CHAPTER 1

CRYPTOGRAPHY

1.1 INTRODUCTION

Cryptography - the art of writing in secret code is the science of communication and information secured and protected from the authority of the unauthorized. Cryptography enables communication of information over an insecure channel with proper security. Modern cryptography doesn’t confine to secure communication alone, but provides applications for software security, security of electronic devices, copyright protection (Digital Rights Management (DRM)) and more.

The word “cryptography” is derived from the Greek word “kryptos” which means “hidden”. Cryptography is an essential feature for secure communications by converting ordinary information into meaningless text and provide authentication and it is effectively synonymous with encryption, where an unreadable state of information is converted to a readable state as shown in figure 1.1.

![Crypto Process for Secure Data](image-url)

**Figure 1.1 Crypto Process for Secure Data[2]**
Cryptographic algorithms have computational security by enabling improvements in integer factorization and faster computing technology which has been introduced. ‘Cipher’, a pair of algorithm which creates encryption and the decryption a reversal of encryption and the performance of the ‘Cipher’ is controlled both by the algorithm used and also by a Crypto Key which is secret, a string of characters small in length, useful to decrypt the cipher text. A crypto system is an ordered list of elements of possibly finite plaintexts, possible finite cipher texts, finite possible keys and the encryption and decryption algorithms that correspond to each possible key.

1.2 ENCRYPTION STANDARDS

In general, cryptography was largely concerned with information confidentiality, comprehensible form of messages converted into an incomprehensible one and back to the other end, making it unreadable by interceptors without knowledge of the secrecy. Encryption was used to ensure Secrecy in communication, such as Spies, military leaders, and diplomats. But in recent, the confidentiality concern has grown beyond, which includes message integrity techniques (ie): identity of sender/receiver authentication, digital signatures, and proofs that are interactive and secure computation among others.

The classical cipher types are transposition ciphers, rearranging the letters in a message and substituting cipher that replaces letters or a group of letters with other letters or group of letters systematically. Modern cryptography concentrates on four objectives:- they are, Confidentiality (the information can be understood by none to whom it wasn’t intended), Integrity (the information cannot be altered in space between sender and the intended receiver without detection of the alteration), Non-repudiation (the sender of the information cannot oppose their intentions in the transmission or creation of the information), Authentication (the sender and receiver can clarify with each other identity and the origin/destination of the information).
• **Data Confidentiality**: Information is made available to only authorized individuals or processes. During the process of communication, confidentiality is necessary to assure that only the parties involved in communication can view data exchanges.

• **Data integrity**: The property of data being unaltered or not destroyed in an unauthorized manner. The need for data integrity is clear if the transmitted data moves across a non-secure network like Internet.

• **Non-repudiation**: Non-repudiation is protection against repudiation and which is of two types:

  • “**Non-repudiation with proof of origin**” provides the recipient of data with proof that proves the origin of the data and thus protects the recipient against an attempt by the originator to falsely deny sending the data. Its role is to prove that a particular transaction took place by establishing accountability of information about a particular event or action to its originating entity.

  • “**Non-repudiation with proof of receipt**” provides the originator of data with evidence proving that data was received as addressed and securely and thus protects the originator against an attempt by the recipient to refusal of receiving the data.

• **Authentication**: This process is to prove the identity of an entity that can be based on something known as a password, an encryption key or encryption card, biometric measurements that include *retinal scans* or *voice recognition*. 
1.2.1 Public keys and Private keys

The Public and the Private keys pair comprises of two unique cryptographic keys. The Public Key is made available to everyone through a publicly accessible directory. On the other hand, the Private Key remains confidential to its owner. The data encrypted with Public Key may only be decrypted by its corresponding Private Key and vice versa. The Public key cryptography can be updated with better confidentiality. However another important aspect of Public Key Cryptography is its ability of creation of a Digital Signature.

Many enterprise applications require the use of both Public and Private Key Encryptions. When exchange of a message, whether encrypted or not, we need to verify its reliability. Someone, particularly in public networks, may alter the message. Data-integrity verification includes authenticating the beginning of the message. Once we accept that the message is from an authenticated user and was not modified after creation, want to consider whether the sender can deny sending the message by claiming that someone has stolen the cryptographic key has used it to authenticate the message. Therefore, non-repudiation is an important feature of cryptographic systems.

