INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2026, VOL. 72, NO. 2, PP. 1-10
Manuscript received January 20, 2026; revised March, 2026. doi: 10.24425/ijet.2026.157912

Resource-efficient and high-speed PRINCE cipher for enhanced RFID security

Mahendra Shridhar Naik , Chaitra S N, Shashikiran S, and Shashikala K S

***Abstract*—Radio Frequency Identification (RFID) is a core technology inside the rapidly expanding Internet of Things (IoT), with several applications in various industries. However, RFID systems have significant privacy and security risks. To address these difficulties, this paper proposes an efficient RFID-based mutual authentication protocol (MAP) that uses the PRINCE lightweight cipher. The proposed PRINCE cipher utilizes a pipelined design supported by an enhanced control mechanism for round operations, resulting in significantly improved throughput and operating efficiency. This design's integrated hardware architecture allows it to carry out both encryption and decryption duties smoothly. The protocol uses five encryption and two decryption operations during the mutual authentication process between RFID tags and readers to provide a safe connection. Performance evaluations show that the suggested PRINCE cipher outperforms existing solutions in terms of area utilization and efficiency. Operating at 224 MHz, the architecture has a remarkable latency of 3.5 clock cycles and a throughput of 4.11 Gbps. When integrated into the RFID-based MAP, it achieves a throughput of 374.6 Mbps, an efficiency of 140.7 Kbps/Slice, and a processing latency of 0.355 µs. Such performance shows its effectiveness and suitability as a robust security solution for IoT environments.**

***Keywords*—Cryptography; Block Cipher; Authentication; PRINCE Cipher; Latency; Throughput**

I. INTRODUCTION

LIGHTWEIGHT and cost-effective devices are essential for enabling widespread and accessible distributed systems in the future. Technologies such as smart cards, Radio Frequency Identification (RFID), and Wireless Sensor Nodes (WSNs) constitute fundamental components of such pervasive networks. In RFID-based systems, intelligent card readers play a crucial role in safeguarding private medical data and authenticating users. Essential functions of any RFID system include object identification, tracking, alert generation, and user authentication. The three primary applications of RFID technology encompass supply chain management, item identification, and consumer product surveillance [1], [2].

RFID systems consist of two main components: readers (interrogators) and tags (labels). An RFID tag is a compact electronic device containing an antenna and an affordable integrated chip, detectable by nearby readers, sometimes even from considerable distances. Apart from the Electronic Product Code (EPC), RFID tags hold additional item-specific information stored in a centralized database. As a result, RFID technology is very useful for many purposes like managing inventory, tracking supplies, identifying vehicles, preventing fraud, monitoring animals, and checking the environment, which makes RFID systems great for use in many different areas.

However, significant privacy and security concerns associated with the communication link between readers and tags hinder the widespread adoption of RFID technology. Researchers have proposed various cryptographic techniques, such as mutual authentication protocols (MAPs), to secure this communication channel. Based on their computational complexity and supported operations, these MAPs are classified into four categories: simple, full-lightweight, fledged, and ultralightweight [6]. Lightweight techniques are further categorized according to their internal structures. These approaches effectively mitigate various attacks, including denial of service (DoS), counterfeiting, replay attacks, man-in-the-middle, intercepted verification, and internal threats involving tags [7].

Robust, undetectable, and resilient security measures must be integrated into RFID tags and readers to ensure secure authentication. Security schemes typically involve arrangements with databases, consumers, and servers, implementing strong protection mechanisms such as forward and backward undetectable approaches between tags and readers. Essential security features provided by such protection models include access control, detection of equipment manipulation, accessibility, non-repudiation, privacy, reliability, secure resetting, and authentication. To safely verify RFID tags with readers, we need good methods for key management, flexible lightweight techniques, strong encryption, and easy-to-use lightweight or ultra-lightweight encryption methods. For establishing reliable mutual authentication between tags and readers, ultra-lightweight block cipher techniques are preferable over stream cipher techniques [8], [9].

Ultra-lightweight solutions are specifically recommended for cost-sensitive RFID systems, which are widely adopted and expected to replace barcodes. The major constraint on these RFID tags is resource limitation, as they cannot accommodate advanced processors, extensive memory, or significant bandwidth due to cost restrictions [10].

This manuscript introduces an efficient RFID-based MAP employing the proposed PRINCE cipher designed explicitly for IoT applications. The proposed work delivers high throughput, superior execution time, and minimal chip area, making it well-suited for IoT environments. The contributions of this manuscript are summarized as follows:

•The proposed PRINCE cipher utilizes pipeline stages with

Mahendra Shridhar Naik and Chaitra S N are with NITTE (Deemed to be University), NMAM Institute of Technology (NMAMIT), Nitte. (e-mail: mahendrasnaik@gmail.com).

Shashikiran S. and Shashikala KS are with New Horizon College of Engineering, Bengaluru.

© The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY 4.0, https://creativecommons.org/licenses/by/4.0/), which permits use, distribution, and reproduction in any medium, provided that the Article is properly cited.

a round operation control mechanism to enhance throughput using minimal resources.

• Both encryption and decryption processes in the PRINCE algorithm share the same architecture, significantly reducing chip area and power consumption.

• The proposed RFID-based MAP demonstrates reduced execution time for tag and reader authentication, with updated seed values ensuring synchronization and secure communication.

• Comparative performance analysis highlights significant improvements to the proposed design over recent, existing approaches.

II. BACKGROUND

The recent works of authentication with security using different approaches for different applications are discussed in this section. Sidorov et al. [11] present the ultralightweight MA-based RFID protocol for blockchain-enabled devices. The MA-based RFID uses a database to provide the secured blockchain for the supply chain management system. The work discusses the ultralightweight protocol with a collision analysis of the design. The formal and security analyses are discussed in detail. The communication, storage, and computational costs are evaluated in detail. Lu et al. [12] describe the linear feedback shift register (LFSR) based lightweight Tripling (LT) MAP with an RFID Tag chip. The design uses an analogue front-end (AFE) module, Radio-frequency (RF) limiter, Voltage generator, and Amplitude shift keying (ASK) based modulators. The LT MAP uses the LFSR scheme for Reader and Tag authentication. The work realizes the randomness test to evaluate the pass rate and consumes 117 μW of Power. Hosseinzadeh et al. [13] discuss the robust adversary model for RFID MAP.

The adversary model is designed and deployed in server-mounted authentication protocol (SMAP) to improve the security features. Hosseinzadeh et al. [14] explain the enhanced authentication protocol (AP) for the RFID system. The Rabin-based AP is used to realize the security, formal and performance analysis. Zhu et al. [15] describe the secured RFID-based MAP for healthcare applications. The work reviews the existing security, weakness, and scalability issues and solves them with a new MAP approach. The Quadratic residue theorem is used for Secured MAP and realizes the security and performance analysis. The work discusses communication, storage, and computational costs in detail. Trinh et al. [16] present the lightweight block cipher-based MAP for IoT devices. The Craft-based lightweight block cipher is used as a security algorithm in AMP. The work realizes the informal and formal security analysis with a cost comparison. Naeem et al. [17] explain the RFID MAP using elliptic Curve Cryptography (ECC) with secured and scalable features for IoT applications. The work discusses the MAP and informal analysis in detail. The security evaluation is validated using the Proverif tool. The work obtains the computation cost for the Tag and Reader within 6.7114 milliseconds. Sharma et al. [18] discuss the ECC-based RFID MAP for the Internet of Vehicles (IoV). The MAP uses the setup, Tag, and server authentication stages. The Server and Tag authentication, scalability, availability, anonymity, and forward security features are discussed.

Zhong et al. [19] explain the MAP of RCIA protocol in RFID systems based on logic event theory (LET). The LET proof system, including formal foundation theory and axiom system, is discussed in detail with proofs. The vigorous MA is provided for Tag and Reader using RCIA protocol based on LETs. Wang et al. [20] describe the lightweight MAP for edge IoT nodes with Physical unclonable Function (PAF). The MAP has setup, registration, and authentication phases to realize the security analysis: the work analysis, security functions, and costs in detail. Cai et al. [21] present the RFID Tag/Mutual Authentication (MA) one step beyond the process. The work discusses unpredictable-based privacy notations. The storage and communication overheads are discussed with existing works in detail with improvements.

