RESULTS ON THE MULTIPLICATION IN FINITE FIELDS OF CHARACTERISTIC THREE USING MODIFIED POLYNOMIAL REPRESENTATION AND NORMAL ELEMENTS IN BINARY FIELDS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

CANAN ÖZEL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
CRYPTOGRAPHY

MARCH 2013

Approval of the thesis:

RESULTS ON THE MULTIPLICATION IN FINITE FIELDS OF CHARACTERISTIC THREE USING MODIFIED POLYNOMIAL REPRESENTATION AND NORMAL ELEMENTS IN BINARY FIELDS

submitted by **CANAN ÖZEL** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Department of Cryptography, Middle East Technical University** by,

Prof. Dr. Bülent Karasözen
Director, Graduate School of **Applied Mathematics** _____

Prof. Dr. Ferruh Özbudak
Head of Department, **Cryptography** _____

Prof. Dr. Ferruh Özbudak
Supervisor, **Department of Mathematics, METU** _____

Assist. Prof. Dr. Sedat Akleylek
Co-supervisor, **The Department of Computer Engineering, Ondokuz Mayıs University** _____

Examining Committee Members:

Prof. Dr. Ersan Akyıldız
Department of Mathematics, METU _____

Prof. Dr. Ferruh Özbudak
Department of Mathematics, METU _____

Assoc. Prof. Dr. Ali Doğanaksoy
Department of Mathematics, METU _____

Assist. Prof. Dr. Burcu Gülmez Temur
Department of Mathematics, Atılım University _____

Dr. Oğuz Yayla
Department of Cryptography, IAM, METU _____

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name: CANAN ÖZEL

Signature :

ABSTRACT

RESULTS ON THE MULTIPLICATION IN FINITE FIELDS OF CHARACTERISTIC THREE USING MODIFIED POLYNOMIAL REPRESENTATION AND NORMAL ELEMENTS IN BINARY FIELDS

Özel, Canan

Ph.D., Department of Cryptography

Supervisor : Prof. Dr. Ferruh Özbudak

Co-Supervisor : Assist. Prof. Dr. Sedat Akleylek

March 2013, 58 pages

In this thesis, we study on the multiplication in finite fields of characteristic three. We use Charlier and Hermite polynomials to represent elements in $\mathbb{F}_{3^n}$ for obtaining alternative representations to the standart polynomial representation. We give multiplication methods in these representations to multiply elements in $\mathbb{F}_{3^n}$. We compute the multiplication and reduction complexities in each representation and compare the complexity results with the ones in the standart polynomial representation. Charlier and Hermite polynomial representations enable us to find irreducible binomials. We show that in some cases, there is a set of irreducible binomials in each representation to do modular reduction with lower addition complexity than the one in the standart polynomial representation. We give a multiplier architecture in Charlier polynomial representation of finite fields $\mathbb{F}_{3^n}$, where $n \equiv 2 \pmod 3$. Then, we examine cubing and inversion operations in this representation. We investigate the matrix-vector product method for multiplication of the field elements in Hermite polynomial representation and we generalize the reduction complexity by using the reduction matrix in this matrix-vector product method. Finally, we focus on the optimal normal basis construction in binary fields and find a connection between optimal normal basis elements and Hermite polynomials in these fields.

Keywords : finite field representation, polynomial multiplication, reduction complexity, normal elements

ÖZ

DEĞİŞTİRİLMİŞ POLİNOM GÖSTERİMİ KULLANILARAK KARAKTERİSTİĞİ ÜÇ OLAN SONLU CİSİMLERDE ÇARPMA ÜZERİNE VE İKİLİK CİSİMLERDE NORMAL ELEMANLAR ÜZERİNE SONUÇLAR

