is deployed on a Virtex 5 FPGA board that offers up to 138 signatures per second (7.257 ms latency) with the secp256k1 elliptic curve. Elhadjyoussef Wajih targets the low power goal of designing an ECDSA system with a 163-bit elliptic curve [21]. Virtex 4 FPGA devices are used to test the system, which offers a 0.609 ms latency for signing. A Low-resource 32-bit Datapath ECDSA was introduced in [22] that offers 1.58 ms latency for signature generation. The proposed approach targets both ASIC and FPGA technology with experiments on Virtex 5 devices. The testing system uses a 163-bit elliptic curve. Anissa Sghaier designed and implemented an ECC-based DSA hardware core on Virtex 5 FPGA, targeting the low power and area goals [19]. The core used the 163-bit elliptic curve instead of 256-bit in our work. Experimental results show that the system can produce a signature in 1.5ms with a throughput of 0.16 Mbps. A reconfigurable ECDSA FPGA-based core for accelerating a soft processor is proposed in [15]. The proposed core can process the elliptic curve from 192 to 521 bits with Virtex 6 devices and produce signatures after 9.145 ms and 11.92 ms, respectively. An FPGA-based high-speed and area-efficient EDCSA processor was presented in [12]. In this work, a 256-bit elliptic curve and Virtex 7 FPGA device are used to test the processor. The proposed processor can provide 173 kbps throughput with 1.48 ms latency for generating a signature. Binh introduced a DSA engine with both an elliptic curve and an

Edward curve with different sizes of curves (256, 384, 512 bits) [8]. The proposed engine is built with Xilinx Virtex 7 FPGA targeting area constraints. The system requires 15.86 ms for the key generation and signing process with SHA-256.

Compared to these proposals, we target edge computing platforms instead of general-purpose systems. In terms of throughput (signatures per second), we outperform these proposals when testing with the dataset provided by NIST.

3 Proposed ECDSA Hardware Architecture

In this section, we introduce our FPGA-based ECDSA computing core. We design the core according to the hardware accelerator paradigm [17] so that the core can be integrated with different hardwired or soft general-purpose processors to develop applications.

3.1 Overview Architecture

Figure 2 illustrates the overview architecture of our ECDSA core. The core communicates with the host processor through a bus interface for transferring input data (parameters, messages, prime number,...) and output results (signed message, public key,...). Exchange registers and a buffer are used for this purpose.

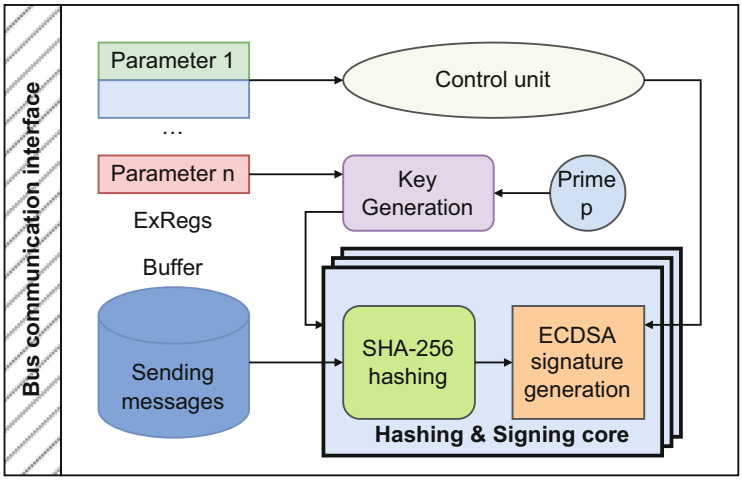

Fig. 2. The overview of the ECDSA hardware core

The main component in this module is the Hashing & Signing core for conducting the hashing and signing steps of the ECDSA process (the second and the third steps). To improve the performance, we implement multiple `Hashing`

`& Signing` cores based on the available resources of the FPGA platforms since this is a time-consuming step. Each core sequentially receives a message from the buffer and conducts SHA-256 hashing before generating the signature according to the ECC algorithm. This work uses SHA-256 to improve security against attacks instead of SHA-1 in proposals presented in Sect. 2.3. To implement the SHA-256 hardware module, we build a hashing table and replace the text of original messages with hashing values. We have yet to show this implementation because this is trivial work and due to space limitations.