Noori et al. [22] discuss the ECC-based RFID MAP for IoT in healthcare applications. The work discusses security analyses like the men-in-middle, replay, MA, forward security, and data integrity. The work realizes Tag and Reader's computational, communication, and storage costs. Wei et al. [23] present the improved Secured authentication protocol (SAP) for lightweight RFID systems using ECC. The work analyzes different attacks and performance metrics with comparative discussion. Li et al. [24] indicates that PRINCE's hardware design can significantly enhance performance and resources. PRINCE optimizes components by using the fewest logic circuits possible. Both the freshly unfolded structure and the novel low-cost architecture reached new levels of efficiency for PRINCE hardware usage. Yli-Mäyry et al. [25] report an innovative SCL leakage in the most profound cycles of an unfolding hardware usage for block ciphers in a selected input attack scenario. SCL arises in the initial stage and manifests in the following inner cycles due to path-activating bias caused by the difference between two sequential inputs. It was found that such a rare SCL existed following a series of experiments employing fully unrolled PRINCE cipher technology built on an FPGA. Additionally, the machinery expense of the Threshold Implementation (TI) countermeasure—a preventative measure for the unrolled deployment—is evaluated and tested.

A general idea to transform a single-tag authentication technique into a secure burst tag validating method with a particular consolidation methodology is presented by Bákiewicz et al. [26]. The main goal is to build a cost-effective batch tag authentication method that provides a reasonable degree of confidentiality in the event of an inaccurate RFID reader. Apart from providing a prompt review of the Bloom filter's performance with other data structures, they also provide a sufficient appraisal of an algorithm that employs it as a consolidation strategy. Xu et al. [27] introduced a new, proper, compact RFID authentication method. It uses a variety of operating options to carry out secure authentication between several connected devices. By using a customizable database as the cloud server's storing process, the proposed solution reduces the possibility of secret data leakage while increasing the Server's effectiveness in retrieving Tag and reader identification. According to the safeguarding investigation, the method resists malicious attacks like proof, replaying, representation, and other attacks. Maurya et al. [28] describes an ultralightweight security mechanism that uses Maximum Distance Separable (MDS) methods and collection homomorphism. Employing group homomorphism properties, they offer a server lookup table that reduces query complexity and addresses scaling issues. They develop a bit-wise function for the connection by utilizing the features of MDS code. According to official and informal security investigations, the

suggested strategy is resilient to several attacks. They evaluate the suggested approach's privacy based on two common standards. The results of the research indicate that the recommended approach.

III. PRINCE CIPHER

Borghoff et al. initially developed the prince cipher in 2012 for pervasive computing applications [29]. The PRINCE cipher supports 64-bit data size and 128-bit key size. The conventional PRINCE cipher can perform encryption/decryption operations by changing the mode. The hardware architecture of the conventional PRINCE cipher is illustrated in Fig 1. The cipher contains two Add Round Key (ARK), two round constant (RC) additions, five regular rounds, one middle round, and five inverse round operations. Each normal round performs substitution box (Sbox) followed by Linear Layer (M), RC addition, and ARK operations. Similarly, each inverse round performs ARK, RC addition Inverse Linear Layer (M-1), and inverse Sbox operations.

The PRINCE cipher performs 11 rounds (normal/middle/inverse), ARKs, and RC addition operations twice. The 128-bit Key (K) is decomposed into two sub-keys (K0, K1) in a concatenation (||) manner (K = K0 || K1). The PRINCE cipher uses an additional whitening Key (Ks) in the Encryption and Decryption process to create confusion and diffusion. The generation and usage of the key and whitening key (Ks) for the Encryption and Decryption process are represented in Equations (1 and 2) as follows:

Mode = 1: Encryption: $K = (K_0 \,||\, K_1)$ and $K_s = (K_0[0] \,||\, K_0[63:2] \,||\, (K_0[1] \oplus K_0[63]))$ (1)

Mode = 0: Decryption: $K = (K_s \,||\, (K_1 \oplus \alpha))$ and $K_0 = (K_s[0] \,||\, K_s[63:2] \,||\, (K_s[1] \oplus K_s[63]))$ (2)

The Encryption process uses Key (K0) for the first ARK operation, Key (K1) for round operations, and Whitening Key (Ks) for the last ARK operation. Similarly, the Decryption process uses a whitening key (Ks) for the first ARK operation, a Key (K0) for round operations, and a Key (K1 ⊕ α) for the last ARK operation. The alpha (α) is a 64-bit constant value (α = c0ac29b7c97c50dd) and ⊕ is an XOR operation. The ARK performs a simple XOR operation with corresponding state input and key. The RC Addition performs a simple XOR operation with corresponding state input and round constant (RC) values. The five normal and inverse rounds use five different RC additions, RC1 to RC5 and RC6 to RC10, respectively, to perform encryption/decryption operations. The RC values in hexadecimal notation are tabulated in Table I.

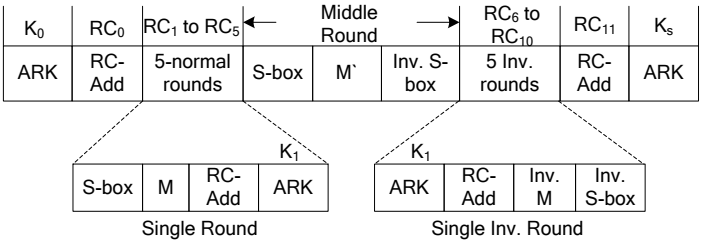

Fig 1 Conventional PRINCE Cipher

The S-box uses a 4-bit data value as input and is replaced with the corresponding S-box value in the Encryption process. The S-box operation repeated 16 times to construct 64-bit S-box output. The inverse S-box output is the same as the S-box operation and is the Decryption process. The S-box and Inverse S-box for the Encryption and Decryption process are tabulated in Table II [30]. The Linear (M) and M` layers contain the 64-bit state input multiplied with 64 x 64 matrix M or M`. The M` layer is used only in the middle round and is constructed based on the alpha (α) reflection property. The M layer matrix uses shift rows to generate M and M^{-1} mapping, represented in Fig 2. The 16 nibbles are used; each is 4-bit in M and M^{-1} mapping. The PRINCE cipher produces the encryption output by performing the XOR operation using the 11th round RC addition output with whitening key (K_s) in the last ARK operation. Similarly, The PRINCE decipher obtains the decryption output by performing the XOR operation using the 11th round RC addition output with Key (K_0) in the last ARK operation.

TABLE I
ROUND CONSTANT (RC) VALUES FOR PRINCE CIPHER

RC Number	RC Value	RC Number	RC Value
RC_0	0000000000000000	RC_6	7EF84F78FD955CB1
RC_1	13198A2E03707344	RC_7	85840851F1AC43AA
RC_2	A4093822299F31D0	RC_8	C882D32F25323C54
RC_3	082EFA98EC4E6E89	RC_9	64A51195E0E3610D
RC_4	452821E638D01377	RC_{10}	D3b5A399CA0C2399
RC_5	BE5466CF34E90C6C	RC_{11}	C0AC29B7C97C50DD

TABLE II
S-BOX AND INVERSE SBOX FOR PRINCE CIPHER

In	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
S-box out	B	F	3	2	A	C	9	1	6	7	8	0	E	5	D	4
Inv. S-box out	B	7	3	2	F	D	8	9	A	6	4	0	5	E	C	1

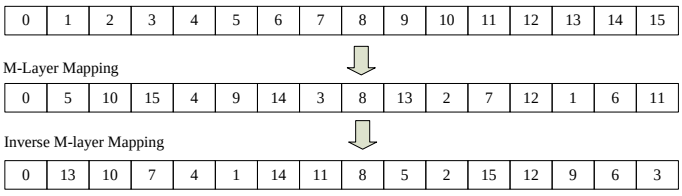

Fig 2 M-layer and Inverse M-Layer mapping

IV. PROPOSED WORK

The proposed PRINCE cipher is constructed for secured mutual authentication of RFID tag and Reader and is discussed in this section. The proposed cipher offers better Latency and throughput for low-resource devices like RFID devices on hardware platforms. The cipher is outperforming current block ciphers with improvement in performance parameters.

