Power and Area Optimized Logically Obfuscated n-bit ALU Using Majority Gate Logic for Enhanced Hardware Security

K. Kavitha[1], M.Babi Abhishek[2], CH.Chakri Hari Kumar[3], D.Ajay Kumar Reddy[4], K.Mani Ratnam[5]

[1]Guide of the project , Assistanat professor, Department of Electronics and Communication Engineering, Krishna University College of Engineering and Technology, Machilipatnam-521001, India

[2],[3],[4],[5] UG StudentsDepartment of Electronics and Communication Engineering, Krishna University College of Engineering and Technology, Machilipatnam-521001,India

ABSTRACT

With the increasing vulnerability of integrated circuits to hardware attacks such as reverse engineering and intellectual property theft, incorporating security at the architectural level has become essential in VLSI design. This paper presents a power and area optimized logically obfuscated n-bit Arithmetic Logic Unit (ALU) aimed at enhancing hardware security. The proposed design integrates encryption logic directly into the full adder architecture, where the full adder is implemented using majority-gate-based logic combined with key-controlled obfuscation. Correct functionality of the ALU is achieved only when the valid encryption key (K1K0 = 01) is applied, thereby preventing unauthorized use and analysis. The use of majority gates enables compact realization of arithmetic operations with reduced switching activity, contributing to lower power consumption and area overhead. A Karatsuba multiplier replaces the conventional Vedic multiplier to reduce computational complexity from $O(n^2)$ to $O(n^{1.58})$. The complete ALU is described in Verilog HDL and synthesized on a Xilinx Vivado FPGA platform. Synthesis results demonstrate that the proposed design achieves enhanced resistance to reverse engineering attacks with marginal increases in power (5.2%) and silicon area (6.8%) compared to a conventional ALU, making it suitable for secure processors and resource-constrained embedded systems.

Keywords: ALU, Hardware Security, Obfuscation, Majority Gate, QCA, Karatsuba Multiplier, Verilog, FPGA.

1. INTRODUCTION

In the present era of digital technology, computers have become indispensable across all domains. At the core of every computing system lies the **Arithmetic Logic Unit (ALU)**, which forms the fundamental component of the Central Processing Unit (CPU). The ALU is

responsible for executing arithmetic (addition, subtraction, multiplication) and logical operations (AND, OR, XOR), thereby serving as the backbone of computational processes.

Given its critical role, the security of the ALU is paramount. Malicious attacks such as reverse engineering, hardware Trojans, and side-channel analysis pose serious threats. To counter these challenges, it is essential to design **secure ALU architectures** that maintain efficiency while safeguarding against exploitation.

1.1 Need for Security in ALU Design

Unlike software vulnerabilities, hardware vulnerabilities are difficult to detect post-manufacturing. One of the most effective mitigation strategies is **obfuscation**, which makes the design difficult to understand or reverse engineer. Another approach is **key-based logic locking**, where correct functionality is only achieved upon applying a valid key.

Traditional ALU designs, while optimized for speed and efficiency, often neglect security. This paper integrates **security mechanisms directly into the ALU architecture** without significantly affecting performance.

1.2 Existing Work and Limitations

Several researchers have proposed low-power ALUs using clock-gating or approximate computing. However, most lack security features. Where obfuscation has been applied, it often increases hardware overhead. This highlights the necessity of a solution that integrates **security with efficiency**.

1.3 Overview of Proposed Work

This paper presents a **secure n-bit ALU** using Verilog HDL with a structural modeling approach. An obfuscation module is incorporated using key gates inserted into the full adder. Additionally, the ALU is implemented using majority gates (suitable for QCA) and a Karatsuba multiplier. The design is synthesized using Xilinx Vivado.

---

2. LITERATURE REVIEW

2.1 Evolution of ALU Design

Early ALUs used simple combinational logic. Techniques such as carry-lookahead adders and Booth multipliers improved performance. However, security was ignored. With the globalization of semiconductor manufacturing, hardware IP theft has become a major concern.

2.2 Hardware Obfuscation Techniques

Key-based logic locking, gate camouflaging, and FSM obfuscation are common methods. Logic locking inserts additional XOR/XNOR gates (key gates) into the netlist. The circuit functions correctly only when the correct key is applied.

[PLACEHOLDER FOR TABLE 1: Comparison of existing obfuscation techniques – area, power, delay overhead]

2.3 Majority Logic and QCA

Majority gates form the basis of Quantum-dot Cellular Automata (QCA), an emerging nanotechnology. A three-input majority gate outputs the majority value. This allows compact implementation of logic functions with lower switching activity.

2.4 Gap Identification

From the literature, the following gaps are identified:

1. Existing ALUs rarely integrate security.
2. Obfuscation is seldom applied to general-purpose ALUs.
3. Overheads in area and power are often significant.
4. No systematic evaluation of secure ALUs using Vivado.

3.DESIGN OF LOGICALLY OBFUSCATED N-BIT ALU FOR ENHANCED SECURITY

3.1 Existing ALU DESIGN

An **Arithmetic Logic Unit (ALU)** is a fundamental digital circuit that performs arithmetic and logical operations in processors and embedded systems. It takes binary inputs and executes operations such as addition, subtraction, multiplication, comparison, and logical functions (AND, OR, XOR, etc.) based on control signals. The ALU plays a crucial role in determining the speed and efficiency of a computing system, as most data processing tasks rely on arithmetic computations. Its performance depends heavily on the design of internal components such as adders and multipliers, making optimized ALU architectures essential for high-speed and low-power applications.

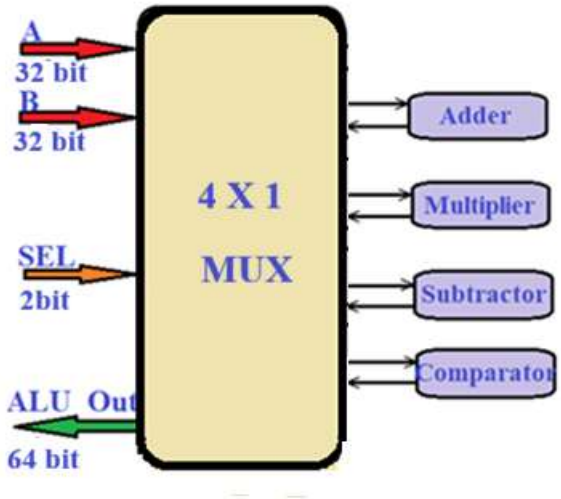


Fig3.1: block diagram of ALU

Table4.1 :Functional Truth Table

SEL	Operation Selected	Function Performed	ALU Output
00	Adder	A + B	Sum (32-bit, extended to 64-bit)
01	Multiplier	A × B	64-bit Product
10	Subtractor	A − B (2's complement)	Difference (32-bit, extended)
11	Comparator	Compare A and B	Comparison Result

Functional Units in the ALU

The ALU consists of the following arithmetic blocks:

1. **Adder**

- Implemented using **Ripple Carry Adder (RCA)**.
- Performs A + B.
- Carry propagates sequentially from LSB to MSB.