Özel, Canan

Doktora, Kriptografi Bölümü

Tez Yöneticisi : Prof. Dr. Ferruh Özbudak

Ortak Tez Yöneticisi : Yrd. Doç. Dr. Sedat Akleylek

Mart 2013, 58 sayfa

Bu tezde, karakteristiği üç olan sonlu cisimlerdeki çarpma üzerinde çalışıyoruz. Standart polinom gösterimine alternatif gösterimler elde etmek için, $\mathbb{F}_{3^n}$' deki elemanları göstermede Charlier ve Hermite polinomlarını kullanıyoruz. $\mathbb{F}_{3^n}$' deki elemanları çarpmak için bu gösterimlerdeki çarpma yöntemlerini veriyoruz. Her bir gösterimde, çarpma ve indirgeme karmaşıklıklarını hesaplıyoruz ve karmaşıklık sonuçlarını standart polinom gösterimindekilerle karşılaştırıyoruz. Charlier ve Hermite polinom gösterimleri indirgenemez iki terimli polinomlar bulabilmemize olanak sağlamaktadır. Bazı durumlarda, standart polinom gösterimindekine göre daha az toplama karmaşıklığı olan modüler indirgeme yapmak için her bir gösterimde indirgenemez iki terimli polinomlar kümesi olduğunu gösteriyoruz. $n \equiv 2 \pmod 3$ olan $\mathbb{F}_{3^n}$ sonlu cisimlerinin Charlier polinom gösteriminde bir çarpan yapısı veriyoruz. Daha sonra küp alma ve tersini alma işlemlerini inceliyoruz. Hermite polinom gösteriminde cisim elemanlarının çarpımı için matris-vektör çarpım yöntemini inceliyoruz ve bu matris-vektör çarpım yönteminde indirgeme matrisini kullanarak indirgeme karmaşıklığını genelleştiriyoruz. Son olarak, ikilik cisimlerde optimal normal tabanların oluşturulmasına odaklanıyoruz ve bu cisimlerde, optimal normal taban elemanları ve Hermite polinomları arasında bir bağlantı buluyoruz.

Anahtar Kelimeler: sonlu cisimlerin gösterimleri, polinom çarpımı, indirgeme karmaşıklığı, normal elemanlar

Onur'a

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to my supervisor Prof. Dr. Ferruh Özbudak for his guidance throughout this work. It would not have been possible to write this thesis without his help and encouraging support. I am also very greatful to Assist. Prof. Dr. Sedat Akleylek, my co-supervisor for his valuable advices and constant help during this period. His insight and suggestions have improved this work.

I would like to thank Dr. Oğuz Yayla for his comments and constructive feedback. Also many thanks to Dr. Turgut Hanoymak for his support at all times I need during my gradute study. I also would like to thank all my friends in the institute, especially Yeliz Yolcu Okur for her encouragement and friendship.

I would like to show my gratitude towards my parents. They were always there for me and they always supported me. Also I am grateful to my sister and her family for their love and support. I would like to thank my parents-in-law and also I am so thankful to my dear grandmother who will always with us.

Lastly, my special thanks go to my loving husband Onur for his unconditional support and love that enabled me to complete this work.

TABLE OF CONTENTS

ABSTRACT vii

ÖZ ix

ACKNOWLEDGMENTS xiii

TABLE OF CONTENTS xv

LIST OF TABLES xvii

CHAPTERS

1 INTRODUCTION 1

2 MATHEMATICAL BACKGROUND 3

2.1 Finite Fields 3

2.2 Extension Fields 5

2.3 Binary Field Arithmetic 7

2.3.1 Multiplication 7

2.3.2 Squaring 9

2.3.3 Inversion 10

3 CHARLIER POLYNOMIAL REPRESENTATION 11

3.1 Charlier Polynomials 11

3.1.1 Charlier Basis 13

3.2 Multiplication in Charlier Representation 13

3.2.1 Irreducible Charlier Binomials 16

3.2.1.1 Reduction 16
3.3 Design of Arithmetic Operations in Charlier Polynomial Representation 18
3.3.1 A Sequential Multiplier 19
3.3.2 Cubing 21
3.3.3 Inversion 22
4 HERMITE POLYNOMIAL REPRESENTATION 25
4.1 Hermite Polynomials and Hermite Basis in $\mathbb{F}_{3^n}$ 25
4.2 Multiplication of Polynomials in Hermite Representation 27
4.2.1 Irreducible Hermite Binomials 29
4.2.1.1 Reduction 29
4.3 Matrix vector product for Hermite Basis 31
5 NORMAL BASIS REPRESENTATION 35
5.1 Optimal Normal Bases 35
5.2 Dickson Polynomials 36
5.3 Hermite Polynomials in $\mathbb{F}_2[x]$ 38
6 CONCLUSION 41
REFERENCES 43
A CONVERSION FROM STANDART POLYNOMIAL REPRESENTATION TO A MODIFIED POLYNOMIAL REPRESENTATION 47
B IRREDUCIBLE CHARLIER BINOMIALS AND REDUCTION 49
C IRREDUCIBLE HERMITE BINOMIALS AND REDUCTION 53
CURRICULUM VITAE 57