4.1 Proposed PRINCE Cipher

The proposed PRINCE cipher is designed using a Datapath processing unit with pipeline stages for round operations and is illustrated in Fig 3. The cipher has four stages, namely: (1) Key generation, (2) Register updation unit, (3) Datapath processing

unit with pipeline stages for state logic updation and (4) finite state machine (FSM) controller for Datapath updation. The algorithmic flow of the proposed PRINCE cipher is illustrated in Algorithm 1. The Key generation unit was originally 128-bit and decomposed into sub-keys (K_0, K_1). K_0 uses the MSB side of 64 bits [127:64], and K_1 uses the LSB side of 64 bits [63:0]. These subkeys are used further as Datapath or round operation and Register updation processes. The whitening Key (K_s) is used in round operations for cipher and decipher processes.

Register updation unit: The register updation unit has mainly four different sets of registers, namely, The key registers (K_{r0}, K_{r1}, K_{rs1}), round registers (R_{3_reg}, R_{8_reg}, M_{R_reg}), state register (st_reg), and control register (c_reg). This register updation unit updates the corresponding register sets based on cipher operation. The PRINCE cipher operation starts after resetting all the register sets. After resetting, the register sets like round registers (R_{3_reg}, R_{8_reg}, M_{R_reg}) are updated with new round values (R_{3_new}, R_{8_new}, M_{R_new}). Similarly, the state enable signal (st_en) is activated to update the state register (st_reg) with a new state register value (st_new). The key enable signal (k_en) is activated to update the key registers (K_{r0}, K_{r1}, K_{rs}) by new key values (K_{n0}, K_{n1}, K_{ns}). The control enable signal (c_en) is activated to replace with control register (c_reg) with a new control register value (c_new). These four register sets are used further in the Datapath unit and FSM controller for cipher and decipher operations.

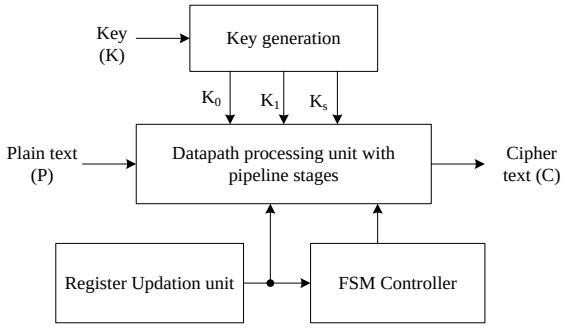

Fig 3 Proposed PRINCE Cipher

Algorithm-1: Operational flow of proposed PRINCE Cipher

Input: Plain text (P), 128-bit Key (K);
Output: Cipher text (C), done;
Define: All the control and state registers

1. Register Updation:
   a. **if** (rst = 1), **then** reset all the registers (key, round, state) else update round registers (R_{3_reg}, R_{8_reg}, M_{R_reg}) with new values (R_{3_new}, R_{8_new}, M_{R_new});
   b. **if** (st_en = 1) **then** update state register (st_reg) with new value (st_new);
   c. **if** (k_en = 1) **then** update key register (K_{r0}, K_{r1}, K_{rs}) with new values (K_{n0}, K_{n1}, K_{ns});
   d. **if** (c_en = 1) **then** update control register (c_reg) with new value (c_new);
2. Data path processing (Pipeline stages):
   a. inp = st_reg $\oplus$ K_{r0}
   b. Generation of Rounds (R_0 to R_{11}) as per Fig 4.
   c. Out = $R_{11} \oplus K_{rs}$;
   d. **if** (i_s = 1) then st_en = 1; k_en = 1; st_new = Plain text (P); //Initialization state
      - If (mode = 1) **then** update the Keys (K_{n0}, K_{n1}, K_{ns}) as per Equation (3);
        else
      - update the Keys (K_{n0}, K_{n1}, K_{ns}) as per Equation (4);
   e. **if** (u_s = 1) **then** st_new = Out; //Updation state;
3. FSM Controller:
   a. Define: five internal states (idle, s0, s1, s2, and su) for control register
   b. Initialize: Reset the registers (i_s, u_s, c_new, c_en) = 0;
   c. **idle state**: i_s = 1; c_new = s0; c_en = 1;
   d. **s0 state:** c_new = s1; c_en = 1;
   e. **s1 state:** c_new = s2; c_en = 1;
   f. **s2 state:** c_new = su; c_en = 1;
   g. **su state:** c_new = idle; u_s = 1; c_en = 1, done = 1;
4. End

Datapath processing unit: This processing unit performs the round operations of the PRINCE cipher. The cipher performs 11 rounds (R) of operation, including five normal, one middle round and five inverse rounds. The state register (st_reg) performs the XOR operation with the key register (K_{r0}) to start the round operation, and their results are stored in a temporary register (inp). The 11 rounds perform the round operation in a pipeline manner, and its flow is shown in Fig 4. The initial round output (R_0) is generated by performing the round operation using a temporary register (inp) and key register (K_{r1}). Next, perform the five normal rounds of operation using the key register (K_{r1}) and RCs (RC_1 to RC_5) and store these round operation outcomes in registers like (R_1, R_2, R_{3_new}, R_4 and R_5). The middle round (M_R) is designed using S-box, M` layer, and inverse-S-box operation. The outcomes of the middle round operation are stored in M_{R_new}. The five inverse rounds of operations are performed using the key register (K_{r1}) and RCs (RC_6 to RC_{10}), storing these round operation outcomes in registers like (R_6, R_7, R_{8_new}, R_9 and R_{10}). The last round output (R_{11}) is obtained by performing the round operation using the register (R_{10}) and key register (K_{r1}). The final pipeline-stage output (Out) is obtained by performing the XOR operation using the last round output (R_{11}) and whitening key register (K_{rs}). The RCs from Table I, S-box and Inverse-Sbox from Table II, M-layer and Inverse M-Layer mapping from Fig 2 are used for the proposed round operation. The Datapath processing unit performs the key and state updation using in state modes like initialization and updation state. Once the initialization state (i_s) is active, then the original Plain text (P) is stored in the new state register (st_new) by enabling the state enable (st_en) and key enable (k_en) signals. During this time, the new keys are updated based on the mode for Encryption and decryption and are represented using Equation (3 and 4) as below.

mode = 1: Encryption: $K = (K_{n0} \,||\, K_{n1})$ and $K_{ns} = (K_{n0}[0] \,||\, K_{n0}[63:2] \,||\, (K_{n0}[1] \oplus K_{n0}[63]))$ (3)

mode = 0: Decryption: $K = (K_{ns} \,||\, (K_{n1} \oplus \alpha))$ and $K_{n0} = (K_{ns}[0] \,||\, K_{ns}[63:2] \,||\, (K_{ns}[1] \oplus K_{ns}[63]))$ (4)

The updation state (u_s) is activated only after all the round operations are completed and later updates the final pipeline - stage output (Out) as new state register output (st_new).

FSM Controller: The controller has five states: the idle state, followed by three pipeline states (s0, s1 and s2) and the last update state (su). These states are part of the control register to initiate and update the Datapath processing unit. The idle state activates the initialization state (i_s) by enabling the control enable signal (c_en). The s0/s1/s2 states are updated one after the other during pipeline stages with active c_en = 1. Lastly, the update state (su) activates the updation state (u_s) by enabling the c_en signal. After updation, done signal is activated, which indicates that either Encryption or decryption is completed based on modes of operation.

