

# RTL Design and FPGA Implementation of Optimized Elliptic Curve Cryptography for Secure Data Encryption and Decryption

Ramani Ponguru<sup>1</sup>,

<sup>1</sup>PG Scholar, Department of ECE,  
JNTUA College of Engineering  
Ananthapuramu, Anantapur,  
Andhra Pradesh, India,  
ramaniponguru950@gmail.com

Dr.G.Mamatha<sup>2</sup>

Associate Professor,  
Department of ECE, JNTUA College of  
Engineering Ananthapuramu, Anantapur,  
Andhra Pradesh, India,  
gmamatha.ece@jntua.ac.in

**Abstract**— Elliptic Curve Cryptography (ECC) is a widely used public-key cryptographic technique that provides strong security with relatively smaller key sizes than conventional algorithms such as RSA. This paper presents the design, RTL verification, and FPGA implementation of an optimized ECC processor for efficient encryption and decryption. The proposed architecture incorporates modular arithmetic units for addition, subtraction, multiplication, and inversion to support elliptic-curve point operations and efficient scalar multiplication. The design is developed using Verilog HDL and functionally verified through RTL simulation and waveform analysis in Xilinx Vivado. The verified processor is implemented on a Zynq-7000-based ZedBoard FPGA, where ECC encryption and decryption operations are executed on hardware. On-chip validation is performed using Virtual Input/Output (VIO) and Integrated Logic Analyzer (ILA) for hardware control and internal signal monitoring. The implementation is evaluated in terms of resource utilization, operating frequency, latency, and power consumption, demonstrating the practical feasibility of optimized ECC hardware.

**Keywords**— Elliptic Curve Cryptography (ECC), RTL Design, Scalar Multiplication, Verilog HDL, FPGA Implementation, Zed Board, Encryption, Decryption, Hardware Verification.

## I. INTRODUCTION

Elliptic Curve Cryptography (ECC) is an important public-key cryptographic technique that provides strong security with comparatively smaller key

sizes. ECC is based on the mathematical properties of elliptic curves and primarily employs point

addition, point doubling, and scalar multiplication operations. An elliptic curve over a finite field can be represented in simplified form as  $y^2 = x^3 + ax + b \pmod{p}$ , where (a) and (b) are curve parameters. The fundamental ECC operation is scalar multiplication, represented as  $(Q=kP)$ , where (P) denotes a base point, (k) represents the scalar value, and (Q) is the resulting point. The security of ECC relies on the computational difficulty of the Elliptic Curve Discrete Logarithm Problem (ECDLP) [6].

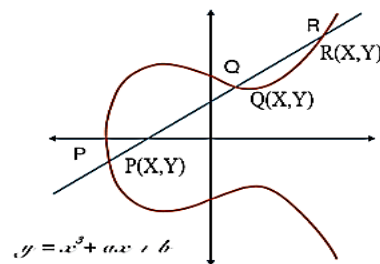


Fig.1.Elliptical curve

Compared with conventional public-key algorithms such as RSA, ECC achieves comparable security with significantly smaller key sizes. The reduced key size can decrease memory requirements, computational overhead, and hardware resource utilization, making ECC. ECC provides the same security as RSA but with much smaller key lengths. As security needs increase, RSA keys must grow exponentially to maintain protection, while ECC keys grow linearly. This creates a widening efficiency ratio, making ECC significantly faster



and less resource-intensive for modern encryption as shown in table 1.

TABLE 1: ECC and RSA Key Length

| ECC Key Length (bits) | RSA Key Length (bits) | Approximate Ratio |
|-----------------------|-----------------------|-------------------|
| 224                   | 2048                  | 1:9               |
| 256                   | 3072                  | 1:12              |
| 384                   | 7680                  | 1:20              |

Despite its security and key-size advantages, ECC involves computationally intensive operations, particularly scalar multiplication through repeated point addition and doubling [11]–[13]. Efficient implementation of modular arithmetic is therefore essential for reducing execution overhead and improving FPGA resource utilization [14]. This work addresses these challenges through a structured and optimized hardware architecture for ECC processing.