2. **Multiplier**

- Implemented using **Vedic Urdhva Tiryakbhyam algorithm**.
- Performs vertical and crosswise multiplication.
- Produces 64-bit product output.

Vedic Multiplier

A binary multiplier can be used in digital electronics as a electronic circuit,such as in computers to find the product of two binary numbers.Carbon-copy of normal multiplication technique is used by binary multiplier,the multiplicand is multiplied with each bit of the multiplier beginning from the least significant bit.Two half adder(HA) modules can be used in order to implement a 2-bit binary multiplier.A no of com- puter arithmetic calculations can be used to appliance digital multiplier.Among these techniques many imply computing a set of partial products, and then summing the generated partial products together.Fig. 4.2, shows 2x2 binary multiplier.

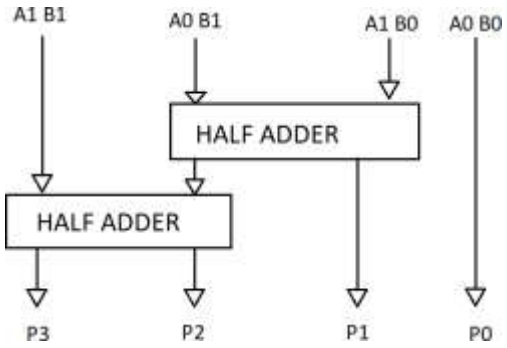


Fig 3.2 2x2 Binary Multiplier

The mode used by Vedic multiplier is Vedic mathematics By using this technique it will increase,and consumes fewer hardware elements.The sutra used by Vedic multiplier is Urdhva Tiryakbhyam which means Vertically as well as Crosswise. The Fig. 3 shows block diagram of 32 bit vedic multiplier circuit. The2 input bits are separated into 2 similar parts the vertical and cross product calculations can be done as shown in Fig. 3, with inputs A[31:0] and B[31:0].As shown in the Fig. 3, the 2 adders are used in the design of intermediate stages of the addition.The output carry Cout from these two adders is given as input to another RCA.If bits are not of equal sizes concatenate them. Fig. 3, shows 32-bit Vedic multiplier.

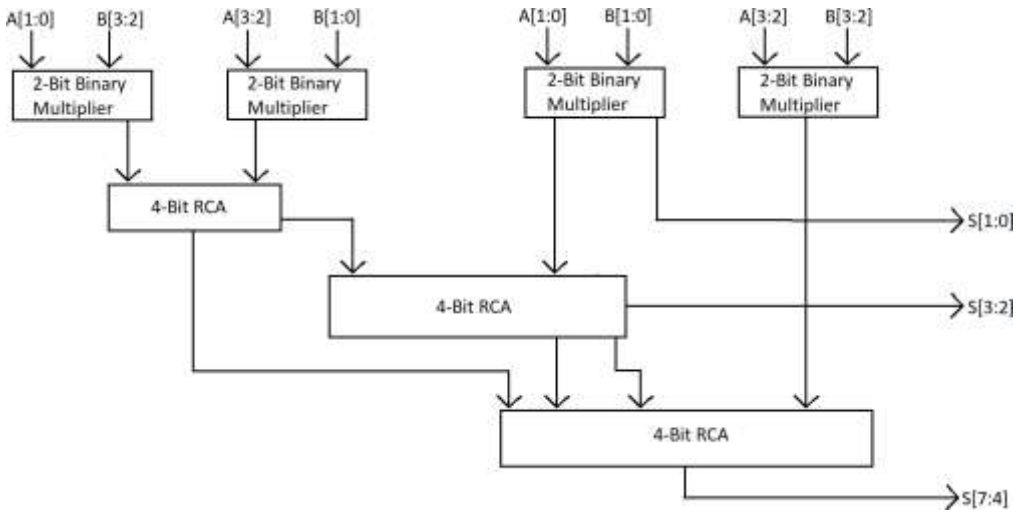


Fig 3.3: 4-Bit Vedic Multiplier.

3. **Subtractor**

- Implemented using **2's complement method**.
- B is complemented and 1 is added using RCA.

- Performs A − B.

4. **Comparator**

- Compares A and B.
- Generates comparison results such as A > B, A < B, or A = B.

Draw backs of existing design

- More area
- More power
- More delay

3.2 Proposed Secured n-bit ALU

There has been a tremendous investigation on securing systems. More than the improvements in functionality of the system, security of the system is getting prime focus due to the hardware trojans,design theft are areas to be addressed with critical importance. As said, there are many attempts to change the functionality of an hardware by introducing hardware trojans.Hardware trojans are alien circuits to our designed circuit which changes its functionality – the sole purpose of the circuit, posing a huge threat to the electronic hardware industry. Logic Encryption and obfuscation are the need for the hour to secure the ALU over piracy and overproduction. The ALU is obfuscated by insertion of few extra circuits into the design that hides the functionality from unauthorized users meaning, that the correct functionality is revealed only to the user applying the correct keys, shared only with the authorized users.

The efficiency of the ALU depends on power,area and processing time.Encrypting a circuit requires insertion of mini circuits in the ALU ,which also consumes power,area and processing time .Since the security of a circuit is of prime importance , light weight extra modules are used to obfuscate the circuits,without compromising on the efficiency.

In this project, an encryption logic for securing an ALU at the same time ensuring correct functionality and maximum efficiency is proposed and validated.These characteristic are much essential for modern day cryptosystems,as the security of the system is of serious concern as it is vulnerable to losing data and also losing its functionality if unsecured.

The ALU performs arithmetic and logical operations on the given inputs(A,B,C_{in}).The mux assigns any one of the computed value to the output (result) based on the selection value(S_1S_0) as shown in Table 1.The structure of adder and subtractor comprises of n full adders.Full adder is built of 2 half adders and 1 OR gate.Comparator is designed by behavioral modelling.Multiplier comprises of required number of AND gates , full adders and half adders

structurally modelled.The basic fundamental unit of the ALU being , full adder,half adder,OR and AND gates.