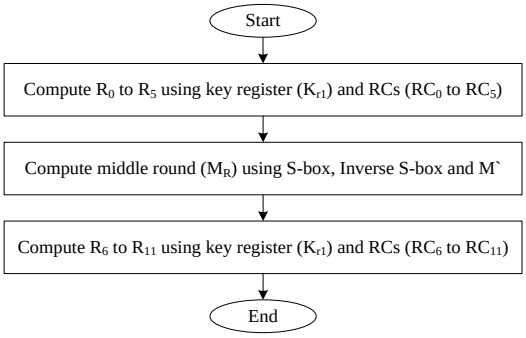

Fig 4 Flow of Pipeline stages in Datapath processing unit

4.2 RFID-based MAP using PRINCE Cipher

An RFID-based MAP using a proposed PRINCE cipher for IoT applications is discussed in this section. The proposed RFID-based MAP contains a Server database, Reader, and Tag information with multiple Encryption and Decryption processes. The parameters and descriptions used in RFID-based MAP are tabulated in Table III. A few of the assumptions are mandatory to perform authentication between Tag and Reader.

TABLE III.
PARAMETERS AND THEIR DESCRIPTIONS USED IN RFID MA PROTOCOL

Parameters	Description	Parameters	Description
K	128-bit Key	S_{C1}, S_{C2}	Server-side Ciphers-1,2
E	PRINCE encryption	S_{D1}, S_{D2}	Server-side Deciphers-1,2
D	PRINCE decryption	R_C	Reader Cipher after Encryption
S	Seed Value	T_C	Tag Cipher after Encryption
ID_T	Tag ID	T_R	Tag Response after Encryption
ID_R	Reader ID	US_T	Updated Seed value at tag-side
ID_S	Server ID	US_S	Updated Seed value at Server-side

The database must know the 128-bit key (K) of the PRINCE cipher, 64-bit Seed (S), 64-bit Tag Identification (ID_T), and 64-bit Reader ID (ID_R). The Reader must know the 128-bit Key and Reader ID. The Tag should have information about the 128-bit Key, Tag Identification (ID_T), and 64-bit Seed (S). The RFID-based MAP process is illustrated in Fig 5. The operation of the RFID-based MAP using the proposed PRINCE cipher is discussed as follows:

- *Query:* The server database sends the query to the Tag via Reader. First, perform encryption (E) operation using K and ID_R to generate Server-side Cipher-1 (SC_1). The SC_1 data is passed to the Reader. The Reader performs a decryption (D) operation for SC_1 data and generates Server-side Decipher-1 (SD_1). Perform XOR operation between SD_1 and ID_R to generate the Reader Cipher (R_C) value. The R_C data acts as required data to Tag.
- *Reader to Tag Authentication Process*: Initially, Tag generates Tag cipher (T_C) data by performing encryption operation using Seed value (E(S)). If the T_C matches the received R_C, then update the seed value (US_T) by performing ($R_C \oplus K$), and the Reader is authenticated at the Tag side. If not, the matches Tag has to wait until further query. The Tag has to Respond to the Reader (T_R) by performing encryption operation E ($US_T \oplus ID_T$).
- *Tag to Reader Authentication Process*: The Reader receives the Tag response (T_R) and forwards it to the server database for reader authentication. The database performs E (S) and D (T_R) to generate the SC_2 and SD_2. These values SC_2 and SD_2 are used to perform the XOR operation with a Key to generate the ID at the server side (ID_S) and used further to extract the Tag's ID. If the ID_S and ID_T match, then update the server-side (US_S) seed value by performing ($SC_2 \oplus K$), and the Tag is authenticated on the Server side.
- *Synchronization between Tag and Reader*: If the updated Seed at Tag-side (US_T) value matches the updated Seed at server-side (US_S), the synchronization between Tag and Reader is successful and ready for secured data communication.

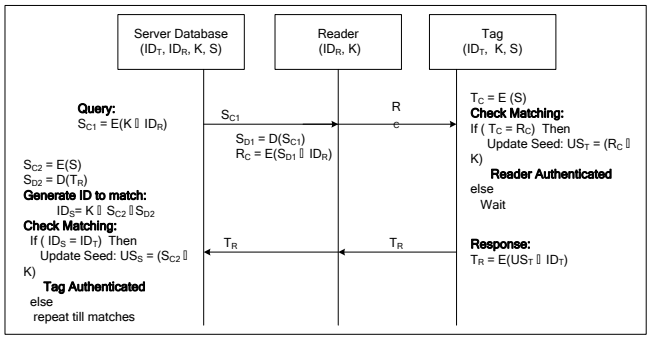

Fig 5 RFID – Mutual Authentication Protocol

The detailed results of the suggested PRINCE Cipher and RFID-MA protocol using PRINCE are discussed in this section. The design modules are constructed using Verilog HDL on the Xilinx ISE platform and synthesized using Artix-7 FPGA (Device: XC7A100T-5CSG324). The mentor graphics-based Modelsim simulator verifies and visualizes the simulation waveform. The chip area utilization contains slices, Lookup Tables (LUTs), LUT- Flip-flops (FFs), and Power. The parameters like Latency in terms of clock cycles (CC), Throughput (Gbps), and Hardware efficiency (Mbps/slice) are considered for performance realization.

The simulation results of the RFID-MA using the PRINCE cipher are illustrated in Fig 6. The global clock (clk) is activated at 100 MHz with active low reset (rst) to start the RFID-MA

process. Define 128-bit key, 64-bit Seed, and ID (Tag and Reader) values. Based on the RFID-MA protocol, If the values of the Reader and Tag ciphers match, then the Reader is authenticated (Reader_Auth). Similarly, after the encryption and decryption process on the server side, the obtained ID (ID_Match) matches with Tag ID (ID_Tag) then Tag is authenticated (Tag_Auth). The synchronization of both Tag and Reader is achieved only after seed updation. Once the updated seed values match Tag and Reader, RFID authentication is successful with synchronization.

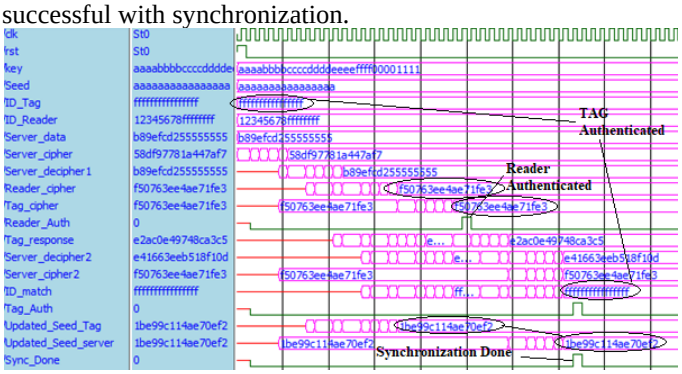

Fig 6 Simulation Results of RFID-MA using PRINCE *Cipher*

6.1 Performance analysis

The resource usage of Conventional and proposed PRINCE cipher on Artix-7 FPGA is tabulated in Table IV. The suggested or proposed PRINCE cipher improves the Slices and LUTs by around 55 % and 11.64 % to the conventional cipher approach. The proposed approach uses pipeline stages with suitable control mechanisms for round operation, which minimizes the hardware resources and improves the performance metrics than the conventional cipher approach. The operating frequency of the proposed cipher is 224.4 MHz and improved by around 5.5 % than the conventional cipher approach due to the concurrent mechanism. The total Power of the proposed cipher is 190 mW and improved by around 24.6 % than the conventional cipher approach. The round operation is performed sequentially in a conventional cipher, which uses more LUTs and Power on a chip than the proposed PRINCE cipher. The proposed PRINCE Cipher uses only 3.5 CC as Latency, achieving a throughput of 4.11 Gbps, with a hardware efficiency of 15.8 Mbps/Slice. The proposed cipher improves the Latency by around 12.5 %, throughput by around 5.5 % and efficiency by 57.5 % more than the conventional approach.

