International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V73I12P102

Volume 73 Issue 12, 10-23, December 2025

Original Article

Fast and Low Power Implementation of RSA on Ternary Galois Field

Srividya B V[1], Nagarathna[2], Soumya S[3], Harikeerthan M K[4]

[1,2]*Department of Electronics and Telecommunication Engineering, Dayananda Sagar College of Engineering, Bengaluru, Karnataka, India*
[3]*Department of Electronics and Communication Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, India.*
[4]*Department of Civil Engineering, Dayananda Sagar Academy of Technology and Management, Bengaluru, Karnataka, India.*

[1]*Corresponding Author : srividyabv@gmail.com*

Received: 08 March 2025 Revised: 14 November 2025 Accepted: 25 November 2025 Published: 19 December 2025

Abstract *- Data transmission and reception are getting more expensive. As a result, it is now more challenging to store and maintain the security of information. This necessitates a secure connection in each application. One potential method for data security is cryptography. Besides defense-related uses, cryptography has many other uses in the modern world, including social networking, email, and e-commerce. Cryptography is a crucial element of pre-designed interfaces, such as embedded systems. The majority of cryptographic methods involve modular operations involving Galois Fields GF (p^m), where 'm' stands for power and 'p' for base, which are prime numbers. This research uses the Forced Stack approach, which has a low leakage power, to implement the Ternary Galois field. This work reports a low-power and high-frequency RSA cryptosystem to accomplish the goal. In order to implement the algorithm in hardware, Ternary circuits that are capable of efficiently carrying out the encryption, decryption, key generation, and key storage operations for the RSA algorithm have been designed using Cadence Virtuoso.*

Keywords - *Low power ternary VLSI circuits, Ternary galois field, RSA algorithm, Encryption, Decryption.*

1. Introduction

The RSA algorithm, a popular public-key cryptography scheme, requires a lot of computing power since it uses huge integers for operations like modular exponentiation. For modular arithmetic, it is observed that the Galois Field is the most widely used and prevalent application in cryptography. Every byte of data, in a finite field, is a vector by representation, since encryption and decryption are greatly simplified and manipulated using modular arithmetic. Using mathematical procedures to jumble up data that is a vector by representation in a Galois Field is simple and effective.

This research proposes hardware implementation of RSA over a Ternary Galois Field GF (3^m), (TGF), which is an intricate concept that combines finite field arithmetic and asymmetric cryptography, especially in ternary (base-3) systems. This method, in contrast to traditional RSA over integers, necessitates constructing the entire cryptosystem around polynomial arithmetic over a finite field, where all values are GF (3) elements. This idea is used to enable ternary computing architectures for non-binary cryptography systems. For embedded and high-performance systems, hardware implementation of RSA using Cadence tools offers a more effective and safe option than software-based methods.

1.1. Ternary Logic

Unlike traditional binary logic, which has two levels, ternary logic has three levels. Unbalanced Ternary digits, represented by the numbers 0, 1, and 2, are used in this article. Ternary data could accommodate larger data quantities in the same amount of storage as binary data. N trits are used to represent ternary data, if binary data is represented by N bits. For instance, if N = 10, there are 59049 Trits and 1024 bits. Ternary has the main advantage compared to binary, where each wire can transport more multivalued logical data while occupying less space on the chip. Because of the lower expected connection cost, it is given more weightage than binary.

1.2. Galois Field

In mathematics, Galois fields, also known as finite fields, are a concept of abstract algebra that deals with structures that are finite. It is a set of a finite number of elements that can be subjected to two rules-based operations, such as addition and

multiplication. The conditions for these operations keep the Galois Field closed; therefore, any action on such components will yield a result that also corresponds to the same set. Galois Fields are useful in many areas, such as error correction, coding theory, and cryptography, because of their mathematical properties. GF (p^m) accepts values like 2, 3, 5... Where 'p' is the prime number as well as the base value of the field, while 'm' stands for the power, which indicates the Galois Field size.

The characteristics of the Galois field are as follows:

- Commutativeness
- finite size
- closure
- associativity
- distribution
- Existence of identity & inverse elements.

Therefore, applications involving modular arithmetic are better suited for the use of Galois field elements.

1.2.1. Binary Galois field

The field elements of the Binary Galois field GF (2^m) are represented as a binary combination of the digits 0 and 1.

1.2.2. Ternary Galois Field (TGF)

Three integers, 0, 1, and 2, are used to represent the Galois field elements in Ternary in the TGF GF(3^m). Let us look at the TGF as an example. According to Table 1, GF (3^2) contains nine elements. Each of these components is shown in decimal, ternary, and polynomial form. The primitive polynomial p(x) used to construct the elements of GF (3^2) is chosen as

$$p(x) = x^2 + x + 2$$

Since 0, 1, and 2 are not roots of p(x), an assumption is made by choosing αone of them as the root of p(x)

$$p(x = \alpha) = \alpha^2 + \alpha + 2 = 0, \alpha^2 = -\alpha - 2$$

Considering the modular operation, the element $\alpha^2 = 2\alpha + 1$……. (2)

Table 1. Elements of $GF(3^2)$

Elements	Polynomial Representation	Vector form	Decimal representation
		Weightage = $\alpha^0\alpha^1$	Weightage = $3^0 3^1$
0	0	(00)3	(0)
1	1	(10)3	(1)
α	α	(01)3	(3)
α^2	$1 + 2\alpha$	(12)3	(7)
$\alpha^3 = \alpha^2 \times \alpha$	$2 + 2\alpha$	(22)3	(8)
$\alpha^4 = \alpha^3 \times \alpha$	2	(20)3	(2)
$\alpha^5 = \alpha^4 \times \alpha$	2α	(02)3	(6)
$\alpha^6 = \alpha^5 \times \alpha$	$2 + \alpha$	(21)3	(5)
$\alpha^7 = \alpha^6 \times \alpha$	$1 + \alpha$	(11)3	(4)

The goal of this research work is to use CMOS VLSI architecture for RSA encryption and decryption over Ternary Galois fields. This demands the use of low-power ternary logic gates, which are described in the sections to follow.

1.3. Implementing Logic Gates for TGF

The logic gates that operate on Ternary data are required to be designed and implemented for the construction of TED. These gates are implemented using multi-threshold CMOS gates.

Circuits that use ternary logic have three different logic levels. Logical 0, Logical 1, and Logical 2. The logic circuits that operate on Ternary data, discussed in this article, have three logic levels: logic 0 implemented at 0V, logic 1 implemented at 0.9V, and logic 2 implemented at 1.8V. All digital circuits, including Memory, Ternary encoders, Ternary decoders, multiplexers, Ternary ALU, Ternary microprocessors, and computers, are fundamentally built using logic gates such as Ternary AND, Ternary OR, Ternary inverters, Ternary NAND, Ternary NOR, and Ternary XOR. Due to the technical improvements, it is now possible to design and implement logic circuits that operate on Ternary data.

Several techniques, comprising memristor logic, CMOS logic, and CNTFET, are used for building digital circuits and ternary logic gates. The goal of this research effort is to build an RSA Cryptosystem that works on Ternary Galois field data, using multi-threshold CMOS Transistors.

1.4. CMOS design for reduced Leakage Power

Due to the superiority of dynamic power, leakage power was previously ignored. With the scalability of technology, leakage current is one of the main causes of cumulative IC currents. It has been enhanced significantly and is inevitable. While not in use or while no operations are being performed, the device endures static power loss.

The primary objective of this research is to provide alternative efficient low-power solutions for Very Large-Scale Integration (VLSI) designers. Techniques for reducing the leakage power of CMOS circuits are the main topic. The leakage power of 0.18µm technology is relatively low and roughly equivalent to the dynamic power usage. It is important to apply the techniques for the minimization of leakage power at the circuit level that operates on Ternary data. This article

focuses on the design and implementation of a low-power RSA cryptosystem that operates on Ternary Galois field data. All the relevant logic circuits that operate on unbalanced Ternary data are implemented in Cadence Virtuoso GPDK045 technology, using the concept of forced stacking.