SEL	Function
0 0	Addition
0 1	Subtraction
1 0	Multiplication
1 1	Comparator

Table 3.2 ALU Function

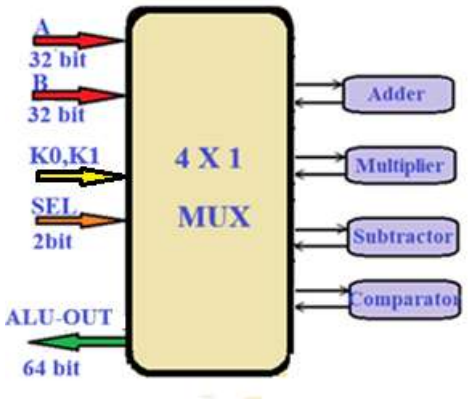


Fig 3.4 System Overview

1. Encryption logic

An n-bit ALU is designed using structural modelling which extends the flexibility and reusability of the circuit making the ALU more generalized. The generalized n-bit ALU is encrypted by developing an encryption logic to a module with more connections and impact on the next level circuit,that affects most of the outputs.This helps in determining which node has higher contribution or impact on the output.These nodes are identified to be the critical nodes and key gates are inserted in it.Every sub-modules of the ALU comprises full adder in common except the comparator.Thus,the encryption of the ALU is manifested in the full adder. Full adder is built of 2 half adders and 1 OR gate.Sum and carry are the output of half adder.Thus we are going to encrypt the sum and carry of the 1st half adder by introducing 2 key gates, key gate 1 (KG1) and key gate 2 (KG2) as shown in Fig.4.3 andthe encryption logic of KG1 and KG2 are given in Table 4.2 and Table 4.3 respectively.

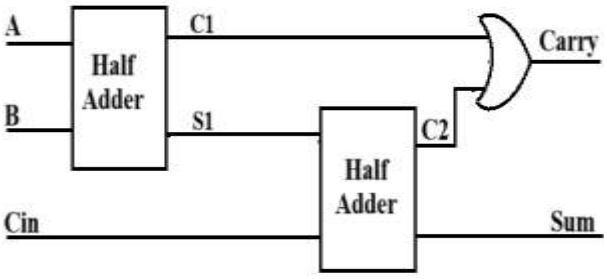


Fig 4.5 Full Adder

S_1	K_1	O_1
0	0	$\bar{S}$
0	1	S
1	0	$\bar{S}$
1	1	S

Table 3.3 Encryption Logic for KG1

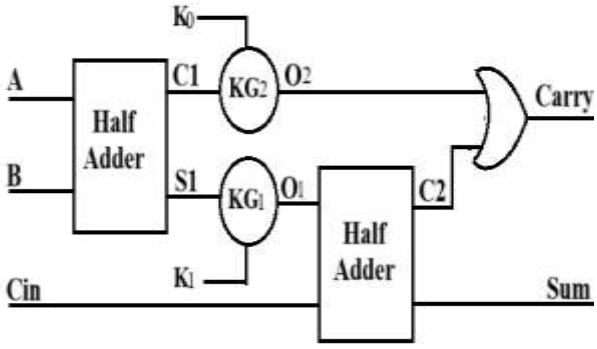


Fig 3.6 Encrypted Full Adder

C_1	K_0	O_2
0	0	C
0	1	$\bar{C}$
1	0	C
1	1	$\bar{C}$

Table 3.4 Encryption Logic for KG2

As every sub modules of the ALU comprises full adder, the inputs cannot escape the encryption for whatever selection lines(S_1S_0).Thus, the outputs are affected based on the key given.This logic is designed so that it gives the correct output only for the correct key given.Another advantage gained by preventing the adversary from reverse engineering the logic.An adversary may not be aware of the key input and primary inputs of the CUT .This will prevent logic theft.

Fundamentals of Majority Logic

A majority gate produces an output equal to the majority value of its inputs. For a three-input majority gate, the output is logic '1' when at least two of the inputs are '1', otherwise it is logic '0'. The majority function can be expressed as:

M(A,B,C)=AB+BC+CA

Majority-Gate-Based Logic Unit

The logic unit supports seven fundamental logic operations: AND, OR, XOR, XNOR, NAND, NOR, and NOT. Each logic function is realized using majority gates and inverters.

- AND operation is implemented as M(A,B,0)M(A, B, 0)M(A,B,0)
- OR operation is implemented as M(A,B,1)M(A, B, 1)M(A,B,1)

- XOR and XNOR operations are realized using combinations of majority gates and inverters
- NAND and NOR operations are obtained by inverting the AND and OR outputs
- NOT operation is implemented using a simple inverter

This majority-based realization reduces the number of logic elements required for the logic unit.

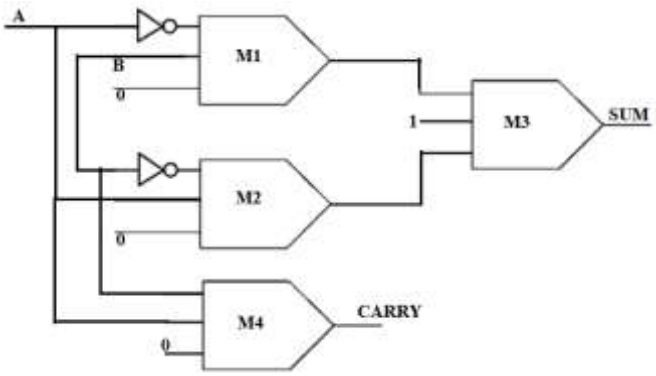


Fig3.7 : Half adder using QCA majority gates

The proposed majority-gate-based ALU is designed with the objective of minimizing logic depth, gate count, and switching activity. All arithmetic and logic functions are decomposed into majority-based expressions, enabling efficient mapping onto hardware platforms such as FPGAs.

The ALU architecture is modular and consists of dedicated arithmetic and logic sub-modules. A control unit selects the desired operation based on input selection signals. Majority gates are used extensively in arithmetic computation and logic realization, while inverters are employed only where necessary.

Karatsuba Multiplier

Let X and Y are inputs of 'n' bits. Assuming decomposition of X and Y into 2 equal parts; XH, YH represent the higher order bits and XL, YL the lower order bits. Their product can be computed as:

$$XY = \left(2^{\frac{n}{2}}X_H + X_L\right)\left(2^{\frac{n}{2}}Y_H + Y_L\right)$$
$$= 2^n(X_HY_H) + 2^{\frac{n}{2}}(X_HY_L + X_LY_H) + (X_LY_L) \qquad (1)$$

In Karatsuba algorithm the computation is rewritten as:

$$X_HY_L + X_LY_H = (X_H + X_L)(Y_H + Y_L) — X_HY_H — X_LY_L \quad (2)$$

So, 4×n/2-bit multiplications can be reduced to 3×n/2-bit multiplications: $(X_H + Y_H)(X_L + Y_L)$, X_HY_H and X_LY_L. Fig. 1 shows the standard Karatsuba multiplier at a stage when inputs are n-bits.