3.2 ECDSA Signature Generation

This section presents the details of the ECDSA signature generation sub-module. According to the ECDSA process's operations hierarchy, the signature generation sub-module ultimately requires addition/subtraction modular and multiplication modular operations [7]. Therefore, we present our proposed efficient modular addition/subtraction and modular multiplier architectures as shown in Fig. 3 and Fig. 4, respectively.

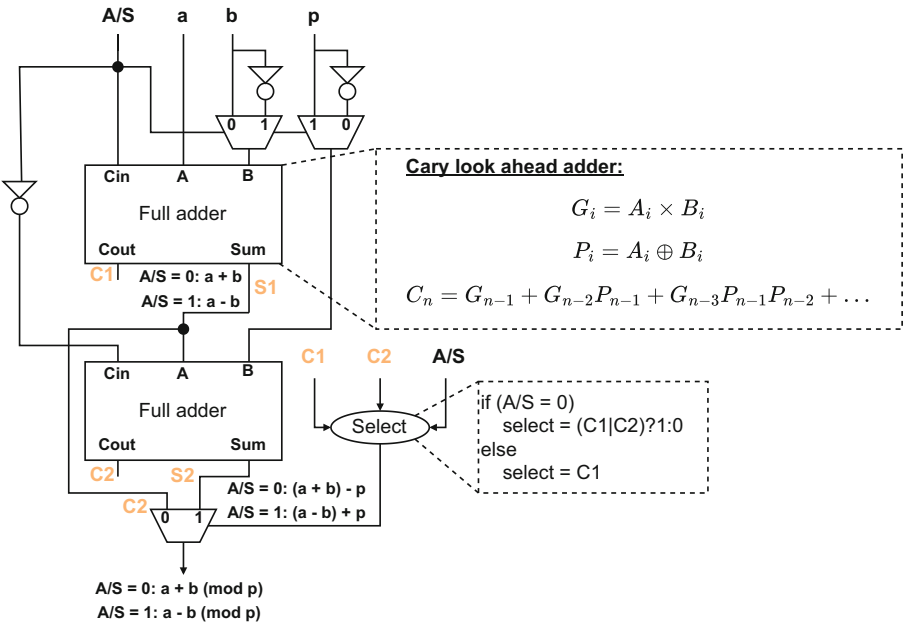

Fig. 3. The modular addition and subtraction

Modular addition and subtraction are the leading operators in the signature generation process. The two operators add or subtract two values a and b before applying the modulo with a selected prime number p to produce $a + b(\text{mod } p)$ or $a - b(\text{mod } p)$ $(a, b \in [0, p-1])$. Figure 3 presents our hardware architecture for computing these operators based on the algorithms presented in [1]. The `A/S` signal allows the core to select modular addition or subtraction.

To calculate modular addition $a+b(\text{mod } p)$, the `A/S` is set to 0 for adding two numbers a and b at the first round. We build our `Full-adder` modules with the carry look-ahead approach to save processing time. The sum of the first adder $S1$ is then added with 2's complement of p for performing $S1 - p$. Finally, the `Select` block selects the final results based on carry-out values `C1` and `C2`. If any carry-out value is 1, then the `S2` output of the second `Full-adder` is the final result; otherwise, selecting the `S1` output of the first `Full-adder`. When `A/S` is set to 1, the core computes modular subtraction $a - b(\text{mod } p)$. Like the modular addition, the first `Full-adder` subtracts b from a by applying 2's complement addition operator. Then, the second `Full-adder` adds the result of the first one with p. The final result is selected based on the value of carry-out bit C1.

