Design and Implementation of ECC-Based RFID Tag for Wireless Communications on FPGAs

Neelappa and N.G. Kurahatti

Abstract This paper proposes an elliptic curve cryptography (ECC) and direct spread spectrum (DSS) to implement Radio-Frequency Identification (RFID) tag chip for highly secure wireless communication. Digital baseband controller (DBC) and nonlinear feedback shift register (NLFSR) have been used to generate control signal, data transfer, and random number sequences for ECC processor and DSS compatible with ISO/IEC 14443. In order to achieve optimized resources in field programmable gate array (FPGA) for ECC, point multiplication, reusable registers, and asynchronous counter are adopted. The proposed work has been implemented on two Spartan 6 FPGAs. Wireless communication between them has been established via Zigbee modules. Single-user DS-SS system using pseudo-chaotic sequence as spreading sequence and RFID transmitter and receiver in FPGA development kit targeted to Xilinx's Spartan 6 device XC6S45-2tq324 has been implemented. The proposed elliptic curve processor (ECP) in digital baseband controller (DBC) needs only 8.58 K gate area and has a delay of 7.509 ns. The synthesis results show that the power consumption of DBC including ECP and other units in transmitter and receiver is only 381.58 μW at 35,810 kHz. Considerable improvement in power dissipation, area, and delay has been achieved. Security of the data has been ensured by using ECC.

Keywords Elliptic curve cryptography · Direct spread spectrum
Nonlinear feedback shift register · Digital baseband controller · Protocol and detector · Radio-frequency identification tag

Neelappa (✉)
Government of Engineering College, Kushal nagar, Karnataka, India
e-mail: neel_m_d@yahoo.co.in; neel.m.dy@gmail.com

N.G. Kurahatti
East Point College of Engineering, Bengaluru, Karnataka, India
e-mail: ngkurahatti@yahoo.co.in

M.S. Reddy et al. (eds.), *International Proceedings on Advances in Soft Computing, Intelligent Systems and Applications*, Advances in Intelligent Systems and Computing 628, https://doi.org/10.1007/978-981-10-5272-9_38

1 Introduction

One of the features of improvements of RFID is an auto proof of identity expertise. It is widely used for proof of identity, control chain organization, wireless sensor networks (WSNs), ad hoc wireless communication, and other applications. With the development of the Internet of Things (IOT) and WSN, the demand on security-related RFID systems has expanded [1]. RFID applications require low-power and low-cost implementation with high data security. To satisfy security, a suitable public key cryptography scheme is required. RFID passive tags obtain energy from radio frequency signals transmitted from the reader and have limited power supply. Therefore, these tags cannot use the energy-demanding algorithms such Revert-Shamir-Adleman (RSA) cryptography. Elliptic curve cryptography (ECC), proposed by Koblitz [2], has been employed, the main advantages being by employing ECC are smaller key sizes and offers comparable security level as RSA [3]. This feature has been incorporated in the implementation of RFID tag chips. The essential operation in the elliptic curve cryptosystems is scalar point multiplication. The point multiplication is achieved by finite field arithmetic computations such as field addition, field multiplication, field squaring, and inversion. Tawallebh et al. [4] proposed processor architecture for elliptic curves over prime fields. The speed of point multiplication is increased by proper selection of the coordinate system [5]. A number of hardware implementations for elliptic curve cryptography have been proposed in the literature, but only a few of them are aimed for low-end devices. The proposed implementation is emphasis on speed, area, and power which are based on FPGA technology [6].

In this paper, we propose an ECC algorithm based on projective coordinates, which can be adopted for both binary and prime fields for resource limited RFID tags. The data is encoded and decoded by chaos-based direct spread spectrum/ binary phase shift keying (DSS/BPSK) modulation. The proposed RFID system with DSS/ECC offers improved security feature and makes analysis of the RFID tag in terms of area, speed, and power. The proposed work has been implemented and tested on Spartan-6 FPGA boards.

The paper proceeds as follows: Sect. 2 describes mathematical foundation. Section 3 describes RFID transmitter, and Sect. 4 describes RFID receiver. Section 5 describes experimental results and comparison and conclusion.

2 Mathematical Foundation

In cryptography, two of the most studied fields are finite fields of characteristic two, denoted by $GF\ (2^m)$ and prime fields. Advantage of $GF\ (2^m)$ fields is the simple hardware required for computation of commonly used arithmetic operations such as addition and squaring. Addition and squaring in $GF\ (2^m)$ can be performed by a simple XOR operation. These operations are simpler than the addition and squaring circuits of a *GF* (*P*) field. In the proposed paper, ECC operations are performed on binary field.

2.1 Finite Field Operations (2^m)

EC over field $F(2^m)$ includes arithmetic of integer with length m bits. The binary string can be declared as polynomial:

Binary string: $(b_{m-1} \ldots b_1\, b_0)$

Polynomial: $b_{m-1}\, y^m - 1 + b_{m-1}\, y^{m-2} + \ldots + b_2y^2 + b_1y + b_0$ where $b_i = 0$. For example, $y^3 + y^2 + y$ is polynomial for a four-bit binary number 1110.