1.2.2 Methods of encryption

There are innumerable methods of encryption that can be fixed which would allow all but the most persistent attackers from successfully cracking the code. The encryption is difficult to crack only if one doesn’t know how the encryption algorithm was used. The usage of such algorithms is quite insecure, for several reasons.

If the method is exposed, not only one message but all messages encrypted using the one method is now easily interpretable. Also, the skill to
keep the information secret from enemy is compromised if more than one person is made aware of it. Such algorithms employ what is known as “security through obscurity.”

- The following are the order for a cryptography method to be reasonably secure:
  - It should be kept secret, and it should not fall into the hands of the enemy without any difficulty
  - The system must be practical, mathematical, interpretable.
  - It should be applicable to communication through telegraph
  - Its key should be communicable and retainable without the help from the written notes and changeable or modifiable at the will of the correspondents
  - It should be convenient, and its usage and function should not require the concourse of many people
  - At last, it is indispensable to give the environment that commands its application (ie) the system is easy to use, without needing mental strain or the knowledge of a long series of regulations to observe and understand.

### 1.3 USING KEYS

The key is the only thing that must have to be kept secret from an attacker to prevent unwanted decryption. A key is some data, whether it is a number or word or phrase or string of bits, which is used to both in encryption and decryption of a file. The way in which information is encrypted changes depends on what key is usually used to encrypt it.
There are two methods of encryption.

- Symmetric Cryptography and
- Asymmetric Cryptography

1.3.1 Symmetric cryptography

The secret key cryptography is also referred as symmetric encryption. Symmetric-key cryptography refers to encryption methods in which same key is shared between both the sender and receiver. When an encryption algorithm uses the same key to decryption as well as encryption, it is said to use symmetric-key cryptography. Such key is handy for encrypted storage as opposed to transfer. It is dangerous to use this type of encryption for exchange of data since a method must be devised so that both parties during communication are aware of these secret key.

Symmetric-key systems are simple, faster but their main pitfall is that the secure way of key exchange is involved. Public-key encryption avoids this plen because the public key can be exchanged in a in secure way, and the private key is never transmitted.

Symmetric key ciphers are implemented as either block ciphers or stream ciphers. A block cipher enciphers the data in plaintext blocks as opposed to individual characters, the same form of input as used by a stream cipher. Stream ciphers operate on a single bit (byte or computer word) at a time and implement some form of feedback mechanism so that there is a constant change in the key. A block cipher is named because the scheme encrypts one block of data at a time using the same key on each block. As usual, the same plain text block will encrypt to the same cipher text when using the same key always in a block cipher whereas the plaintext will encrypt to a different cipher text in a stream cipher.
The most popular symmetric-key systems are the Data Encryption Standard and Advanced Encryption Standard (AES). Despite its deprecation as an official standard, DES stays popular. It is used across a wide range of applications, starting from ATM encryption to e-mail privacy and secure remote access. Various block ciphers have been designed and released, with considerable quality variation. Most of them have been thoroughly broken as FEAL.

Stream ciphers are in contrast to the 'block' type ciphers that create an arbitrarily long stream of key material combined with the plaintext or character. In a stream - cipher, the output stream is produced depending on a hidden state internally located which change as the cipher operates. That internal state is initially built using the secret key. RC4 is a commonly used stream cipher.

Cryptographic hash functions take a message input of any length, and smaller outputs produced, fixed length hash which can be used in a digital signature. For good hash functions, two messages producing same hash should not be applicable for an attacker. Message Authentication Codes (MACs) are similar to cryptographic hash functions except a secret key can be used to authenticate the hash value upon receipt.

1.3.2 Asymmetric Cryptography

The alternative to using a symmetric-key algorithm is asymmetric keys. Using this method, different keys are used to encrypt and decrypt files. The two keys namely, a public key and a private key are used and these keys are mathematically related. The public key is the one used to encrypt data. It can be known by anyone. However, the private key, which is used to decrypt the file, is only known by only the user. These are used in public-key encryption.
To differentiate it with symmetric key and public key encryption is denoted as asymmetric key encryption. In public key encryption, the key can be passed between the parties openly without any hidden keys or published in a public repository for shared open access, but the private key remains private. Data-decryption of a public key encrypted data could be done only using a private key. A likely procedure is followed in the vice-versa as the data encrypted with the private key can be decrypted only using the public key. Anyone can use a person’s public key to encrypt information, but do not with the knowledge of that person’s private key, cannot decrypt the same.

