A

Project Report On

Design and Implementation of Advanced Encryption Standard

Submitted in partial fulfilment of the requirements for award of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND ELECTRICAL COMMUNICATION ENGINEERING

Under the guidance of

Mr. L. VASUDEVA MURTHY

Head, CED

Of

ECIL-ECIT

By

P.N. VAMSHI (13EC10044)

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

A

Project Report On

Design and Implementation of Advanced Encryption Standard

Submitted in partial fulfilment of the requirements for award of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND COMMUNICATION ENGINEERING

Under the guidance of

Mr. L. VASUDEVA MURTHY

Head, CED

Of

ECIL-ECIT

By

K. MUKESH NAIDU
B. DEVA KUMAR
NATIONAL INSTITUTE OF TECHNOLOGY DURGAPUR

A

Project Report On

Design and Implementation of Advanced Encryption Standard

Submitted in partial fulfilment of the requirements for award of the degree of

BACHELOR OF TECHNOLOGY

In

ELECTRONICS AND COMMUNICATION ENGINEERING

Under the guidance of

Mr. L. VASUDEVA MURTHY

Head, CED

Of

ECIL-ECIT

By

J. HARSHAVARDHAN REDDY

(NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA)

DECLARATION

We hereby declare that the project entitle **Design and Implementation of Advanced Encryption Standard** submitted in partial fulfilment of the requirements for the award of degree of **Bachelor of Technology** in **Electronics and Electrical Communication Engineering**. This dissertation is our original work and the project has not formed the basis for the award of any degree, associate ship, fellowship or any other similar titles and no part of it has been published or sent for the publication at the time of submission.

P.N. VAMSHI (IIT KHARAGPUR)
K. MUKESH NAIDU (NIT DURGAPUR)
B. DEVA KUMAR (NIT DURGAPUR)
J. HARSHAVARDHAN REDDY (NIT ROURKELA)

ACKNOWLEDGEMENT

We wish to take this opportunity to express our deep gratitude to all those who helped, encouraged, motivated and have extended their cooperation in various ways during our project work. It is our pleasure to acknowledgement the help of all those individuals who was responsible for foreseeing the successful completion of our project.

We would like to thank **Mr. L. VASUDEVA MURTHY (Head, CED)** and express our gratitude with great admiration and respect to our project guide **Mr. CH GOUTHAM RAJ** for their valuable advice and help throughout the development of this project by providing us with required information without whose guidance, cooperation and encouragement, this project couldn't have been materialized.

Last but not the least we would like to thank the entire respondents for extending their help in all circumstances.

P.N. VAMSHI (IIT KHARAGPUR)
K. MUKESH NAIDU (NIT DURGAPUR)
B. DEVA KUMAR (NIT DURGAPUR)
J. HARSHAVARDHAN REDDY (NIT ROURKELA)

ABSTRACT

Advanced Encryption Standard (AES), a Federal Information Processing Standard (FIPS), is an approved cryptographic algorithm that can be used to protect electronic data. The AES can be programmed in software or built with pure hardware. However, Field Programmable Gate Arrays (FPGAs) offer a quicker and more customizable solution. This project presents the AES algorithm with regard to FPGA and Verilog language. Xilinx 12.1 and Module Sim software is used for simulation and optimization of the synthesizable Verilog code. Synthesizing and implementation (i.e. Translate, Map and Place and Route) of the code is carried out on Xilinx - Project Navigator, ISE 12.1 suite. All the transformations of both Encryption and Decryption are simulated using an iterative design approach in order to minimize the hardware consumption. Xilinx XC3S500 device of Spartan Family is used for hardware evaluation. This project proposes a method to integrate the AES encrypted and the AES decrypted. This method can make it a very low-complexity architecture, especially in saving the hardware resource in implementing the AES Sub Bytes module and Mix columns module etc. Most designed modules can be used for both AES encryption and decryption. Besides, the architecture can still deliver a high data rate in both encryption/decryption operations. The proposed architecture is suited for hardware-critical applications, such as GPON network security, ATM Machines, smart card, PDA, and mobile phone, etc.

Organization Profile

ECIL was setup under the Department of Atomic Energy on 11th April, 1967 with a view to generate a strong indigenous capability in the field of professional grade electronic. The initial accent was on self-reliance and ECIL was engaged in the Design Development Manufacture and Marketing of several products emphasis on three technology lines viz. Computers, control systems and communications. ECIL thus evolved as a multi-product company serving multiple sectors of Indian economy with emphasis on import of country substitution and development of products and services that are of economic and strategic significance to the country.

Over the year, ECIL pioneered the development of various complex electronics products without any external technological help and scored several **"FIRSTS"** in these fields prominent among them being country's

- First Digital Computer
- First Solid State TV
- First Control & Instrumentation of Nuclear Power plants
- First Earth Station Antenna
- First Computerized Operator Information System
- First Radiation Monitoring & Detecting System
- First Automatic Message Switching System
- First Operation & Maintenance Centre for E-108 Exchange
- First Programmable Logic Controller
- First Solid State Cockpit Voice Recorder
- First Electronic Voting Machines

Mission

ECIL's mission is to consolidate its status as a valued national asset in the area of strategic electronics with specific focus on Atomic Energy, Defence, Security and such critical sectors of strategic national importance.

Objectives

- To continue services to the country's needs for the peaceful uses Atomic Energy. Special and Strategic requirements of Defence and Space, Electronics Security System and Support for Civil aviation sector.

- To establish newer Technology products such as Container Scanning Systems and Explosive Detectors.
- To re-engineer the company to become nationally and internationally competitive by paying particular attention to delivery, cost and quality in all its activities.
- To explore new avenues of business and work for growth in strategic sectors in addition to working realizing technological solutions for the benefit of society in areas like Agriculture, Education, Health, Power, Transportation, Food, Disaster Management etc.

Divisions

The Company is organized into divisions serving various sectors, national and Commercial Importance. They are Divisions serving nuclear sector like Control & Automation Division (CAD), Instruments & Systems Division (ISD), Divisions Serving defence sector like Communications Division (CND), Antenna Products Division (APD), Servo Systems Division (SSD) etc., Divisions handling Commercial Products are Telecom Division (TCD), Customer Support Division (CSD), Computer Education Division (CED).

Exports

ECIL is currently operating in major business EXPORT segments like Instruments and systems design, Industrial/Nuclear, Servo Systems, Antenna Products, Communication, Control and Automation and several other components.

Services

The company played a very significant role in the training and growth of high calibre technical and managerial manpower especially in the fields of Computers and Information Technology. Though the initial thrust was on meeting the Control & Instrumentation requirements of the Nuclear Power Program, the expanded scope of self-reliance pursued by ECIL enabled the company to develop various products to cater to the needs of Defence, Civil Aviation, Information & Broadcasting, Tele communications, etc.

INTRODUCTION TO AES:

The **Advanced Encryption Standard** (**AES**), also referenced as **Rijndael** (its original name), is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001.

For AES, NIST selected three members of the Rijndael family, each with a block size of 128 bits, but three different key lengths: 128, 192 and 256 bits.

AES has been adopted by the U.S. government and is now used worldwide. It supersedes the Data Encryption Standard (DES), which was published in 1977. The algorithm described by AES is a symmetric-key algorithm, meaning the same key is used for both encrypting and decrypting the data.

In the United States, AES was announced by the NIST as U.S. FIPS PUB 197 (FIPS 197) on November 26, 2001.This announcement followed a five-year standardization process in which fifteen competing designs were presented and evaluated, before the Rijndael cipher was selected as the most suitable.

AES became effective as a federal government standard on May 26, 2002 after approval by the Secretary of Commerce. AES is included in the ISO/IEC 18033-3 standard. AES is available in many different encryption packages, and is the first publicly accessible and open cipher approved by the National Security Agency (NSA) for top secret information when used in an NSA approved cryptographic module.

INTRODUCTION TO CRYPTOGRAPHY:

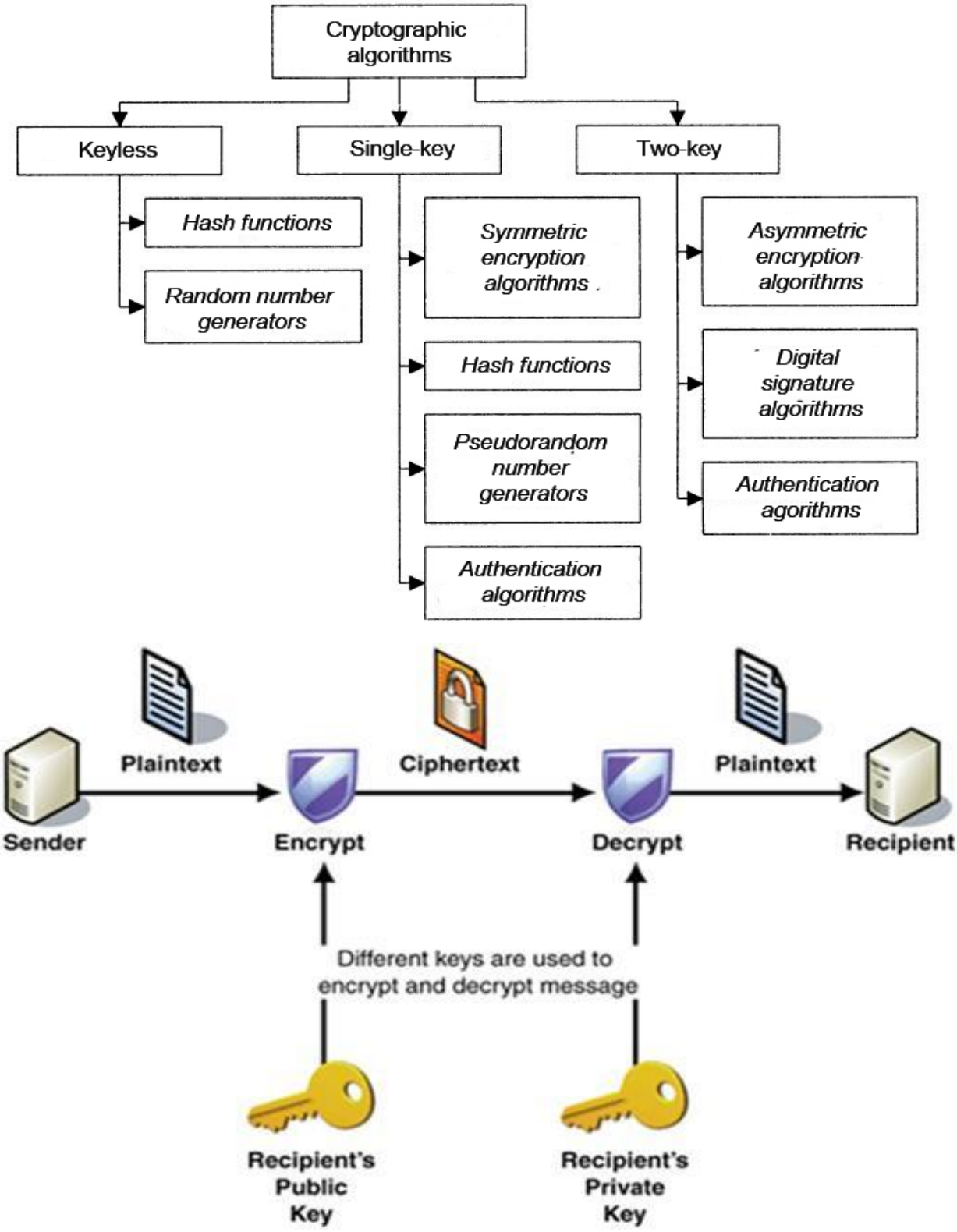
Cryptography is the science of secret codes, enabling the confidentiality of Communication through an insecure channel. It protects against unauthorized parties by preventing unauthorized alteration of use. Generally speaking, it uses a cryptographic system to transform a plaintext into a cipher text, using most of the time a key.