In addition

If $A = y^3 + x^2 + 1$ and $B = y^2 + y$ are two polynomial, then $A + B$ is called polynomial addition that returns $y^3 + 2y^2 + y + 1$ after taking mod 2 over coefficients as $A + B = y^3 + y + 1$. On binary representation, polynomial addition can be achieved by simple XOR of two numbers. If $A = 1101$ and $B = 0110$, then $A + B = A$ XOR $B = A + B = (1011)_2$.

In multiplication

If $A = y^3 + y^2 + 1$ and $B = y^2 + x$ are two polynomials, then $A * B$ is called polynomial multiplication that returns $y^5 + y^3 + y^2 + y$, $m = 4$. The result should be reduced to a degree less than 4 by irreducible polynomial $y^4 + y + 1$.

$y^5 + y^3 + y^2 + y \pmod{f(y)} = (y^4 + y + 1)(y + y^5 + y^3 + y^2 + y) = 2y^5 + y^3 + 2y^2 + 2y = y^3$ (after reducing the coefficient on mod 2).

If $A = (1101)_2$ and $B = (0110)_2$, then $A * B = (1000)_2$.

3 Methodology

In this proposed work, the approach is divided into two sections: (i) block diagrams of DSS and ECC-based RFID tag chip at transmitter and (ii) block diagram of DSS and ECC-based RFID tag chip at receiver.

3.1 Transmitter

Figure 1 shows the block diagram of DSS and ECC-based RFID tag at transmitter section. A typical transmitter-embedded RFID tag chip can be divided into six parts: analog front end (AFE), NLFSR, EEPROM, ECC processor, wireless Zigbee transmitter, and digital baseband controller. AFE realizes the comprehensive functions of physical layer according to the RFID protocol, tag id stored in reuse register. NLFSR generates sequences of numbers for pseudo-chaotic sequence

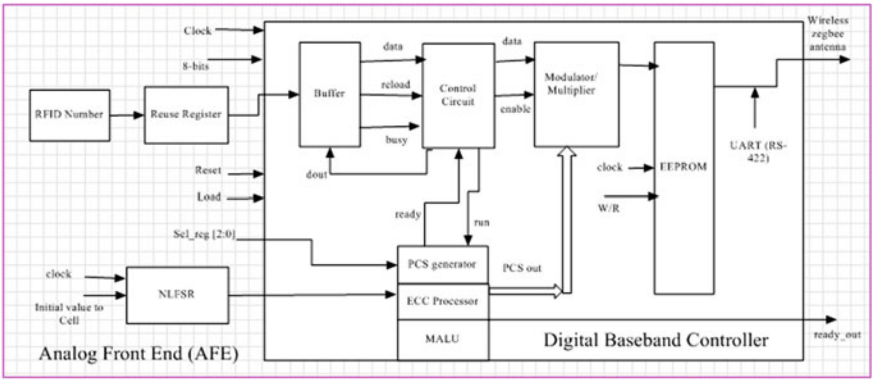


Fig. 1 Proposed block diagram of DSS and ECC-based RFID tag chip at transmitter section

(PCS) generator and ECC processor, AFE including carrier signal, clock generation, and reset signal generation. PCS creates randomness for each authentication so that the data in the authentication is highly secure. EEPROM is used for storing private or public information, such as the private key, base point of elliptic curve (EC), and the EC equation parameters. Utilizing the stream line bus structure, baseband controller integrates system controller, memory interface, buffer, multiplier, PCS, and ECC processor into one unit.

3.2 Direct Spread Spectrum (DSS)

In DSS system, the data bits are generated using PCS generator are spread. To generate PCS sequence, initially all 8-bit registers from R_1 to R_8 have to be initialized based on two sets of initial values. To load eight registers of the PCS generator, set load = 1 and using three signal pins of sel_reg registers R_1–R_8 are selected. The 8-bit initial values to each of these eight registers are loaded using eight signal pin reg_init internally [5]. After loading the initial values to all the eight registers, ready out signal pin gives an indication to the user. At the same time, signal ready is set to high and gives an indication to the control circuit to start its operation. Prior to loading the initial values to the registers R_1–R_8, the busy and done signal should be high and low, respectively. After receiving the 8-bit tag id data frame from RFID tag, the control circuit permits the PCS generator to generate PCS sequence by setting signal run = 1. It also enables the multiplier by setting enable = 1 and indicates the buffer that it is busy by setting the signal busy = 1. During this time, the PCS generator starts generating the 64 bits of PCS sequence. The control circuit then transfers one bit at a time serially to the multiplier, where it is multiplied by the 64 bits of generated PCS sequence resulting in a 64 bits of spread sequence, and the same is transmitted. After the first RFID number is serialized into bit spread by 64 bits of PCS sequence, the second data bit is received in the multiplier and is multiplied by the next 64 bits of the PCS sequence. Thus, the PCS generator generates a total of 512 bits to spread all the 8 bits of RFID data. After transmitting all the 512 bits with respect to one data (8 bits), the control circuit makes signal done = 1 and busy = 0 and then it accepts next frame of 8 bits of RFID id data.

The transmitter operation is carried out using direct spectrum sequence by multiplying the RFID tag number $b(t)$ with PCS sequence $p(t)$ which acts as a carrier signal for modulator. The spreading signal $s(t)$ is then modulated with $p(t)$ by means of BPSK, and the resultant of DSS/BPSK signal is given by

$$e(t) = Ap(t)b(t)\sqrt{2p}\cos(2\pi f_0 t) \tag{1}$$

where A and f_0 are the amplitude and frequency of the carrier. The modulated signal $e(t)$ is transmitted through wireless Zigbee via serial communication protocol universal asynchronous receiver/transmitter (UART) shown in Fig. 1.