The proposed Verilog-based ECC processor integrates dedicated modular arithmetic and point-operation units with FSM-based control for coordinated scalar multiplication. Unlike simulation-only implementations, the design is synthesized and experimentally validated on a Zynq-7000-based ZedBoard FPGA using VIO and ILA. The study reports hardware resource utilization, operating frequency, latency, and power consumption, providing practical implementation-level evidence of the processor's efficiency and suitability for resource-constrained secure embedded systems.

## II. LITERATURE REVIEW

Elliptic Curve Cryptography (ECC) has been widely studied for hardware security, with recent research focusing on scalar multiplication, modular arithmetic optimization, resource reduction, and latency improvement.

Khalid Javeed et al. [1] proposed a compact scalar multiplication architecture using unified  $GF(p)$  arithmetic and the Montgomery Ladder, achieving 151 MHz frequency and 0.67 ms latency on a Virtex-7 FPGA. N. S. Saju and S. K. N. [2] developed a flexible Verilog-based ECC processor using FSM control, although its flexibility increased resource utilization. Other works have investigated

optimized modular multiplication, inversion, dedicated arithmetic units, and parallel architectures [3]–[5].

Despite these developments, limited work simultaneously addresses arithmetic efficiency, control optimization, FPGA resource utilization, and comprehensive hardware-level verification. Simulation alone cannot fully demonstrate actual FPGA behavior, while limited internal signal observation complicates hardware validation.

To address this gap, this work presents an optimized RTL-based ECC processor implemented on a Zynq-7000-based ZedBoard FPGA. Xilinx Vivado is used for verification and implementation, while VIO and ILA enable on-chip control and signal monitoring. Resource utilization, frequency, latency, and power are evaluated to provide a complete assessment from RTL verification to practical FPGA execution.

## III. PROPOSED METHODOLOGY

The proposed ECC architecture is an optimized Verilog HDL-based RTL design for FPGA cryptographic processing. It performs scalar multiplication using reusable modular addition, subtraction, multiplication, and inversion units to reduce hardware redundancy and computational latency. An FSM coordinates data loading, arithmetic operations, encryption/decryption, and result generation. The design is developed and verified in Xilinx Vivado and implemented on a ZedBoard with the Zynq-7000 XC7Z020 device. FPGA validation includes resource, timing, frequency, power, and internal signal analysis using ILA.

The processor accepts an elliptic-curve point  $PP$  and scalar key  $k$  to generate  $Q=kP$ . Input registers store the required operands, while the FSM controls sequential scalar processing and arithmetic execution. Each scalar bit is processed through point doubling and conditional point addition using modular arithmetic units. Intermediate coordinates are stored in datapath registers and updated during execution, with the final point transferred to output registers and completion status asserted.

The figure 2 represents a hardware architecture flowchart for Elliptic Curve Scalar Multiplication

(ECSM) (specifically showing a typical top-down hierarchy of point operation scheduling inside a *Scalar Processing Unit* using underlying arithmetic blocks).

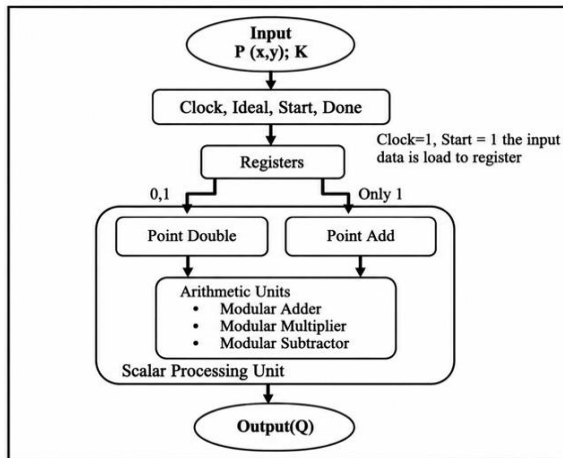


Fig.2: Data Flow Architecture for Double-and-Add Point Multiplication

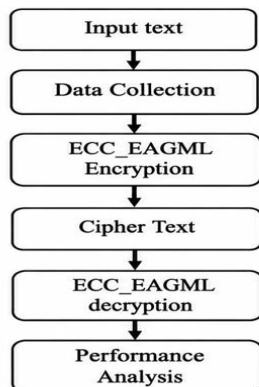


Fig 3: ECC-EAGML Encryption and Decryption Workflow

The process begins with the generation of the required elliptic-curve parameters and cryptographic keys. The input plaintext is then processed by the encryption stage using the recipient's public key to generate the corresponding cipher text. The resulting encrypted data is represented as a cipher text pair and can be transmitted through a communication channel while maintaining the confidentiality of the original information. At the receiving end, the cipher text is processed by the decryption stage using the corresponding private key to recover the original plaintext. The overall workflow consists of key generation, plaintext encryption, cipher text transmission, and plaintext recovery, providing a

structured foundation for the subsequent implementation of the ECC-ElGamal algorithm.

