VLSI ARCHITECTURE FOR SUBSTITUTION BOX

DISSERTATION - II

Submitted In partial fulfilment of the
Requirement for the Award of the
Degree of

MASTER OF TECHNOLOGY
IN
Electronics and Communication Engineering

By

KADAMATI SHRAVAN KUMAR

Under the Esteemed Guidance of

ABHISHEK KUMAR

Department of Electronics and communication Engineering
Lovely Professional University
Phagwara-144411, Punjab (India)
MAY 2017

DECLARATION

I hereby declare that the work presented in this Dissertation entitled "**VLSI Architecture for Substitution Box**" in partial fulfilment of the requirements of the award of degree of M.Tech (Electronics and Communication Engineering), submitted to Schoolof Electronics and Electrical Engineering, Lovely Professional University, Phagwara is an authentic record of my own work, carried out during the period since August 2016 to May 2017 under supervision of Abhishek Kumar. The matter presented in this Dissertation has not been submitted in any other University/Institute for the award of M.Tech Degree or any other Diploma/Degree.

Signature of the Student

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Signature of the Supervisor

Abhishek Kumar

The M. Tech Viva-Voce Examination of K Shravan Kumar has been held on _____and accepted.

Signature of H.O.D

Department of Electronics and Communication Engineering,

Lovely Professional University, Phagwara.

CERTIFICATE

This is to certify that the Dissertation entitled **"VLSI Architecture for Substitution Box"** is a record of bonafied work carried out by **K Shravan Kumar** under my guidance and supervision for the partial fulfilment of the degree of Master of Technology in Electronics and Communication Engineering during the academic session August 2016 – May 2017 at Lovely Professional University – Phagwara.

To the best of my knowledge, the results embodied in this Dissertation-II work have not been submitted to any university or institute for the award of any degree or diploma.

Dissertation Supervisor
Abhishek Kumar
Associate Professor
Department of ECE
Lovely Professional University
Phagwara

.

ACKNOWLEDGEMENT

I take this opportunity to acknowledge the co-operation, good will and both moral and technical support extended by several individuals out of which this Dissertation II has evolved. I always cherish my association with them.

I express my sincere and deepest regards to my supervisor **Abhishek Kumar** for his valued guidance during this period of my Dissertation. This Dissertation work was enabled and sustained by his vision and ideas. His scholarly guidance and invaluable suggestions motivated me to complete my Dissertation work successfully.

I owe a great many thanks to **Abhishek Kumar** for spending his valuable time. I considered my-self extremely fortunate to have this opportunity of associating with him.

I express my sincere thanks to the head of the department (ECE) and the members of Schoolof Electronics and Electrical Engineering, Lovely Professional University for their cooperation.

I would like to express my deep gratitude to my parents. Their continuous love and support gave me strength for pursuing my dream. Last but not the least; I am thankful to my friends who have been a source of encouragement and inspiration throughout the duration of this Dissertation.

ABSTRACT

Substitution-Box (S-BOX) is a primary computation block in the algorithm Advanced Encryption Standard (AES). Sub-bytes transformation is most complex steps in term of cost and implementation, it mapped each byte of a state is mapped to a different value. ROM or LUT based S-BOX contains a pre-computed value of each state, requires high amount of memory and suffers from unavoidable access time. Composite field arithmetic is more suitable for S-box implementation of high speed AES encryption. In this paper, we have proposed secured and low-power hardware architecture for the substitution Box. Proposed S-BOX architecture is implemented using Cadence Virtuoso at CMOS90nm technology. The hardware comparison results show that as S-BOX has critical path delay of 6.01 ns and power consumption of 5.776 nw**.**A worked example is being provided which can help for better understanding and functionality of the internal operations.

CONTENTS

Title	Page No.
Declaration	**i**
Certificate	**ii**
Acknowledgement	**iii**
Abstract	**iv**
Table of Contents	**v**
List of figures	**vi**
CHAPTER –1	**1-2**
INTRODUCTION	1
1.1 Reasons for implementing AES on Hardware	1
CHAPTER –2	**3-4**
REVIEW OF LITERATURE	3
2.1 Different high speed VLSI architecture for AES implementation	3
CHAPTER –3	**5-7**
BASIC PRINCIPLE OF CRYPTOGRAPHY	5
3.1 Types of Cryptography	6

CHAPTER –4 **8-11**

ADVANCED ENCRYPTION STANDARD 8

4.1 Evaluation of AES 10

4.2 Operation of AES 10

4.3 Encryption Process 11

CHAPTER –5 **12-22**

IMPLEMENTATION OF SUBSTITUTION BOX 12

5.1Composite Field of Arithmetic S-Box 15

5.2 Addition Operation in GF(2^4) 17

5.3 Squaring Operation in GF(2^4) 17

5.4 Multiplication with Constant λ 18

5.5 Galois Field GF(2^4) Multiplication 19

5.6 Multiplication with Constant φ 20

5.7 Galois field GF(2^2) Multiplication 20

5.8 Multiplicative Inversion in GF(2^4) 21

5.9 Worked Example 22

CHAPTER – 6 **23-40**

RESULTS AD DISCUSSION 23

6.1 Designing of S-box 23

6.2 Schematic for XOR and their simulated output 23

6.3 Schematic for Affine Transformation and their simulated output 25

6.4 Schematic for Squarer in GF(2^4) and their simulated output 27

6.5 Schematic for Multiplication with constant λ in GF(2^4) and their simulated output 29

6.6 Schematic for Multiplication operation in GF(2^4) and their simulated output 31

6.7 Schematic for Multiplicative inversion in GF(2^4) and their simulated output 33

6.8 Schematic for Multiplier operator of GF(2^2) 35

6.9 Schematic for Multiplication with constant φ 35

6.10 Schematic for Inverse Affine Transformation and their simulated output 36

6.11 Schematic for Substitution Box 38

CHAPTER – 7 **41**

CONCLUSION 41

7.1 FUTURE WORK 41

REFERENCES **42-43**

LIST OF FIGURES

Figure no.	Figure Names	Page no.
3.1.1	Secret Key Cryptography	6
3.1.2	Public Key Cryptography	6
3.1.3	Hash Function	7
4.2	Schematic of AES	11
4.3	Round Process	11
5.1	Sub-byte Transformation	12

5.1.1	Sub-bye and inverse sub-byte transformation in composite field	15
5.1.2	The Conventional S-Box architecture in Composite field	16
5.1.3	Meaning of Symbols used in figure 5.1.1	17
5.3	Logical Hardware Diagram of Squarer for GF(2^4)	18
5.4	Logical Hardware Diagram for Multiplication with Constant λ	19
5.5	Logical Hardware implementation for GF(2^4)	20
5.6	Logical Hardware implementation of multiplication with φ	20
5.7	Logical Hardware implementation of GF(2^2 Multiplication	21
5.8	Logical Hardware implementation of Squarer Multiplier Approach	22
5.8.1	Logical Hardware implementation of Multiple Decomposition Approach	22
5.9	A worked example for computing the sub-byte operation	22
6.2.1	Schematic of XOR Gates	23
6.2.2	Simulated Output for XOR	24
6.2.3	Power Calculation for XOR	24
6.2.4	Delay Calculationfor XOR	25
6.3.1	Schematic of Affine Transformation	25
6.3.2	Simulated Output for Affine Transformation (1)	26
6.3.3	Simulated Output for Affine Transformation (2)	26
6.3.4	Power Calculation for Affine Transformation	26
6.3.5	Delay Calculationfor Affine Transformation	27
6.4.1	Schematic of Squarer in GF(2^4)	27

6.4.2	Simulated Output for Squarer in GF(2^4)	28
6.4.3	Power Calculation for Squarer in GF(2^4)	28
6.4.4	Delay Calculationfor Squarer in GF(2^4)	29
6.5.1	Schematic of Multiplication with constant λ in GF(2^4)	29
6.5.2	Simulated Output for Multiplication with constant λ in GF(2^4)	30
6.5.3	Power Calculation for Multiplication with constant λ in GF(2^4)	30
6.5.4	Delay Calculation for Multiplication with constant λ in GF(2^4)	31
6.6.1	Schematic of Multiplication operation in GF(2^4)	31
6.6.2	Simulated Output for Multiplication operation in GF(2^4)	32
6.6.3	Power Calculation for Multiplication operation in GF(2^4)	32
6.6.4	Delay Calculationfor Multiplication operation in GF(2^4)	33
6.7.1	Schematic of Multiplicative Inversion in GF(2^4)	33
6.7.2	Simulated Output for Multiplicative Inversion in GF(2^4)	34
6.7.3	Power Calculation for Multiplicative Inversion in GF(2^4)	34
6.7.4	Delay Calculationfor Multiplicative Inversion in GF(2^4)	34
6.8.1	Schematic of Multiplier Operator of GF(2^2)	35
6.9.1	Schematic of Multiplication with constant φ	35
6.10.1	Schematic of Inverse Affine Transformation	36
6.10.2	Simulated output for Inverse Affine Transformation (1)	36
6.10.3	Simulated output for Inverse Affine Transformation (2)	37
6.10.4	Power Calculation for Inverse Affine Transformation	37

6.10.5	Delay Calculation for Inverse Affine Transformation	38
6.11.1	Schematic of Substitution Box	38
6.11.2	Simulated Output for Substitution Box (1)	39
6.11.3	Simulated Output for Substitution Box (2)	39
6.11.4	Power Calculation for Substitution Box	39
6.11.5	Delay Calculation for Substitution Box	40

LIST OF TABLES

Table no.	**Table Name**	**Page no.**
1.1	Characteristic Design for Algorithm Such as ASIC and FPGA	2
4.1	Comparison of different Cryptography Techniques	9

5.1.1	Sub-Bytes Transformation Table	13
5.1.2	Inverse Sub-Bytes Transformation Table	13
5.8.1	Pre-computed results of the multiplicative inverse operation in GF (2^4)	21
7.1	Results of Different Research papers of power, delay and Power delay product	41

CHAPTER-1

INTRODUCTION

In the modern world, cryptography is not only for the defence applications but it is also helping for the many other applications such as E-mails, E-commerce, Social networking etc. Cryptography is the main factor in predesigned interfaces such as embedded systems. As the number of applications and networking sites are huge now a day, it is becoming increase day by day. The transmitting and receiving data information become higher so storing and the information safely is becoming a difficult task. In all the applications, we need a secured connection which can't read by third party this can be done only through cryptography. Some of the Cryptographic algorithms are Data Encryption Standard algorithm (DES), Triple Data Encryption Standards (3DES) and Advanced Encryption Standard (AES) these are very helpful algorithms now and those are standardized by national Institute of Standards and Technology (NIST) [1].Still many of the researchers and Ethical hackers are constantly trying to attack these algorithms by using some new concepts called brute force, side channel etc. Some of the attacks were got succeeded with Data Encryption Standard (DES) in 1993, where the published cryptanalysis attack can crack the Data Encryption Standard (DES) easily at that particular situation. The Advanced Encryption Standard (AES) is considered as one of the best high secured data connection which is published in cryptographic algorithms; published by the National Institute for Standards and Technology (NIST). The Advanced Encryption Standard (AES) is became high secured data connection because it has a good secured margin it is not so simple to understand. AES is private key symmetric block cipher it helps in changing the key modules which can help to crack the algorithm. AES incorporates 128-bit data and the keys available for us are 128/192/256 bit keys. There is another option as Triple Data Encryption Standards (3DES) but it was not so effective because it is a bit slower than the Advanced Encryption Standard (AES) and very stronger than 3DES. The life time of the Advanced Encryption Standard (AES) is 20-30 years so it is very difficult to crack the algorithm. This can be implemented by so many applications, such as C, JAVA. It will provide us the full application and design specifications [2]. It is also helps in several applications such as ATM, phones and web browser. From the AES algorithm by the many applications here there is an S-box architecture has been implemented.

1.1 Reason for implementing AES on Hardware

The AES algorithm can implement in the both hardware as well software. When we want to select any Encryption speed and cost are two major concerns.Software implementations provide lowprotection and very slow to design. Due to the increasing in speedand to high level security hardware implementation of algorithms are designed. Algorithm means choosing the cipher text as much as secure as possible we make those as possible it will be secured. But now a day's lot of problems are there people are trying to attack all the applications oriented design issues and has lot of desired problem comes to us so if we use only software related program we

will have less secured data. Software implementing will also have some limitations sometimes the computer performance will be less, speed of ram and processing performance may be low it will always depend on the host computer. This impact will show the total performance of the system [3].

Application specific integrated circuits (ASIC) but it requires huge cost and time to market (TTM) is very high. The quality of hardwarerequires throughput, hardware, cost, speed. The newly implemented hardware should help in cost, speed and area which have to reduce the time to encrypt or decrypt the information and must give high speed, less cost and high throughput.

Every design has some pros and cons which are shown in table 1.1. Software implementation also has a major role here when we don't need a high level security and easy usage. Less in security still speed will be very high and protection will be very less not to help anything in securing purpose. That means the protection is very low and effort that much for very low security is useless so we are using hardware implementation to make it more secure.

Table 1.1 Characteristic design for algorithm such asApplication specific integrated circuits (ASIC) and Field Programmable Gate array (FPGA)

Parameter	ASIC	FPGA	Software (GPPs)
Speed	Very high	High	Moderately high
Cost of implementation	High cost	Moderate cost	Low cost
Implementation cycle	Mode time taken	Moderate time taken	Less time taken
Tools	High cost	Low cost	Low cost
Upgrades and maintenance	Moderate cost	Low cost	Low cost
Key security	High	Moderately high	Less

CHAPTER -2

REVIEW OF LITERATURE

The literature review is divided into two sections: Different hardware AES implementations in FPGA and Design Methodologies to achieve required Goals. The first section is studied in order to find how efficient AES algorithms have been implemented in FPGAs so far. The second section gives different design methodologies to achieve our desired goals. This section also suggests the hardware and languages required for the efficient implementation of AES algorithm

2.1 Different high speed VLSI architecture for AES Implementation

- In 2015, Hamdan.O.Alanazi, B.B.Zaidan, Hamid A.Jalab had published an article where he worked on 3DES, DES and AES with Nine factor where he compared all the 3 algorithms and sees the best one and if we use the nine factors will achieve an good efficiency, flexibility and security purpose it is very ideal it will works for good security purpose in this he said that this nine factor technique takes time to crypt a data and to verify the maximum possibility keying for 60 billion seconds, that are to be agreed and proceed to proves the Advanced Encryption Standard is better than Data Encryption Standard and 3Data Encryption Standard is the final conclusion given by this article.

- In 2015, Gurpreet singh and Supriya had published an article about the study of Encryption algorithms of RSA, DES, 3DES and AES for information security in this he compressed all the 4 techniques but in this article, he told what is RSA but it is not so security purpose it was not so effective and secrete and private information has easily noticed or seen significance in the study of law of ethics, law of most recent information systems. Encryption has provided the securing the recipient knowledge about the subject.

- In 2014, Challa vamsi Krishna, N.shiva kumar, Dr. D subba Rao had published an article on design implementation of composite fields S-Box suing AES 256 Algorithm in this article he has implemented the mix columns and inverse mix columns operation which is major operation in the AES method of cryptography. In the implementation of this AES-256 algorithm has a plaintext of 128 bits and key of 256 bits size. The number of rounds of operations in AES-256 bits algoritm is 14 rounds. The key generation process of AES 256 is different from other AES algorithms. In this he has implemented the Verilog code for the mix column and inverse mix column. [19]

- In 2012, Ali Akbar Pammu, Kwen-siong Chong and Bah-Hwee Gwee had published an article on Secured Low power overhead compensator Look-up-Table (LUT) Substitution Box (S-Box) Architecture there is a high-power dissipation always compromises with its

security feature under correlation power of low power overhead. S-Box architecture by using AND and OR gates to realise the multiplexing circuit to minimize the power variation for each input pattern.it variance in a multiplexing circuit and the power dissipation as the leakage information to secure against the CPA attack. The AND/OR gates and with a compensator, has the highest security than the reported designs. He said that the AES design for some secured ubiquitous electronics, including lot applications

- In 2011, Fakir Sharif Hossain, Md. Liakot Ali had published an article on a novel Bye-Substitution Architecture for AES Cryptosystem and simulated her a new AES substitution technique. He implemented and initiated in three different hardware combinations in 0.35 μm CMOS technology. The design in also simulated in Quartus-II simulator in FPGA platform to determine its power, delay and area. The proposed pipeline architecture if S-box shows that the throughout can be maximizes by reducing the delay of the critical path. It is very capable in the power design of significantly outperforming the existing solutions in terms of power, delay ad area as measured by simulation.

- In 2010, Edwin NC Mui has been published a paper and he implemented the practical implementation of Rijndael S-Box using combinational logic he implemented the typical ROM based in a look-up-table implementation which access time is fixed and unbreakable. In this paper, the construction procedure for implementing a 2-stage pipeline combinational logic based S-box is presented and illustrated in step-by-step manner. Now he presently implemented Typical ROM is based on look-up-table, the presented implementation is both capable of higher speeds since it can be pipelined and small in terms of area occupy. This compact and high speed architecture allows the S-Box to be used in both area limited and demanding throughput AES chips for various applications, ranging from small smart cards to high speed servers.

- In 2004, keshab K.parhi and Xinmiao Zhang had published an article about novel high speed architecture for the hardware implementation of Advanced Encryption Standard algorithm he implemented by using Sub Bytes and Inverse Sub Bytes transformations of AES algorithm, proposed design also implemented the combinational logics only he used in this entirely. He was used the look-up-tables and he used to the conventional Substitution Box. It makes 79% more efficient in terms of throughput which is speediest one which helps in working much efficient.

CHAPTER-3

BASIC PRINCIPLE OF CRYPTOGRAPHY

Cryptography is a method of securing the data can be done by the cryptography using encryption and decryption.

