Energy efficiency analysis of selected public key cryptoschemes

by

Tanushree Banerjee

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2018

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Public key cryptosystems in both classical and post-quantum settings usually involve a lot of computations. The amount as well as the type of computations involved vary among these cryptosystems. As a result, when the computations are performed on processors or devices, they can lead to a wide range of energy consumption. Since a lot of devices implementing these cryptosystems might have a limited source of power or energy, energy consumption by such schemes is an important aspect to be considered.

The Diffie-Hellman key exchange is one of the most commonly used technique in the classical setting of public key cryptographic shceme, and elliptic curve based Diffie-Hellman (ECDH) has been in existence for more than three decades. An elliptic curve based post-quantum version of Diffie-Hellman, called supersingular isogeny based Diffie-Hellman (SIDH) was developed in 2011. For computations involved in ECDH and SIDH, elliptic curve points can be represented in various coordinate systems. In this thesis, a comparative analysis of energy consumption is carried out for the affine and projective coordinate based elliptic curve point addition and doubling used in ECDH and SIDH. We also compare the energy consumption of the entire ECDH and SIDH schemes.

SIDH is one of the more than sixty algorithms currently being considered by NIST to develop and standardize quantum-resistant public key cryptographic algorithms. In this thesis, we use a holistic approach to provide a comprehensive report on the energy consumption and power usage of the candidate algorithms executed on a 64-bit processor.

Acknowledgements

First and foremost, I would like to thank God Almighty for giving me the strength, knowledge, ability and opportunity to undertake this research study and complete it satisfactorily. Without His blessings, this achievement would not have been possible.

The research of this thesis was done under the supervision of Professor Anwar Hasan. I wish to take this opportunity to express my sincere gratitude for his endless support and valuable guidance that have helped me succeed and provided me with the insights that I needed to progress. I am also indebted to Natural Sciences and Engineering Research Council of Canada for their generous financial support and to Electrical and Computer Engineering Department, University of Waterloo for the teaching assistantships during my MASc program.

I thank all the professors who taught me, expecially Professors Alfred Menezes, Andrew Morton, Mahesh Tripunitara and Mark Crowley. My heartfelt appreciation to all current and previous group members. Many thanks to my colleagues Mohannad and Raghavendra for all their suggestions and insights over the past two years. I would also like to thank my friend Subhra for encouraging and supporting me throughout this period.

Finally, I must express my profound gratitude to my parents and to my fiancé for providing me with unfailing support and keeping me motivated through the process of research and writing this thesis. This accomplishment would not have been possible without their support. Thank you all.

Dedication

This thesis is dedicated to my parents Partha Banerjee and Chandrima Banerjee for their endless love, support and encouragement. This accomplishment has been possible only because of their blessings.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CMOS Complementary Metal Oxide Semiconductor Technology

ECC Elliptic Curve Cryptosystem/Cryptography

ECDH Elliptic Curve based Diffie-Hellman

ECDLP Elliptic Curve Discrete Logarithm Problem

MSR Model Specific Registers

NIST National Institute of Standards and Technology

PAPI Performance Application Programming Interface

PQC Post Quantum Cryptograhy

RAPL Running Average Power Limit

SIDH Supersingular Isogney based Diffie-Hellman

Chapter 1

Introduction

1.1 Motivation

Energy consumption is an important factor for any cryptographic scheme implemented on devices having limited energy resources. Cryptographic protocols mostly involve various mathematical computations based on certain algorithms. Implementations for such algorithms in software or hardware require power and thereby they consume energy. In the past, a lot of research work has been dedicated towards improving the efficiency of the algorithms used in cryptographic schemes in terms of memory and speed. There has not been much investigations on the energy consumption by public key cryptographic schemes, when implemented in software. This energy consumption is dependent on both power usage and their execution time.

The very idea of communicating securely on public channels was introduced in 1976 as Diffie-Hellman key exchange [51] which is based on the hardness of solving the *discrete logarithm problem*. Elliptic curve based Diffie-Hellman (ECDH) is a protocol for key exchange and provides security in the classical setting. Using analogous ideas of key exchange, an elliptic curve based quantum safe protocol was later developed which is termed as supersingular isogeny based Diffie-Hellman (SIDH).

This thesis investigates the relative energy efficiency of ECDH and SIDH, where SIDH in quantum safe and ECDH is not. We also consider the NIST organized post-quantum cryptography (PQC) round 1 submissions (including SIDH which is submitted as SIKE). Although public key cryptosystems such as RSA [103] are widely used, we do not consider them in this work. Software implementations of these protocols are executed on a 64 bit processor for this investigation.

1.2 Contributions

In the first part of this thesis, implementations of both ECDH and SIDH are executed on a 64 bit processor in order to study their corresponding power and energy consumption values. It is already known that representation of elliptic curve points using projective coordinates are more efficient than affine coordinates with respect to execution *time*. In this thesis, a detailed comparison of power usage and energy consumption between affine and standard projective coordinates while executing elliptic curve based point doubling and addition is presented. In addition to this, we investigate the energy efficiency of the overall SIDH and ECDH key exchanges using standard projective coordinates. This work has been accepted at [20] and is also available on the CACR (Centre for Applied Cryptographic Research, University of Waterloo) website [22].

Another contribution of this thesis is a consolidated report on power usage and energy consumption of candidate algorithms which were submitted to the NIST post-quantum cryptography standardization process. In this investigation, optimized implementations of all the submitted signature schemes and encryption/encapsulation techniques are executed on a 64 bit processor in order to accumulate their energy consumption data. This work is under submission for possible publication and is also available on the CACR website [21].

1.3 Thesis organization

The rest of this thesis is organized as follows. Chapter 2 provides some preliminaries on ECDH and SIDH. Chapter 3 gives an overview of the tools and methodologies used. Chapter 4 presents our investigation results on power usage and energy consumption of affine and projective coordinate based elliptic curve point addition and doubling, followed by ECDH and SIDH power usage and energy requirements. Chapter 5 consists of an extensive comparison of the NIST round 1 submissions in terms of their energy consumption. Finally concluding remarks are provided in Chapter 6. Scopes and possibilities of future work related to this thesis are also added in the same chapter.

Chapter 2

Overview of elliptic curve based pre- and post-quantuam cryptosystems

2.1 Introduction

Elliptic curve cryptography (ECC) was proposed in 1985 independently by Neal Koblitz [76] and Victor Miller [92]. Since then a large variety of security implementations in public key cryptography have been done using elliptic curves. ECC is used in many practical applications [36] such as smart grids, vehicular communication, RFID, secure shell (SSH) [111], transport layer security (TLS) [34] and Bitcoin [96]. Public-key algorithms are particularly crucial since they provide digital signatures and establish secure communication without requiring in-person meetings. ECC provides a secure means of exchanging keys among communicating hosts using the Diffie–Hellman (DH) key exchange algorithm referred to as ECDH [51]. The possibility of the emergence of quantum computers in the near future poses a serious threat against the security of widely-used public key cryptosystems such as RSA and ECC.

Fortunately, there exist public key cryptographic algorithms that are believed to be safe against quantum computer based attacks. Most of these quantum safe algorithms are either code-, lattice-, hash- or multivariate-based. Amongst all the other known post-quantum cryptosystems, an elliptic curve based algorithm has recently received considerable attention. It relies on the Diffie-Hellman construction using the isogeny of supersingular elliptic curves, referred to as SIDH [70]. The best known classical and quantum attacks against the underlying Diffie-Hellman problem of SIDH are both exponential in the size of the underlying finite field. This makes SIDH quite promising as a post-quantum crypto candidate.

The fundamental operation underlying ECC is elliptic curve point multiplication, which in turn uses point doubling and addition. An elliptic curve can be represented using various coordinate systems [64]. For each such coordinate system, the formulae for computing point addition and doubling are different, as a result the speed of computation is also different. Therefore a good choice of coordinate system is an important factor for elliptic curve point multiplications. The use of affine coordinates and projective coordinates for elliptic curve point operations is well known [109]. To the best of our knowledge, there has not been any work done that reports the power and energy consumption values corresponding to the use of such coordinates. In this thesis, we provide insight into the differences in energy consumption between affine and projective coordinate based point addition and doubling used in ECDH and SIDH. We then use projective coordinates to report the differences in energy consumption between the entire key exchange of ECDH and SIDH.

We next provide details of the parameters that are used in the cryptographic protocols of ECDH and SIDH. Both of these schemes involve scalar point multiplication, and are defined on different finite fields. However, SIDH requires isogeny evaluation and computation, unlike the ECDH scheme.

2.2 ECDH scheme

Diffie–Hellman key exchange establishes a shared secret between the two communicating nodes that intend to securely exchange data over a public network. The scheme uses the multiplicative group of integers modulo p, where p is prime [64]. The elliptic curve Diffie–Hellman protocol is a variant that uses an additive group formed by points on a suitably chosen elliptic curve instead of the multiplicative group of integers modulo p.

Elliptic curve based Diffie-Hellman protocol relies on the difficulty of computing the Elliptic Curve Discrete Logarithm Problem (ECDLP) [51]. The ECDLP is following: *Given an elliptic curve E defined over a finite field F_p, a point P of order n on E, and a point Q that is a multiple of P, finding the integer $l \in [0, n-1]$, such that $Q = lP$.*
The ECDH scheme works in the following way. Suppose Alice and Bob are communicating over a public channel. The following are the parameters used :-

E = Chosen elliptic curve defined over F_p

P = publicly known base point on E of order n

a = Alice's private key, known only to Alice, and chosen randomly from $[0, n-1]$

b = Bob's private key, known only to Bob, and chosen randomly from $[0, n-1]$

aP = Alice's public key

bP = Bob's public key

abP = Shared secret key

ECC mainly exploits the algebraic structure of the elliptic curves over a finite field. Elliptic curve based cryptographic schemes use the smallest sized keys and as a result is one of the most popular choices for many practical applications. In terms of security, ECDLP is hard to solve and there is no polynomial time attack known against it. Known classical attacks against the ECDLP have exponential time complexity.

There exist several standards on selecting safe curves for implementations in ECC. The standard developed by NIST FIPS 186-2 [59] is used in this study. Based on their standards, factors such as required security level, underlying prime field and appropriate curve on that field are to be chosen. Elliptic curves can be either chosen from pseudorandom curves or special curves. In the former one, coefficients of the curves are generated from the output of a seeded cryptographic hash function SHA-1. And in case of special curves, the coefficients and the underlying field are selected such that the efficiency of the elliptic curve operations can be optimized. Pseudorandom curves can be defined over prime fields of order p or any binary field of order 2^m. Elliptic curves defined over binary fields provide easier implementations, and many relevant algorithms and implementations have been reported, e.g., [65], [48], [86]. However, there are some concerns that faster attacks might be discovered [105], [73], [78]. Therefore pseudorandom elliptic curves defined over prime fields are considered in this work, where the security level of the implementation is decided by the size of the chosen prime.

In this study, implementation of ECDH is done such that it provides the classical bit security of 128 and 192. We use the NIST recommended curves P-256 and P-384 in this investigation [59]. The pseudorandom curves are in the short Weierstrass form [109]:

$$y^2 = x^3 + a \cdot x + b \mod p \tag{2.1}$$

This curve is defined over the field F_p, where p is a prime of length 256 bits or 384 bits for providing 128 bits and 192 bits of security. The value of the coefficient b is also provided by

NIST. The coefficient a is chosen to be -3, so that it would require fewer field computations for elliptic curve point operations.

For the curve P-256, the prime used is $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$ [64]. The value of the coefficient b and also the base point $P = (g_x, g_y)$ is provided by NIST as follows:

b =0x5ac635d8aa3a93e7b3ebbd55769886bc651d06b0cc53b0f63bce3c3e27d2604b

g_x =0x4fe342e2fe1a7f9b8ee7eb4a7c0f9e162bce33576b315ececbb6406837bf51f5

g_y =0x4fe342e2fe1a7f9b8ee7eb4ac0f9e162bce33576b315ececbb6406837bf51f5

Similarly for P-384 the curve, the prime used is $p = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$. The value of the coefficient b and also the base point $P = (g_x, g_y)$ is given by:

b =0xb3312fa7e23ee7e4988e056be3f82d19181d9c6efe8141120314088f5013875ac65639
8d8a2ed19d2a85c8edd3ec2aef

g_x =0xaa87ca22be8b05378eb1c71ef320ad746e1d3b628ba79b9859f741e082542a385502f
25dbf55296c3a545e3872760ab7

g_y =0x3617de4a96262c6f5d9e98bf9292dc29f8f41dbd289a147ce9da3113b5f0b8c00a60b1
ce1d7e819d7a431d7c90ea0e5f

The prime p listed above have a special property such that they can be written as the sum or difference of small numbers of the powers of 2, making reductions modulo p easier.

2.3 SIDH scheme

The basic idea of SIDH key exchange is based on Diffie-Hellman. However, the properties of isogenies between supersingular elliptic curves over $F_{p_s^2}$ are exploited to attain quantum resistance. Security of SIDH depends on the following hard problem:

Given two supersingular elliptic curves, say E and E', defined over $F_{p_s^2}$, find an isogeny $\phi : E \rightarrow E'$ of degree l^e where l=2 or 3.

An isogeny $\phi : E \rightarrow E'$ is a rational map from the curve E to E' satisfying $\phi(0) = 0$ and $\phi(E) \neq 0$ [109]. Two supersingular elliptic curves E and E' are isogenous to each other over $F_{p_s^2}$ if and only if $\#E(F_{p_s^2}) = \#E'(F_{p_s^2})$, that is number of points on both the isogenous elliptic curves are same. It should be noted that $F_{p_s^2}$ is a quadratic extension of the prime field F_{p_s} where the prime p_s is of the form $({l_A}^{e_A} \cdot {l_B}^{e_B} \pm 1)$ [47]. The integers l_A and l_B are small primes which are in the case of SIDH are 2 and 3. The integers e_A and e_B indicate the number of degree-l_A and degree-l_B isogenies to be computed at the sender and receiver end, respectively. The extension field is formed as $F_{p_s^2} = F_{p_s}(i)$ where $i^2 + 1 = 0$. The order of the supersingular elliptic curve is $(l_A^{e_A} \cdot l_B^{e_B})^2$. The choice of the underlying field decides the security level of the implementation. For $l \in \{l_A, l_B\}$ and $e \in \{e_A, e_B\}$, the SIDH secret keys are isogenies of the base curve E of degree l^e, which are in one-to-one correspondence with the cyclic subgroups of $E_{A,B}(F_{p_s^2})$ of order l^e, that form the kernels of the isogeny [70]. In SIDH, the supersingular elliptic curves are all defined over $F_{p_s^2}$.

Given a finite subgroup $H \subseteq E(F_{p_s^2})$, there exists a unique separable isogeny $\phi : E \rightarrow E'$ with kernel$(\phi) = H$; the degree of ϕ is $|H|$. Velu′s formula [115] can be used to find the isogeny ϕ and the isogenous curve E' which is also denoted by E/H. For arbitrary subgroups, Velu′s formula is computationally infeasible. Therefore this particular SIDH implementation, that we have discussed in this thesis, uses isogenies over subgroups that are powers of 2 and 3. The Kummer variety of Montgomery curve, a special form of elliptic curve has been used in the SIDH implementation.

Let $A, B \in F_{p_s^2}$ satisfy $B(A^2 - 4) \neq 0$ in $F_{p_s^2}$ (where the characteristic of $F_{p_s^2} \neq 2$). The Montgomery curve $E_{A,B}$ defined over $F_{p_s^2}$ is denoted by $E_{A,B}/F_{p_s^2}$. The set of points $P = (x, y)$ are solutions in $F_{p_s^2}$ to the curve equation

$$B \cdot y^2 = x^3 + A \cdot x^2 + x \tag{2.2}$$

The Montgomery curve used for SIDH has the coefficients $A = 0$ and $B = 1$. So the public starting curve is

$$y^2 = x^3 + x \tag{2.3}$$

This is a special instance of the Montgomery curve of order $(2^{e_A} \cdot 3^{e_B})^2$ and e_A and e_B are the two numbers that define the finite field $F_{p_s^2}$, where $p_s = 2^{e_A} \cdot 3^{e_B} - 1$ [69]. In this particular implementation, values of e_A and e_B are 250 and 159 for 128 bits classical security level, and 372 and 239 for 192 bits security level.

