IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Majority Logic Gate Implementation Of Aes Algorithm

[1]Vineela Nagabathula, [2]Prasanth Varasala,

[1]M.tech Scholar, [2]Head of the Department,
[1]Electronics and Communication Engineering,
[1]Pragati Engineering College,

Abstract: In this study QCA based encryption standard is implemented in which and gates and xor gates are designed using majority logic gates and not gates. QCA technology has a great advent in the upcoming years. In AES the key operation is done by xor gates. In this 128 bit AES encryption and decryption is implemented using verilog coding and the same is synthesized and simulated using the Xilinx vivado tools.

Index Terms **–** AES, FPGA, QCA

I. INTRODUCTION

Data security can be accessed by safe communication channels, data coding technology and trusted third parties. In this article, AES –Algorithm was developed to provide expensive data. One of the most attractive and most traders is QCA. QCA, to develop high performance digital scenarios. For this reason, QCA has paid more attention in recent years.

A square nanostructure of electron wells with free electrons is called a quantum-dot cellular automaton (QCA). There are four quantum dots in each cell. The four corners [1] contain the four dots. Two free electrons can be used to charge the cell. The electrons tunnel to their correct spot during the clock changeover by use of the clocking mechanism. Majority logic gates and inverter gates make up the internal architecture. These gates are used to implement the AES algorithm.

This is how the remainder of the paper is structured: The literature study is covered in Section II, and the AES algorithm is briefly described in Section III. The AES encryption design employing the QCA approach is shown in Section IV. Results are displayed in Section V, and a comparison table is presented in Section VI.

II. LITERATURE REVIEW

The authors Yu.W and S. Köse detail the implementation of a modified lightweight AES algorithm in FPGA, utilizing operations like mix columns, shift rows, and substitution bytes in parallel. The hardware implementation of AES offers enhanced security and entails lower costs, along with reduced hardware utilization.

The authors S. Bri and S. Oukili employed a 5-stage pipeline design to achieve maximum frequency and enhance speed.[5].

Berna et al. [7] conducted a control examination attack on an ASIC AES implementation. They focused on side-channel attacks, which make it challenging to retrieve the secret information.

K. Kong, Y. Shang, and R. Lu introduced [8] a highly efficient method for majority logic synthesis of arbitrary Boolean functions. Approach yields a minimal majority expression and an optimal QCA layout for any specified three-variable Boolean function, aiming to produce high-quality decomposed Boolean networks is demonstrated.

III AES

The Advanced Encryption Standard (AES) is a symmetric block cipher that includes three variations: AES-128, AES-192, and AES-256[2]. Depending on the key length employed, the algorithm is executed in 10, 12, or 14 rounds respectively. Its structure is founded on the AES-128 Encryption.

A block of 128 data bits, along with a 128-bit cipher key, undergoes processing. Each round operation consists of four transformations: ADD Round Key, SHIFT Rows, SUB BYTES, and MIX Columns. The initial operation, ADD Round Key, involves performing an XOR operation between the 128-bit keys and the data, as illustrated in the figure.

Additionally, the final round of the algorithm is distinct from the prior nine rounds because it omits the Mix Columns transformation. Conversely, the cipher key is processed through a Key-Schedule operation, which expands it into 10 keys for the 10 rounds of the algorithm. In the suggested approach, QCA [9] logic is utilized, employing majority logic gates.