1. Encryption and Decryption

Encryption means converting some data in to unreadable form that can be called as Encryption. This is very helpful between a sender and a receiver like the data which is sending by the sender and the information coming to the receiver must be safe and secure for that we have to take the data from the sender and encrypt the data into some unreadable form and then we have to decrypt the same data and send to the receiver. The reverse form of encryption is called as Decryption. By using the Encryption and Decryption we need some additional information which must not be understandable by third party called “key”. For the Encryption and Decryption, we have different keys. [5]

2. Authentication

Authentication plays an important role in key based cryptography. In this if ABC send a message to the XYZ and XYZ wants to know that is that message really sends by ABC; this is possible if ABC performs some action that only XYZ knows only ABC can do it. Authentication means fuzzyfying information which can be known that who sent to whom. [5]

3. Integrity

Integrity is a big problem which we are facing now a day, loosing of messages between a sender and receiver that mean here cryptography as to make sure that the messages which are coming from the sender are correctly receiving by the receiver or not but here we have to ensure that the messages coming from receiver are not altered which as to take care by the cryptography hash. [5]

4. Non-Repudiation

Missing of messages, suppose XYZ has sent a message and the ABC didn’t get that message still XYZ send it to ABC here miss consumption takes place cases like these will be happen on the cryptography one of the best way to eradicate this through digital signatures. [5]

3.1 Types of Cryptography

There are three types of cryptography techniques:

- Secret key Cryptography
- Public key cryptography
- Hash Functions

1. Secret Key Cryptography

Secret key cryptography is a technique where sender sends a message as a key and the end user receiver tries to decrypt the message which is send by the sender. This is a single key usage here we use the same key to encrypt and same key to decrypt so this is called symmetric cryptography. [7]

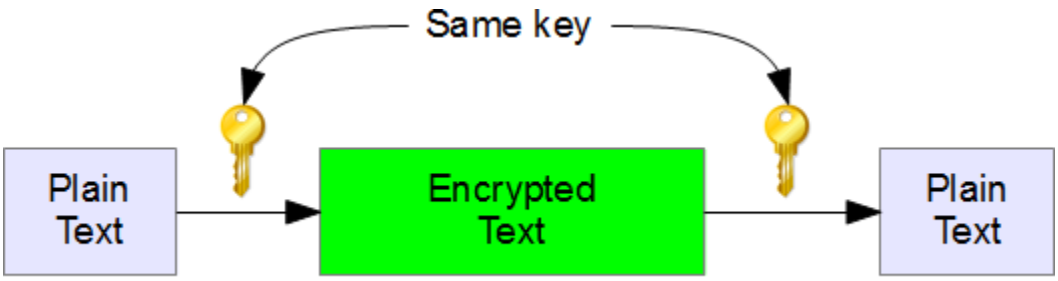


Fig 3.1.1 Secret key Cryptography

There is problem in this technique that is if we distribute the key which is send by the sender and receiver then there is a big challenge we are facing is the message can be understand by the one who knows the key. [7]

2. Public Key Cryptography

Public key cryptography is another technique that the key is public and here we use different keys still a sender sends a message in one key and the receiver receives the same message to decrypt with a different key so this type of technique is called asymmetric encryption

Fig 3.1.2 Public key Cryptography

In this we have two keys pubic key and private key. Here we use public key for whom we want to communicate the message with. If XYZ sends a message using public key, then the receiver receives the message and decrypt using private key. [8]

3. Hash Functions

Hash function is a different one which doesn't involve in public or private key cryptography. It only checks for the length that it tries to compute. It always checks for the integrity of messages. For this cryptography, as to ensure the message has been originated by the sender only without alter or without any miscomputing or any misalignment or any virus. [9]

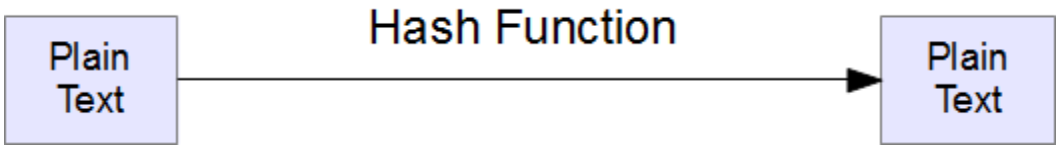


Fig 3.1.3 Hash Function

CHAPTER-4

ADVANCED ENCRYPTION STANDARD

National Institute for Standards and Technology (NIST) has been issued a crypt standard Data Encryption Standard in this it is very speed and doesn't have a chance to decrypt the sender message. This DES can only be used in legacy systems and also uses triple DES. There is a special attraction in triple DES those are it has 168-bit key and other one is difficult to decrypt. Triple DES is special in specifications which can't be shown or not visible. Triple DES is vulnerable and doesn't allow in the attack of brute-force attack on DES. Cryptanalysis attack is very good resistant for the triple DES but here is some security issue in triple DES.There is a drawback in the triple DES is the security issue. For this we have DES but it only works on hardware but very difficult to do in the software part [12]. The DES has been started or designed on mid-1970, DES is slow in round keying but triple DES will work for the three rounds so DES is slower than triple DES. There is a drawback in both DES and triple DES is both will only use 64-bit size. It will work efficiently and effectively for the security and efficiency. For this we need a large block size is required.

National Institute for Standards and Technology (NIST) issued in 1997 a new standard came that's named as Advanced Encryption standard (AES). AES offers high security as well as effective to use than the triple DES. National Institute for Standards and Technology (NIST) declares that we need some requirements as it must have block cipher with 128 bit and also supports the 128, 192 and 256 bits. National Institute for Standards and Technology (NIST) proposal started searching for best cipher but all proposals were accepted at starting stage and next stage only 15 algorithms are accepted and in the next stage only 5 are accepted and in the final stage totally Rijndael algorithm is accepted and finally it got selected because of the good efficiency and security it has and it also a symmetric block cipher. Ultimately the two researchers who have researched on the rijndael algorithm are from Belgium Dr. Joan Daemen and Dr. Vincent Rijmen. Finally, NIST announced AES algorithm is the best algorithm than the triple DES in symmetric cipher and also it has higher security than the triple DES and efficiency wise AES is far better than the triple DES. Ultimately AES secure place in cryptography field and the both researchers who researched on the AES got a successful achievement and the criteria is worth. [13]

4.1 Evaluation of AES

This evaluation will worth a lot and evaluate the candidates for the best in their part. NIST tries to have some evaluation process that it must have some concern topic and it is specifically bonded for something and application oriented how much those algorithms are worth and then NIST kept three categories of criteria are:

1. **Security**: This is basic requirement in cryptanalysis an algorithm must have this security much efficiently and effectively. AES has declared best in all algorithms because brute-force attack doesn't work on AES. Attacking on AES is very difficult task it. Crypt analysis is helping on the point of AES attacking became a most difficult task.

2. **Cost:**Application coming to AES must be cost effective. It must be practical enough and good to use in practical. If we have high computation, then the usability of particular algorithm will be very speed and effectiveness of such an algorithm will be very high if we make it for less cost it will be more efficient.

3. **Flexibility:**Flexibility for such an algorithm which given and high security and has must be user friendly and it must be simple to implement the hardware and software environment.

4. **Suitability:**It must be very suitable for the security reasons and suitable for many different algorithms which tries to crack and not be every efficient to understand and crack it can be 64 or 128 or 256-bit keying. It must not be much cost effective and must have wide range of suitability environment and hardware and software implementation must be very easy when compared to other algorithms and security.

Table 4.1 Comparison of different cryptography techniques

Parameter	**AES**	**3DES**	**DES**	**RC2**	**Blow Fish**	**RC6**
Key size(bits)	I28,192 or 256 bits	168 or 112 bits	56 bits	8-128 or 64 bits	32-448 bits	128,192 or 256 bits
Cipher type	Symmetric block cipher	Symmetric block cipher	Symmetric block cipher	Symmetric cipher	Symmetric cipher	Symmetric cipher
Data size(bits)	128,192 or 256 bits	64 bits	64 bits	64 bits	64 bits	128 bits
Developed	2000	1978	1977	1987	1993	1998
Security	Considered secure	Secure but slow process	Inadequate	Vulnerable	Vulnerable	Vulnerable
Possible keys	2^{128}, 2 $x^2+x+\lambda$ ^{192} or 2^{256}	2^{112} or 2^{168}	2^{56}	2^{64} or 2^{128}	2^{32} or 2^{448}	2^{128}, 2^{192} or 2^{256}
Rounds	10(128 bits) 12(192 bits) 14(256 bits)	48	16	16 of type mixing, 2 of mashing	16	20

4.2 Operation of AES

AES is the symmetric cryptography this algorithm which is used in the AES which helps for the security purpose. It was because a fast of fastest in the all algorithms. AES became a replacement for the triple DES and triple DES need key size is too high and for the AES the key size is low when compared to the triple DES. And estimated power is very less and it works efficiently and effectively. And triple DES tries to be against the attack of key search attack but still it became much slower than all algorithms this is the big draw back in the triple DES so NIST considered the AES is the best of all algorithms because the against or defense was nice against all the resources. And triple DES tries to be against the attack of key search attack but still it became much slower than all algorithms this is the big draw back in the triple DES so NIST considered the AES is the best of all algorithms because the against or defense was nice against all the resources.The importantfeatures of AES are

- Symmetric key
- Symmetric block cipher
- Stronger and faster than all the algorithms
- Software and hardware is but simple when compared to all the algorithms
- It provides 128-bit data encryption
- It can be implemented in C and JAVA
- It will provide full specifications and design details as open source

AES algorithm is an iterative algorithm. AES will take the input as 128-bit keying and forward through and content that could be done through and it will change or convert into the 16bytes plain text format. Then interchange then rearrange the all bits in different manner and shuffle by the way in rows and columns in a particular manner and passes the values. Then it will take all the particular rearranged combinations into the replaced way and that can be taken as a input combination for a particular substitution box.If we take a single box that mean 128-bit will be taken as 16-bytes then we choose that 16-bytes of memory into the 4 rounds of memory that mean 4 rows and 4 columns. These 16 bytes of memory will be classified in to 4 rounds totally that could be like 4 rows and 4 columns will be involve in this entire matrix. AES is far better in the many ways than the triple DES has the backdrop. AES will take 10 rounds for the 128-bit keying and triple DES take a lot more rounds for the 128-bit keying. Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key. AES is far better in the many ways than the triple DES has the backdrop. AES will take 10 rounds for the 128-bit keying and triple DES take a lot more rounds for the 128-bit keying. Unlike DES, the number of rounds in AES is variable and depends on the length of the key.AES is far better in the many ways than the triple DES has the backdrop. AES will take 10 rounds for the 128-bit keying and triple DES take a lot more rounds for the 128-bit keying. Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES is far better in the many

ways than the triple DES has the backdrop. AES will take 10 rounds for the 128-bit keying and triple DES take a lot more rounds for the 128-bit keying. Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES is far better in the many ways than the triple DES has the backdrop. AES will take 10 rounds for the 128-bit keying and triple DES take a lot more rounds for the 128-bit keying. Unlike DES, the number of rounds in AES is variable and depends on the length of the key.The schematic of AES structure is given in the following illustration.

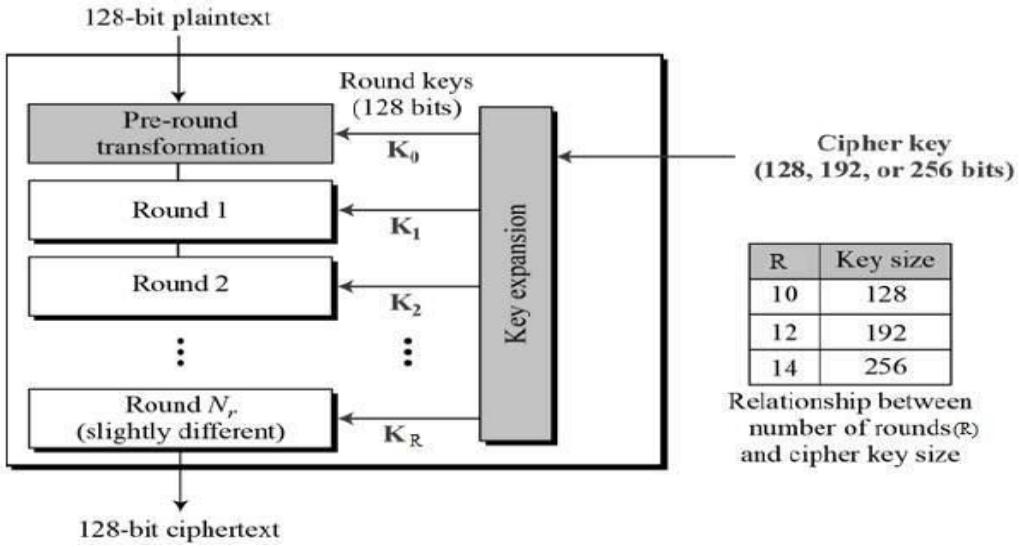


Fig 4.2 Schematic diagram of AES

4.3 Encryption Process

The Encryptions process is totally about first we have a plain text in the add round key and it will be added with the cipher key and send to the round process in round process it will be of 4 stages totally that mean 4 sub rounds it will be having first is sub bytes and then shift rows then mux columns then add round key it will be become a major issue in the entire block one wound which is shown below the rest of the rounds will be done by the same process for 4 times.

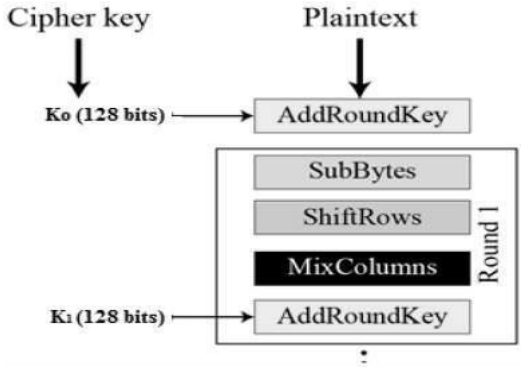


Fig 4.3 Round process

CHAPTER-5

IMPLEMENTATION OF SUBSTITUTION BOX

Substitution box is utilized for the encryption and for decryption it will uses the inverse substitution box. This substitution box is mainly used for the decoding purpose. Objective is to decode the particular matrix and utilize for the decoding the number of bits which given as an input. Substitution box is a non-linear byte which is free to merge and rearrange the bits and used for the Substitution box. Sub byte transformation is used for the converse of the GF factor it depends on and relatively changes and rearrange. Then interchange then rearrange the all bits in different manner and shuffle by the way in rows and columns in a particular manner and passes the values. Then it will take all the particular rearranged combinations into the replaced way and that can be taken as an input combination for a particular substitution box.

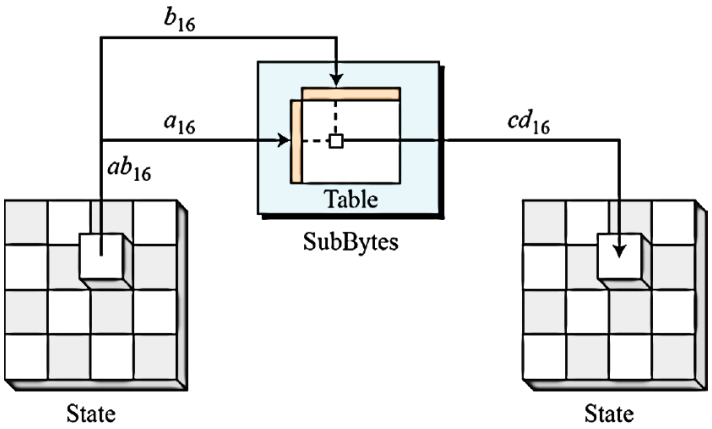


Figure 5.1 Sub Bytes Transformation

According to substitution box there are two types of components they are one it will first it will take the number of bits it uses to and then it will characterize the data which sends will make it correct and send back to it and second one is it will set the segments in the substitution box these two are combined get a new byte with 4 bits from one and four bits from other combined get an eight bytes

Inverse sub byte is a reverse process of sub byte. It is comparatively made by utilizing the data by mapping and then rearranging the number of bits it will be used to map. The sub-byte has some changes which can be done through the S-box only. [2]

There are of two types

(a) Using the Substitution box table
(b) Using the composite field

There are two tables available for the sub-bye and the inverse sub-bye transformation the table-5.1.1 indicates the sub byte table and the table one indicates the inverse sub byte table with is reverse of sub byte.

		b															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
a	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Table 5.1.1 Sub-Byte Transformation Table

		b															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
a	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Table 5.1.2 Inverse Sub-Byte Transformation Table

Sub-byte or Substitution box which can be called as the S-box and the invers sub-byte or inverse substitution box is called as the inverse S-box

Previously implemented S-box

In the previous implementation of S-box this Sub byte is a straight forward implementation which is in the S-box there is a ROM which is present in it that could store a lot of values that ROM will store the values according to the Look-up-table. In this all the 256 bits of values are stored in the ROM and then the input value is wired completely and that input which we used or tries to select then we will get that particular output which we have fixed in the ROM earlier. The input wire is fixed t the ROM address bus when we used to click the input which we need it will go to that particular address which we have stored earlier will be comes and send through the ROM as an output. But there is big problem in this ROM implementation that is the delay a lot of delay which is involved in this circuit because ROM as some fixed access capability which can only has a fixed time for every operation. For read and write operation which has some fixed some so delay become very high. In terms of the hardware implementation the implementing the ROM was very expensive. For the effective way, we can go for the combinational logic implementation. S-box implementation is a bit difficult task. Using the combinational logic, we can implement the same ROM with the less delay factor. Sub-byte having some advantage in the less area implementation and also it will occupy very less area. It has the increasing the performance on it and clock frequency get increased.

