CHAPTER 1

INTRODUCTION

1.1 CRYPTOGRAPHY

Cryptography is an art and science of message conceal related to network security. The objective of cryptography can be explained by assuming a scenario, suppose that someone wants to send a message either by letter or by an electronic mail to a receiver and the sender needs the message to be highly confidential. Due to the reason that, there is a possibility of someone else can open the letter or read the electronic mail. The only clarification to this problem is Cryptography. Encryption and Decryption are the two major separations which plays a major role in cryptography with the help of mathematical expressions. In a cryptographic terminology, the original, unconcealed message is called plain text and the coded message is called cipher text. The process of converting plaintext to cipher text is known as encryption or enciphering and the process of converting cipher text to plaintext is known as decryption or deciphering. The study of encryption and decryption is known as cryptography.

In cryptography, as applied in the commercial world, is concerned with a number of problems, the most important of these are,

Confidentiality: The process of nondisclosure of information except to another authorized person.
**Authentication**: The process of providing evidence of identity of the sender to recipient, so that the recipient can be assured that the person sending the data is the one he or she claims to be.

**Integrity**: The simple description of integrity says that the message must be as it is sent by the sender without any amendment (Blake 2005).

**Non-repudiation**: The method to ensure that information cannot be disclaimed. Once the non-repudiation process is in place, the sender cannot deny being the originator of the information (Janagan 2012).

### 1.1.1 Cryptographic Algorithms

Both encryption and decryption process make use of secret keys and cryptographic algorithms. There are two broad categories of key based algorithms, namely symmetric or secret key algorithm and asymmetric or public key algorithm (RSA Laboratories 2002). The difference between these two algorithms lies in the number of keys used. Symmetric algorithm as the name says, uses the same key for both encryption and decryption process, whereas public key algorithm uses two different keys namely public key and private key for encryption and decryption process. Public key is the common key and private key is unique for both sender and receiver. Public key is small when compared to the size of private key.

#### 1.1.1.1 Symmetric key cryptography

The symmetric key algorithm can be classified as stream cipher and block cipher. Stream cipher encrypts a single bit of plaintext at a time, whereas block cipher takes a number of bits and encrypts them as a single unit. One of the indispensable symmetric key algorithms is Data Encryption Standard (DES). Due to its increase in the computing power, the basic version
of DES cannot be considered sufficiently safe anymore. Alternate and powerful cipher algorithm is Advanced Encryption Standard (AES) and it was standardized in 2001 (Seleborg 2007).

The major disadvantage of symmetric key algorithm is that both the sender and the receiver have to agree on a common key and a secure channel is required between them in order to exchange the key. Here the major problem arises at the time of key distribution. Since key distribution happens through Internet, which is extremely vulnerable to attacks so that there are lot of security problems have been involved in secret key algorithm and these problems can be overcome by using public key cryptography.

1.1.1.2 Public key cryptography

Public key cryptography uses a key pair instead of one secret key. The key used for encryption is the public key and is freely distributable, for instance it can be placed on one of the public key repositories on the Internet. For decryption, private key is used, which is always kept secret by the key holder. This private key is not transferred to anyone and is stored securely by the holder.

The two most important first generation public key algorithms used to secure the Internet are Rivest-Shamir-Adleman (RSA) and Diffie-Hellman (DH) (Anoop 2005). The security of the former is based on the difficulty of factoring the product of two large prime numbers. The latter is related to a problem known as discrete logarithm problem for finite fields. The majority of public key systems in use exercise 1024 bit parameters for RSA and Diffie-Hellman.

The public key cryptosystem algorithms can be categorized into three different groups namely,
1. **Algorithm based on Discrete Logarithm Problem (DLP)**

   A method to solve a given instance of DLP is dependent on the size of the parameters, and each time, the parameter size increases the difficulty of solving the problem (Hankerson 2004).

2. **Algorithm based on Integer Factorization Problem (IFP)**

   Integer Factorization Problem is in which its hardness is important for the security of the RSA public key encryption. The problem of hardness results from the difficulty of finding the prime factorization of a given positive integer $n$ (Koblitz 1987).

3. **Algorithm based on Elliptic Curve Discrete Logarithm Problem (ECDLP)**

   The challenging part of this problem is to find the positive integer $K$, from a given two points $P$ and $Q$ on an elliptic curve over a finite field, such that $Q = K \cdot P$ (Menezes 1993).

   The most computationally intensive operation for DL and IFP are based on scalar exponentiation. These operations are performed using very long operand to meet the required key size. The operands in ECDLP algorithms are smaller than Discrete Logarithm (DL) systems.