3.3 ECC Processor

Prime field-based ECC processor with high-speed operating frequency of 50 MHz and scalar multiplication to perform both point addition and point doubling in affine coordination is adopted in this work. Figure 2 shows the overall ECC dual-field architecture with input/output buffers, control unit, data selector, register file, and ECC scalar multiplication. The data is fed into an input buffer and read the output buffer through I/O interfacing. ECC parameters are written into the buffer before the computation. All operations are controlled by the control unit. The control instructions are stored in the control register and decoded by the main controller architecture of ECC arithmetic unit. The Karatsuba multiplier [7] is used to perform point addition and doubling for both fields. Results are stored in the register files.

3.3.1 Scalar Multiplication

The ECC scheme requires the point and scalar multiplication defined as follows:

$$Q = kP = P + P + \cdots + P\,(k \text{ times})$$

Here, P denotes a point on the elliptic curve and k is a random integer. Point addition and point doubling play a key role in scalar multiplication.

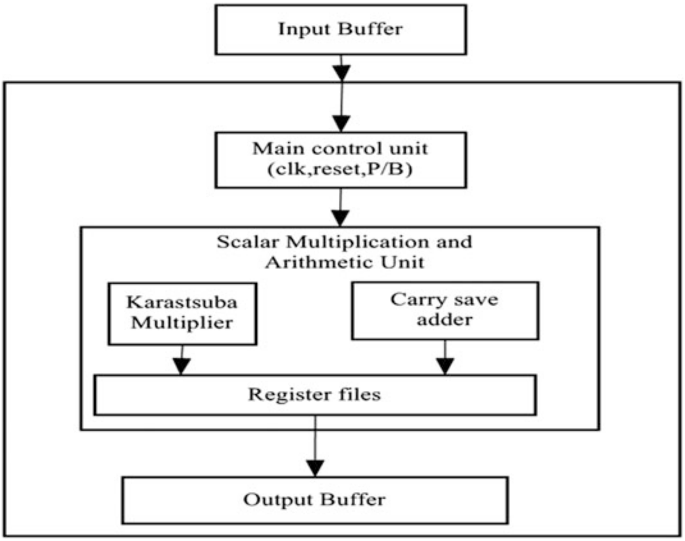


Fig. 2 Proposed architecture of the ECC processor (GF^{163})

The scalar multiplication algorithm is as follows:

```
Input : $k = (k_{n-1,}k_{n-2-------}k_{1,}k_0), P$;
Output $= [k]\,P$;
$R_0 = 0$; $R_1 = P$;
For $i = n - 1$ down to 0
do
$b = k_i$; $R_{1-b} = R_{1-b} + R_b$;
$R_b = 2R_b$;
end for;
return $R_0$
```

An elliptic curve equation over prime field is given by

$$y^2 \bmod p = (x^3 + ax^2 + b) \bmod P \tag{2}$$

where a and b are the parameters and x and y are the points on curves.

Binary field equation is given by

$$y^2 + xy = x^3 + ax^2 + b \tag{3}$$

ECC over binary field achieves high performance without considering carry and modular reduction. These fields are optimal for the use in hardware in terms of area and speed.

3.3.2 Binary Field

The most important elliptic curve equation is: $y^2 + xy = x^3 + ax^2 + b$ [8].

In binary field, addition is performed by an EX-OR operation and multiplication is polynomial based. The result is reduced by using the irreducible polynomial. Squaring is achieved by shift operation.

Point Addition Over Binary Field

In this work, one point is in projective coordinate and another point is an affine coordinate. The resulting point will be in projective coordinate which avoids the inversion operation.

Algorithm for addition is as follows:

Inputs: A(x_2, y_2), $Q(X_4,Y_4,Z_4)$.
Outputs: $R(X_3,Y_3,Z_3)$.

$$\begin{aligned}
&A = Y_4 + y_2 * Z_4^2;\\
&B = X_4 + x_2 * Z_4;\\
&C = B * Z_4;\\
&Z_3 = C * C;\\
&D = x_2 * Z_3;\\
&E = A + B * B + aC;\\
&X_3 = A * A + C * E;\\
&I = D + X_3;\\
&J = A * C + Z_3;\\
&F = I * J;\\
&K = Z_3 * Z_3;\\
&Y_3 = \ F + x_2 * K + y_2 * K
\end{aligned}$$

The point doubling operation is to add a point on the elliptic curve with itself. In these equations, 'a' and 'b' are considered as parameters of an elliptic curve.

Algorithm for point doubling is as follows:

Inputs: (x_1, y_1, z_1);
Outputs: $(x_4, y_4, z_4);$

$$\begin{aligned}
&z_4 = z_1^2 * x_1^2,\\
&x_4 = x_1^4 + bz_1^4,\\
&y_4 = \left(y_1^2 + az_4 + bz_1^4\right) * x_4 + z_4 * bz_1^4.
\end{aligned}$$

3.3.3 Prime Field

In prime field, each elliptic curve addition and doubling requires a fixed number of modular multiplications, square, additions, shifts, and similar basic arithmetic operations. The actual number of these operations depends on the progression of the curve. The operation of the multiplication and square operations that dominate the running time, which scales exactly as the number of arithmetic operations to be performed.