### ECC KEY GENERATION

**Input:** Elliptic curve parameters  $E(a, b, p)$  and base point  $G$

**Output:** Public Key  $Q$  and Private Key  $d$   
 $y^2 = x^3 + ax + b \pmod{p}$

### ECC-ELGAMAL ENCRYPTION

**Input:** Plaintext message  $M$ , receiver public key  $Q$

**Output:** Cipher text  $(C_1, C_2)$

### ECC-ELGAMAL DECRYPTION

**Input:** Cipher text  $(C_1, C_2)$ , private key  $d$

**Output:** Original plaintext message  $M$

The complete algorithm ensures secure communication, reduced computational complexity, and efficient implementation for cryptographic systems.

## IV. FPGA IMPLEMENTATION OF THE PROPOSED ECC

The proposed ECC-ElGamal processor is developed using Verilog HDL and implemented on an FPGA platform using Xilinx Vivado. The RTL architecture integrates the ECC computational units required for scalar multiplication and elliptic-curve point operations. Functional verification is performed through RTL simulation, while FPGA-level operation is controlled and monitored using the Virtual Input/output (VIO) and Integrated Logic Analyzer (ILA) cores. VIO provides configurable control and input signals, whereas ILA captures internal processing signals, enabling real-time observation and validation of the implemented ECC operations.

The proposed ECC processor incorporates an FSM-based control mechanism and dedicated control/status registers to coordinate cryptographic operations. The FSM manages the sequential execution through IDLE, RUN, and DONE states, while the registers provide organized storage for input data, key information, ECC parameters, and

control signals. The START and MODE signals initiate and select the required operation, while RESET and CLOCK ensure proper initialization and synchronous operation. Status signals provide completion information, enabling deterministic control and reliable execution of the ECC processing sequence.

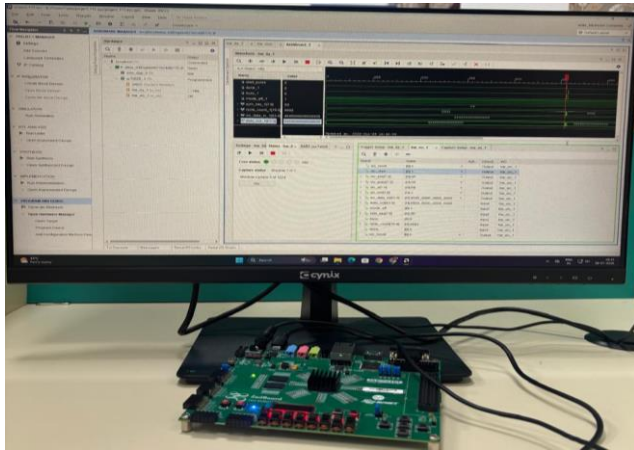


Fig 4: FPGA Implementation

### V. RESULTS AND DISCUSSION

The proposed ECC-ElGamal architecture is developed and simulated using Verilog HDL to verify its functional behavior for secure communication applications. The simulation environment incorporates FSM-based control and the required arithmetic modules to validate the key-generation, encryption, and decryption operations. The obtained results are analyzed in terms of functional correctness, processing behavior, FPGA resource utilization, timing performance, and hardware-level verification. These results demonstrate the practical implementation characteristics of the proposed architecture for FPGA-based cryptographic applications.