Coming to the details of key exchange technique, suppose Alice and Bob wish to communicate over a public channel. Then, the following are the parameters used:

- A prime p_s of the form $l_A^{e_A} \cdot l_B^{e_B} - 1$
- A supersingular elliptic curve E_0 defined over the field $F_{{p_s}^2}$
- Fixed points P_A, P_B, Q_A and Q_B are selected randomly on the elliptic curve E_0 such that the order of the points P_A and Q_A is $l_A^{e_A}$ and that of the points P_B and Q_B is $l_B^{e_B}$.

Furthermore, (P_A, Q_A) are independent, which can be checked by computing the Weil pairing $e(P_A, Q_A)$ in $E_0[l_A^{e_A}]$ and verifying that the result has order $l_A^{e_A}$. Same thing is done for P_B and Q_B.

Next, during the key exchange at both the nodes, computation of isogeny mapping of a particular degree is required, followed by generation of the isogenous curve. Given below is a brief description of the sequence in which the computations are done at both the ends by Alice and Bob:

1. Alice chooses m_A, n_A, randomly from $[0, l_A^{e_A} - 1]$, not both divisible by l_A and computes the isogeny $\phi_A : E_0 \to E_A$, where the kernel of ϕ_A is $m_A[P_A] + n_A[Q_A]$
2. Next she evaluates $\phi_A(P_B)$ and $\phi_A(Q_B)$ and transmits $\phi_A(P_B)$, $\phi_A(Q_B)$ and the curve E_A to Bob
3. Similarly, Bob chooses m_B, n_B, randomly from $[0, l_B^{e_B} - 1]$, not both divisible by l_B and computes the isogeny $\phi_B : E_0 \to E_B$, where the kernel of ϕ_B is $m_B[P_B] + n_B[Q_B]$
4. Bob evaluates $\phi_B(P_A)$ and $\phi_B(Q_A)$ and transmits them along with the curve E_B to Alice
5. After receiving $\phi_B(P_A)$, $\phi_B(Q_A)$ and E_B from Bob, Alice computes $\phi_{AB} : E_B \to E_{AB}$ where the kernel of ϕ_{AB} is $m_A\phi_B(P_A) + n_A\phi_B(Q_A)$
6. In the same way, Bob computes $\phi_{BA} : E_A \to E_{BA}$ where the kernel of ϕ_{BA} is $m_B\phi_A(P_B) + n_B\phi_A(Q_B)$
7. Finally both the nodes compute the shared secret key as the j-invariant of the final evaluated isogenous curve, i.e., E_{AB} for Alice and E_{BA} for Bob. By the properties of isogenies we have $j(E_{AB}) = j(E_{BA})$.

Above is the basic idea behind the key exchange. There has been quite an amount of research done for further compressing the key sizes, decreasing the amount of computations involved, increasing speed of implementation etc. [3], [58], [15]. The mathematical details involved in the computation of isogenies is beyond the scope of this thesis. Interested readers might want to refer to [47], [32], [50], [45], [71], [70], [69] as a sample of recent research.

2.4 Comparative remarks

It must be noted that although both the key exchanges involved in ECDH and SIDH operate on the basic idea of Diffie-Hellman, they have fundamental differences in terms of the algorithms used. In case of ECDH, there is a requirement for elliptic curve based scalar point multiplication. Scalar point multiplication is computing $k \cdot Q$ where Q is a point on the elliptic curve and k is an integer. It is used in ECC as a means of producing a one-way function. Various algorithms are available for this computation such as:

- Double and add [64]
- Windowed method
- Sliding window method
- w-ary non-adjacent form (w-ary NAF)
- Montgomery ladder [93]

The first one mentioned above (double and add algorithm) is the simplest technique known. The other techniques mentioned provide some kind of optimization. Montgomery ladder for point multiplication computes in fixed amount of time, thereby protecting the system from side channel attacks [77]. In w-ary non-adjacent form (w-ary NAF), the points are

Table 2.1: Sizes of keys and the primes used in ECDH and SIDH [47] protocols

Classical bit security level	ECDH (Length of keys and primes in bits)				SIDH (Length of keys and primes in bits)			
	Prime	Private Key	Public Key	Shared Secret Key	Prime	Private Key	Public Key	Shared Secret Key
128	256	256	512	256	503	256	3024	1008
192	384	384	768	384	751	384	4512	1504

represented in a different form in order to reduce the number of operations required for point multiplication [33]. In this investigation we have used the simplest algorithm, namely double and add, for scalar point multiplication in ECDH. The reason behind using this is discussed later in Section 4.1. Moreover, it should also be noted that all the computations in ECDH are done on a fixed elliptic curve, which is known publicly.

For SIDH, there are multiple isogenous elliptic curves involved. Each time the scalar point multiplication is performed on a different curve. The main operations involved in this technique, which are different from the aforementioned ones, are:

- Computing isogenies of degrees $l_A^{e_A}$ and $l_B^{e_B}$
- Evaluating isogenies of degrees $l_A^{e_A}$ and $l_B^{e_B}$

Both of these are computed using Velu's formula [115]. The degree of isogeny computed at both the communicating nodes are different which imply Alice and Bob perform different kinds of computations, unlike ECDH. As mentioned before, in this work we are considering the classical bit security levels of 128 and 192. Table 2.1 provides the public, private and shared secret key sizes used for both the security levels with ECDH and SIDH. The public key size is almost 6 times larger for SIDH than for ECDH for the same level of security. This demands extra storage in the device that implements the scheme. Private keys are of same size but the shared secret key of SIDH is around 4 times larger than the ones used in ECDH [69], [47].

2.5 Summary

In this chapter we have provided preliminary details of ECDH and SIDH, both of which are based on the same fundamental key exchange technique. The overall key exchange sequence used in both the schemes are also provided. The internal details of the isogeny computation of SIDH has been ignored, as we are only concentrating on the comparison of the two schemes on the basis of their energy efficiency.

Chapter 3

Tools and methodologies

3.1 Overview

When a program is executed on a general purpose processor or CPU, the latter draws power from an energy source, such as a battery. The energy consumed by the processor to execute the program is essentially the product of the average power usage (or consumption) and the execution time. In this section, we provide a brief overview of various factors that affect the processor's power usage and tools available to monitor the power.

Today's CPUs are based on Complementary Metal Oxide Semiconductor (CMOS) Technology. CMOS technology theoretically only dissipates power when switching of states occurs accounting for the *dynamic power* of CPU. There is however also some leakage which is known as *static power*. Therefore, the total power dissipation can be written as:

$$CPU_{TotalPowerDissipation} = Power_{Dynamic} + Power_{Static}$$

The *static power* dissipation depends mostly on these two factors:

- Subthreshold conduction
- Tunneling current through gate oxide layer

It has been determined that this tunneling power dissipation is one of the major components of power dissipation. As the size of processors is getting smaller, the metal oxide layer becomes thinner, making it easier for electrons to tunnel through the insulating layer. So at a particular supply voltage, tunnelling is the largest factor of leakage. On the other hand, the former situation arises when the transistor is in subthreshold region, by leakage of current between the source and the drain.

Dynamic power dissipation is controlled by factors such as:

- Transition
- Short-circuit power dissipation

where transition power arises from the voltage source charging up the gates as if it is a capacitor and then the capacitor discharging to the ground following the equation $P_{transition} = 0.5 \cdot C \cdot V^2$

Interestingly, different types of power measurement counters (core and uncore) are available on the smartphone, laptop, desktop and other hardware. These performance counters are used to provide information about how a particular operating system or the related applications are functioning on a real-time basis. They can monitor power usage when a particular instruction or process gets executed. Some of the performance counters that affect power usage in a mobile device are as follows:

- Instructions per cycle (IPC): Power usage of a processor is dependent on its activity. If IPC is high, the processor is likely to use more power.
- Fetch counters: Processors execute huge number of instructions speculatively. In case of branches in codes, branch prediction mechanism has a role to play. So, the fetched instructions, branch correct predictions (BCP), branch mis-predictions (BMP) contribute to power consumptions.
- Miss/Hit counters: Upon cache misses, the processor stalls. Thus, the events such as cache hits, cache miss, TLB (Translation Lookaside Buffer) miss may impact the power consumed.
- Retired instructions counters: Depending on the type of the retired instructions, different functional units are exercised. If some of these executions are power-hungry then they can influence power consumption.
- Stalls : Processors stall due to dependencies cause some power usages.

There are some software/hardware options available for monitoring or measuring the battery discharge rate while the program is executed on a laptop or any other processor. Most of the Intel chips since the introduction of the SandyBridge architecture, have RAPL (Running Average Power Limit) feature. This is primarily an estimation of the power used, although some Haswell server models apparently have actual power measurement due to onboard voltage regulation. The primary intent of RAPL is to control power usage on a chip, but it also has power and energy measurement capabilities that make it interesting for Performance Application Programming Interface (PAPI) [114]. This PAPI is a platform independent interface, available to monitor the processor events and relate software processes with the hardware in almost real time. The PAPI RAPL provides power data by relying on the values of RAPL MSR (Model Specific Registers). Mainly two basic types of events can be reported from RAPL. They are:

- Dynamic energy readings from various components of the chip ($PACKAGE\_ENERGY$), DRAM ($DRAM\_ENERGY$), CPU ($PP0\_ENERGY$), GPU ($PP1\_ENERGY$) etc.
- Static fixed values for thermal specifications, maximum and minimum power caps, and time windows over which power is monitored.

Software is available to read and monitor such power usage values while the machine executes instructions. For Unix based operating systems some of the exclusively available power monitoring options are:

- Powerstat [75]: This is a program that measures the power consumption of a mobile processor that has a battery power source. After monitoring, it calculates the average, standard deviation and min/max of the gathered power usage data. There are options provided in its syntax to specify the sampling frequency, duration, etc., during measurement.
- PowerTop [110]: This is a terminal-based diagnosis tool developed by Intel that helps to monitor power usage by programs running on a Linux system when it is not plugged on to a power source. An important feature of this piece of software is that it provides an interactive mode which allows a user to experiment with different power management settings.
- LibMSR [83]: This tool provides a convenient interface to access MSRs and to utilize their full functionality. The main target was Intel SandyBridge processor. Later, there have been plans to provide support for other generation processors as well.

Next, the available options to monitor power usage on both Windows and Ubuntu are as follows:

- Microsoft joulemeter [62]: This software based power monitoring tool, developed by Microsoft, can measure energy impact on disk, CPU, screen etc. It can also be used with desktops, but in that case an expensive piece of hardware, known as Watt's Up Meter would also be required.

- Intel's performance counter monitor (PCM) [116]: This provides sample C++ routines and utilities to estimate the internal resource utilization. PCM tool reports energy consumed by the socket and DRAM in the last one second. Therefore, the energy consumed by the system in the last one second is also a measure of power (energy per second). The performance output and the power usage can also be saved in .csv format. This software is responsible to monitor several other events also in abstraction.

- Intel power gadget [119]: This is a software-based power usage monitoring tool enabled for Intel Core processors (from 2nd Generation up to 7th Generation Intel Core processors) developed by Seung-Woo Kim et al. It is available for Windows, Ubuntu and Mac platforms. It gives accurate power usage data at user defined sampling rate and for desired time duration. Therefore, it is used in this work.

- Energy consumption tool, Visual Studio 2013: This package is part of the Performance Profiler of Visual Studio 2013. However, this CPU Usage tool works on only Desktop apps and Windows Store apps exclusively.

In addition to the above mentioned options, there have been other arrangements used in the past to get power and energy measurements precisely from laptops. For example, Farkas et al. [56] described the use of a shunt resistor to be inserted in series with the power source or battery of the laptop. Next, a precise voltmeter would measure the voltage values over the resistor continuously. Power used could be evaluated using Ohm's law. This technique was further validated in [55].

3.2 Intel power gadget

This gadget was developed for Windows, Ubuntu and Mac. But currently the Ubuntu version is not working properly. Hence, the application of version 3.5 for Windows operating system is used in our work to monitor the power usage with this application. The Windows version is considered to be an accurate data logger and also flexible for usage according to user reviews from Intel Applications Forum. Figure 3.1 is a screenshot of the application showing the graphs of power consumption while being used.

This gadget includes an application, driver and libraries to monitor and estimate real-time processor package power information in Watts using the energy counters in the processor, which is collected at a user defined sampling rate and logged onto a .csv file. There are additional features that include estimation of power on multi-socket systems. The multi-socket support essentially evaluates the energy from MSR on a per-socket basis and provides an estimate of power drawn per socket. So, the data that can be extracted from the gadget outputs are: processor power (Watts), temperature (Celsius), CPU Utilization, DRAM Power and frequency (MHz) in real-time via graph displayed in the GUI. It also allows to log the power and frequency measurements and save it in csv format. C/C++ Application Programming Interface (API) is also available in this gadget, for accessing this power and frequency data in programs.

It has been noticed that the power consumption values are negligible prior to execution of the codes for the purpose of monitoring power usage. And once the program execution starts, the power usage value rises which is reflected on the topmost display dial of the gadget.

3.3 Experimental details

In this section, we focus on describing the procedure that is adopted in this investigation. As mentioned above, the power usage is monitored using the Intel gadget. Also, Windows operating system is used for building and executing all the implementations, as only the Windows version of the power gadget has been providing accurate results.

In terms of the C implementations that are executed, Linux Subsystem is used as the platform. Linux Subsystem(Ubuntu 16.04) is a compatibility layer for running Linux binary executables natively on Windows (we used Windows 10). The complier that we are using is gcc 5.4 on Intel i7-6700 Skylake, 64 bit processor. Windows based platforms like Eclipse or Visual Studio have certain constraints, which make the process of compilation more complicated, especially when the NIST based submissions were designed keeping in mind Unix based system. So in order to maintain consistency, all the programs in this work is executed on the Ubuntu platform through the subsystem. Therefore, while executing the programs, the power usage is tracked simultaneously using Intel power gadget 3.5 and logged in .csv files with a sampling resolution of 50 msecs. After each session of logging power data, the average power consumed is considered for computation of energy consumption, at each instance.

Implementations of elliptic curve based point addition and doubling are done using standard projective coordinates and affine coordinates. These computations are done for both ordinary elliptic curve of Weierstrass form (as used by ECDH) and for

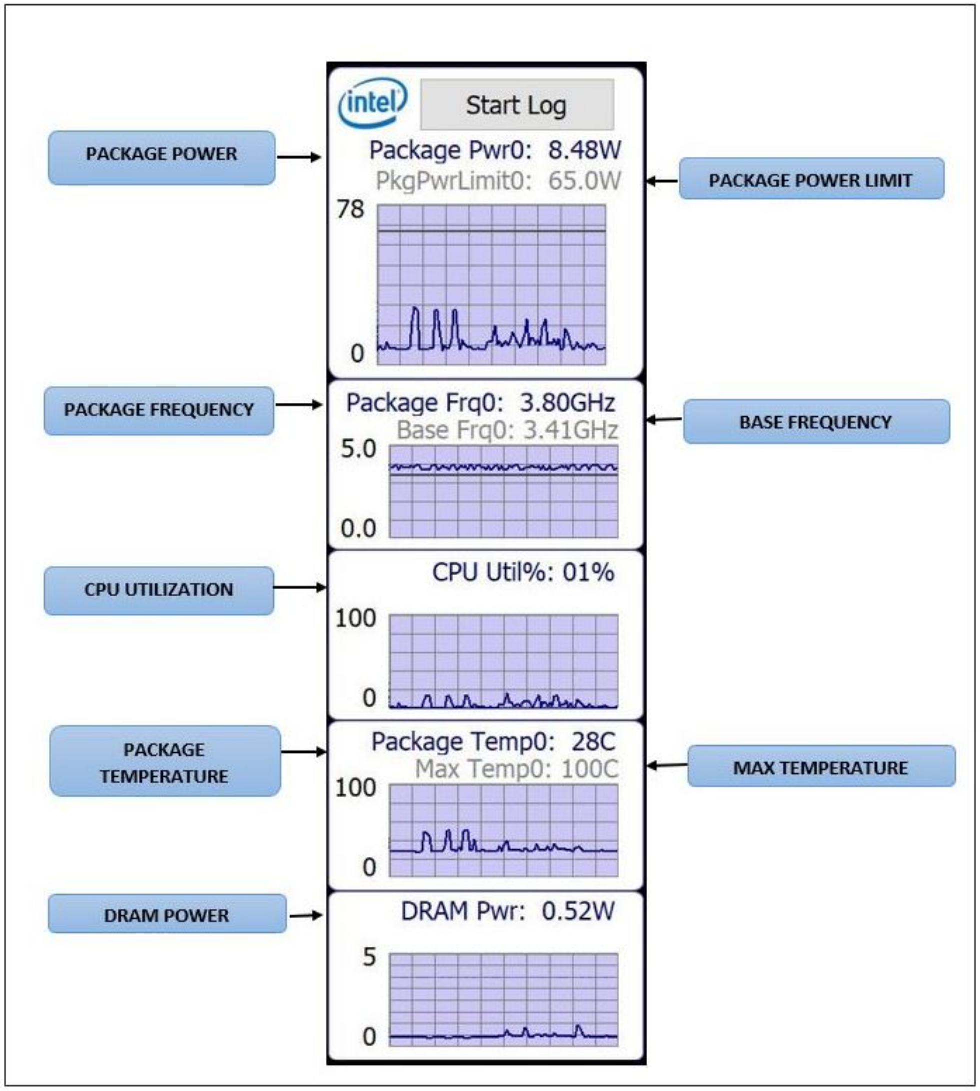

Figure 3.1: Intel power gadget 3.5 interface

supersingular elliptic curve, which is considered in Kummer variety of Montogomery curve (as used in SIDH). Next, power profiling along with benchmarking of their execution time is done. The coordinate system that consumes the least energy is then used to represent points on elliptic curve, in the ECDH and SIDH schemes, in order to compare their relative energy efficiency during the key exchange.