Point Addition Over Prime Field

For an elliptic curve defined over $GF(P)$, the normal elliptic point (x, y) is projected to (X_1, Y_1, Z_1), where $x = X/Z^2$, and $y = Y/Z^3$, and the second point considered is affine point that is (x_2, y_2).

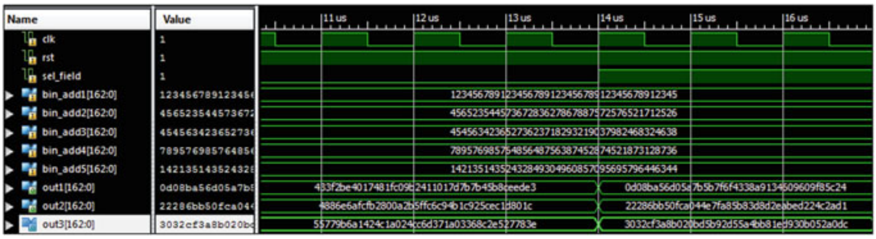
Algorithm for a point addition is as follows:

Inputs: $Q = (X_4,Y_4,Z_4)$, $A = (x_2,y_2)$
Output: $R = (X_3,Y_3,Z_3) = P + Q$;

$$A = X_4;$$
$$B = x_2 * Z_1^2;$$
$$C = A - B;$$
$$D = Y_1;$$
$$E = y_2 * Z_1^3;$$
$$F = D - E;$$
$$G = A + B;$$
$$H = D + E;$$
$$Z_3 = Z_1 * C;$$
$$X_3 = F^2 - G * C^2;$$
$$I = G * C^2 - 2 * X_3;$$
$$Y_3 = \left(I * F - H * C^2\right)/2;$$

In the $GF(P)$, the point doubling algorithm is as follows:

Inputs: $P = (X_1,Y_1,Z_1)$, a;
Output: $Q = (X_4,Y_4,Z_4) = 2P$;



Name	Value
clk	1
rst	1
sel_field	1
bin_add1[162:0]	123456789123456
bin_add2[162:0]	456523544573672
bin_add3[162:0]	454563423652736
bin_add4[162:0]	789576985764856
bin_add5[162:0]	142135143524328
out1[162:0]	0d08ba56d05a7b5
out2[162:0]	22286bb50fca044
out3[162:0]	3032cf3a8b020bd

11 us
12 us
13 us
14 us
15 us
16 us
123456789123456789123456789123456789 12345
45652354457367283627867887572576521712526
4545634236527362371829321903798246832463 8
7895769857548564875638745287452187312873 6
1421351435243284930496085709569579644634 4
433f2be4017481fc09b2411017d7b7b45b8eede3
0d08ba56d05a7b5b7f6f4338a9134609609f85c24
4886e6afcfb2800a2b5ffc6c94b1c925cec1d801c
22286bb50fca044e7fa85b83d8d2eabed224c2ad1
55779b6a1424c1a024cc6d371a03368c2e527783e
3032cf3a8b020bd5b92d55a4bb81ed930b052a0dc

Fig. 3 Simulated results of ECC processor in GF (2^{163}) for prime and binary fields

$$
\begin{aligned}
&A = 3 * X_1^2 + a * Z_1^4;\\
&B = 4 * X_1 * Y_1^2;\\
&X_4 = A^2 - 2 * B;\\
&Z_4 = 2 * Y_1 * Z_1;\\
&C = 8 * Y_1^4;\\
&Y_4 = A * (B - X_4) - C;
\end{aligned}
$$

ECC processor has been designed for 163 bits for both binary and prime fields, and to select particular field, sel_field control signal selects either binary field or prime field, when it is ‘1’ binary field is selected else prime field. The clock frequency for ECC processor is 100 MHz which is generated from Spartan 6 FPGA. Reset clears all internal registers and memories. Out1, Out2, and Out3 are the keys generated from ECC based on field selection and are shown in Fig. 3.

4 Receiver

Receiver block diagram is as shown in Fig. 4. It consists of detector, control circuit, PCS generator, and demodulator. The initial 64-bit data is loaded into eight registers from R_1 to R_8 which are used in the transmitter, and after the initial data loaded, ready control signal will become high (Logic 1) and it is control signal to PCS generator through control circuit. During the process phase, the control circuit maintains the signal ld = 1, enable = 0, and run = 1. Now, the detector block is initialized with a process sequence, the first 512 bits of the PCS sequence are loaded into the eight lower registers, each length is 64 bits of the detector. The synchronization between transmitter and receiver is important in a DSS system when the receiver block uses pseudo-chaotic sequence. When the control circuit of receiver generates ld = 1, then the 64 bits of the PCS producer stored into the shift register 1 of 64-bit detector. After loading all the 512 bits into shift register 1 of eight 64-bit detectors, control circuit resets ld = 0, so that the receiver bits are moving to the shift register 2. After receiving each bit from channel, the detector