1.1.2 **RSA Algorithm**

   RSA algorithm belongs to block cipher technique in which the plain text and cipher text are integers between 0 to $n-1$ for some $n$. A typical size for $n$ is 1024 bits. The attacks against the RSA cryptosystems are based on solving the Integer Factorization Problem (IFP). The RSA algorithm is the
best known algorithm for integer factorization family of cryptosystems where
the strength of the cryptosystem lies in the mathematical difficulty of
factoring large integers. In this scheme integers of large size are chosen so as
to make brute force attack (trying with all possible keys) factorization
infeasible (Caelli 1999).

Factoring is the act of splitting an integer into a set of smaller
factors which, when multiplied mutually form the original integer. For
example, the factors of 21 are 3 and 7, the factoring problem is to find 3 and 7
when 21 is given. Prime factorization requires splitting an integer into factors
that are prime numbers and each integer has a unique prime factorization.

Multiplying two prime integers together is easy, but as far as
known, factoring the product of two (or more) prime numbers is much
difficult (Menezes 1993). As mentioned before, the RSA cryptosystem is
based on Integer Factorization Problem. To setup RSA cryptosystem, a user
picks two large primes $p$ and $q$, and computes their product $n = pq$. It is
well known that the order of $G$ (Group) is \( \phi(n) = (p - 1)(q - 1) \), where
\( \phi \) denotes the Euler phi function. Clearly, the user can compute the group
order \( \phi (n) \). The user’s public key is the pair of integers \((n, b)\) and the private
key is $a$ (Menezes 1993).

RSA is an efficient algorithm in crypto field, but the key length of
RSA increases the load on the application. This problem can be overcome
through Elliptic Curve Cryptography (ECC). Therefore, Elliptic Curve
Cryptography is the only efficient public key algorithm to compete with RSA.
1.1.3 Elliptic Curve Cryptography

ECC is a newly emerged public key cryptographic algorithm. It is younger than RSA and other cryptographic algorithms. Although, as Hankerson, Menezes and Vanstone (2004) state, Elliptic curve had been focused by researchers and algebraists in the middle of nineteenth century. In 1985, the first elliptic curve cryptographic application has been discovered by Neal Koblitz and Victor Miller (Koblitz A 2011). Their work was based on the algorithm for factoring integers relying on properties of elliptic curves published one year earlier, in 1984, by Hendrik Lenstra. Many organizations like Certicom (Certicom 2009) and Sun Microsystems invested time and money in the research on ECC (Gura 2009).

The main advantage of ECC over RSA is its key size. For instance, it is widely accepted that 160 bit ECC offers equal security as 1024 bit RSA. This significant difference makes ECC especially attractive for applications on constrained environments as shorter key sizes are translated to less power, minimum storage requirements and reduced computing time.

The attacks against elliptic curve cryptosystem are based on solving Elliptic Curve Discrete Logarithm Problems (ECDLP). The existing methods currently used is for attacking the discrete logarithm problem which depends on having a finite abelian group $G$. Hence these methods can also be applied to the analogue of Discrete Logarithm Problems (DLP) (Stallings 2003). These methods are generally much slower because of the additional complexity of elliptic curve point addition and point doubling operations.

The comparison between two most popular public key cryptographic algorithms namely ECC (Elliptic Curve Cryptography) and RSA (Rivest-Shamir-Adleman) is described in Table 1.1.
Table 1.1 Comparisons between RSA and ECC