Furthermore, the same idea is extended to investigate the energy consumption of the NIST round 1 PQC submissions. Initially there had been 69 submissions, out of which 5 were broken in terms of security by the time NIST held the first post-quantum cryptography standardization conference in April 2018. In this study, we omit those five submissions. All candidate algorithms include software implementations for different security levels mentioned by NIST. It has been tested that the execution times of the implementations do not depend on whether Ubuntu operating system or Linux Subsystem is used. So, energy consumption is reported for all these schemes, and categorized, based on their security levels, encryption, encapsulation or signature techniques etc. It should be noted that the implementations that have been considered are only based upon the submissions to round 1 of the standardization process, and do not take into consideration any optimizations and changes that might have been subsequently incorporated.

3.4 Summary

In this chapter, we have provided the available options for tools/gadgets, which can monitor power consumption on software devices. In addition, the functioning of Intel power gadget along with the procedure adopted for this investigation has also been described.

Chapter 4

Comparative study on energy consumption of ECDH and SIDH

4.1 Preliminaries

Since all the elliptic curve based operations rely mostly on the points chosen on the curve, representation of the points play an important role in the overall performance of the cryptographic scheme. Through energy efficiency analysis, we show that projective coordinate based points are far more efficient than affine coordinate based operations in terms of power and energy consumption. Hence, in order to investigate the energy consumption of ECDH and SIDH, we implement them using standard projective coordinates (also known as projective coordinates).

Elliptic curve point operations are one of the most important steps in both the cryptosystems. The computation of SIDH is adapted from Microsoft's implementation [47]. On the other hand, ECDH is implemented using the regular "double and add" algorithm

[91] for public key and shared secret key generation. Power consumption corresponding to these schemes is then recorded while the code is executed on the above mentioned processor. Energy efficiency is analyzed by considering the cumulative power consumption throughout the execution of these algorithms.

With respect to energy efficiency of ECDH, OpenSSL's implementation [2] has also been considered. Since such implementations have undergone a lot of improvements and optimizations in terms of speed over a long period of time, it doesn't lead to a fair comparison with current SIDH implementations, which is still in the process of getting optimized. Moreover, unlike Microsoft's SIDH implementation, ECDH implementation by OpenSSL uses Jacobian projective coordinates and w-ary Non-adjacent form(w-NAF) based point multiplication. Therefore, in this paper, in order to make a sensible comparison, ECDH is implemented with regular scalar point multiplication using standard projective coordinates on short Weierstrass form of elliptic curve defined on a prime field with primes of size 256 bits for 128 bit classical security level and 384 bits for 192 bit security level.

For the case of SIDH, computations are done on the curves defined over the quadratic extension of a prime field. From this point onwards, in this work we will refer to the Kummer variety of Montgomery curve used for SIDH implementation as C-503^2 and C-751^2 when the length of the prime p_s corresponding to the underlying field $F_{p_s^2}$ is 503 bits and 751 bits respectively.

A slightly different version of this work has been accepted by IEEE TrustCom 2018 [20]. In this thesis all the implementations are built on Linux Subsystem and compiled using gcc 5.4, whereas in the TrustCom paper the implementations were built using Visual Studio 2015. Different compilers have resulted in some changes in figures of the tables, but the trend of overall investigation deductions remain very similar.

4.2 Elliptic curve point addition and doubling

Affine coordinates are the most basic coordinate representation wherein a point P on the curve is comprised of the x and y coordinates that is (x_p, y_p), whereas standard projective coordinates are represented as $P = (X : Y : Z)$ and $Z \neq 0$. The latter can be converted to the affine form as $x = X/Z, y = Y/Z$. In the projective case, since the coordinate is in the form of a ratio, there is no unique way to represent an affine point with projective coordinates. This also leads to some loss in information. However, in projective coordinates, there is no requirement for performing inversions of field elements while performing point doubling or point addition, which helps to reduce the cost of the operations to some extent, as each inversion computation takes more time than multiplication or squaring operations.

4.2.1 Point addition and doubling for ECDH

In ECDH, the short Weierstrass curve (see Equation 2.1) is used. In this section, we provide the formula required to perform elliptic curve point addition and point doubling on this curve. Let the point getting doubled be $P(x_p, y_p)$ and after doubling $Q(x_q, y_q) = 2 \cdot P$. Then (x_q, y_q) can be expressed as follows [64]:

$$\lambda = \frac{3 \cdot x_p^2 + a}{2 \cdot y_p}$$

$$x_q = \lambda^2 - 2 \cdot x_p$$

$$y_q = \lambda \cdot (x_p - x_q) - y_p$$

In case of point addition, let the two distinct input points be $P(x_p, y_p)$ and $Q(x_q, y_q)$. After addition let $R(x_r, y_r) = P + Q$. Then (x_r, y_r) can be computed as follows:

$$\lambda = \frac{y_q - y_p}{x_q - x_p}$$
$$x_r = \lambda^2 - x_p - x_q$$
$$y_r = \lambda \cdot (x_p - x_r) - y_p$$

Next, we present the formulae for point doubling using standard projective coordinates [30] on the same curve. Now while using standard projective coordinates, the elliptic curve which was in the short Weierstrass form can be represented as

$$Y^2 \cdot Z = X^3 + a \cdot X \cdot Z^2 + b \cdot Z^3 \tag{4.1}$$

Let the point getting doubled be $P(X_P, Y_P, Z_P)$ and after doubling be $Q(X_Q, Y_Q, Z_Q) = 2 \cdot P$. Then, (X_Q, Y_Q, Z_Q) can be written as follows [46] :

$$\begin{aligned} X_Q &= 2 \cdot Y_P \cdot Z_P[9 \cdot ({X_P}^2 - {Z_P}^2)^2 - 8 \cdot X_P \cdot {Y_P}^2 \cdot Z_P] \\ Y_Q &= 3 \cdot ({X_P}^2 - {Z_P}^2) \cdot [12 \cdot X_P \cdot {Y_P}^2 \cdot Z_P - \\ &\qquad 9 \cdot ({X_P}^2 - {Z_P}^2)^2] - 8 \cdot {Y_P}^4 \cdot {Z_P}^2 \\ Z_Q &= 8 \cdot {Y_P}^3 \cdot {Z_P}^3 \end{aligned}$$

In case of point addition using projective coordinates, let the points getting added be $P(X_P, Y_P, Z_P)$ and $Q(X_Q, Y_Q, Z_Q)$, after adding $R(X_R, Y_R, Z_R) = P+Q$. Then, (X_R, Y_R, Z_R)

can be expressed as [46]:

$$
\begin{aligned}
U &= Y_Q \cdot Z_P - Y_P \cdot Z_Q \\
V &= X_Q \cdot Z_P - X_P \cdot Z_Q \\
A &= U^2 \cdot Z_P \cdot Z_Q - V^3 - 2 \cdot V^2 \cdot X_P \cdot Z_Q \\
X_R &= V \cdot A \\
Y_R &= U \cdot (V^2 \cdot X_P \cdot Z_Q - A) - V^3 \cdot Y_P \cdot Z_Q \\
Z_R &= V^3 \cdot Z_P \cdot Z_Q
\end{aligned}
$$

4.2.2 Point addition and doubling for SIDH

The supersingular elliptic curve used in the SIDH protocol was defined in Section 2.3 by Equation 2.2. The formulae for point doubling on affine coordinates are given below. Let the point getting doubled be $P(x_p, y_p)$ and after doubling $Q(x_q, y_q) = 2 \cdot P$. The coordinates (x_q, y_q) can be computed as [97] :

$$
\begin{aligned}
\lambda &= \frac{3 \cdot x_p^2 + 2 \cdot A \cdot x_p + 1}{2 \cdot B \cdot y_p} \\
x_q &= B \cdot \lambda^2 - A - 2 \cdot x_p \\
y_q &= (3 \cdot x_p + A) \cdot \lambda - B \cdot \lambda^3 - y_p
\end{aligned}
$$

In case of point addition, the two distinct input points be $P(x_p, y_p)$ and $Q(x_q, y_q)$. After point addition let $R(x_r, y_r) = P + Q$; then (x_r, y_r) can be expressed as [97]:

$$
\begin{aligned}
\lambda &= \frac{y_q - y_p}{x_q - x_p} \\
x_r &= B \cdot \lambda^2 - A - x_p - x_q \\
y_r &= (2 \cdot x_p + x_q + A) \cdot \lambda - B \cdot \lambda^3 - y_p
\end{aligned}
$$

Kummer variety of Montgomery curves were used for the purpose of avoiding computations involving y-coordinates. Therefore, using projective coordinates, let the point getting doubled be $P(X_P, Z_P)$ and after doubling be $Q(X_Q, Z_Q) = 2 \cdot P$. This point doubling algorithm also takes as input two constants A and C, that depend on the curve being used. Then, the resultant coordinates of point Q can be represented as [93]

$$X_Q = C \cdot (X_P - Z_P)^2 \cdot (X_P + Z_P)^2$$

$$Z_Q = 4 \cdot X_P \cdot Z_P \cdot [A \cdot 4 \cdot X_P \cdot Z_P + C \cdot (X_P - Z_P)^2]$$

Let the points getting added be $P(X_P, Z_P)$ and $Q(X_Q, Z_Q)$, after adding $R(X_R, Z_R) = P + Q$. This point addition algorithm takes an extra input that is coordinates of the point $P - Q$ $(X_{P-Q} : 1)$, where the z-coordinate is assumed to be 1. The resultant point R's coordinates can be computed as follows [93] :

$$X_R = [(X_P + Z_P) \cdot (X_Q - Z_Q) + (X_P - Z_P) \cdot (X_Q + Z_Q)]^2$$

$$Z_R = (X_{P-Q}) \cdot [(X_P + Z_P) \cdot (X_Q - Z_Q) - (X_P - Z_P) \cdot (X_Q + Z_Q)]$$

Based on the above given formulae, point addition and doubling are implemented on both the curves. The corresponding results on energy efficiency are presented above in Tables 4.1 and 4.2.

4.3 Effect of coordinate systems on power and energy consumptions

In this section we discuss the power and energy consumption of elliptic curve point additions and point doublings corresponding to the curves used in ECDH and SIDH

Table 4.1: Power and energy consumption using affine and projective coordinates on the ECDH curves P-256 and P-384

Operations	128 bits				192 bits			
	Power(Watts)		Energy(mJoules)		Power(Watts)		Energy(mJoules)	
	Affine	Projective	Affine	Projective	Affine	Projective	Affine	Projective
Doubling	25.43	25.21	8.62	0.21	26.77	26.01	27.31	0.49
Adding	26.55	25.86	8.73	0.17	27.41	25.74	27.95	0.54

Table 4.2: Power and energy consumption using affine and projective coordinates on the SIDH curves C-503^2 and C-751^2

Operations	128 bits				192 bits			
	Power(Watts)		Energy(mJoules)		Power(Watts)		Energy(mJoules)	
	Affine	Projective	Affine	Projective	Affine	Projective	Affine	Projective
Doubling	26.78	25.21	69.09	1.535	27.77	26.38	168.02	2.26
Adding	26.70	26.48	66.21	1.353	27.39	25.17	162.42	2.15

for both the security levels of 128 and 192. The notation used in this section is given in Section 4.1.

4.3.1 Non-supersingular elliptic curves P-256 and P-384

As mentioned earlier, in case of ECDH an elliptic curve in the short Weierstrass form [38] is used, defined by Equation 2.1. Using double and add algorithm, any scalar point multiplication performed on points defined on elliptic curve is basically point addition and point doubling. Table 4.3 provides the comparison on number of operations such as multiplications, inversions and squarings, that are required on F_p for point doubling and point addition using the affine and projective coordinates on the ordinary elliptic curve, to be used in ECDH. Here, p is a prime of size either 256 bits or 384 bits, as recommended in [59]. The number of multiplications refer to only the finite field multiplications involved.

Table 4.3: Number of prime field operations used in affine and standard projective coordinates based point addition and doubling for the ECDH curves.

Instructions	Affine Coordinate		Projective Coordinate	
	Double	Add	Double	Add
Multiplication	2	2	7	12
Squaring	2	1	3	2
Addition	4	0	11	1
Subtraction	4	6	5	6
Inversion	1	1	0	0

Inversions are computed on the prime field by using Fermat's Little Theorem [6]. Since affine coordinates involve inversions, they are naturally slower, which gets reflected in their overall energy consumption as well (refer to Table 4.1). It can be seen that affine coordinates requires marginally more power for all its computations when compared to projective coordinate based computations. In terms of energy, it requires on average around 45 to 55 times more energy than what projective coordinate based computations consume over the curves P-256 and P-384. These figures of energy consumption are mostly affected by the clock cycles required for the respective computations.

4.3.2 Supersingular elliptic curves C-503^2 and C-751^2

All the computations are performed on points over Montgomery curves as used in SIDH scheme (refer to Equation 2.3). Affine points on elliptic curves are represented as $P = (x, y)$ where each x and y coordinate is an element of $F_{p_s^2}$. Similarly projective Kummer coordinates [47] are represented as $(X : Z)$, where each of the coordinate is an element of $F_{p_s^2}$.

Let two elements on $F_{p_s^2}$ be $M = m_0 + i \cdot m_1$ and $N = n_0 + i \cdot n_1$, where $m_0, m_1, n_0, n_1 \in F_{p_s}$ and $i^2 + 1 = 0$. Finite field operations involving these elements in F_{p_s} are as follows

(a) Addition/Subtraction of M and N is

$$\begin{aligned} M + N &= (m_0 + i \cdot m_1) \pm (n_0 + i \cdot n_1) \\ &= (m_0 \pm n_0) + i \cdot (m_1 \pm n_1) \end{aligned}$$

i.e., it requires two additions/subtractions in F_{p_s}.

(b) Multiplication of M and N (using the Karatsuba scheme [74]) can be performed as

$$\begin{aligned} M \cdot N &= (m_0 + i \cdot m_1) \cdot (n_0 + i \cdot n_1) \\ &= (m_0 \cdot n_0 - m_1 \cdot n_1) + \\ &\quad i \cdot ((m_0 + m_1) \cdot (n_0 + n_1) - m_0 \cdot n_0 - m_1 \cdot n_1) \end{aligned}$$

requiring three multiplications, three additions (one of size $2|p_s|$ and the other two of size $|p_s|$ each, where $|x|$ is the bit-length of x) and two subtractions of size $2|p_s|$ on F_{p_s}.

(c) Squaring M on F_{p_s} will be done as follows

$$\begin{aligned} M^2 &= (m_0 + i \cdot m_1)^2 \\ &= (m_0 + m_1) \cdot (m_0 - m_1) + i \cdot (2 \cdot m_0 \cdot m_1) \end{aligned}$$

It requires two multiplications, two additions (one of size $2|p_s|$ and the other of size $|p_s|$) and one subtraction of size $|p_s|$.