In a broader sense Cryptography is best known as a way of keeping the contents of a message secret. Confidentiality of network communications, for example, is of great importance for e-commerce and other network applications. However, the applications of cryptography go far beyond simple confidentiality. In particular, cryptography allows the network business and customer to verify the authenticity and integrity of their transactions. If the trend to a global electronic marketplace continues, better cryptographic techniques will have to be developed to protect business transactions. Sensitive information sent over an open network may be scrambled into a form that cannot be understood by a hacker or

eavesdropper. This is done using a mathematical formula, known as an encryption algorithm, which transforms the bits of the message into an unintelligible form. The intended recipient has a decryption algorithm for extracting the original message. There are many examples of information on open networks, which need to be protected in this way, for instance, bank account details, credit card transactions, or confidential health or tax records. Cryptosystems can provide confidentiality, authenticity, integrity, and non-repudiation services. It does not provide availability of data or systems.

- ***Confidentiality*** means that unauthorized parties cannot access information.
- ***Authenticity*** refers to validating the source of the message to ensure the sender is properly identified.
- ***Integrity*** provides assurance that the message was not modified during transmission, accidentally or intentionally.
- ***Non repudiation*** means that a sender cannot deny sending the message at a later date, and the receiver cannot deny receiving it. So if your boss sends you a message telling you that you will be receiving a raise that doubles your salary and it is encrypted, encryption methods can ensure that it really came from your boss, that someone did not alter it before it arrived to your computer, that no one else was able to read this message as it travelled over the network, and that your boss cannot deny sending the message later when he comes to his senses.

CLASSIFICATION OF CRYPTOGRAPHY:



TYPES OF CIPHER:

There are two classes of algorithm in encryption, an asymmetric key and symmetric key. The following subsections describe the both classes and a brief discussion of algorithms is added as well.

ASYMMETRIC KEY OR PUBLIC KEY:

In an asymmetric key algorithm, there are two keys. One must be public and it is used to encrypt the data. The other key is a private one and it is used to decrypt the information. In communication between **A** and **B**, **A** uses the public key *Ke* of **B** to encrypt the message, in a way that only **B** (*neither **A***) can decrypt this message using his private key *Kd*. This system is also used to sign a message digitally (Mao, 2003). Rivest-Shamir-Adleman (RSA) is widely used asymmetric key algorithm for decades and Elliptic Curve Cryptography (ECC) as an alternative to RSA which offers highest security with small bit length of key.

SYMMETRIC KEY OR PRIVATE KEY:

In a symmetric or private key algorithm, in the ordinary case, the communication only uses only one key. A user **A** sends the secret private key *K*c to a **B** user before the start of the communication between them. Both sides use the same private key to encrypt and decrypt the Exchanged information. Data Encryption Standard (DES) and CAST128 are example of symmetric key algorithm.

CLASSIFICATION OF PRIVATE KEY CRYPTOGRAPHY:

There are two classes of private-key cryptography scheme which are commonly distinguished as block ciphers and stream ciphers.

STREAM CIPHER:

A stream cipher is a type of symmetric encryption algorithm. Stream ciphers can be designed to be exceptionally fast, much faster than any block cipher. While block ciphers operate on large blocks of data, stream ciphers typically operate on smaller units of plaintext, usually bits. The encryption of any particular plaintext with a block cipher will

result in the same cipher text when the same key is used. With a stream cipher, the transformation of these smaller plaintext units will vary, depending on when they are encountered during the encryption process. A stream cipher generates what is called a key stream (*a sequence of bits used as a key*). Encryption is accomplished by combining the key stream with the plaintext, usually with the bitwise XOR operation. The generation of the key stream can be independent of the plaintext and cipher text, yielding what is termed a synchronous stream cipher, or it can depend on the data and its encryption, in which case the stream cipher is said to be self-synchronizing. Most stream cipher designs are for synchronous stream ciphers (Stinson, 2002).

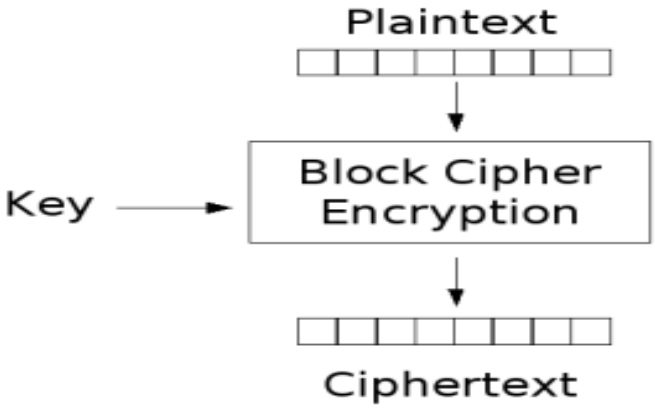
Key Features of Stream Ciphers:

- Stream cipher treats the message as a stream of bits and performs mathematical functions on them individually.
- Operate on small units of plaintext, bits
- Symmetric encryption
- Usually implemented in hardware
- Encrypts by operating on a continuous data stream
- Some stream cipher use stream generator
- Statistically unpredictable
- Much faster than any block cipher
- Effective Stream algorithm contains
- Long period of no repeating patterns within key stream values
- Statistically unpredictable
- The key stream is not linearly related to the key

BLOCK CIPHER

Block cipher is a type of symmetric-key encryption algorithm that transforms a fixed-Length block of plaintext data into a block of cipher text data of the same length. This Transformation takes place under the action of a user-provided secret key. Decryption is Performed by applying the reverse transformation to the cipher text block using the same secret key. The fixed length is called the block size, and for many block ciphers, the block

size is 64 and the block size increase to 128, 192 or 256 bits as processors become more sophisticated. The cipher like DES, Triple-DES and Blowfish are example of block cipher.



Key Features of Block Ciphers:

- Operate on fixed size blocks of plain text
- Breaks the plaintext into blocks and encrypts each with the same algorithm
- Apply an identical encryption algorithm and key to each block
- The properties of a cipher should contain confusion and diffusion
- Spread the plaintext character over many cipher text characters. Done using permutations
- Different unknown key values cause confusion putting the bits within the plaintext through many functions cause diffusion
- Accomplished through p-boxes
- Conceals statistical connection using substitution
- Accomplished through s-boxes
- Block cipher use S-boxes. An S-box is non-linear because it generates a 4-bits output string from 6 bits' input
- Are more suitable for software implementations, because they work with blocks of data which is usually the width of a data bus (64 bits).

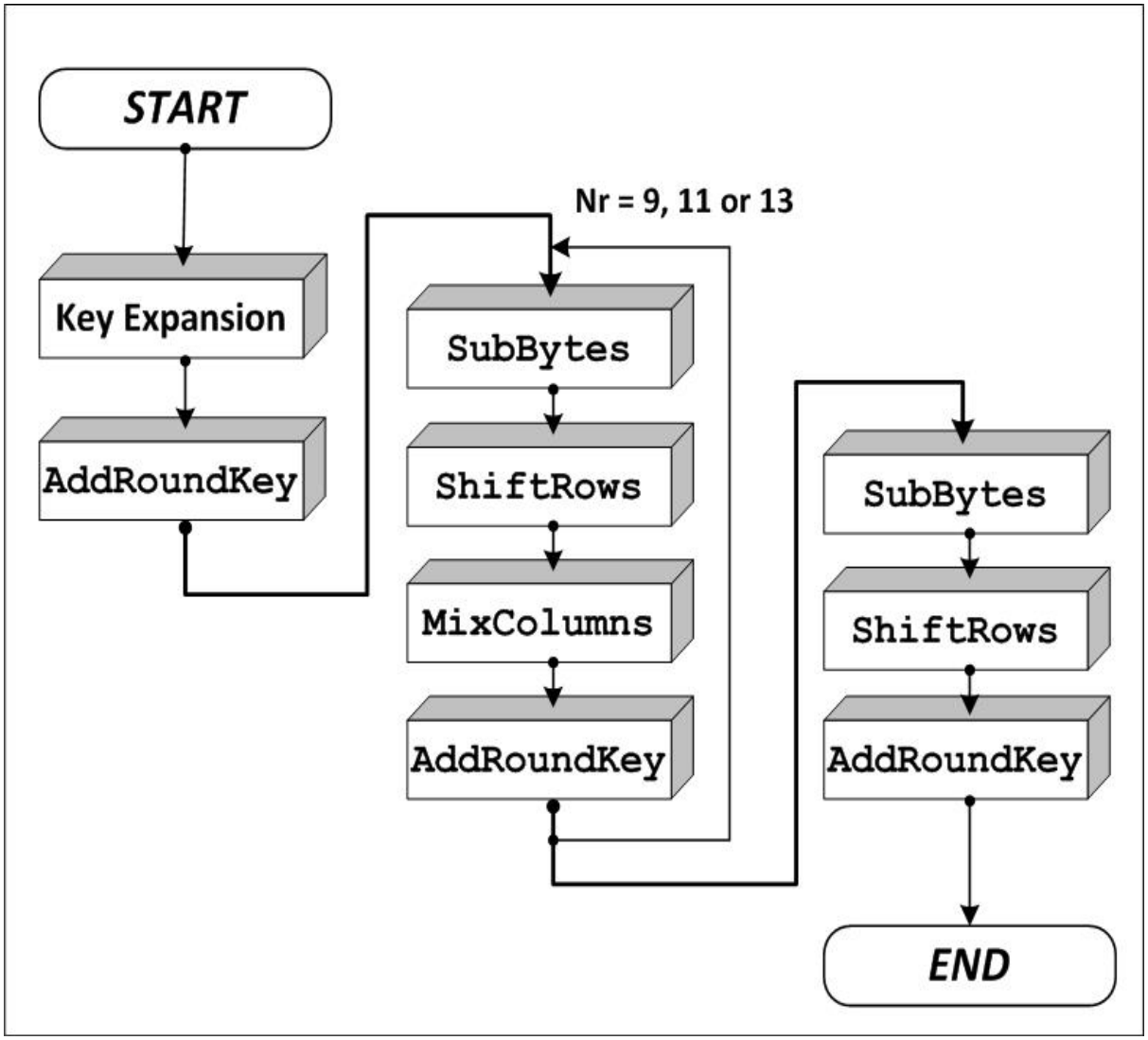
ADVANCED ENCRYPTION STANDARD (AES) CRYPTOGRAPHY

2.1 EVOLUTION OF AES

DES is now considered to be insecure for many applications. This is chiefly due to the 56-bit key size being too small; DES keys have been broken in less than 24 hours. There are also some analytical results which demonstrate theoretical weaknesses in the cipher, although they are infeasible to mount in practice. The algorithm is believed to be practically secure in the form of Triple DES, although there are theoretical attacks. In recent years, the cipher has been Superseded by the Advanced Encryption Standard (AES).