<table>
<thead>
<tr>
<th>RSA Algorithm</th>
<th>ECC Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key Generation</td>
<td>Key Generation</td>
</tr>
<tr>
<td>Select (p, q)</td>
<td>(Eq(a, b), G)</td>
</tr>
<tr>
<td>Calculate (n = pq)</td>
<td>User A Key Generation</td>
</tr>
<tr>
<td>Calculate (\phi(n) = (p - 1)(q - 1))</td>
<td>Select (n_A n_A &lt; n)</td>
</tr>
<tr>
<td>Calculate (d \equiv e^{-1}(mod \phi(n)))</td>
<td>Calculate (P_A = n_A \times G)</td>
</tr>
<tr>
<td>Public key: (PU = {e, n})</td>
<td>User B Key Generation</td>
</tr>
<tr>
<td>Private key: (PR = {d, n})</td>
<td>Select (n_B n_B &lt; n)</td>
</tr>
<tr>
<td>Calculate (P_A n_A)</td>
<td>Calculate (P_B = n_B \times G)</td>
</tr>
<tr>
<td>Encryption</td>
<td>Encryption</td>
</tr>
<tr>
<td>Plain Text : (M &lt; n)</td>
<td>Plain Text : (P_m)</td>
</tr>
<tr>
<td>Cipher Text : (C = M^e \mod n)</td>
<td>Cipher Text : (C_m = {KG, P_m + KP_B})</td>
</tr>
<tr>
<td>Decryption</td>
<td>Decryption</td>
</tr>
<tr>
<td>Cipher Text : (C)</td>
<td>Cipher Text : (C_m)</td>
</tr>
<tr>
<td>Plain Text : (M = C^d \mod n)</td>
<td>Plain Text:</td>
</tr>
<tr>
<td>Notations</td>
<td>Notations</td>
</tr>
<tr>
<td>(M \rightarrow \text{Plain Text})</td>
<td>(P_m \rightarrow \text{Plain Text})</td>
</tr>
<tr>
<td>(C \rightarrow \text{Cipher Text})</td>
<td>(C_m \rightarrow \text{Cipher Text})</td>
</tr>
<tr>
<td>(p, q \rightarrow \text{Prime Numbers})</td>
<td>(P_A \rightarrow A's \text{ Public Key})</td>
</tr>
<tr>
<td>(n \rightarrow \text{Integer})</td>
<td>(n_A \rightarrow A's \text{ Private Key})</td>
</tr>
<tr>
<td>(\phi(n) \rightarrow \text{Euler's phi})</td>
<td>(P_B \rightarrow B's \text{ Public Key})</td>
</tr>
<tr>
<td>(d, e \rightarrow \text{Multiplicative inverse modulo } \phi(n))</td>
<td>(n_B \rightarrow B's \text{ Private Key})</td>
</tr>
<tr>
<td>(e \rightarrow \text{Sender known value})</td>
<td>(G \rightarrow \text{Point on EC whose order large value } n)</td>
</tr>
<tr>
<td>(d \rightarrow \text{Receiver known value})</td>
<td>(K \rightarrow \text{Secret Key})</td>
</tr>
<tr>
<td>Security</td>
<td>Security</td>
</tr>
<tr>
<td>Integer Factorization Problem</td>
<td>Elliptic Curve Discrete Logarithmic Problem</td>
</tr>
</tbody>
</table>

The operational distinctiveness of RSA over ECC has been compared by Robshaw and Yin (Robshaw 1997). When comparing public key cryptographic systems, there are two distinct factors to take into account, namely security and efficiency.
Security: Security of cryptography is based on the existence of cryptosystem which is in wide use.

Efficiency: The computation required to perform public key and private key transformations and the number of bits communicated to transfer an encrypted message or signature.

RSA is the most commonly preferred public key cryptography scheme (Cilardo 2011), but recently new technology gains more attention on elliptic curve cryptography. ECC is relatively a new asymmetric cryptographic scheme, which provides the same level of security with much smaller key size compared to RSA scheme. The major advantage of ECC over RSA lies in the size of the key. Table 1.2 shows the variation of key sizes by comparing two public key cryptographic algorithms RSA and ECC.

Table 1.2 Comparison on Keys

<table>
<thead>
<tr>
<th>Key Size</th>
<th>ECC Key Size</th>
<th>RSA Key Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>56</td>
<td>112</td>
<td>512</td>
</tr>
<tr>
<td>80</td>
<td>160</td>
<td>1024</td>
</tr>
<tr>
<td>112</td>
<td>224</td>
<td>2048</td>
</tr>
<tr>
<td>128</td>
<td>256</td>
<td>3072</td>
</tr>
<tr>
<td>192</td>
<td>384</td>
<td>5120</td>
</tr>
<tr>
<td>256</td>
<td>512</td>
<td>15360</td>
</tr>
</tbody>
</table>

The mathematical problems in which elliptic curve cryptosystem rely on is the discrete logarithm problem. The shorter key length is used in the implementation of elliptic curve cryptosystem. The reason behind is that, there is unknown sub exponential time on the mathematical problem (RSA Laboratories 2002).
The advantage of elliptic curve cryptography over RSA is that Elliptic Curve Discrete Logarithm Problem (ECDLP) runs in full exponential time (Johnson 2001). ECC is considered as convenient to be implemented on limited resource applications, specifically in achieving a higher level of security while using binary field, which is suitable to be used in hardware designs. Another scope for ECC is the usage of less memory space. The key generation in ECC algorithm does not require large memory to perform the needed calculations.