2. Literature Review

Significant progress has been made in the area of integrated circuits in recent years. One of them makes use of two distinct signal levels and multivalued logic circuits. They are multipliers and memory. The aforementioned subject is the main focus of Kesta's work, which also gives an overview of MVL circuits by including details on their architecture and number representation. About 70% of a VLSI circuit's space is dedicated to connectivity, insulation is around 20%, and the devices take around 10%, according to V.T. Gaikwad. The interconnect takes up a significant amount of a VLSI device, which restricts the binary logic. This research has examined the designs of ternary-valued logic circuits in comparison to multivalued logic. In order to overcome interconnect-related restrictions in binary logic, V T Gaikwad et al. investigate the design of ternary logic gates in VLSI circuits. With an emphasis on 45nm technology, it simulates CMOS ternary logic gates utilizing Micro Wind EDA tools for possible VLSI implementation [1, 2].

The significance of ternary gates in both current and future technology is covered by the authors of [3, 4]. The injected voltage method for simulating ternary logic gates (TNOT, TNAND, and TNOR) is covered in the study. Using a suitable device, the power dissipation is measured. At the architectural, circuitry, and device levels, researchers have offered a variety of approaches to reducing leakage power.

2.1. Level of the Device

While taking into account the device level [5-7], the standby or leakage current is reduced by scaling the depth of the junction [W], the channel length [L], and its oxide thickness [Tox]. This needs to be completed before fabricating a transistor. As there is progress in the semiconductor field, several researchers have proposed heuristic geometries with multiple gates, such as FINFET, which contributes to the significant reduction of short channel effects and also dissipation of power (I2).

2.2. Circuit Level

Substrate biasing, sleep stacking, source biasing, LECTOR, force stacking, drain gating approach, GALOER, zigzag keeper approach, and sleepy-keeper approach are just a few of the leakage minimization techniques available at the circuit level. Other methods include Hybrid techniques such as MOS transistors with Double Threshold, Dynamic MOS, MOS transistors whose Threshold can be varied, and MOS transistors with more than one Threshold and selective input vector techniques (Vector Controlled, circuits with bootstrapped technique), and many more.

2.3. Architectural Level

Multiple modes management is a technique used at the architectural level to minimize leakage current by putting unused memory into sleep or standby states and permitting only a Static Random-Access Memory(SRAM) to remain ON [8, 9]. In comparison to the various techniques, the forced stack technique generates an extremely low static power. The principle behind the forced stack technique is that the dual transistors in the OFF state, when connected in series, provide a significant reduction in leakage power compared to using a single device [12]. When compared to the leakage of a single device, the stack's leakage current is incredibly low. However, this broadens the scope. A single-bit ternary multiplier using a Carbon Nanotube Field-Effect Transistor (CNTFET) is presented by author Erfan Shahrom [10]. The circuit in this paper is designed to enable VDD/2 in order to output directly logic '1', thereby eliminating direct current from the source to ground. VDD and VDD/2 are the two supply voltages used to do this. This is accomplished by appropriately partitioning the truth table and using two-level output gates. To approximate the result for lower Trit CNTFETs based Ternary full adders are used. [11] In the proposed research work, RSA over the Ternary Galois field is implemented, using Low-power Multi-Threshold circuits. In the RSA algorithm, exponentiation is the most important circuit. The shift-add technique was used by the authors of this research to suggest an exponentiation circuit. To enhance the operating frequency, the binary bit distribution technique is used in the implementation, where the most significant bit is eliminated. This research synthesized and simulated the HDL code [12].

The authors of [13] have used the 2048-bit based Montgomery and systolic array design to implement the reconfigurable RSA encryption and decryption technique to achieve a reduction in area and a significant increase in baud rate. Parallel approaches are used by the authors of [14] for factorization of very large integers, which is a computationally intensive process. In this research, the Ternary Galois Field RSA algorithm is implemented with leaf nodes that are designed using Multi-Threshold MOSFETs, resulting in a low-power cryptosystem.

3. Implementing Various Leaf Cells, Such as Ternary Logic Gates, as well as Circuits

This work employs multi-threshold transistors and the forced stack technique to build ternary logic gates. To save power, the most often used strategy is designing Multi-Vth MOS transistors in the form of a sleep transistor network, commonly referred to as the forced stacking technique. To substantially decrease the leakage power (static), high-voltage threshold transistors are utilized.

3.1. Ternary Inverter

Figure 1 depicts the suggested low-power ternary inverter that uses a multi-threshold transistor and a forced stack technique. This pictorial representation illustrates that the

Low Power Ternary Inverter employs two low threshold voltage pMOS in addition to nMOS and two high threshold voltage pMOS in addition to nMOS. The pull-up and pull-down networks of the forced stack inverter are the same width. Using the sleepy transistor technique, with a high Voltage threshold, Pull-down and pull-up circuits are employed. When the electronic gadget is in standby mode, these sleep trans-resistors are activated, which greatly reduces the current leakage. Both transistors are turned off at the same time in the OFF state, and due to the stacking effect, the reverse gate to source potential reduces leakage power.

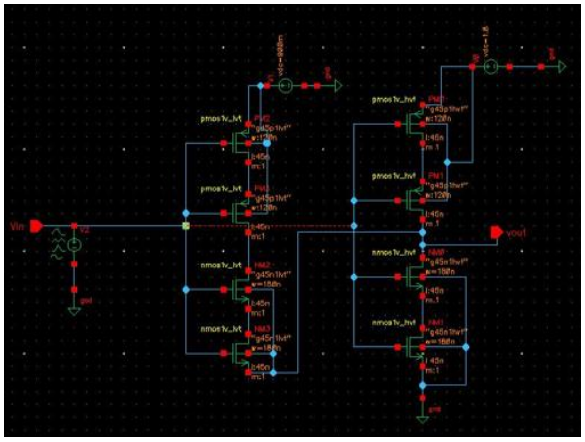

Fig. 1 Forced Stack (FS) ternary inverter based on MVT MOSFETs

According to the Truth table in Table 2, 1.8V, 0.9V, and 0V are the outputs that are obtained by forcing the inputs of 0V, 900mV, and 1.8V, respectively.

Table 2. Inverter (Ternary)

Input Vin;	Ternary Output	Output Voltage
0=(0V)	2	1.8V
1=(0.9V)	1	0.9V
2=(1.8V)	0	0V

Figure 2 displays the results of a Ternary inverter, which has low power and uses a forced stack technique along with multiple-threshold MOSFETs. Apart from the significant decrease in static leakage power, these outcomes are found to be identical to those of the Ternary Inverter with multi-threshold transistors.

After implementing the inverters that operate on Ternary data, with different design strategies, it has been found that, when compared to ternary inverters with resistive load and multi-threshold MOSFETs, the forced stack ternary inverter with multiple-threshold MOSFETs exhibits leakage power, around 582.125f Watt, which is the lowest among the other circuits in comparison. In comparison to Ternary inverters with the largest leakage power dissipation, the forced stack inverter with multiple-threshold MOSFETs exhibits a static leakage power of approximately 10.87%.