LIST OF TABLES

Table 3.1 Charlier polynomials in $\mathbb{F}_3[x]$ 12
Table 3.2 Reduction Complexity 18

Table 4.1 Hermite polynomials in $\mathbb{F}_3[x]$ 26
Table 4.2 Reduction Complexity 31

Table 5.1 Dickson polynomials in $\mathbb{F}_2[x]$ 37
Table 5.2 Hermite polynomials in $\mathbb{F}_2[x]$ 39
Table 5.3 Polynomials generated by $A_n(x)$ over $\mathbb{F}_2$ 40

Table B.1 Irreducible Charlier Binomials 49
Table B.2 Reduction Complexity of β_i 51

Table C.1 Irreducible Hermite Binomials 53
Table C.2 Reduction Complexity of β_i 56

CHAPTER 1

INTRODUCTION

Finite field arithmetic is widely studied in cryptographic applications, coding theory and computer algebra [27]. Recently, in elliptic curve cryptography and pairing based cryptography there has been an interest on the implementation of cryptographic systems based on odd characteristic finite fields $\mathbb{F}_{p^n}$, where p is a prime [12], [13]. For instance, supersingular elliptic curves over $\mathbb{F}_{3^n}$ are used for the efficient implementations of the algorithms in pairing based cryptography [10]. The algorithms require the basic arithmetic operations such as field addition, subtraction, multiplication and cubing. Due to their complexities, multiplication and cubing are the important operations for pairing implementations. In the implementation of Tate pairing, multiplication and cubing in $\mathbb{F}_{3^{6m}}$ are required [26]. As a result of these, the studies have been increased in the area of hardware and software implementations of arithmetic operations in finite fields of characteristic three [3], [11], [19], [22], [35].

Multiplication is the most time consuming operation in finite field arithmetic. The other complex operations such as exponentiation and inversion are done by performing the multiplication operation iteratively. Multiplication of finite field elements can be performed in two steps: polynomial multiplication and modular reduction. Since the complexity of the multiplication depends on the number of nonzero terms in the reduction polynomial, it is desirable to use a low weight irreducible polynomial as the reduction polynomial. The selection of the representation of finite field elements and the irreducible polynomial has a crucial role on the efficiency of the arithmetic implementations. In $\mathbb{F}_{3^n}$, there is no irreducible binomial except for $x^2 - 2$ [36]. Therefore, irreducible trinomials, when trinomials do not exist, quadrinomials or pentanomials are preferred to construct $\mathbb{F}_{3^n}$ [2]. A polynomial basis or a normal basis is used to represent the elements of $\mathbb{F}_{3^n}$. In Chapter 2, the mathematical background about the finite fields and the arithmetic of finite fields are reviewed.

The studies on the hardware and the software implementations of the polynomial basis multiplication in binary extension fields appear intensively in the literature. However, there is no similar interest for the polynomial basis multiplication in general extension fields. In the recent studies such as [23] and [24], Dickson polynomials are used to represent binary extension fields for efficient field multiplication. In [4], Hermite polynomials and Charlier polynomials were proposed to represent binary fields to obtain subquadratic multiplication complexity. In this thesis, we modify these ideas for $\mathbb{F}_{3^n}$ and we present new polynomial basis representations using the orthogonal Charlier polynomials and Hermite polynomials for $\mathbb{F}_{3^n}$. In Chapter 3, we give the Charlier polynomial representation for finite fields of characteristic three and we choose some irreducible binomials to represent the field elements. We present the multiplica-