TABLE IV.
RESOURCE USAGE AND IMPROVEMENTS OF CONVENTIONAL AND PROPOSED PRINCE CIPHER ON ARTIX-7 FPGA

Resources	Conventional Prince	Proposed Prince	% Improvements
Slices	579	260	55.09
LUTs	1683	1487	11.64
Max. Frequency (MHz)	212.067	224.467	5.52
Dynamic Power (mW)	170	98	42.3
Total Power (mW)	252	190	24.6
Latency (CC)	4	3.5	12.5
Throughput (Gbps)	3.88	4.11	5.5
Efficiency (Mbps/Slices)	6.7	15.8	57.5

The resource usage of the suggested RFID-based MAP using PRINCE cipher on Artix-7 FPGA is tabulated in Table V. The RFID-based MAP utilizes slices of 2 % and LUTs of 18 %, operates at 207.8 MHz, and consumes a total power of 891 mW. The RFID-based MAP performance five and two times of encryption and decryption operations respectively, to obtain the secured authentication between Tag and Reader. The PRFID-MA protocol uses 35.5 CC as Latency and achieves a throughput of 374.6 Mbps, with an efficiency of 140.78 Kbps/Slice. The RFID-MA process with synchronization between Tag and Reader is completed with a processing time of 0.355 µs.

TABLE V.
RESOURCE USAGE OF PROPOSED RFID-BASED MAP USING PRINCE CIPHER ON ARTIX-7 FPGA

Resources	Utilization
Slices	2661
LUTs	12029
Max. Frequency (MHz)	207.801
Dynamic Power (mW)	806
Total Power (mW)	891
Latency (CC)	35.5
Throughput (Mbps)	374.62
Efficiency (Kbps/Slices)	140.78
Execution Time (µs)	0.355

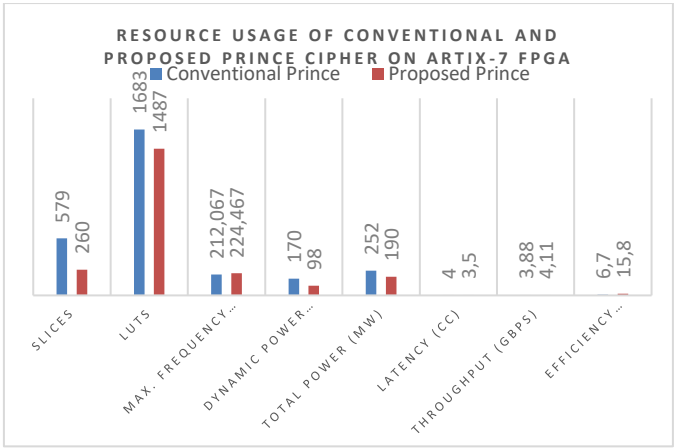

Fig 7 Resource usage of Conventional and Proposed PRINCE cipher on Artix-7 FPGA

Fig 7 presents a comparative analysis of the resource usage and performance metrics between the conventional and proposed PRINCE cipher implementations on the Artix-7 FPGA platform. The proposed architecture demonstrates significant improvements across all evaluated parameters. It reduces the slice usage from 579 to 260 and LUTs from 1683 to 1487, highlighting its area-efficient design. Additionally, the proposed cipher operates at a higher maximum frequency of 224.467 MHz compared to 212.067 MHz for the conventional version. Power efficiency is also improved, with dynamic power consumption dropping from 170 mW to 98 mW and total power from 252 mW to 190 mW. In terms of performance, the proposed cipher achieves a lower latency of 3.5 clock cycles versus 4 cycles and delivers a higher throughput of 4.11 Gbps compared to 3.88 Gbps. Notably, its efficiency, measured in Mbps per slice, is significantly enhanced, rising from 6.7

Mbps/Slice in the conventional design to 15.8 Mbps/Slice. These results confirm the proposed PRINCE cipher's superiority in terms of area, power, latency, throughput, and efficiency, making it well-suited for secure, lightweight applications in RFID and IoT environments.

6.2 Performance Comparison with existing PRINCE ciphers

The performance comparison of the proposed PRINCE cipher with existing PRINCE ciphers is tabulated in Table VI. The FPGA device, chip area (slices), obtained Frequency, Latency, Throughput (Gbps), and hardware efficiency parameters are considered for performance comparison. The Virtex-4/6 and Kintex-7 FPGA devices are considered for comparing existing PRINCE Cipher designs with the proposed approach. The PRINCE cipher [31] is designed and implemented on Virtex-4 and Virtex-6 FPGAs. The PRINCE cipher [30] is processed sequentially for round operation and is sequential. The design consumes more resources (Slices and LUTs) and operates at a lower frequency than the proposed PRINCE cipher. The Latency of PRINCE [31] is 1 CC and is better than the proposed PRINCE; however, it lags with the throughput and efficiency than the proposed PRINCE cipher. The low-cost PRINCE cipher [32] is implemented on Virtex-4. The internal round operation structure is complex, consuming more resources and using more CC (11) as Latency.

Regarding performance indicators on the identical Virtex-4 and Virtex-6 FPGAs, the suggested PRINCE cipher functioned better than the PRINCE cipher [32]. The PRINCE cipher [32] is designed for quantum cryptography on Virtex-4 and Kintex-7, which offers 1 CC as a latency. The designed PRINCE [33] cipher is constructed as a traditional approach, and the round process employed in the PRINCE cipher is more complicated and utilizes more resources on hardware than the suggested approach. The suggested PRINCE cipher uses minimum hardware and better performance metrics than the PRINCE cipher [33].

TABLE VI.
PERFORMANCE COMPARISON OF PROPOSED PRINCE CIPHER WITH EXISTING PRINCE CIPHERS

Designs	FPGA Device	Slices	Max. Frequency (MHz)	Latency (CC)	Throughput (Gbps)	Efficiency (Mbps/Slices)
Ref [30]	Virtex-4	956	31.76	1	2.032	2.126
Ref [31]	Virtex-4	1305	136.036	11	2.081	1.595
Ref [32]	Virtex-4	1026	31.72	1	2.029	1.978
This work	Virtex-4	896	137.3	3.5	2.52	2.81
Ref [30]	Virtex-6	482	65.38	1	4.18	8.68
Ref [31]	Virtex-6	831	172.27	11	2.71	3.25
This work	Virtex-6	260	230.04	3.5	4.21	16.17
Ref [32]	Kintex-7	539	61.42	1	3.931	7.29
This work	Kintex-7	260	285.17	3.5	5.22	20.05

Fig 8 illustrates the performance comparison of the proposed PRINCE cipher with various existing PRINCE cipher implementations across different FPGA platforms, including Virtex-4, Virtex-6, and Kintex-7. The comparison covers key metrics such as the number of slices used, maximum operating frequency, latency, throughput, and efficiency. The proposed work, implemented on the Kintex-7 FPGA, exhibits a significant reduction in resource utilization, particularly in terms of slices, compared to other designs on similar and older FPGA platforms. It also achieves a higher maximum frequency, contributing to improved throughput. In terms of latency, the proposed design maintains competitive performance while substantially enhancing throughput (measured in Gbps) and efficiency (measured in Mbps per slice). This demonstrates that the proposed PRINCE cipher outperforms prior implementations by achieving better speed and computational efficiency with lower hardware overhead, making it highly suitable for lightweight and secure applications in resource-constrained environments such as IoT and RFID systems.

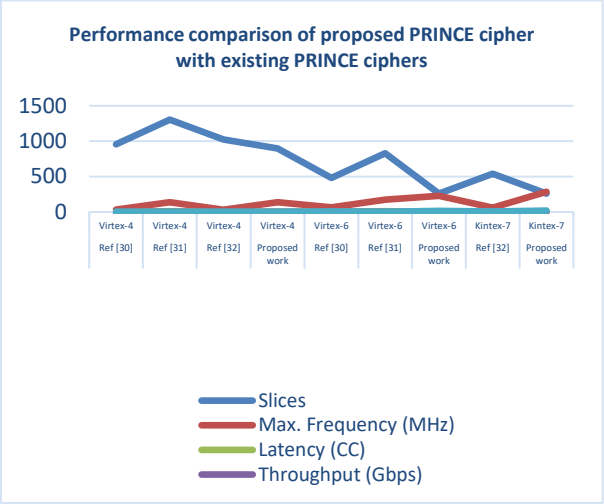

Fig 8 Performance comparison of proposed PRINCE cipher with existing PRINCE ciphers

6.3 Performance Comparison with existing lightweight block ciphers

Table VII illustrates the performance comparison between the suggested PRINCE cipher and current lightweight BC techniques on different FPGAs. With 128-bit keys, the SIMON and SPECK encryption algorithms [34] are primarily made for the IoT gadgets running Spartan-3 FPGA. The two versions are built with a key scheduling strategy and conventional techniques. The SPECK and SIMON architectures are constructed using the Feistel-based framework. Nevertheless, round key creation in key management is carried out sequentially, resulting in increased latency and chip area consumption. The suggested PRINCE cipher operates at a lower frequency than the SIMON and SPECK ciphers [34] because of sequential processing in key management. On the identical Spartan-3 FPGA, the suggested PRINCE cipher outperforms SIMON and SPECK encryption algorithms [33] regarding throughput, efficiency, and slice count. The Spartan-3 FPGA is used to build the bit-slicing method of the lightweight RECTANGLE encryption [34]. The cipher reduces the number of resources on the device by using fundamental computations

for scheduling and binding, yet, the control structure is intricate to produce the cipher results. Regarding the identical Spartan-3 FPGA performance indicators, the suggested PRINCE cipher functioned better than the RECTANGLE encryption [35].