1.3.3 One-way Functions

There exist mathematical functions that are trickier to reverse than to apply. These types of functions are known as one-way functions. The computer will take fraction of second to calculate the result of the function which is exponentially harder to discover and provides some result. A method by which information can be easily encoded but by which it is nearly impossible to decode without special information. While there is no application for the above function in cryptography, one method that is used often is called factoring. The method of finding all prime numbers that make up any given number grows exponentially harder the larger the number that is to be factored.

1.4 TYPES OF CRYPTOGRAPHIC ALGORITHM

The most widely used Cryptographic Algorithms are:

- RSA Algorithm
- DES Algorithm
Cryptographic Hash Algorithms and

One Time Pad

1.4.1 RSA algorithm

RSA is one of the first practicable public-key cryptosystems and is widely used for data transmission in a safer way. In such a cryptosystem, the encryption key is public and differs from the secret key for decryption. Public-key Cryptography, also known as Asymmetric cryptography, uses two unique statistically linked keys, one public and one private. The public key shared with everyone, but the private key is maintained secret. In RSA cryptography, the public keys and the private keys can be used to encrypt a message; the key used to encrypt a message has an alternative key which is used to decrypt it.

RSA has become the most widely used asymmetric algorithm for this attribute: RSA is widely used for securing sensitive data especially the data that are sent through insecure network like internet. In RSA, this asymmetry is based on the factoring problem, a practical difficulty of factoring the product of two large prime numbers.

The RSA algorithm was publicly described in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman at MIT; the letters RSA are the initials of their surnames.

Many protocols like SSH, Open PGP, S/MIME, and SSL/TLS rely on RSA for encryption and digital signature functions. It's also used in software programs, browsers are some of the obvious example, which requires to establish a secure connection over an insecure network lane like the Internet or validate a digital signature. Of the many operations performed the RSA signature verification is one of the most frequently performed operations in IT
Characteristics of RSA Algorithm

Any one of the two related keys can be used for Encryption, with the other used for Decryption. The process of RSA algorithm is shown in figure 1.2.

- **Plaintext:** The plaintext is the data or readable message that is fed as the input, the algorithm is also the input.

- **Encryption algorithm:** Various transformations on the plaintext is provided by Encryption algorithm.

- **Public and private keys:** A pair of keys selected so as to perform two processes, one of them is encryption and the other is decryption. The transformations involved in the algorithm executed by these keys depend on the type of the key that is provided as input.

- **Ciphertext:** This is the scrambled message produced as output. It is based on the Plaintext and the key. For a given message, two different cipher texts are produced by two different keys.

- **Decryption algorithm:** The original plaintext is produced by the algorithm accepting the cipher text and the matching key as inputs.
Figure 1.2  The Process of RSA[77]
Figure 1. 3 Key Operations in RSA Algorithm [77]

Operations of RSA Algorithm:

1. Key Generation
2. Encryption and
3. Decryption

Key Generation

RSA includes a public key and private key. The public key is visible to everyone, it is used to encrypt messages. Only the private key can be used to decrypt the message that has been encrypted using the public key. The keys for the RSA algorithm are generated in the following way as described in the figure 1.3:

Step 1: Different primes of p and q should be generated

Step 2: Modulus, \( n = p \times q \) is calculated
Step 3: Quotient $\varphi(n) = (p - 1) \times (q - 1)$ is calculated

Step 4: An integer $e$ should be selected such that $1 < e < \varphi(n)$ and $\gcd(\varphi(n), e) = 1$

Step 5: A value for $d$ which is private exponent is calculated such that $d = e^{-1} \mod \varphi(n)$

Step 6: Public Key = $[e, n]$ and

Step 7: Private Key = $[d, n]$

**Encryption:** A message to any person demanding security can be encrypted by his/her public key (that could be officially listed like phone numbers). The ciphertext generation is explained briefly in figure 1.4.