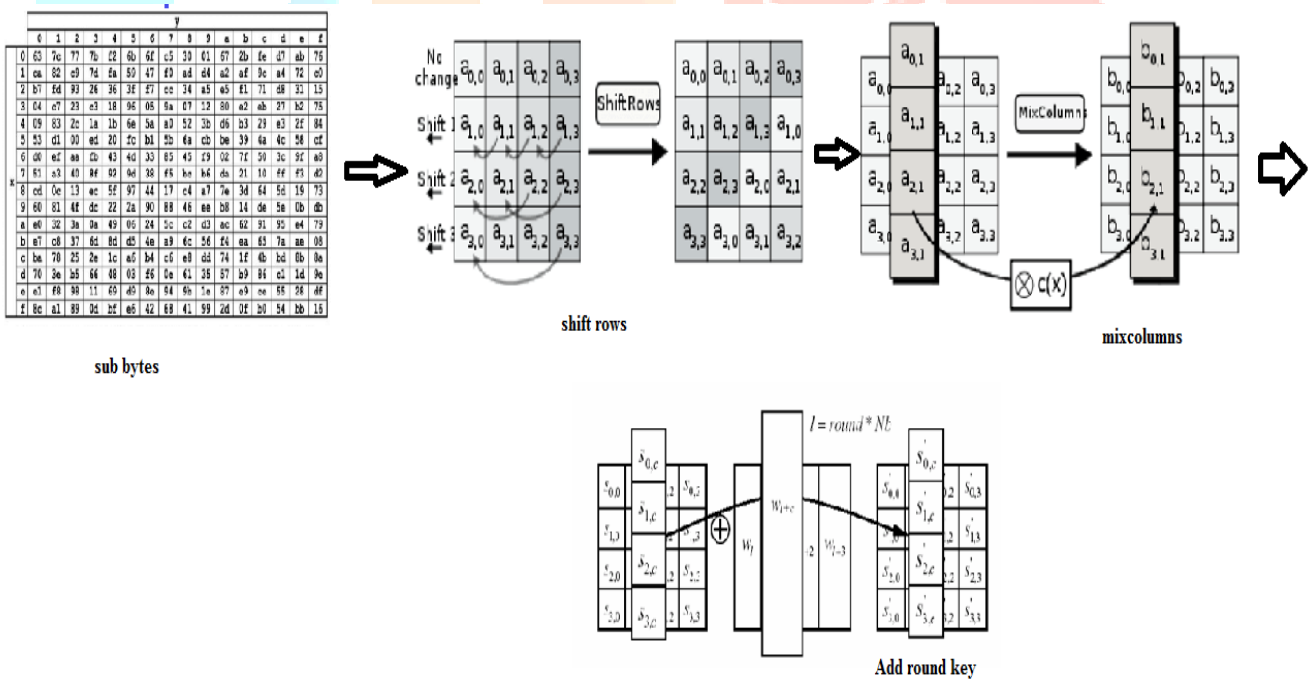


Fig 1: Steps involved in AES algorithm

IV PROPOSED METHOD USING MAJORITY LOGIC GATES

In the proposed method majority logic gates are used for implementing the XOR operation. It requires three majority logic gates and two not gates for satisfying the xor operation condition. For the initial majority logic gates one of the input is given as constant ‘0’ value and for the other majority late the second input is given as constant ‘1’. The below figure shows the steps involved in the AES encryption.

The input and output for the AES algorithm consists of sequences of 128 bits. These sequences are referred to as blocks and the numbers of bits they contain are referred to as their length. The Cipher Key for the AES algorithm is a sequence of 128, 192 or 256 bits. Other input, output and Cipher Key lengths are not permitted by this standard. In this project 128 bit key and input is considered.