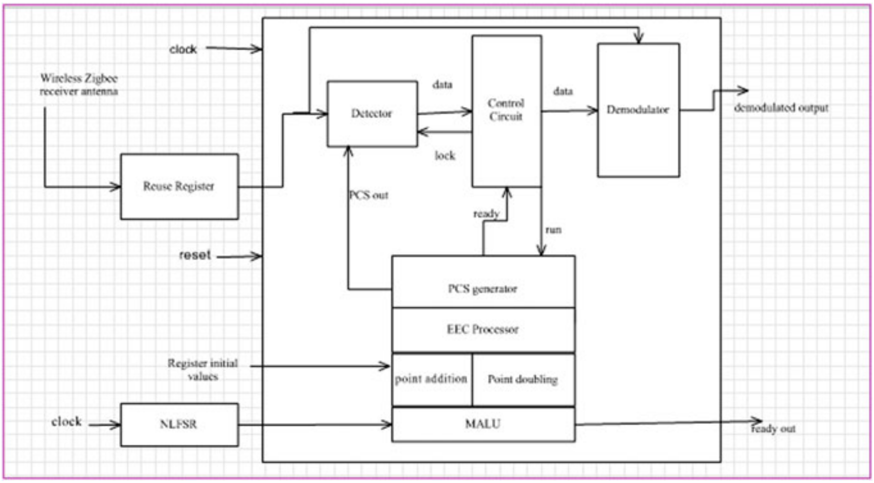


Fig. 4 Proposed block diagram of the DSS and ECC-based RFID tag chip at receiver

block multiplies each bit with shift register 1 and shift register 2 and sums up the output of the bit-wise multiplier, if the sum exceeds the selected threshold value, detector block decides the received bit is logic 1 or 0. The mathematical analysis of the receiver is as follows.

The received signal $r(t)$ at the input of the receiver is the sum of the transmitted signal $e(t)$ and channel noise $n(t)$. Firstly, the received signal is multiplied with the pseudorandom number (PN) sequence and resulting signal $g(t)$ is given as follows:

$$\begin{aligned} g(t) &= r(t).p(t) = (e(t) + n(t))p(t) \\ &= \mathrm{Ab}(t).p^2(t)\cos(2\pi f_0 t) + n(t)p(t) \\ &= \mathrm{Ab}(t).\cos(2\pi f_0 t) + n(t).p(t). \end{aligned} \tag{4}$$

Signal (t) is multiplied with the sinusoidal carrier, and resulting signal $s(t)$ is given is as follows:

$$\begin{aligned} s(t) &= Ag(t)\cos(2\pi f_0 t) \\ &= Ab(t))\cos^2(2\pi f_0 t) + An(t)p(t)\cos(2\pi f_0 t) \\ &= A^2/b(t) + A^2/b(t)\cos(4\pi f_0 t) + An(t)p(t)\cos(2\pi f_0 t). \end{aligned} \tag{5}$$

The signal $s(t)$ is fed to the integrator whose output is reset to zero by the trigger of each pulse of the train $p(t)$. It means that the integration period of each bit is equal to the corresponding interpulse interval which is also the corresponding bit duration. Before each reset instance, the output signal of the integrator $i(t)$ is sampled. The output value of the sampler at the instance t_{n+1} is given by.

$$
\begin{aligned}
i(t_{n+1}) &= \int\limits_{t_n}^{t_{n+1}} s(t)\mathrm{d}t \\
&= \int\limits_{0}^{T_{bn}} \frac{A^2}{2} b(t)\mathrm{d}t + \int\limits_{0}^{T_{bn}} \frac{A^2}{2} b(t)\cos(4\pi f_0 t)dt + \int\limits_{0}^{T_{bn}} An(t)p(t)\cos(2\pi f_0 t)\mathrm{d}t \\
&= \frac{A^2}{2} b_n T_{bn} + 0 + \int\limits_{0}^{T_{bn}} An(t)p(t)\cos(2\pi f_0 t)\mathrm{d}t
\end{aligned}
\tag{6}
$$

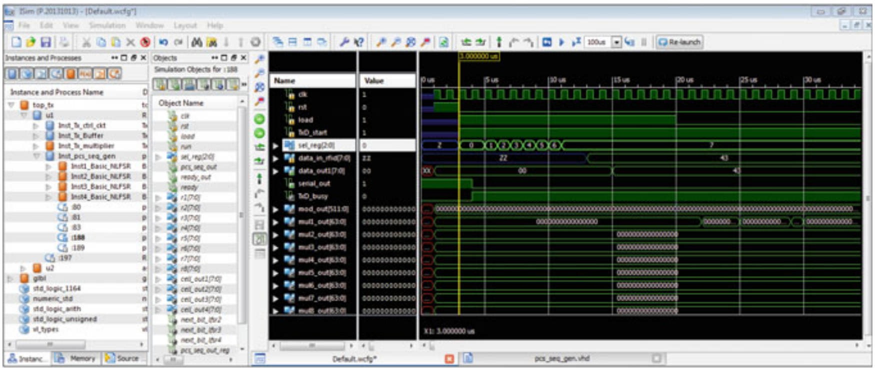
where ($A^2/2bn$) is the energy of the desired signal. $\int_0^{T_{bn}} \frac{A^2}{2} b(t)\cos(4\pi f_0 t)\mathrm{d}t$ equal to zero because the period T_{bn} is a multiple of the carrier cycle. $\int_0^{T_{bn}} An(t)c(t)$ $\cos(2\pi f_0 t)\mathrm{d}t$ is the energy produced by the channel noise. It is noted that the correlation between $p(t)$ and $An(t)$ $\cos(2\pi f_0 t)$ is very low, and hence the noise energy is much less than the signal energy. The resulting sample is fed to the decision device to recover the binary value of the *n*th bit as follows:

$$
\begin{aligned}
b_n &= \{1, i(t_{n+1}) \geq 0, \\
&= \{0, i(t_{n+1}) < 0.
\end{aligned}
\tag{7}
$$