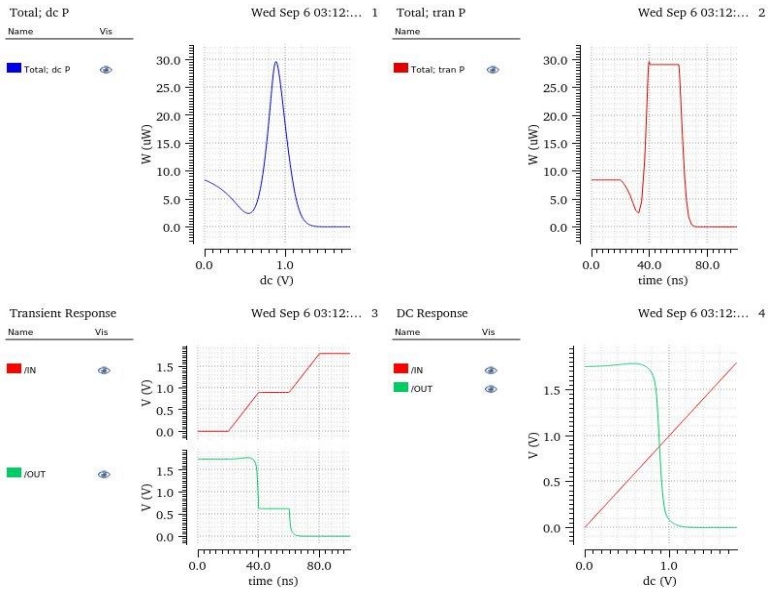

Fig. 2 Ternary inverter with forced stack approach

For this reason, the Forced Stack Multi-Threshold Transistor (FSMT) approach is employed to design all the Ternary Gates and Logic Circuits. Ternary AND, OR, NAND, NOR, Positive Threshold Inverter (PTI), Negative Threshold Inverter (NTI), Multiplexer, Decoder, Adder, Multiplier, and Galois Field adder have been designed and implemented using FSMT.

3.2. Ternary Gate

Figure 3 depicts the block diagram of a logic gate with a low-power technique that can operate on Ternary data utilizing forced-stack CMOS transistors. It consists of two stages, a network of pull-up and pull-down. Stage 1 is implemented using High Vth (hvt) MOSFETS, while stage 2 is constructed using Low Vth (lvt) MOSFETS. To pull the

output to the two extremities of logic zero and logic 2, stage 1 is used. Stage 2 is conducted when the output has to be at the intermediate level (logic 1). Likewise, multi-input low-power NAND, NOR, and XOR Ternary gates have been implemented accordingly. As an illustration, consider the implementation of a ternary NAND gate using multi-threshold transistors and the forced stack technique. For every input scenario listed in Table 3, the circuit's functionality has been verified.

$$T_{NAND} = \overline{min\{A, B\}} \quad (1)$$

Equation (1) states that the result of a ternary AND is the minimum of its inputs.

To achieve the NAND operation, the pMOS and nMOS networks are connected in parallel and series, respectively. The Ternary complement of the Ternary AND gate is provided by the Ternary NAND.

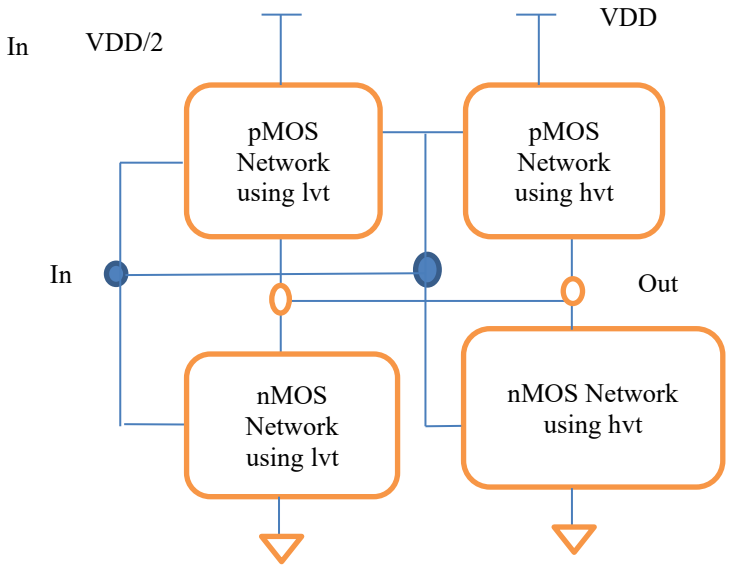

Fig. 3 Schematic of low-power FSMT ternary gate

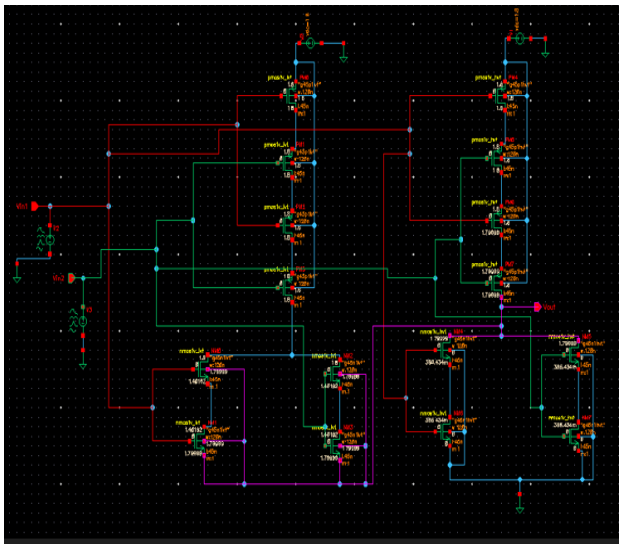

Fig. 4 Schematic of FSMT Ternary NOR gate

Each network consists of two pairs of MOSFETs connected in series and parallel, accordingly.

3.3. Ternary NOR Gate

Figure 4 illustrates the implementation of a Ternary NOR gate using multi-threshold transistors and the forced stack technique. This is in accordance with the block diagram of a Ternary gate depicted in Figure 3. For every input scenario listed in Table 3, the circuit's functionality has been confirmed.

$$T_{NOR} = \overline{max\{A, B\}} \quad (2)$$

Equation (2) states that the result of Ternary OR is the Maximum of its inputs. The Ternary complement of the Ternary OR gate is provided by Ternary NOR. To achieve the NOR operation, the pMOS and nMOS networks are connected in series and parallel, respectively. Table 3 provides the truth table for NAND and NOR, which operate on Ternary data.

Table 3. Ternary NOR Gate and Ternary NAND Gate

X	Y	TNOR	TNAND
0	0	2	2
0	1	1	2
0	2	0	2
1	0	1	2
1	1	1	1
1	2	0	1
2	0	0	2
2	1	0	1
2	2	0	0

3.4. Ternary- Decoder

The ternary decoder circuit (Figure 5) has one input and 3 outputs, which are either at 0 or 2. According to the truth table, the single input, which is in ternary, gets decoded into a unary output. As shown in Table 4, when Vin is 0, 1, or 2, the corresponding decoded outputs Vout0, Vout1, and Vout2 will be 2. In the expressions 4 and 5, the multipliers A^2, A^1, A^0, and B^2, B^1, B^0 are generated using decoders from single inputs of A and B having 3 different voltage levels.

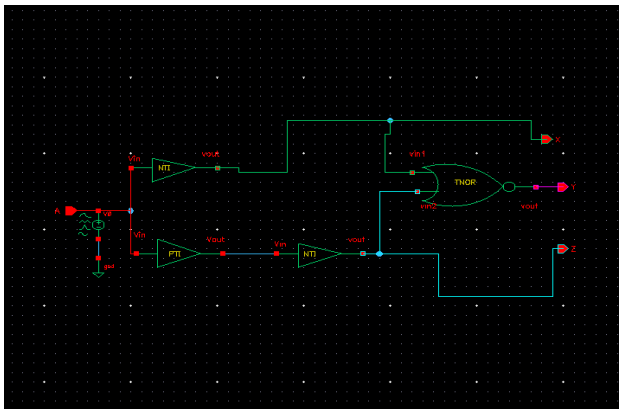

Fig. 5 Schematics of a ternary decoder

Table 4. Ternary decoder truth table

Input	V2	V1	V0
2	2	0	0
1	0	2	0
0	0	0	2

3.5. Multiplexer

In Ternary Logic circuits, the fundamental unit for building a Multiplexer is a Decoder. As an illustration, let us look into a 3:1 Multiplexer. This combinational circuit has 3 inputs X, Y, Z, a single input S for selecting among multiple inputs, and a single output Y. The equation of the multiplexer output Y is given by

$$Y = S_0 * X + S_1 * Y + S_2 * Z \quad (3)$$

Where S_0, S_1, and S_2 are the select lines corresponding to Ternary values with voltages of 0V (logic 0), 900mV (logic 1), and 1.8V (logic 2), respectively. The decoder is responsible for generating the select lines for the multiplexer.