The Advanced Encryption Standard (AES) Algorithm, adopted by the U.S. government in 2001, is a block cipher transforms 128-bit data blocks under a 128-bit, 192-bit or 256-bit secret key, by means of permutation and substitution. In January 1997, the National Institute of Standards and Technology (NIST) announced the initiation of an effort to develop the AES and made a formal call for algorithms on September 12, 1997. After reviewed the results of this Preliminary research, the algorithms MARS, RC6TM, Rijndael, Serpent and Two fish were selected as finalist. And further reviewed public analysis of the finalist, NIST has decided to propose Rijndael as the new Advanced Encryption Standard (AES) on 2nd October 2000. It is expected to replace the DES and Triple DES so as to fulfil the stricter data security requirement because its enhanced security levels.

In the summer of 2001, AES replaced the aging DES as the Federal Information Processing Encryption Standard (FIPS). DES is seen as reaching the end of its life, as cracking of its cipher is seen to be more tractable on current computer hardware. The AES algorithm will be used for many applications within the government an in the private sector. Breaking an AES encrypted cipher text by trying all possible keys is currently computationally infeasible with technology advances.



2.3 BACKGROUND MATHEMATICS:

This section provides a brief introduction to the fundamental mathematical concepts of finite fields needed to understand. For in-depth discussion on the subject, one should refer to Joan Daemen (1999), BrianGladman (2002) and FIPS (2001). Several operations in AES are defined at byte level, with bytes representing elements in the finite field GF (28). Other operations are defined in terms of 4-byte words. This section introduces the basic mathematical concepts needed for the AES algorithm.

2.3.1 GALLOY'S FIELD TRANSFORMATION (2^8)

The elements of a finite field can be represented in several different ways. For any prime power there is a single finite field, hence all representations of GF (2^8) are isomorphic.

Despite this equivalence, the representation has an impact on the implementation complexity. Joan Daemen and Vincent Rijmen (1999) have chosen for the classical polynomial representation.

The byte value in AES is represented as a set of bits (0 or 1) and is represented as the collection of bits separated by comma as $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$. These bytes are interpreted as finite field elements using polynomial representation as

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 \text{ ---------- 2.1}$$

Example 1:

The byte with hexadecimal value '57' (binary 01010111) corresponds with polynomial

$$x^6 + x^4 + x^2 + x + 1 \text{ ----------2.2}$$

2.3.2 FINITE FIELD ADDITION

The addition of two finite field elements is achieved by adding the coefficients for Corresponding powers of their polynomial representations, this addition being performed in GF (28), that is, modulo 2, so that 1 + 1 = 0. Consequently, addition and subtraction are both equivalent to an exclusive-and (**XOR**) operation on the bytes that represent field elements.

Addition operations for finite field elements will be denoted by the symbol "+".

Polynomial: $(x^6 + x^4 + x + 1) + (x^7 + x^6 + x^3 + x) = x^7 + x^4 + x^3 + 1$

Binary: {01010011} + {11001010} = {10011001}

Hexadecimal: {53} + {CA} = {99}

2.3.3 FINITE FIELD MULTIPLICATION

Multiplication in a finite field is multiplication modulo an irreducible reducing polynomial used to define the finite field. (I.e., it is multiplication followed by division using the reducing polynomial as the divisor—the remainder is the product.) The symbol "•" may be used to denote multiplication in a finite field.

Rijndael's finite field

Rijndael uses a characteristic 2 finite field with 256 elements, which can also be called the Galois field **GF**(2^8). It employs the following reducing polynomial for multiplication:

$x^8 + x^4 + x^3 + x + 1$.

For example, {53} • {CA} = {01} in Rijndael's field because

$(x^6 + x^4 + x + 1).(x^7 + x^6 + x^3 + x) =$

$(x^{13} + x^{12} + x^9 + \mathbf{x^7}) + (x^{11} + x^{10} + \mathbf{x^7} + x^5) + (x^8 + \mathbf{x^7} + x^4 + x^2) + (\mathbf{x^7} + x^6 + x^3 + x) =$

$x^{13} + x^{12} + x^9 + x^{11} + x^{10} + x^5 + x^8 + x^4 + x^2 + x^6 + x^3 + x =$

$x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + x^2 + x$

and

$x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + x^2 + x$ modulo $x^8 + x^4 + x^3 + x^1 + 1 =$

(11111101111110 mod 100011011) = {3F7E mod 11B} = {01} = 1 (decimal), which can be demonstrated through long division (shown using binary notation, since it lends itself well to the task. Notice that exclusive OR is applied in the example and not arithmetic subtraction, as one might use in grade-school long division.

```
    11111101111110 (mod) 100011011
   ^100011011
    1110000011110
   ^100011011
     110110101110
   ^100011011
      10101110110
   ^100011011
       0100011010
      ^100011011
        00000001
```

(The elements {53} and {CA} are multiplicative inverses of one another since their product is 1.)

2.3.4 MULTIPLICATIVE INVERSE

In mathematics, multiplicative inverse of a number a, is the number which, when Multiplied by x, yields 1 or (a · x) = 1.

Example 4:

The multiplicative inverse of 3 modulo 11 is 4 because 4 is the solution to $(3 \cdot x) \bmod 11 = 1$. In hexadecimal notation, {03} mod {0B} = 1.

In calculating multiplicative inverse for a set of 8 bits numbers, there would be a set of 256 different byte values. Multiplicative inverse is used later in SubByte and InvSubByte transformation.

2.4 THE STATE

Internally, the AES algorithm's operations are performed on a two-dimensional array of bytes called the **State**. The State consists of four rows of bytes, each containing ***Nb*** bytes, where

Nb is the block length divided by 32. In the State array denoted by the symbol *s*, each individual byte has two indices, with its row number *r* in the range $0 \varepsilon r < 4$ and its column number *c* in the range $0 \varepsilon c <$ ***Nb***. This allows an individual byte of the State to be referred to as either Sr,c or s[*r*,*c*]. For this standard, ***Nb***=4, i.e., $0 \varepsilon c < 4$

At the start of the Cipher and Inverse Cipher the input - the array of bytes *in*0, *in*1, … *in*15 is Copied into the State array as illustrated in Fig. 3. The Cipher or Inverse Cipher operations are then conducted on this State array, after which its final value is copied to the output the array of bytes *out*0, *out*1, … *out*15.

Hence, at the beginning of the Cipher or Inverse Cipher, the input array, *in*, is copied to the State array according to the scheme:

$s[r, c] = in[r + 4c]$ for $0 \varepsilon r < 4$ and $0 \varepsilon c < Nb$, ----------2.8

At the end of the Cipher and Inverse Cipher, the State is copied to the output array *out* as follows:

$out[r + 4c] = s[r, c]$ for $0 \varepsilon r < 4$ and $0 \varepsilon c < Nb$. ---------- 2.9

2.4.1 THE STATE AS AN ARRAY OF COLUMNS

The four bytes in each column of the State array form 32-bit **words**, where the row Number *r* provides an index for the four bytes within each word. The state can hence be

interpreted as a one-dimensional array of 32 bit words (columns), *w*0...*w*3, where the column number *c* provides an index into this array. Hence, for the example in Fig. 2.1, the State can be considered as an array of four words, as follows:

w0 = s0,0 s1,0 s2,0 s3,0 w2 = s0,2 s1,2 s2,2 s3,2

w1 = s0,1 s1,1 s2,1 s3,1 w3 = s0,3 s1,3 s2,3 s3,3. ----------2.10

2.5 STANDARD AES ALGORITHM SPECIFICATIONS

- For the AES algorithm**,** the length of the input block, the output block and the State is 128 bits. This is represented by Nb = 4, which reflects the number of 32-bit words (number of columns) in the State.
- For the AES algorithm**,** the length of the Cipher Key, *K*, is 128, 192, or 256 bits**.** The key length is represented by Nk = 4, 6, or 8, which reflects the number of 32-bit words (number of columns) in the Cipher Key.
- For the AES algorithm, the number of rounds to be performed during the execution of the algorithm is dependent on the key size. The number of rounds is represented by Nr, where Nr =10 when Nk = 4, Nr = 12 when Nk = 6, and Nr = 14 when Nk = 8.

DESCRIPTION OF THE AES ALGORITHM

1) **Key Expansions**—round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.
2) **Initial Round:**

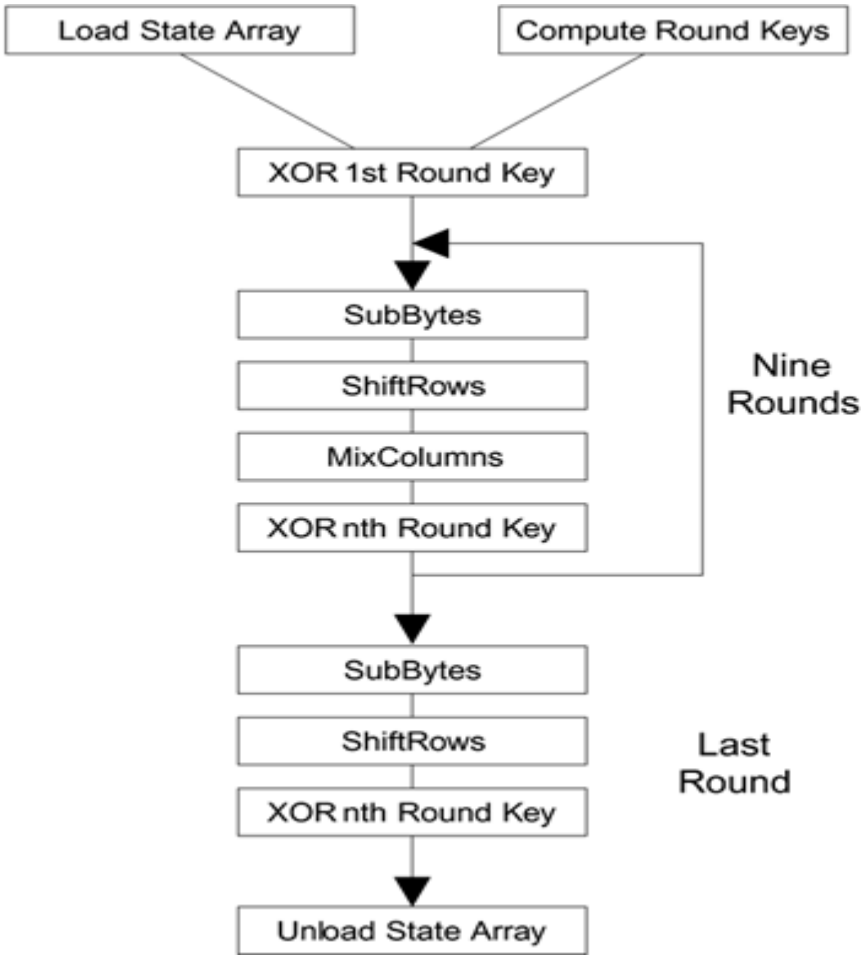
AddRoundKey—each byte of the state is combined with a block of the round key using bitwise xor.

3) **Rounds**

    a) SubBytes
    b) ShiftRows
    c) MixColumns
    d) AddRoundKey

4) **Final Round (no MixColumns)**

a) SubBytes
b) ShiftRows
c) AddRoundKey



AES is an iterated block cipher with a fixed block size of 128 and a variable key length.

The different transformations operate on the intermediate results, called *state*. The state is a rectangular array of bytes and since the block size is 128 bits, which is 16 bytes, the rectangular array is of dimensions 4x4. (In the Rijndael version with variable block size, the row size is fixed to four and the number of columns varies. The number of columns is Advanced Encryption

Standard (AES) the block size divided by 32 and denoted Nb). The cipher key is similarly pictured as a rectangular array with four rows. The number of columns of the cipher key, denoted Nk, is equal to the key length divided by 32.