$$(\text{Cipher}) \ C = M^e \mod n$$

![Figure 1.4 Ciphertext Generation](image)

**Decryption:** The person to whom the secret message is addressed can alone decrypt it using the private key.

$$(\text{Plain text}) \ M = C^d \mod n \text{ where}$$
n = a modulus for modular arithmetic

φ(n) = the quotient of n

e = an integer that is relatively prime to φ(n)

[This guarantees that e will possess a multiplicative inverse modulo φ(n)]

d = an integer (that is the multiplicative inverse of e modulo φ(n))

The Basic Idea of RSA Algorithm

- An individual A aspiring to send messages confidentially uses a pair of integers \{e, n\} as his / her public key.

- Another party B requiring to send a message M to A in a secure manner will encrypt M using A’s public key \{e, n\} to create cipher text C. Subsequently, C can be decrypted by A alone using his/her private key \{d, n\}.

- If the plaintext message M is too long, RSA may be chosen by B as a block cipher for encrypting the message meant for A. The block size is likely to be half the number of bits required to represent the modulus n, when RSA is used as a block cipher, if the modulus required is 1024 bits, for instance, for its representation, 512 – bit blocks data representation would be used.

A Worked Example of RSA Algorithm

Two primes: \(p=11\) and \(q=13\) are selected by Aaron as his RSA keys. The modulus value for it is \(n=p\times q=143\). The quotient of n \(\phi(n)=(p-1)(q-1)=120\). Then he chooses 7 for his RSA public key e and later
using the Extended Euclidean Algorithm he calculates his RSA private key which gives his final result as 103.

Rob wants to send Aaron a message M that’s encrypted so he obtains Aaron’s RSA public key \((n, e)\) which in this example is \((143, 7)\). His plaintext message is just the number 9 and is encrypted into ciphertext \(C\) as follows:

\[
M^e \mod n = 9^7 \mod 143 = 48 = C
\]

When Aaron receives Rob’s message he decrypts it by using his RSA private key \((d, n)\) as follows:

\[
C^d \mod n = 48^{103} \mod 143 = 9 = M
\]

In order to digitally sign a message using RSA keys, Alice would create a hash or message digest of his message to Rob, encrypt the hash value with his RSA private key and include it with the message. Rob can clarify using this that sender of the message is Aaron and assure that the message hasn’t been altered during the due course by decrypting the hash value with Aaron’s public key. If this value matches with the hash of the original message, then only Aaron could have sent it (authentication and non-repudiation) and the message is exactly as he wrote it (integrity).

Security of RSA Algorithm

The security of RSA relies when factoring large integers as large complex computational difficulties arises. As there occurs increase in computing power and more efficient factoring algorithms are discovered, the ability to factorize larger and larger numbers increases too. Encryption strength is proportional to key size, and doubling key length delivers an exponential increase in strength, although it dejects performance.
RSA keys are typically 1024- or 2048-bits long, but it is widely believed that 1024-bit keys could be factorized further, which is why government and industry are migrating towards a minimum key length of 2048-bits. Barring startling changes in quantum computing, it should be a long time before longer keys are required, an alternative to RSA for implementing public-key cryptography is found in elliptic curve cryptography as it is gaining favour with many security experts.

1.4.2 DES Algorithm

The DES (Data Encryption Standard) Algorithm was devised in the 1970’s as a result of the need for a governmental standard for encrypting sensitive information. A symmetric key method is used by the DES, so that the key that is used to encrypt information is the same key used to decrypt it. In the case of DES, the key is a 64-bit number, of which 56 bits are randomly generated and encryption process uses it directly. The remaining 8 bits are confined for error detection.

It was suggested, even back when the standard was first introduced, that 56 bits may be too small for optimal security. As time went on, more and more people became skeptical of the security of the standard. In 1993 the governmental standard was updated to Triple DES, a much more secure form of DES. Finally, in 1997, the original DES was cracked, and has since been replaced by AES (the Advanced Encryption Standard). Since then, DES has been cracked much more efficiently numerous times, rendering it virtually useless.