ECC has been accepted commercially and adopted by standardizing bodies such as American National Standards Institute (ANSI X9.62 1999), Institute of Electrical and Electronics Engineers (IEEE-P1363-2000), International Organization for Standardization (ISO/IEC15946 2002), Standards for Efficient Cryptography Group (Certicom, SEC 1. Elliptic Curve Cryptography 2000) and National Institute of Standards and Technology (NIST 2000) (Certicom, SEC 1 2000).

1.2 ELLIPTIC CURVE MONTGOMERY ALGORITHM

Elliptic curve is one of the strongest public key cryptosystem, which is generally used for authentication protocols. The performance of such cryptosystem is primarily determined by effectively implementing scalar multiplication and exponentiation. For performance as well as for physical security reasons, it is often advantageous to recognize them by using hardware. The discussion for scalar events and hardware implementation are as follows.

In general, there are three methods used to compute the Elliptic Curve scalar events, namely Barrett Algorithm, Montgomery Algorithm and Classical Algorithm (Phung 2006). Compared to the three algorithms,
Montgomery algorithm is chosen to be the best for hardware implementation. The Montgomery algorithm is also an effective method to compute modulo exponentiation, because this algorithm speeds up the scalar events and exponentiation processes (Knezevic 2010).

Most of the hardware implementations of Elliptic curve systems were based on Montgomery algorithm. Montgomery algorithm developed a very efficient technique to compute in the group associated to an elliptic curve over a binary finite field $F(2^m)$, in which prime field is $F(p)$ (Tenca 2000). The Montgomery algorithm proves to be useful for point operations in ECC. The point operations can be classified into point addition and point doubling. More precisely, instead of sending a point as part of cryptographic protocol, there are variety of ways to perform Elliptic Curve based Montgomery operations (Vercauteren 2012).

1.2.1 Classification of Montgomery Algorithm

The classification of Montgomery algorithm for elliptic curve cryptography focuses on addition and multiplication operations.

1.2.1.1 Montgomery addition

Montgomery addition is an efficient method for scalar operation with an arbitrary modulus, particularly suitable for the implementations on general purpose computers and embedded microprocessors.

1.2.1.2 Montgomery multiplication

Montgomery multiplication is capable of improving the performance of an integer and binary polynomial multiplications. A number of unified architectures have been proposed for Montgomery Multiplication.
Elliptic curve based on Montgomery multiplication algorithms are very efficient for security. Montgomery Multiplication was introduced by P.L. Montgomery to improve the performance of computation (Muurling 2002).

Montgomery multiplication is defined both in prime $F(p)$ and binary fields $F(2^m)$. The Galois (G) of prime field is $GF(p)$ multiplication, which operates on numbers and for binary field is $GF(2^m)$ multiplication, which operates on polynomials (Gaj 2011). In $GF(p)$, Montgomery multiplication is defined as

$$MonPro(a,b) = ab^{r-1} \mod p,$$

where $a, b$ are two prime numbers, $r$ is an integer, and modulus ‘$p$’ of length ‘$n$’ bits. The Montgomery multiplication algorithm speeds up the scalar multiplication. It computes the Montgomery product as

$$MonPro(a,b) = ab^{r-1} \mod n$$

Given $a, b < n$ and $r$ such that the greatest common denominator $(n, r) = 1$. Although the algorithm works for any $r$ that is relatively prime to $n$, it is more useful when $r$ is taken to be a power of 2 (Olszyna 2011). In $GF(2^m)$, given the input polynomials $a(x), b(x)$ and $p(x)$ where $a(x)$ and $b(x)$ are the polynomials to be multiplied and $p(x)$ is the polynomial used to represent the modulus. Montgomery multiplication is defined as

$$MonPro(a, b) = a(x)b(x)x^{-n} \mod p(x).$$

$GF(2^m)$ incorporates on the polynomial leading carry for arithmetic (Lai 2010).

The polynomial based on Montgomery multiplier algorithm, which purely depends on the polynomial aspects. Here four parameters are used namely $A(x), B(x), C(x)$ and $P(x)$. $A(x), B(x)$ and $C(x)$ are the variables and $P(x)$ is the polynomial based mod property.
1.3 SIDE CHANNEL ATTACKS

Many tasks which are difficult to be performed with symmetric key cryptography can be performed in an elegant way, using public key cryptography. Security has long been a major concern in computing communication systems, and substantial research effort has been devoted for addressing it. Cryptographic algorithms including symmetric ciphers, public key ciphers and hash functions forms a set of primitives that can be used as building blocks to construct security mechanisms that target specific objectives.