A state:	A key:
| a0,0 | a0,1 | a0,2 | a0,3 |	| k0,0 | k0,1 | k0,2 | k0,3 |
| a1,0 | a1,1 | a1,2 | a1,3 |	| k1,0 | k1,1 | k1,2 | k1,3 |
| a2,0 | a2,1 | a2,2 | a2,3 |	| k2,0 | k2,1 | k2,2 | k2,3 |
| a3,0 | a3,1 | a3,2 | a3,3 |	| k3,0 | k3,1 | k3,2 | k3,3 |

It is very *important* to know that the cipher input bytes are mapped onto the state bytes in the order a0,0, a1,0, a2,0, a3,0, a0,1, a1,1, a2,1, a3,1 ... and the bytes of the cipher0 key are mapped onto the array in the order k0,0, k1,0, k2,0, k3,0, k0,1, k1,1, k2,1, k3,1 ... At the end of the cipher operation, the cipher output is extracted from the state by taking the state bytes in the same order. AES uses a variable number of rounds, which are fixed: A key of size 128 has 10 rounds. A key of size 192 has 12 rounds. A key of size 256 has 14 rounds.

PRE ROUND OPERATION

In this operation, a given data input (128 bits) is bitwise XORed with User defined Key(128 bits) to generate a cipher text of 128bits.

Example:

Input = 32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34
Cipher Key = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c
Output = 19 3d e3 be a0 f4 e2 2b 9a c6 8d 2a e9 f8 48 08

a0,0	a0,1	a0,2	a0,3		k0,0	k0,1	k0,2	k0,3		b0,0	b0,1	b0,2	b0,3
a1,0	a1,1	a1,2	a1,3	**XOR**	k2,0	k2,1	k2,2	k2,3	=	b2,0	b2,1	b2,2	b2,3
a2,0	a2,1	a2,2	a2,3		k1,0	k1,1	k1,2	k1,3		b1,0	b1,1	b1,2	b1,3
a3,0	a3,1	a3,2	a3,3		k3,0	k3,1	k3,2	k3,3		b3,0	b3,1	b3,2	b3,3

Where: b (i,j) = a(i,j) XOR k(i,j)

32	88	31	e0		2b	28	ab	09			
43	5a	31	37	**XOR**	7e	ae	f7	cf			
f6	30	98	07		15	d2	15	4f			
a8	8d	a2	34		16	a6	88	3c			

Note : b (i,j) = a(i,j) XOR k(i,j)

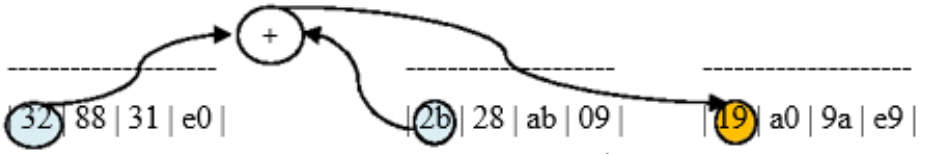
b(0,0)= a(0,0) XOR k(0,0)

32 XOR 2b

0 0 1 1 0 0 1 0=32

0 0 1 0 1 0 1 1=2b

0 0 0 1 1 0 0 1=19



32	88	31	e0		2b	28	ab	09		19	a0	9a	e9
43	5a	31	37	**XOR**	7e	ae	f7	cf	=	3d	f4	c6	f8
f6	30	98	07		15	d2	15	4f		e3	e2	8d	48
a8	8d	a2	34		16	a6	88	3c		be	2b	2a	08

THE SUB-BYTES STEP:

SubBytes operation is a non-linear byte substitution, operating on each byte of the state independently. The **substitution table (S-Box)** is invertible and is constructed by the composition of two transformations:

1. Take the multiplicative inverse in**Rijndael's finite field**

2. Apply an affine transformation as described below

$$b'_i = b_i + b_{(i+4) \bmod 8} + b_{(i+5) \bmod 8} + b_{(i+6) \bmod 8} + b_{(i+7) \bmod 8} + c_i$$

for 0 <i< 8 , where bi is the ith bit of the byte, and c*i* is the ith bit of a byte *c* with the value {63} or {01100011}. Here and elsewhere, a prime on a variable indicates that the variable is to be updated with the value on the right. In matrix form, the affine transformation element of the S- box can be expressed as

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
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

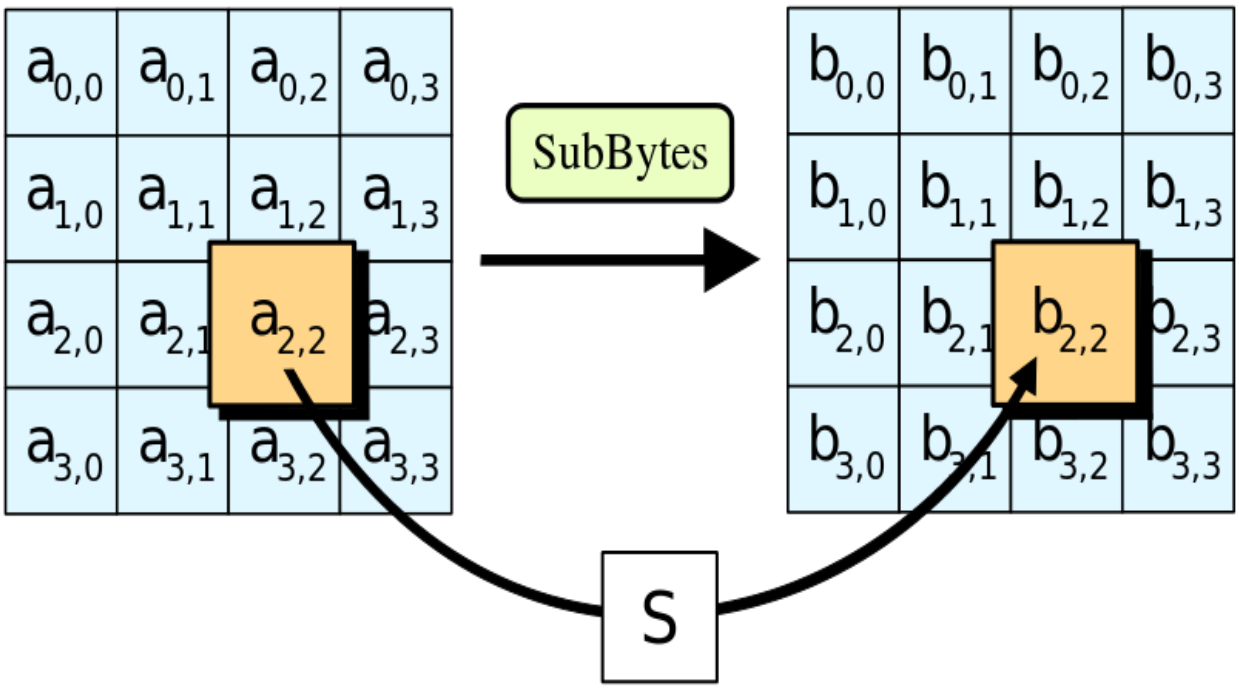
2. Then apply the affine transformation over GF(2). Binary "00000000" = Hex {00} will be transformed into "01100011"={63}.

$$\begin{bmatrix} b7 \\ b6 \\ b5 \\ b4 \\ b3 \\ b2 \\ b1 \\ b0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} a7 \\ a6 \\ a5 \\ a4 \\ a3 \\ a2 \\ a1 \\ a0 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

If $(a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$={CA}, then the computation over GF(2) is performed as follows.

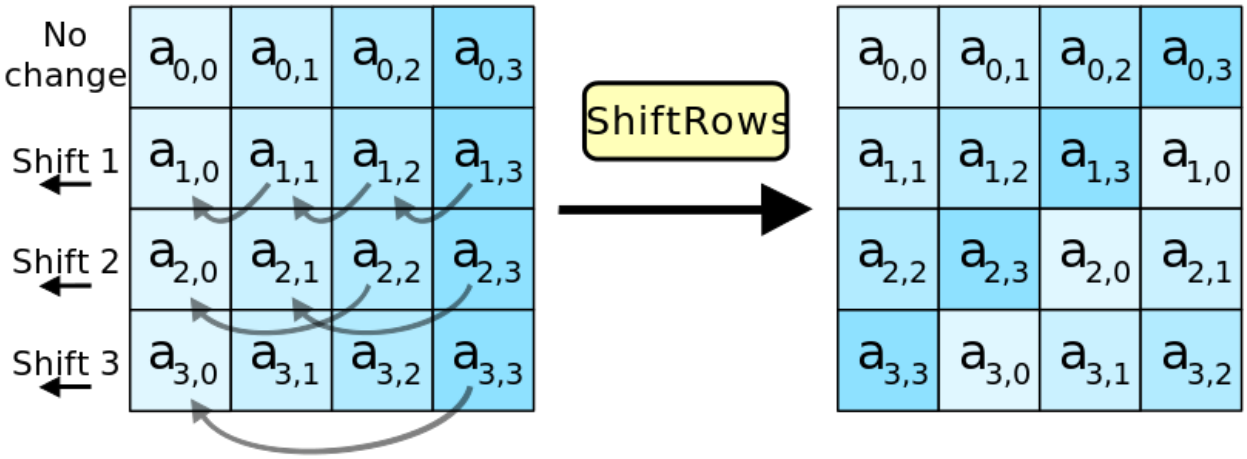
$$\begin{bmatrix} b7 \\ b6 \\ b5 \\ b4 \\ b3 \\ b2 \\ b1 \\ b0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \oplus \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \cdot 1 \oplus 1 \cdot 1 \oplus 1 \cdot 0 \oplus 1 \cdot 0 \oplus 1 \cdot 1 \oplus 0 \cdot 0 \oplus 0 \cdot 1 \oplus 0 \cdot 0 \oplus 0 \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \\ \cdots \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

{CA} will be transformed into "11101101 ={ED}.



SHIFT ROW OPERATION

In this operation, each row of the state is cyclically shifted to the left, depending on the row index. The 1st row is shifted 0 positions to the left. The 2nd row is shifted 1 position to the left. The 3rd row is shifted 2 positions to the left. The 4th row is shifted 3 positions to the left.



Example:

SubByte Output is given as an input to *ShiftRow*/Operation.