Construction of S-Box

This is the steps which are used to illustrate in the construction of the multiplicative inverse using a S-box module in a composite field arithmetic. In this a big block involved in it that is an affine transformation and inverse affine transformation both is used for the implementation in the S-box. sub byte and inverse sub byte are very similar but still it is making a lot of changes and their operation is involving these both sub byte and inverse sub byte so the implementation of sub byte and inverse sub byte are very important let us discuss about that in this paper. In this first we have to take the input in an 8-bit in a matrix form that mean first we are taking a 8-bit input which is used to make or construct the affine transformation and in the affine transformation. There is a block of matrix which is used to multiply with the input which is already given. Those are multiplied and given to the affine transformation which are used to the send in to two halves' that mean one is used to give to the xor for the operation. This is made to make it added and squarer GF(x^2) this was used to take the inputs in a random form that mean in a sequential way which it is already set to be. That will be gone to take place for the multiplied constant that will make a multiplier to be multiply with the lot many other variables and connected with the xor and passed. That could have been passed through the multiplicative inverse this multiplicative factor is quiet challenging to make it. This multiplicative inverse can be done in so many other ways total changes are made in this block named multiplicative

inverse. This block is totally made up of the two squarer blocks in series and other two are in series but these two are connected in parallel connection. There are connected to the squarer GF(x^2) and passed through the inverse affine to calculate and multiple with an inverse matrix and make it an 8-bit as an output. This is called as the sub byte output.[12]

Let's have some example that is let's take a simple binary number (x^2+x+1) *(x+1) = x^3+2x^2+2x+1 then at finally single values has to be taken that mean the final result is x^3+1 after that we have to reduce the value. And we can also calculate using a single binary number then 111*011=10101⊕111=10010 which can be normally return as the x^4+x. when the degree of result is more than m-1 that mean we have to take modulo for more x^2+x+1 is irreducible to the x^4+x which as to further multiplied with the x+1 then the value which comes is the final value. 10010⊕00011 is xor and we will get a result 10001. If the degree is again more than m-1 then now we have to multiply with the 10001⊕10010=00011 now we have to check the value is greater than m-1 or less than m. if the value is less than m that mean this is the final answer that is x+1 which is equal to 10001⊕10010=011. The bit string representation is x+1. [7]

The individual bit representation can be done by the values which are viewed in the GF(2^8) polynomial. It will be look like as the (x^2+x+1) *(x+1) = q^3+1 in GF(2^8). The irreducible polynomial that can be look like as Ax^2+x+1. Thus the elements in GF(2^8) may be represented as A will be the most significant value. The above equation finally says the multiply, addition, subtraction and multiplication and inversion in GF(2^4) that could be of Galois field. These all can be done by multiple transformed and sub divided and come to connect by all the things finally multiplicative inversion circuit GF(2^8). [15]

5.1 COMPOSITE FIELD ARITHEMATIC S-BOX

Regardless, it is not capable for applications requiring high throughput as ROM getting to incorporates one complete clock cycle for mapping one 8-bits state part and along these lines 16 clock cycles are required to change the 128 bits data.To fabricate the throughput, parallel ROMs are required achieving broad size of chip region. In this way, a more conceivable plan is to execute a S - box is by using composite field number which uses just method of reasoning segments as a piece of the utilization.The S-BOX substitution starts by finding the multiplicative in reverse of the data in GF (2^8), and after that applying the relative change. wanders for the one byte forward and inverse Sub-Byte change using composite field Algorithm. [12]

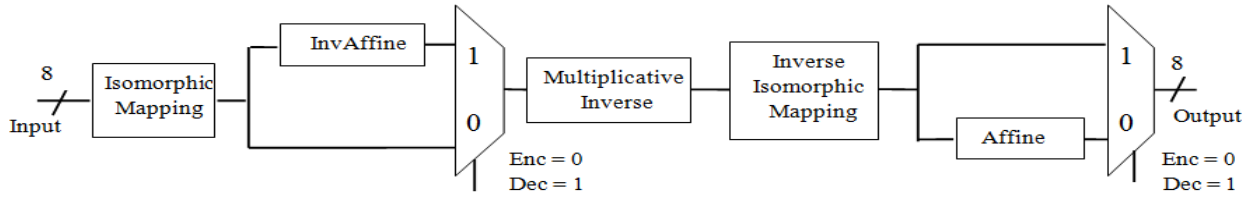


Figure 5.1 Sub-Byte and Inverse Sub-Byte transformation in composite field

To find the S-BOX change first multiplicative opposite of GF (2^8) then relative change figured. Moreover, for Inverse Sub-ByteFirst Inverse Affine change then multiplicative turn around must be discovered.There is one important operation incorporate here, which is to find the multiplicative inverse in GF (2^8). This ought to be conceivable by breaking the GF (2^8) segments in GF (2^4) and some more reliable squares.Any subjective polynomial in GF (2^8) can be addressed as bx+c using an irreducible polynomial x^2+Ax+B. Here, b is the most basic nibble and c is the smallest gigantic nibble. The multiplicative reverse can be done by using

$$(bx+c)^{-1} = b(b^2B+bcA+c^2)^{-1}x+(c+bA)(b^2B+bcA+c^2)^{-1}$$

$$= b(b^2\lambda+c(b+c))^{-1}x+(c+b)(b^2\lambda+c(b+c))$$

Where, A=1, B=λ, as the irreducible polynomial used is x2+x+λ. Figure 2.9 exhibits the square outline to find the multiplicative banter in GF (2^8) using GF (2^4). Figure 5.1.1 shows the suggestions the pictures used as a piece of Figure 5.1.2.The mapping structure in different fields nearby the irreducible polynomials is according to the following

$$\begin{aligned} &\text{GF}(2^2) \rightarrow \text{GF}(2) && : x^2+x+1 \\ &\text{GF}((2^2)^2) \rightarrow \text{GF}(2^2) && : x^2+x+\varphi \\ &\text{GF}(((2^2)^2)^2) \rightarrow \text{GF}((2^2)^2) && : x^2+x+\lambda \end{aligned}$$

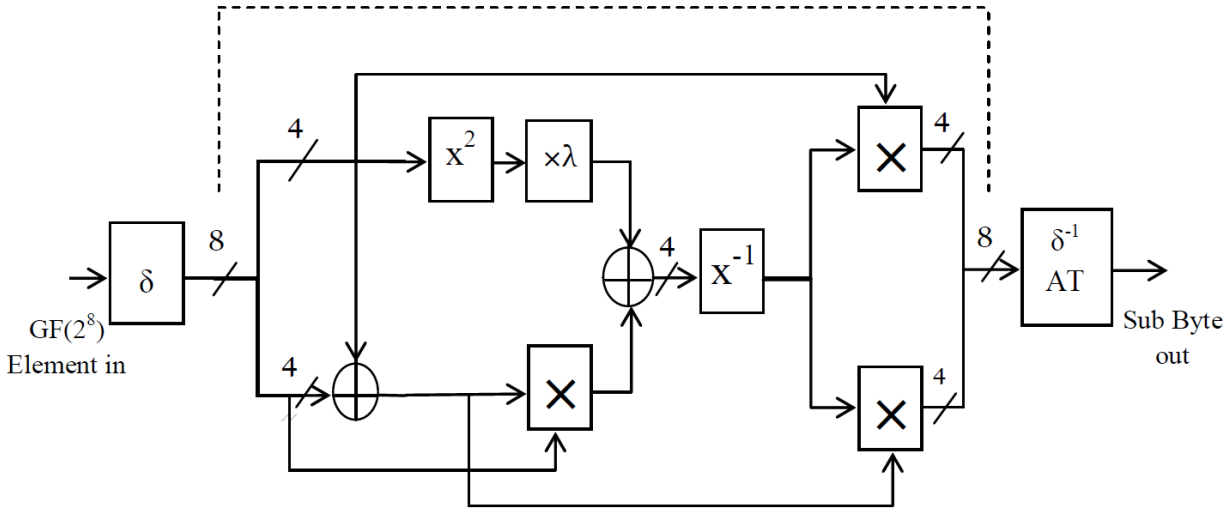


Figure 5.1.1 The conventional S-box architecture in composite field

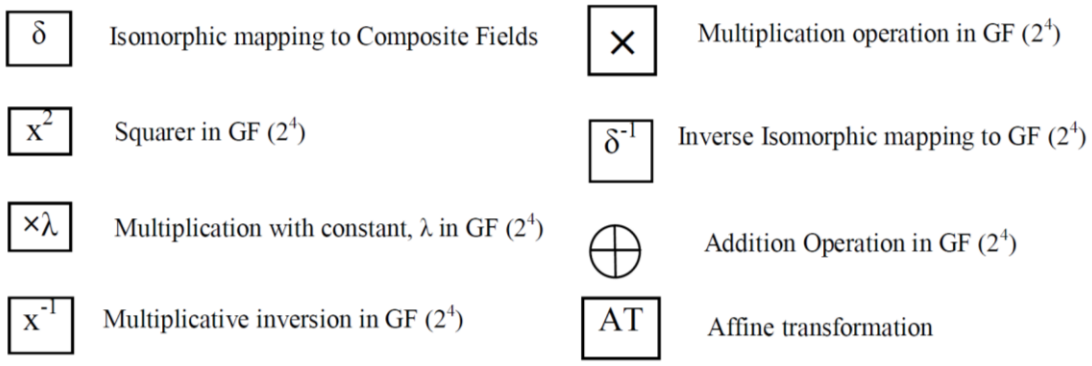


Figure 5.1.2 Meaning of symbols used in Figure 5.1.1

Isomorphic mapping is the underlying stride performed on the 8 bits sub byte input. The yield of the isomorphic mapping is given to the commitment of multiplicative switch (MI) module. Thusly, chat isomorphic mapping and relative changes are the implies that take after. [17]

5.2 Addition operation in GF (2^4)

The extension operation in Galois Field can be meant clear bitwise XOR operation between the two parts.

5.3 Squaring operation in GF (2^4)

The squaring operation of 4 bits, i.e. x^2 term can be modulo diminished using the irreducible polynomial, $x^2 + x + \varphi$. It can lessen into lower demand of Galois field, by setting $x^2 = x + \varphi$ and supplanting it into x^2. Doing above operation GF (2^4) is changed over into GF (2^2), here nibble is changed over into 2-bit stream. [16]

$$K = q_h{}^2 x + q_l{}^2$$

$$K = q_h{}^2 (x + \varphi) + q_l{}^2$$

Where k {k3k2k1k0} is the four bits yield of squarer and q $\{q_3q_2q_1q_0\}$ is the data bit steam, here qh, kh, ql, and kl are higher 2 bits of q and k and lower 2 bits of q and k independently. By and by GF (2^2) can be changed over into GF (2), the x^2 term can be supplanted $x^2 = x + 1$. For the occasion of x^3, it can be gained by expanding x^2 by x, i.e. $x^3 = x(x) + x = x^2 + x$. resulting to Substituting for x^2, $x^3 = x + 1 + x$. The two x terms are shown which balance each other, leaving just $x^3 = 1$. Playing out all the substitution yield bit stream can be learned by data bit streams in GF (2). [16]

$$K = q_3{}^2 x^2 + q_3 q_2 x^2 + q_3 q_2 x + q_2{}^2$$
$$K_1 = q_3 x^2 + q_2 x + q_1 x^2 + q_0$$

Here we understand that relative term in XORing operation will get wiped out, after that polynomial substitution needs to do which is discussed some time recently. It gives the sensible expression for all the yield bits. The condition can be recognized acknowledgement of reasoning chart using a XOR operation.

$$K_3 = q_3$$
$$K_2 = q_3 \oplus q_2$$
$$K_1 = q_2 \oplus q_1$$
$$K_0 = q_3 \oplus q_1 \oplus q_0$$

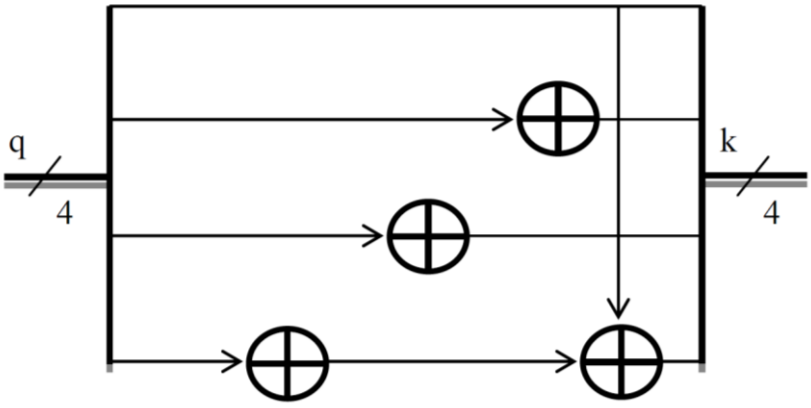


Figure 5.3 Logical hardware diagram of squarer for GF (2^4)

5.4 Multiplication with constant, λ

The expansion with steady λ, which regard is {1100} in GF (2^4) will give the polynomials. Modulo lessening can be performed by substituting $x^2 = x + \varphi$ using the irreducible polynomial to yield the intelligible expression.The last yield bits k as data bits q can be figured using irreducible polynomial. There are signifying three XOR entryway is required to execute the expansion with λ [11]. There are two XOR doors in essential way which will give the most extraordinary postponement

$$K_3 = q_2 \oplus q_0$$
$$K_2 = q_3 \oplus q_2 \oplus q_1 \oplus q_0$$
$$K_1 = q_3$$

$$K_0 = q_2$$

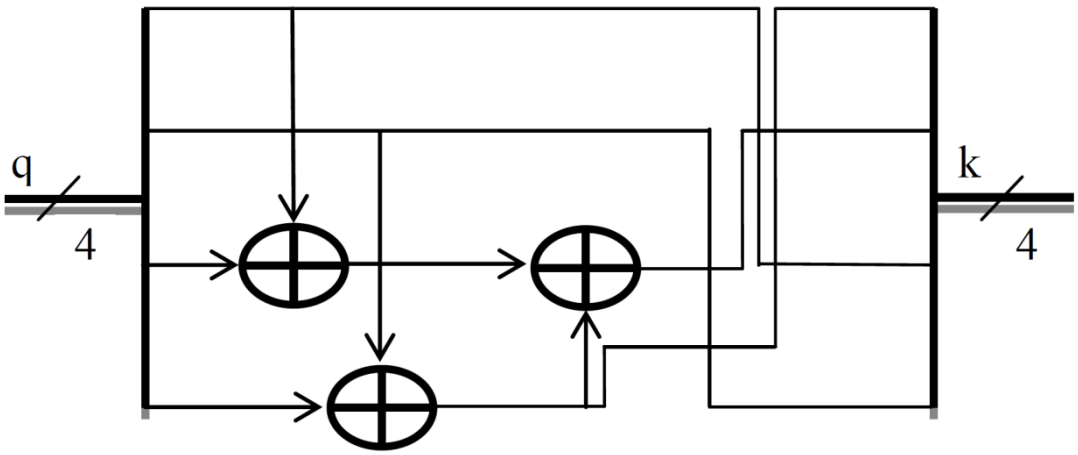


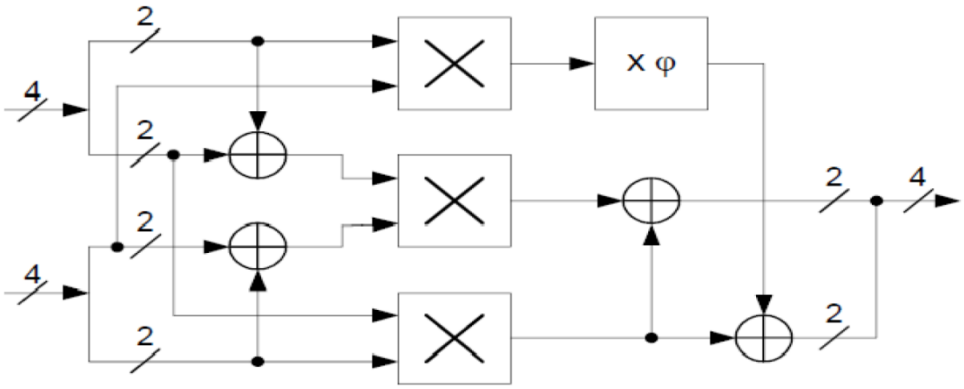
Figure 5.4 Logical hardware implementation for multiplication with constant, λ

5.5 Galois field GF (2^4) multiplication

The GF (2^4) multiplier is a crucial part to locate the multiplicative reverse utilizing composite field number juggling operation. It requires more equipment to execute in combinational reason. It is consequences of 4 bits with 4 bits and results in like way in 4 bits. Let k = qw, where k in the 4 bits joined yield and 'q' and 'w' are 4 bits inputs. It can be watched that expansion and improvement operation in GF (2^2), augment in GF (2^2) is a basic area in it. It can be changed over into a lower sort of Galois field utilizing irreducible polynomial present. Figure 5.5 demonstrates the good and fashioned apparatus execution of GF (2^4) augmentation. Here „+" addresses the XOR operation. [16]

$$K = k_h x + k_1$$

$$K = (q_h w_h + q_h w_l + q_l w_h)x + q_h w_h \varphi + q_l w_l$$



GF (2^2) multiplier.

Figure 5.5 Logical hardware implementation of GF (2^4) multiplier

5.6 Multiplication with constant, φ

The extension with unsurprising φ, which has a continuing respect φ = $\{10\}^2$ is a piece of GF (2^2). It has helpful futile respect that can be moreover tended to as combinational technique for thinking. Figure 5.6 displays the equipment execution of combinational strategy for thinking. Here k is a two bit and also q is also two bit [12].

$$K_1 = q_1 \oplus q_0$$

$$K_0 = q_1$$

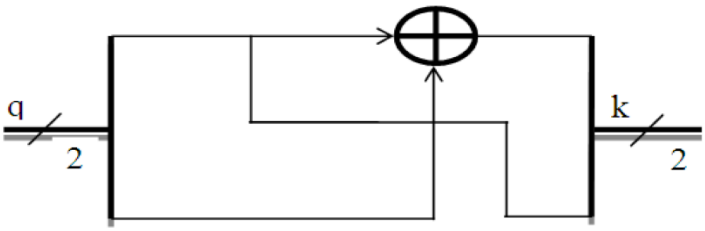


Figure 5.6 Hardware implementation of multiplication with φ

5.7 Galois field GF (2^2) multiplication

The Galois field (2^2) multiplier is the critical section in GF (2^4) increment, which exist in the essential way. It can be addressed in the data bit streams by using irreducible polynomial. It moreover realized using combinational reason. Figure 5.7 shows its gear execution in composite field math. Here k is two bits yield and ‘q’ and ‘w’ are the two-bit commitment of the fragment [15].

$$K_1 = q_1 w_1 \oplus q_0 w_1 \oplus q_1 w_0$$

$$K_0 = q_1 w_1 \oplus q_0 w_1$$

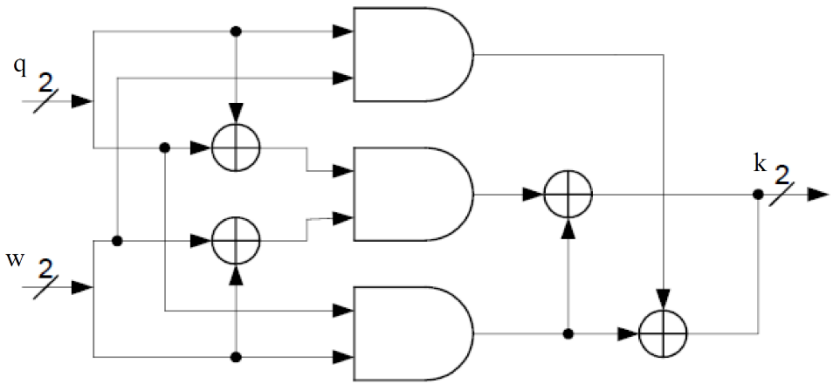


Figure 5.7Logical Hardware implementation of GF (2^2) multiplication

5.8 Multiplicative Inversion in GF (2^4)

The multiplicationinversion of a q and where q is a component of the GF (2^4) is a middle of the road segment of multiplicative opposite. It has determined a recipe to figure the multiplicative reverse of q, with the end goal is of *+. The reverse of every individual bit will be processed by the intelligent condition and also a pre-figured esteem which can put away in RAM. The pre-computed esteem is found in table 5.1 that will have predicted to locate the multiplication converse [5].