5 Results and Comparison with Methods Proposed in the Literature

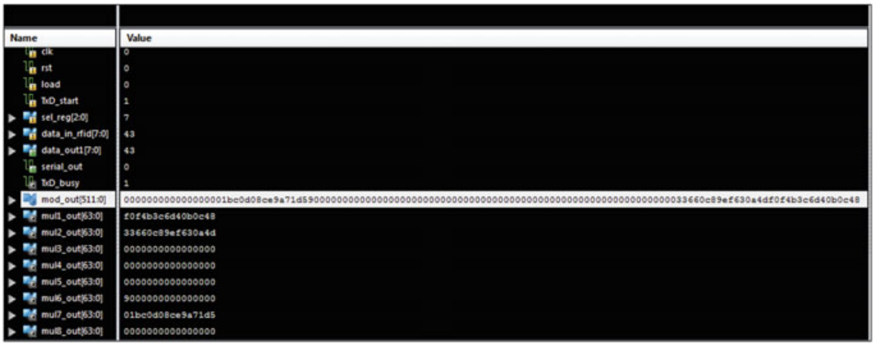
In the proposed work, the results obtained from simulation are verified on two Spartan 6 FPGAs (XCS6LX45). At the transmitter section, one FPGA along with Zigbee module and RFID tag has been used to process and transmit the modulated signal. At the receiver section, another FPGA along with Zigbee module and RFID tag has been used to retrieve the RFID tag number which is displayed on LCD of receiver FPGA. The 163-bit ECC processor using scalar multiplication (both binary and prime fields) has been incorporated in transmitter section to secure tag number. Figure 5 shows loading of initial values into eight registers from R_1 to R_8 using sel_reg control signal.

Before transmitting and after modulation, all 64-bit data is concatenated to get 512 bits and transmitted along with RFID tag number, The output of multiplier with 512 bits at RFID transmitter is shown in Fig. 6.



Name	Value
clk	1
rst	0
load	1
TxD_start	1
sel_reg[2:0]	0
data_in_rfid[7:0]	22
data_out1[7:0]	00
serial_out	1
TxD_busy	0
mod_out[511:0]	000000000000
mul1_out[63:0]	000000000000
mul2_out[63:0]	000000000000
mul3_out[63:0]	000000000000
mul4_out[63:0]	000000000000
mul5_out[63:0]	000000000000
mul6_out[63:0]	000000000000
mul7_out[63:0]	000000000000
mul8_out[63:0]	000000000000

Fig. 5 Loading initial values into R_1–R_8 registers at RFID transmitter



Name	Value
clk	0
rst	0
load	0
TxD_start	1
sel_reg[2:0]	7
data_in_rfid[7:0]	43
data_out1[7:0]	43
serial_out	0
TxD_busy	1
mod_out[511:0]	000000000000000001bc0d08ce9a71d590000000000000000000000000000000000000000000000000000000000000000000000000000033660c89ef630a4df0f4b3c6d40b0c48
mul1_out[63:0]	f0f4b3c6d40b0c48
mul2_out[63:0]	33660c89ef630a4d
mul3_out[63:0]	0000000000000000
mul4_out[63:0]	0000000000000000
mul5_out[63:0]	0000000000000000
mul6_out[63:0]	9000000000000000
mul7_out[63:0]	01bc0d08ce9a71d5
mul8_out[63:0]	0000000000000000

Fig. 6 Concatenation of all 64-bit multiplier outputs to get 512 bits at RFID transmitter

PCS generator generates spread spectrum with each size of 64 bits stored in different registers and then applied to both modulator and multiplier; these two modules generate the final modulated signals. These modulated signals are transmitted through Zigbee transmitter using UART protocol with the baud rate of 11,520 bits per second. The simulated results of tag number and multiplier output are shown in Fig. 7.

In receiver section, Zigbee receives the data serially through R_x pin and stores in 8-bit register to get 512-bit data which contains both RFID tag number and carrier signal and then sends to detector to decode tag number. After demodulation, RFID tag number is displayed on LCD of FPGA board. The demodulated tag number at the receiver section is as shown in Fig. 8.

The proposed work has implemented using Cadence software tools to measure the area, power, and delay. Table 1 shows the comparison of present work and existing work. The proposed ECP in DBC needs only 8.58 K gate area and has a