SubByte o/p

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

Shift Row
No rotate
rotate left 1
rotate left 2
rotate left 3

Shift Row Output

d4	e0	b8	1e
bf	b4	41	27
5d	52	11	98
30	ae	f1	e5

MIX COLUM OPERATION

The **Mix Columns** transformation operates on the State column-by-column, treating each column as a four-term polynomial as described in Sec.2.2.5. The columns are considered as polynomials over GF (2^8) and multiplied modulo $x4 + 1$ with a fixed polynomial $a(x)$, given by

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\} \text{ ----------2.12}$$

The above equation can be described in the matrix form as below

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < Nb$$

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

This can also be seen as the following:

$$\begin{aligned} b_0 &= 2a_0 + 3a_1 + 1a_2 + 1a_3 \\ b_1 &= 1a_0 + 2a_1 + 3a_2 + 1a_3 \\ b_2 &= 1a_0 + 1a_1 + 2a_2 + 3a_3 \\ b_3 &= 3a_0 + 1a_1 + 1a_2 + 2a_3 \end{aligned}$$

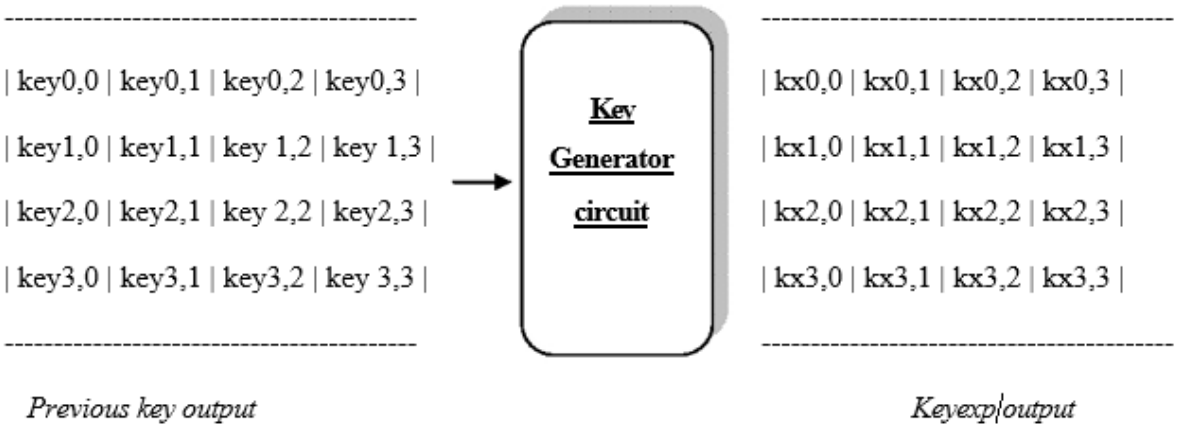
Where "+'' xor operation.

Example:

```
      State                MixColumn Matrix          Mixed
┌             ┐        ┌             ┐        ┌             ┐
| d4 e0 b8 1e |        | 02 03 01 01 |        | 04 e0 48 28 |
| bf b4 41 27 |        | 01 02 03 01 |        | 66 cb f8 06 |
| 5d 52 11 98 |        | 01 01 02 03 |        | 81 19 d3 26 |
| 30 ae f1 e5 |        | 03 01 01 02 |        | e5 9a 7a 4c |
└             ┘        └             ┘        └             ┘
```

KEY GENERATOR OPERATION

The key generator circuit functions to generate unique key for every round operation in AES algorithm. Key expander (or generator) operation basically follows five steps to generate a unique key for each round. *User defined* is fed as an input to Key expander circuit to find the key generated output.



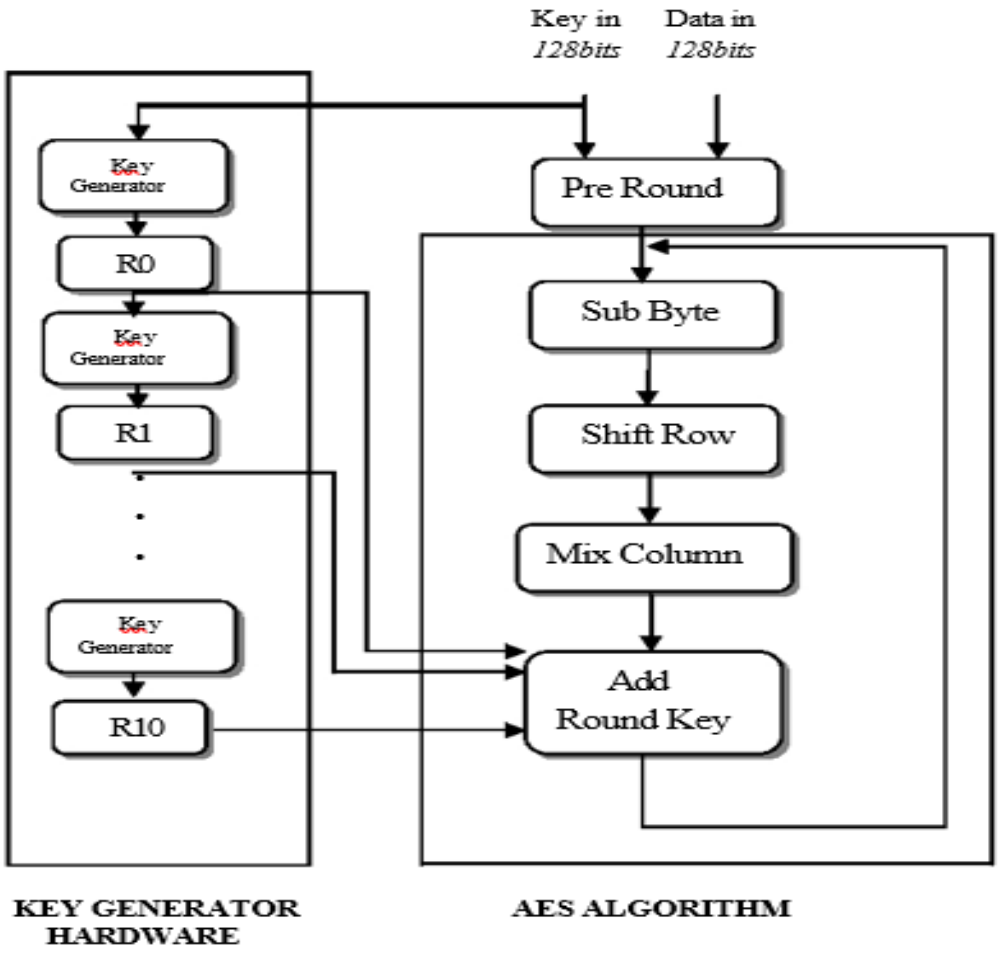


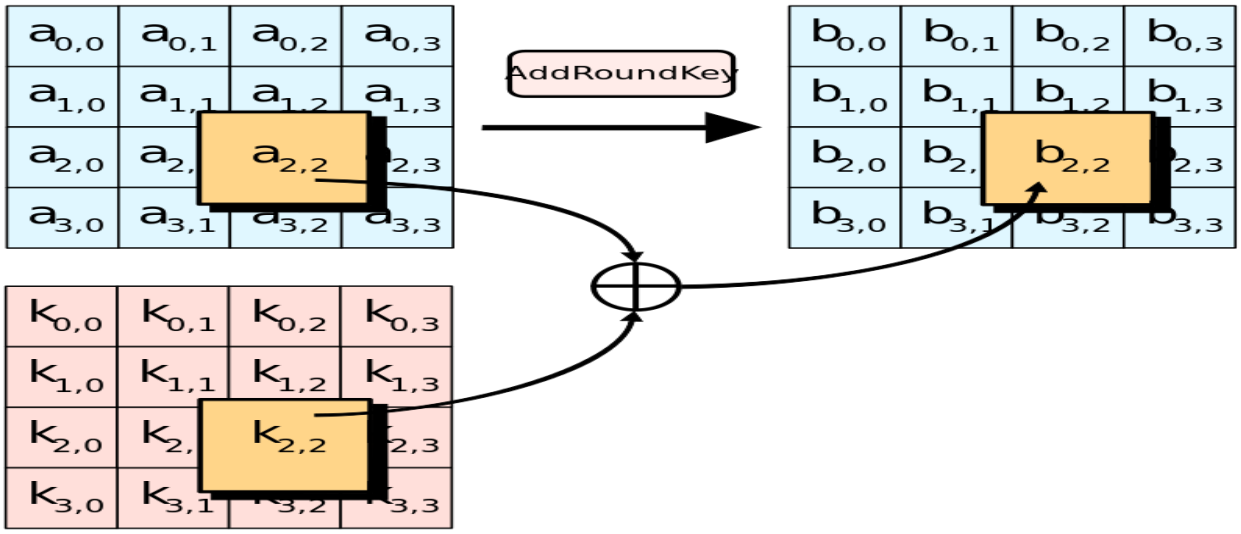
Figure 2.9 Key Generator Structure

Note: RCON values or Round constant values are derived from G.F Transformation. A unique RCON value is predefined by deriving from G.F Transform for each round operation.

Round	RCON value
R0	01
R1	02
R2	04
R3	08
R4	10
R5	20
R6	40
R7	80
R8	1B
R9	36

ADD ROUND KEY OPERATION

The primary function of Add Round Key Operation is to associate *keyexpander* output generated by key generator Circuit to the AES algorithm. In this operation, a Round Key is applied to the state by a simple bitwise XOR. The Round Key is derived from the Cipher Key by the means of the key schedule. The Round Key length is equal to the block key length (=16 bytes).



Add round key Output is given by XORing of *Keyexp* output and *Mixcolumn* output. The above output is the encrypted output of round 1. The Add round key output is again feedback to the Sub Byte transformation through feedback loop for 2nd round of operation end the same process is repeated until it completes 10 rounds of operation.

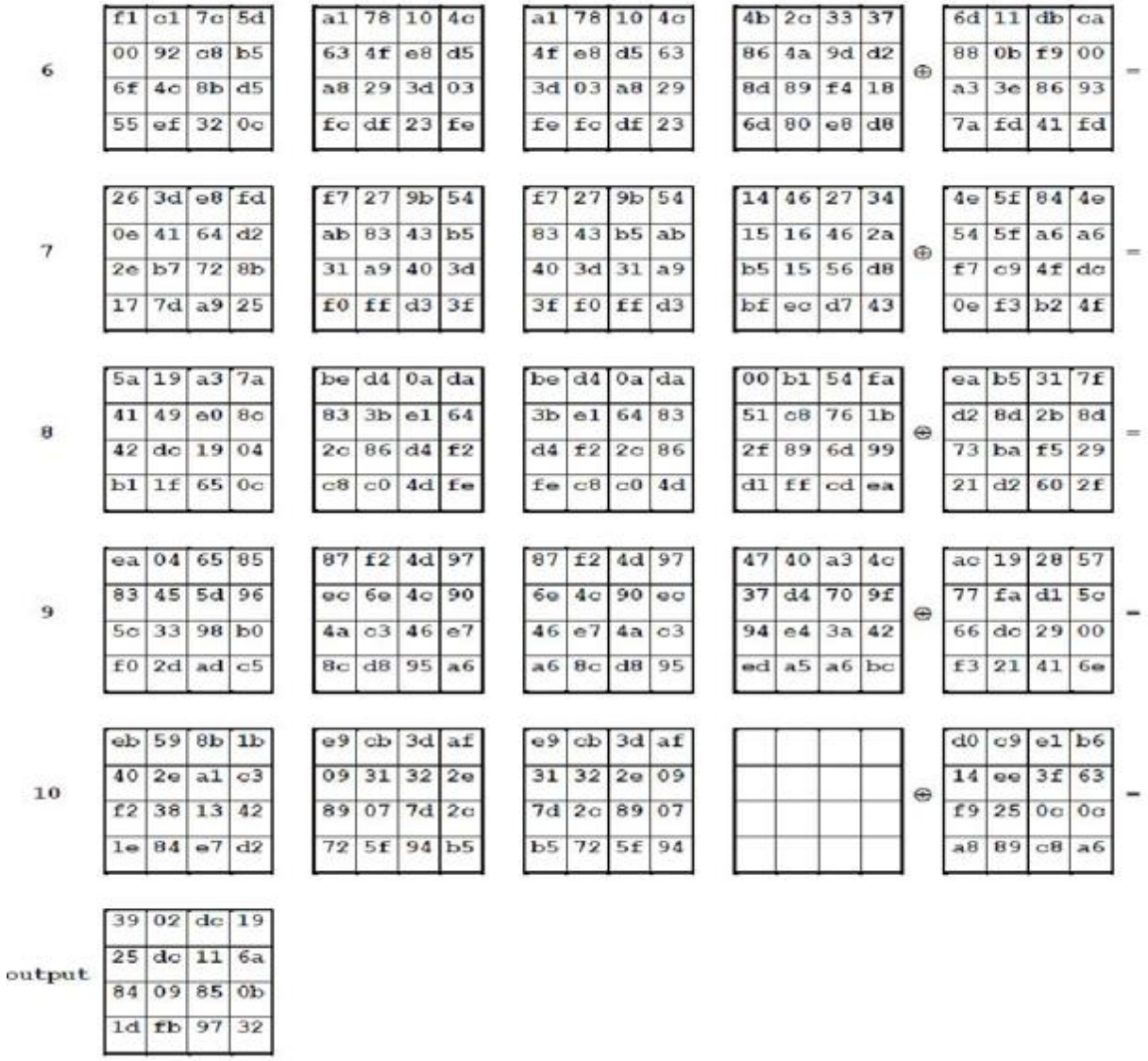
TABULAR VERIFICATION OF AES ALGORITHM

The following diagram shows the values in the State array as the Cipher progresses for a bloc length and a Cipher Key length of 16 bytes each (i.e., $Nb = 4$ and $Nk = 4$).