Table 5.8.1: Pre-computed results of the multiplicative inverse operation in GF (2^4)

Q	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**a**	**B**	**C**	**D**	**e**	**f**
q^{-1}	**0**	**1**	**3**	**2**	**f**	**c**	**9**	**b**	**A**	**6**	**8**	**7**	**5**	**E**	**d**	**4**

The table which is containing the consequences ofmultiplication ineverse in the hexa-decimal which is appeared previously.

The condition thatis beneath served to equipment execution in the combined rationale where

$$q_3^{-1} = q_3 + q_3q_2q_1 + q_3q_0 + q_2$$

$$q_2^{-1} = q_3q_2q_1 + q_3q_2q_0 + q_3q_0 + q_2 + q_2q_1$$

$$q_1^{-1} = q_3 + q_3q_2q_1 + q_3q_1q_0 + q_2q_0 + q_2 + q_1$$

$$q_0^{-1} = q_3q_2q_1 + q_3q_2q_0 + q_3q_1 + q_3q_1q_0 + q_3q_0 + q_2 + q_2q_1 + q_2q_1q_0 + q_1 + q_0$$

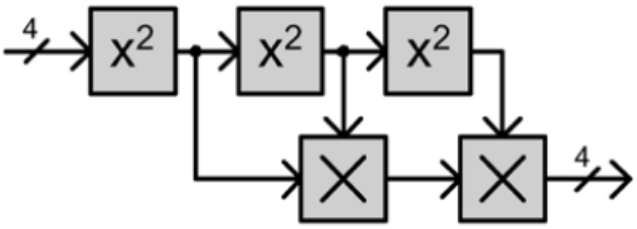


Figure 5.8Logical hardware implementation forSquarer Multiplier approach

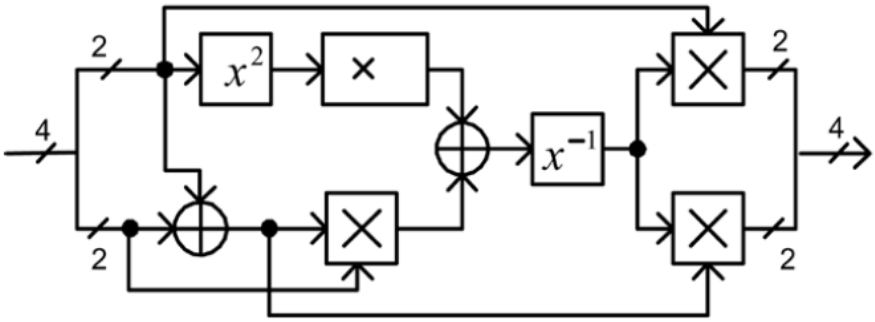


Figure5.8.1Logical hardware implementation formultiple decomposition approach

5.9 Worked Example

Figure 4.5 underneath represents a worked illustration utilizing the duplication table, multiplication inverse table in a piece chart appeared in Figure 4.5

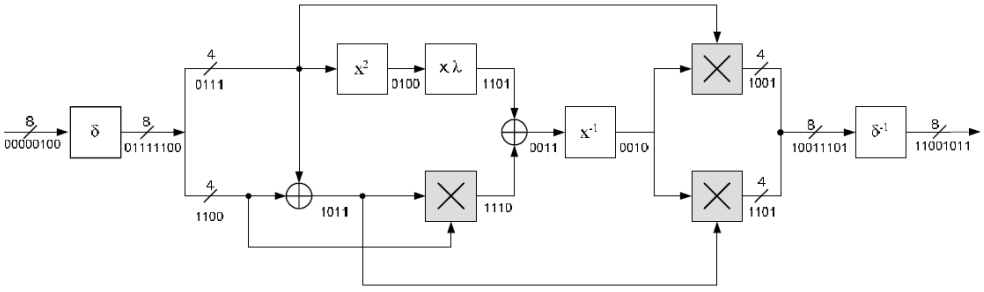


Figure 5.9 A worked example for computing the Sub-Byte operation

The above illustration demonstrates the esteem which is given to relative change and sent to every piece it will changes the qualities and it will move and duplicate and changes every one of the qualities and goes to the converse augmentation and experiences every one of the squares of 24 squares and combined passed and goes to the relative change and get the yield.

CHAPTER-6

RESULTS AND DISCUSSION

S-Box i.e.., substitution box shapes the middle like builder square of the gear use of anAdvanced Encryption Standard calculation. In this area addresses a Complementary MOS plan of the Substitution Box whichas low power and quick GF (2^8) Galois Field augmentation will be established on it. FPGA execution was not proper for some of the applications which are fundamentally due to size, district, postponement and power confinements. It is difficult to finish significantly arranged execution using FPGA utilization. The proposed engineering shows that XOR is the noteworthy portion which is used to do the expansion operation in Galois Field composite field math. The streamlining of the arrangement has been done by proposing novel circuit for litter parts like XOR entryway and other circuit fragments like Galois Field (GF) multiplier. The XOR has been made using slightest number out of transistors and it has high commotion edge and low power use when appeared differently in relation to existing XOR diagrams. The full circuit configuration is required for little contraptions like splendid cards, institutionalized distinguishing pieces of proof, net dealing with a record, Messaging, and high rate of data transmission [5].

6.1 DESIGNING OF SUBSTITUTION BOX

It is unmistakably understood that the execution of S-Box requires incalculable operations whose successful and low power use can realize improved CMOS S-Box equipment gear layout. The different 2-input XOR door diagrams have been depicted to enhance the execution for a few number of utilizations. A XOR entryway uses six transistors including an inverter, is mimicked in Cadence utilizing 180nm development innovation [9].

6.2 SCHEMATIC FOR XOR AND THEIR SIMULATED OUTPUT

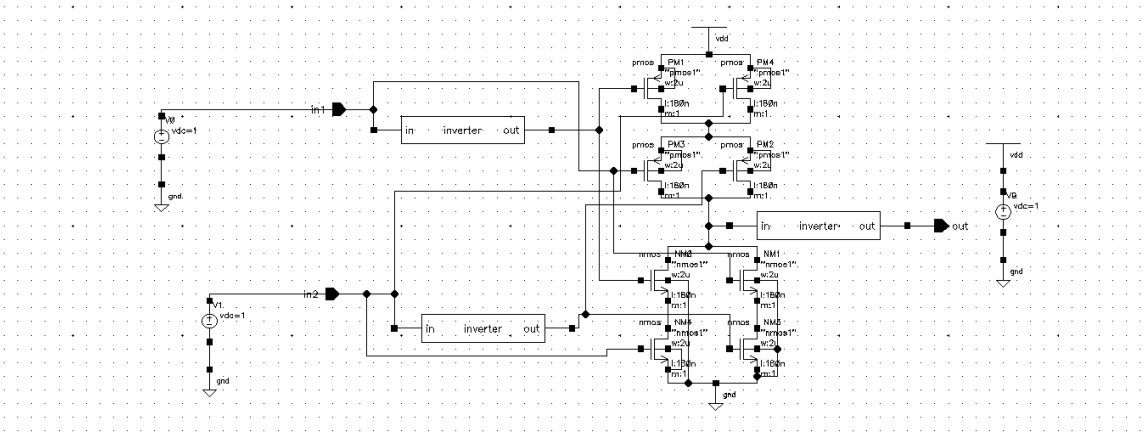


Fig 6.2.1 schematic of XOR gate

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results forXOR. Designed block in cadence is simulated and simulated waveform is described here.

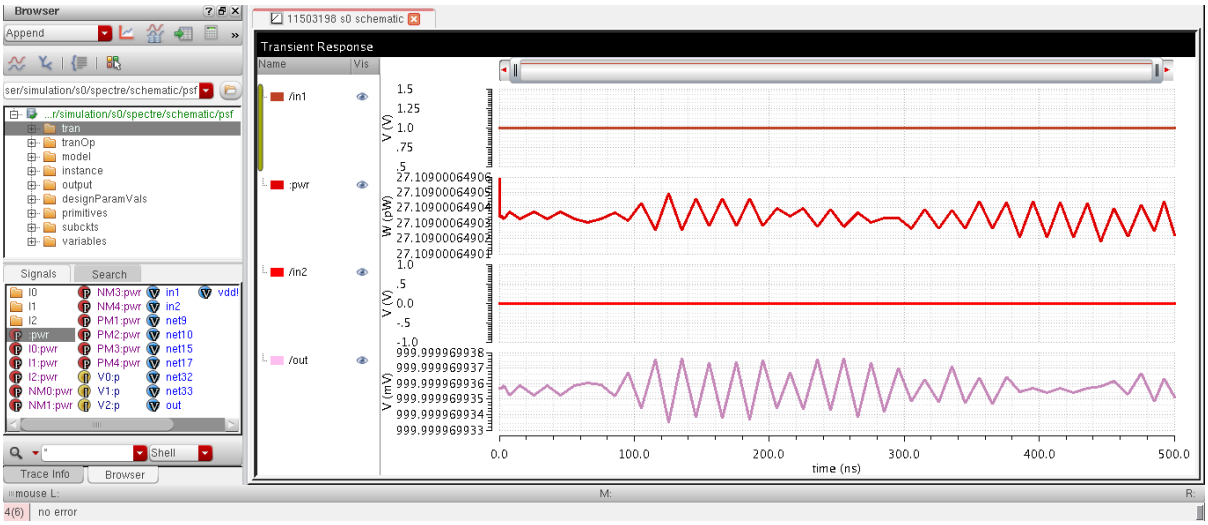


Fig 6.2.2 simulated output for XOR

Figure 6.2.3 shows power consumption and Average power consumed is 27.11 pw.

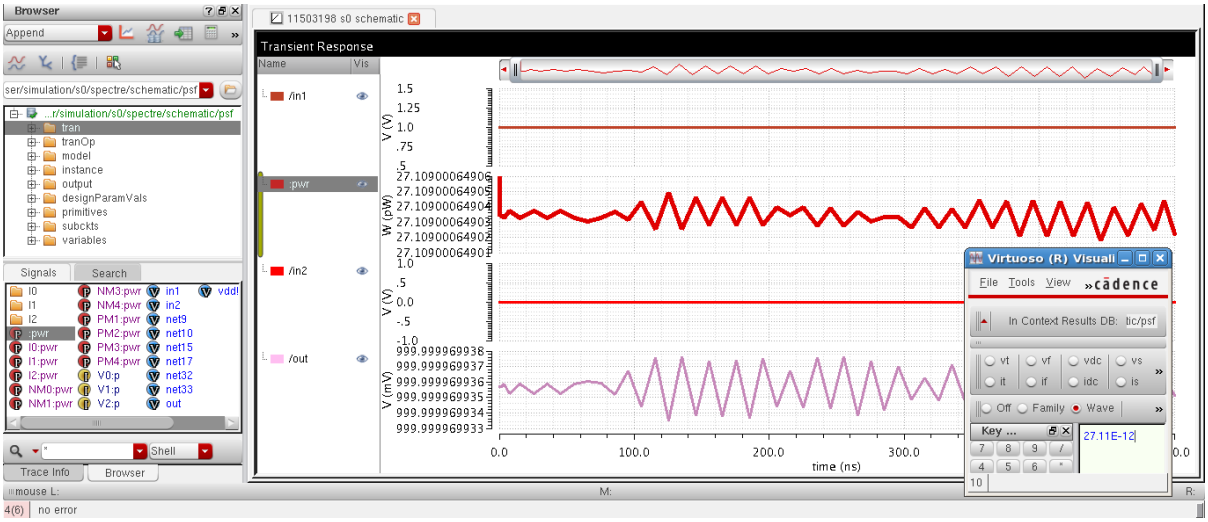


Fig 6.2.3 Power calculation for XOR

Figure 6.2.4 shows delay and total delay is -185.3 ps.

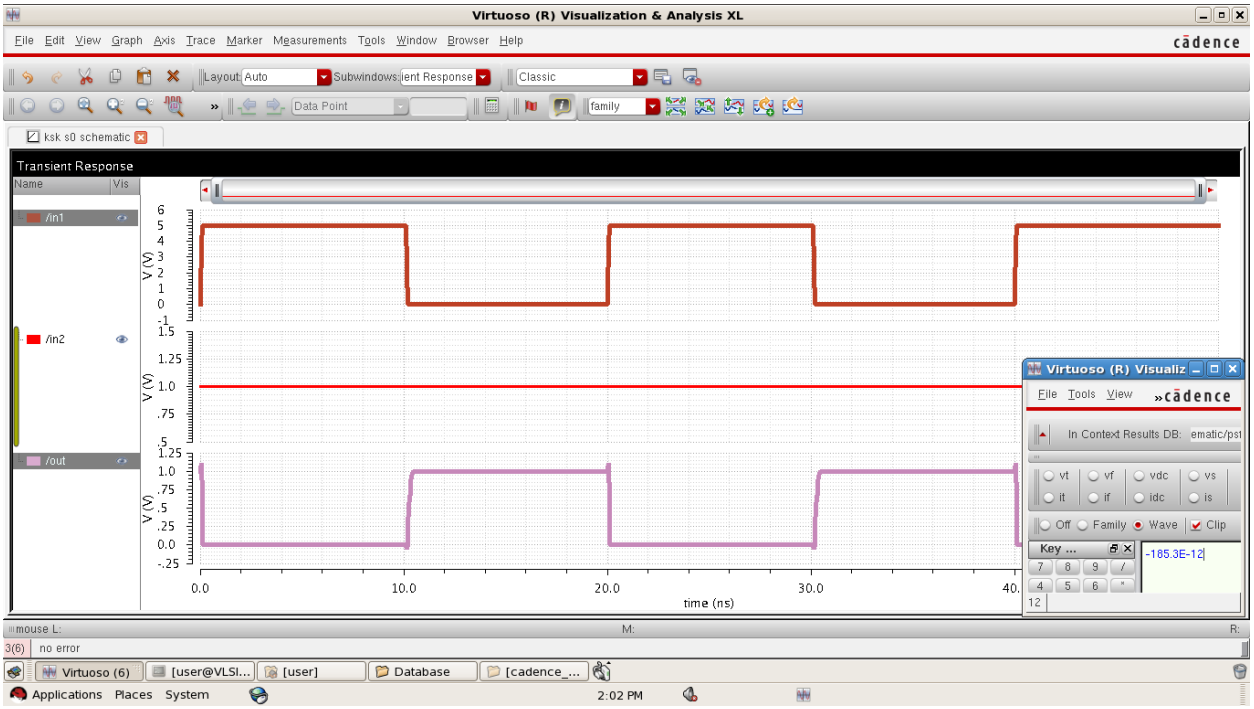


Fig 6.2.4 delay calculation for XOR

6.3SCHEMATIC FOR AFFINE TRANSFORMATION AND THEIR SIMULATED OUTPUT

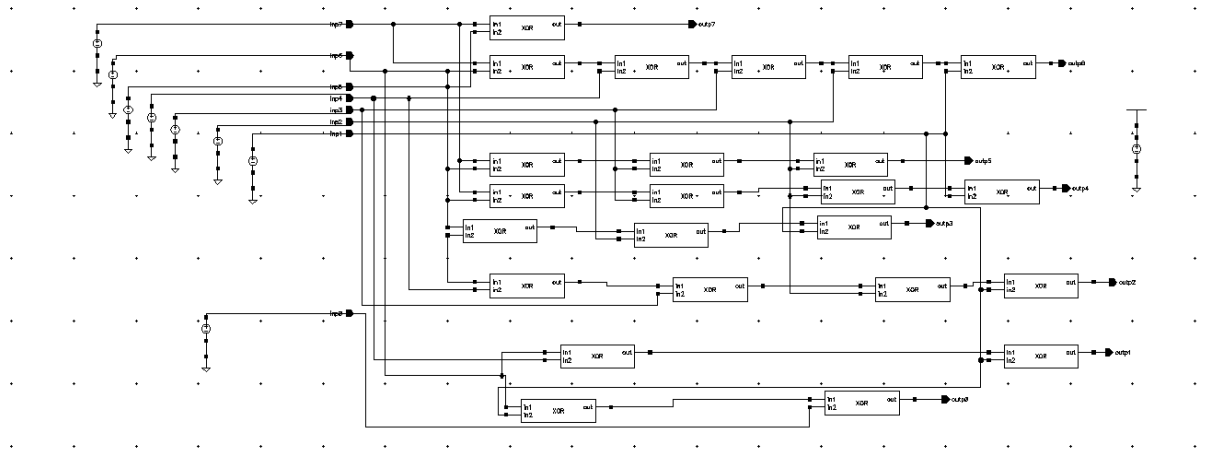


Fig 6.3.1 schematic for Affine Transformation

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results forAffine Transformation. Designed block in cadence is simulated and simulated waveform is described here.