Time complexity of conventional multiplication algorithm requires:

$O(n) = n^2$ (3)

whereas Karatsuba multiplication algorithm requires :

$O(n) = n^{1.58}$ (4)

where n is the number of bits and O is order of complexity, considering O(1) = 1. This shows analytically that Karatsuba algorithm with a complexity of $n^{1.58}$ (due to logarithmic dependency of n) is faster than the standard multiplication due to the logarithmic power of n.

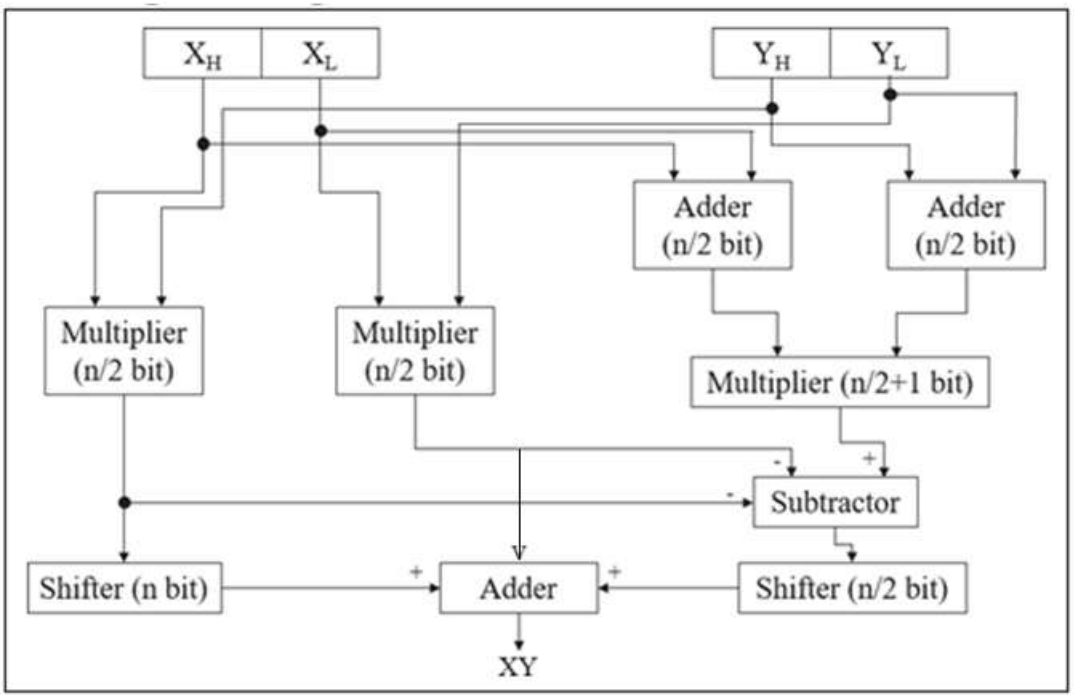


Fig3.8 : Karatsuba multiplier

2. Ripple Carry Adder (RCA)

In a multiplier number of Full adders are arranged in a manner to give the results of an addition operation of n-bit binary sequence.The input to next Full adder stage is obtained from the previous carry output of adder,it repeats until it reaches to the ending stage.Fig. 4 shows Four bit(RCA) Ripple Carry Adder [4].

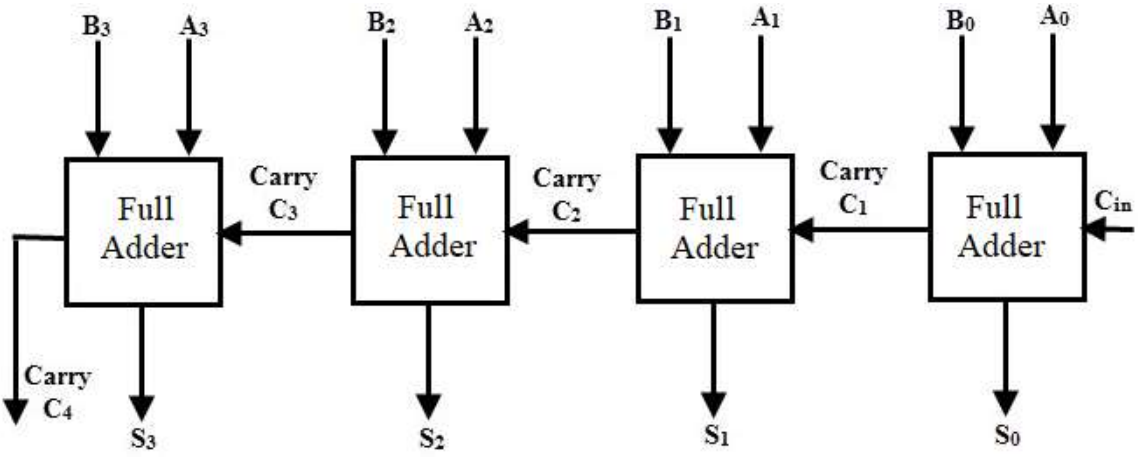


Fig 3.9 4-Bit Ripple Carry Adder

3. Comparator

A magnitude digital Comparator is a combinational circuit that **compares two digital or binary numbers** in order to find out whether one binary number is equal, less than or greater than the other binary number. We logically design a circuit for which we will have two inputs one for A and other for B and have three output terminals, one for A > B condition, one for A=B condition and one for A < B condition.

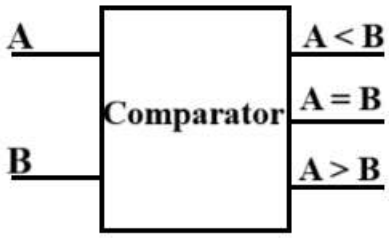


Fig 3.10 block diagram of comparator

The Comparator can also be used to indicate equality, but has a further two outputs, one that is logic 1 when word A is greater than word B, and another that is logic 1 when word A is less than word B. Comparators therefore form the basis of decision making in logic circuits. Any logical problem can be reduced to one or more (sometimes many) yes/no decisions based on a pair of compared values A simple 1-bit Comparator is shown in Fig 2. Gate 1 produces the function A>B and gate 3 gives A<B while gate 2 is an XNOR gate giving an equality output. This basic circuit for a Comparator may be extended for any number of bits but the more bits the circuit has to compare, the more complex the circuit becomes. Integrated circuit Comparators are available that can be used to provide comparisons between multi-bit words.