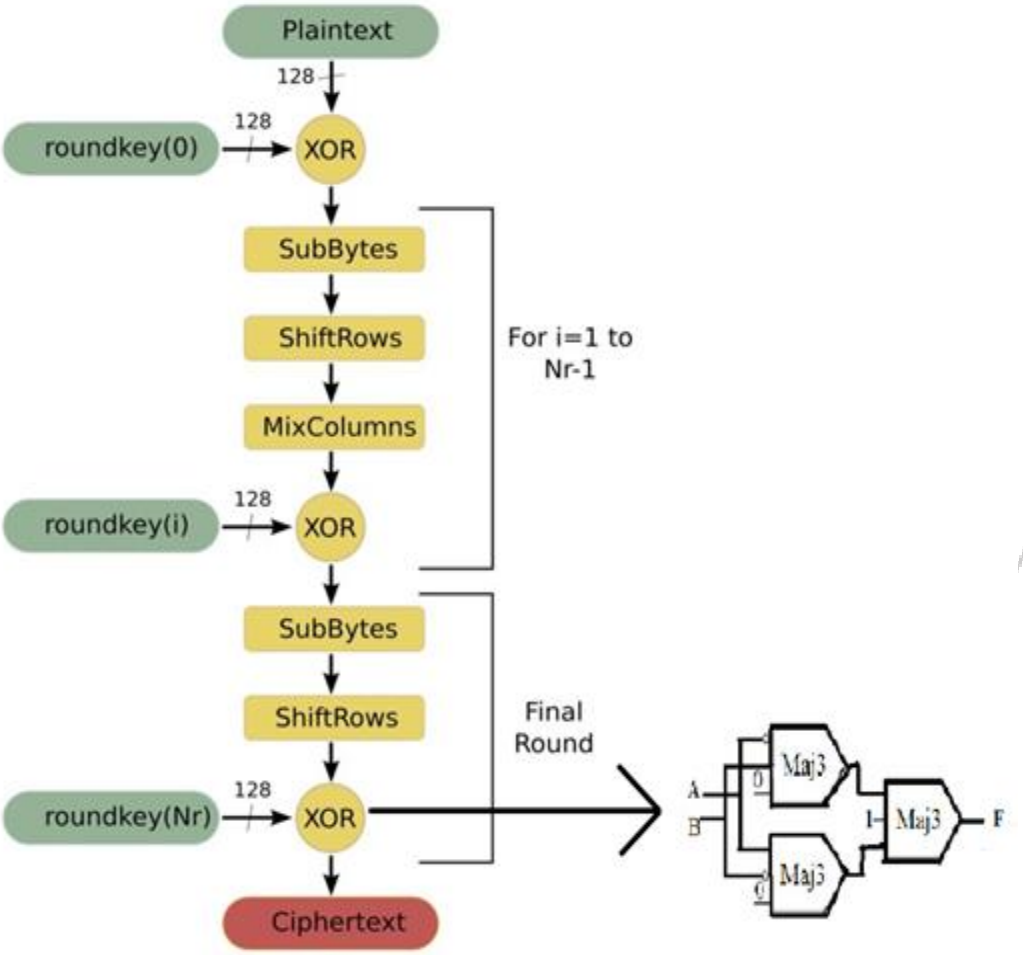


Fig 2: AES Algorithm using Majority logic gates for xor operation

4.1 MAJORITY LOGIC GATES

The fundamental logic gates inherently available within the QCA technology are the inverter and the majority logic gates. A majority gate returns true if and only if more than 50% of its inputs are true.
A majority gate is a logical gate that returns true if more than half of its inputs are true.