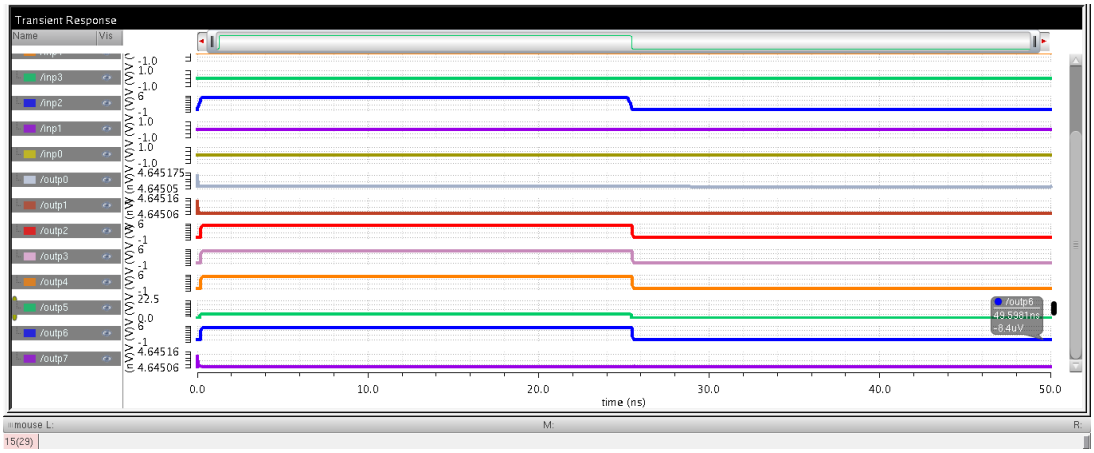


Fig 6.3.2 Simulated output for Affine Transformation (1)

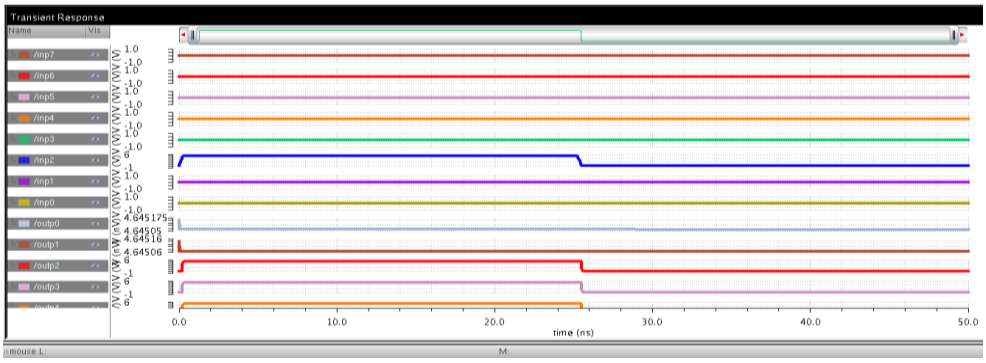


Fig 6.3.3 Simulated output for Affine Transformation (2)

Figure 6.3.4 shows power consumption and Average power consumed is 898.9 pw.

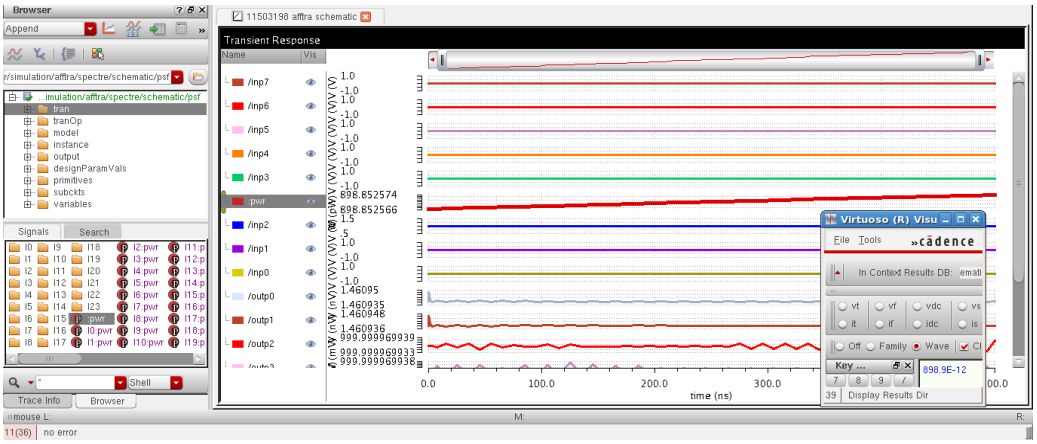


Fig 6.3.4 Power calculation for Affine Transformation

Figure 6.3.5 shows total delay is 9.82 ns.

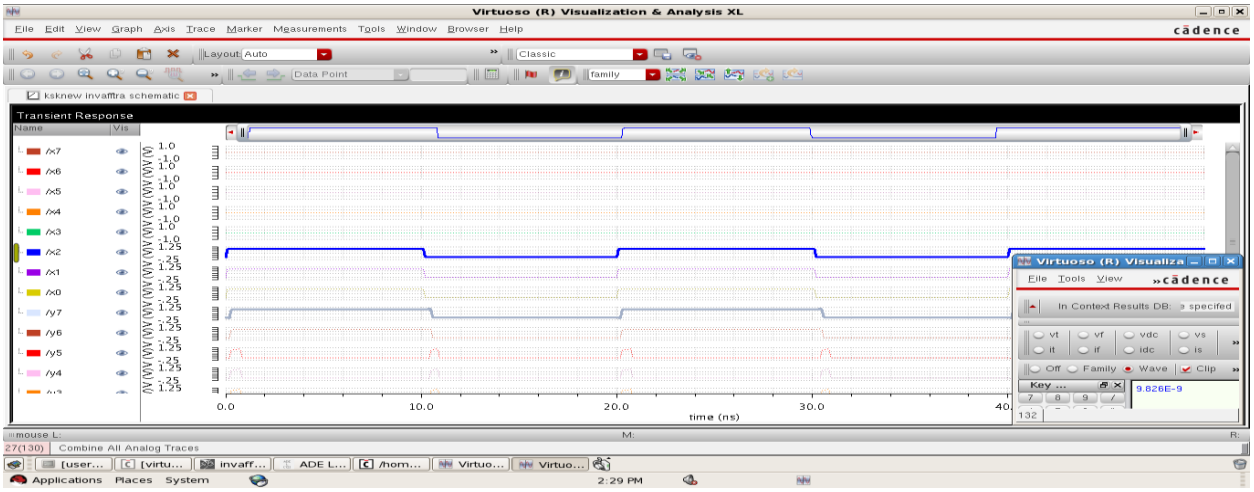


Fig 6.3.5Delay calculation for Affine Transformation

6.4SCHEMATIC FOR SQUARER IN GF (2^4) AND THEIR SIMULATED OUTPUT

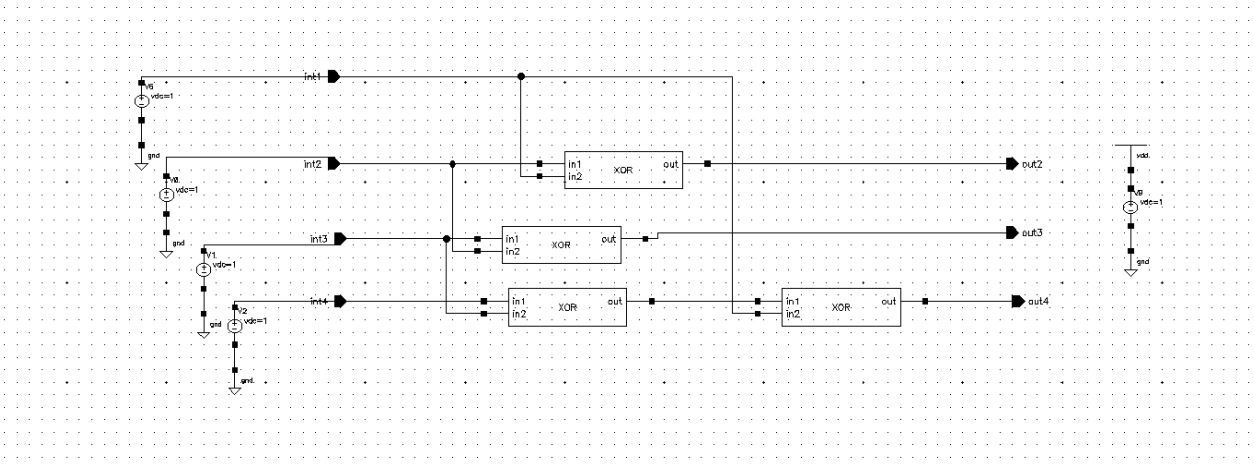


Fig 6.4.1 schematic for Squarer in GF (2^4)

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results forSquarer in GF (2^4). Designed block in cadence is simulated and simulated waveform is described here.

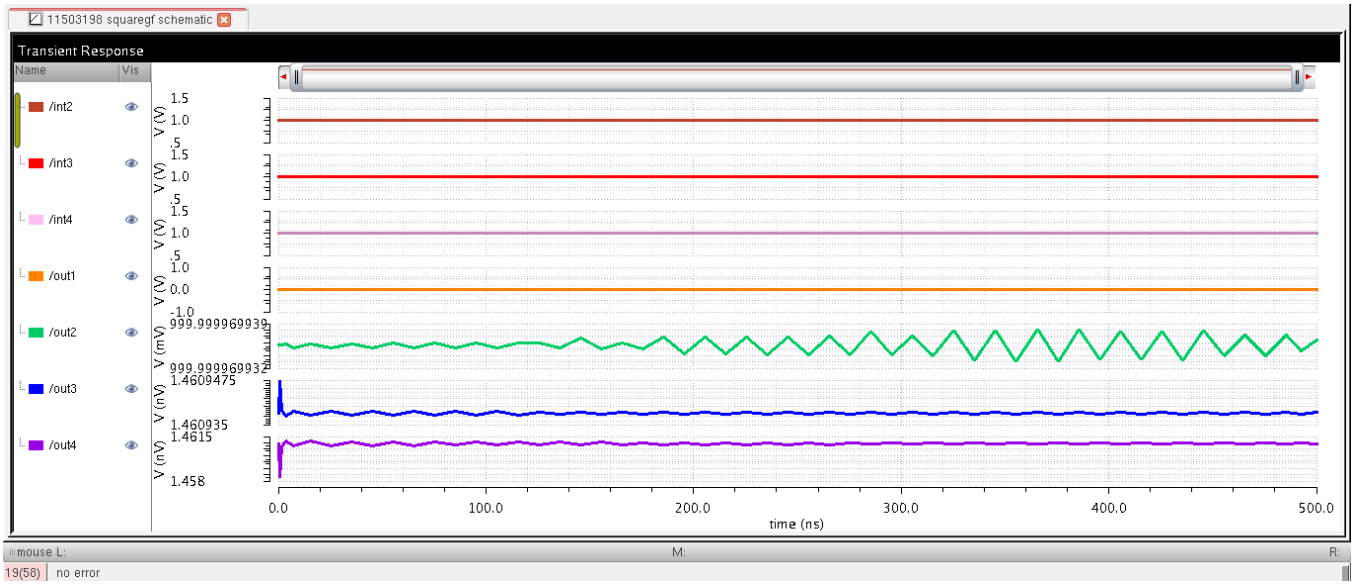


Fig 6.4.2 simulated output for Squarer in GF (2^4)

The figure 6.4.3 shows power consumption and Average power consumed is 122.9 pw.

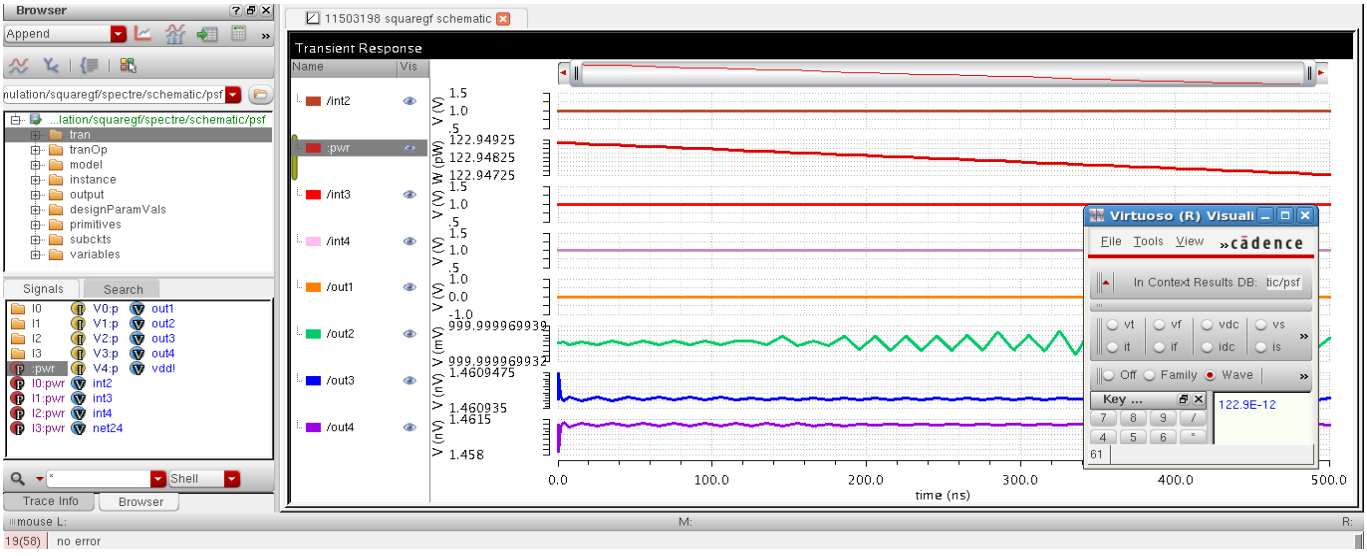


Fig 6.4.3 Power calculation for Squarer in GF (2^4)

The figure 6.4.4 shows delay is 5.776 ns.

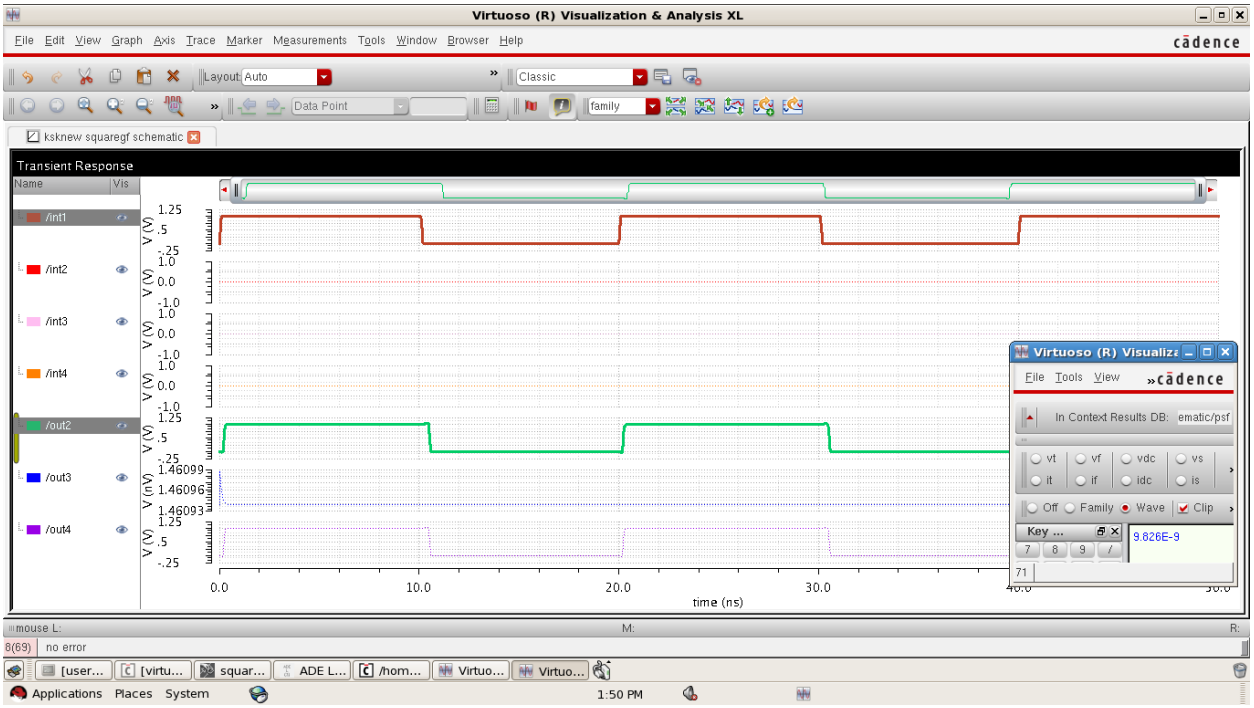


Fig 6.4.4Delay calculation for Squarer in GF (2^4)

6.5SCHEMATIC FOR MULTIPLICATION WITH CONSTANT 'λ' IN GF(2^4) AND THEIR SIMULATED OUTPUT

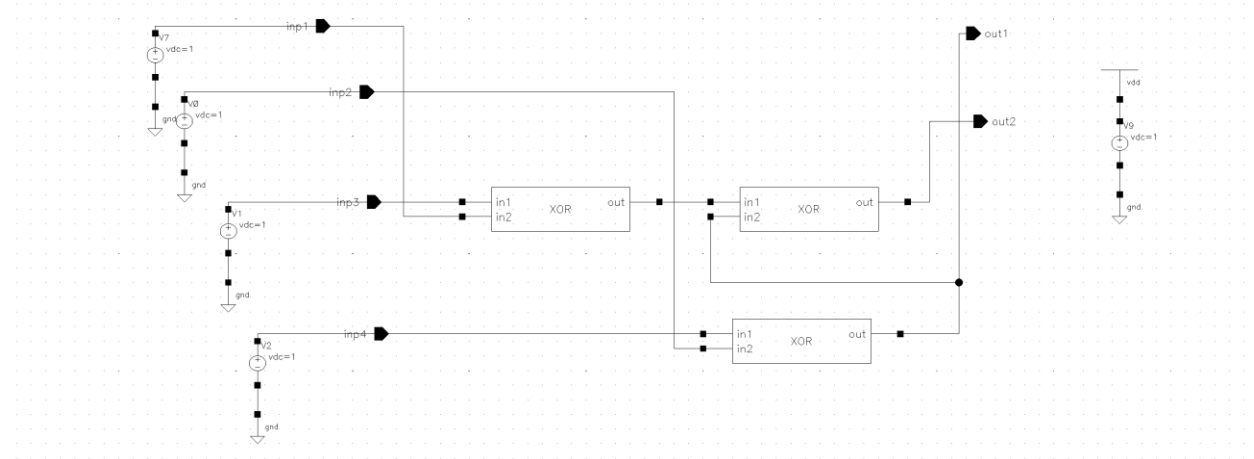


Fig 6.5.1 schematic of multiplication with constant λ in GF (2^4)

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results formultiplication with constant λ in GF (2^4). Designed block in cadence is simulated and simulated waveform is described here.