tion and reduction operations in this representation and we give the complexity results of these operations. For some irreducible Charlier binomials, we obtain better results in reduction complexity according to the standart polynomial representation. We design a sequential multiplier in this representation for the elements of $\mathbb{F}_{3^n}$, where $n \equiv 2 \pmod 3$. We give the cubing operation and we show that by using the irreducible Charlier binomials to represent the elements of $\mathbb{F}_{3^n}$, where $n \equiv 2 \pmod 3$, the reduction in cubing operation requires less additions and scalar multiplications than the reduction in cubing of an element represented with standart polynomials. We also mention about the inversion operation in this representation. In Chapter 4, we represent the elements in $\mathbb{F}_{3^n}$ by using Hermite polynomials and we explain how to obtain an arithmetic design for multiplication and modular reduction. We obtain a set of irreducible binomials in Hermite polynomial representation to get faster modular reduction than in standart polynomial representation. In this representation, we give the multiplication of two elements in $\mathbb{F}_{3^n}$ by using matrix vector multiplication design and we construct reduction matrix. We compute the multiplication and reduction complexities in each representation in view of the number of multiplications and additions. All computations are done by using Maple [29].

Another method for representing finite field elements is the use of normal bases. Especially in hardware implementations of binary fields, the normal basis representation is quite adventageous than the polynomial basis representation [34]. Squaring of an element in the normal basis representation of a binary field is performed simply by a cycle shift of the coordinates of the element, thus it is almost free of cost in hardware. In [27], by Theorem 2.35 it is shown that for any field $\mathbb{F}_q$ and any extension field $\mathbb{F}_{q^n}$, there always exists a normal basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. In [33], optimal normal bases are reported which allow efficient implementations of arithmetic operations in finite fields $\mathbb{F}_{q^n}$ and afterwards all optimal normal bases are determined in [16]. Although there always exists a normal basis for every finite field, it is not valid in the case of optimal normal bases. In [33], the constructions that give all the optimal normal bases and a large family of normal bases of low complexity are given. Dickson polynomials are also used for the construction of optimal normal bases. In [28], the connection between optimal normal bases and Dickson polynomials are presented. In Chapter 5, we recall all these constructions and connections of optimal normal bases and we find a relationship between optimal normal basis elements and Hermite polynomials over $\mathbb{F}_2$. Finally, we give the conclusion part in Chapter 6.

CHAPTER 2

MATHEMATICAL BACKGROUND

This chapter contains preliminaries in finite fields and applications of finite fields. Also the binary field arithmetic is given.

2.1 Finite Fields

In this section, a brief introduction to the theory of finite fields is given. For details and proofs, see [27]. We first give the definition of a field.

Definition 2.1.1. *A field $F = (F, +, \cdot)$ is a set F together with two binary operations on F such that,*

- *F is a ring,*
- *$(F/\{0\}, \cdot)$ forms a commutative group, where 0 is the identity element of the group $(F, +)$.*

Example 2.1.2. *$(\mathbb{R}, +, \cdot)$ and $(\mathbb{Q}, +, \cdot)$ are fields.*

Example 2.1.3. *The set of integers modulo p, where p is a prime, together with addition and multiplication operations forms a field. Since $gcd(a, p) = 1$ for $1 \leq a < p$, then a has a multiplicative inverse in $\mathbb{Z}_p$. This field has finite elements.*

Recall that if a field contains finitely many elements then it is called *finite*. The *order* of a finite field is the number of elements in the field. A finite field of order q is denoted by $\mathbb{F}_q$. Finite fields are also referred to as Galois Fields.

Theorem 2.1.4. *(Existence and uniqueness of finite fields). Finite fields of order q exist if and only if $q = p^m$ for some prime p and some integer $m \geq 1$. Moreover every finite field of order q is unique up to isomorphism.*

If $m = 1$ then $\mathbb{F}_p$ is called a *prime field*. If $m \geq 2$, then $\mathbb{F}_{p^m}$ is called an *extension field*. For a proof see [27]. Now, we give the definition of the characteristic of a finite field.

Definition 2.1.5. *The characteristic of a finite field is the least positive integer n such that $\sum_{i=1}^{n} 1 = 0$.*

It can be shown that the characteristic n of a finite field must be a prime number ≥ 2. Before we give the extension fields in the next section, we introduce the concept of polynomial rings.

Definition 2.1.6. *Let R be a commutative ring, then a polynomial $A(x)$ with an indeterminate x over R is expressed by the following sum $A(x) = a_n x^n + a_{n-1}x^{n-1} + .. + a_0$ where the coefficients a_i's are in R and $n \geq 0$. The largest n, where $a_n \neq 0$ is called the degree of $A(x)$ and denoted by $deg(A(x))$.*

The coefficient a_n is called the *leading coefficient* of $A(x)$, if $a_n = 1$ then $A(x)$ is called a *monic* polynomial. A polynomial is called as a *constant* polynomial if $A(x) = a_0$.