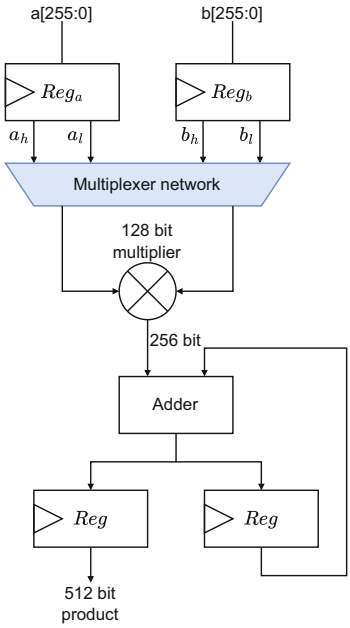

Fig. 4. The modular multiplication

Along with the modular addition and subtraction operators, ECDSA requires many modular multiplication operations. The hardware architecture shown in Fig. 4 represents our pipeline multiplication core. As the SHA-256 and 256-bit

elliptic curves are used for hashing and signature generation, 256-bit multiplication is required. We use the pipeline approach to save computing time by dividing the numbers into high-halves (a_h&b_h) and low-halves (a_l&b_l). These numbers are multiplied and added through a multiplexer network, a 128-bit multiplier, and adders.

4 FPGA Edge Computing Prototype

This section presents our prototype FPGA-based edge computing platform with an integrated ECDSA core. In this work, we use the Xilinx Kria KV260 edge computing system [2] to deploy our prototype. The platform contains a Xilinx Zynq UltraScale+ MPSoC ZU5CG with 256K logic cells, 144 Block RAM (36 Kbits/block), and other components. With this amount of resources, we can build an ECDSA core with two `Hashing and Signing cores`, i.e., two separate messages can be hashed and signed in parallel. Modules for the core are described by Verilog HDL and synthesized with Xilinx Vivado [3]. Along with the reconfigurable logic for building the ECDSA core, the Zynq FPGA chip also contains a hardwired Dual-core ARM Cortex-A53 processor for executing software.

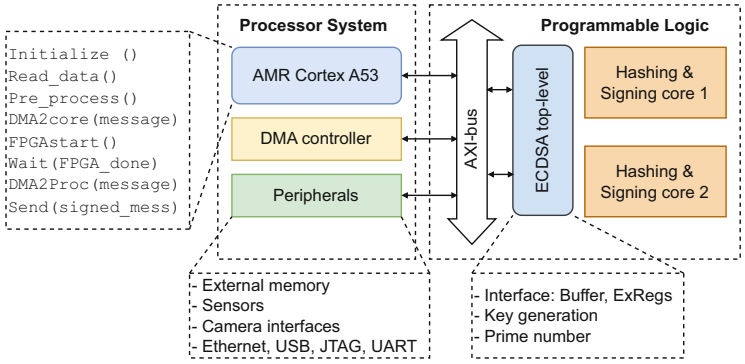

Fig. 5. The prototype system with Xilinx Kria KV260 board

Figure 5 presents the overview of our prototype system. In this prototype, the hardwired ARM processor is a host processor for controlling and collecting data from peripherals like sensors and cameras and communicating with the cloud through the Ethernet interface. Before sending a message to the cloud/server, the host processor starts transferring data to our ECDSA core by Direct Memory Access (DMA) through the AXI bus interface for the signing process. During this interval, the host processor can wait or perform other tasks-the signed messages after that are transferred back from the ECDSA core for sending to the cloud.

5 Experiments

This section presents our experiments for testing and evaluating our ECDSA core and the prototype system. We mainly focus on the signing process's performance; hence, we compare our core with related work in terms of performance and processing throughput.