1.5 AES ALGORITHM

The most widely used Symmetric key algorithm at current trend is advanced encryption standard (AES) which provide bulk data encryption. The
AES algorithm finds popular use in applications such as cellular phones, ATM, smart cards, digital video recorders and hardware accelerator card for e-commerce WWW server to support thousands of simultaneous independent secure connections amongst several others.

There exist many presentations of hardware implementations (ASIC and FPGA) of Rijndael AES algorithms in literature. A variety of techniques for AES implementation includes designs seen in literature, seeking maximum throughput minimum power consumption and yet some others minimize circuitry. The schematic representation of AES is shown in figure 1.5.

Among these approaches, the sub pipelined architecture can achieve maximum speedup and optimum speed-area ratio in non-feedback modes. The major optimization criteria considered are hardware reduction getting maximized and reduction of path delay. For maximizing hardware reduction, the design focuses on computation of non-linear operations.

The design of SB/ISB and MC/IMC transformations forms the key functional units in the AES algorithm. Therefore design of compact AES demands SB/ISB and MC/IMC with maximum hardware reduction. However, the SB/ISB in the AES algorithm are traditionally implemented by look up tables (LUT) of 28*8= 2K bits, to store the pre calculated values of all 256 representations of a byte in GF(2^8).

In addition, the unbreakable delay of LUTs is longer than the total delay of the rest of the transformation in each round, which in turn prevents each round unit from being divided in to multiple sub stages with equal delay to achieve speedup further.

Non-LUT based approaches, employing only combinational logic for computing multiplicative inversion for SB/ISB, using it as unbreakable delay
of LUTs can be avoided, which in turn explores the sub-pipelining further to overcome the limitation of maximum operable clock frequency and makes sure that it doesn’t require any internal memory. However, these approaches involve computations in Galois Field \( \text{GF}(2^8) \), which may involve high hardware complexity. The varieties of composite field constructions are implemented with their own benefits.

The smaller gate count and shortest critical path can be achieved by using different irreducible polynomials and the effect of their coefficients on complexities of the subfield operation. The current work improves on the implementations of the already existing SB/ISB for the optimum design of Inverter, and employs isomorphic mapping, based on subset of parameter dependent implementation options and results leading to a shorter critical path and smallest area. The resource sharing in both bit and byte level will be derived in the following sections. The architecture proposed for the implementation of Integrated MC/IMC transformation is evaluated in terms of total area and critical path delay which are measured in terms of number of gates.

**Significant Characteristics of AES:**

- AES with a block length of 128 bits is a block cipher.
- AES permits 3 unique key lengths: 128, 192, or 256 bits.
- Encryption involves 10 rounds of processing for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys.
- Excepting the last round in each case, other rounds are similar.
- The \( 4 \times 4 \) matrix of bytes is defined as the state array.
- AES also has the notion of a word. A word consists of 4 bytes;
therefore each column of the state array is a word, as is each row.

✓ Each round of processing provides an output state array by working with input state array.

✓ Unlike DES, the decryption algorithm differs substantially from the encryption algorithm.

1.5.1 Structure of AES

![Figure 1.5 AES Schematic]

AES ENCRYPTION

AES DECRYPTION

Figure 1.5 AES Schematic
1.6 AES CIPHER MODES OF OPERATION

AES can be transmitted in five different modes. Various different modes of operation are commonly used to improve the security or performance of block ciphers. Some modes prevent information leakage, by ensuring that identical inputs will create distinct outputs. Other modes also insure that any changes of the encrypted data are recognizeable. There are five operating modes. They are:

1. Electronic Code Book (ECB) Mode,

2. Cipher Block Chaining (CBC) Mode,

3. Cipher Feed Back (CFB) Mode,

4. Output Feed Back (OFB) Mode,

5. Counter Mode (CTR)

1.6.1 Electronic Code Book Mode (ECB)

The Electronic Code Book (ECB) mode is a confidentiality mode that features, for a given key, the assignment of a fixed cipher text block to each plain text block. The Electronic Code Book (ECB) mode is defined as follows:

ECB Encryption: \( C_j = \text{CIPH}_K(P_j) \) for \( j = 1 \ldots n \).