(d) Inverting M can be done as follows

$$M^{-1} = m_0 \cdot ({m_0}^2 + {m_1}^2)^{-1} - i \cdot m_1 \cdot ({m_0}^2 + {m_1}^2)^{-1}$$

So, inversion of an element of $F_{p_s^2}$ on F_{p_s}, requires one inversion, two multiplications, two squarings and one addition of size $2|p_s|$.

In other words, if the operations on the field $F_{p_s^2}$ are translated to the field F_{p_s} they turn out to be as shown below in Table 4.4.

Table 4.4: Number of prime field operations used in affine and projective coordinates based point addition and doubling for the SIDH curve

Instructions	Affine Coordinate		Projective Coordinate	
	Double	Add	Double	Add
Multiplication	27	19	16	13
Squaring	2	2	0	0
Addition	46	26	20	19
Subtraction	24	31	14	14
Inversion	1	1	0	0

Since the comparisons are done with ECDH only point doubling and addition is focused in this paper. However, point tripling (not generally required in ECDH) is also an important and time consuming operation in SIDH, which was optimized in [58]. Table 4.2 provides the power and energy consumed for these operations. Here also it can be seen that energy consumption in elliptic curve point operations using affine coordinates is around 40 to 65 times more compared to that using projective coordinates on the curves of C-503^2 and C-751^2. So, Table 4.1 and Table 4.2's data further validates the energy efficiency of standard projective coordinates compared to affine coordinates.

4.4 Energy consumption of ECDH and SIDH

As per the deductions in Section 4.3, in order to achieve better energy efficiency, all the implementations of the algorithms in both SIDH and ECDH are done using standard projective coordinates. The basic elliptic curve point operations use the same formulae as

Table 4.5: Comparison of power (in Watts) and energy (in milliJoules) consumption between ECDH and SIDH for 128 and 192 bit security levels

Cryptographic operations	128 bits				192 bits			
	ECDH		SIDH		ECDH		SIDH	
	Power	Energy	Power	Energy	Power	Energy	Power	Energy
Alice's Public Key	25.19	63.02	26.83	2796.6	25.87	175.64	27.23	10177.93
Bob's Public Key	25.01	62.38	26.33	3101.1	26.42	176.83	26.97	10141.16
Alice's Shared Secret	25.04	63.13	26.68	2667.42	26.06	176.62	27.26	8892.72
Bob's Shared Secret	25.23	61.94	26.41	2510.46	25.86	176.26	27.33	9307.5

mentioned in Section 4.2. Table 4.5 compares the energy and power consumption between the cryptographic schemes. SIDH consumes slightly more power than ECDH for both the cases. However the energy consumption between them has a huge difference. The comparison has been done at the four major steps of key exchange that is initial key generation by both the parties, followed by the generation of shared key secret.

The aspect of energy consumption is dependent on both the power and time of execution. SIDH consumes a lot more energy than ECDH, where mostly the execution time corresponding to the individual steps in the schemes influence this difference. The isogeny evaluation and computation is one of the most time consuming steps. Table 4.6 provides the ratio of average power and energy used by SIDH cryptosystem to the ECDH cryptosystem. The comparison is done here on the basis of energy consumed by each operation in both ECDH and SIDH. An ECDH operation refers to a public key generation or a shared secret key generation at any of the node (Alice or Bob), involved in the secured communication. In case of ECDH, it is a point multiplication which involves a series of point doubling and point additions. ECDH operations at both the ends of Bob and Alice

Table 4.6: Ratios of power and energy consumption for each SIDH operation compared to each ECDH operation

Bit Security	SIDH/ECDH	
	Power	Energy
128	1.05	44.22
192	1.07	54.61

depend upon the private key which is being used for the point multiplication. Mostly, the time and energy consumed for such public key generation is similar at both the ends.

However in SIDH, when Alice and Bob compute their public keys, evaluation and computation of isogenies of different degrees are involved. In this particular implementation Alice computes isogenies using a kernel, generated by a point of order 2 on the supersingular elliptic curve while Bob uses a kernel, generated by a point of order 3 on the same curve. As mentioned before the prime used in this scheme is of the form $p = {l_A}^{e_A} \cdot {l_B}^{e_B} - 1$. Therefore, Alice evaluates isogenies of degree ${l_A}^{e_A}$ and Bob evaluates isogenies of degree ${l_B}^{e_B}$ respectively. So e_A isogenies of degree l_A and e_B isogenies of degree l_B are computed. The finite field operations involved in this isogeny computations of different degrees are also different. As a result unlike ECDH operations, Alice and Bob end up requiring different amounts of time and energy in their public key and shared secret key generation as shown in Table 4.5. In the field $F_{p_s^2}$ for the above mentioned isogeny computations [47], [69], Alice computes 13 multiplications and 8 squarings whereas Bob computes 9 multiplications and 5 squarings only.

4.5 Battery exhaustion experiment

A practical experiment was performed to determine the number of SIDH and ECDH operations that can be performed on a laptop until its battery gets exhausted. We have used an HP Pavilion Notebook with the specification of the processor as given in Section 3.3, with a battery capacity of 60 Watt hours or 216 KJoules. Since the public key generation and shared secret key generation in SIDH require different computations, involving slightly different execution times, an average computation time is considered as each SIDH operation. On average, the number of ECDH and SIDH operations that could be performed until the battery power of the laptop was exhausted are around 3421000 and 84300 for 128 bits security. This finding is consistent with the Intel power gadget based results reported in the previous section. Also as the bit security level changes from 128 to 192, the number of operations in either ECDH or SIDH that could be performed decreases by around one third.

4.6 Summary

In this chapter, we have presented the energy efficiency of standard projective coordinate based elliptic curve computations compared to affine coordinates based similar computations. Standard projective coordinates have been found to be significantly more energy efficient than the affine coordinates. We have also reported energy consumption of SIDH and ECDH implementations. Our results indicate that SIDH consumes about 45 to 55 times more energy than ECDH. Our findings also suggest that SIDH based on C-751^2 will consume three to four times more energy than the one based on C-503^2. We should however note that, while SIDH is considered quantum-safe, ECDH is not. In addition to the relative energy (in)efficiency of SIDH compared to ECDH, we have reported their actual energy consumption values. To the best of our knowledge this is the first time that such values are reported for SIDH. These values could be an important consideration in the mode of deployment of SIDH in battery operated or energy constrained systems such as hand-held devices, remote sensors and space satellites.

Chapter 5

Energy consumption of NIST PQC candidate algorithms

5.1 Overview

Quantum-safe cryptographic schemes are algorithms that are secure against attacks by both classical and quantum computers. In recent years a lot of research has been done on post-quantum cryptography. The motive behind such research is that, if large scale quantum computers become a reality, then current cryptographic algorithms would require replacement by quantum-safe cryptosystems. This is because quantum computers would completely break all public-key cryptosystems in use today, namely RSA, DSA [79], and elliptic curve cryptosystems. Therefore, before this situation turns more critical, NIST has started the process of developing standards for post-quantum cryptography [42]. Currently there are quite a few post-quantum cryptographic schemes such as lattice-based, code-based, multivariate-based, hash-based cryptosystems etc. The new post-quantum cryptography standards will be used as quantum resistant counterparts to existing standards, including digital signature schemes specified in Federal Information Processing Standards Publication (FIPS) 186 and key establishment schemes specified in NIST Special Publications (SP) 800-56 A and B. Furthermore, this process would also help in the transition from usage of public key cryptosystems to post-quantum cryptosystems, before quantum computers become a reality.

At present, very few of the algorithms have been implemented in hardware. Mostly all the candidates have presented their software implementations suitable for execution on 64 bit processors. Among the 69 initial submissions, around 22 were co-designed by the PQCRYPTO group [68]. Some of those submissions also have implementations for lower end processors such as ARM Cortex-M4. Since we were looking for a comprehensive report based on all these submissions, only software implementations on 64 bit processors are considered in this thesis.

All the submissions to the NIST post-quantum standardization process are available for public scrutiny and are being evaluated based on security, performance and other properties by various stakeholders including the cryptographic community. Although not explicitly part of the evaluation criteria, energy consumption due to the execution of cryptographic algorithms is a very important consideration for battery operated devices such as mobile phones and sensors [80], [102]. If an algorithm's energy consumption on a certain platform is known, then one can easily estimate how many times the algorithm can be executed on the platform before its battery is completely exhausted, providing an added aspect to be considered while deciding the deployment of the algorithm in energy constrained environments. Therefore, the idea of this investigation is to measure the energy efficiency of each of these candidate algorithms. All submissions are available on the NIST website [1] and include detailed description of the proposed algorithms along with reference to relevant articles. For brevity, overviews of those algorithms are not provided in this thesis.

All the implemented algorithms are executed for 100 iterations to measure their execution time and also record their average power usage. Energy consumption is computed using the execution time and average power usage data. In all the tables in this chapter, execution time is reported in milliseconds and energy consumption is reported in milliJoules.

5.2 Work process and methodology

According to the criteria set by NIST, there are broadly three different kinds of submissions:

- Public key signatures
- Public key encryption
- Key encapsulation mechanism

In signature schemes the subroutines that are benchmarked, their definitions are given below:

```
int crypto_sign_keypair(unsigned char *pk, unsigned char *sk)

int crypto_sign(unsigned char *sm, unsigned long long *smlen,
const unsigned char *m, unsigned long long mlen,
const unsigned char *sk)
```

```
int crypto_sign_open(unsigned char *m, unsigned long long *mlen,
const unsigned char *sm, unsigned long long smlen,
const unsigned char *pk)
```

They are responsible for private and public key pair generation, signing of message and verification of the signature. Similarly, public key encryption schemes are supposed to include key pair generation, encryption of the message to generate ciphertext and then decryption of the ciphertext :

```
int crypto_encrypt_keypair(unsigned char *pk, unsigned char *sk)

int crypto_encrypt(unsigned char *c, unsigned long long *clen,
const unsigned char *m, unsigned long long mlen,
const unsigned char *pk)

int crypto_encrypt_open(unsigned char *m, unsigned long long
*mlen, const unsigned char *c, unsigned long long
clen, const unsigned char *sk)
```

Lastly, key encapsulation schemes are comprised of key pair generation, encapsulation of the message and finally decapsulation:

```
int crypto_kem_keypair(unsigned char *pk, unsigned char *sk)

int crypto_kem_enc(unsigned char *ct, unsigned char *ss,
const unsigned char *pk)

int crypto_kem_dec(unsigned char *ss, const unsigned char *ct,
```

```
const unsigned char *sk)
```

NIST has also recommended some guidelines and format for these subroutines such as additional functions that are to be used in these subroutines. The key pair generation subroutines require random input generation which is done using the SUPERCOP package [25]. In this study when these subroutines for key generation, encryption, encapsulation, signing etc. are benchmarked on the above mentioned processor, the whole subroutine's power usage and execution time is used to report the energy consumption. That is, in the power usage and energy consumption results, the subroutine for random number generation's contribution is included. Moreover, NIST has provided its classification on the range of security strengths offered by the existing NIST standards in symmetric cryptography, which is expected to offer significant resistance to quantum cryptanalysis. Five main security levels [42] have been provided for this purpose as follows :

- Level I: It should be at least as hard as that of breaking the security of block ciphers using exhaustive key search with 128 bit key, for example AES 128.

- Level II: It should be at least as hard as that of breaking the security of hash functions using collision search with 256 bit hashed message digest, for example SHA256/ SHA3-256

- Level III: It should be at least as hard as that of breaking the security of block ciphers using exhaustive key search with 192 bit key, for example AES 192.

- Level IV: It should be at least as hard as that of breaking the security of hash functions using collision search with 384 bit hashed message digest, for example SHA384/ SHA3-384

- Level V: It should be at least as hard as that of breaking the security of block ciphers using exhaustive key search with 256 bit key, for example AES 256.

Now, not all the candidate algorithms have implementations corresponding to the five above mentioned security levels. In order to make a fair comparison of the schemes on the basis of their power usage and energy consumption, they are grouped into different categories that is encapsulation/encryption or signature and also in different security levels according to availability in the submissions, as shown in the next section. It should be noted that different schemes use different lengths of message according to their structure. Also in signature scheme's execution time and power usage depends on the length of the message being signed. For the purpose of a consistent comparison, the largest message block size mentioned in the supporting documentation, is considered during the benchmarking of the signature scheme codes and also for its corresponding energy consumption.

The implementations submitted in this event include C codes as well as vectorised codes. Again not all submissions have provided vectorised instructions for speed ups. Therefore, in order to make a reasonable comparison, the "optimized implementation" of all the submissions are considered which only includes C implementation without any vectorization. Some of the encryption/encapsulation submissions have provided implementations which are secure specifically against chosen ciphertext attack or chosen plaintext attack. These schemes are separately evaluated based on their security against the attacks as shown in the tables below. Also quite a few of the submissions have provided both encryption and encapsulation algorithms, hence it can be seen that their submission names are repeated in the tables for encapsulation and encryption. It should be noted that in few of the submissions, there are algorithms with the same security levels but different probability of error in decryption or verification etc. For such submissions, the algorithm with the least probability of error is considered in this work.

5.3 Energy efficiency

5.3.1 Public key signatures

Amongst the sixty four valid candidate algorithms, nineteen schemes include signing and verification schemes as shown below in Tables 5.1, 5.2 and 5.3. These tables provide the energy consumption of the algorithms when they were executed on a 64 bit processor laptop (Intel 6700 Skylake). We have observed that their power usages do not vary much across all these schemes, and is generally around 24-28 Watts. The energy consumption of the implementations is mostly influenced by their execution times. Some submissions such as pqNTRUSign [41], SPHINCS Plus [28], Walnut [11], have provided multiple variants of the same algorithm using different parameters. This report provides the energy efficiency for all those variants as well. For a particular security level, amongst all the algorithms submitted in the categories of signing, encryption or encapsulation etc., the most energy efficient and the least ones are in bold characters. There are few submissions of both signature schemes and encryption/encapuslation techniques, that have provided implementation for the II and IV security levels. Therefore, we did not mark the most energy efficient or the least ones in those categories. It should also be noted that there are instances where multiple algorithms require almost similar execution time. This leads to energy consumption values that are quite close, depending also upon their power usage values. In that case we have provided the top five efficient algorithms in Tables 5.11, 5.10 and 5.12 with comparable energy consumption values.