The FPGAs Artix-7 and Virtex-6 are used for implementing the LED cipher [36]. This cipher operates using parallel construction, which reduces hardware consumption and provides a higher frequency. However, the round process employed in the LED cipher is more complicated than the PRINCE cipher. However, the suggested PRINCE cipher outperforms the LED [37] Encryption regarding Latency, throughput, and efficiency on both Artix-7 and Virtex-6 FPGAs. The Virtex-6 FPGA is used in the iterative construction of the PRESENT cipher [38]. More CCs are used in the data path and key management processes to produce the cipher result. The performance measurements are impacted by the key management procedure's use of two S-Boxes and a round counter function for every round in the PRESENT algorithm, which utilizes extra CCs (Latency). Regarding performance indicators on the identical Virtex-6 FPGA, the suggested PRINCE cipher functioned better than the PRESENT encryption [38]. The throughput and efficiency metrics of the XTEA cipher [39] are impacted by the 128 clock cycles required to finish the data encryption with the key management process. With just 3.5 CCs, the suggested PRINCE cipher provides a higher throughput of 4.11 Gbps on Artix-7 FPGA. The Artix-7 FPGA employs a 128-bit key in the PRESENT cipher [40]. Iteratively, the data and key management are carried out concurrently. The PRESENT Cipher's key update method is intricate. Nevertheless, the suggested PRINCE cipher does not have a key scheduling strategy. Only key decomposition procedures are used by the PRINCE cipher for round execution and mode choosing. Compared to the PRESENT-128 algorithm, the suggested PRINCE cipher yields superior outcomes [40].

TABLE VII
PERFORMANCE COMPARISON OF PROPOSED PRINCE CIPHER WITH EXISTING LIGHTWEIGHT BCS

Designs	FPGA Device	Slices	Max. Frequency (MHz)	Latency (CC)	Throughput (Mbps)	Efficiency (Mbps/Slices)
SPECK [33]	Spartan-3	2014	191	52	634	0.315
SIMON [33]	Spartan-3	1751	344	32	770	0.497
RECTANGLE [34]	Spartan-3	483	149	13	773	0.6
Proposed work	Spartan-3	896	74	3.5	1353.2	1.51
LED [35]	Virtex-6	133	482.369	16	1,929.48	14.51
PRESENT [36]	Virtex-6	201	211	136	99.13	0.493
Proposed work	Virtex-6	260	230.04	3.5	4210	16.17
XTEA [37]	Artix-7	316	264	128	80.43	0.34
LED [35]	Artix-7	133	407.15	16	1,628.60	12.25
PRESENT [38]	Artix-7	455	410	32	822	1.8
Proposed work	Artix-7	260	224.467	3.5	4110	15.8

Fig 9 presents a comparative evaluation of the proposed PRINCE cipher against various existing lightweight block ciphers (BCs), including SPECK, MIBS, PRESENT, LED, and KLEIN, implemented on different FPGA platforms such as Spartan-3, Virtex-6, and Artix-7. The comparison is based on critical performance parameters: number of slices, maximum frequency (MHz), latency (clock cycles), throughput (Mbps), and efficiency (Mbps/Slice).

The proposed PRINCE cipher, implemented on Artix-7, demonstrates clear advantages in terms of throughput and efficiency. It achieves the highest throughput, exceeding 4300 Mbps, significantly outperforming all other lightweight ciphers across platforms. Moreover, it maintains a high efficiency, measured in Mbps per slice, indicating optimal use of hardware resources for performance gain.

In contrast, many of the existing lightweight BCs particularly those implemented on older FPGAs like Spartan 3 consume more slices and deliver lower throughput and efficiency. While a few ciphers show competitive frequency and latency characteristics, they fall short in terms of overall performance balance and hardware utilization.

Overall, the fig validates that the proposed PRINCE cipher offers superior performance with a reduced area footprint, higher data rate, and better resource efficiency, making it a compelling choice for low-resource, high-speed applications such as secure RFID and IoT communication.

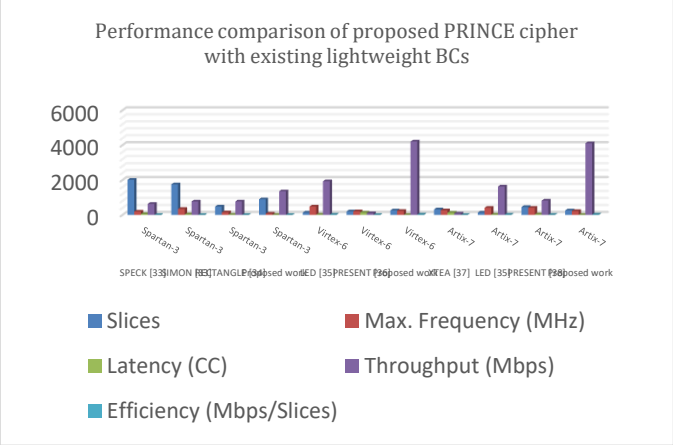

Fig 9 Performance comparison of proposed PRINCE cipher with existing lightweight BC

6.4 Performance Comparison with existing RFID-based systems

The performance comparison of the proposed RFID-based MAP with existing RFID works is tabulated in Table VIII. The comparison includes the FPGA device used in work; the cipher used in the RFID design and performance metrics. Implementing GEN2 protocol [41] for secure authentication of RFID devices using three different ciphers: XTEA, PRESENT and Hummingbird (HB). The GEN2 module for tag identification uses standard detection and response units with a state machine. In a Request-challenge-response approach, the authentication protocol performs encryption and decryption operations twice in cipher block chaining (CBC) mode. The RFID approach [41] using XTEA, PRESENT and HB ciphers is implemented on Spartan-3 FPGA. This approach uses more resources due to CBC modes of operation and the complex challenge-response approach in the RFID process. The proposed approach uses simple MAP for secure data transfer and offers better performance outcomes than an RFID-based

system [41]. The RFID -a system using the XTEA approach [42] uses minimal chip area; however, it uses more Latency and processing time to complete the authentication process. The 128 CCs are needed to complete the XTEA, and 1155 CC are needed to finish the RFID process, affecting the throughput and efficiency measures [42]. The proposed RFID -system using PRINCE cipher offers a greater throughput of 329.46 Mbps on Sparatn-3 FPGA with only 35.5 CCs. The RFID authentication process using PRESENT cipher [43] is designed on Sparatn-3 FPGA. This work offers minimal area than the suggested work, but lags with performance metrics due to Latency and processing time. The suggested PRINCE cipher yields superior outcomes to the RFID-based PRESENT [43]. Overall, the proposed PRINCE cipher and RFID-based approach offer minimal resources and better performance metrics (Latency, throughput and efficiency) than existing PRINCE, other BCs and other RFID-based approaches.