ECB Decryption: \( P_j = \text{CIPH}^{-1}_K(C_j) \) for \( j = 1 \ldots n \).
21

ENCRYPTION                        DECRYPTION

Figure 1.6 The ECB Mode

In ECB encryption, the forward cipher function is applied directly and independently to each block of the plain text. The resulting sequence of output blocks is the cipher text. In ECB decryption, to each block of the cipher text, the inverse cipher function is applied directly and independently. The resulting sequence of output blocks is the plain text. In the ECB mode, under a given key, any given plain text blocks always gets Encrypted to the same cipher text block. In a particular application, If this property is undesirable, the ECB mode should not be used.

1.6.2 Cipher Block Chaining Mode (CBC)

The Cipher Block Chaining (CBC) mode is a confidentiality mode whose encryption process features the combining ("chaining") of the plaintext blocks with the previous ciphertext blocks. The CBC mode requires an Initialization Vector(IV) to combine with the first plaintext block. The IV need not be secret, but it must be unpredictable. The integrity of the IV should be protected.
The CBC mode is defined as follows:

**CBC Encryption:** \[ C_1 = \text{CIPH}_k(P_1 \oplus \text{IV}); \]
\[ C_j = \text{CIPH}_k(P_j \oplus C_{j-1}) \quad \text{for } j = 2 \ldots n. \]

**CBC Decryption:** \[ P_1 = \text{CIPH}^{-1}_k(C_1) \oplus \text{IV}; \]
\[ P_j = \text{CIPH}^{-1}_k(C_j) \oplus C_{j-1} \quad \text{for } j = 2 \ldots n. \]

**Figure 1.7 The CBC Mode**

In CBC encryption, the first input block is formed by exclusive-ORing the first block of the plaintext with the IV. To the first input block the forward cipher function is applied, and the resulting output block is the first block of the ciphertext.
This output block is also "exORed" with the second plain text data block to produce the second input block, and the forward cipher function is given to produce the second output block. This output which is the second ciphertext block, is "exORed" with the next plaintext block to form the next input block. Each successive plaintext block is "exORed" with the previous output/ciphertext block to produce the new input block. The forward cipher function is then given to each input block to produce the ciphertext block. In CBC decryption, the inverse cipher function is then given to the first ciphertext block, and the output is obtained. The inverse cipher function is given to the second ciphertext block, and the output block is "xORed" with the first ciphertext block to recover the second plaintext block. For recovering any plain-text block (except the first), the inverse cipher function is given to the corresponding ciphertext block, and the block resulting is xORed with the previous ciphertext block.

In CBC encryption, the input block for each forward cipher operation depends on the previous forward cipher operation and the forward cipher operations could not be performed in parallel. In CBC decryption, then the input blocks for the inverse cipher function, are available immediately so that multiple inverse cipher operations can be done in parallel.

1.6.3 The Cipher Feedback Mode (CFB)

The Cipher Feedback (CFB) mode is a type of confidentiality mode that characterises the feedback of successive ciphertext segments into the input blocks of the forward cipher to generate output blocks that are xORed with the plaintext to get the ciphertext, and vice versa. The CFB mode requires an IV as the initial input block. The IV must not be secret, but it should not be predictable.
The CFB mode also requires an integer parameter, denoted \( s \), such that \( 1 \leq s \leq b \). In the specification of the CFB mode below, each plaintext segment (\( P_j \)) and ciphertext segment (\( C_j \)) consists of \( s \) bits.

The value of \( s \) is incorporated into the name of the mode. The 1bit, the 8bit, the 64bit, the 128bit CFB mode are some examples.

The CFB mode is defined as follows:

**CFB Encryption:**

\[
I_1 = IV;
\]

\[
I_j = \text{LSB}_{b-s}(I_{j-1}) \oplus C_{j-1} \quad \text{for } j = 2 \ldots n;
\]

\[
O_j = \text{CIPH}_k(I_j) \quad \text{for } j = 1, 2 \ldots n;
\]

\[
C_j = P_j \oplus \text{MSB}_s(O_j) \quad \text{for } j = 1, 2 \ldots n.
\]
CFB Decryption: \[ I_1 = IV; \]

\[ I_j = \text{LSB}_{b-s}(I_{j-1})C_{j-1} \text{ for } j = 2 \ldots n; \]

\[ O_j = \text{CIPH}_k(I_j) \text{ for } j = 1, 2 \ldots n; \]

\[ P_j = C_j \oplus \text{MSB}_s(O_j) \text{ for } j = 1, 2 \ldots n \]