In general, cryptographic algorithms are always implemented in software or hardware on physical devices which interact with and are influenced by their environment. These physical interactions can be instigated and monitored by adversaries like Eve, and may result in leakage of information useful in cryptanalysis (Venkatasubramani 2011). These information are called side channel information, and the attacks exploiting side channel information are called Side Channel Attacks (SCA). The basic idea of SCA is to look at the way in which cryptographic algorithms are implemented.

1.3.1 Introduction to Side Channel Attacks

Side Channel Attacks (SCA) are proposed by Kocher et al (Kocher 1999), in which an attacker observes side channel information such as computing time and power consumption from a cryptographic device. These information attempts to reveal secret messages hidden in that device without breaking it physically. Side Channel Attacks are attacks that are based on “Side Channel Information”. Side Channel Information retrieves the data from the encryption process in execution phase.
Analysis on SCA takes advantage on the implementation of specific
c characteristics to recover the secret parameters involved in the computation. It
is also easy to implement powerful attacks against cryptographic
implementations and their targets range from primitives, protocols, modules,
devices and even to systems. The main principle of SCA is to easily measure
the task because there is a correlation between the physical measurements
taken during computations (e.g., power consumption and computing time)
(Walter 2004).

This is the correlation between the side channel information and the
operations related to the secret key that SCA tries to find. The type of attack
can be identified by the amount of time required for the attack and by
analyzing the attack. These types can be enumerated depending on key factors
such as power analysis and time. According to these factors, it is classified as
Simple Power Analysis, Differential Power Analysis, Timing Analysis and
Fault Analysis. These attacks pose a serious threat to the security of
cryptographic modules. Inconsequence, cryptographic implementations have
to be evaluated for their resistivity against such attacks and the incorporation
of different countermeasures has to be considered.

1.3.2 Classifications of Side Channel Attacks

Traditionally, attacks can be classified as passive attack and active
attack according to the ability of the attacker. Active attacks are mainly
focused on internal circuitry of the cryptographic devices. While passive
attacks in which the adversary uses the standard functionality of the
cryptographic device.

An active attack disturbs the algorithm process to obtain an
abnormal behavior and/or an erroneous computation result that can be
exploited to recover entirely or partially the secrets. On the other hand,
passive attacks are based on the observation of various types of side channel attacks such as timing execution, power consumption, electromagnetic emanations or radio frequency of the chip to gain information on the operations and data used (Ma 2011). The proposed work mainly focused on passive attacks and their performances, which are very difficult to detect. The passive attacks are classified into two groups according to side channel information namely, Simple Analysis and Differential Analysis.

The literature usually classifies side channel attacks with two orthogonal axes namely Invasive and Non-Invasive. Invasive attacks require repackaging the chip to get direct access to its components. Non-invasive attack only exploits externally available information (the emission of which is however often unintentional) such as running time and power consumption. Skorobogatov and Anderson (Skorobogatov 2003) added a new distinction in invasive which is called as semi-invasive attacks. These attacks have the specificity that requires repackaging of the chip to get access to the chip surface, but do not tamper with the upper layer because they do not require electrical contact to the metal surface.

The following section describes about different side channel attacks namely Timing Attack, Fault Induction Attack, Acoustic Attack and Power Analysis Attack.

1.3.2.1 Timing attack

A timing attack is a side channel attack in which an attacker infers the secret information by using computation time as leaked data. Timing attack is also used for statistical analysis to reveal the secret information (Bucci 2006). These attacks are based on measuring the time, it takes to perform the operations on a unit.
Timing attacks exploit the timing information on the cryptographic hardware. Usually the running time of a program is merely considered as a constraint that must be reduced as much as possible by the programmer. More surprising is the fact that the running time of a cryptographic device can also constitute an information channel, providing the attacker with invaluable information on the secret parameters involved. This is the idea behind timing attack. This idea was first introduced by Kocher and then practically implemented against an RSA implementation using the Montgomery algorithm.

1.3.2.2 Fault induction attack

When an electronic device stops working correctly, the most usual reaction is to get rid of it. This insignificant habit may have deep impact in cryptography, these faulty computations are sometimes the easiest way to discover the secret key (Sakamoto 2011 and Karaklajic 2011).

As a matter of fact, a recent and powerful cryptanalysis technique consists of tampering a device in order to perform some erroneous operations. The result of that erroneous behavior will leak information about the secret parameters involved. This is described as fault induction attack.