TABLE VIII
PERFORMANCE COMPARISON OF PROPOSED RFID-BASED MAP WITH EXISTING RFID WORKS

Designs	Ref [41]	Ref [41]	Ref [41]	Ref [42]	Ref [43]	Proposed work
FPGA	Spartan-3	Spartan-3	Spartan-3	Spartan-3	Spartan-3	Spartan-3
Cipher	XTEA	PRESENT	HB	XTEA	PRESENT	PRINCE
Slices	22.55 K	22.32 K	22.29 K	1.22K	1.83K	7.2 K
LUTs	43.32 K	42.84 K	41.90 K	1.136 K	3.58 K	13.95 K
LUT-FFs	2.603 K	2.553 K	2.411 K	2.234 K	0.996K	2.76 K
Frequency (MHz)	60.6	59.16	40.1	125.51	192.6	182.75
Latency (CC)	101	133	133	1155	1285	35.5
Throughput (Mbps)	38.4	28.46	19.29	6.96	9.59	329.46
Efficiency (Kbps/Slice)	1.7	1.28	0.87	5.74	5.24	45.76
Execution Time (µs)	1.01	1.33	1.33	11.55	12.85	0.355

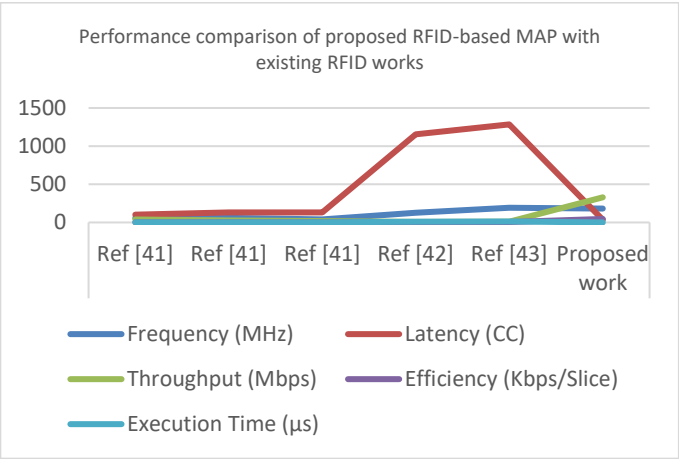

Fig 10 Performance comparison of proposed RFID-based MAP with existing RFID works

CONCLUSION

The manuscript presents a secure RFID-based Mutual Authentication Protocol (MAP) employing the proposed PRINCE cipher. The design is developed using Verilog-HDL within the Xilinx ISE environment and implemented on the Artix-7 FPGA platform. The lightweight PRINCE cipher performs both encryption and decryption operations based on the mode, leveraging pipelined stages in the round function to enhance throughput and efficiency. The proposed RFID-based MAP ensures secure and synchronized communication between the Tag and Reader by utilizing updated seed values on both sides. The implementation demonstrates efficient resource utilization, occupying only 2% of the chip area and operating at a frequency of 207.8 MHz. It achieves a high data rate of 374.62 Mbps and an efficiency of 140.78 Kbps/Slice. Furthermore, the authentication process between the Tag and Reader is completed in just 0.355 µs. Comparative analysis reveals that the proposed PRINCE cipher and MAP outperform existing implementations of PRINCE and other block ciphers (BCs) in terms of latency, throughput, and resource efficiency. Future work will focus on a detailed security analysis of the proposed MAP, along with further performance optimization and validation under various attack models.

REFERENCES

[1] A. Jeng, Li-Chung Chang, and Huang-Kuan Ho, "Survey and remedy of mutual authentication protocols for RFID system," in 2008 International Conference on Machine Learning and Cybernetics, IEEE, Jul. 2008, pp. 3361–3366. https://doi.org/10.1109/ICMLC.2008.4620985

[2] S. M. Mohsin, I. A. Khan, S. M. Abrar Akber, S. Shamshirband, and A. T. Chronopoulos, "Exploring the RFID mutual authentication domain," International Journal of Computers and Applications, vol. 43, no. 2, pp. 127–141, Feb. 2021, https://doi.org/10.1080/1206212X.2018.1533614

[3] Z. Bilal and K. Martin, "Ultra-lightweight Mutual Authentication Protocols: Weaknesses and Countermeasures," in 2013 International Conference on Availability, Reliability and Security, IEEE, Sep. 2013, pp. 304–309. https://doi.org/10.1109/ARES.2013.41

[4] S. Leng, M. Tang, X. Jiang, Z. Zhang, and M. H. Lee, "An improved mutual authentication scheme complaint to EPC Class-1 Generation-2 standard," in 2011 International Conference on Consumer Electronics, Communications and Networks (CECNet), IEEE, Apr. 2011, pp. 3933–3937. https://doi.org/10.1109/CECNET.2011.5768269

[5] S. Han, V. Potdar, and E. Chang, "Mutual Authentication Protocol for RFID Tags Based on Synchronized Secret Information with Monitor," in Computational Science and Its Applications – ICCSA 2007, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 227–238. https://doi.org/10.1007/978-3-540-74484-9_20

[6] Y.-J. Huang, C.-H. Jiang, H.-H. Wu, Y.-H. Hong, and K.-J. Liu, "Mutual Authentication Protocol for RFID System," in 2011 14th IEEE International Conference on Computational Science and Engineering, IEEE, Aug. 2011, pp. 73–80. https://doi.org/10.1109/CSE.2011.27

[7] D. W. Engels, X. Fan, G. Gong, H. Hu, and E. M. Smith, "Ultra-Lightweight Cryptography for Low-Cost RFID Tags : Hummingbird Algorithm and Protocol," 2009. [Online]. Available: https://api.semanticscholar.org/CorpusID:8665972

[8] Y. S. Kang, E. O'Sullivan, D. Choi, and M. O'Neill, "Security Analysis on RFID Mutual Authentication Protocol," 2016, pp. 65–74. https://doi.org/10.1007/978-3-319-31875-2_6

[9] L. Luo and D. Liu, "An Improved Lightweight RFID Mutual-authentication Protocol," in Proceedings of the Second International Conference on Mechanics, Materials and Structural Engineering (ICMMSE 2017), Paris, France: Atlantis Press, 2017. https://doi.org/10.2991/icmmse-17.2017.45

[10] F. Zhu, P. Li, H. Xu, and R. Wang, "A Lightweight RFID Mutual Authentication Protocol with PUF," Sensors, vol. 19, no. 13, p. 2957, Jul. 2019, https://doi.org/10.3390/s19132957

[11] M. Sidorov, M. T. Ong, R. V. Sridharan, J. Nakamura, R. Ohmura, and J. H. Khor, "Ultralightweight Mutual Authentication RFID Protocol for Blockchain Enabled Supply Chains," IEEE Access, vol. 7, pp. 7273–7285, 2019, https://doi.org/10.1109/ACCESS.2018.2890389.

[12] J. Lu, D. Liu, H. Li, C. Zhang, and X. Zou, "A Fully Integrated HF RFID Tag Chip With LFSR-based Light-weight Tripling Mutual Authentication

Protocol," IEEE Access, vol. 7, pp. 73285–73294, 2019, https://doi.org/10.1109/ACCESS.2019.2920437

[13] M. Hosseinzadeh et al., "A New Strong Adversary Model for RFID Authentication Protocols," IEEE Access, vol. 8, pp. 125029–125045, 2020, https://doi.org/10.1109/ACCESS.2020.3007771

[14] M. Hosseinzadeh et al., "An Enhanced Authentication Protocol for RFID Systems," IEEE Access, vol. 8, pp. 126977–126987, 2020, https://doi.org/10.1109/ACCESS.2020.3008230

[15] F. Zhu, "SecMAP: A Secure RFID Mutual Authentication Protocol for Healthcare Systems," IEEE Access, vol. 8, pp. 192192–192205, 2020, https://doi.org/10.1109/ACCESS.2020.3032541

[16] C. Trinh et al., "A Novel Lightweight Block Cipher-Based Mutual Authentication Protocol for Constrained Environments," IEEE Access, vol. 8, pp. 165536–165550, 2020, https://doi.org/10.1109/ACCESS.2020.3021701