$$M(abc) = a \cdot b + a \cdot c + b \cdot c \text{-----(1)}$$

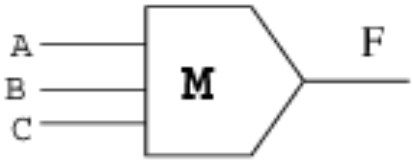


Fig 3: Three input majority logic gate

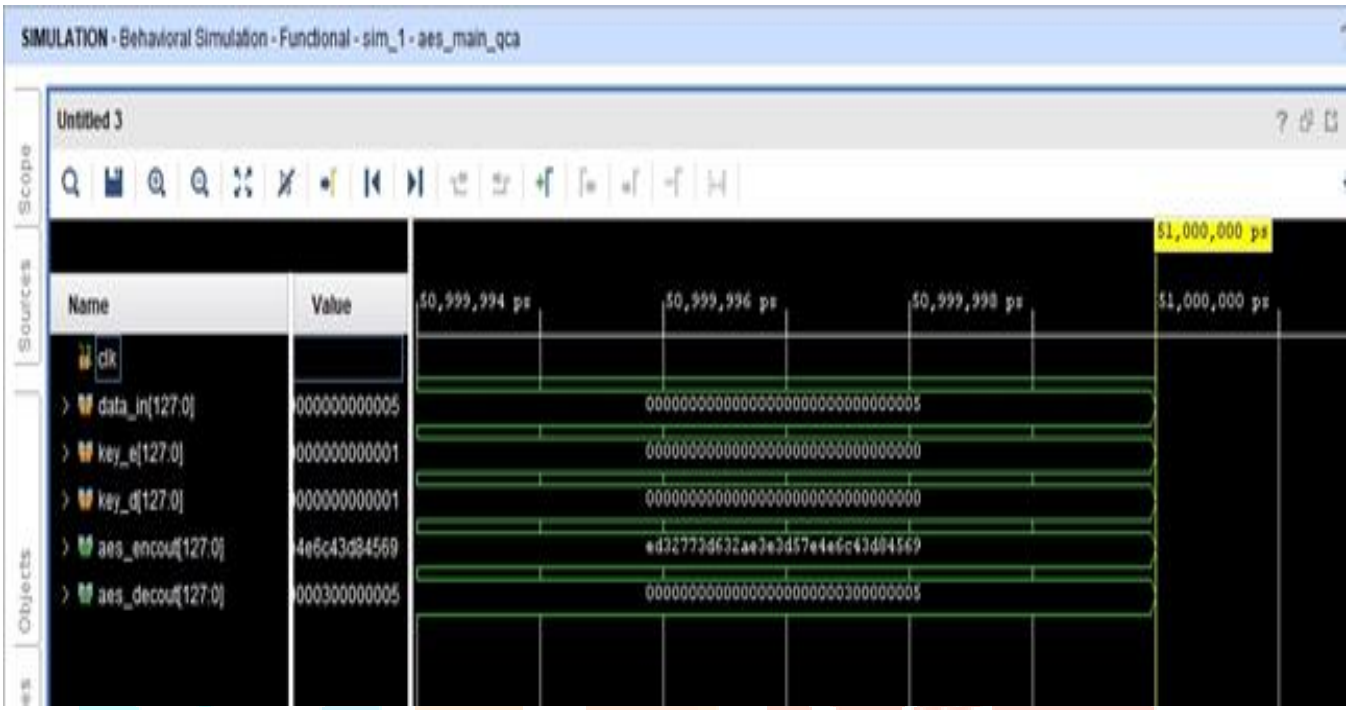
The truth table for the majority logic gate is given in below table. The truth table is for three input majority gate with inputs A,B,C and output F.

Table 1: Truth table of majority logic gate

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

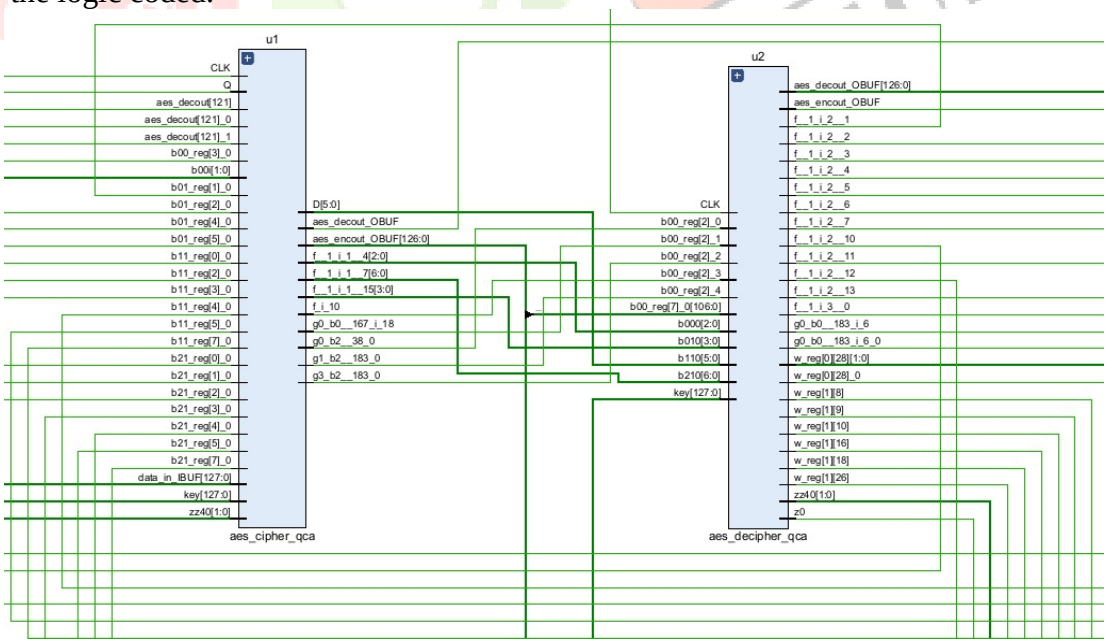
V SIMULATION RESULTS

The simulation results for the 128 bit AES using majority logic gates is shown below figure. The input data is given in hexadecimal and the key is given as 0 for both encrypted and decrypted key. Whatever the input is given the data should be encrypted and the same input data is received at the decrypted side.