1.3.2.3 Acoustic attack

Most side channel attacks research has focused on electromagnetic emanations, power consumption and recently, scatter visible light from CRT (Cathode Ray Tube) displays. However, one of the oldest eavesdropping channels, namely acoustic emanations has received little attention. Acoustic attack uses the dynamic power consumption and electromagnetic radiation which sounds as the output while cryptographic algorithms were executed (Chung 2012). This phase normally attacks typical works by finding some
information about the state of the cipher, which can be learned both by
guessing the key and by checking the value directly.

Very recently, Shamir et al (Shamir 2004) has demonstrated a
preliminary proof of concept which says a correlation exists between the
sound of a processor and its computation.

1.3.2.4 Power analysis attack

A power analysis attack allows the extraction of secret information
from the device at the time of execution. Power analysis attacks are divided
into two types namely Simple Power Analysis (SPA) and Differential Power
Analysis (DPA). These attacks were introduced in 1999 by Kocher et al
(Messerges 1999). They carried out a practical power analysis attack against a
DES implementation in hardware. Coron (Coron1999) was the first to apply
these attacks to elliptic curve cryptographic schemes and proposed SPA
resistant method for point multiplication and DPA resistant method for
randomizing projective coordinates.

Power consumption attacks are based on the observation that the
power consumed at a given time during cryptographic process is related to the
instruction being executed and the data being manipulated. The proposed
work mainly considered one side channel attack, namely power analysis
attack.

1.3.3 Types of Power Analysis Attacks

There are two types of power analysis attacks namely SPA and
DPA. Simple power analysis attacks on public key cryptosystems, in which
the sequence of executed instructions is usually related to the bits of the
private key. Thus, by observing the executed instructions from their different
power traces, the private key can be easily reconstructed. This is the only
possibility if the difference lies between the power traces of the individual instructions.

SPA attack is based on single observed power consumption, while the second type DPA attack combines SPA attack with an error correcting practice using statistical analysis. Many existing countermeasures are vulnerable to the new attacks which includes Zero Power Analysis (ZPA) and Doubling Attack (DA). In addition to its running time, the power consumption of a cryptographic device may provide much information about the operations that take place and the parameters involved in it. This is the idea behind simple and differential power analysis.

1.3.3.1 Simple power analysis attack

A SPA attack is a side channel attack in which an attacker infers the secret information by using power consumption as leaked data. This attack also reveals the secret information by direct observation of a device’s power consumption without the need for statistical analysis. The working principle consists of observing the power consumption during each single execution of a cryptographic algorithm.

Simple Power Analysis is generally based on looking at the visual representation of power consumption of a unit while an encryption operation is being performed. This technique involves direct interpretation of power consumption measurements collected during cryptographic operations. SPA can yield information about a device’s operation as well as the key material. The attacker directly observes the system’s power consumption. The amount of power consumption varies depending on the microprocessor instruction executed. Large features such as DES rounds, RSA operations, etc, may be identified, since the operations performed by the microprocessor vary significantly during different parts of these operations (Mangard 2003).
SPA analysis, for example, can be used to break RSA implementations by revealing difference between multiplication and squaring operations. Similarly, many DES implementations have visible differences within permutations and shifts and can thus be broken using SPA. Because SPA can reveal the sequence of instructions executed, this can be used to break cryptographic implementations in which the execution path depends on the data being processed.

1.3.3.2 Differential power analysis attack

DPA attack is a side channel attack in which an attacker infers the secret information by using statistical analysis of power consumption (Marimuthu 2008). This attack is said to be the most powerful side channel attack. DPA automatically locates correlated regions in a device’s power consumption, and then the attack can be automated. Here little or no information about the target implementation is required.

The goal of the Differential Power Analysis is to identify the secret key, or part of the secret key, used for ciphering the information. The basic idea is to correlate the encrypted data and the power consumed by the device. DPA attack is based on the same basic concept as SPA attack, but uses error correction techniques and statistical analysis to extract very small differences in the power consumption signals.

In addition to large scale power variations due to the instruction sequence, there are effects correlated to data values being manipulated. These variations tend to be smaller and are sometimes overshadowed by measurement errors and other noises.
1.3.3.3 Countermeasures against power analysis attacks

There are various countermeasures against the different types of power analysis attacks namely power balancing technique, reduction of signal size, addition of noise and modification of algorithm design. The descriptions are as follows,

**Power Balancing Technique**

The dummy registers and the logic gates play a major role in power balancing technique. These registers should be included in the design by which useless operations are made to balance power consumption into a constant value. A complementary operation should be performed on these dummy registers to assure that the total power consumption of the unit remain balanced according to higher value. Such technique, by which the power consumption is constant and independent on inputs and key bits, prevents all sorts of power consumption attacks such as SPA and DPA.