In CFB encryption, the first input block is denoted as the IV, and the forward cipher operation is performed to the IV to produce the first output block. The first ciphertext segment are produced by xORing the first plaintext segment with the most significant bits of the first output block. (The remaining b-s bits of the first output block are discarded.) Concatenation of the b-s least significant bits of the IV with the s bits of the first ciphertext segment to form the second input block. Another way of description of the formation of the second input block is that the bits of the first input block circularly shift s positions to the left, and then the s least significant bits of the resultant output is replaced by ciphertext segment.

The process is then repeated with the successive input blocks until a ciphertext segment is produced from every plaintext segment. Each successive input block is enciphered to get an output block. The 's' most significant bits of each output block are usually xORed with the corresponding plaintext segment to get a ciphertext segment. Each ciphertext segment (except the last one) is then “fed back” into the previous input block, as stated above, to form a new input block.

In CFB decryption, the IV the first input block, and each successive input block is produced similar in CFB encryption, by the concatenation of the b-s least significant bits of the previous input block with the s most significant bits of the previous ciphertext. The forward cipher function is given to each input block to form the output blocks. The s most significant bits of the output blocks are then xORed with the corresponding segments of ciphertext to get the plaintext segments back.
In CFB encryption, the input block to each forward cipher function (except the first) usually depends on the result of the previous forward cipher function; therefore, forward cipher operations multiple in number cannot be done in parallel. In CFB decryption, the required forward cipher operations can be done in parallel if the input blocks are first constructed (in series) from the IV and the ciphertext.

1.6.4 The Output Feedback Mode (OFB)

The Output Feedback (OFB) mode is a confidentiality mode that characterises the iteration of the forward cipher on an IV to produce a sequence of output blocks that are xORed with the plaintext to produce the ciphertext, and vice versa. The OFB mode usually requires that the IV is a nonce, i.e., the IV should be unique for each execution of the mode under the key given.

The OFB mode is defined as follows:

**OFB Encryption:**

\[ I_1 = IV; \]
\[ I_j = O_{j-1} \quad \text{for } j = 2 \ldots n; \]
\[ O_j = \text{CIPH}_K(I_j) \quad \text{for } j = 1, 2 \ldots n; \]
\[ C_j = P_j \oplus O_j \quad \text{for } j = 1, 2 \ldots n-1; \]
\[ C^*_n = P^*_n \oplus \text{MSB}_u(O_n). \]

**OFB Decryption:**

\[ I_1 = IV; \]
\[ I_j = O_{j-1} \quad \text{for } j = 2 \ldots n; \]
\[ O_j = \text{CIPH}_K(I_j) \quad \text{for } j = 1, 2 \ldots n; \]
\[ P_j = C_j \oplus O_j \quad \text{for } j = 1, 2 \ldots n-1; \]
\[ P^*_n = C^*_n \oplus \text{MSB}_u(O_n). \]
In OFB encryption, the IV is always transformed by the forward cipher function to form the first output block. The first output block is XORed with the first plaintext block to produce the first ciphertext block. The forward cipher function is then appealed on the first output block to form the second output block. The second output block is XORed with the second plaintext block to get the block of second ciphertext, and the forward cipher function is appealed on the second output block to produce the third output block. Thus, the successive output blocks are produced from concerning the forward cipher function to the previous output blocks, and the output blocks are XORed with the corresponding plaintext blocks to produce the blocks of ciphertext. For the last block, which may be a partial block of \( u \) bits, the most significant \( u \) bits of the last output block are used for doing the XOR operation; the remaining \( b-u \) bits of the last output block are usually discarded.
1.6.5 The Counter Mode

The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. The sequence of counters must have the property that each block in the sequence is different from every other block. This condition is not restricted to a single message: across all of the messages that are encrypted under the given key, all of the counters must be distinct. In this recommendation, the counters for a given message are denoted $T_1, T_2, \ldots, T_n$.