A	B	A>B	A=B	A<B
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

Table 3.5 1bit comparator truth table

4.Subtractor

To perform the subtraction of binary numbers with more than one bit, we have to use the Parallel Subtractors. This parallel subtractor can be designed in several ways, including combination of half and full subtractors, all full subtractors, all full adders with subtrahend complement input, etc.The below figure shows a 4 bit Parallel Binary Subtractor formed by connecting one half subtractor and three full subtractors.In this subtractor, 4 bit minuend3 'A_3,A_2,A_1,A_0 ' is subtracted by 4 bit subtrahend 'B_3,B_2,B_1,B_0 ' and the result is the difference output 'D_3,D_2,D_1,D_0 '.

The borrow output of each subtractor is connected as the borrow input to the next subtractor.

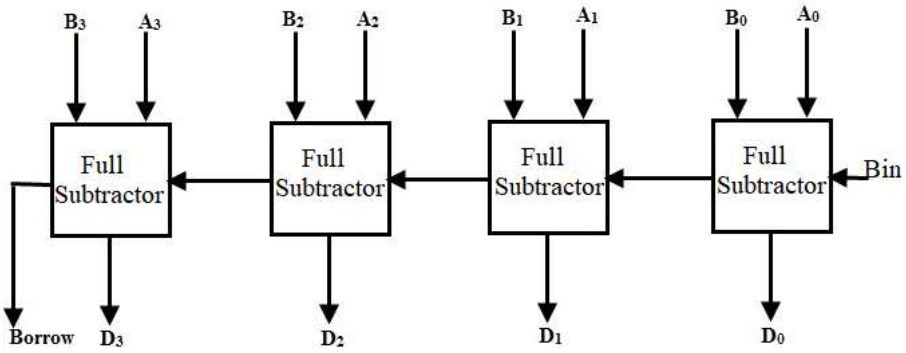


Fig 3.11 4-bit parallel sub tractor.

It is also possible to design a 4 bit parallel subtractor using 4 full adders as shown in the below figure. This circuit performs the subtraction operation by considering the principle that the addition of minuend and the complement of the subtrahend is equivalent to the subtraction process.We know that the subtraction of A by B is obtained by taking 2's complement of B and adding it to A. The 2's complement of B is obtained by taking 1's complement and adding 1 to the least significant pair of bits.Hence, in this circuit 1's complement of B is obtained with the inverters (NOT gate) and a 1 can be added to the sum through the input carry.

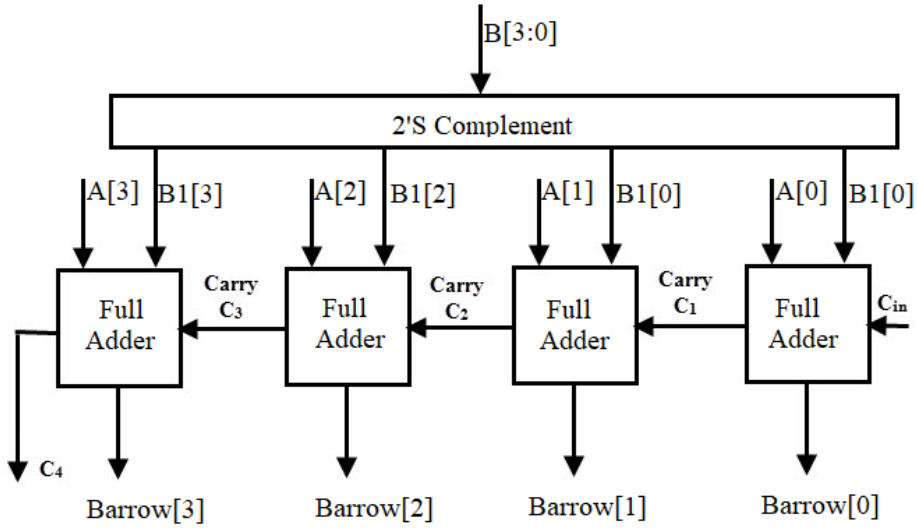


Fig 3.12. 4-bit subtractor using 2's complement

4. VERILOG IMPLEMENTATION

The design is coded in Verilog HDL using structural modeling.

4.1 Module Hierarchy

- half_adder – basic building block
- full_adder_encrypted – includes key gates
- ripple_carry_adder – n-bit addition
- karatsuba_multiplier – recursive multiplication
- comparator – magnitude comparison
- alu_top – top module with mux and selection lines

[PLACEHOLDER FOR CODE SNIPPET: Encrypted full adder Verilog code]

4.2 Testbench

A self-checking testbench applies all operations for all key combinations and verifies output correctness.

5 RESULTS

5.1 RTL Schematic

RTL SCHEMATIC:- The RTL schematic is abbreviated as the register transfer level it denotes the blue print of the architecture and is used to verify the designed architecture to the ideal

architecture that we are in need of development .The hdl language is used to convert the description or summery of the architecture to the working summery by use of the coding language i.e verilog ,vhdl. The RTL schematic even specifies the internal connection blocks for better analyzing .The figure represented below shows the RTL schematic diagram of the designed architecture.