Table 5 depicts the working truth table for a 3:1 Multiplexer.

Table 5. Ternary Multiplexer

Input	Select Lines(S)	Output(Y)
Ain	S0	Ain
Bin	S1	Bin
Cin	S2	Cin

Figure 5 depicts the logic circuitry for a multiplexer based on Ternary gates.

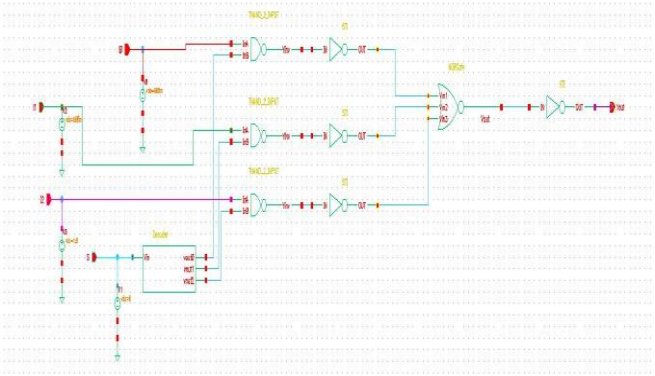

Fig. 6 Ternary multiplexer

3.6. Ternary Multiplier

Table 6 displays the truth table to realize the proposed 1-trit multiplier [15], where each input is forced with logic '0','1', and '2'. This truth table has 3^2=9 combinations in contrast to 2^2=4 combinations in the binary counterpart.

Table 6. Ternary Multiplier truth table and k-map

X	Y	Carry(3^1)	Product(3^0)
2	2	1	1
2	1	0	2
2	0	0	0
1	2	0	2
1	1	0	1
1	0	0	0
0	2	0	0
0	1	0	0
0	0	0	0

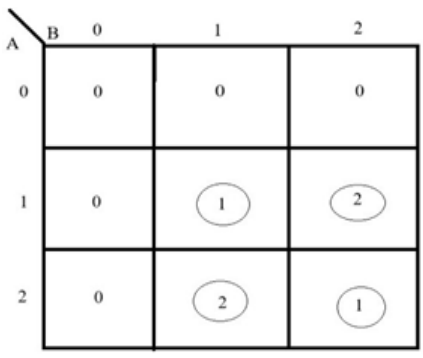

Fig. 7 Grouping the logic levels in a k-map

From the truth table, ternary logic expressions are realized for product (PP0) and carry (PP1) by grouping the logic levels in a k-map as shown in Table 6:

The expressions obtained for product (PP0) and carry (PP1) are as follows:

$$PP0 = A^2B^1 + A^1B^2 + 1.(A^1B^1 + A^2B^2) \quad (4)$$

$$PP1 = 1.(A^2B^2) \quad (5)$$

To implement ternary logic expressions 3 and 4, there is a necessity of decoded inputs, i.e., A^0, A^1, A^2, and B^0, B^1, B^2, which are generated by decoders. The following Figure 8 diagram shows the block diagram of a ternary multiplier, which requires TNAND, TNOR, STI, PTI, and NTI gates, as well as a Ternary decoder. These gates are realized with the Forced Stack technique to implement a low-power ternary multiplier as elaborated in earlier sections.

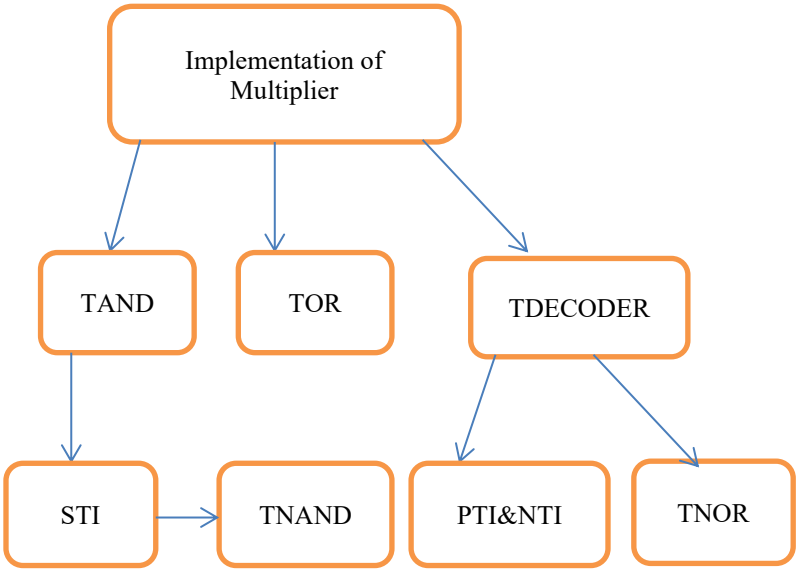

Fig. 8 Implementation of 1- Trit Multiplier

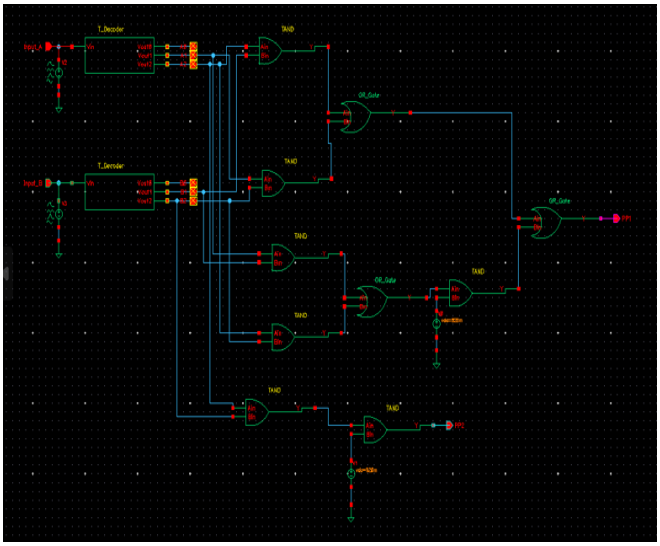

Fig. 9 Schematic of 1-trit Multiplier

Table 7. Half Adder operating on Ternary Data

X	Y	Cout(3^1)	Sum(3^0)
2	2	1	1
2	1	1	0
2	0	0	2
1	2	1	0
1	1	0	2
1	0	0	1
0	2	0	2
0	1	0	1
0	0	0	0

3.7. Ternary Half Adder

In Figure 10, the Ternary Half Adder is depicted. It can operate at logic levels 0, 1, or 2, and has two inputs, X and Y, and two outputs, Cout and Sum. As FSMT logic gates have been used, it is called a Low Power Ternary Half Adder. The truth table is shown in Table 7.