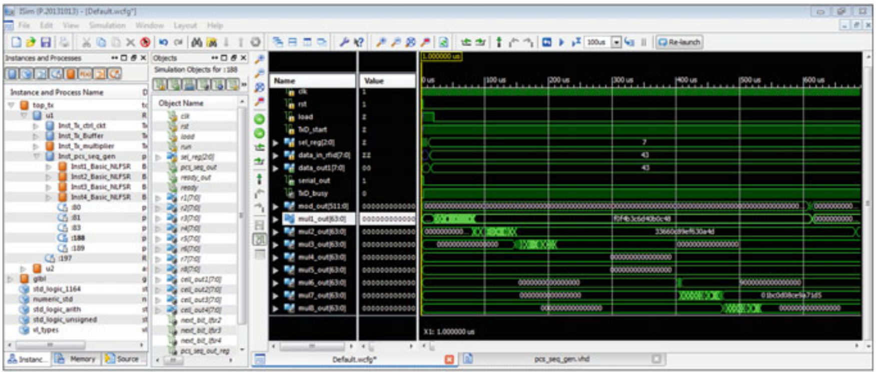


Fig. 7 RFID tag number and 64-bit multiplier outputs at RFID transmitter

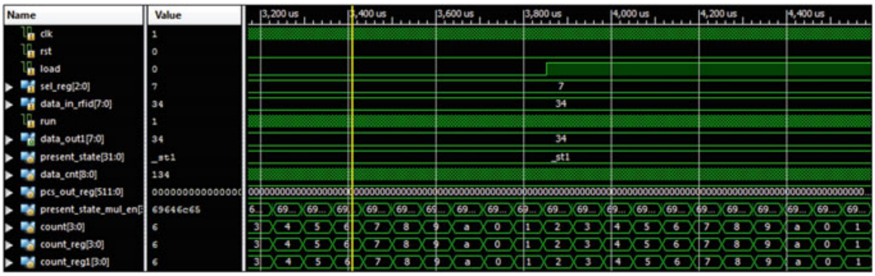


Fig. 8 Demodulated tag number at receiver section

Table 1 Comparison of performance characteristics of power consumption, frequency, delay, and area

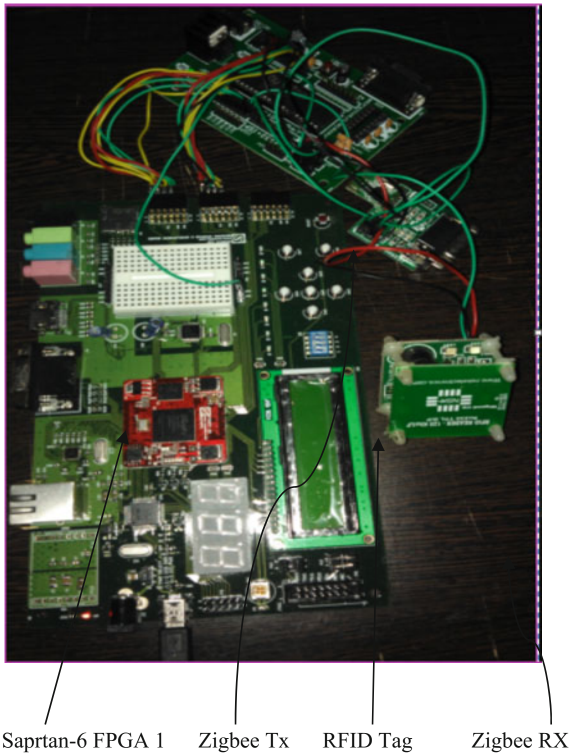
Ref. No.	Digit size	Field size	Frequency (kHz)	Power (μW)	ECP area (gates) K	Delay (ns)	Technology
This work	8	GF(2^{163})	35,810	383.17	8.58	7.50	FPGA
01	8	GF(2^{163})	13,536	253	14.1	13.2	Umc 0.13 μm
06	8	GF(2^{163})	2454	320.3	12.5	13.1	Umc 0.13 μm
07	8	GF(2^{163})	13560	NA	12.5	244	Umc 0.13 μm
08	8	GF(2^{163})	400	7.3	56.7	31.8	Umc 0.13 μm
10	8	GF(2^{163})	1059	148.5	13.2	547.8	Umc 0.13 μm
15	8	GF(2^{163})	13,536	208.4	21.8	12.5	Umc 0 .13 μm
16	8	GF(2^{100})	500	400	18.7	NA	TSMC 0.18 μm
17	8	GF(2^{163})	200	15	6.1	NA	TSMC 0.18 μm
18	8	GF(2^{163})	76	4.88	13.78	NA	65 nm CMOS

delay of 7.50 ns. The synthesis results show that the power consumption of DBC including ECP and other units in transmitter and receiver is only 381.58 μW at 35,810 kHz.

5.1 Hardware Implemented Results

The proposed work has been implemented and tested on two FPGA boards, one for transmission is shown in Fig. 9 and other for receiver is shown in Fig. 10. The receiver module receives tag number and displays on LCD and LED.