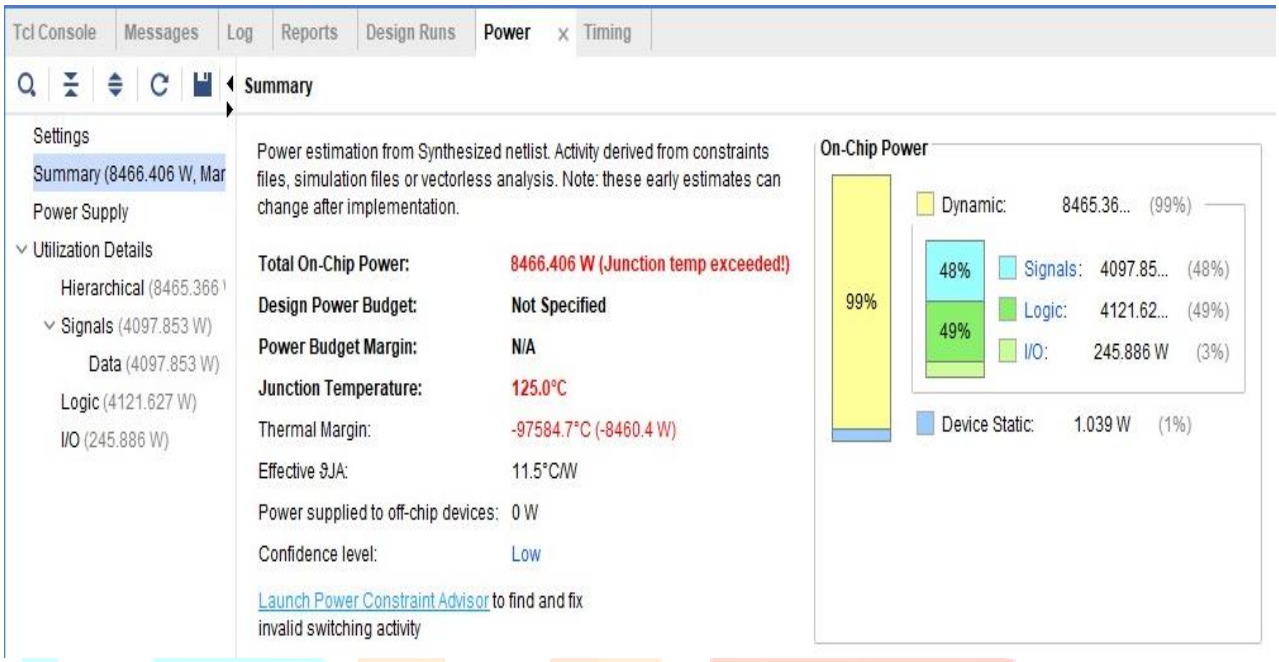
5.1 RTL SCHEMATIC OF AES USING MAJORITY LOGIC GATES

The RTL schematic is displayed in the form of logic gates. It gives the schematic representation in the form of gates for the logic coded.