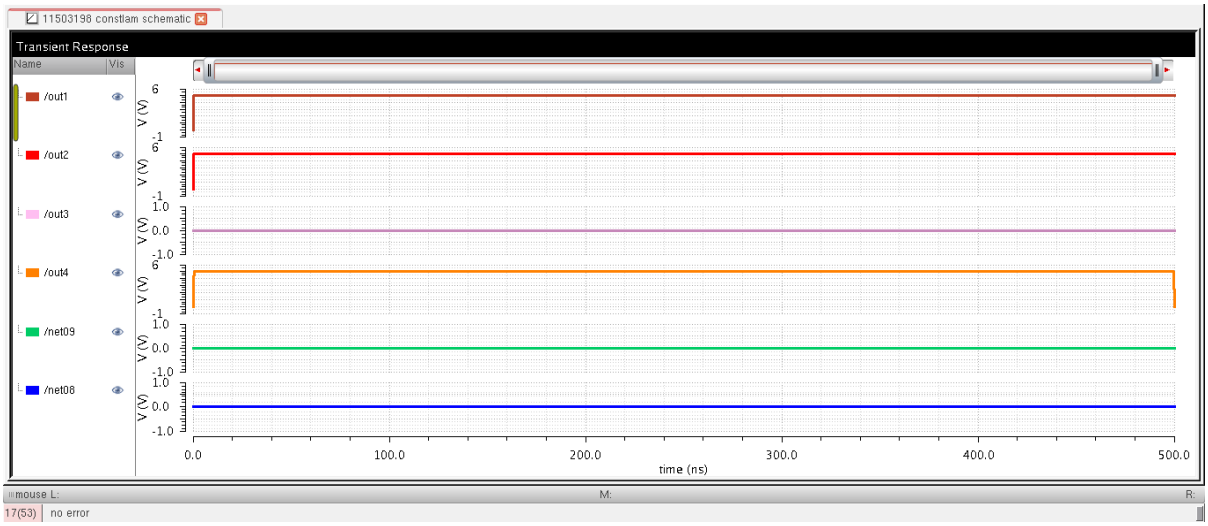


Fig 6.5.2 Simulated output for multiplication with constant λ in GF (2^4)

Figure 6.5.3 shows power consumption and Average power consumed is 9.054 µw.

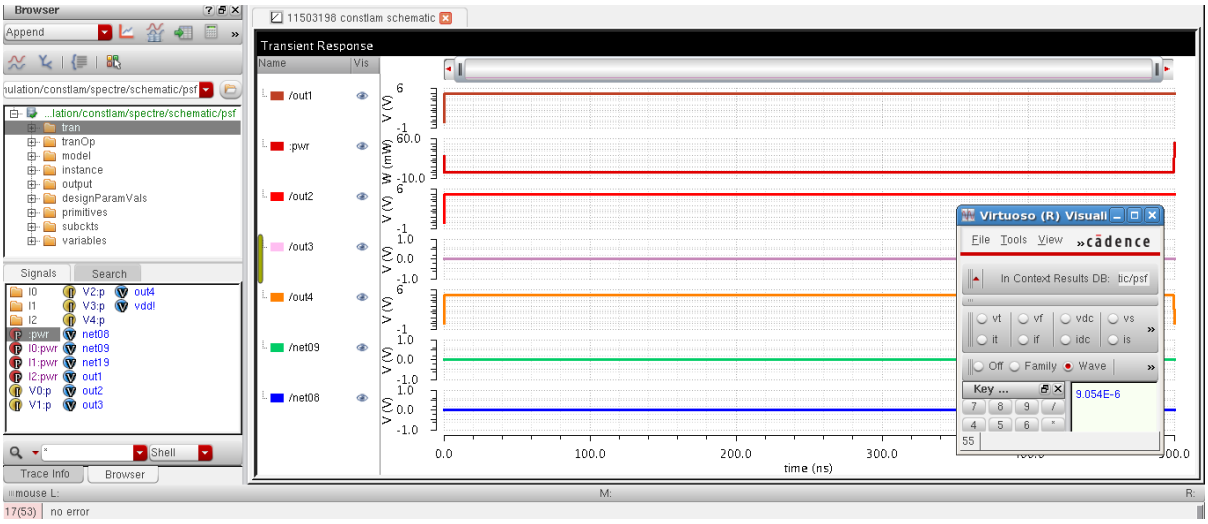


Fig 6.5.3 Power calculation for multiplication with constant λ in GF (2^4)

The figure 6.5.4 shows delayis 9.52 ns.

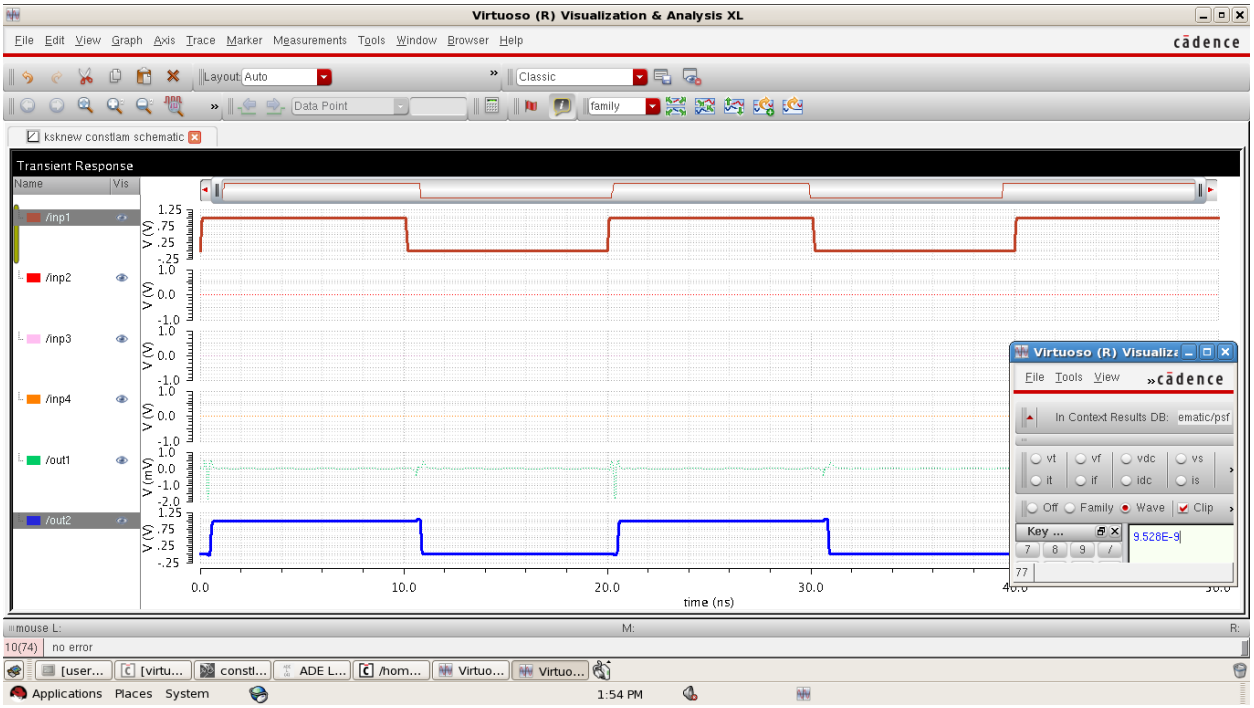


Fig 6.5.4Delay calculation for multiplication with constant λ in GF (2^4)

6.6SCHEMATIC FOR MULTIPLICATION OPERATION IN GF (2^4) AND ITS SIMULATED OUTPUT

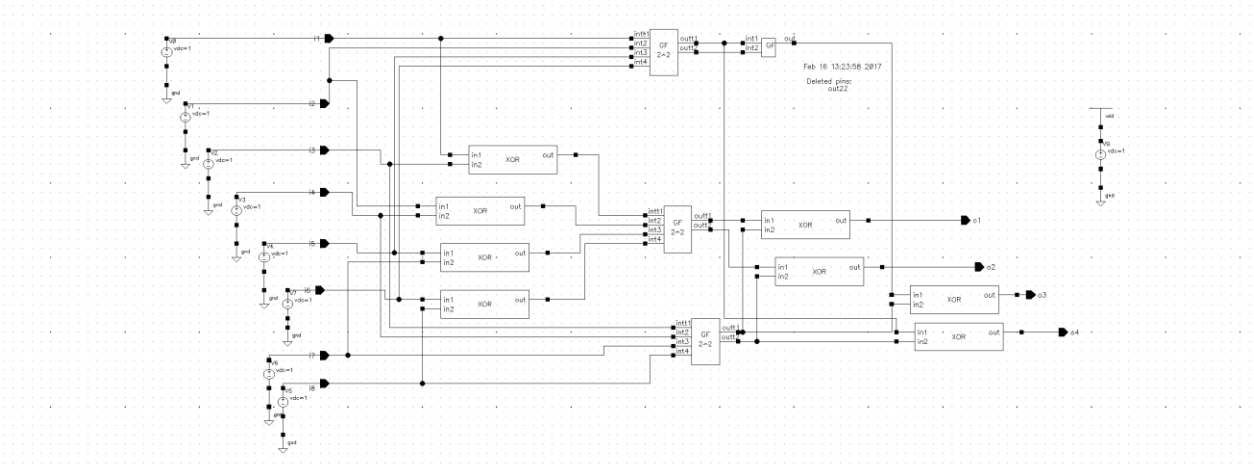


Fig 6.6.1 schematic for multiplication operation in GF (2^4)

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results formultiplication operation in GF (2^4). Designed block in cadence is simulated and simulated waveform is described here.

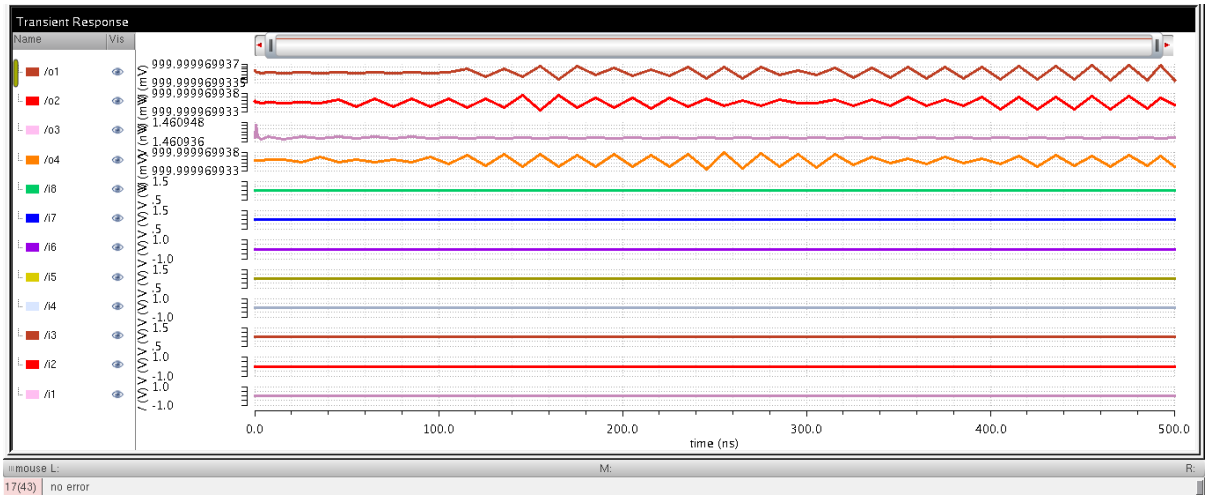


Fig 6.6.2 Simulated output for multiplication operation in GF (2^4)

Figure 6.6.3 shows power consumption and Average power consumed is 805.1 pw.

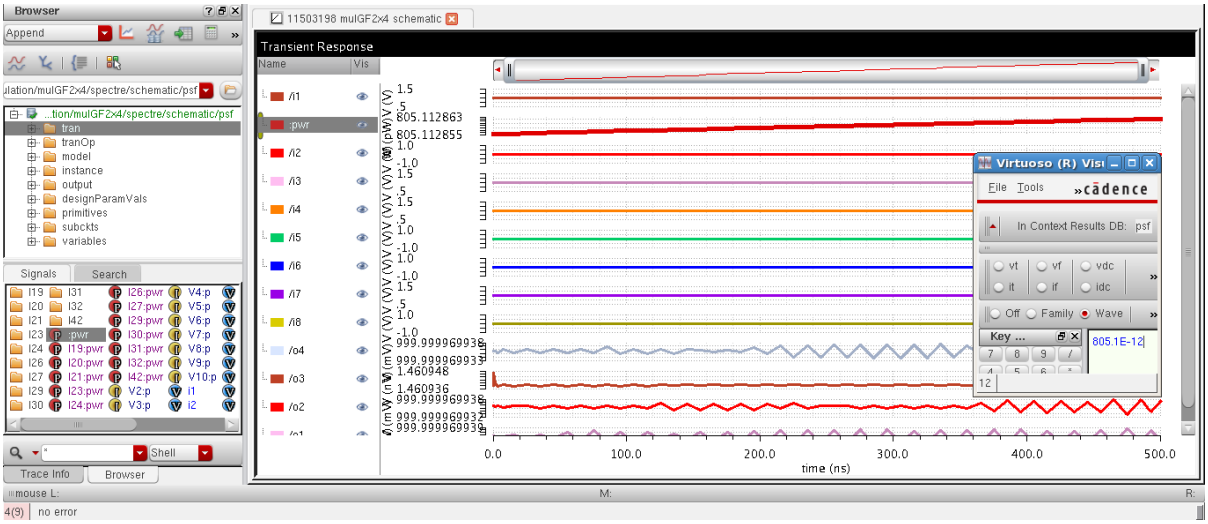


Fig 6.6.3 Power calculation for multiplication operation in GF (2^4)

Figure 6.6.4 shows delay is 9.414 ns.

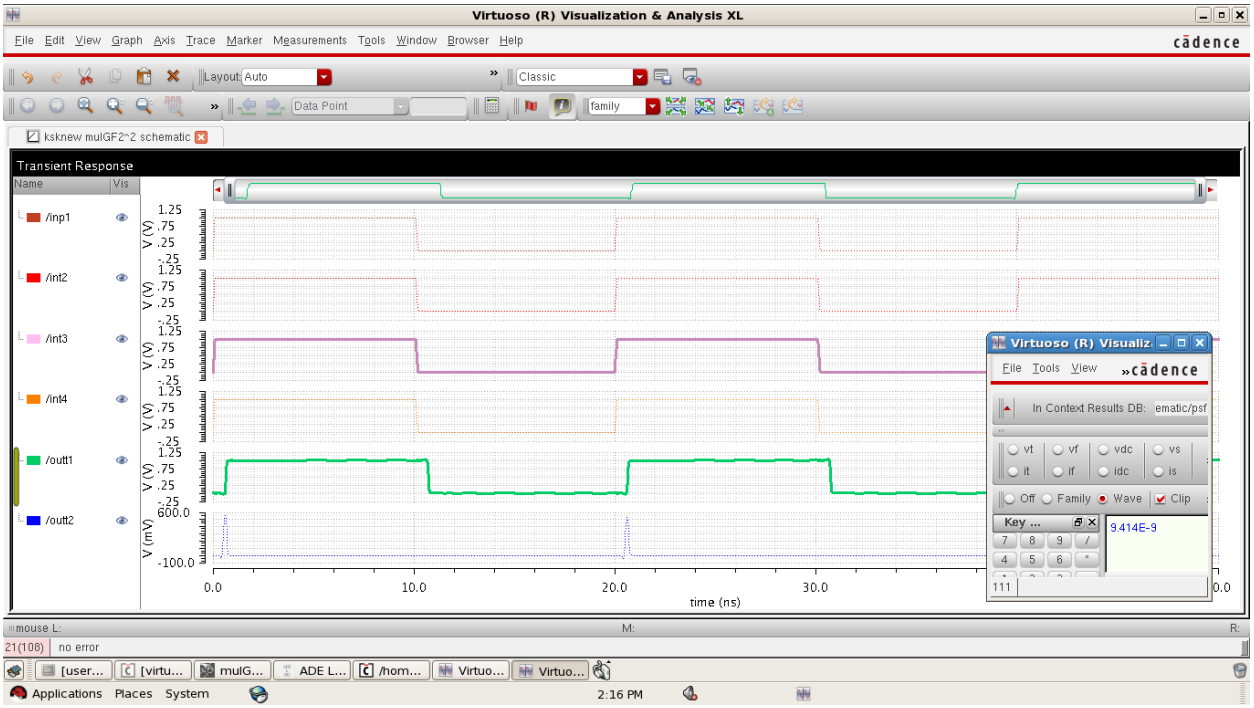


Fig 6.6.4Delay calculation for multiplication operation in GF (2^4)

6.7SCHEMATIC FOR MULTIPLICATIVE INVERSE IN GF (2^4) AND ITS SIMULATED OUTPUT

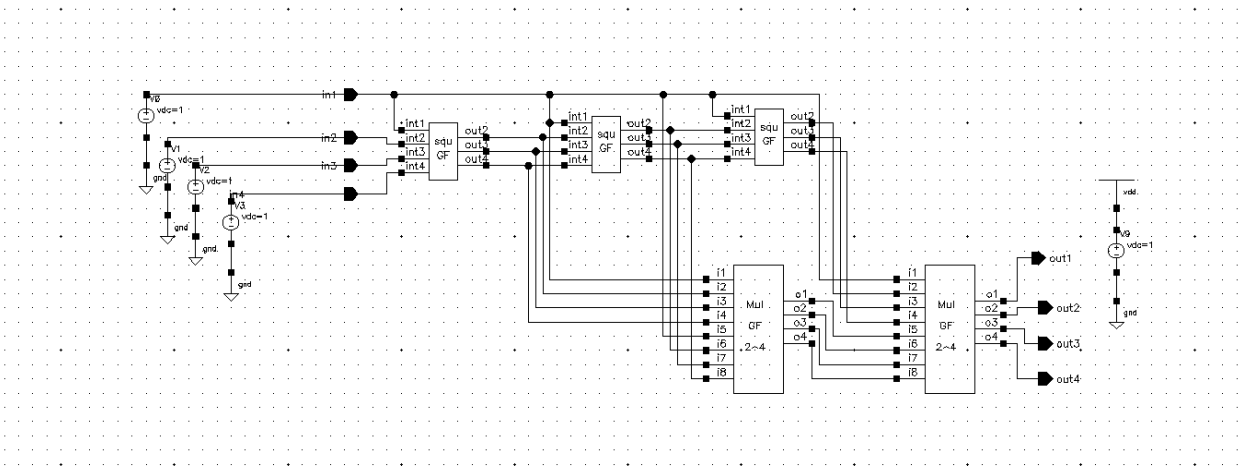


Fig 6.7.1 schematic for Multiplicative Inverse in GF (2^4)

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results forMultiplicative Inverse in GF (2^4). Designed block in cadence is simulated and simulated waveform is described here.