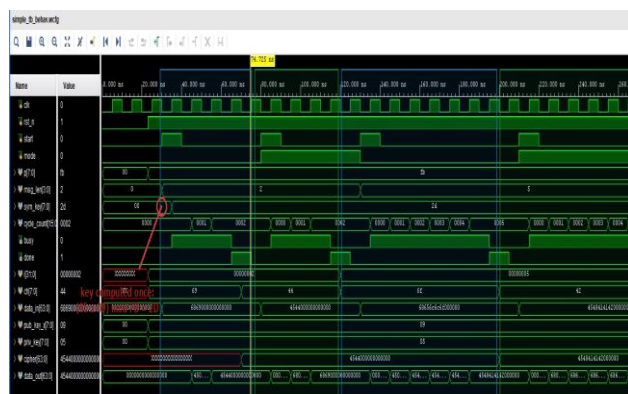


Fig 5: Simulation Results Key generation

The RTL simulation waveform shown in Fig. 4 demonstrates the functional operation and control sequencing of the proposed ECC-ElGamal architecture. The active-low rstnrst\_n signal initializes the design, while clk provides the synchronous timing reference.

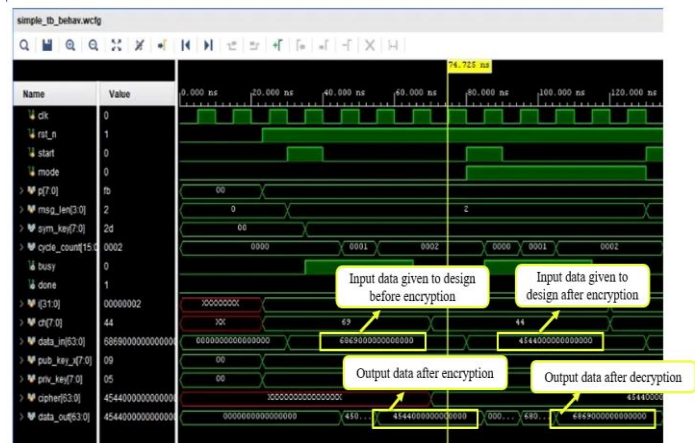


Fig 6: Simulation Results For Encryption And Decryption

The mode signal selects the encryption or decryption operation, and the start signal initiates ECC processing. The input registers data\_indata\\_in, private key/public\_key\_x, and prime field\\_field provide the required data and cryptographic parameters. The resulting processed value is observed at data\_outdata\\_out. The waveform confirms the expected control-signal sequencing, input loading, and generation of the corresponding cryptographic output during the simulated operation.

#### A. FPGA Hardware Implementation and Validation ILA & VIO-Based

The verified ECC-ElGamal RTL design was synthesized and implemented on the target FPGA platform using Xilinx Vivado. The generated bitstream was programmed onto the FPGA board for functional hardware validation. The required cryptographic inputs and control signals were applied through the configured hardware interface, and the processing status and output activity were observed using the onboard LEDs.

In addition, the Integrated Logic Analyzer (ILA) was integrated into the design to monitor selected internal control and data signals during hardware execution. The observed hardware behavior was consistent with the expected RTL operation.

The implemented ECC-ElGamal design was further validated using the Integrated Logic Analyzer (ILA) available in Xilinx Vivado. The ILA was configured to monitor the principal control and data signals during hardware execution, including the clock, reset, operation mode, start control, input data, cryptographic parameters, and processed output. The captured waveform provides cycle-level visibility into the internal operation of the design and enables verification of signal sequencing and data processing under actual FPGA conditions. The observed transitions and output response confirm consistency between the RTL simulation and hardware execution.

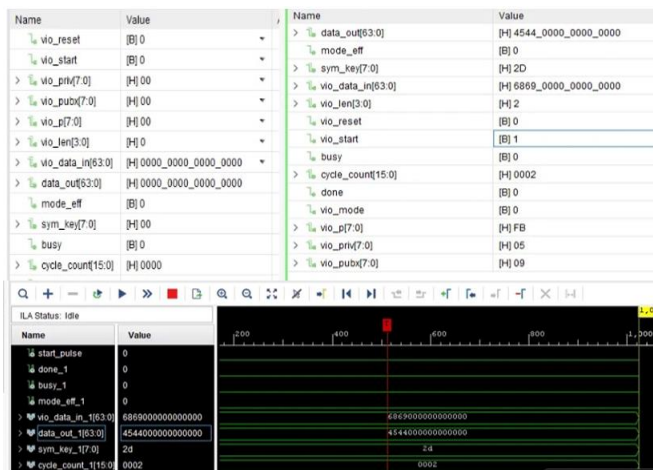


Fig.7 Hardware Verification and Waveform Results for the Encryption Process using VIO and ILA.