5.1 Experimental Setup

For testing and validating the proposed ECDSA core, Xilinx Vivado is used for simulating. The prototype system with the validated ECDSA core is then synthesized to the Xilinx Kria KV260 board. The test vector provided by NIST [14] is applied to the simulation and actual system testing process. Figure 6 illustrates a testcase with simulation using the NIST test vector.

```
private=c5c20cf07687e946628012{217447efcd89783594e3be3ffb9bca1deb00420d3,
message=f9643cb57a24131fb061601c7ce0e5a2254ceebd7f737b7f126dcbdb5ce8aea0e2a112a6c0b9fb3f5dfdc1a154bc0c43518481
e70fe8e57c2a5b463c7ac46f45
signature=xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
private=c5c20cf07687e946628012f217447efcd89783594e3be3ffb9bca1deb00420d3
message=f9643cb57a24131fb061601c7ce0e5a2254ceebd7f737b7f126dcbdb5ce8aea0e2a112a6c0b9fb3f5dfdc1a154bc0c43518481
e70fe8e57c2a5b463c7ac46f45
signature=1df1294e2cc2982422011fe6c10dceb5e20e1ce379652a2332b86066aed53c17232d8aa5b07724a117b294aa7fb57e73ab4b
56a16662b600ae81a9cc5b5bb499
run: Time (s): cpu = 00:00:04 ; elapsed = 00:00:07 . Memory (MB): peak = 1753.125 ; gain = 0.000
```

Fig. 6. Simulation result of a message from NIST

A computer connected to the board through the SSH connection functions as a cloud server to receive signed messages for comparison with results provided by NIST. This computer is also used to measure the elapsed time that the system needs for the entire signing process (including data communication overhead between the host processor and cores). The elapsed time is used to compare the performance of the proposed system with related work in the literature and calculate throughput in terms of signatures per second. Figure 7 shows a testing case with the test vector provided by NIST.

```
ubuntu@kria:~/server/file$ source run.sh
remove from slot 0 returns: 0 (Ok)
bram_wrapper: loaded to slot 0
Message: f9 64 3c b5 7a 24 13 1f b0 61 60 1c 7c e1 e5 a2 25 4c ee bd 7f 73 7b 7f 12 6d cb db 5c e8 ae a1 e2 a1
12 a6 c0 b9 fb 3f 5d fd c1 a1 54 bc 0c 43 51 84 81
Private key:c5 c2 0c f0 76 87 e9 46 62 80 12 f2 17 44 7e fc d8 97 83 59 4e 3b e3 ff b9 bc a1 de b0 04 20 d3
Signature: 1d f1 29 4e 2c c2 98 24 22 01 1f e6 c1 0d ce b5 e2 0e 1c e3 79 65 2a 23 32 b8 60 66 ae d5 3c 17 23
2d 8a a5 b0 77 24 a1 17 b2 94 aa 7f b5 7e 73 ab 4b
Compare with expected value: PASS
Latency: 0.597298 ms
```

Fig. 7. Testing result of a message from NIST on Kria KV260

To analyze the hardware resource usage, working frequency, and power consumption, we extract data from the synthesis report generated by Vivado. Based on the latency for processing messages, we compute processing throughput and compare it with other work.

5.2 Synthesis Results

The ECDSA core, developed by Verilog-HDL, and the entire system are synthesized and configured on the Kria board by Xilinx Vivado 2023.2. Synthesis report shows that with two `Hashing & Signing` cores, the system can work at 106.2 MHz and requires 3.1 W total power consumption. Table 1 details hardware resource usage for the system when the system is deployed with one and two `Hashing & Signing` cores.

Table 1. Hardware resources usage for the proposed system with one or two `Hashing & Signing` cores

Resources	2 cores		1 core		Available
	Amount	Percentage	Amount	Percentage	
Look-up tables (LUTs)	60,389	51.56%	32,186	27,48%	117,120
Flip-flop (FFs)	28,932	12,35%	16,557	7.07%	234,240
DSP	66	5.28%	50	4.01%	1,248
Block RAMs (BRAMs)	4	0.85%	4	0.85%	352

As shown in the table, our system uses up to 51,56% hardware resources of the FPGA device, i.e., we can integrate more cores to improve the performance of the computing platform further. With the hardware accelerator computing paradigm, the system also allows us to incorporate other cores for particular purposes, such as video encoder/decoder [16] or attack detection [18] to improve processing ability and overall performance of the IoT edge computing platform.