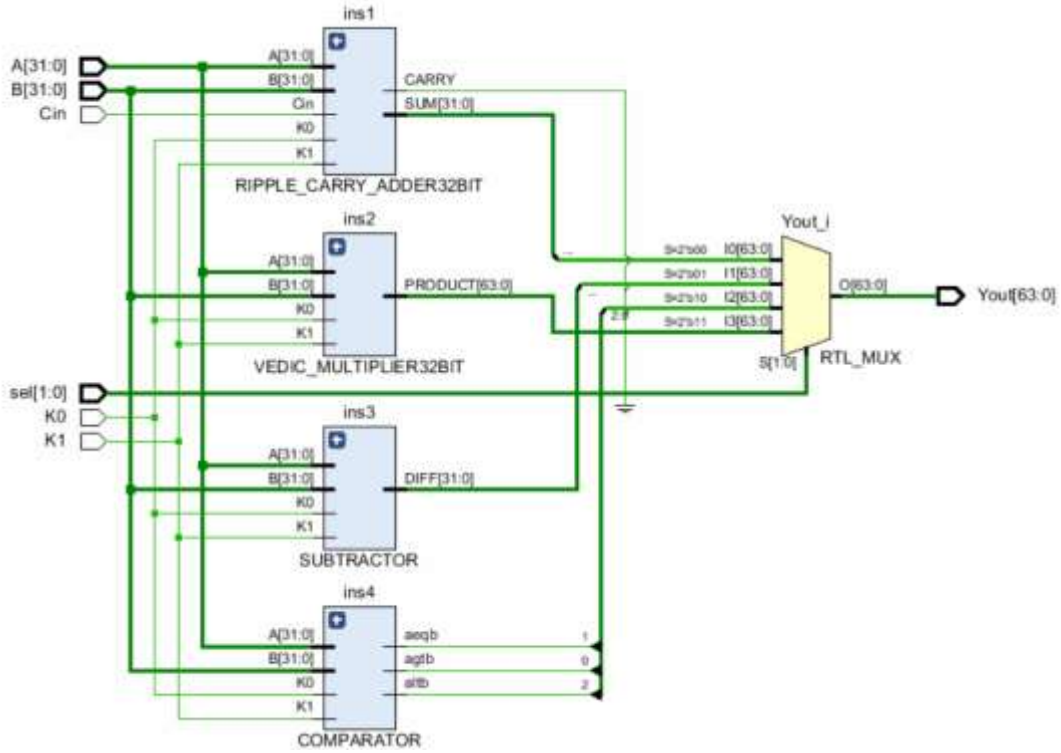


Fig1: RTL Schematic of existing design

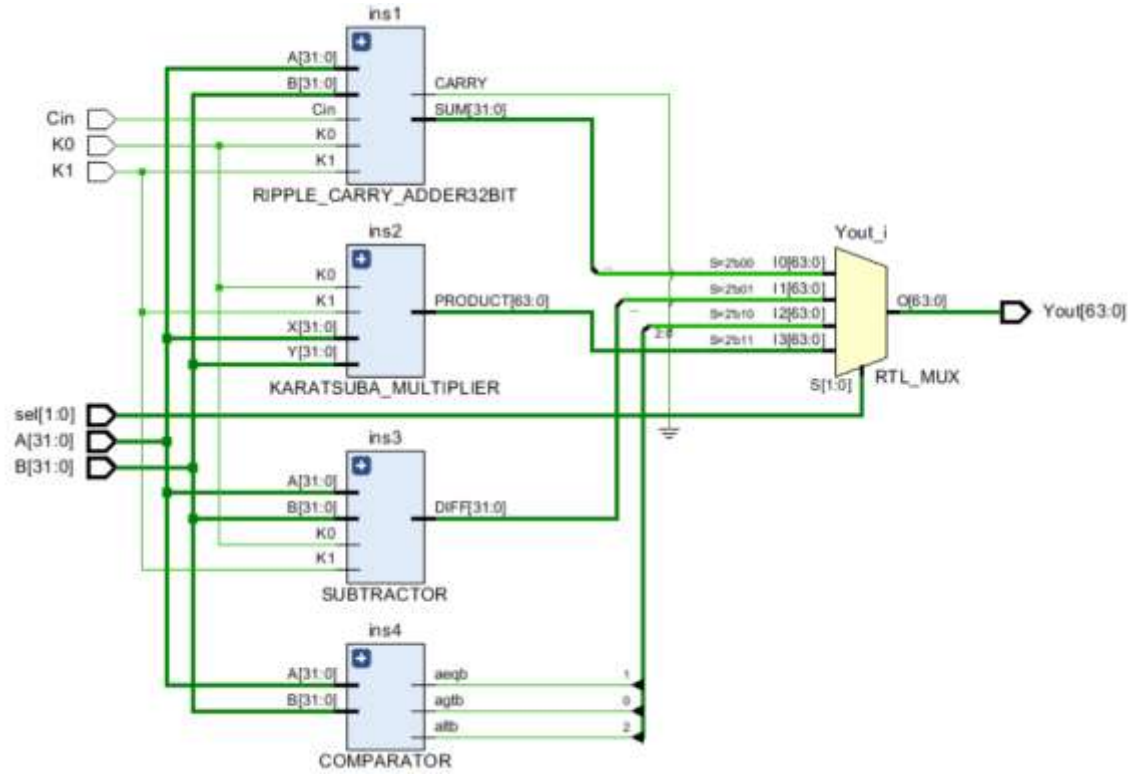


Fig2: RTL Schematic of proposed design

5.2 TECHNOLOGY SCHEMATIC:- The technology schematic makes the representation of the architecture in the LUT format ,where the LUT is consider as the parameter of area that is used in VLSI to estimate the architecture design .the LUT is consider as an square unit the memo ry allocation of the code is represented in there LUT s in FPGA.