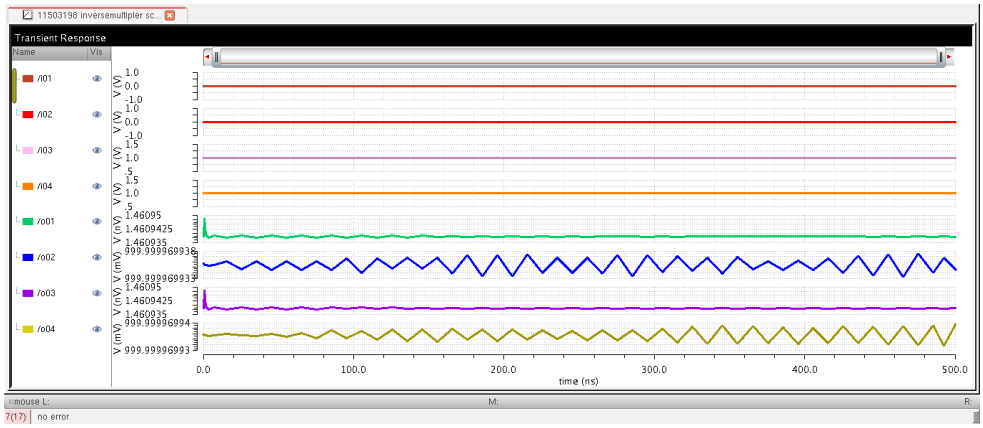


Fig 6.7.2 Simulated output for Multiplicative inverse in GF (2^4)

Figure 6.7.3 shows power consumption and Average power consumed is 2.061 nw.

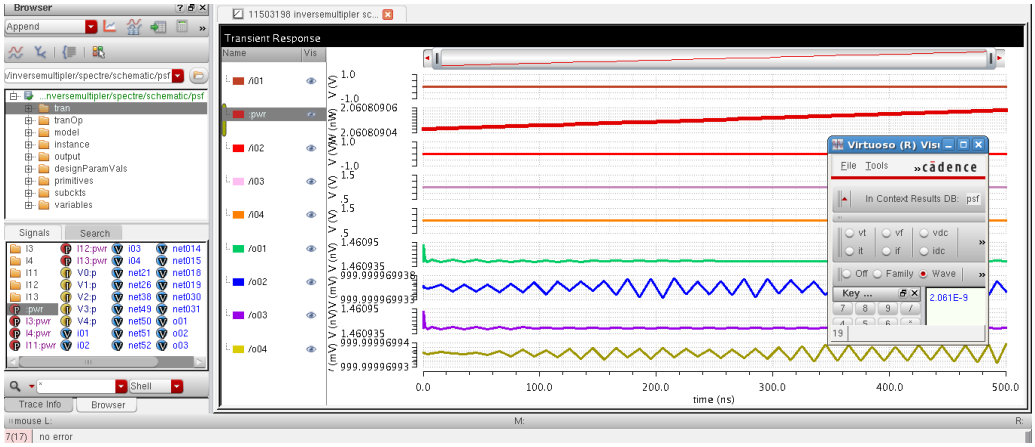


Fig 6.7.3 Power calculation for Multiplicative inverse in GF (2^4)

Figure 6.7.4delayandtotal delay is 6.08 ns.

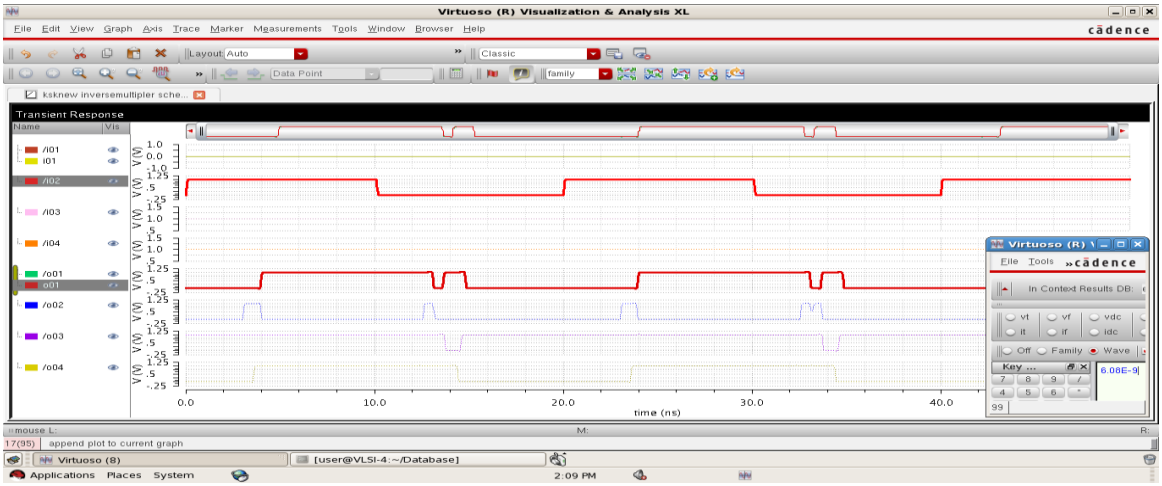


Fig 6.7.4Delay calculation for Multiplicative inverse in GF (2^4)

6.8SCHEMATIC FOR MULTIPLIER OPERATOR OF GF (2^2)

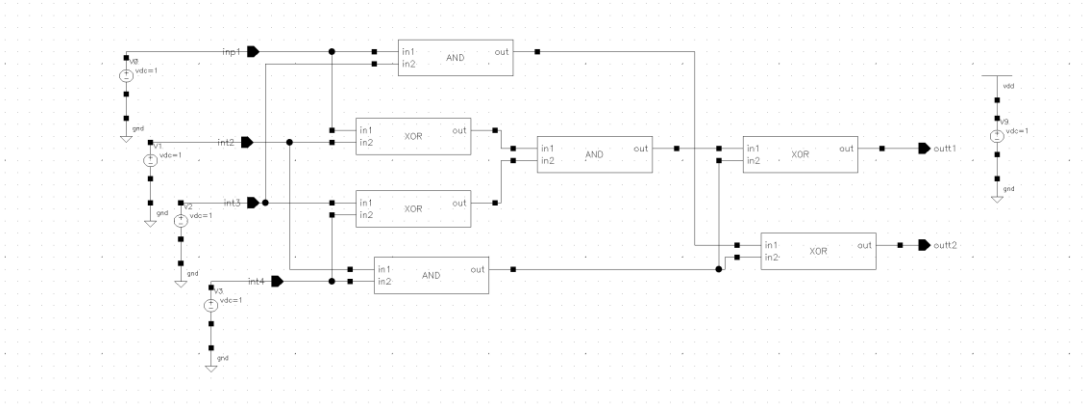


Figure 6.8.1 Schematic for Multiplier operator of GF (2^2)

6.9SCHEMATIC FOR MULTIPLICATION WITH CONSTANT φ

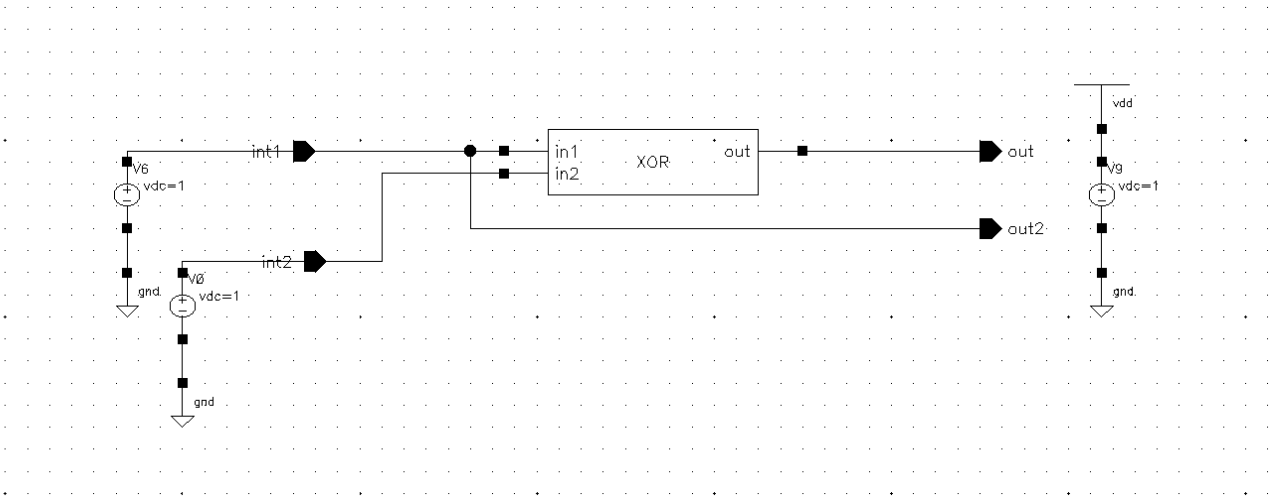


Figure 6.9.1 Schematic for Multiplication with constant φ

6.10SCHEMATIC FOR INVERSE AFFINE TRNSFORMATION AND THEIR SIMULATED OUTPUT

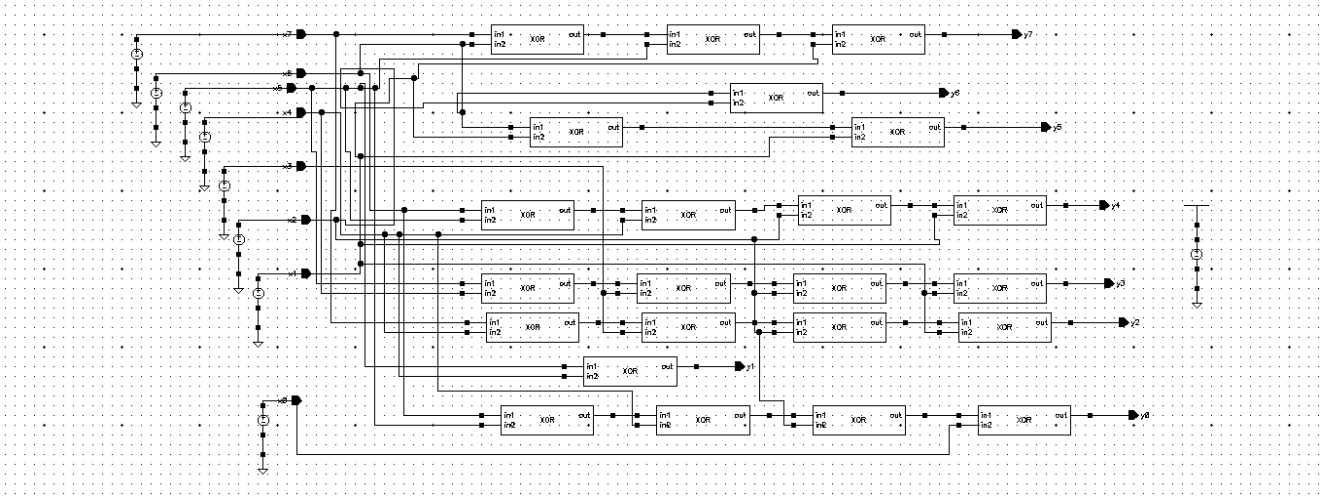


Fig 6.10.1 schematic of Inverse Affine Transformation

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results forSubstitution Box. Designed block in cadence is simulated and simulated waveform is described here.

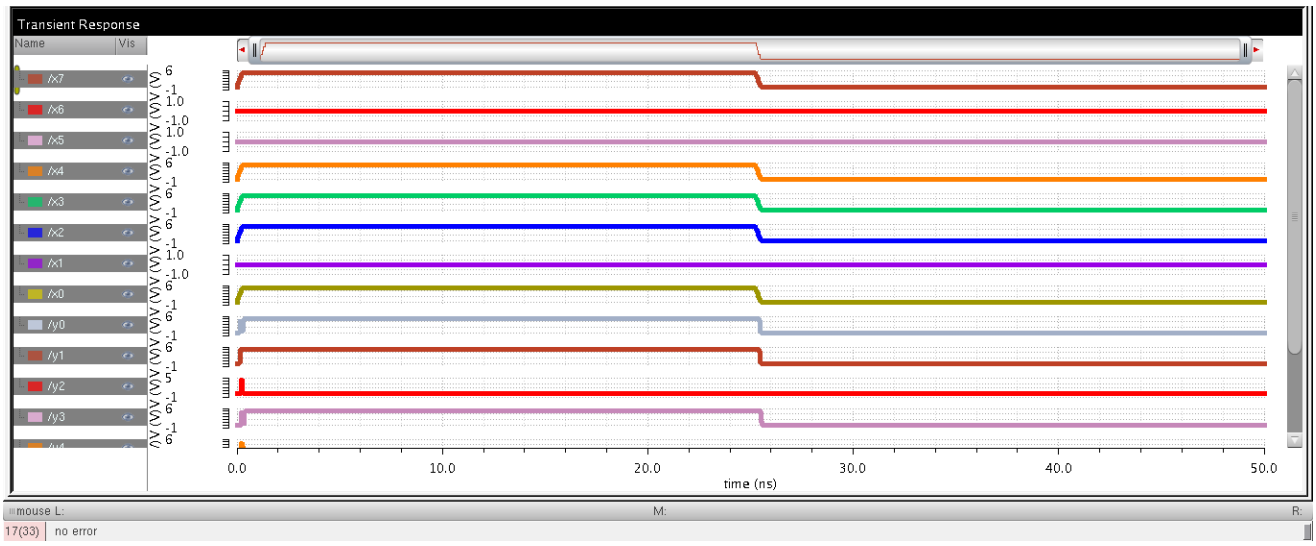


Fig 6.10.2 simulated output for Inverse Affine Transformation (1)

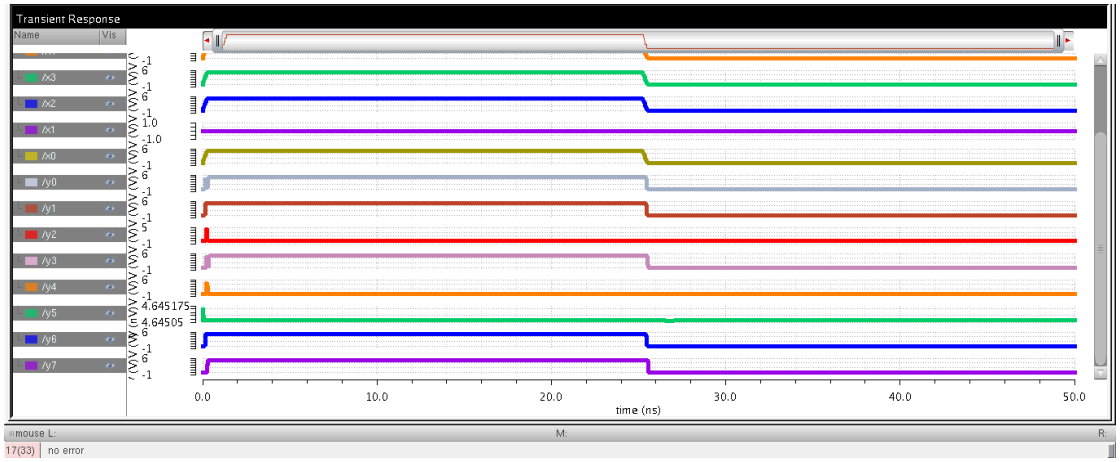


Fig 6.10.3 Simulated output for Inverse Affine Transformation (2)

Figure 6.10.4shows power consumption and Average power consumed 717.7 pw.