The sum of two polynomials where each has degree n is expressed as

$$A(x) + B(x) = \sum_{i=0}^{n} (a_i + b_i)x^i$$

The product of two polynomials $A(x) = \sum_{i=0}^{n} a_i x^i$ and $B(x) = \sum_{j=0}^{m} b_j x^j$ is defined as follows

$$A(x) \cdot B(x) = \sum_{k=0}^{n+m} \sum_{i+j=k} a_i b_j x^k$$

Definition 2.1.7. *Let R be a commutative ring. The set of polynomials over R with the polynomial addition and multiplication is called a polynomial ring and denoted by $R[x]$.*

In this thesis, we study polynomials over fields and let $F[x]$ denote the polynomial ring over a field F. In this case, the division operation is applicable to the elements of $F[x]$. A polynomial $B(x) \in F[x]$ divides $A(x) \in F[x]$ if there exists a polynomial $C(x) \in F[x]$ such that $A(x) = B(x)C(x)$. $B(x)$ is said to be a *divisor* of $A(x)$ or $A(x)$ is said to be a *multiple* of $B(x)$. Also we can say that $A(x)$ is *divisible* by $B(x)$. The following theorem gives the division algorithm that expresses the divison in $F[x]$.

Theorem 2.1.8. *For any element $B(x) \neq 0$ in $F[x]$ and for any $A(x) \in F[x]$, there exist unique polynomials $Q(x), R(x) \in F[x]$ such that $A(x) = Q(x)B(x) + R(x)$, where $deg(R(x)) < deg(Q(x))$.*

Polynomials in $F[x]$ have many common properties with integers, since division is defined over both sets. For instance, the greatest common divisor of polynomials $A(x)$, $B(x)$ is an analogue to the greatest common divisor of integers a, b and it is denoted by $gcd(A(x), B(x))$. The two polynomials are said to be *relatively prime* if $gcd(A(x), B(x)) = 1$. Euclidean Algorithm is used to compute the greatest common divisor of two polynomials. As in the case of integers, $F[x]$ has prime elements which are called as irreducible polynomials.

Definition 2.1.9. *Let $P(x) \in F[x]$ be a polynomial with $deg(P(x)) > 0$ and $P(x) = B(x)C(x)$, where $B(x), C(x) \in F[x]$. $P(x)$ is called irreducible over F if either $B(x)$ or $C(x)$ is a constant polynomial.*

Definition 2.1.10. *Let $A(x) \in F[x]$ be a polynomial, then an element $\beta \in F$ is called a root of $A(x)$ if $A(\beta) = 0$.*

Remark 2.1.11. *If a polynomial $P(x)$ is irreducible over F, then it does not have a root in F, otherwise it can be written as $P(x) = (x - \beta)Q(x)$.*

As given in Example 2.1.3, $\mathbb{Z}_p$ is a prime field and also denoted by $\mathbb{F}_p$. To construct the extension fields of these fields, irreducible polynomials which are prime elements of $\mathbb{F}_p[x]$ are used, in this manner they are fundamentally important polynomials. Before giving the resulting theorem, we recall the residue class rings.

Let $R[x]$ be a polynomial ring and $A(x) \in R[x]$ be a nonzero polynomial, then $(A(x))$ is a principal ideal and makes a partition of $R[x]$ into disjoint cosets. They are called residue classes modulo $A(x)$ and denoted by $U(x)+(A(x))$, where $U(x) \in R[x]$. Two polynomials $U(x)$ and $V(x)$ are congruent modulo $A(X)$ if their residue classes, $U(x) + (A(x))$ and $V(x) + (A(x))$ are identical, i.e., $A(x)$ divides $(U(x) - V(x))$. For each residue class, a unique representative $N(x) \in R[x]$ with $deg(N(x)) < deg(A(x))$ is used and we can find it as follows. By using the division algorithm, we divide $U(x)$ by $A(x)$ and we get unique polynomials $M(x)$ and $N(x)$ such that $U(x) = M(x)A(x) + N(x)$, where $deg(N(x)) < deg(A(x))$. This is called *reduction* modulo $A(x)$. Now, we can use $N(x) + (A(x))$ uniquely to represent the residue class of polynomials containing $U(x)$. The set of residue classes of polynomials in $R[x]$ with degree less than $deg(A(x))$ is called *residue class ring* of $R[x]$ modulo $A(x)$ and is denoted by $R[x]/(A(x))$.