[17] M. Naeem, S. A. Chaudhry, K. Mahmood, M. Karuppiah, and S. Kumari, "A scalable and secure RFID mutual authentication protocol using ECC for Internet of Things," International Journal of Communication Systems, vol. 33, no. 13, Sep. 2020, https://doi.org/10.1002/dac.3906

[18] S. Sharma, B. Kaushik, M. K. I. Rahmani, and Md. E. Ahmed, "Cryptographic Solution-Based Secure Elliptic Curve Cryptography Enabled Radio Frequency Identification Mutual Authentication Protocol for Internet of Vehicles," IEEE Access, vol. 9, pp. 147114–147128, 2021, https://doi.org/10.1109/ACCESS.2021.3124209

[19] X. ZHONG, M. XIAO, T. ZHANG, K. YANG, and Y. LUO, "Proving Mutual Authentication Property of RCIA Protocol in RFID Based on Logic of Events," Chinese Journal of Electronics, vol. 31, no. 1, pp. 79–88, 2022, https://doi.org/https://doi.org/10.1049/cje.2021.00.101

[20] H. Wang, J. Meng, X. Du, T. Cao, and Y. Xie, "Lightweight and Anonymous Mutual Authentication Protocol for Edge IoT Nodes with Physical Unclonable Function," Security and Communication Networks, vol. 2022, pp. 1–11, Jan. 2022, https://doi.org/10.1155/2022/1203691

[21] S. Cai, Y. Li, C. Ma, S. S. M. Chow, and R. H. Deng, "Prove You Owned Me: One Step beyond RFID Tag/Mutual Authentication," 2022. [Online]. Available: https://arxiv.org/abs/2210.10244

[22] D. Noori, H. Shakeri, and M. Niazi Torshiz, "An elliptic curve cryptosystem-based secure RFID mutual authentication for Internet of things in healthcare environment," EURASIP J Wirel Commun Netw, vol. 2022, no. 1, p. 64, Dec. 2022, https://doi.org/10.1186/s13638-022-02146-y

[23] G. Wei, Y. Qin, and W. Fu, "An Improved Security Authentication Protocol for Lightweight RFID Based on ECC," J Sens, vol. 2022, pp. 1–6, Feb. 2022, https://doi.org/10.1155/2022/7516010

[24] L. Li, J. Feng, B. Liu, Y. Guo, and Q. Li, "Implementation of PRINCE with resource-efficient structures based on FPGAs," Frontiers of Information Technology & Electronic Engineering, vol. 22, no. 11, pp. 1505–1516, Nov. 2021, https://doi.org/10.1631/FITEE.2000688

[25] V. Yli-Mayry et al., "Diffusional Side-Channel Leakage From Unrolled Lightweight Block Ciphers: A Case Study of Power Analysis on PRINCE," IEEE Transactions on Information Forensics and Security, vol. 16, pp. 1351–1364, 2021, https://doi.org/10.1109/TIFS.2020.3033441

[26] P. Blaskiewicz, B. Drzazga, L. Krzywiecki, D. Stygar, and P. Syga, "RFID Batch Authentication—A Usable Scheme Providing Anonymity," IEEE Access, vol. 10, pp. 85368–85383, 2022, https://doi.org/10.1109/ACCESS.2022.3197795

[27] C. Xu, W. Wei, and S. Zheng, "Efficient Mobile RFID Authentication Protocol for Smart Logistics Targets Tracking," IEEE Access, vol. 11, pp. 4322–4336, 2023, https://doi.org/10.1109/ACCESS.2023.3234959

[28] P. K. Maurya, H. Ghosh, and S. Bagchi, "MDS Code Based Ultralightweight Authentication Protocol for RFID System," IEEE Access, vol. 11, pp. 10563–10577, 2023, https://doi.org/10.1109/ACCESS.2023.3239530

[29] J. Borghoff et al., "PRINCE – A Low-Latency Block Cipher for Pervasive Computing Applications," 2012, pp. 208–225. https://doi.org/10.1007/978-3-642-34961-4_14

[30] H. K. Kharidu and V. Sudha, "FPGA implementation of EEG based hardware optimized data encryption technique for IoT applications," Integration, vol. 102, p. 102381, May 2025, https://doi.org/10.1016/j.vlsi.2025.102381

[31] Y. A. Abbas, R. Jidin, N. Jamil, M. R. Z'aba, M. E. Rusli, and B. Tariq, "Implementation of PRINCE algorithm in FPGA," in Proceedings of the 6th International Conference on Information Technology and Multimedia, IEEE, Nov. 2014, pp. 1–4. https://doi.org/10.1109/ICIMU.2014.7066593

[32] B. Rashidi, "Low-cost and two-cycle hardware structures of PRINCE lightweight block cipher," International Journal of Circuit Theory and Applications, vol. 48, no. 8, pp. 1227–1243, Aug. 2020, https://doi.org/10.1002/cta.2832

[33] A. A. Abdullah and N. R. Obeid, "Efficient Implementation for PRINCE Algorithm in FPGA Based on the BB84 Protocol," J Phys Conf Ser, vol. 1818, no. 1, p. 012216, Mar. 2021, https://doi.org/10.1088/1742-6596/1818/1/012216

[34] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in Proceedings of the 52nd Annual Design Automation Conference, New York, NY, USA: ACM, Jun. 2015, pp. 1–6. https://doi.org/10.1145/2744769.2747946

[35] S. Feizi, A. Nemati, A. Ahmadi, and V. A. Makki, "A high-speed FPGA implementation of a Bit-slice Ultra-Lightweight block cipher, RECTANGLE," in 2015 5th International Conference on Computer and Knowledge Engineering (ICCKE), IEEE, Oct. 2015, pp. 206–211. https://doi.org/10.1109/ICCKE.2015.7365828

[36] M. S. Naik, D. K. Sreekantha, and K. V. S. S. S. S. Sairam, "An Efficient Low-Latency and High Throughput LED Cipher Architecture for IoT Security on a Hardware Platform," SN Comput Sci, vol. 5, no. 7, p. 908, Sep. 2024, https://doi.org/10.1007/s42979-024-03275-5

[37] M. S. Naik, D. K. Sreekantha, and K. V. Sairam, "Enabling low-latency IoT communication for resource-constrained devices with the led cipher and decipher protocol," Indonesian Journal of Electrical Engineering and Computer Science, vol. 34, no. 2, p. 1170, May 2024, https://doi.org/10.11591/ijeecs.v34.i2.pp1170-1180

[38] C. A. Lara-Nino, A. Diaz-Perez, and M. Morales-Sandoval, "Lightweight Hardware Architectures for the Present Cipher in FPGA," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 64, no. 9, pp. 2544–2555, Sep. 2017, https://doi.org/10.1109/TCSI.2017.2686783

[39] R. Anusha and V. Veena Devi Shastrimath, "LCBC-XTEA: High Throughput Lightweight Cryptographic Block Cipher Model for Low-Cost RFID Systems," 2019, pp. 185–196. https://doi.org/10.1007/978-3-030-19813-8_20

[40] R. Bharathi and N. Parvatham, "Light-Weight Present Block Cipher Model for IoT Security on FPGA," Intelligent Automation & Soft Computing, vol. 33, no. 1, pp. 35–49, 2022, https://doi.org/10.32604/iasc.2022.020681

[41] S. Seshabhattar, S. K. Jagannatha, and D. W. Engels, "Security implementation within GEN2 protocol," in 2011 IEEE International Conference on RFID-Technologies and Applications, IEEE, Sep. 2011, pp. 402–407. https://doi.org/10.1109/RFID-TA.2011.6068669

[42] R. Anusha and V. V. D. Shastrimath, "RFID-MA XTEA: Cost-Effective RFID-Mutual Authentication Design Using XTEA Security on FPGA Platform," International Journal of Electronics and Telecommunications, pp. 623–629, Jul. 2021, https://doi.org/10.24425/ijet.2021.137855

[43] B. Ramachandra and S. E. Peter, "Secured authentication of radio-frequency identification system using PRESENT block cipher," International Journal of Electrical and Computer Engineering (IJECE), vol. 13, no. 5, p. 5462, Oct. 2023, https://doi.org/10.11591/ijece.v13i5.pp5462-5471