The CTR mode is defined as follows:

CTR Encryption: \[ O_j = \text{CIPH}_k(T_j) \quad \text{for } j = 1, 2 \ldots n; \]
\[ C_j = P_j \oplus O_j \quad \text{for } j = 1, 2 \ldots n-1; \]
\[ C^*_{n} = P^*_n \oplus \text{MSB}_u(O_n). \]

CTR Decryption: \[ O_j = \text{CIPH}_k(T_j) \quad \text{for } j = 1, 2 \ldots n; \]
\[ P_j = C_j \oplus O_j \quad \text{for } j = 1, 2 \ldots n-1; \]
\[ P^*_{n} = C^*_{n} \oplus \text{MSB}_u(O_n). \]
In CTR encryption, the forward cipher function is invoked on each counter block, and the resulting output blocks are exclusive-ORed with the corresponding plaintext blocks to produce the ciphertext blocks. For the last block, which may be a partial block of \( u \) bits, the most significant \( u \) bits of the last output block are used for the exclusive-OR operation; the remaining \( b - u \) bits of the last output block are discarded. In CTR decryption, the forward cipher function is invoked on each counter block, and the resulting output blocks are exclusive-ORed with the corresponding ciphertext blocks to recover the plaintext blocks.

For the last block, which may be a partial block of \( u \) bits, the most significant \( u \) bits of the last output block are used for the exclusive-OR operation; the remaining \( b - u \) bits of the last output block are discarded. In both CTR encryption and CTR decryption, the forward cipher functions can
be performed in parallel; similarly, the plaintext block that corresponds to any particular ciphertext block can be recovered independently from the other plaintext blocks if the corresponding counter block can be determined. Moreover, the forward cipher functions can be applied to the counters prior to the availability of the plaintext or ciphertext data.

1.7 SCOPE OF THE WORK

The available channel bandwidth and severe necessities of real-time data processing makes cryptography an indispensable process for many digital data communication applications and always requires more robustness in data security. Therefore, fast and accurate encryption standard is highly desirable to assure much reduced power analysis attack, while maintaining good decrypted information quality.

1.8 OBJECTIVE OF THE WORK

An attempt has been made in the present work to enhance the robustness of Advanced Encryption Standard by employing different schemes.

A pipelined algorithm is proposed to achieve a high data throughput when compared with other methods. A rapid, robust and area efficient composite field implementation of the byte substitution phase is designed using an optimum number of pipeline stages is called fully pipelined structure against DPA for the prevention of AES implemented in FPGA.

The following are the possible countermeasures for the Differential Power Analysis Attack in AES to set objectives to realize the goal.

- Avoid register
- Make cipher impure [pseudo corrupt]
• Known Power variation
• Increased complexity
• Masked password method [data splitting and data masking]
• Combinational circuit and
• Asynchronous / Randomized process or steps

1.9 ORGANIZATION OF THE THESIS

Chapter 1 provides an overview of cryptography, particularly about Advanced Encryption Standard, its advantages and about Side Channel Attacks. The motivation, scope and prime objectives of the present work and organization of the thesis are presented in this chapter.

Extensive literature survey related to the Differential Power Analysis attacks and its preventive counter measures in AES has been significantly reviewed and presented in chapter 2
Summary of review of literature is also furnished.

Chapter 3 discusses about problems identified in AES with side channel attacks (SCA) and possible solution to overcome such attacks. This chapter describes the types of SCAs and its prevention by polynomial based masked S-box technique in AES.

Chapter 4 presents discussion about differential power analysis attack and correlation power analysis attack and simulation results of captured power traces and shows that the hacker is unable to disclose the secret key even after 17,320 power traces. This chapter also discusses about another DPA counter measure technique.

Chapter 5 discusses about the mix-column transform in AES and shows introducing pipelining in mix-column and inverse mix-column, improvement in speed of AES circuit is achieved.

Chapter 6 deals with how memory complexity is dramatically reduced using the Content-Addressable Memory (CAM) compared to the SRAM based S-box and Inverse S–box look-up tables and also discusses about the new hardware sharing architecture is applied to implement the proposed high-speed secured encryption algorithm.

Chapter 7 concludes the thesis by emphasizing the major inference of the study. A summary of research contribution and the scope for future studies are also incorporated in this chapter.