The cryptographic core was validated via hardware-in-the-loop (HIL) testing inside Xilinx Vivado using Virtual Input/output (VIO) and Integrated Logic Analyzer (ILA) cores.

As illustrated in Fig. 7, the encryption engine receives a 64-bit plaintext 6869\_0000\_0000\_0000 via the vio\_data\_in register and a 7-bit symmetric key (sym\_key = 2D). Driving vio\_start high initiates processing, which completes successfully within 2 clock cycles (cycle\_count = 0002), producing the cipher text 4544\_0000\_0000\_0000 at data\_out.

Conversely, Fig. 8 displays the loopback decryption phase. Applying the captured ciphertext

4544\_0000\_0000\_0000 to vio\_data\_in under identical system parameters yields the original plaintext 6869\_0000\_0000\_0000 at the data\_out register. The real-time ILA waveforms confirm the structural and mathematical correctness of both the encryption and decryption pipelines under hardware execution

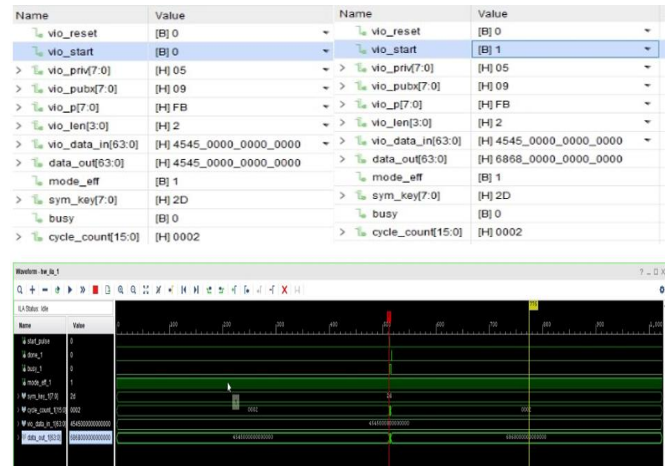


Fig.8: Hardware Verification and Waveform Results for the Decryption Process using VIO and ILA.

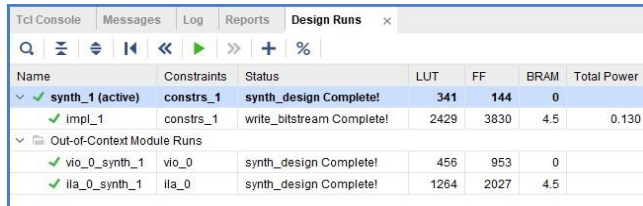
### B. FPGA Resource Utilization and Power Analysis

The power characteristics of the implemented ECC-ElGamal architecture were evaluated using the Vivado power analysis report.. The obtained power results provide an implementation-level assessment of the energy requirements of the proposed ECC architecture under the evaluated operating conditions.

| Time    | Signal event                                                            |
|---------|-------------------------------------------------------------------------|
| 0–23 ns | rst_n = 0, all outputs 0                                                |
| ~25 ns  | Inputs loaded: priv=05, pub_x=09, p=FB, len=2, data_in=6869_..., mode=0 |
| ~30 ns  | start pulses 1 clock; busy↑1; sym_key→ 2D                               |
| +1 clk  | data_out = 4500_0000... ; cycle_count = 1                               |

|        |                                                                      |
|--------|----------------------------------------------------------------------|
| +2 clk | data_out = 4544_0000... ;<br>cycle_count = 2; busy↓0, done<br>pulses |
| ~75 ns | mode→1, data_in = 4544_... (the<br>cipher), start pulses             |

TABLE 2: Step-by-Step, What Happens at Each Instant



| Name                       | Constraints | Status                    | LUT  | FF   | BRAM | Total Power |
|----------------------------|-------------|---------------------------|------|------|------|-------------|
| synth_1 (active)           | constrs_1   | synth_design Complete!    | 341  | 144  | 0    |             |
| impl_1                     | constrs_1   | write_bitstream Complete! | 2429 | 3830 | 4.5  | 0.130       |
| Out-of-Context Module Runs |             |                           |      |      |      |             |
| vio_0_synth_1              | vio_0       | synth_design Complete!    | 456  | 953  | 0    |             |
| ila_0_synth_1              | ila_0       | synth_design Complete!    | 1264 | 2027 | 4.5  |             |