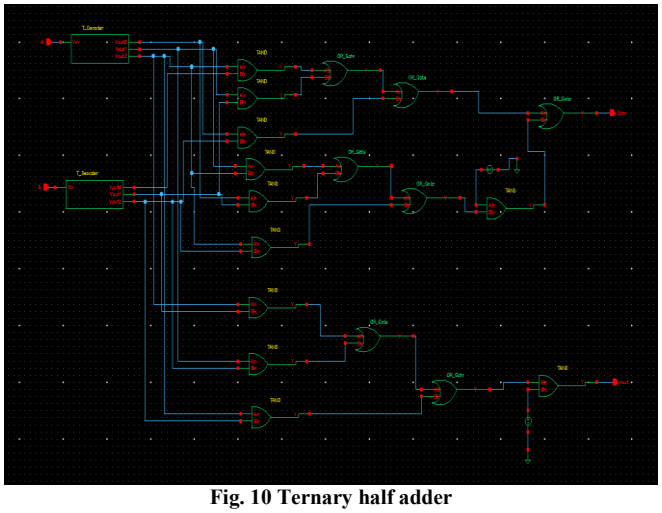

Fig. 10 Ternary half adder

3.8. Ternary Full Adder

In Figure 11, the Ternary Full Adder [11] is depicted. It can operate at logic levels 0, 1, or 2, and has three inputs, X, Y, and Cin, and two outputs, Cout and Sum. FSMT logic gates have been used to implement the circuit. For this reason, it is called a Low Power Ternary Full Adder.

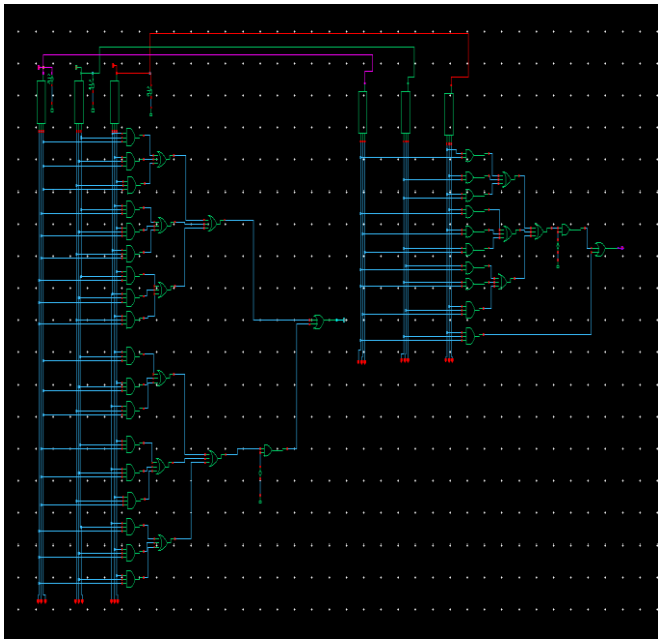

Fig. 11 Schematic of Ternary Full Adder

Table 8 presents the truth table for the Ternary Full Adder.

Table 8. Ternary full adder

X	Y	Cin	Cout(3^1)	Sum(3^0)
2	2	2	2	0
2	2	1	1	2
2	2	0	1	1
2	1	2	1	2
2	1	1	1	1
2	1	0	1	0
2	0	2	1	1
2	0	1	1	0
2	0	0	0	2
1	2	2	1	2
1	2	1	1	1
1	2	0	1	0
1	1	2	1	1
1	1	1	1	0
1	1	0	0	2
1	0	2	1	0
1	0	1	0	2
1	0	0	0	1
0	2	2	1	1
0	2	1	1	0
0	2	0	0	2
0	1	2	1	0
0	1	1	0	2
0	1	0	0	1
0	0	2	0	2
0	0	1	0	1
0	0	0	0	0

The proposed Forced Stack (FS) Ternary gates and Ternary Digital circuits are used as leaf cells for the implementation of the RSA Cryptosystem.

4. RSA Algorithm

RSA is a cryptographic algorithm that uses public keys [18, 19]. The RSA algorithm consists of three main steps: key generation, encryption, and decryption.

4.1. Key Generation

The key generation algorithm is used to produce the public key and private key. The two keys needed for the RSA algorithm $'e','d'$are the outputs of this technique, whereas p1 and p2 are the chosen inputs.

Choose two huge prime numbers, p1 and p2, of equal length in order to create these two keys.

Determine the function $\varphi(n)$ and the modulus number$n$.

Calculate $gcd(\varphi(n), n) = 1$ for a positive integer $'e'$such that. $1 \prec e \prec n$ In this context, $'e'$ it is referred to as the public key or encryption key.

Next, calculate the private key or decryption key$'d'$, such that, $1 \prec d \prec n$, so that,

$$ed\ mod(\varphi(n)) = 1 \quad (6)$$

The multiplicative inverse of the encryption key $'e'$ yields the decryption key$'d'$.

4.2. Encryption

The following algorithm can be used to calculate the RSA encryption process. Senders are required to accomplish modular exponentiation. First, obtain the recipient's public key (e, n) and the plaintext to be delivered, denoted by M. Following receipt of e, n, M encrypts the plaintext into a coded or cipher text and sends the cipher text to the recipient via the channel.

The Cipher Text

$$C = (M^e)\ mod\ n \quad (7)$$

4.3. Decryption

The following algorithm presents the decryption process. The algorithm must carry out the encryption process in reverse. To obtain the original plaintext, the recipient must decode the data using his private key (d, n). To accomplish this, compute the modular exponentiation of C with regard to the modulus n in order to decrypt the cipher text that was received from the sender.

The recovered plaintext is

$$M = (C)^d\ mod\ n \quad (8)$$

In this work, a method for encrypting and decrypting data using the RSA algorithm over the Ternary Galois Field is presented. All the circuits are designed and implemented using Low Power Forced Stack Multi Threshold Gates.

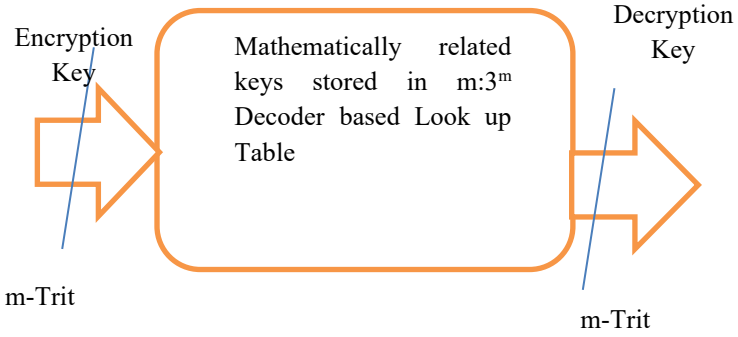

Fig. 12 Schematic of key generation in RSA

5. RSA Algorithm over Ternary Galois Field

5.1. Key Generation

Figure 12 shows the circuit related to the Key Generation algorithm. The Encryption Key $'e'$ is an m-Trit Ternary Value, which takes 0, 1, and 2 as its logic levels. The decryption key $'d'$is computed using the expression,

$$e \times d = 1\, mod(\,(3^m - 1)) \qquad (9)$$

A lookup table based on a Ternary decoder is used to store the values of 'd' that are computed using equation (9).

5.2. Storing a Message in the Accumulator

Figure 13 shows the circuit for storing the Message M in the Accumulator Register.

To begin with, an Accumulator Register is loaded with the Message M that needs to be encrypted.

The next section elaborates on the use of the accumulator register in encryption and decryption. Cascading m-Flip-flops, which are included in the Cadence Library, are used to build the Accumulator Register.