**Reduction of Signal Size**

One approach to prevent DPA attacks is to reduce signal sizes. Reduction of signal size countermeasure uses shielding device to reduce the size of signal available to an attacker. Generally Reduction of signal size cannot reduce the signal size to zero. For example constant execution path code leaks information by choosing various operations. The operations are power consumption, balancing hamming weights and state transitions (or) by shielding the devices.

**Addition of Noise**

Another approach against DPA involves introducing noise into power consumption measurements. Like signal size reductions, adding noise
increases the number of samples required for an attack, possibly to an unfeasibly large number. In addition, execution timing and order can be randomized to generate a similar effect (Rahuman 2010). Again, noise alone increases the number of samples required, however this increase is high enough to make the sampling unfeasible.

**Modification of Algorithm Design**

A final approach against DPA attacks involves designing cryptosystems with realistic assumptions for the basic hardware. Non-linear key update procedures can be employed to ensure that the power traces cannot be correlated between transactions (Anderson 1996).

As a simple example, a 169 bit hash key is generated using Secure Hash Algorithm (SHA). Before using it as a key all partial information should be destroyed, as an attacker might have gathered about the key. Similarly, aggressive use of exponent and modulus modification processes in public key schemes can be used to prevent attackers from accumulating data across large number of operations. This may solve the problem, but it does require design changes in the algorithm and protocol themselves, which are likely to make the resulting product noncompliant with standards and specifications.

**1.4 VLSI PLATFORM**

This topic mainly focuses on accomplishment platforms and hardware implementations of Elliptic Curve Montgomery. Elliptic Curve is an attractive and an alternative application for public key cryptography. This is because shorter keys can be used in elliptic curve when compared to other traditional public key cryptography. Thus secure communication is suggested as an important application for elliptic curve.
Field Programmable Gate Arrays (FPGAs) plays a major role and acts as the preferred platform for implementation. Traditionally, the uses of FPGA’s in hardware devices are prevented since it consume high power. Nowadays certain low power FPGAs are introduced and utilized in many research sectors.

In this thesis FPGA is implemented and worked on by Xilinx device tools. The first FPGA’s were introduced in the year 1970s. They were slow in performance and expensive to manufacture, eventhough the development of FPGA is rapid. Nowadays, device tools are in the mainstream of the markets as their capacity and performance have grown to meet the demands of modern digital designs. Particularly, low start-up costs, low financial risk and ease of design changes are the most compelling advantages of FPGAs (Esmaildoust 2012). Therefore, in markets a variety of FPGAs have grown rapidly in recent years.

Two major synthesis tools for FPGA devices are Xilinx and Altera. In this thesis, the proposed algorithm is implemented using Xilinx.

1.4.1 Structure of FPGA

FPGA enumerates various families such as Spartan II, Spartan IIE, Spartan III, Virtex II and Virtex III. In this thesis the main focus is given for Spartan III and an overview of the structure of Xilinx Spartan III FPGA device is discussed in this section.

Spartan III family of FPGA is specifically designed to overcome the needs for high cost and sensitive consumer electronic applications. This family builds on the success of the earlier Spartan IIE family. Spartan III made minor updates in Spartan IIE by increasing the amount of logic resources, the capacity of internal RAM, the total number of I/Os and the overall performance as well as by improving clock management functions.
Spartan III devices are like FPGAs, which consists of both non-programmable and programmable areas. The non-programmable area includes the Configuration logic, Boundary Scan logic and other components. The programmable area includes portions of the Input/Output Blocks (IOBs), Digital Clock Managers (DCMs), Configurable Logic Blocks (CLBs) and the initial content of the RAMs. Spartan III device configuration is purely based on FPGA based configuration (Gong 2010).

This configuration is required to define the Look Up Table (LUTs), signal routing, flip-flop reset polarity, Input/Output Blocks (IOB) voltage standards and all other aspects of the user design. The structure of FPGA consists of two important factors namely Configuration Modes and Reading Configuration Bits.

Configuration Modes

Spartan III devices can be configured through Select MAP (Slave Parallel) interface, serial interface or through Boundary-Scan (JTAG) interface. The configuration mode must be specified by setting the appropriate logic levels on the mode pins (M2, M1 and M0). The IOBs, CLBs and all other user configurable logic are configured in exactly the same way regardless of configuration mode.