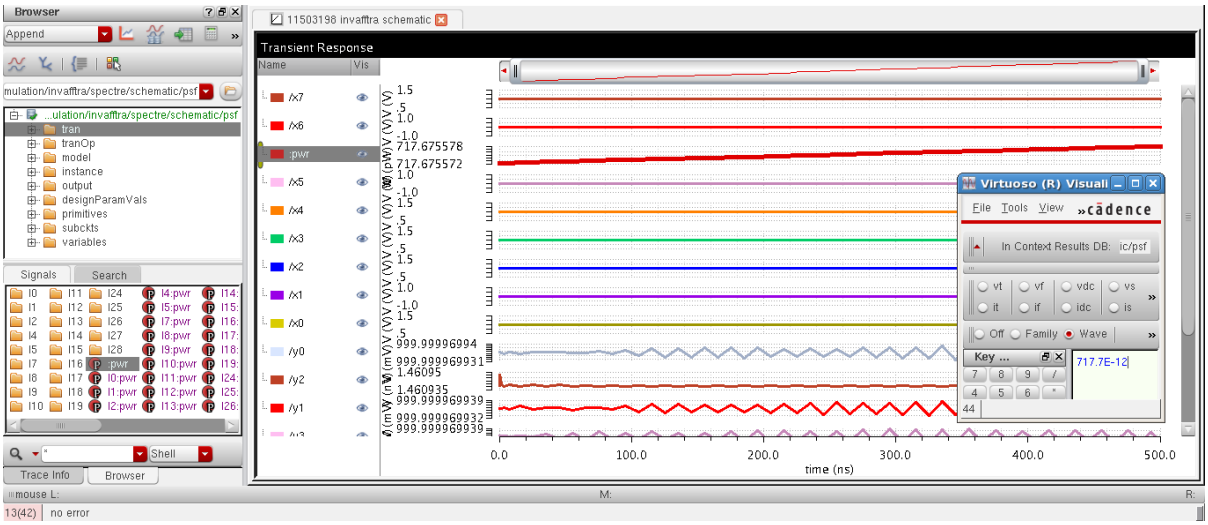


Fig 6.10.4 Power calculation for Inverse Affine transformation

The figure 6.10.5 shows delay is 647.4 ps.

Fig 6.10.5Delay calculation for Inverse Affine transformation

6.11SCHEMATIC FOR S-BOX AND THEIR SIMULATED OUTPUT

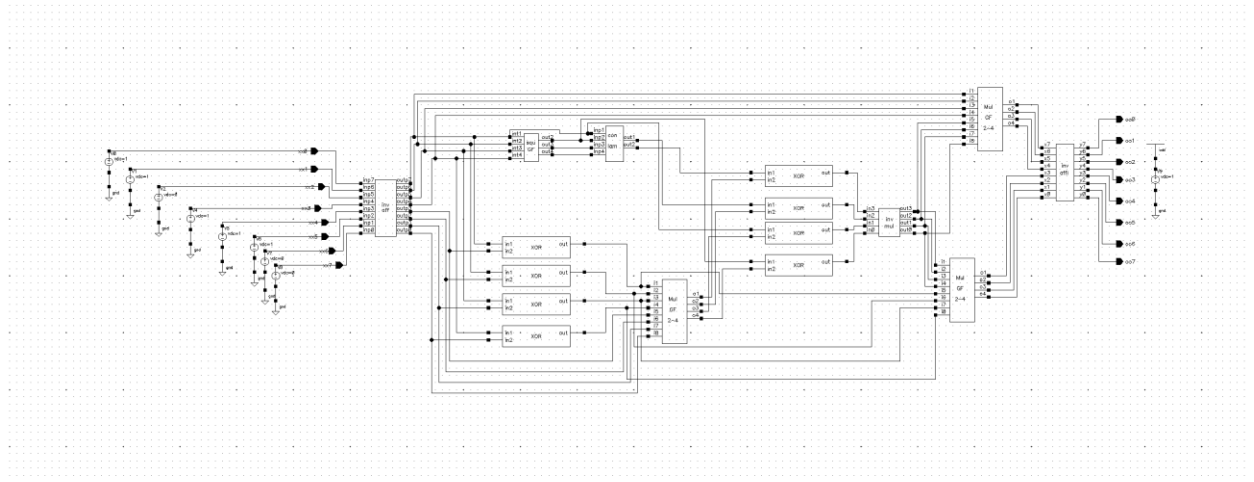


Fig 6.11.1 schematic of Substitution box

We used Cadence tool for simulating our design. In following subsection, I'm describing the simulation results forSubstitution Box. Designed block in cadence is simulated and simulated waveform is described here.

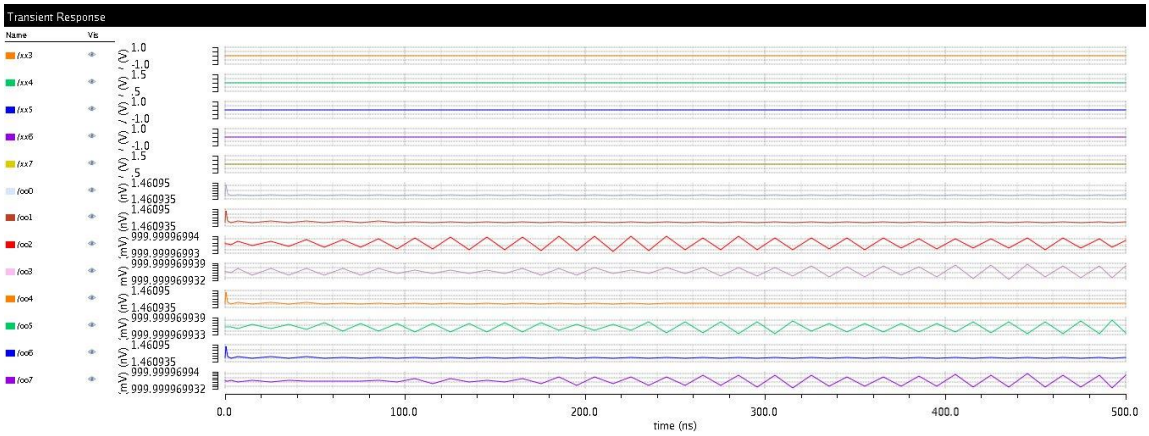


Fig 6.11.2 simulated output for Substitution Box (1)

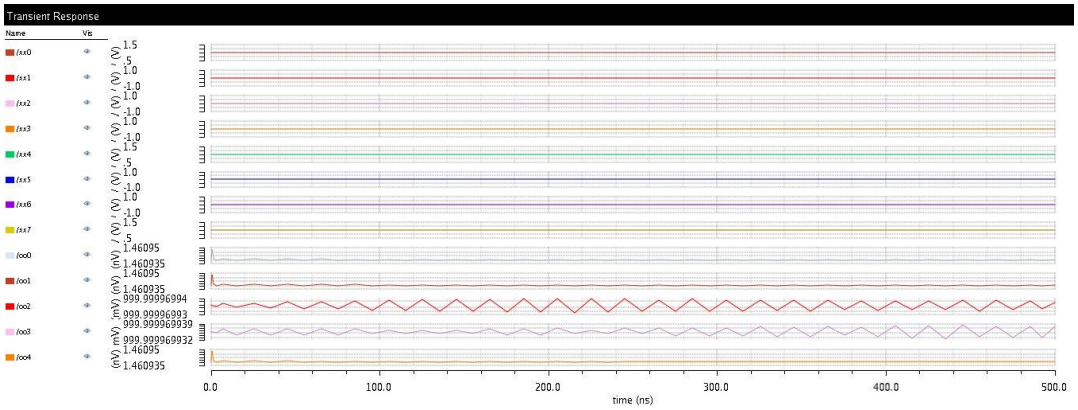


Fig 6.11.3 simulated output for Substitution Box (2)

Figure 6.11.4 shows power consumption and Average power consumed is 5.776 nw.

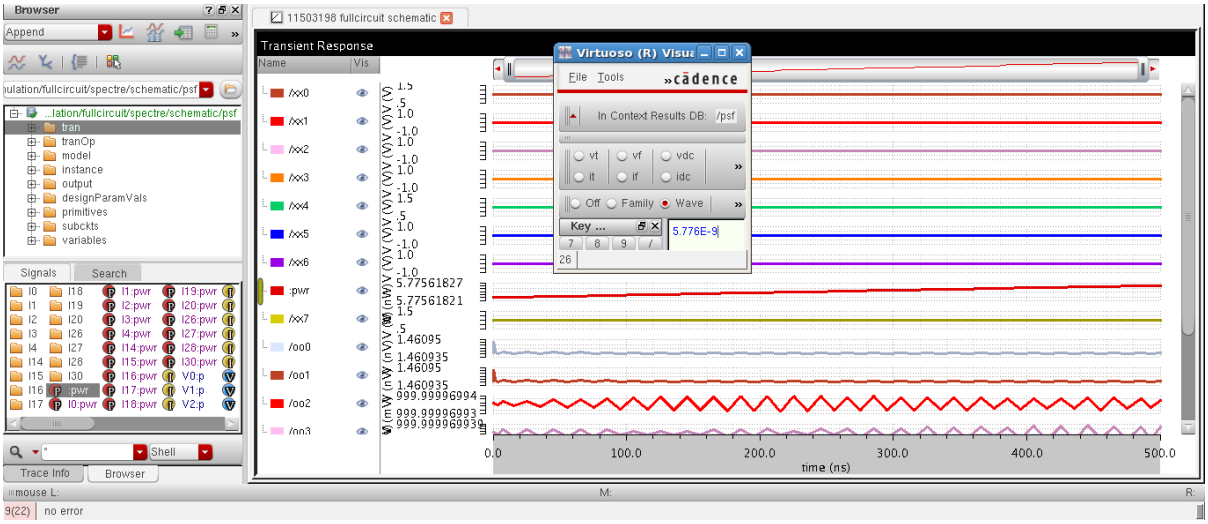


Figure 6.11.4 Power calculation for Substitution Box

Figure 6.11.5 shows delay is 6.015 ns.

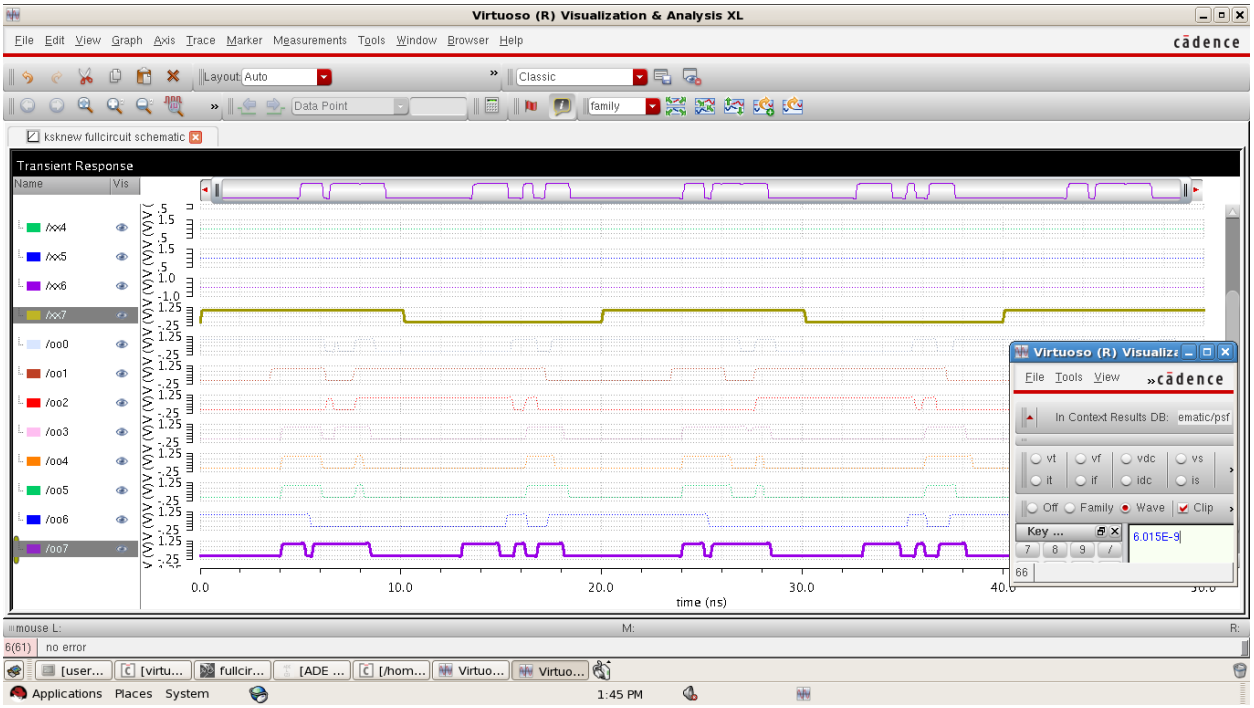


Figure 6.11.5 Delay calculation for Substitution Box

CHAPTER -7

CONCLUSION

we have optimized VLSI architecture of Substitution Box for Advanced Encryption Standards algorithm. In the previous work they tried to working on Look-up-table(LUT) based S-Box there is a more leakage power and it is not secure against CPA attack. But in this work we used composite-field S-Box which is very secure against CPA attack. GF factor multiplication is flexible and powerful. The same GF factor has used as a building block in our Architecture. The masking strategy against differential power attack is also more conventional. The Schematic/Architecture of that Substitution Box in composite field havehigh speed, good delay factor and low areas. In this we have calculated power and delay of each and every block.Accordingly, in the composite field arithmetic wasgrabbedto decrease the complexity byseveral number ofimplementations of inversionin subfield GaliosFactor (2^4) is compared. From theSubstitution Box, Advanced Encryption Standard architecture has been implemented using the merging technique. The architecture of the Substitution Box has implemented in 180 nm technology using Cadence tool fromthe XOR gate which will havehigh speed and low power consuming. The architecture which is designed i.e. was Substitution Box which can consumes 5.776 nw of power and has a delay factor up to 6.01ns.

7.1 FUTURE WORK

- Full custom design of AES.
- Video encryption in real-time

Table 7.1 Results of different research papers of power, delay and power delay product

	[6]	**[7]**	**[13]**	**[16]**	**[19]**	**This Work**
Power	22.6μw	15.3μw	12.1μw	8.6 nw	15.7 nw	5.776nw
Delay	8.2 ns	14.3 ns	12.4 ns	9.2 ns	10.5 ns	6.01ns
PDP	$1.853x10^{-13}$	$2.18x10^{-13}$	$1.50x10^{-13}$	$7.91x10^{-17}$	$1.64x10^{-16}$	$3.41x10^{-17}$

REFERENCES

[1] B.A. Forouzan and D. Mukhopadhyay, Cryptography and Network Security, 2nd Ed.,Tata McGraw Hill, New Delhi, 2012.

[2] M. I. Soliman, G. Y. Abozaid, "FPGA implementation and performance evaluation of a high throughput crypto coprocessor," *Journal of Parallel and Distributed Computing*, Vol. 71 (8), pp.1075-1084, Aug. 2011.

[3] V. K. Pachghare, Cryptography and information security, E. E. Ed., PHI Learning, New Delhi, 2009.

[4] M. Mozaffari-Kermani and A. Reyhani-Masoleh, "A Lightweight High-Performance Fault Detection Scheme for the Advanced Encryption Standard Using Composite Fields," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 19 (1), pp. 85-91, Jan. 2016.

[5] X. Zhang, Keshab. K. Parhi, "High-Speed VLSI Architectures for the AES Algorithm," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 12 (9), pp. 957-967, Sep. 2013.

[6] M. Jridi and A. AlFalou, "A VLSI implementation of a new simultaneous images compression and encryption method," 2010 IEEE International Conference on VLSI, Vol. 12 (9), pp.75-79, July 2014.

[7] Hamdin.O.Alanazi, B.B.Zaidan, A.A.Zaidan and Hamid A.Jalab "VLSI Architectures for the AES Algorithm in Substitution box," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 2 (3), pp. 357-367, Mar. 2015.

[8] Chih-Pin Su, Tsung-Fu Lin, Chih-Tsun Huang, and Cheng-Wen Wu, "A High- Throughput Low-Cost AES," IEEE Communications Magazine, Vol.41 (12), pp.86-91, Dec. 2014.

[9] L. Ali, I. Aris, F. S. Hossain and N. Roy, "Design of an ultra-high speed AES processor for next generation IT security," Computers and Electronics Engineering, Vol.37 (6), pp.1160-1170, Nov. 2014.

[10] K.H. Chang, Y.C. Chen, C. C. Hsieh, C. W. Huang and C. J. Chang, "Embedded a Low Area 32-bit AES for Image Encryption/Decryption Application," IEEE International Symposium on Circuits and Systems, Vol.17 (4), pp. 1922-1925, May 2011.

[11] J. M. G. Criado, M. A. V. Rodriguez, J. M. S. Perez, J. A. G. Pulido, "A new methodology to implement the AES algorithm using dynamic reconfiguration," Integration, the VLSI Journal, Vol.43(1), pp. 72-80, Jan. 2010.

[12] I. Hammad, K. E. Sankary and E. E. Masry, "High-Speed AES Encryptor With Efficient Merging Techniques," IEEE Embedded Systems Letters, Vol.2 (3), pp.67- 71, Sept. 2010.

[13] N. Ahmad, R. Hasan, W. M. Jubadi, "Design of AES S-Box using combinational logic optimization," IEEE Symposium on Industrial Electronics & Applications, Vol.12 (9), pp.696-699, Oct. 2010.

[14] Jarvinen, K., Tommiska, M., and Skytta, "Implementation of AES using Xilinx",*IEEE Transactions on Very Large Scale Integration (VLSI) Systems*,vol.13(7), pp. 207-215, February 2017.

[15] Kimmo Järvinen, Matti Tommiska and Jorma Skyttä, "Full-AES Crypto Design with a twisted BDD S-Box Architecture", IEEE Proceedings - Information Security, vol. 12(7), pp. 686-691, Jul. 2015.

[16] A. Rudra, P.K. Dubey, C.S. Jutla, V. Kumar, J.R. Rao, and P. Rohatgi. "Efficient Rijndael Encryption Implementation with Composite Field Arithmetic", IEEE Symposium on Industrial Electronics & Applications,vol.9(6), pp 175–188, May 2015.

[17] Tim Good and Mohammed Benaissa, "High-speed VLSI Architectures for the AES algorithm", IEEE Transactions on Circuit and Systems-I, Vol. 53(7), pp. 167-171, July 2016.

[18] Pallavi atha, Suresh gyan and V. Raj kumar, "Design and implementation of AES algorithm over FPGA using VHDL", IEEE Symposium on Industrial Electronics & Applications,vol.5(1), pp.58-62, Aug 2016.

[19] Challa vamsi Krishna, N.shiva kumar and Dr.D. subba Rao, "Design Implementation of composite field S-Box using AES-256 Algorithm", IEEE Symposium on Industrial Electronics & Applications,vol.3(12), pp. 43-51, Dec 2016.