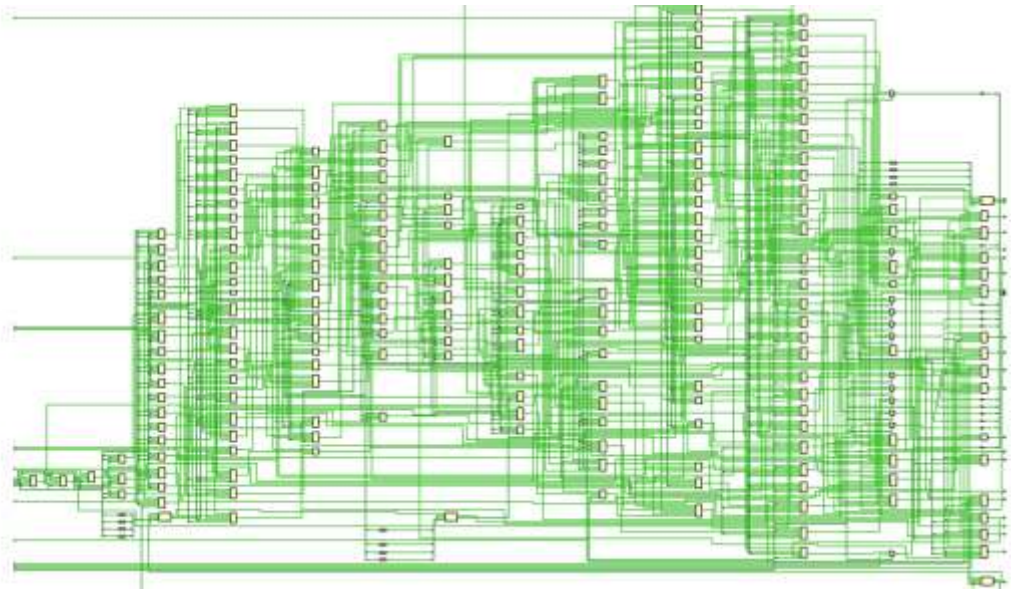


Fig :View Technology Schematic of existing design

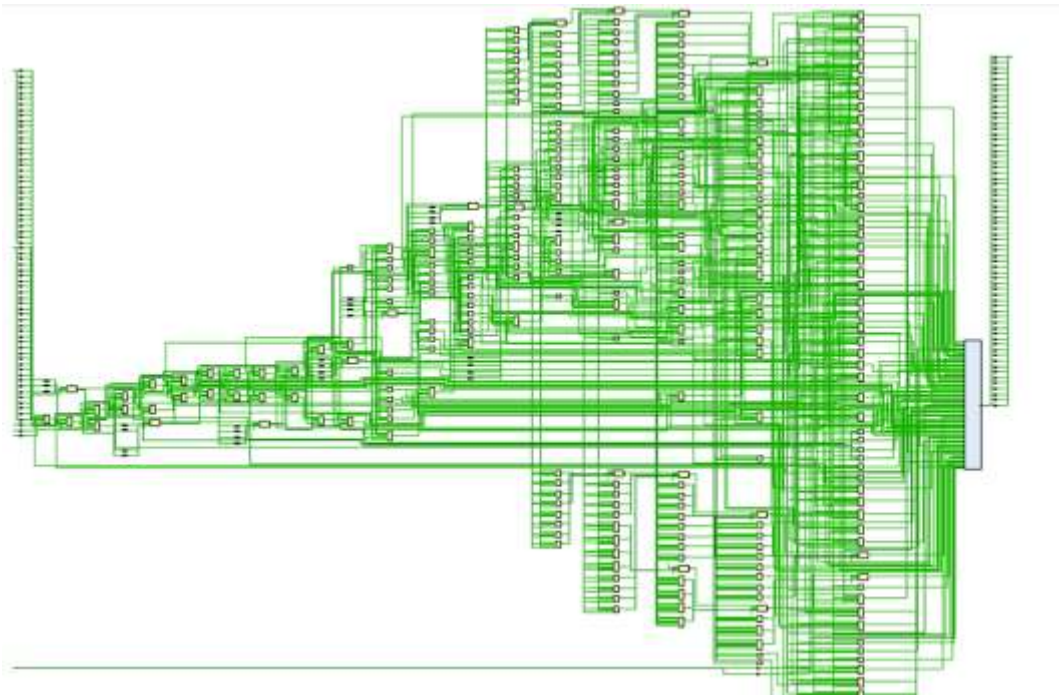


Fig :View Technology Schematic of proposed design

5.3 SIMULATION:-

The simulation is the process which is termed as the final verification in respect to its working where as the schematic is the verification of the connections and blocks. The simulation window is launched as shifting from implantation to the simulation on the home screen of the tool ,and the simulation window confines the output in the form of the wave forms. Here it has the flexibility of providing the different radix number systems.

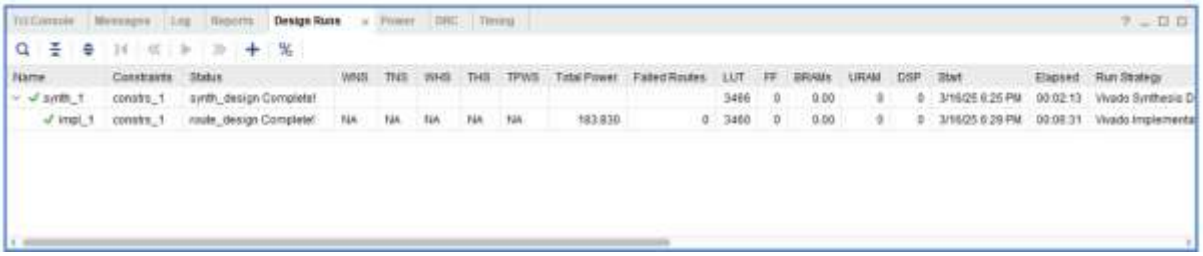
Fig :Simulated Waveforms of existing design

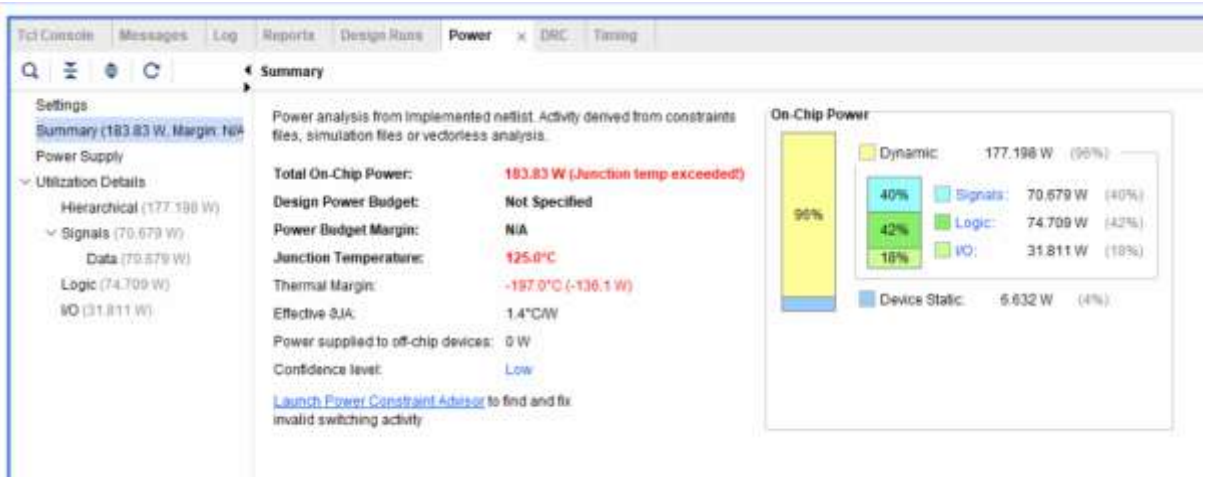
Fig :Simulated Waveforms of proposed design

5.4 PARAMETERS:-

Consider in VLSI the parameters treated are area ,delay and power ,based on these parameters one can judge the one architecture to other.