Reading Configuration Bits from a Spartan III Device (Read back)

After configuring a Spartan III device, it is possible to read back the configuration logic to verify that the configuration remains as the user intended. Configuration Read back may be performed either by Select MAP mode or by Boundary Scan mode. Read back and all other Boundary Scan operations are available regardless of which configuration mode the device is in. However, read back is not possible through serial interfaces.
1.4.2 Elliptic Curve Cryptography on FPGAs

FPGAs are very attractive accomplishment platforms for Elliptic Curve Cryptographic (ECC) implementations, because of its speed and reprogrammability. On the other hand, the reprogrammability is in much importance, especially in ECC implementations where the parameters of the implementations must be changed frequently for security reasons.

1.4.3 A Generic FPGA Design Flow

![Image of FPGA Design Flow]

**Figure 1.1 FPGA Design Flow**

A generic FPGA design flow is presented in Figure 1.1. The design entry process is usually performed using Hardware Description Language (HDL). In this thesis Verilog Hardware Description Language (VHDL) is used for implementation. Functionality of the design is verified in the functional simulation process (Bening 2002). If the functional simulation is performed successfully, the synthesis process is performed. Else the design entry process must be performed again, means corrections must be made to the HDL code. In the synthesis process, a synthesis program decides which logic resources are needed to implement the design described in the HDL code (Fiskio 2008).
The synthesis program also optimizes the logic for the specific target device. Thus, the synthesis is a device dependent process. After the synthesis process, a place and route process is performed. In the placement phase, logic blocks given by the synthesis process are placed into the FPGA logic array (CLBs in Xilinx devices).

After the placement phase, the blocks are connected in the routing phase which is also an iterative process. Finally, a configuration bit stream is generated and the FPGA device is programmed. The design flow presented above is a generic design flow and this design flow is used for implementation.

### 1.4.4 Verilog

Work on IEEE 1364-2001 Verilog standard began in January 1997. Two major goals were established.

- Enhance the Verilog language to help with today’s intellectual property modeling issues.


In this thesis, the hardware implementation is based on Verilog.
Block Diagram

Figure 1.2 Block Diagram
Figure 1.2 explains the overall work of this thesis both in hardware and software.

1.5 THESIS ORGANIZATION

Chapter 1 focused on various fundamental aspects of Elliptic Curve Cryptography. An overview of cryptographic algorithms and basic level of Elliptic Curve Cryptographic operations are discussed. Elliptic Curve based on Montgomery Scalar Multiplication Algorithm plays an efficient role on the proposed work for carrying the scalar multiplication operations. A background on power analysis attacks (SPA and DPA) on these operations and the countermeasures from the foundation of the work were subsequently presented. Finally, the base of VLSI platform is explained, which is used for the implementation phase.

Chapter 2 represents the literature review of Elliptic Curve Montgomery Scalar Multiplication algorithm. It elaborates about both hardware and software implementations. In hardware implementation the main focus is on, how Elliptic Curve interacts with hardware, how Power Analysis attacks on Montgomery Algorithms can be overcome and how the power consumption can be measured through FPGA. The software implementation targets on execution time of the algorithm. Finally previous works for Addition, Doubling and Bit serial algorithms were discussed.

Chapter 3 focuses on multiplier and adder suited to calculate the power analysis for the proposed system. In adder section, comparison was made with Ripple Carry Adder and finally Carry Save Adder is proved to be the best. Similar way in multiplier section, comparison was made with different multipliers such as, Array Multiplier, Booth Multiplier and Wallace Tree Multiplier. Finally, as per results of the comparative study, Booth Multiplier is proved to be the best for implementation.
Chapter 4 investigates the power analysis of existing algorithms using the hardware design. The proposed system concentrates on File Register Architecture and Countermeasures. File register architecture is implemented to reduce the execution steps of the proposed algorithm. This reduction leads to minimum consumption of power. The countermeasures are Algorithm Modification and Power Balancing Technique. Algorithm Modification explains how the algorithm is modified and how the rules are interacted with the algorithm, and Power Balancing Technique explains the theoretical approach on power consumption.

Chapter 5 demonstrates the proposed design setup. The setup procedure explains the proposed logic synthesis, FPGA pin assignment and windows operation used in FPGA with Xilinx Devices. Both Elliptic Curve Montgomery Scalar Multiplication operations and Elliptic Curve Montgomery Scalar Bit Serial Multiplication algorithm results and their hardware and software analysis were discussed in Chapter 6. Chapter 7 explains the conclusion and discusses the future enhancements of this work.