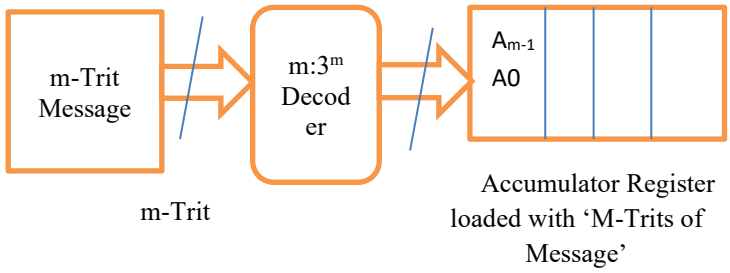

Fig. 13 Schematic of storing the message in the accumulator register

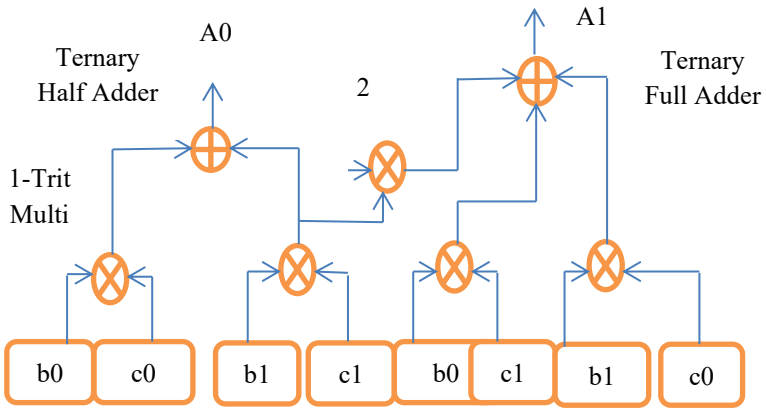

Fig. 14 Galois field multiplier

5.3. Multiplication Circuit

Figure 14 depicts the schematic for Galois Field Multiplication over GF (3^2). As an illustration, let us consider GF (3^2). In GF (3^2), the two numbers to be multiplied are 'b' and 'c'. Where $b = b0 + b1\alpha and c = c0 + c1\alpha$. The values of b0, c0, b1, and c1 are in Ternary (0, 1, and 2). The product of these two values

$$bc = (b0 + b1\alpha)(c0 + c1\alpha)$$
$$(b0c0 + b0c1\alpha + b1c0\alpha + b1c1\alpha^2)$$
$$(b0c0 + (b0c1 + b1c0)\alpha + b1c1\alpha^2)$$
Since $\alpha^2 = 2\alpha + 1$
$$bc = b0c0 + (b0c1 + b1c0)\alpha + b1c1(2\alpha + 1)$$

Rearranging the terms,

$$bc = (b0c0 + b1c1) + (b0c1 + b1c0 + 2b1c1)\alpha$$

The terms $b0c0, b1c1, b0c1, b1c0$ are obtained using a 1-trit modular multiplier. These terms are added using Modular Ternary Adders.

5.4. Encryption

RSA belongs to the class of public key cryptosystems, which involves two keys, one for encryption and its mathematically related key for decryption. The Ternary logic circuit for performing RSA Encryption is shown in Figure 15.

Encryption is performed using the encryption key 'e'.
While the decryption Key is 'd'

The relation between 'e' and 'd' is given by $e * d = 1\, mod(3^m - 1)$

The cipher text obtained after encryption is $C = M^e\, mod(GF(3^m))$

The plaintext can be recovered from the cipher text after performing Decryption. It is mathematically computed as $M = C^d\, mod(GF(3^m))$

As an Illustration, if e=5, then $C = (M)^5\, mod(GF(3^m))$

$$C = (M \times M \times M \times M \times M)\, mod(GF(3^m))$$

$(1)_3$ M

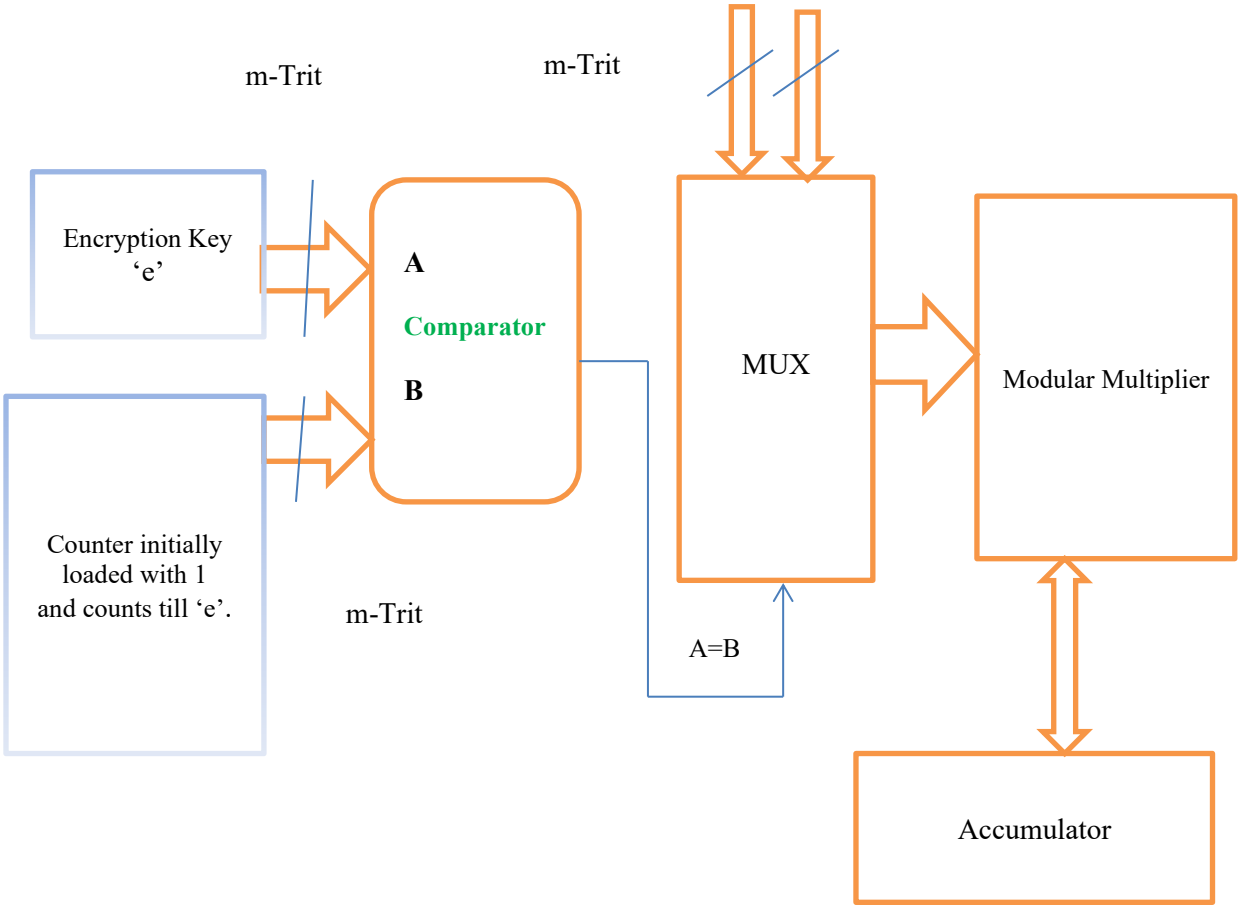

Fig. 15 Schematic of RSA encryption

As an Illustration, if e=5, then $C = (M)^5 \, mod(\, GF(3^m))$

$$C = (M \times M \times M \times M \times M) \, mod(\, GF(3^m))$$

Table 9. Encryption process

Counter	Action Performed	Accumulator Register
Initially	Accumulator Register A is loaded with M	Acc=M
1	• Comparing the Count value with 'e=5'. • Since the inputs to the Comparator are not equal, the Multiplexer selects the Message M and passes it to the modular multiplier. • Accumulator contents and M are multiplied. The computed product is stored in the Accumulator register. • Counter is incremented.	Acc=Acc*M=M^2
2	• Same as above	Acc=Acc*M=M^3
3	• Same as Above	Acc= Acc*M =M^4
4	• Same as Above	Acc= Acc*M =M^5
5	• Comparing the Count value with 'e=5.' • Since the inputs to the Comparator are equal, the Multiplexer selects input 1 and passes it to the modular multiplier. • Accumulator contents and Ternary 1 are multiplied. The computed product is stored in the Accumulator register. • The counter value is cleared.	Acc=Acc*1=M^5

As an Illustration, if the Message to be encrypted is $M = \alpha^3$, the encryption key e=5. Then the cipher text

$$C = M^e \, mod(\, GF(3^m))$$
$$C = ((\alpha^3)^5) \, mod(\, GF(3^m))$$

The values shown in Table 10 are the results obtained for a specific value of the Message M. The final Encrypted Cipher text is obtained in the Accumulator Register.