Table 5.1: Energy consumption during **key generation of public key signature schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
CRYSTALSDilithium [54]	24.98	0.04	0.99	25.17	0.06	1.51	25.21	0.09	2.26	24.6	0.12	2.96	-	-	-
DRS [100]	25.34	452.02	11454. 18	-	-	-	25.77	454.12	11702.67	-	-	-	25.61	456.78	11698.13
DualModeMS [57]	26.41	698131	**18437639.71**	-	-	-	-	-	-	-	-	-	-	-	-
FALCON [60]	25.32	6.29	159.26				26.05	11.6	302.18				25.12	19.04	497.32
GeMSS [37]	26.67	33.43	891.57	-	-	-	25.93	142.34	3690.87	-	-	-	26.18	358.94	9397.04
Gravity SPHINCS [72]	-	-	-	26.57	388.23	10315.27	-	-	-	-	-	-	-	-	-
Gui [98]	26.34	623	16409.82	-	-	-	26.12	25337	**661802.44**	-	-	-	25.7	92346	**2373292.2**
HiMQ3 [107]	24.88	0.02	0.49	-	-	-	-	-	-	-	-	-	-	-	-
HiMQ3F	24.78	0.03	0.74	-	-	-	-	-	-	-	-	-	-	-	-
LUOV [31]	-	-	-	26.67	7	186.69	-	-	-	26.55	31.2	828.36	26.93	57.8	1556.554
MQDSS [43]	-	-	-	26.12	0.85	22.2	-	-	-	26.62	1.97	52.44	-	-	-
pqNTRUSign Gaussian	-	-	-	-	-	-	-	-	-	-	-	-	27.13	48.75	1322.58
pqNTRUSign Uniform	-	-	-	-	-	-	-	-	-	-	-	-	27.05	47.34	1280.54
Picnic-FS [40]	25.67	0.005	0.13	-	-	-	25.92	0.016	**0.41**	-	-	-	25.34	0.032	**0.81**
Picnic-UR	27.05	0.004	**0.1**	-	-	-	26.93	0.017	0.46	-	-	-	27.13	0.04	1.08
Post-Quantum RSA Sign [29]	-	-	-	27.45	1350.26	3706463.7	-	-	-	-	-	-	-	-	-
pqsigRM [81]	26.78	5260	140862.8	-	-	-	26.79	1026.17	27491.09	-	-	-	27.1	13553.2	367291.72
qTESLA [10]	26.85	0.94	25.23	-	-	-	26.66	1.39	37.05	-	-	-	27.11	2.94	79.7
RaCoSS [94]	25.74	200.4	5158.296	-	-	-	-	-	-	-	-	-	-	-	-
Rainbow [52]	27.13	367.33	9965.66	26.97	1449.09	39081.95	27.43	21248.7	582851.84	27.11	13801.8	374166.79	27.52	47220.97	1299521.09
SPHINCS Plus(SHA256F) [28]	26.38	2.75	72.545	-	-	-	26.86	4.99	134.03	-	-	-	27.11	18.76	508.58
SPHINCS Plus(SHA256S)	25.99	84.43	2194.33	-	-	-	26.32	163.73	4309.37	-	-	-	27.05	299.53	8102.28
SPHINCS Plus(SHAKE256F)	27.36	5.28	144.46	-	-	-	27.84	7.87	219.1	-	-	-	27.33	22.64	618.75
SPHINCS Plus(SHAKE256S)	26.98	171.35	4623.02	-	-	-	27.02	250.7	6773.91	-	-	-	26.94	320.33	8629.69
Walnut BKL [11]	26.53	0.27	7.16	-	-	-	26.67	0.6	16	-	-	-	-	-	-
Walnut StochasticWrite	26.45	0.27	7.14	-	-	-	26.92	0.6	16.15	-	-	-	-	-	-
Walnut Dehornoy	25.81	0.27	6.96	-	-	-	26.32	0.63	16.58	-	-	-	-	-	-

Table 5.2: Energy consumption during **signing of public key signature schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
CRYSTALSDilithium	26.78	0.18	4.82	27.32	0.3	8.19	26.89	0.42	11.29	25.12	0.41	10.29	-	-	-
DRS	27.34	22.9	626	-	-	-	26.81	25.51	683.91	-	-	-	27.11	26.28	712.45
DualModeMS	26.27	1846	48494.42	-	-	-	-	-	-	-	-	-	-	-	-
FALCON	26.73	0.145	3.87	-	-	-	26.88	0.23	**6.18**	-	-	-	27.01	0.28	**7.56**
GeMSS	26.79	318.96	8544.93	-	-	-	26.52	729.75	19352.97	-	-	-	27.18	1106.32	30069.77
Gravity SPHINCS	-	-	-	26.73	1.68	44.9	-	-	-	-	-	-	-	-	-
Gui	25.83	31.4	811.06	-	-	-	26.12	11343	**296279.16**	-	-	-	26.78	474589	**12709493.42**
HiMQ3	25.87	0.012	**0.31**	-	-	-	-	-	-	-	-	-	-	-	-
HiMQ3F	25.02	0.035	0.87	-	-	-	-	-	-	-	-	-	-	-	-
LUOV	-	-	-	25.88	26.8	693.58	-	-	-	26.11	80.6	2104.46	26.32	163.3	4298.05
MQDSS	-	-	-	26.33	70.36	1852.57	-	-	-	26.48	222.43	5889.94	-	-	-
pqNTRUSign Gaussian	-	-	-	-	-	-	-	-	-	-	-	-	26.65	107.81	2873.13
pqNTRUSign Uniform	-	-	-	-	-	-	-	-	-	-	-	-	26.94	63.59	1713.11
Picnic-FS	27.54	3.2	88.12	-	-	-	26.33	12.38	325.96	-	-	-	26.94	47.25	1272.91
Picnic-UR	26.5	4.2	111.3	-	-	-	26.78	16.2	433.83	-	-	-	27.11	50.34	1364.71
Post-Quantum RSA Sign	-	-	-	28.01	43.42	1216.19	-	-	-	-	-	-	-	-	-
pqsigRM	26.46	25684.8	**679619.80**	-	-	-	26.53	1846.5	49462.5	-	-	-	26.82	1754.8	47063.73
qTESLA	26.39	0.62	16.36	-	-	-	25.94	3.59	93.12	-	-	-	26.07	6.79	177.01
RaCoSS	26.26	10.15	266.53	-	-	-	-	-	-	-	-	-	-	-	-
Rainbow	25.72	0.21	5.4	26.01	0.53	13.78	25.97	3.2	83.1	26.14	2.31	60.38	26.1	3.87	101
SPHINCS Plus(SHA256F)	26.89	86.91	2337	-	-	-	26.2	137.33	3598.04	-	-	-	25.72	426.53	10970.35
SPHINCS Plus(SHA256S)	27.04	1298.11	35100.89	-	-	-	26.79	3527.1	94491	-	-	-	26.78	3641.14	97509.73
SPHINCS Plus(SHAKE256F)	28.06	153.93	4319.27	-	-	-	27.96	208.62	5833.01	-	-	-	28.12	512.84	14421.06
SPHINCS Plus(SHAKE256S)	25.74	2399.61	61765.96	-	-	-	26.14	5057.47	132202.26	-	-	-	26.37	3627.23	95650.055
Walnut BKL	27.15	20.19	548.15	-	-	-	26.73	69.71	1863.34	-	-	-	-	-	-
Walnut StochasticWrite	26.93	9.56	257.45	-	-	-	27.18	25.47	692.27	-	-	-	-	-	-
Walnut Dehornoy	27.43	9.1	249.61	-	-	-	26.87	24.4	655.62	-	-	-	-	-	-

Table 5.3: Energy consumption during **verification of public key signature schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
CRYSTALSDilithium	26.13	0.04	1.04	25.97	0.07	1.81	25.38	0.10	2.53	24.67	0.13	3.2	-	-	-
DRS	26.34	222.71	5866.18	-	-	-	26.17	224.68	**5879.87**	-	-	-	25.69	226.95	**5830.34**
DualModeMS	26.36	1913	**50426.68**	-	-	-	-	-	-	-	-	-	-	-	-
FALCON	25.83	0.025	**0.64**	-	-	-	26.13	0.044	**1.15**	-	-	-	26.93	0.052	**1.4**
GeMSS	27.19	0.067	1.82	-	-	-	26.82	0.143	3.83	-	-	-	26.87	0.394	10.58
Gravity SPHINCS	-	-	-	26.69	0.01	0.26	-	-	-	-	-	-	-	-	-
Gui	27.25	0.045	1.23	-	-	-	26.88	0.347	9.32	-	-	-	27.12	0.689	18.68
HiMQ3	26.82	0.075	2.01	-	-	-	-	-	-	-	-	-	-	-	-
HiMQ3F	24.89	0.087	2.16	-	-	-	-	-	-	-	-	-	-	-	-
LUOV	-	-	-	25.93	16.5	427.84	-	-	-	26.05	44.5	1159.22	26.31	83.9	2207.4
MQDSS	-	-	-	27.11	52.35	1419.2	-	-	-	26.98	167.18	4510.51	-	-	-
pqNTRUSign Gaussian	-	-	-	-	-	-	-	-	-	-	-	-	27.11	1.25	33.88
pqNTRUSign Uniform	-	-	-	-	-	-	-	-	-	-	-	-	26.45	1.87	49.46
Picnic-FS	26.11	2.2	57.44	-	-	-	25.67	8.34	214.08	-	-	-	26.06	30.9	805.25
Picnic-UR	27.43	3.11	85.3	-	-	-	26.97	11.36	306.37	-	-	-	27.05	34.64	937.01
Post-Quantum RSA Sign	-	-	-	28.05	5.78	162.13	-	-	-	-	-	-	-	-	-
pqsigRM	26.12	81.1	2118.33	-	-	-	26.45	58.57	1549.17	-	-	-	26.67	298.92	7972.19
qTESLA	27.32	0.12	3.28	-	-	-	27.26	0.25	6.81	-	-	-	26.96	0.32	8.63
RaCoSS	26.58	9.86	262.07	-	-	-	-	-	-	-	-	-	-	-	-
Rainbow	26.13	0.11	2.87	26.45	0.43	11.37	26.82	3.1	83.14	26.23	1.52	39.86	26.71	3.28	87.6
SPHINCS Plus(SHA256F)	26.63	3.65	97.2	-	-	-	26.41	7.37	194.64	-	-	-	25.89	10.57	273.65
SPHINCS Plus(SHA256S)	27.11	1.44	39.03	-	-	-	26.94	2.92	78.66	-	-	-	27.04	5.54	149.8
SPHINCS Plus(SHAKE256F)	25.34	6.57	166.48	-	-	-	26.08	11.2	292.09	-	-	-	26.12	12.2	318.66
SPHINCS Plus(SHAKE256S)	26.87	3.04	81.68	-	-	-	26.31	4.45	117.07	-	-	-	25.89	5.3	137.21
Walnut BKL	26.42	0.22	5.81	-	-	-	26.78	0.77	20.62	-	-	-	-	-	-
Walnut StochasticWrite	27.02	0.11	2.97	-	-	-	27.31	0.31	8.46	-	-	-	-	-	-
Walnut Dehornoy	26.91	0.12	3.23	-	-	-	27.33	0.35	9.56	-	-	-	-	-	-

Table 5.4: Energy consumption during **key pair generation of public key encryption schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
Compact LWE [82]	-	-	-	-	-	-	26.93	0.163	4.39	-	-	-	-	-	-
GiophantusR [5]	26.34	12.14	319.76	-	-	-	26.78	22.03	589.96	-	-	-	27.04	32.16	869.6
Guess Again [108]	-	-	-	-	-	-	-	-	-	-	-	-	27.67	38.7	1070.82
AKCN MLWE [122]	-	-	-	-	-	-	-	-	-	26.89	0.153	4.11	-	-	-
KINDI-ENCRYPT [23]	-	-	-	-	-	-	26.53	0.07	**1.85**	-	-	-	26.71	0.16	4.27
LAC [84]	-	-	-	28.1	0.026	0.73	-	-	-	27.88	0.085	2.36	27.87	0.088	**2.45**
LEDA PKC [18]	26.89	16.66	447.98	-	-	-	26.34	70.31	1851.96	-	-	-	26.88	201.562	5417.93
LIMA CCA [87]	27.17	0.42	11.41	27.03	0.77	20.81	27.56	0.86	23.7	-	-	-	27.45	1.53	41.99
LIMA CPA	26.94	0.42	11.31	27.04	0.77	20.82	26.76	0.86	23.01	-	-	-	27.02	1.53	41.34
Lizard CCA [44]	26.54	10.78	286.1	-	-	-	26.98	24.06	649.13	-	-	-	27.05	42.81	1158.01
RLizard CCA	26.78	0.04	**1.07**	-	-	-	26.93	0.08	2.15	-	-	-	27.04	0.1	2.7
LOTUS Encrypt [99]	27.09	9.79	265.21	-	-	-	26.63	18.91	503.57	-	-	-	27.15	26.34	715.13
McNIE 3Q [61]	27.17	109.1	**2964.24**	-	-	-	27.52	193.2	**5316.86**	-	-	-	28.2	336.78	**9497.2**
McNIE 4Q	26.89	95.02	2555.08	-	-	-	27.34	166.32	4547.18	-	-	-	27.97	336.72	9418.05
NTRUEncrypt PKE [66]	26.72	0.33	8.81	-	-	-	25.94	1.04	26.97	-	-	-	26.37	39.58	1043.72
Round2-u Encrypt [16]	27.14	0.25	6.78	26.88	0.42	11.29	27.07	0.58	15.7	27.45	0.6	16.47	27.62	0.62	17.12
Round2-n Encrypt	28.32	2.58	73.06	28.05	3.76	105.46	27.96	4.02	112.4	27.59	6.06	167.19	28.32	8.34	236.18
Titanium CPA [112]	27.14	0.61	16.55	-	-	-	27.39	0.6	16.43	-	-	-	27.26	0.85	23.17

Table 5.5: Energy consumption during **key pair encryption of public key encryption schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
Compact LWE	-	-	-	-	-	-	26.44	2.87	75.88	-	-	-	-	-	-
GiophantusR	26.36	22.01	**580.18**	-	-	-	26.4	49.88	**1316.83**	-	-	-	26.43	78.99	2097.7
Guess Again	-	-	-	-	-	-	-	-	-	-	-	-	27.06	2634	**71276.04**
AKCN MLWE	-	-	-	-	-	-	-	-	-	25.49	0.38	9.68	-	-	-
KINDI-ENCRYPT	-	-	-	-	-	-	27.11	0.09	2.43	-	-	-	26.68	0.2	5.33
LAC	-	-	-	26.59	0.04	1.28	-	-	-	26.83	0.13	3.38	26.4	0.16	4.32
LEDA PKC	27.08	4.68	126.73	-	-	-	26.95	15.1	406.94	-	-	-	27.34	40.1	1096.33
LIMA CCA	26.33	0.37	9.74	26.67	0.68	18.13	26.7	0.75	20.02	-	-	-	26.73	1.41	37.68
LIMA CPA	27.53	0.38	10.46	27.74	0.69	19.14	27.56	0.77	21.22	-	-	-	27.5	1.4	38.5
Lizard CCA	26.89	0.02	**0.54**	-	-	-	27.45	0.048	**1.31**	-	-	-	27.32	0.07	**1.91**
RLizard CCA	27.16	0.02	**0.54**	-	-	-	27.2	0.05	1.36	-	-	-	27.35	0.07	**1.91**
LOTUS Encrypt	26.12	0.08	2.09	-	-	-	26.31	0.11	2.89	-	-	-	27.07	0.19	5.14
McNIE 3Q	26.81	1.03	27.61	-	-	-	26.53	2.09	55.44	-	-	-	26.71	3.12	83.33
McNIE 4Q	27.13	0.12	3.25	-	-	-	26.97	1.54	41.53	-	-	-	27.02	3.32	89.7
NTRUEncrypt PKE	26.86	0.06	1.61	-	-	-		0.09	2.41	-	-	-	26.49	61.66	1633.37
Round2-u Encrypt	27.68	0.31	8.58	27.21	0.52	14.15	27.32	0.69	18.85	27.16	0.74	20.09	27.54	0.77	21.2
Round2-n Encrypt	26.88	5.37	144.34	26.74	7.87	210.44	26.85	8.6	230.91	26.94	13.98	376.62	27.03	12.21	330.03
Titanium CPA	26.85	0.56	15.036	-	-	-	26.92	0.56	15.07	-	-	-	26.86	0.82	22.02

Table 5.6: Energy consumption during **key pair decryption of public key encryption schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
Compact LWE	-	-	-	-	-	-	26.59	0.35	9.3	-	-	-	-	-	-
GiophantusR	26.56	41.31	**1097.2**	-	-	-	26.73	94.37	**2522.51**	-	-	-	26.83	151.34	4060.45
Guess Again	-	-	-	-	-	-	-	-	-	-	-	-	27.23	1.38	37.57
AKCN MLWE	-	-	-	-	-	-	-	-	-	25.78	0.451	11.6	-	-	-
KINDI-ENCRYPT	-	-	-	-	-	-	26.83	0.11	2.95	-	-	-	25.94	0.25	6.48
LAC	-	-	-	27.12	0.03	0.76	-	-	-	26.94	0.096	2.58		27.57 0.104	2.8
LEDA PKC	27.49	28.12	773	-	-	-	27.56	61.979	1708.14	-	-	-	27.72	167.18	**4634.22**
LIMA CCA	26.75	0.47	12.57	26.84	0.9	24.16	26.92	0.96	25.84	-	-	-	27.03	1.84	49.73
LIMA CPA	27.34	0.125	3.41	27.42	0.22	6.03	27.38	0.23	6.3	-	-	-	26.85	0.45	12.08
Lizard CCA	27.12	0.03	**0.81**	-	-	-	26.98	0.06	**1.51**	-	-	-	27.14	0.09	**2.44**
RLizard CCA	27.24	0.03	0.82	-	-	-	27.35	0.07	1.91	-	-	-	27.32	0.1	2.73
LOTUS Encrypt	26.43	0.13	3.43	-	-	-	26.37	0.24	6.32	-	-	-	26.61	0.41	10.91
McNIE 3Q	26.73	2.02	53.99	-	-	-	26.92	3.04	81.83	-	-	-	27.05	5.11	138.22
McNIE 4Q	27.32	1.05	28.68	-	-	-	27.35	2.04	55.79	-	-	-	27.29	5.04	137.54
NTRUEncrypt PKE	26.12	0.07	1.83	-	-	-	26.31	0.2	5.26	-	-	-	26.55	104.58	2776.59
Round2-u Encrypt	27.58	0.06	1.65	27.67	0.08	2.21	27.7	0.08	2.21	28.22	0.09	2.54	28.14	0.11	3.09
Round2-n Encrypt	26.89	8.1	217.8	26.92	11.96	321.96	27.02	12.67	342.34	27.56	19.95	549.82	27.45	19.1	524.3
Titanium CPA	27.13	0.09	2.44	-	-	-	27.31	0.1	2.73	-	-	-	26.98	0.15	4.05