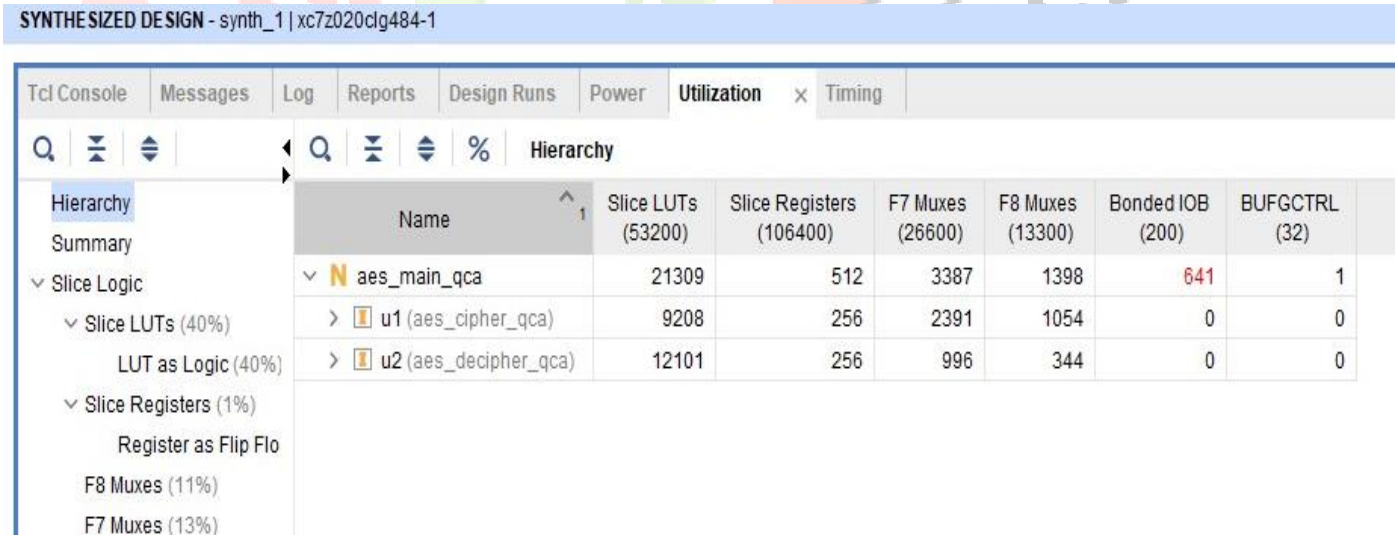
5.2 POWER REPORT OF AES USING MAJORITY LOGIC GATES

The power report is shown below. As the input bit representation is for 128 bit for both the input data as well as key it is consuming power of 8466.406 W . The total on chip power consumption is both Static power + Dynamic power.



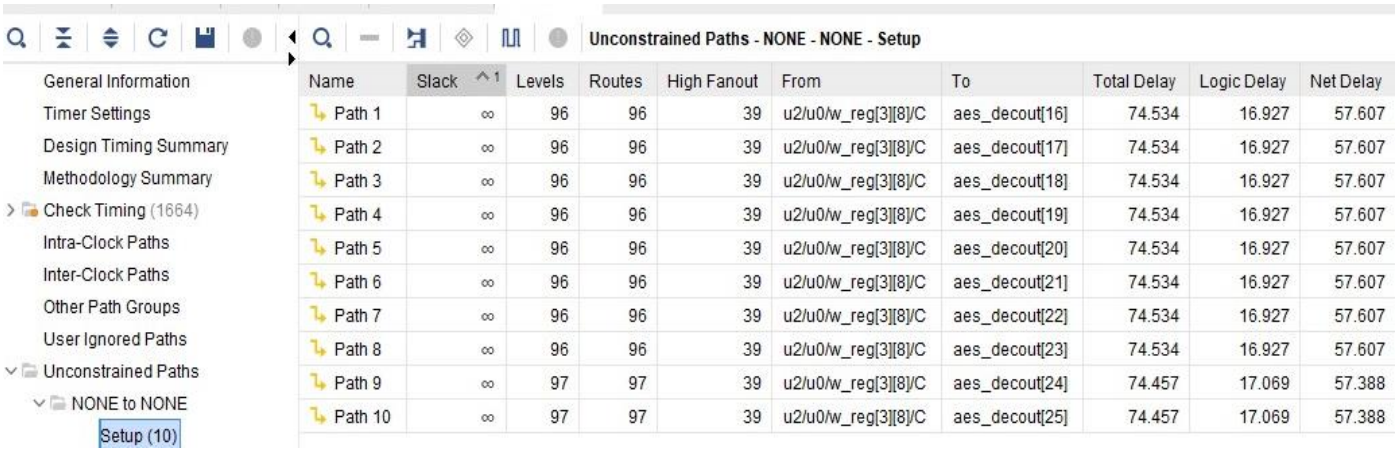
5.3 UTILISATION OR AREA REPORT OF AES USING MAJORITY LOGIC GATES

The utilization report is given for the zynq family in which the total LUTs are 53200 out of which AES using majority logic gates is utilizing 21309 and similarly the available registers are 106400 and used are 512.