Fig.9, Resource Utilization

As shown in Fig. 9, the obtained resource utilization provides a quantitative measure of the logic and storage elements required for the implemented architecture, while demonstrating a compact hardware footprint for the evaluated ECC-ElGamal design.

| Metric    | Base Paper                      | Optimization                                                       |
|-----------|---------------------------------|--------------------------------------------------------------------|
| Power     | Total: 3.72W<br>Static: 0.49W   | Total: 0.185W<br>Static: 0.122W<br>Dynamic: 0.033W<br>Temp: 25.3°C |
| Frequency | 100 MHz<br>(Processor)          | 7.49 MHz                                                           |
| LUT       | Used: 2812<br>Available: 93120  | Used: 341<br>Available: 134600                                     |
| FF        | Used: 10094<br>Available: 46560 | Used: 144<br>Available: 269200                                     |

TABLE 3: Performance and Resource Utilization Comparison

This table compares hardware resource utilization and power consumption. The optimized design achieves massive efficiency gains over the base

paper, reducing total power from 3.72W to 0.185W. Furthermore, the hardware footprint is drastically minimized, cutting LUT usage by 87% and FFs by 98%, despite a lower operating frequency of 7.49 MHz

### C. Simulation vs Hardware

The design is verified because two independent references:

1. RTL simulation,
2. Physical FPGA

produce identical, bit-exact values for the key, the ciphertext, the recovered plaintext and the latency

| Quantity       | RTL simulation<br>(Vivado/Icarus) | FPGA hardware<br>(VIO/ILA) |
|----------------|-----------------------------------|----------------------------|
| sym_key        | 0x2D                              | 0x2D                       |
| Cipher of "hi" | 4544...                           | 4544...                    |
| Recovered text | "hi"                              | "hi"                       |
| Cycle count    | 2                                 | 2                          |

TABLE 4: Compression between Simulation Results and Hardware Results

TABLE 4 summarizes the functional verification by comparing RTL simulation with physical FPGA hardware execution. Under both environments using a symmetric key of 0x2D, the encryption and decryption of the payload "hi" yielded identical cipher texts (4544...) and successful recovery within precisely two clock cycles, validating the architectural consistency.

## VI. CONCLUSION

This paper presented the design, RTL verification, and physical FPGA implementation of an optimized Elliptic Curve Cryptography (ECC) processor tailored for efficient encryption and decryption. Developed using Verilog HDL and verified using Xilinx Vivado, the architecture seamlessly

accelerates underlying modular arithmetic and point operations. Hardware-in-the-loop (HIL) validation on a Zynq-7000-based ZedBoard FPGA—driven by Virtual Input/output (VIO) and Integrated Logic Analyzer (ILA) cores—confirmed zero-deviation functional alignment between behavioral RTL simulation and on-chip execution, completing the entire cryptographic pipeline in precisely 2 clock cycles.

Physical implementation results demonstrate exceptional efficiency gains over the baseline architecture. The optimized core operates at 7.49 MHz while achieving a 95.0% reduction in total power consumption, drawing only 0.185 W compared to the 3.72 W baseline. Furthermore, the design achieves ultra-compact area optimization, reducing lookup table (LUT) utilization by 87.8% (down to 341) and flip-flop (FF) consumption by 98.5% (down to 144). These metrics validate that the proposed high-security, ultra-low-power processor is highly practical for resource-constrained edge devices and embedded IoT applications. Future research will explore scaling the architecture to larger prime fields and integrating side-channel attack countermeasures.