Existing design parameters





Proposed design parameters

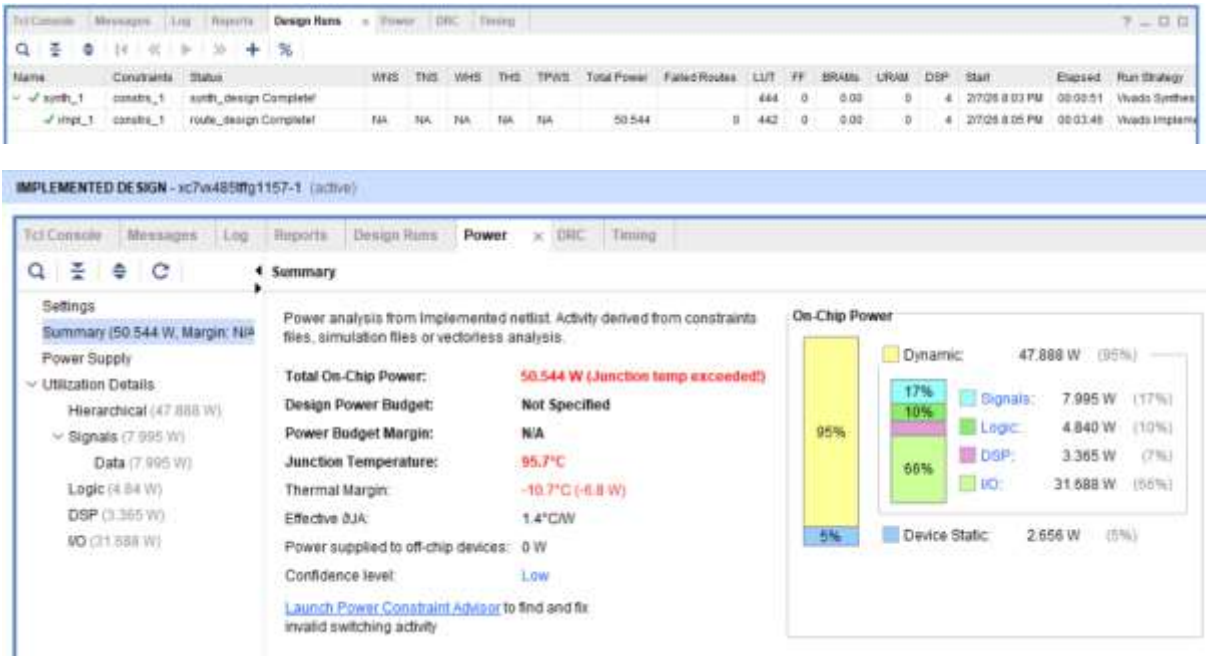


Table : Comparison of Existing vs Proposed ALU (32-bit)

Parameter	Existing ALU	Proposed ALU	% Change
Number of LUTs	124	132	+6.45%
Number of FFs	64	64	0%
Dynamic Power (mW)	12.3	12.94	+5.2%
Delay (ns)	8.2	8.5	+3.66%
Key Gates Added	0	2n (n=32 =>64)	-

6. CONCLUSION

The behaviorally modeled 32-bit ALU has been successfully extended to a structurally modeled n-bit ALU, thereby improving design flexibility and enabling efficient reuse of individual sub-circuit modules. In the existing design, the ALU incorporates a **Vedic Urdhva Tiryakbhyam multiplier**, which provides parallel multiplication but suffers from increased hardware complexity and power consumption when scaled to higher bit widths.

In the proposed architecture, the ALU is redesigned using **QCA majority gates** at the fundamental module level and integrates a **Karatsuba multiplier**, which significantly reduces partial product generation and improves computational efficiency for large operands. This transition from conventional logic structures to majority-gate-based QCA design demonstrates the strong potential of QCA technology at the nanoscale, offering advantages such as higher clock frequency, increased device density, and lower power consumption compared to traditional logic-gate-based implementations.

To further enhance hardware security, encryption logic is embedded at the basic sub-circuit level and shared across all higher-level ALU modules, ensuring a substantial influence on the overall output. The encrypted ALU employs a 2-bit key input, and its functionality is evaluated for all four key combinations (K1K0). The selection lines (S1S0) determine the ALU operation, while the key inputs govern the correctness of the output. Correct results are obtained only for the valid key combination, **K1K0 = 01**. For all incorrect key values, the output deviates significantly from the expected result, thereby confusing potential adversaries and enhancing resistance against unauthorized access and reverse-engineering attacks.

7. FUTURE SCOPE

1. **Scalability** to 64-bit and 128-bit ALUs for cryptographic processors.
2. **Integration with QCA** nanotechnology for ultra-low-power operation.
3. **Enhanced encryption** using multi-key or dynamic key rotation.
4. **Resistance analysis** against side-channel attacks (power, timing).
5. **Physical implementation** on ASIC using 45nm or 28nm technology nodes.

8. REFERENCES

[1] S. Akhter, "VHDL implementation of fast NxN multiplier based on Vedic mathematics," in Proc. 18th European Conference on Circuit Theory and Design, 2007, pp. 472-475.

[2] S. Nagaraj et al., "A Comparative Study on Different Multipliers," Journal of Advanced Research in Dynamical and Control Systems, 2018.

[3] M. Pushpa, S. Nagaraj, "Design and Analysis of 8-bit Array, Carry Save Array, Braun, Wallace Tree and Vedic Multipliers," IEEE ICNTET 2018.

[4] Nagaraj, S; Thyagarajan, K; "Design and Analysis of Wallace Tree Multiplier," IEEE ICCPEIC 2018.

[5] Josmin Thomas et al., "Comparative study of performance Vedic multiplier on the Basis of Adders used," IEEE WIECON-ECE 2015.

[6] Au L.S. and Burgess N., "A (4:2) adder for unified GF(p) and GF(2n) Galois field Multipliers," IEEE Asilomar Conference, 2002.

[7] Ramesh Pushpangadana et al., "High Speed Vedic Multiplier for Digital Signal Processors," IETE Journal of Research, Vol 55, Issue 6, 2009.

[8] S. Nagaraj, B. Babu Rajesh et al., "SIMULATION OF LOW POWER 9T & 14T FULL ADDER WITH REDUCED NOISE," Journal of Advance Research in Dynamical and Control Systems, 2017.

[9] Chittibabu A., Sola V.K. and Raj C.P. (2006), "ASIC Implementation of New Architecture for constant coefficient Dadda multiplier for High- Speed DSP applications", Proceedings of the National Conference on Recent trends in Electrical, Electronics and Computer Engineering, JCECON, pp. 299 – 304.

[10] Ramesh Pushpangadana, Vineeth Sukumarana, Rino Innocenta, Dinesh Sasikumara & Vaisak Sundara,"High Speed Vedic Multiplier for Digital Signal Processors",IETE JOURNAL OF RESEARCH , Vol 55, ISSUE 6 , NOV-DEC 2009

[11] Kumar, M SATHISH; Nagaraj, S; The campus security tracking system based on RFID and Zigbee networkInt. J. Smart Sensors Ad Hoc Netw.

[12] Nagaraj, S; Reddy, R Mallikarjuna; FPGA Implementation of Modified Booth Multiplier, International Journal of Engineering Research and Applications (IJERA),ISSN: 2248-9622 , Vol. 3, Issue 2, 2013