5.3 Performance Evaluation and Comparison

In this section, we thoroughly evaluate the performance of our system, focusing on three critical metrics: latency for generating a signature, throughput in signatures per second, and throughput in bits per second. Our evaluation is grounded in a comprehensive dataset sourced from NIST, which forms the basis for our rigorous testing procedures. This dataset comprises numerous test cases, each meticulously crafted to contain 64 bytes of data alongside a 32-byte private key. Through careful experimentation and analysis, we aim to provide a detailed understanding of our system's capabilities and limitations across various scenarios. The results of our evaluations, shown in Table 2, shed light on the performance and throughput of our system, particularly when utilizing two `Hashing and Signing` cores. These insights are a foundation for optimizing system design and enhancing overall performance in real-world applications.

Table 2. Performance comparison

Proposal	Device	Elliptic (bits)	Latency (ms)	Signatures/s	Throughput
[9]	Virtex 5	256	7.257	138	N/A
[21]	Virtex 4	163	0.609	1,642	NA
[22]	Virtex 5	163	1.58	633	173 Kbps
[19]	Virtex 5	163	1.5	667	0.16 Mbps
[15]	Virtex 6	192	9.145	109	N/A
		521	11.92	84	
[12]	Virtex 7	256	1.48	676	NA
[8](1)	Virtex 7	256	19.8	202	N/A
		384	44.44	90	
		512	83,34	48	
Ours(2)	**Zynq**	**256**	**0.595**	**3,361**	**2.46 Mbps**

(1)The work in [8] built 4 ECDSA cores. (2) Our work also built two ECDSA cores.

As illustrated in the table, our system demonstrates remarkable efficiency, boasting the most negligible latency in signature generation compared to existing methodologies. Notably, our performance surpasses even proposals employing the 163-bit Elliptic curve, underscoring the robustness and effectiveness of our approach. Moreover, in terms of signatures per second, the utilization of dual `Hashing & Signing` cores offers a significant advantage, propelling us to achieve 2× the throughput of the following best system in this comparative analysis. This superior performance highlights the efficacy of our system architecture and reinforces its viability for demanding real-world applications where speed and efficiency are paramount concerns.

6 Conclusion

Security is paramount in various domains, particularly in Internet of Things (IoT) applications like healthcare and monitoring. Hence, cryptographic algorithms are commonly utilized on IoT edge platforms to safeguard the integrity and confidentiality of data. Elliptic curve cryptography (ECC) stands out as an appealing and efficient approach for implementing digital signature algorithms (DSA), suitable for numerous security applications. In this study, we propose the development of an FPGA-based DSA hardware computing core featuring the ECC algorithm, known as ECDSA, aimed at enhancing the processing throughput of IoT edge platforms. Thanks to the massive hardware resources of FPGA devices, multicore architecture has been applied to further improved the system throughput. Our system is deployed on the Kria KV260 edge computing platform equipped with a Xilinx Zynq UltraScale+ FPGA device. Evaluation using test vectors provided by the National Institute of Standards and Technology (NIST)

demonstrates that our edge computing platform can achieve a signature generation rate of up to 3,361 signatures per second, with a processing throughput reaching up to 2.46 Mbps.