Table 5.7: Energy consumption during **key pair generation of public key encapsulation schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
BIGQUAKE [24]	26.36	301	7934.36	-	-	-	26.48	2754	**72925.92**	-	-	-	26.44	5171	**136721.24**
BIKE [12]	25.88	0.24	6.21	-	-	-	26.08	5.81	151.52	-	-	-	25.97	0.64	16.62
CFPKM [39]	26.71	183	4887.93	-	-	-	26.53	490	12999.7	-	-	-	-	-	-
Classic McEliece [26]	-	-	-	-	-	-	-	-	-	-	-	-	27.59	936.11	25827.27
CRYSTALSKyber [35]	26.34	0.15	3.95	-	-	-	25.98	0.255	6.62	-	-	-	25.58	0.37	9.46
DAGS [19]	-	-	-	-	-	-	26.43	11.35	299.98	-	-	-	26.82	107.73	2889.31
DING [53]	27.17	1.42	38.58	-	-	-	-	-	-	-	-	-	26.98	2.77	74.73
DME [85]	-	-	-	-	-	-	25.72	25.79	663.31	-	-	-	25.82	95.51	2466.06
EMBLEM [106]	24.97	0.039	0.97	-	-	-	-	-	-	-	-	-	-	-	-
FRODO [95]	26.13	0.373	9.74	-	-	-	26.57	0.745	19.79	-	-	-	-	-	-
Hila5 [104]	-	-	-	-	-	-	-	-	-	-	-	-	27.05	1.29	34.89
HQC [89]	26.63	0.16	4.26	-	-	-	26.51	0.53	14.05	-	-	-	26.32	0.68	17.89
AKCN MLWE	-	-	-	-	-	-	-	-	-	26.71	0.1	2.67	-	-	-
OKCN MLWE	-	-	-	-	-	-	-	-	-	26.58	0.1	2.65	-	-	-
OKCN SEC	-	-	-	-	-	-	-	-	-		-	-	25.93	0.13	3.37
AKCN SEC	-	-	-	-	-	-	-	-	-		-	-	26.04	0.13	3.38
KINDI-KEM	-	-	-	-	-	-	27.13	0.07	1.89	-	-	-	27.04	0.16	4.32
LAKE [13]	26.73	0.61	16.3	-	-	-	26.19	0.7	18.33	-	-	-	26.81	0.65	17.42
LEDA KEM [17]	26.78	14.06	376.52	-	-	-	26.49	57.81	1531.38	-	-	-	26.79	176.042	4716.16
Lepton [121]	26.82	0.0084	**0.22**	-	-	-	26.92	0.0246	**0.64**	-	-	-	27.01	0.025	**0.67**
LIMA CCA	27.51	0.42	11.55	27.13	0.85	23.06	27.62	0.9	24.85	-	-	-	27.44	1.56	42.8
LIMA CPA	26.82	0.43	11.53	27.04	0.79	21.36	26.95	0.9	24.25	-	-	-	27.11	1.56	42.29
Lizard KEM	26.39	2.5	65.97	-	-	-	26.58	10.46	278.02	-	-	-	26.77	5.78	154.73
RLizard KEM	27.33	0.04	1.09	-	-	-	27.26	0.08	2.18	-	-	-	27.24	0.107	2.62
LOCKER [14]	27.14	2.96	80.33	-	-	-	26.92	3.35	90.18	-	-	-	27.05	3.6	97.38
LOTUS Kem	26.78	10.02	268.33	-	-	-	27.13	18.13	491.86	-	-	-	26.81	26.44	708.85
Mersenne-756839 [4]	-	-	-	-	-	-	-	-	-	-	-	-	26.81	6.02	161.39
NewHope CCA [9]	27.1	0.16	4.33	-	-	-	-	-	-	-	-	-	27.05	0.33	8.92
NewHope CPA	26.94	0.154	4.14	-	-	-	-	-	-	-	-	-	26.89	0.3	8.06
NTRUEncrypt KEM	26.71	0.33	8.81	-	-	-	26.77	0.83	21.41	-	-	-	28.68	39.79	1061.59
NTRU-HRSS-KEM [67]	27.11	53.74	1456.89	-	-	-	-	-	-	-	-	-	-	-	-
NTRU Prime [27]	-	-	-	-	-	-	-	-	-	-	-	-	27.03	3.03	81.9
NTS-KEM [7]	26.93	16.54	445.42	-	-	-	26.58	44.98	1195.56	-	-	-	26.94	87.92	2368.56
Old Manhattan [101]	26.88	72.1	1938.04	-	-	-	27.05	139.2	3765.36	-	-	-	27.12	238.2	6459.98
Quroboros-R [88]	26.11	0.1	2.61	-	-	-	26.38	0.11	2.63	-	-	-	26.17	0.14	3.66
Post-Quantum RSA KEM	-	-	-	26.53	1336.76	35464.24	-	-	-	-	-	-	-	-	-

Continued from the previous page

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
QC-MDPC [117], [118]	-	-	-	-	-	-	26.67	87.03	2321.09	-	-	-	-	-	-
Ramstake [113]	27.18	2.35	63.87	-	-	-	26.94	10.88	293.1	-	-	-	-	-	-
RLCE-KEM [120]	26.85	390.06	**10473.11**	-	-	-	26.45	1554.88	41126.576	-	-	-	26.78	3853.39	103193.78
Round2-u KEM	26.41	0.14	3.69	26.11	0.14	3.65	26.32	0.65	17.10	26.57	0.5	13.28	26.39	0.29	7.65
Round2-n KEM	27.15	2.56	69.5	26.89	2.88	77.44	27.08	3.83	103.71	27.22	5.3	144.26	26.98	5.24	141.37
RQC [90]	27.06	0.27	7.3	-	-	-	26.82	0.45	12.06	-	-	-	27.14	0.76	20.62
SABER [49]	26.44	0.08	2.11	-	-	-	26.62	0.18	4.79	-	-	-	26.34	0.32	8.42
SIKE [69]	26.59	26.4	701.97	-	-	-	27.18	85.99	2337.2	-	-	-	-	-	-
Three Bears [63]	-	-	-	26.97	0.02	0.54	-	-	-	27.05	0.03	0.81	27.1	0.06	1.62
Titanium CCA	26.66	0.64	17.06	-	-	-	26.53	0.73	19.36	-	-	-	26.88	0.97	26.07

5.3.2 Public key encryption/encapsulation

Fourteen submissions focus on implementing public key encryption schemes with quantum safe algorithms. Tables 5.4, 5.5 and 5.6 provide the values of power usage, execution time and energy consumption of these implemented schemes. The most/least energy efficient in a particular group has been indicated with bold characters.

Around thirty nine submissions implemented public key encapsulation in this PQC standardization process. Tables 5.7, 5.8 and 5.9 provide the values of energy consumption corresponding for these schemes. Some candidate algorithms have provided both encapsulation and encryption techniques. So the same submission name has been reported for the different tables with the tags of -ENCRYPT or -KEM accordingly.

5.3.3 Other observations

In the previous subsections we have seen categorization of the submitted algorithms based on their energy efficiency for a particular security level. Furthermore, broadly all these algorithms come under the categories of well known post-quantum crypto tech-

Table 5.8: Energy consumption during **key encapsulation of public key encapsulation schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
BIGQUAKE	26.47	1.3	34.41	-	-	-	26.71	3.2	85.47	-	-	-	26.54	4.5	119.43
BIKE	25.88	0.229	5.95	-	-	-	26.11	0.23	6	-	-	-	26.08	1.1	28.68
CFPKM	26.41	188	**4965.08**	-	-	-	27.16	492	**13362.72**	-	-	-	-	-	-
Classic McEliece	-	-	-	-	-	-	-	-	-	-	-	-	27.16	0.34	9.23
CRYSTALSKyber	27.12	0.22	5.96	-	-	-	27.98	0.336	9.4	-	-	-	27.08	0.47	12.72
DAGS	-	-	-	-	-	-	26.66	0.0096	**0.256**	-	-	-	25.89	0.026	**0.673**
DING	26.98	2.01	54.22	-	-	-	-	-	-	-	-	-	27.12	4.01	108.75
DME	-	-	-	-	-	-	26.18	0.12	3.19	-	-	-	26.07	0.847	22.08
EMBLEM	25.52	0.928	23.47	-	-	-	-	-	-	-	-	-	-	-	-
FRODO	25.74	0.522	13.43	-	-	-	26.13	1.028	26.65	-	-	-	-	-	-
Hila5	-	-	-	-	-	-	-	-	-	-	-	-	26.69	1.23	32.82
HQC	25.73	0.4	10.29	-	-	-	26.18	0.94	24.6	-	-	-	26.42	1.3	34.34
AKCN MLWE	-	-	-	-	-	-	-	-	-	27.1	0.12	3.25	-	-	-
OKCN MLWE	-	-	-	-	-	-	-	-	-	26.49	0.13	3.44	-	-	-
OKCN SEC	-	-	-	-	-	-	-	-	-	-	-	-	26.53	0.21	5.57
AKCN SEC	-	-	-	-	-	-	-	-	-	-	-	-	26.79	0.23	6.16
KINDI-KEM	-	-	-	-	-	-	27.31	0.09	2.45	-	-	-	26.91	0.21	5.65
LAKE	25.92	0.11	2.85	-	-	-	26.31	0.11	2.89	-	-	-	26.44	0.12	3.17
LEDA KEM	25.84	2.083	53.82	-	-	-	26.21	13.542	354.93	-	-	-	26.1	35.417	924.38
Lepton	26.55	0.02	0.56	-	-	-	26.34	0.06	1.63	-	-	-	26.72	0.06	1.6
LIMA CCA	25.88	0.37	9.57	26.23	0.73	19.14	26.11	0.76	19.84	-	-	-	27.08	1.56	42.24
LIMA CPA	25.74	0.37	9.52	25.11	0.7	17.57	25.49	0.84	21.41	-	-	-	26.17	1.43	37.42
Lizard KEM	26.44	0.31	8.19	-	-	-	26.61	0.54	14.36	-	-	-	26.38	0.69	18.2
RLizard KEM	26.87	0.02	**0.53**	-	-	-	27.17	0.06	1.63	-	-	-	27.35	0.08	2.18
LOCKER	26.23	0.47	12.33	-	-	-	26.18	0.48	12.56	-	-	-	26.35	0.52	13.7
LOTUS Kem	26.72	0.08	2.13	-	-	-	26.63	0.11	2.92	-	-	-	27.91	0.19	5.3
Mersenne-756839	-	-	-	-	-	-	-	-	-	-	-	-	26.71	9.23	246.53
NewHope CCA	27.32	0.25	6.83	-	-	-	-	-	-	-	-	-	26.85	0.5	13.425
NewHope CPA	26.59	0.22	5.84	-	-	-	-	-	-	-	-	-	27.16	0.4	10.86
NTRUEncrypt KEM	28.13	0.06	1.68	-	-	-	27.11	0.12	3.25	-	-	-	26.52	61.83	1639.73
NTRU-HRSS-KEM	26.53	1.23	32.63	-	-	-	-	-	-	-	-	-	-	-	-
NTRU Prime	-	-	-	-	-	-	-	-	-	-	-	-	27.15	6.26	169.95
NTS-KEM	26.55	0.02	0.53	-	-	-	26.43	0.12	3.17	-	-	-	27.08	0.15	4.06
Old Manhattan	26.76	36.2	968.71	-	-	-	27.2	66.8	1816.96	-	-	-	27.23	147.34	**4012.06**
Quroboros-R	26.38	0.18	4.74	-	-	-	26.11	0.22	5.74	-	-	-	26.79	0.26	6.96
Post-Quantum RSA KEM	-	-	-	26.66	8.39	223.67	-	-	-	-	-	-	-	-	-

Continued from the previous page

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
QC-MDPC	-	-	-	-	-	-	26.52	6.05	160.44	-	-	-	-	-	-
Ramstake	27.21	4.34	118.09	-	-	-	26.52	19.82	525.62	-	-	-	-	-	-
RLCE-KEM	26.21	1.78	46.65	-	-	-	26.38	4.02	106.04	-	-	-	26.78	11.74	314.39
Round2-u KEM	27.18	0.34	9.24	26.87	0.57	15.31	27.24	2.71	73.82	27.18	0.44	11.95	26.95	0.59	15.9
Round2-n KEM	28.1	5.38	151.17	27.33	6.09	166.43	27.21	7.68	208.97	26.82	10.68	286.43	27.06	10.98	297.11
RQC	26.78	0.58	15.53	-	-	-	27.13	1.46	39.6	-	-	-	26.86	1.72	46.19
SABER	26.67	0.22	5.86	-	-	-	26.68	0.34	9.07	-	-	-	27.11	0.53	14.36
SIKE	27.06	43.22	1169.53	-	-	-	26.63	140.98	3754.29	-	-	-	-	-	-
Three Bears	-	-	-	26.54	0.04	1.06	-	-	-	26.68	0.04	1.06	26.32	0.08	2.1
Titanium CCA	25.86	0.59	15.25	-	-	-	26.14	0.67	17.51	-	-	-	26.44	0.92	24.32

niques such as lattice based, code based, multivariate, hash based etc. Few submissions also correspond to some different techniques other than the aforementioned ones such as GiophantusR [5] which deals with the underlying problem of solving indeterminate equations. In addition to that, the submissions such as Guess Again [108], Mersenne-756839 [4], Picnic [40], Postquantum RSA [29], Walnut [11] etc. are based on some novel problem which has not been explored before in any post-quantum cryptographic schemes. Therefore, based on these underlying problem, Tables 5.10, 5.11 and 5.12 again categorizes the submitted algorithms and mentions the top five in each group which seems to be energy efficient. It should be noted that these tables report the efficient algorithms considering all the five security levels. In case of code based cryptography there are only two signature schemes pqsigRM [81] and RaCoSS [94], both of which require significant amount of energy for their algorithm execution. Hence, they are not reported in Table 5.11. Also, for multivariate based cryptosystems, there are only two encapsulation submissions, namely CFPKM [39] and DME [85], again with the same issue of high energy consumption and as a result omission from Table 5.12. In the category of hash-based cryptosystems, there are

Table 5.9: Energy consumption during **key decapsulation of public key encapsulation schemes** where time is in milliseconds, power in Watts and energy in milliJoules