5.4 TIMING REPORT OF AES USING MAJORITY LOGIC GATES

Propagation delay is the time it takes for an input to change and for the output to change in response. setup time is the minimum amount of time that data must be stable before the clock edge.

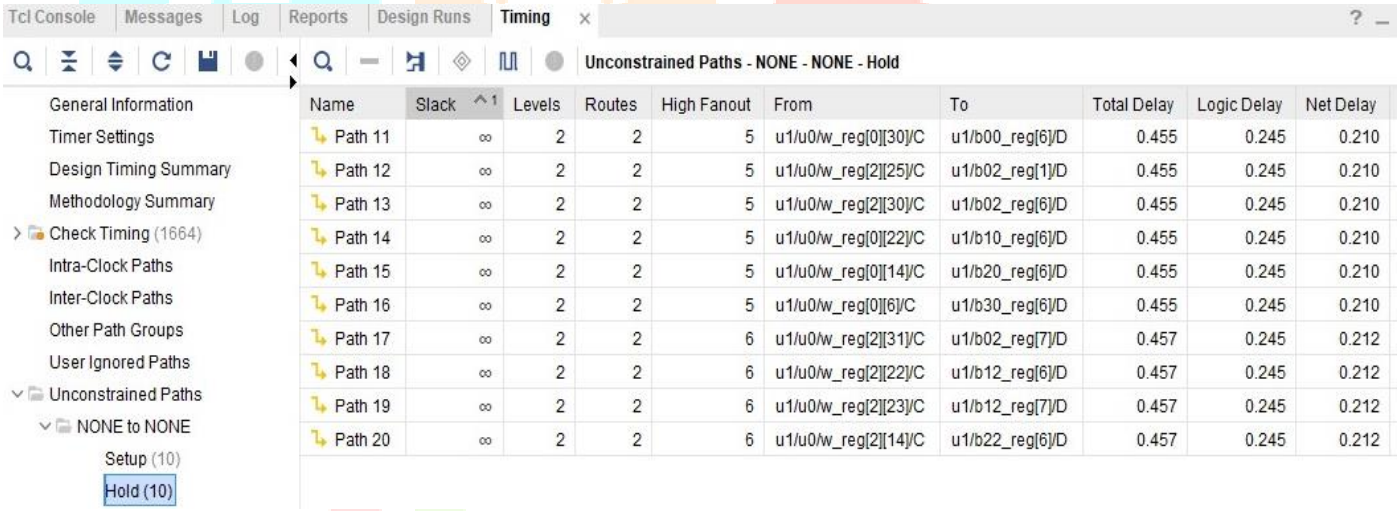


Unconstrained Paths - NONE - NONE - Setup

General Information
Timer Settings
Design Timing Summary
Methodology Summary
Check Timing (1664)
Intra-Clock Paths
Inter-Clock Paths
Other Path Groups
User Ignored Paths
Unconstrained Paths
NONE to NONE
Setup (10)

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay
Path 1	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[16]	74.534	16.927	57.607
Path 2	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[17]	74.534	16.927	57.607
Path 3	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[18]	74.534	16.927	57.607
Path 4	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[19]	74.534	16.927	57.607
Path 5	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[20]	74.534	16.927	57.607
Path 6	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[21]	74.534	16.927	57.607
Path 7	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[22]	74.534	16.927	57.607
Path 8	∞	96	96	39	u2/u0/w_reg[3][8]/C	aes_decout[23]	74.534	16.927	57.607
Path 9	∞	97	97	39	u2/u0/w_reg[3][8]/C	aes_decout[24]	74.457	17.069	57.388
Path 10	∞	97	97	39	u2/u0/w_reg[3][8]/C	aes_decout[25]	74.457	17.069	57.388

Hold time is the minimum amount of time that data input must remain stable after the clock edge.