Acknowledgement. We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

References

1. Elliptic Curve Arithmetic, pp. 75–152. Springer New York, New York, NY (2004). https://doi.org/10.1007/0-387-21846-7_3
2. AMD Xilinx: kria kv260 vision AI starter kit user guide (ug1089) (2022), DOIhttps://docs.amd.com/r/en-US/ug1089-kv260-starter-kit Accessed Jan 31
3. AMD Xilinx: Xilinx vivado design suite (2023). https://www.xilinx.com/products/design-tools/vivado.html Accessed 31 Jan 2024
4. Biookaghazadeh, S., Zhao, M., Ren, F.: Are FPGAs suitable for edge computing? In: USENIX Workshop on Hot Topics in Edge Computing (HotEdge 18). USENIX Association, Boston, MA (2018). https://www.usenix.org/conference/hotedge18/presentation/biookaghazadeh
5. Chaudhary, S., Johari, R., Bhatia, R., Gupta, K., Bhatnagar, A.: Craiot: concept, review and application(s) of IoT. In: 2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), pp. 1–4 (2019). https://doi.org/10.1109/IoT-SIU.2019.8777467
6. Chen, L., Moody, D., Regenscheid, A., Robinson, A.: Digital signature standard (DSS) (2023-02-02 05:02:00 2023). https://doi.org/10.6028/NIST.FIPS.186-5
7. Di Matteo, S., Baldanzi, L., Crocetti, L., Nannipieri, P., Fanucci, L., Saponara, S.: Secure elliptic curve crypto-processor for real-time IoT applications. Energies **14**(15), 4676 (2021)
8. Do-Nguyen, B.K., Pham-Quoc, C., Tran, N.T., Pham, C.K., Hoang, T.T.: Multi-functional resource-constrained elliptic curve cryptographic processor. IEEE Access **11**, 4879–4894 (2023)
9. Glas, B., Sander, O., Stuckert, V., Müller-Glaser, K.D., Becker, J.: Prime field ECDSA signature processing for reconfigurable embedded systems. Int. J. Reconfigurable Comput. **2011**, 1–12 (2011)
10. Hasan, M.: IoT in healthcare: 20 examples that'll make you feel better (2020). https://www.ubuntupit.com/iot-in-healthcare-20-examples-thatll-make-you-feel-better Accessed 22 May 22
11. Hossain, M.M., Fotouhi, M., Hasan, R.: Towards an analysis of security issues, challenges, and open problems in the internet of things. In: 2015 IEEE World Congress on Services, pp. 21–28. IEEE (2015)
12. Islam, M.M., Hossain, M.S., Hasan, M.K., Shahjalal, M., Jang, Y.M.: FPGA implementation of high-speed area-efficient processor for elliptic curve point multiplication over prime field. IEEE Access **7**, 178811–178826 (2019)
13. Lanner: examples of IoT devices in your next smart home (2018). https://www.lanner-america.com/blog/5-examples-iotdevices-next-smart-home Accessed 22 May
14. NIST Computer Security Resources Center: cryptographic algorithm validation program (2023). https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/digital-signatures Accessed 31 Jan 31

15. Panjwani, B.: Scalable and parameterized hardware implementation of elliptic curve digital signature algorithm over prime fields. In: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 211–218. IEEE (2017)
16. Pham-Quoc, C.: FPGA-based hardware/software codesign for video encoder on IoT edge platforms. In: International Conference on Computational Science and its Applications, pp. 82–96. Springer (2023). https://doi.org/10.1007/978-3-031-37117-2_7
17. Pham-Quoc, C., Heisswolf, J., Werner, S., Al-Ars, Z., Becker, J., Bertels, K.: Hybrid interconnect design for heterogeneous hardware accelerators. In: 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 843–846. IEEE (2013)
18. Pham-Quoc, C., Thinh, T.N.: FPGA-enabled efficient framework for high-performance intrusion prevention systems. In: International Conference on Computational Science and its Applications, pp. 83–98. Springer (2023). https://doi.org/10.1007/978-3-031-37120-2_6
19. Sghaier, A., Zeghid, M., Massoud, C., Machout, M.: Design and implementation of low area/power elliptic curve digital signature hardware core. Electronics **6**(2), 46 (2017)
20. Statista Research Department: internet of things - number of connected devices worldwide 2015–2025 (2016), https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ Accessed 01 April 2023
21. Wajih, E., Noura, B., Mohsen, M., Rached, T.: Low power elliptic curve digital signature design for constrained devices. Int. J. secur. (IJS) **6**(2), 1–14 (2012)
22. Youssef, N.B.H., Youssef, W.E.H., Machhout, M., Tourki, R., Torki, K.: A low-resource 32-bit datapath ecdsa design for embedded applications. In: 2014 International Carnahan Conference on Security Technology (ICCST), pp. 1–6. IEEE (2014)