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
BIGQUAKE	26.53	1.6	42.44	-	-	-	26.38	10.2	269.07	-	-	-	26.57	14.7	390.58
BIKE	25.62	0.99	25.36	-	-	-	26.47	2.48	65.64	-	-	-	26.18	6.13	160.48
CFPKM	26.73	176	**4704.48**	-	-	-	26.52	502	**13313.04**	-	-	-	-	-	-
Classic McEliece	-	-	-	-	-	-	-	-	-	-	-	-	27.24	82.78	2254.92
CRYSTALSKyber	25.92	0.266	6.89	-	-	-	26.08	0.404	10.53	-	-	-	25.28	0.555	14.03
DAGS	-	-	-	-	-	-	26.36	0.046	**1.21**	-	-	-	26.57	0.17	4.51
DING	26.56	1.33	35.32	-	-	-	-	-	-	-	-	-	26.73	2.59	69.23
DME	-	-	-	-	-	-	26.24	0.59	15.48	-	-	-	26.45	4.19	110.82
EMBLEM	25.77	0.96	2.73	-	-	-	-	-	-	-	-	-	-	-	-
FRODO	26.14	0.52	13.59	-	-	-	26.26	1.03	27.04	-	-	-	-	-	-
Hila5	-	-	-	-	-	-	-	-	-	-	-	-	26.58	0.02	**0.53**
HQC	26.17	0.92	24.07	-	-	-	26.78	1.7	45.52	-	-	-	26.34	2.56	67.43
AKCN MLWE	-	-	-	-	-	-	-	-	-	26.85	0.02	0.53	-	-	-
OKCN MLWE	-	-	-	-	-	-	-	-	-	27.27	0.02	0.54	-	-	-
OKCN SEC	-	-	-	-	-	-	-	-	-	-	-	-	26.67	0.05	1.33
AKCN SEC	-	-	-	-	-	-	-	-	-	-	-	-	28.05	0.04	1.4
KINDI-KEM	-	-	-	-	-	-	26.81	0.12	3.21	-	-	-	26.86	0.25	6.71
LAKE	25.84	0.48	12.4	-	-	-	26.17	0.8	20.93	-	-	-	26.38	1.07	28.22
LEDA KEM	26.58	28.12	747.42	-	-	-	26.73	55.20	1475.49	-	-	-	26.37	154.16	4065.19
Lepton	26.83	0.02	**0.53**	-	-	-	26.71	0.07	1.87	-	-	-	26.93	0.07	1.88
LIMA CCA	25.94	0.47	12.19	26.16	0.94	24.59	26.47	0.98	25.94	-	-	-	25.83	1.9	49.07
LIMA CPA	26.68	0.125	3.33	26.43	0.23	6.07	27.11	0.24	6.5	-	-	-	26.86	0.45	12.08
Lizard KEM	26.47	0.36	9.52	-	-	-	26.73	0.66	17.64	-	-	-	27.23	0.81	22.05
RLizard KEM	27.13	0.03	0.81	-	-	-	27.23	0.07	1.9	-	-	-	26.93	0.11	2.96
LOCKER	26.46	1.73	45.77	-	-	-	26.38	1.78	46.95	-	-	-	26.51	2.39	63.35
LOTUS Kem	26.47	0.12	3.17	-	-	-	26.72	0.23	6.14	-	-	-	26.88	0.43	11.55
Mersenne-756839	-	-	-	-	-	-	-	-	-	-	-	-	27.16	18.18	493.76
NewHope CCA	27.08	0.28	7.58	-	-	-	-	-	-	-	-	-	27.11	0.57	15.72
NewHope CPA	26.83	0.04	1.07	-	-	-	-	-	-	-	-	-	26.49	0.08	2.11
NTRUEncrypt KEM	27.87	0.08	2.22	-	-	-	27.43	0.17	4.66	-	-	-	27.71	109.1	3023.16
NTRU-HRSS-KEM	26.85	3.58	96.12	-	-	-	-	-	-	-	-	-	-	-	-
NTRU Prime	-	-	-	-	-	-	-	-	-	-	-	-	27.23	9.35	254.6
NTS-KEM	26.48	0.2	5.29	-	-	-	26.67	0.36	9.6	-	-	-	26.95	0.83	22.36
Old Manhattan	27.16	40.17	1091.01	-	-	-	27.32	79.8	2180.13	-	-	-	26.85	163.32	**4385.14**
Quroboros-R	26.56	0.41	10.88	-	-	-	26.47	0.78	20.64	-	-	-	26.81	1.12	30.02
Post-Quantum RSA KEM	-	-	-	26.75	46.99	1256.98	-	-	-	-	-	-	-	-	-

Continued from previous page

Scheme	Security level I			Security level II			Security level III			Security level IV			Security level V		
	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy	Power	Time	Energy
QC-MDPC	-	-	-	-	-	-	27.13	71.8	1947.93	-	-	-	-	-	-
Ramstake	27.02	8.92	241.01	-	-	-	27.31	38.46	1050.34	-	-	-	-	-	-
RLCE-KEM	26.86	3.48	93.47	-	-	-	26.53	8.29	219.93	-	-	-	26.57	26.51	704.37
Round2-u KEM	27.04	0.13	3.51	27.12	0.35	9.49	26.96	1.93	52.03	27.15	0.34	9.23	27.26	0.28	7.63
Round2-n KEM	27.86	2.62	72.99	28.12	3.66	102.91	27.94	4.03	112.6	28.14	5.84	164.34	28.23	5.71	161.2
RQC	26.73	1.54	41.16	-	-	-	26.37	3.95	104.16	-	-	-	27.08	4.88	132.15
SABER	27.17	0.27	7.33	-	-	-	26.84	0.52	13.95	-	-	-	27.18	0.71	19.29
SIKE	26.86	46.11	1238.51	-	-	-	27.24	151.85	4136.4	-	-	-	-	-	-
Three Bears	-	-	-	26.76	0.05	1.34	-	-	-	26.92	0.06	1.61	26.58	1.06	28.17
Titanium CCA	26.13	0.68	17.76	-	-	-	26.57	0.77	20.45	-	-	-	25.93	1.07	27.74

Table 5.10: The energy efficient lattice based cryptographic algorithm submissions

Signing			Encapsulation/Encryption		
Key Generation	Sign	Verify	Key Generation	Enc	Dec
CRYSTALSDilithium	CRYSTALSDilithium	CRYSTALSDilithium	EMBLEM, KCL,	Lizard, Lepton,	Lepton, KCL
-	-	-	Lizard, Lepton,	LAC, KINDI,	New Hope CPA,
-	-	-	Round 2, LAC	LOTUS	Lizard, Round 2-u

two submissions namely Gravity - SPHINCS [72] and SPHINCS Plus [28], both consuming quite an amount of energy. And SIKE [69] is the only submission for supersingular elliptic curve isogeny based cryptography SIDH.

5.4 Summary

In this chapter, we have reported the energy consumption of all the NIST round 1 candidate algorithms [1], when they are executed on 64 bit Intel 6700 Skylake Processor, 3.4 GHz. We have consolidated our energy consumption data based on security levels and cryptographic operations. An overwhelming majority of the candidate algorithms are cate-

Table 5.11: The energy efficient code based cryptographic algorithm submissions

Encapsulation/Encryption schemes		
Key Generation	Enc	Dec
OuroborosR, HQC, BIKE, RQC, LAKE	NTS-KEM, LAKE, OuroborosR, BIKE, Classic McEliece, DAGS	OuroborosR, LAKE, Hila5, DAGS, NTS-KEM

Table 5.12: The energy efficient multivariate based cryptographic algorithm submissions

Signature schemes		
Key Generation	Sign	Verify
HiMQ3, HiMQ3F	Rainbow, HiMQ3, HiMQ3F	Gui, GeMSS, HiMQ3, HiMQ3F

gorized as either lattice, code or multi-variate based, and we identify leading energy efficient schemes from each category. There have been reports published analyzing the technicalities of these submissions. For example Martin et al. [8] investigated the lattice based cryptosystem's asymptotic run time. However, except for [20] where we compare energy efficiency of the classical elliptic curve Diffie-Hellman (ECDH) relative to SIDH/SIKE, there has not been any prior evaluation of energy consumption of the NIST round 1 post-quantum candidate algorithms.

In certain applications, energy constrained devices will perform signing and decryption operations while the more powerful servers will verify and encrypt. From Table 5.2, one can compute the median energy consumption for Level I signing algorithms to be 266.53 milli Joules and the corresponding algorithm is RaCoSS [94]. A practical experiment was performed to determine the number of signing operations for this particular submission RaCoSS, that can be performed on the same processor (as mentioned in Section 3.3 with a battery capacity of 60 Watt hours or 216 KJoules) until its battery gets exhausted. The experimental results showed around 800,000 signing operations, which is consistent with the Intel power gadget based results reported in the table.

Chapter 6

Concluding remarks

6.1 Summary and deductions

In this thesis, we have considered the energy consumed by various public key cryptosystems when they are executed on a 64 bit general purpose processor. To this end, first the power usage of the cryptosystems has been tracked using the Intel power gadget and then the energy consumed is determined by multiplying the average power usage and the execution time.

We have reported energy consumption of elliptic curve point addition and doubling for ECDH and SIDH when the curve points are represented using affine and projective coordinate systems. We have also compared ECDH with SIDH in terms of their energy consumption. Finally, we have reported a comprehensive comparison of the energy consumed by the NIST round 1 PQC candidate algorithms.

Our results show that projective coordinates are around 45 to 65 times more energy efficient than affine based representation. The operation of inversion, required in case of affine representation of coordinates, is implemented using Fermat's little theorem. Perhaps this algorithm is not a very efficient technique for this operation and as a result have increased execution time and energy consumption for affine based representation by a huge amount. In terms of the overall key exchange scheme, ECDH is around 45 to 55 times more energy efficient that SIDH for both the aforementioned security levels. Finally, our results indicate that some of the NIST PQC candidate algorithms are more energy efficient that the classical ECDH.

6.2 Future work

As can be seen from the previous chapters, the variations in power usages by the cryptographic schemes considered here are mostly small. An algorithm's energy consumption is the product of its average power usage and the execution time. We do not expect the power usages to vary considerably if an algorithm undergoes further optimization. As a result, algorithm optimization based reduction in execution time is likely to yield roughly a proportionate reduction in energy consumption, assuming the same C based implementation.

Vectorized and/or floating-point instruction based implementations add another degree of freedom to the effort of reducing execution time and energy consumption. Vectorized and floating-point instructions use some part of the processor that are not used by regular integer instructions. So, investigating the relative energy efficiencies of the candidate algorithms from PQC NIST submissions for the vectorized implementations would be interesting.

In this work, only standard projective coordinates and affine coordinates are explored for energy consumptions. Other coordinate systems could be used to implement the key exchange and then an analogous comparative analyis can be performed. Also in case of comparison between affine and projective coordinate based representations, the inversion operation for affine coordinates could be computed using more efficient algorithms such as extended Euclidean algorithm, binary gcd algorithm etc. [64]

In this work, software implementations on 64 bit processors have been investigated. It is not known with certainty if similar relative energy efficiency will hold for implementations on processors with different data paths such as 8, 16, or 32 bit processors. Additionally, hardware based implementations of the above mentioned cryptographic schemes could be compared for energy consumption.

References

[1] NIST Round 1 Submissions. National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[2] OpenSSL project, Cryptography and SSL/TLS toolkit. Available at https://www.openssl.org, Accessed 26 June 2018.

[3] Gora Adj, Daniel Cervantes-Vázquez, Jesús-Javier Chi-Domínguez, Alfred Menezes, and Francisco Rodríguez-Henríquez. On the cost of computing isogenies between supersingular elliptic curves. Technical report, Cryptology ePrint Archive, Report 2018/313, 2018. https://eprint. iacr. org/2018/313, Accessed 26 June 2018.

[4] Divesh Aggarwal, Antoine Joux, Anupam Prakash, and Miklos Santha. A new public-key cryptosystem via Mersenne numbers. Technical report, Cryptology ePrint Archive, Report 2017/481, 2017. http://eprint. iacr. org/2017/481, 2017.

[5] Koichiro Akiyama, Yasuhiro Goto, Shinya Okumura, Tsuyoshi Takagi, Koji Nuida, Goichiro Hanaoka, Hideo Shimizu, and Yasuhiko Ikematsu. A public-key encryption scheme based on non-linear indeterminate equations (Giophantus).

[6] A Adrian Albert. Modern Higher Algebra, Chicago, 1937. *Zentralblatt MATH*, 19.

[7] Martin Albrecht, Carlos Cid, Kenneth G. Paterson, Cen Jung Tjhai, and Martin Tomlinson. NTS-KEM. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[8] Martin R Albrecht, Benjamin R Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the LWE, NTRU schemes! 2017.

[9] Erdem Alkim, Roberto Avanzi, Joppe Bos, Léo Ducas, Antonio de la Piedra, Thomas Pöppelmann, Peter Schwabe, and Douglas Stebila. NewHope. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[10] Erdem Alkim, Nina Bindel, Johannes Buchmann, Özgür Dagdelen, Edward Eaton, Gus Gutoski, Juliane Krämer, and Filip Pawlega. Revisiting TESLA in the quantum random oracle model. In *International Workshop on Post-Quantum Cryptography*, pages 143–162. Springer, 2017.

[11] Iris Anshel, Derek Atkins, Dorian Goldfeld, and Paul E. Gunnells. WALNUT digital signature scheme. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[12] Nicolas Aragon, Paulo S. L. M. Barreto, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Shay Gueron, Tim Güneysu, Carlos Aguilar Melchor, Rafael Misoczki, Edoardo Persichetti, Nicolas Sendrier, Jean-

Pierre Tillich, and IMB Gilles Zémor. BIKE. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[13] Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, and Gilles Zémor. LAKE–Low rank parity check codes key exchange at first round submission to the NIST post-quantum cryptography call, 2017.

[14] Nicolas Aragon, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Olivier Ruatta, Jean-Pierre Tillich, and Gilles Zémor. LOCKER. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[15] Reza Azarderakhsh, David Jao, Kassem Kalach, Brian Koziel, and Christopher Leonardi. Key compression for isogeny-based cryptosystems. In *Proceedings of the 3rd ACM International Workshop on ASIA Public-Key Cryptography*, pages 1–10. ACM, 2016.

[16] H. Baan, S. Bhattacharya, Garcia-Morchon, R. Rietman, L. Toihuizen, J.L. Torre-Arce, and Z. Zhamig. Round2: KEM and PKE based on GLWR. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[17] Marco Baldi, Alessandro Barenghi, Franco Chiaraluce, Gerardo Pelosi, and Paolo Santini. LEDAkem: a post-quantum key encapsulation mechanism based on QC-

LDPC codes. In *International Conference on Post-Quantum Cryptography*, pages 3–24. Springer, 2018.

[18] Marco Baldi, Alessandro Barenghi, Franco Chiaraluce, Gerardo Pelosi, and Paolo Santini. LEDA. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[19] Gustavo Banegas, Paulo S. L. M. Barreto, Brice Odilon Boidje, Pierre-Louis Cayrel, Gilbert Ndollane Dione, Kris Gaj, Cheikh Thi´ecoumba Gueye, Richard Haeussler, Jean Belo Klamti, Ousmane N'diaye, Duc Tri Nguyen, Edoardo Persichetti, and Jefferson E. Ricardini. DAGS. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[20] Tanushree Banerjee and Anwar Hasan. Energy Efficiency Analysis of Elliptic Curve based Cryptosystems. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2018 IEEE 17th International Conference*. IEEE, 2018.

[21] Tanushree Banerjee and M. Anwar Hasan. Energy consumption of candidate algorithms for NIST PQC standards. Technical report, Centre for Applied Cryptographic Research (CACR) at the University of Waterloo http://cacr.uwaterloo.ca/, Accessed 26 June 2018.

[22] Tanushree Banerjee and M. Anwar Hasan. Energy efficiency analysis of elliptic curve based cryptosystems. Technical report, Centre for Applied Cryptographic Research (CACR) at the University of Waterloo http://cacr.uwaterloo.ca/, Accessed 26 June 2018.

[23] Rachid El Bansarkhani. KINDI. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[24] Magali Bardet, Elise Barelli, Olivier Blazy, Rodolfo Canto, Alain Couvreur, Philippe Gaborit, Ayoub Otmani, Nicolas Sendrier, and Jean-Pierre Tillich. BIG QUAKE. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[25] Daniel J Bernstein. SUPERCOP: System for unified performance evaluation related to cryptographic operations and primitives, 2009.

[26] Daniel J. Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, Jakub Szefer, and Wen Wang. Classic McEliece. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[27] Daniel J Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU Prime. *IACR Cryptology ePrint Archive*, 2016:461, 2016.

[28] Daniel J Bernstein, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M Lauridsen, et al. SPHINCS. 2017.

[29] Daniel J Bernstein, Nadia Heninger, Paul Lou, and Luke Valenta. Post-quantum RSA. In *International Workshop on Post-Quantum Cryptography*, pages 311–329. Springer, 2017.

[30] Daniel J Bernstein and Tanja Lange. Faster addition and doubling on elliptic curves. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 29–50. Springer, 2007.