Tcl Console | Messages | Log | Reports | Design Runs | Timing

Unconstrained Paths - NONE - NONE - Hold

General Information
Timer Settings
Design Timing Summary
Methodology Summary
Check Timing (1664)
Intra-Clock Paths
Inter-Clock Paths
Other Path Groups
User Ignored Paths
Unconstrained Paths
NONE to NONE
Setup (10)
Hold (10)

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay
Path 11	∞	2	2	5	u1/u0/w_reg[0][30]/C	u1/b00_reg[6]/D	0.455	0.245	0.210
Path 12	∞	2	2	5	u1/u0/w_reg[2][25]/C	u1/b02_reg[1]/D	0.455	0.245	0.210
Path 13	∞	2	2	5	u1/u0/w_reg[2][30]/C	u1/b02_reg[6]/D	0.455	0.245	0.210
Path 14	∞	2	2	5	u1/u0/w_reg[0][22]/C	u1/b10_reg[6]/D	0.455	0.245	0.210
Path 15	∞	2	2	5	u1/u0/w_reg[0][14]/C	u1/b20_reg[6]/D	0.455	0.245	0.210
Path 16	∞	2	2	5	u1/u0/w_reg[0][6]/C	u1/b30_reg[6]/D	0.455	0.245	0.210
Path 17	∞	2	2	6	u1/u0/w_reg[2][31]/C	u1/b02_reg[7]/D	0.457	0.245	0.212
Path 18	∞	2	2	6	u1/u0/w_reg[2][22]/C	u1/b12_reg[6]/D	0.457	0.245	0.212
Path 19	∞	2	2	6	u1/u0/w_reg[2][23]/C	u1/b12_reg[7]/D	0.457	0.245	0.212
Path 20	∞	2	2	6	u1/u0/w_reg[2][14]/C	u1/b22_reg[6]/D	0.457	0.245	0.212

VI COMPARISON TABLE FOR DIFFERENT FPGA' IN SIMULATION

The comparison is given for different FPGA's families .it shows number of LUT's and registers and input and output pins used for the 128 bit AES algorithm.

Table 2: Comparison for different FPGA families

PARAMETER	ZYNQ FPGA	VERTEX 7	ARTIX 7
Registers (Used/Available)	384/106400	384/40800	384/126800
LUT's	20755/53200	20758/204000	20737/63400
flipflops	20532/20916	20535/20919	20514/20898
IOB's	641	641	641
delay	57.775ns	57.401ns	74.925ns

CONCLUSION

In this an efficient AES is implemented using majority logic gates which consumes less number of registers and LUTs which shows that the area utilization is less when compared with the conventional AES algorithm. The design is implemented using Xilinx tools and reports are analyzed for different FPGA's.

[1] S. Perri and P. Corsonello, "New metho dology for the design of efficient binary addition in QCA," IEEE Trans.Nanotechnol., vol. 11, no. 6,pp. 1192–1200, Nov. 2012

[2]. Y. Wang, A. Kumar and Y. Ha, "FPGA-based High Throughput XTSAES Encryption/Decryption for Storage Area Network," International Conference on Field-Programmable Technology (FPT), IEEE, Shanghai, pp.268-271, 2014.

[3] FIPS 197, "Advanced Encryption Standard (AES)," November 26, 2001.

[4] M. James, D. S. Kumar. 2016, "An implementation of modified lightweight advance encryption standard in FPGA" Procedia Technology, Elsevier, vol. 25

[5] S. Bri and S. Oukili, "High speed efficient FPGA implementation of pipelined AES S-Box," High school of Technology, Moulay Islamil University Meknes, Morocco, 2016.

[6] J. Goodwin, P.R. Wilson, "Advanced encryption standard (AES) implementation with increased DPA resistance and low overhead," IEEE Transaction, pp. 3286–3289, 2008.

[7] M. J. Atallah, M. Blanton, N. Fazio, K. B. Frikken, "Dynamic and efficient key management for access hierarchies,"ACM Trans. Inf. Syst. Secur., vol. 12, no. 3, pp. 1–43, 2009.

[8] K. Kong, Y. Shang, and R. Lu, "An optimized majority logic synthesis methology for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 9, no. 2, pp. 170–183, Mar. 2010.

[9] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernestein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, pp. 49–57, 1993.

[10].S. Perri and P. Corsonello, "New methodology for the design of efficient binary addition in QCA," *IEEE Trans. Nanotechnol.*, vol. 11, no. 6, pp. 1192–1200, Nov. 2012.