Input = 32 43 f6 a8 88 5a 30 8d 31 31 98 a2 e0 37 07 34

Cipher Key = 2b 7e 15 16 28 ae d2 a6 ab f7 15 88 09 cf 4f 3c

Round Number	Start of Round	After SubBytes	After ShiftRows	After MixColumns	Round Key Value	
input	32 88 31 e0 43 5a 31 37 f6 30 98 07 a8 8d a2 34				2b 28 ab 09 7e ae f7 cf 15 d2 15 4f 16 a6 88 3c	⊕ =
1	19 a0 9a e9 3d f4 c6 f8 e3 e2 8d 48 be 2b 2a 08	d4 e0 b8 1e 27 bf b4 41 11 98 5d 52 ae f1 e5 30	d4 e0 b8 1e bf b4 41 27 5d 52 11 98 30 ae f1 e5	04 e0 48 28 66 cb f8 06 81 19 d3 26 e5 9a 7a 4c	a0 88 23 2a fa 54 a3 6c fe 2c 39 76 17 b1 39 05	⊕ =
2	a4 68 6b 02 9c 9f 5b 6a 7f 35 ea 50 f2 2b 43 49	49 45 7f 77 de db 39 02 d2 96 87 53 89 f1 1a 3b	49 45 7f 77 db 39 02 de 87 53 d2 96 3b 89 f1 1a	58 1b db 1b 4d 4b e7 6b ca 5a ca b0 f1 ac a8 e5	f2 7a 59 73 c2 96 35 59 95 b9 80 f6 f2 43 7a 7f	⊕ =
3	aa 61 82 68 8f dd d2 32 5f e3 4a 46 03 ef d2 9a	ac ef 13 45 73 c1 b5 23 cf 11 d6 5a 7b df b5 b8	ac ef 13 45 c1 b5 23 73 d6 5a cf 11 b8 7b df b5	75 20 53 bb ec 0b c0 25 09 63 cf d0 93 33 7c dc	3d 47 1e 6d 80 16 23 7a 47 fe 7e 88 7d 3e 44 3b	⊕ =
4	48 67 4d d6 6c 1d e3 5f 4e 9d b1 58 ee 0d 38 e7	52 85 e3 f6 50 a4 11 cf 2f 5e c8 6a 28 d7 07 94	52 85 e3 f6 a4 11 cf 50 c8 6a 2f 5e 94 28 d7 07	0f 60 6f 5e d6 31 c0 b3 da 38 10 13 a9 bf 6b 01	ef a8 b6 db 44 52 71 0b a5 5b 25 ad 41 7f 3b 00	⊕ =
5	e0 c8 d9 85 92 63 b1 b8 7f 63 35 be e8 c0 50 01	e1 e8 35 97 4f fb c8 6c d2 fb 96 ae 9b ba 53 7c	e1 e8 35 97 fb c8 6c 4f 96 ae d2 fb 7c 9b ba 53	25 bd b6 4c d1 11 3a 4c a9 d1 33 c0 ad 68 8e b0	d4 7c ca 11 d1 83 f2 f9 c6 9d b8 15 f8 87 bc bc	⊕ =


6
f1 c1 7c 5d
00 92 c8 b5
6f 4c 8b d5
55 ef 32 0c
a1 78 10 4c
63 4f e8 d5
a8 29 3d 03
fc df 23 fe
a1 78 10 4c
4f e8 d5 63
3d 03 a8 29
fe fc df 23
4b 2c 33 37
86 4a 9d d2
8d 89 f4 18
6d 80 e8 d8
⊕
6d 11 db ca
88 0b f9 00
a3 3e 86 93
7a fd 41 fd
=
7
26 3d e8 fd
0e 41 64 d2
2e b7 72 8b
17 7d a9 25
f7 27 9b 54
ab 83 43 b5
31 a9 40 3d
f0 ff d3 3f
f7 27 9b 54
83 43 b5 ab
40 3d 31 a9
3f f0 ff d3
14 46 27 34
15 16 46 2a
b5 15 56 d8
bf ec d7 43
⊕
4e 5f 84 4e
54 5f a6 a6
f7 c9 4f dc
0e f3 b2 4f
=
8
5a 19 a3 7a
41 49 e0 8c
42 dc 19 04
b1 1f 65 0c
be d4 0a da
83 3b e1 64
2c 86 d4 f2
c8 c0 4d fe
be d4 0a da
3b e1 64 83
d4 f2 2c 86
fe c8 c0 4d
00 b1 54 fa
51 c8 76 1b
2f 89 6d 99
d1 ff cd ea
⊕
ea b5 31 7f
d2 8d 2b 8d
73 ba f5 29
21 d2 60 2f
=
9
ea 04 65 85
83 45 5d 96
5c 33 98 b0
f0 2d ad c5
87 f2 4d 97
ec 6e 4c 90
4a c3 46 e7
8c d8 95 a6
87 f2 4d 97
6e 4c 90 ec
46 e7 4a c3
a6 8c d8 95
47 40 a3 4c
37 d4 70 9f
94 e4 3a 42
ed a5 a6 bc
⊕
ac 19 28 57
77 fa d1 5c
66 dc 29 00
f3 21 41 6e
=
10
eb 59 8b 1b
40 2e a1 c3
f2 38 13 42
1e 84 e7 d2
e9 cb 3d af
09 31 32 2e
89 07 7d 2c
72 5f 94 b5
e9 cb 3d af
31 32 2e 09
7d 2c 89 07
b5 72 5f 94
⊕
d0 c9 e1 b6
14 ee 3f 63
f9 25 0c 0c
a8 89 c8 a6
=
output
39 02 dc 19
25 dc 11 6a
84 09 85 0b
1d fb 97 32

SUMMARY OF ADVANCED ENCRYPTION STANDARD ALGORITHM

Salient Features of AES:

- **Security**
  1. Actual security: compared to other submitted algorithms (at the same key and block size).

  2. Randomness: the extent to which the algorithm output is indistinguishable from a random permutation on the input block.

  3. Soundness: of the mathematical basis for the algorithm's security.

  4. Other security factors: raised by the public during the evaluation process, including any attacks which demonstrate that the actual security of the algorithm is less than the strength claimed by the submitter.

- **Cost**
  1. Licensing requirements: NIST intends that when the AES is issued, the algorithm(s) specified in the AES shall be available on a worldwide, non-exclusive, royalty-free basis.

  2. Computational efficiency: The evaluation of computational efficiency will be applicable to both hardware and software implementations. Round 1 analysis by NIST will focus primarily on software implementations and specifically on one key-block size combination (128-128); more attention will be paid to hardware implementations and other supported key-block size combinations during Round 2 analysis. Computational efficiency essentially refers to the speed of the algorithm. Public comments on each algorithm's efficiency (particularly for various platforms and applications) will also be taken into consideration by NIST.

  3. Memory requirements: The memory required to implement a candidate algorithm for both hardware and software implementations of the algorithm will also be considered during the evaluation process. Round 1 analysis by NIST will focus primarily on software implementations; more attention will be paid to hardware implementations during Round 2. Memory requirements will include such factors as gate counts for hardware implementations, and code size and RAM requirements for software implementations.

- **Algorithm and implementation characteristics**
  1. Flexibility: Candidate algorithms with greater flexibility will meet the needs of more users than less flexible ones, and therefore, inter alia, are preferable. However, some extremes of functionality are of little practical application (e.g., extremely short key lengths); for those cases, preference will not be given. Some examples of flexibility may include (but are not limited to) the following:

2. The algorithm can accommodate additional key- and block-sizes (e.g., 64-bit block sizes, key sizes other than those specified in the Minimum Acceptability Requirements section, [e.g., keys between 128 and 256 that are multiples of 32 bits, etc.])

3. The algorithm can be implemented securely and efficiently in a wide variety of platforms and applications (e.g., 8-bit processors, ATM networks, voice & satellite communications, HDTV, B-ISDN, etc.).

4. The algorithm can be implemented as a stream cipher, message authentication code (MAC) generator, pseudorandom number generator, hashing algorithm, etc.

5. Hardware and software suitability: A candidate algorithm shall not be restrictive in the sense that it can only be implemented in hardware. If one can also implement the algorithm efficiently in firmware, then this will be an advantage in the area of flexibility.

6. Simplicity: A candidate algorithm shall be judged according to relative simplicity of design.

KEY FACTS ABOUT AES ALGORITHM

The AES specified three key sizes: 128, 192 and 256 bits. In decimal terms, this means that there are approximately:

3.4×10^{38} possible 128-bit keys;

6.2×10^{57} possible 192-bit keys; and

1.1×10^{77} possible 256-bit keys.

In comparison, DES keys are 56 bits long, which means there are approximately

7.2×10^{16} possible DES keys. Thus, **there are on the order of 10^{21} times more AES 128-bit keys than DES 56-bit keys.**

In the late 1990s, specialized "DES Cracker" machines were built that could recover a DES key after a few hours. In other words, by trying possible key values, the hardware could determine which key was used to encrypt a message.

Assuming that one could build a machine that could recover a DES key in a *second* (i.e., try 2^{55} keys per second), then it would take that machine **approximately 149 thousand-billion (149 trillion) years to crack a 128-bit AES key.** To put that into perspective, the universe is believed to be less than 20 billion years old.

No one can be sure how long the AES - or any other cryptographic algorithm - will remain secure. However, NIST's Data Encryption Standard (DES) was a U.S. Government standard for approximately twenty years before it became practical to mount a key exhaustion attack with specialized hardware. The AES supports significantly larger key sizes than what DES supports. Barring any attacks against AES that are faster than key exhaustion, then even with future advances in technology, AES has the potential to remain secure well beyond twenty years.

APPLICATIONS OF AES ALGORITHM IN VARIOUS FIELD

Secure Communication

- ATM
- DVD C
- Secure Networks
- Secure video surveillance systems
- IEEE 802.11i (Wi-Fi), IEEE 802.15.3, IEEE 802.15.4 (Zigbee), MBOA (WiMedia), 802.16e, Wibree.
- Secure Storage
- Defence application
- Confidential Corporate Documents
- Government Documents
- FBI Files
- Personal Storage Devices

ATM:

- One common ATM security vulnerability involves so-called phantom withdrawals, in which cash is taken from a cardholder's account, but neither the customer nor the bank admits liability. Phantom withdrawals are sometimes the result of fraud on the part of the customer, but ATMs can also be tricked into accepting bogus, skimmed or cloned cards. ATMs generate a coded message, known as an Authorization Request Cryptogram, which card issuers use to authenticate the card and card data.
- ATMs originally used a mathematical formula, or algorithm, known as the Data Encryption Standard, to encrypt personal identification numbers. DES encrypts data

in 64-bit blocks using a 56-bit encryption key and was, at one time, an official Federal Information Processing Standard in the United States. However, increases in computing power for personal computers have rendered DES insecure for ATM applications; ATMs using DES have been breached within 24 hours.