[31] Ward Beullens, Bart Preneel, Alan Szepieniec, and Frederik Vercauteren. LUOV. Technical report, National Institute of Standards and Technology, 2017 available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`, Accessed 26 June 2018.

[32] Jean-François Biasse, David Jao, and Anirudh Sankar. A quantum algorithm for computing isogenies between supersingular elliptic curves. In *International Conference in Cryptology in India*, pages 428–442. Springer, 2014.

[33] Ian Blake, Gadiel Seroussi, and Nigel Smart. *Elliptic curves in cryptography*, volume 265. Cambridge university press, 1999.

[34] Simon Blake-Wilson, Bodo Moeller, Vipul Gupta, Chris Hawk, and Nelson Bolyard. Elliptic curve cryptography (ECC) cipher suites for transport layer security (TLS). 2006.

[35] Joppe Bos, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, and Damien Stehlé. Crystals–Kyber: a CCA-secure module-lattice-based KEM. *IACR Cryptology ePrint Archive*, 2017:634, 2017.

[36] Joppe W Bos, J Alex Halderman, Nadia Heninger, Jonathan Moore, Michael Naehrig, and Eric Wustrow. Elliptic curve cryptography in practice. In *International Conference on Financial Cryptography and Data Security*, pages 157–175. Springer, 2014.

[37] Antoine Casanova, Jean-Charles Faugère, Gilles Macario-Rat, Jacques Patarin, Ludovic Perret, and Jocelyn Ryckeghem. *GeMSS: A Great Multivariate Short Signature*. PhD thesis, UPMC-Paris 6 Sorbonne Universités; INRIA Paris Research Centre, MAMBA Team, F-75012, Paris, France; LIP6-Laboratoire d'Informatique de Paris 6, 2017.

[38] Wouter Castryck, Steven D Galbraith, and Reza Rezaeian Farashahi. Efficient arithmetic on elliptic curves using a mixed Edwards-Montgomery representation. *IACR Cryptology ePrint Archive*, 2008:218, 2008.

[39] Olive Chakraborty, Jean-Charles Faugère, and Ludovic Perret. *CFPKM: A Key Encapsulation Mechanism based on Solving System of non-linear multivariate Polynomials 20171129*. PhD thesis, UPMC-Paris 6 Sorbonne Universités; INRIA Paris; CNRS, 2017.

[40] Melissa Chase, David Derler, Steven Goldfeder, Claudio Orlandi, Sebastian Ramacher, Christian Rechberger, Daniel Slamanig, and Greg Zaverucha. Post-quantum zero-knowledge and signatures from symmetric-key primitives. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1825–1842. ACM, 2017.

[41] Cong Chen, Jeffrey Hoffstein, William Whyte, and Zhenfei Zhang. pqNTRUSign. Technical report, National Institute of Standards and Technology, 2017

available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[42] Lily Chen, Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray Perlner, and Daniel Smith-Tone. *Report on post-quantum cryptography*. US Department of Commerce, National Institute of Standards and Technology, 2016.

[43] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-Pass MQ-Based Identification to MQ-Based Signatures. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 135–165. Springer, 2016.

[44] Jung Hee Cheon, Duhyeong Kim, Joohee Lee, and Yongsoo Song. Lizard Public Key Encryption. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[45] Andrew Childs, David Jao, and Vladimir Soukharev. Constructing elliptic curve isogenies in quantum subexponential time. *Journal of Mathematical Cryptology*, 8(1):1–29, 2014.

[46] Henri Cohen, Atsuko Miyaji, and Takatoshi Ono. Efficient elliptic curve exponentiation using mixed coordinates. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 51–65. Springer, 1998.

[47] Craig Costello, Patrick Longa, and Michael Naehrig. Efficient algorithms for supersingular isogeny Diffie-Hellman. In *Annual Cryptology Conference*, pages 572–601. Springer, 2016.

[48] Amir K Daneshbeh and M Anwarul Hasan. Area efficient high speed elliptic curve cryptoprocessor for random curves. In *Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. International Conference on*, volume 2, pages 588–592. IEEE, 2004.

[49] Jan-Pieter D'Anvers, Angshuman Karmakar, Sujoy Sinha Roy, and Frederik Vercauteren. Saber: Module-LWR based key exchange, CPA-secure encryption and CCA-secure KEM. In *International Conference on Cryptology in Africa*, pages 282–305. Springer, 2018.

[50] Christina Delfs and Steven D Galbraith. Computing isogenies between supersingular elliptic curves over Fp. *arXiv preprint arXiv:1310.7789*, 2013.

[51] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[52] Jintai Ding and Albrecht Petzoldt. Current state of multivariate cryptography. *IEEE Security & Privacy*, 15(4):28–36, 2017.

[53] Jintai Ding, Tsuyoshi Takagi, Yuntao Wang, and Xinwei Gao. DING. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[54] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS-Dilithium: a lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, 2018.

[55] Alexandre Wagner Chagas Faria, Leandro Pfleger de Aguiar, Daniel da Silva Diogo Lara, and Antônio Alfredo Ferreira Loureiro. Comparative Analysis of Power Consumption in Arithmetic Algorithms Implementation. *Revista de Informática Teórica e Aplicada*, 18(2):234–250, 2011.

[56] Keith I Farkas, Jason Flinn, Godmar Back, Dirk Grunwald, and Jennifer M Anderson. Quantifying the energy consumption of a pocket computer and a java virtual machine. *ACM SIGMETRICS Performance Evaluation Review*, 28(1):252–263, 2000.

[57] Jean-Charles Faugère, Ludovic Perret, and Jocelyn Ryckeghem. *DualModeMS: A Dual Mode for Multivariate-based Signature 20170918 draft.* PhD thesis, UPMC-Paris 6 Sorbonne Universités; INRIA Paris; CNRS, 2017.

[58] Armando Faz-Hernández, Julio López, Eduardo Ochoa-Jiménez, and Francisco Rodríguez-Henríquez. A faster software implementation of the supersingular isogeny Diffie-Hellman key exchange protocol. *IEEE Transactions on Computers*, 2017.

[59] PUB Fips. 186-2. Digital Signature Standard (DSS). *National Institute of Standards and Technology (NIST)*, 20:13, 2000.

[60] Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Prest, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. Falcon: Fast-Fourier, lattice-based, compact signatures over NTRU. *Submission to NIST Post-Quantum Competition*, 17, 2017.

[61] Lucky Galvez, Jon-Lark Kim, Myeong Jae Kim, Young-Sik Kim, and Nari Lee. McNie: Compact McEliece-Niederreiter Cryptosystem. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/

projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[62] M Goraczko, A Kansal, J Liu, and F Zhao. Joulemeter: Computational energy measurement and optimization, 2011.

[63] Mike Hamburg. Module-LWE key exchange and encryption: The three bears. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[64] Darrel Hankerson, Alfred J Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[65] M Anwar Hasan, Nicolas Meloni, Ashkan H Namin, and Christophe Negre. Block recombination approach for subquadratic space complexity binary field multiplication based on Toeplitz matrix-vector product. *IEEE Transactions on Computers*, 61(2):151–163, 2012.

[66] Jeffrey Hoffstein, Jill Pipher, John M. Schanck, Joseph H. Silverman, William Whyte, and Zhenfei Zhang. NTRUEncrypt : A lattice based encryption algorithm. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[67] Andreas Hulsing, Joost Rijneveld, John M. Schanck, and Peter Schwabe. NTRU-HRSS-KEM. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[68] PQCRYPTO ICT. 645622 (2015). https://pqcrypto.eu.org/, Accessed 26 June 2018.

[69] David Jao, Reza Azarderakhsh, Matthew Campagna, Craig Costello, Amir Jalali, Brian Koziel, Brian LaMacchia, Patrick Longa, Michael Naehrig, Joost Renes, et al. Supersingular Isogeny Key Encapsulation. 2017.

[70] David Jao and Luca De Feo. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *International Workshop on Post-Quantum Cryptography*, pages 19–34. Springer, 2011.

[71] David Jao and Vladimir Soukharev. A subexponential algorithm for evaluating large degree isogenies. *Algorithmic number theory*, pages 219–233, 2010.

[72] Guillaume Endignoux Jean-Philippe Aumasson. Gravity-SPHINCS. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[73] Koray Karabina. Point decomposition problem in binary elliptic curves. In *International Conference on Information Security and Cryptology*, pages 155–168. Springer, 2015.

[74] Anatolii Karatsuba and Yuri Ofman. Multiplication of many-digital numbers by automatic computers. In *Doklady Akad. Nauk SSSR*, volume 145, page 85, 1962.

[75] Colin Ian King. Powerstat. https://github.com/ColinIanKing/powerstat, Accessed 26 June 2018.

[76] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, 48(177):203–209, 1987.

[77] Paul C Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In *Annual International Cryptology Conference*, pages 104–113. Springer, 1996.

[78] Michiel Kosters and Sze Ling Yeo. Notes on summation polynomials. *arXiv preprint arXiv:1503.08001*, 2015.

[79] David W Kravitz. Digital signature algorithm, July 27, 1993. US Patent 5,231,668.

[80] K Sathish Kumar, R Sukumar, and P Asrin Banu. An experimental study on energy consumption of cryptographic algorithms for mobile hand-held devices. *International Journal of Computer Applications (0975–8887) Volume*, 2012.

[81] Wijik Lee, Young-Sik Kim, and Jong-Seon No. A new signature scheme based on punctured Reed–Muller code with random insertion. *arXiv preprint arXiv:1711.00159*, 2017.

[82] Dongxi Liu, Nan Li, Jongkil Kim, and Surya Nepal. Compact-LWE: Enabling practically lightweight public key encryption for leveled IoT device authentication. Technical report, Cryptology ePrint Archive, Report 2017/685, 2017. http://eprint. iacr. org/2017/685.

[83] LLC LLNS. LibMSR. https://software.llnl.gov/libmsr/, Accessed 26 June 2018.

[84] Xianhui Lu1, Yamin Liu, Dingding Jia, Haiyang Xue, Jingnan He, and Zhenfei Zhang. LAC. Technical report, National Institute of Standards and Technology, 2017

available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[85] I Luengo. DME : A public key, signature and KEM system based on double exponentiation with matrix exponents. *preprint*, 2017.

[86] Jonathan Lutz and M Anwarul Hasan. High performance elliptic curve cryptographic co-processor. In *Wireless Network Security*, pages 3–42. Springer, 2007.

[87] Emmanuela Orsini Martin R. Albrecht, Yehuda Lindell, Valery Osheter, Kenneth G. Paterson, Guy Peer, and Nigel P. Smart. LIMA : A PQC Encryption Scheme. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[88] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Adrien Hauteville, Gilles Zémor, and INSA-CVL Bourges. Ouroboros-R. 2017.

[89] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, Edoardo Persichetti, Gilles Zémor, and INSA-CVL Bourges. Hamming Quasi-Cyclic (HQC). Technical report, National Institute of Standards and Technology, 2017 available at https://pqc-hqc.org/doc/hqc-spec.pdf, Accessed 26 June 2018.

[90] Carlos Aguilar Melchor, Nicolas Aragon, Slim Bettaieb, Loïc Bidoux, Olivier Blazy, Jean-Christophe Deneuville, Philippe Gaborit, and Gilles Zémor. Rank Quasi Cyclic (RQC), first round submission to the NIST post-quantum cryptography call, 2017.

[91] Alfred Menezes, Minghua Qu, Scott Vanstone, and Kennedy Jim Sutherland. Elliptic curve systems. In *IEEE P1363, Part 4: Elliptic Curve Systems*. Citeseer, 1995.

[92] Victor S Miller. Use of elliptic curves in cryptography. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985.

[93] Peter L Montgomery. Speeding the Pollard and elliptic curve methods of factorization. *Mathematics of computation*, 48(177):243–264, 1987.

[94] Kirill Morozov, Partha Sarathi Roy, and Kouichi Sakurai. On unconditionally binding code-based commitment schemes. In *Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication*, page 101. ACM, 2017.

[95] Michael Naehrig, Erdem Alkim, Joppe W. Bos, L´eo Ducas, Patrick Longa, Ilya Mironov, Valeria Nikolaenko, Chris Peikert, Ananth Raghunathan, and Douglas Stebila. FrodoKEM : Learning With Errors Key Encapsulation. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[96] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[97] Katsuyuki Okeya, Hiroyuki Kurumatani, and Kouichi Sakurai. Elliptic curves with the Montgomery-form and their cryptographic applications. In *International Workshop on Public Key Cryptography*, pages 238–257. Springer, 2000.

[98] Albrecht Petzoldt, Ming-Shing Chen, Jintai Ding, and Bo-Yin Yang. HMFEv-an efficient multivariate signature scheme. In *International workshop on post-quantum cryptography*, pages 205–223. Springer, 2017.

[99] Le Trieu Phong, Takuya Hayashi, Yoshinori Aono, and Shiho Moriai. LOTUS. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[100] T Plantard, A Sipasseuth, C Dumondelle, and W Susilo. Diagonal dominant reduction for lattice-based signature, 2018.

[101] Thomas PLANTARD. Odd Manhattan. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[102] Joseph C Pry and Richard K Lomotey. Energy consumption cost analysis of mobile data encryption and decryption. In *Mobile Services (MS), 2016 IEEE International Conference on*, pages 178–181. IEEE, 2016.

[103] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[104] Markku-Juhani O Saarinen. HILA5: On Reliability, Reconciliation, and Error Correction for Ring-LWE Encryption. In *International Conference on Selected Areas in Cryptography*, pages 192–212. Springer, 2017.

[105] Igor Semaev. New algorithm for the discrete logarithm problem on elliptic curves. *arXiv preprint arXiv:1504.01175*, 2015.

[106] Minhye Seo, Jong Hwan Park, Dong Hoon Lee, Suhri Kim, and Seung-Joon Lee. Emblem : Error-blocked multi-bit LWE based KEM. Technical report, National

Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[107] Kyung-Ah Shim, Cheol-Min Park, and Namhun Koo. An Existential Unforgeable Signature Scheme Based on Multivariate Quadratic Equations. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 37–64. Springer, 2017.

[108] Vladimir Shpilrain, Mariya Bessonov, Alexey Gribov, and Dima Grigoriev. Guess Again. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[109] Joseph H Silverman. *The arithmetic of elliptic curves*, volume 106. Springer Science & Business Media, 2009.

[110] Intel Open Source. Powertop. https://01.org/powertop, 2013.

[111] Douglas Stebila and Jon Green. Elliptic-Curve Algorithm Integration in the Secure Shell Transport Layer. 2009.

[112] Ron Steinfeld, Amin Sakzad, and Raymond Kuo Zhao. Titanium: Proposal for a NIST Post-Quantum Public-key Encryption and KEM Standard. 2017.

[113] Alan Szepieniec. Ramstake. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[114] Dan Terpstra, Heike Jagode, Haihang You, and Jack Dongarra. Collecting performance data with PAPI-C. In *Tools for High Performance Computing 2009*, pages 157–173. Springer, 2010.

[115] Jacques Vélu. Isogénies entre courbes elliptiques. *CR Acad. Sc. Paris.*, 273:238–241, 1971.

[116] Thomas Willhalm, Roman Dementiev, and Patrick Fay. Intel Performance Counter Monitor. https://software.intel.com/en-us/articles/intel-performance-counter-monitor, Accessed 26 June 2018.

[117] Atsushi Yamada. Key encapsulation mechanisms, March 6 2018. US Patent 9,912,479.

[118] Atsushi Yamada, Edward Eaton, Kassem Kalach, Philip Lafrance, and Alex Parent. QC-MDPC. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[119] Mike Yi. Intel Power Gadget. https://software.intel.com/en-us/articles/intel-power-gadget-20, Accessed 26 June 2018.

[120] Wang Yongge. Quantum resistant random linear code based public key encryption scheme RLCE. In *Information Theory (ISIT), 2016 IEEE International Symposium on*, pages 2519–2523. IEEE, 2016.

[121] Yu Yu and Jiang Zhang. Lepton : Key Encapsulation Mechanisms from a variant of Learning Parity with Noise. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.

[122] Yunlei Zhao, Zhengzhong Jin, Boru Gong, and Guangye Sui. OKCN/AKCN/C-NKE: A Modular and Systematic Approach to Key Establishment and Public-Key Encryption Based on LWE and Its Variants. Technical report, National Institute of Standards and Technology, 2017 available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions, Accessed 26 June 2018.