Table 10. Illustration of computing the Cipher text for $M = (\alpha^3)$

Counter	Action Performed	Accumulator Register(Acc)
Initially	Accumulator Register(Acc) is loaded with the value of M	$Acc = (\alpha^3)$
1	Acc=Acc*M	$Acc = (\alpha^3 \times \alpha^3 = \alpha^6)$
2	Acc=Acc*M	$Acc = (\alpha^6 \times \alpha^3 = \alpha^1)$
3	Acc=Acc*M	$Acc = (\alpha^1 \times \alpha^3 = \alpha^4)$
4	Acc=Acc*M	$Acc = (\alpha^4 \times \alpha^3 = \alpha^7)$
5	Acc=Acc*(1)	$Acc = (\alpha^7 \times \alpha^0 = \alpha^7)$

The Cipher Text $C = (\alpha^7)$

As per Table 10, the Cipher Text in Ternary is $C = (11)_3$

5.5. Decryption

The Ternary logic circuit for performing RSA decryption is shown in Figure 16.

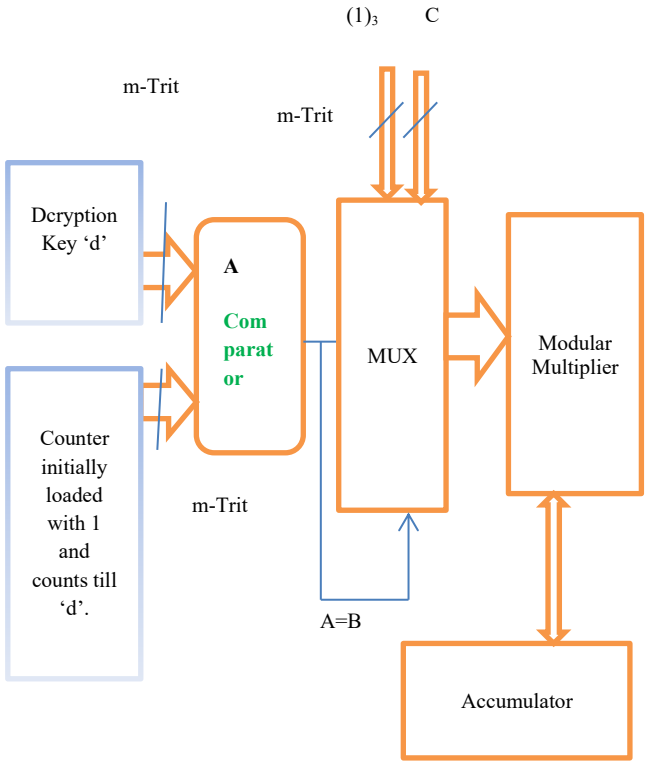

Fig. 16 Schematic of RSA decryption

Table 11. Decryption process

Counter	Action Performed	Accumulator Register
Initially	Accumulator Register A is loaded with Cipher Text C	Acc=C
1	• Comparing the Count value with 'd=5'. • Since the inputs to the Comparator are not equal, the Multiplexer selects the Cipher Text C and passes it to the modular multiplier. • Accumulator contents and C are multiplied. The computed product is stored in the Accumulator register. • Counter is incremented.	Acc=Acc*C=C^2
2	• Same as above	Acc=Acc*C=C^3
3	• Same as Above	Acc= Acc*C =C^4
4	• Same as Above	Acc= Acc*C =C^5
5	• Comparing the Count value with 'd=5'. • Since the inputs to the Comparator are equal, the Multiplexer selects input 1 and passes it to the modular multiplier. • Accumulator contents and Ternary 1 are multiplied. The computed product is stored in the Accumulator register. • The counter value is cleared.	Acc=Acc*1=C^5

As an Illustration,

The Cipher Text needs to be decrypted using the decryption key d=5. Then, the recovered plain text is $M = C^d \, mod(\, GF(3^m))$

$$M = ((\alpha^7)^5) \, mod(\, GF(3^m))$$

According to Table 12,

Table 12. Illustration of computing Plaintext/Message for $C = (\alpha^7)$

Counter	Action Performed	Accumulator Register(Acc)
Initially	Accumulator Register(Acc) is loaded with the value of Cipher Text C	$Acc = (\alpha^7)$
1	Acc=Acc*C	$Acc = (\alpha^7 \times \alpha^7 = \alpha^6)$
2	Acc=Acc*C	$Acc = (\alpha^6 \times \alpha^7 = \alpha^5)$
3	Acc=Acc*C	$Acc = (\alpha^5 \times \alpha^7 = \alpha^4)$
4	Acc=Acc*C	$Acc = (\alpha^4 \times \alpha^7 = \alpha^3)$
5	Acc=Acc*(1)	$Acc = (\alpha^{3} \times \alpha^0 = \alpha^3)$ Which is the same as the Original Message

The Recovered Message $M = (\alpha^3)$

According to Table 12, the Recovered Message $M = (22)_3$

The values shown in Table 12 are the results obtained for a specific value of the Cipher Text C. The final decrypted Cipher text is obtained in the Accumulator Register.

6. Results

Figure 17 shows the results obtained after performing encryption. The computations are based on GF (32). The message/Plaintext considered is$M = (\alpha^3) = (22)_3$. The Message M is (22) in Ternary. Ternary 2 is 1.8V. The Encryption Key $'e' = 5 = (21)_3$. Ternary 1 is around 0.9V. In the schematic, the Message(M) is denoted as M1 and M0, which are 1.8V. The encryption key is (12) in Ternary. The voltage levels are around 1.066V and 1.629V, respectively. The cipher text C is $(11)_3$. The voltage levels are obtained as 1.066V. This indicates the correctness of the results as per the values available in Table 10.

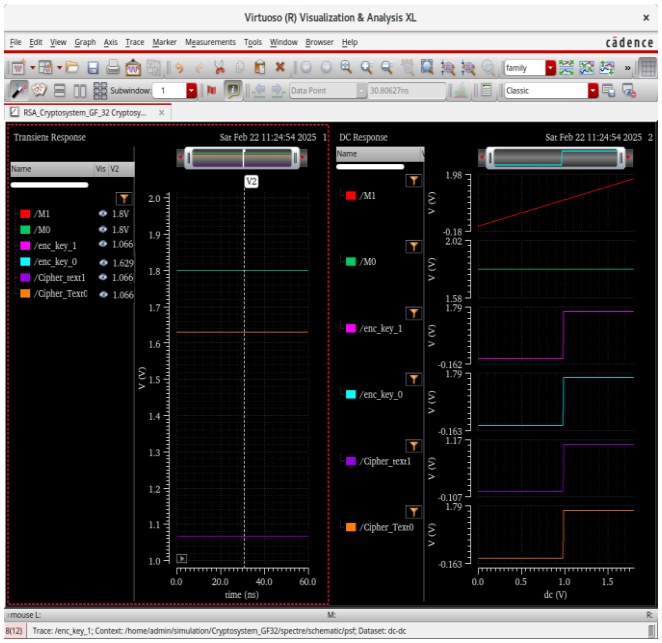

Fig. 17 Encryption results

Figure 18 shows the results obtained after performing decryption. The computations are based on GF (3^2).

The Cipher text considered is

$C = (\alpha^7) = (11)_3$. The Cipher Text is (11) in Ternary. Ternary 1 is around 900mV.

The decryption Key $'d' = 5 = (12)_3$. Ternary 1 is 0.9V, and Ternary 2 is 1.8V, respectively.

In the schematic, the Cipher text (C), denoted as Cipher_text1 and Cipher_text0 is around 1.066V. It is treated as Ternary 1. Upon performing decryption, Message (M), denoted as M1, M0 is obtained around 1.8V. The decrypted Message is $(22)_3$.