## VII. REFERENCE

- [1] K. Javeed, F. N. Sibai, and A. El-Moursy, "A Compact and Low Latency SPM Architecture for ECC Cryptosystems," *IEEE Access*, vol. 12, pp. 147552–147563, 2024, doi: 10.1109/ACCESS.2024.3472900.
- [2] N. S. Saju and S. K. N., "Design and Execution of Highly Adaptable Elliptic Curve Cryptographic Processor and Algorithm on FPGA Using Verilog HDL," in *Proc. Int. Conf. Communication, Control and Information Sciences (ICCISc)*, India, 2021, pp. 1–6, doi: 10.1109/ICCISc52257.2021.9484881.
- [3] A. R. Alharbi, M. M. Hazzazi, S. S. Jamal, A. Aljaedi, A. Aljuhni, and D. J. Alanazi, "DCryp-Unit: Crypto Hardware Accelerator Unit Design for Elliptic Curve Point Multiplication," *IEEE Access*, vol. 12, pp. 17823–17835, 2024, doi: 10.1109/ACCESS.2024.3358210.
- [4] K. Javeed, A. El-Moursy, and D. Gregg, "EC-Crypto: Highly Efficient Area-Delay Optimized Elliptic Curve Cryptography Processor," *IEEE Access*, vol. 11, pp. 56649–56662, 2023, doi: 10.1109/ACCESS.2023.3282781.
- [5] K. Javeed, A. El-Moursy, and D. Gregg, "E2CSM: Efficient FPGA Implementation of Elliptic Curve Scalar Multiplication over Generic Prime Field  $GF(p)$ ," *Journal of Supercomputing*, vol. 80, pp. 50–74, 2024, doi: 10.1007/s11227-023-05428-4.
- [6] M. M. Hazzazi, M. Rashid, S. S. Jamal, F. Alblehai, S. Nooh, and M. U. Rehman, "A Compact 3-Stage Pipelined Hardware Accelerator for Point Multiplication of Binary Elliptic Curves over  $GF(2^{233})$ ," *IEEE Access*, vol. 12, pp. 162510–162521, 2024, doi: 10.1109/ACCESS.2024.3487074.
- [7] Y. Li, "Area-Time-Efficient High-Radix Modular Inversion Algorithm and Hardware Implementation for ECC over Prime Fields," *Computers*, vol. 13, no. 10, Art. no. 265, 2024, doi: 10.3390/computers13100265.
- [8] M. Rashid, O. S. Sonbul, M. Y. I. Zia, M. Arif, A. Sajid, and S. S. Alotaibi, "Throughput/Area-Efficient Accelerator of Elliptic Curve Point Multiplication over  $GF(2^{233})$  on FPGA," *Electronics*, vol. 12, no. 17, Art. no. 3611, 2023, doi: 10.3390/electronics12173611.
- [9] V. Tanksale, "Efficient Elliptic Curve Diffie-Hellman Key Exchange for Resource-Constrained IoT Devices," *Electronics*, vol. 13, no. 18, Art. no. 3631, 2024, doi: 10.3390/electronics13183631.
- [10] Y. Li, "Hardware Implementations of Elliptic Curve Cryptography Using Shift-Sub Based Modular Multiplication Algorithms," *Cryptography*, vol. 7, no. 4, Art. no. 57, 2023, doi: 10.3390/cryptography7040057.
- [11] S. N., D. Reddy, and V. Sridhar, "Efficient Implementation of Scalar Multiplication for Elliptic Curve Cryptography Using Ancient Indian Vedic Mathematics over  $GF(p)$ ," *Int. J. Comput. Appl.*, vol. 49, no. 15, pp. 46–50, Jul. 2012, doi: 10.5120/7643-0730.
- [12] K. V. Patel, M. V. Shah, P. P. Prajapati, and A. J. Kshatriya, "Design and Implementation of



Effective Elliptic Curve Cryptography Accelerator Using Hardware/Software Co-Design on Zynq Board," *Int. J. Eng. Trends Technol.*, vol. 70, no. 8, pp. 327–335, 2022, doi: 10.14445/22315381/IJETT-V70I8P234.

[13] X. Pottier, T. de Ruijter, J. Bertels, W. Legiest, M. Van Beirendonck, and I. Verbauwhede, "OPTIMSM: FPGA Hardware Accelerator for Zero-Knowledge MSM," *IACR Trans. Cryptogr. Hardw. Embedded Syst.*, vol. 2025, no. 2, pp. 489–510, Mar. 2025, doi: 10.46586/tches.v2025.i2.489-510.

[14] D. Wang, Y. Lin, J. Hu, C. Zhang, and Q. Zhong, "FPGA Implementation for Elliptic Curve Cryptography Algorithm and Circuit with High Efficiency and Low Delay for IoT Applications," *Micromachines*, vol. 14, no. 5, Art. no. 1037, 2023, doi: 10.3390/mi14051037.