Secure Networks:

Encryption is where data is rendered hard to read by an unauthorised party. Since encryption can be made extremely hard to break, many communication methods either use deliberately weaker encryption than possible, or have backdoors inserted to permit rapid decryption. In some cases, government authorities have required backdoors be installed in secret. Many methods of encryption are also subject to "man in the middle" attack whereby a third party who can 'see' the establishment of the secure communication is made privy to the encryption method, this would apply for example to interception of computer use at an ISP.

FUTURE CHALLENGES: WHAT LIES BEYOND AES…?

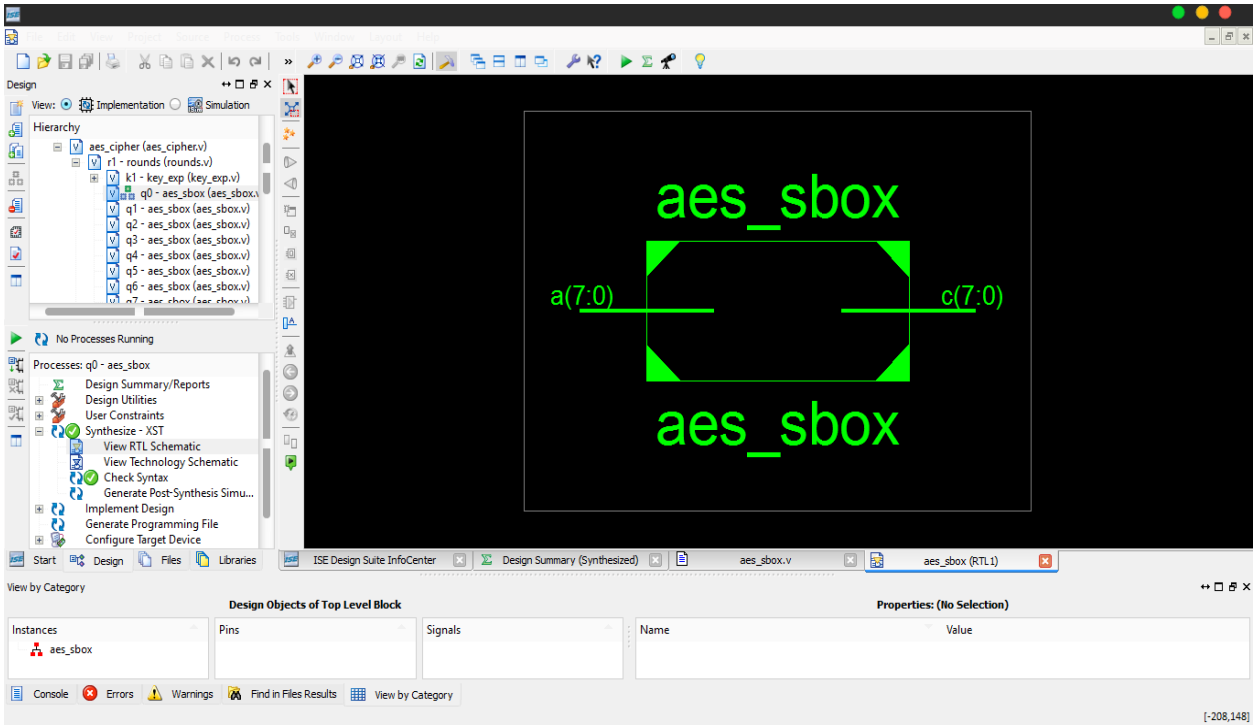
NIST is in the process of initiating a number of other cryptographic activities, including a standard specifying modes of operation for symmetric key block ciphers (e.g., AES), an HMAC standard, a key management standard, a new and enlarged hash function that is consistent with the AES key sizes, and an increase in key sizes for the Digital Signature Algorithm (DSA).

CONCLUSION:

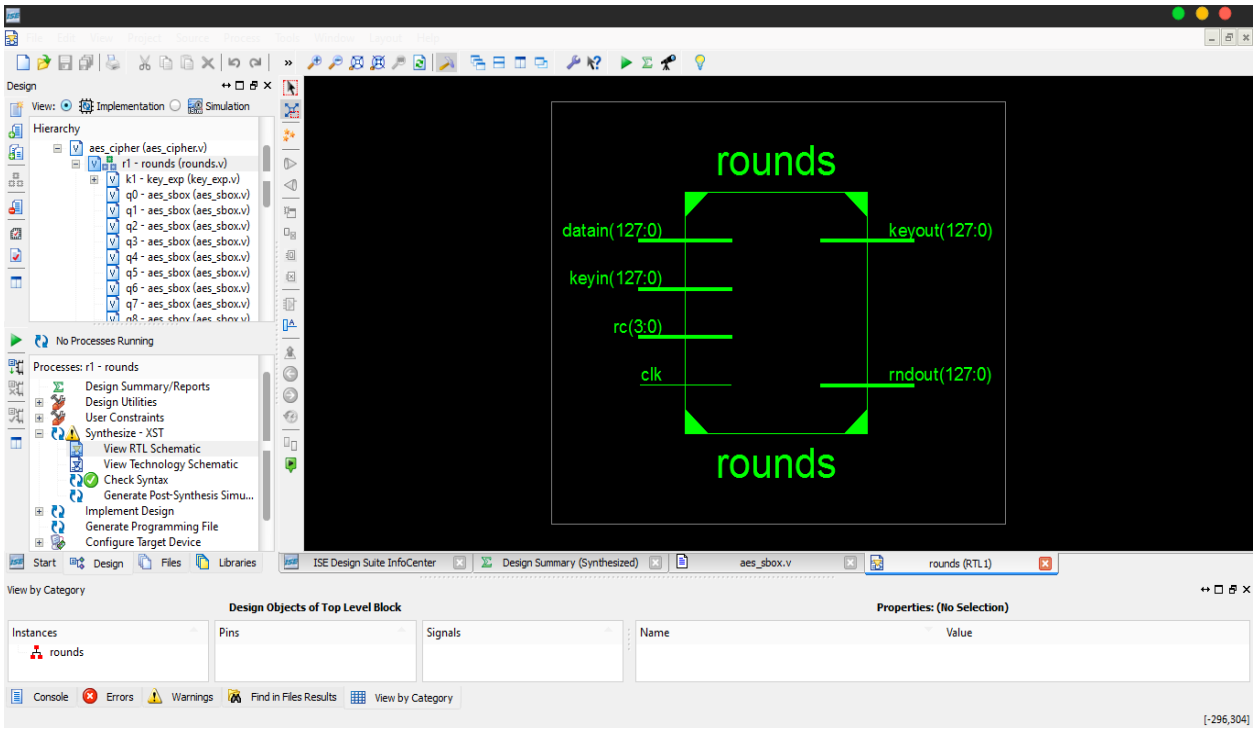
Optimized and Synthesizable VERILOG code is developed for the implementation of both encryption and decryption process. Each program is tested with some of the sample vectors provided by NIST and output results are perfect with minimal delay. Therefore, AES can indeed be implemented with reasonable efficiency on an FPGA, with the encryption and decryption taking an average of 320 and 340 ns respectively (for every 128 bits). The time varies from chip to chip and the calculated delay time can only be regarded as approximate. Adding data pipelines and some parallel combinational logic in the key scheduler and round calculator can further optimize this design.

SIMULATION RESULT:

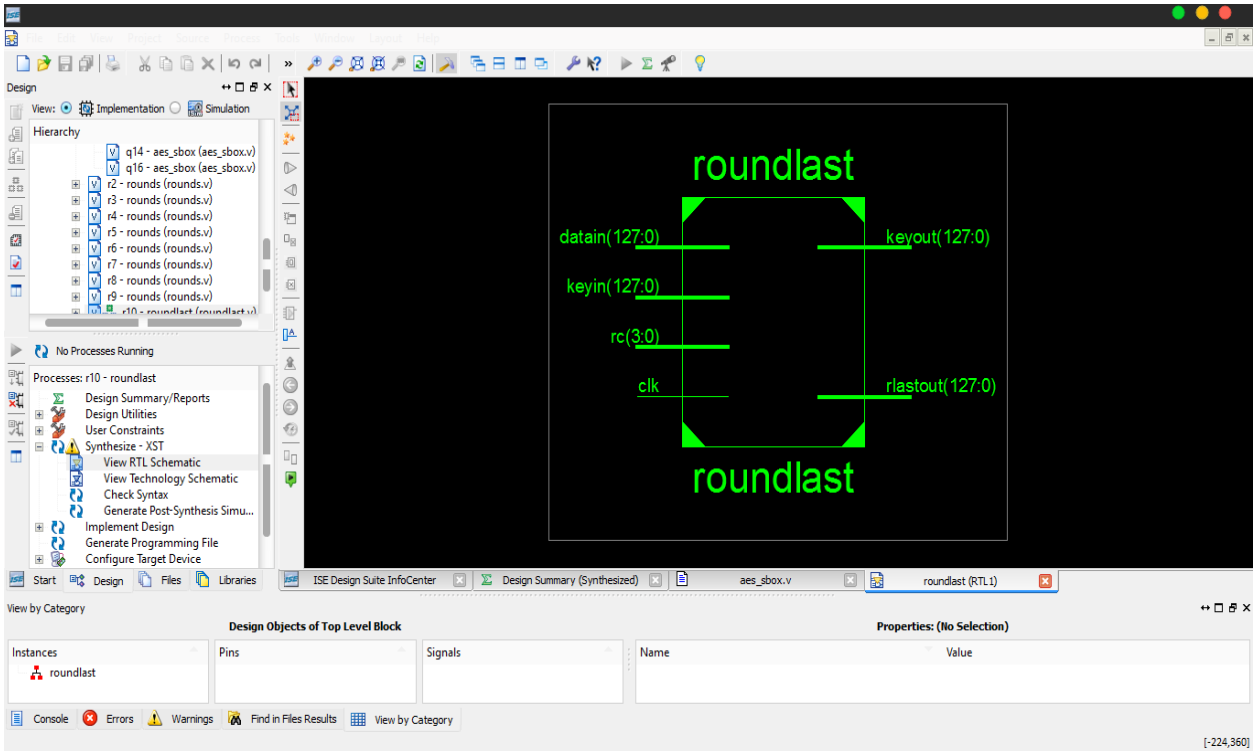
1) S-BOX



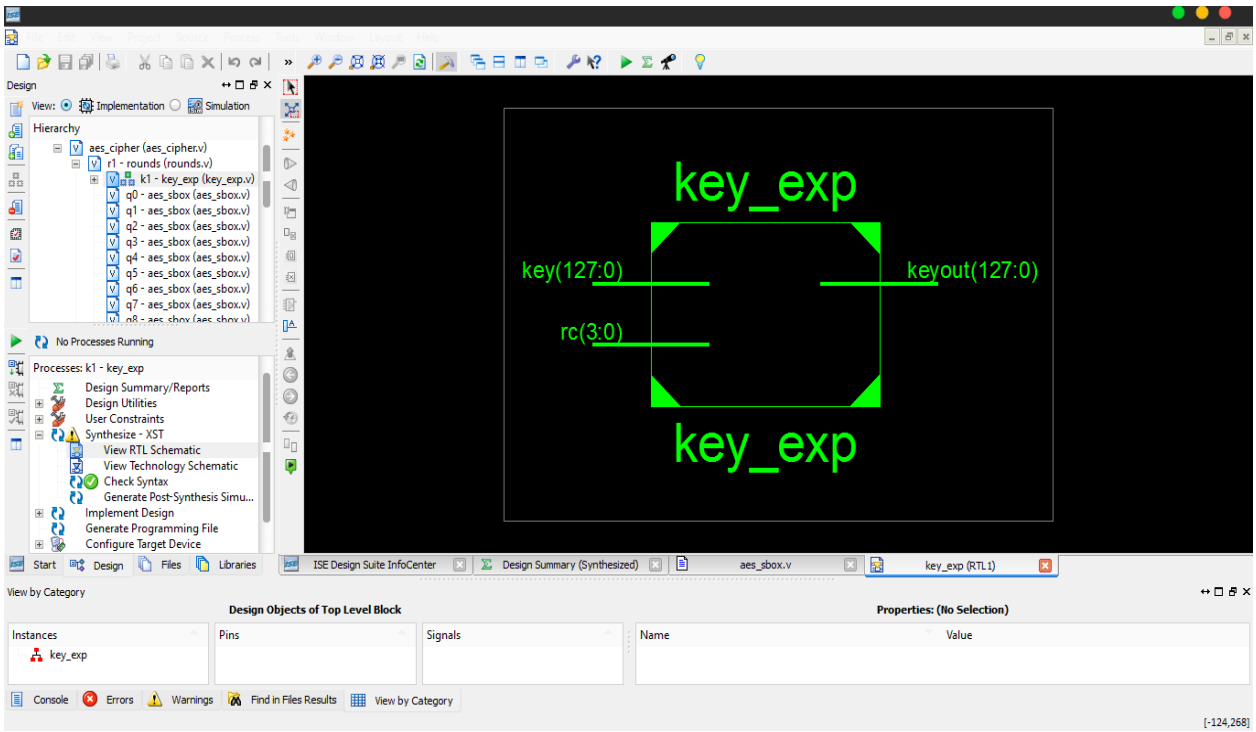
2) ROUNDS OPERATION



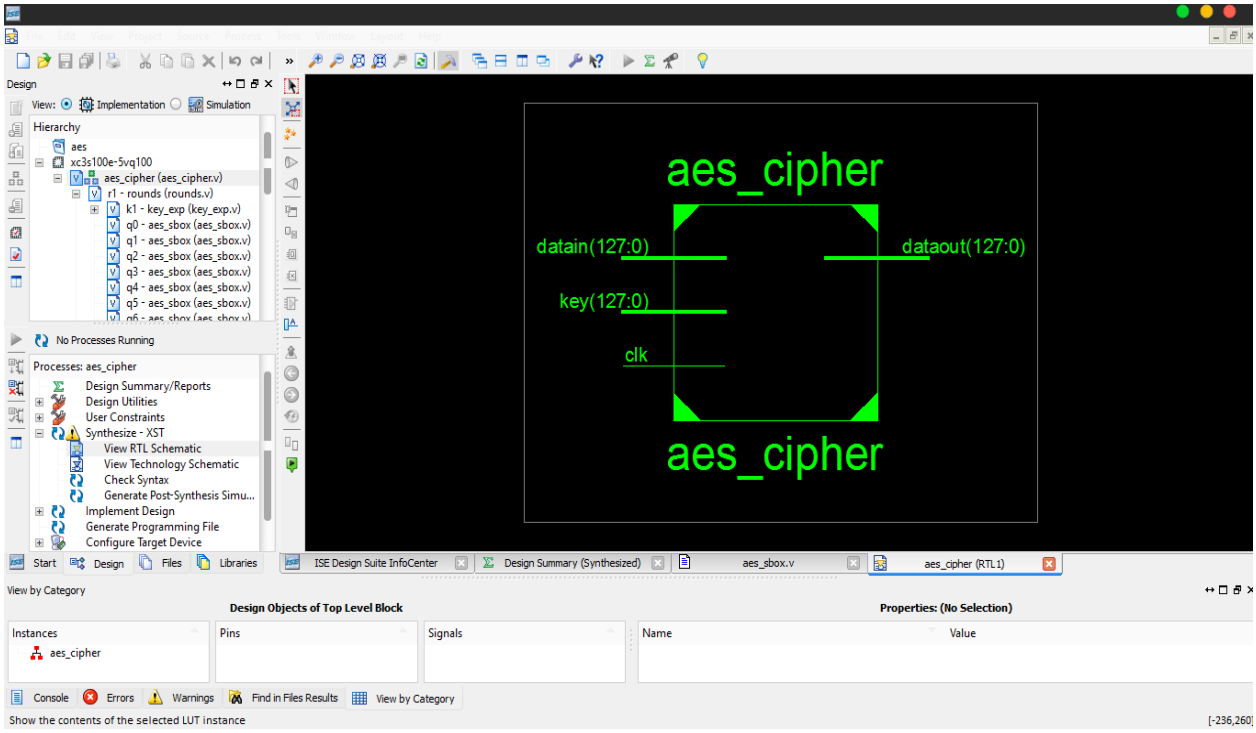
3) LAST ROUND OPERATION



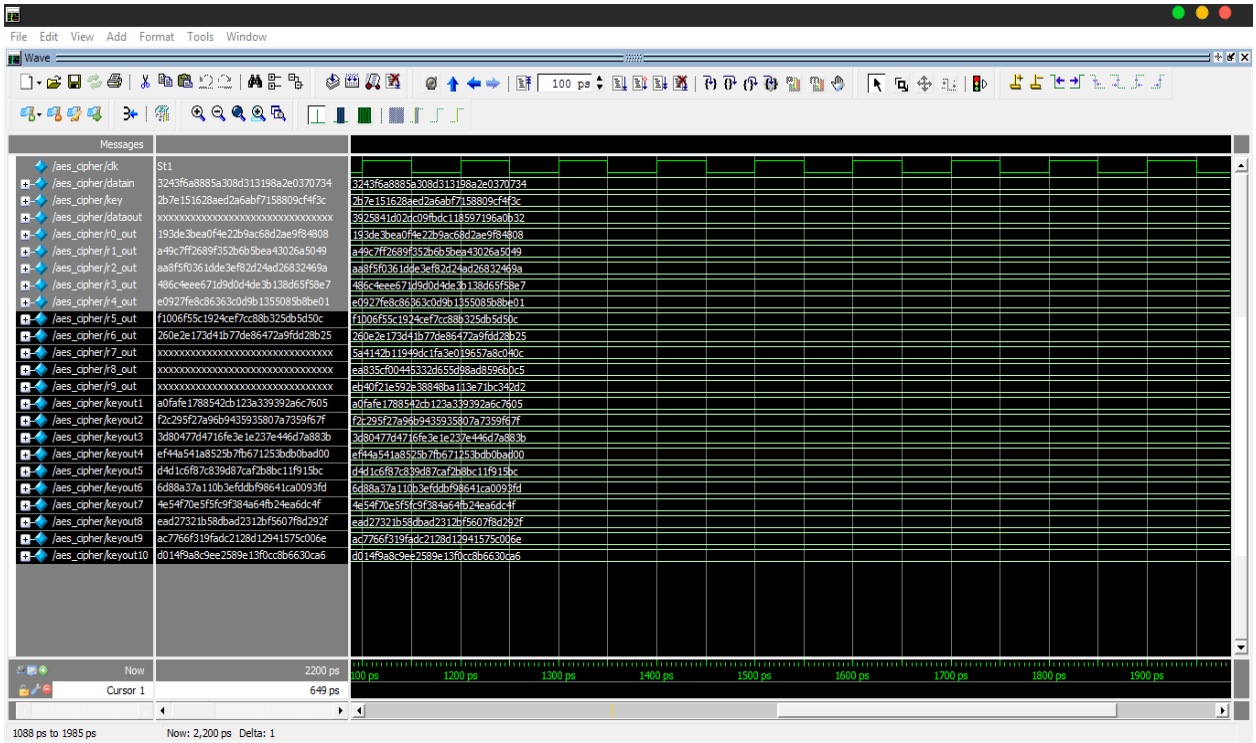
4) KEY GENERATOR



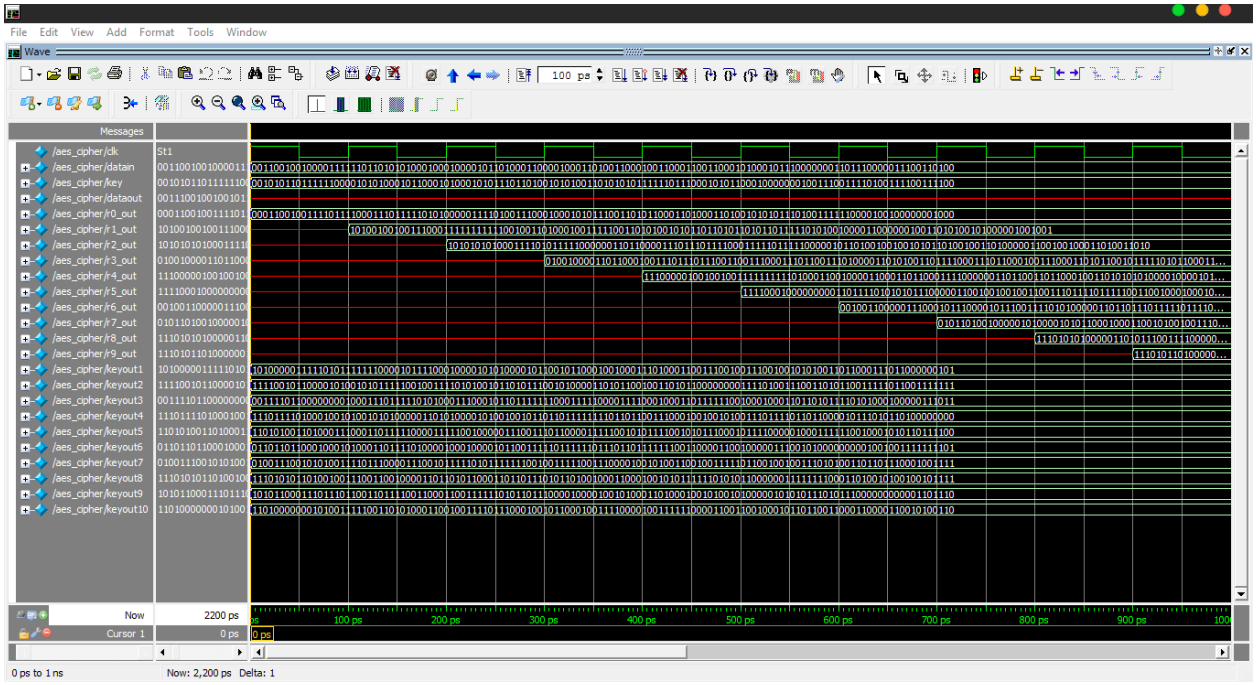
5) AES CIPHER BLOCK (TOP MODULE)



6) ENCRYPTED DATA IN BINARY FORM



7) ENCRYPTED DATA IN HEXADECIMAL FORM



PROJECT DONE BY:

P.N. VAMSHI (IIT KHARAGPUR)
K. MUKESH NAIDU (NIT DURGAPUR)
B. DEVA KUMAR (NIT DURGAPUR)
J. HARSHAVARDHAN REDDY (NIT ROURKELA)