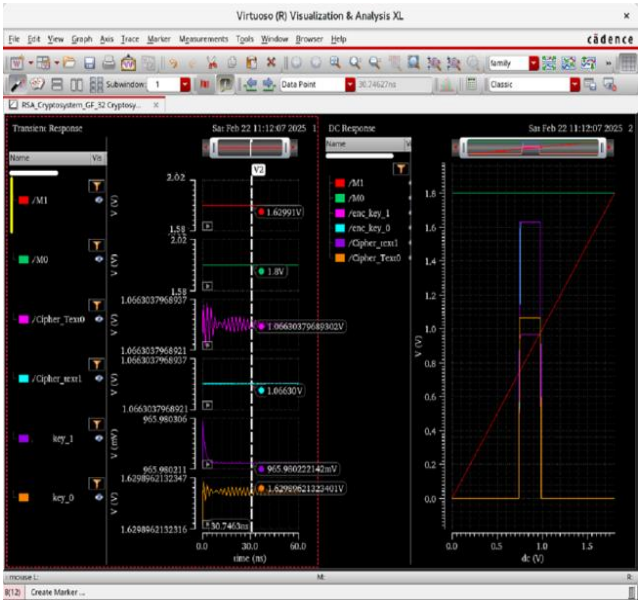

Fig. 18 Decryption Results

It can be seen from Table 13 that the Ternary multiplier outperforms the conventional Binary multiplier in terms of Transient power and DC power. Also, the authors of [20, 21] and according to the survey, many researchers have implemented RSA on FPGA and have provided the synthesized reports, which include the LUTs utilized, frequency, and the latency. This article proposes the architecture for the implementation of the RSA cryptosystem based on the Ternary Galois field using MOSFETs.

Table 13. Comparative analysis of DC and transient power (in µW)

Sl.No.	Circuit	Transient Power	DC Power
1	T-NOT	12.5	12.5
2	FSMT T-NOT	6	6
3	Multiplier(Binary)	185.44	144.55
4	Multiplier(Ternary)	108.89	121.24

7. Conclusion

Functional verification of every ternary logic circuit (as detailed in the paper) is carried out using the schematics described. In this study, a Ternary RSA cryptosystem with minimal power consumption is implemented. Modular exponentiation is the main focus. In order to get low power, Ternary multipliers are used to accomplish this. In order to create a power analysis table, the multiplier's DC power and transient power are measured. It is evident from Table 12 that ternary multipliers use less transient power and DC power than their binary equivalents. Ternary multipliers have only 81.5% of the overall power of their binary counterparts. Additionally, the Table shows that only 66.6% of the total ports needed in binary circuits are needed in ternary circuits to compute the product. This article discusses the low-power VLSI architecture for the RSA cryptosystem, which can operate on Ternary data. However, the work can be extended to adopt the delay calculations and thereby optimize the design.

References

[1] V.T. Gaikwad, and P.R. Deshmukh, "Implementation of Low Power Ternary Logic Gates using CMOS Technology," *International Journal of Science and Research (IJSR)*, vol. 3, no. 10, pp. 2221-2224, 2014. [Publisher Link]

[2] V.T. Gaikwad, and P.R. Deshmukh, "Design of CMOS Ternary Logic Family based on Single Supply Voltage," *2015 International Conference on Pervasive Computing (ICPC)*, Pune, India, pp. 1-6, 2015. [CrossRef] [Google Scholar] [Publisher Link]

[3] Sunmean Kim, Taeho Lim, and Seokhyeong Kang, "An Optimal Gate Design for the Synthesis of Ternary Logic Circuits," *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jeju, Korea (South), pp. 476-481, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[4] A.P. Dhande, Satish S. Narkhede, and Shridhar S. Dudam, "VLSI Implementation of Ternary Gates using Tanner Tool" *2014 2nd International Conference on Devices, Circuits and Systems (ICDCS)*, Coimbatore, India, vol. 137, pp. 1-5, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[5] A. Steegen et al., "65nm CMOS Technology for Low Power Applications," *IEEE International Electron Devices Meeting, 2005, IEDM Technical Digest*, Washington, DC, pp. 64-67, 2005. [CrossRef] [Google Scholar] [Publisher Link]

[6] K. Koh et al., "Highly Manufacturable 100nm 6T Low Power SRAM with Single Poly-Si Gate Technology," *2003 International Symposium on VLSI Technology, Systems and Applications, Proceedings of Technical Papers, (IEEE Cat. No.03TH8672)*, Hsinchu, Taiwan, pp. 64-67, 2003. [CrossRef] [Google Scholar] [Publisher Link]

[7] S. Zhao et al., "Transistor Optimization for Leakage Power Management in a 65nm CMOS Technology for Wireless and Mobile Applications," *Digest of Technical Papers, 2004 Symposium on VLSI Technology*, HI, USA, pp. 14-15, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[8] C.H. Kim et al., "PVT-Aware Leakage Reduction for on-Die Caches with Improved Read Stability," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 170-178, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[9] Krisztián Flautner et al., "Drowsy Caches: Simple Techniques for Reducing Leakage Power," *Proceedings 29th Annual International Symposium on Computer Architecture*, Anchorage, AK, pp. 148-157, 2002. [Google Scholar] [Publisher Link]

[10] Erfan Shahrom, and Seied Ali Hosseini, "A New Low Power Multiplexer Based Ternary Multiplier using CNTFETs" *AEU- International Journal of Electronics and Communications*, vol. 93, pp. 191-207, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[11] Aiman Malik, Md. Shahbaz Hussain, and Mohd. Hasan, "An Approximate Ternary Full Adder using Carbon Nanotube Field Effect Transistors," *2022 5th International Conference on Multimedia, Signal Processing and Communication Technologies (IMPACT)*, Aligarh, India, pp. 1-6, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[12] Sheba Diamond Thabah et al., "Fast and Area Efficient Implementation of RSA Algorithm," *Procedia Computer Science*, vol. 165, pp. 525-531, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[13] Chi-Chia Sun et al., "VLSI Design of a RSA Encryption/Decryption Chip using Systolic Array based Architecture," *International Journal of Electronics*, vol. 103, no. 9, pp. 1538-1549, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[14] Sapna Saxena, and Bhanu Kapoor, "State of the Art Parallel Approaches for RSA Public Key Based Cryptosystem," *International Journal on Computational Science & Applications (IJCSA)*, vol. 5, no. 1, pp. 1-8, 2015. [Google Scholar] [Publisher Link]

[15] Jongho Yoon et al., "Optimizing Ternary Multiplier Design with Fast Ternary Adder," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, no. 2, pp. 766-770, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[16] Xiao-Yuan Wang et al., "A Review on the Design of Ternary Logic Circuits," *Chinese Physics B*, vol. 30, no. 12, pp. 1-20, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[17] Ashwinikumar P. Dhande, Vijay T. Ingole, and Vikram R. Ghiye, *Ternary Digital System: Concepts and Applications*, SM Online Publishers LLC, 2014. [Google Scholar]

[18] Oded Goldreich, *Foundations of Cryptography: Basic Applications*, vol. 2, Cambridge University Press, 2001. [Google Scholar] [Publisher Link]

[19] Douglas Stinson, *Cryptography: Theory and Practice*, 2nd ed., CRC/C&H, 2002. [Google Scholar] [Publisher Link]

[20] Srinivasan Rajavelu et al., "Single Chip Efficient FPGA Implementation of RSA and DES or Digital Envelop Scheme," *WSEAS Transactions on Communications*, vol. 3, no. 2, 664-669., 2004. [Google Scholar] [Publisher Link]

[21] Rehan Fozia, Fozia Hanif Khan, and Mohammad Umair, "Cryptosystem an implementation of RSA using Verilog." *International Journal of Computer Networks and Communications Security*, vol. 1, No. 3, pp. 102-109. 2013. [Google Scholar]