Fig. 9 Hardware implementation for transmission



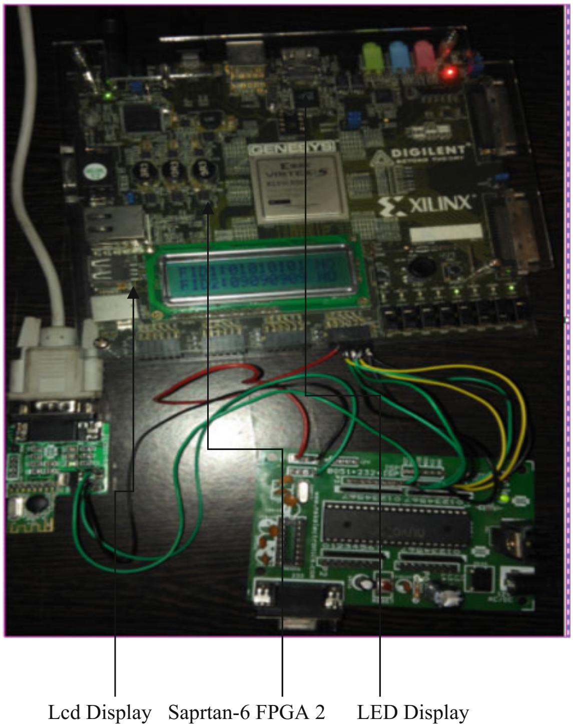


Fig. 10 Hardware implementation for receiver

5.2 *Conclusion*

Single-user DS-SS system using pseudo-chaotic sequence as spreading sequence and RFID transmitter and receiver in FPGA development kit targeted to Xilinx's Spartan 6 device XC6S45-2tq324 has been implemented. Transmission has been achieved through Zigbee. Considerable improvement in power dissipation, area, and delay has been achieved. Security of the data has been assured using ECC.

References

1. Zilong, Liu , Dongsheng Liu, Xuecheng Zou, Hui Lin, and Jian Cheng. 2014. Design of an Elliptic Curve Cryptography Processor for RFID Tag Chips *Sensors* 14: 17883–17904
2. Koblitz, N. 1987. Elliptic Curve Cryptosystems. *Mathematics of Computation* 48: 203–220.
3. Lai, Jyu-Yuan, and Chih-Tsun Huang. 2011. Energy-Adaptive Dual-Field Processor for High-Performance Elliptic Curve Cryptographic Applications. *IEEE Transactions on VLSI* 19 (8).
4. Kocabas, U., J. Fan, and I. Verbauwhede. 2010. Implementation of binary Edwards curves for very-constrained devices. In *Proceedings of 2010 the 21st International Conference on*

Application-Specific Systems Architectures and Processor (*ASAP*), Rennes, France, 7–9 July 185–191.

5. Hein, D., J. Wolkerstorfer, and N. Felber. 2009. ECC is Ready for RFID-a proof in silicon. In *Selected Areas in Cryptography*, vol. 5381 ed. Roberto, M.A., and K. Liam, 401–413. Heidelberg, Germany: Springer.
6. Azarderakhsh, R., and A.R. Masoleh. 2013. High-Performance Implementation of Point Multiplication on Koblitz Curves. *IEEE Transactions on Circuits and Systema II Express Briefs* 60: 41–45.
7. National Institute of Standards and Technology. Recommended Elliptic Curves for Federal Government Use. Available online: http://csrc.nist.gov/groups/ST/toolkit/documents/dss/NISTReCur.pdf (Accessed on 15 Sept 2014).
8. Ting, Hsin-Yu, and Chih-Tsun Huang. Design of Low-Cost Elliptic Curve Cryptographic Engines for Ubiquitous Security. 978-1-4799-2776-0/14/$31.00 ©2014 IEEE.
9. Lai, Jyu-Yuan, and Chih-Tsun Huang. 2008. High-Throughput Cost-Effective Dual-Field Processors and the Design Framework for Elliptic Curve Cryptography. *IEEE Transactions on VLSI* 16 (11).
10. Sakiyama, K., L. Batina, B. Preneel, and I. Verbauwhede. 2007. Multi-Corecurve-Based Cryptoprocessor with Reconfiguarable Modular Arithmetic Logic Units Over GF(2^{m}). *IEEE Transacions on Computers* 56 (9): 1269–1282.
11. Lee, Y.K., K. Sakiyama, L. Batina, and I. Verbauwhede. 2008. Elliptic-Curve-Based Security Processor for RFID. *IEEE Transactions on Computers* 57: 1514–1527.
12. Kumar, S., and C. Paar. 2006. Are standards compliant elliptic curve cryptosystems feasible on RFID? In *Proceedings of Workshop on RFID Security, Graz, Austria*, 12–14, July 2006.
13. Ansari, B., and M.A. Hasan. 2008. High-Performance Architecture of Elliptic Curve Scalar Multiplication. *IEEE Transactions on Computers* 57 (11): 1143–1153.
14. Sakiyama, K., E. De Mulder, B. Preneel, and I. Verbauwhede. 2006. A Parallel Processing Hardware Architecture for Elliptic Curve Cryptosystems. *In Proceedings of IEEE International Conference Acoustic, Speech Signal Process* (*ICASSP*), vol. 3. Toulouse, France 904–907.
15. Shylashree, N., A. Deepika, and V. Sridhar. 2012. High-Speed FPGA-Based Elliptic Curve Cryptography Using Mixed Co-ordinates. *Journal of Discrete Mathematics and Cryptography* 46: 8511–8516 (Elixir).
16. Liu, Dongsheng, Zilong Liu, Zhenqiang Yong, Xuecheng Zou, and Jian Cheng. 2015. Design and Implementation of an ECC-Based Digital Baseband Controller for RFID Tag Chip. *IEEE Transactions on Industrial Electronics* 62 (7).
17. G Gaubatz et al. 2005. State of the Art in Ultra-Low Power Public Key Cryptography for Wireless Sensor Networks. In *3rd IEEE Per Com 2005 Workshops*, 146–150.
18. L Batina et al. 2007. Public-Key Cryptography on the Top of a Needle. *ISCAS* 2007: I831–1834.