Theorem 2.1.12. *Let F be a field and $A(x) \in F[x]$. The residue class ring $F[x]/(A(x))$ is a field if and only if $A(x)$ is irreducible over F.*

2.2 Extension Fields

Firstly, we give the definition of an extension field.

Definition 2.2.1. *Let F be a field and H be a subset of F such that H is a field under the same operations. Then H is called a subfield of F and F is called an extension field of H.*

If F is a finite field (also H is a finite field), then F is considered as a vector space over H. The *degree* of F over H is the dimension of the vector space F over H. The other finite fields except prime fields, $\mathbb{F}_p$ are extension fields of these fields. One can say that an extension field contains a prime field as its subfield. A prime field is also called the *base field* of its extension fields. An extension field can be constructed by using an irreducible polynomial of degree n and a base field $\mathbb{F}_p$ and denoted by $\mathbb{F}_{p^n}$ where n is called the extension degree over $\mathbb{F}_p$. The construction is done as follows:

Let $P(x) \in \mathbb{F}_p[x]$ be an irreducible polynomial of degree a positive integer n. Let β be a root of $P(x)$, so $P(\beta) = 0$. Then the residue class ring $\mathbb{F}_p[x]/(P(x))$ is a field and it contains all the poynomials $a_{n-1}\beta^{n-1} + a_{n-2}\beta^{n-2} + .. + a_1\beta^1 + a_0$, where $a_i \in F_p$. The operations in this field are polynomial addition and polynomial multiplication modulo $P(x)$.

For every base field $\mathbb{F}_p$ and every positive integer n there exists an irreducible polynomial in $\mathbb{F}_p[x]$ of degree n, so we can construct any $\mathbb{F}_{p^n}$ by using irreducible polynomials. The following theorem introduces a fact about the roots of an irreducible polynomial.

Theorem 2.2.2. *Let $P(x)$ be an irreducible polynomial in $\mathbb{F}_p[x]$ with degree n. It has a root β in $\mathbb{F}_{q^n}$, moreover all roots β, β^q,..,$\beta^{q^{n-1}}$ are in $\mathbb{F}_{q^n}$.*

The field containing all the roots of an irreducible polynomial $P(x)$ over $\mathbb{F}_q$, is an extension field of $\mathbb{F}_q$ and called as the *splitting field* of $P(x)$ over $\mathbb{F}_q$.

Definition 2.2.3. *If $\mathbb{F}_{q^n}$ is an extension field of $\mathbb{F}_q$ and β is an element in $\mathbb{F}_{q^n}$ then β, β^q,..,$\beta^{q^{n-1}}$ are called the conjugates of β with respect to $\mathbb{F}_q$.*

Now, we give an important mapping from $\mathbb{F}_{q^n}$ to $\mathbb{F}_q$ which uses the conjugates of an element in $\mathbb{F}_{q^n}$.

Definition 2.2.4. *Let $\beta \in F = \mathbb{F}_{q^n}$ and $H = \mathbb{F}_q$, the trace of β from F to H, denoted by $Tr_{F/H}(\beta)$, is defined as*

$$Tr_{F/H}(\beta) = \beta + \beta^q + .. + \beta^{q^{n-1}}$$

We can say that the trace of an element in F over H is the sum of all conjugates of the element with respect to H. We recall that an extension field is a vector space over its base field, so it has a basis as a vector space. The number of basis elements is equal to the extension degree over the base field. If we think about F as a vector space over H, then the trace mapping is a linear transformation from F onto H. In applications of finite fields, the choice of the basis of the field is very important for implementing the operations such as addition, multiplication, computing powers etc. In this thesis, we deal with two major bases of finite fields. Now, we give the definitions of these bases below.

Definition 2.2.5. *Let $P(x) \in \mathbb{F}_q[x]$ be an irreducible polynomial of degree n and β be a root of $P(x)$. Then the set $1, \beta, \beta^2, .., \beta^{n-1}$ is a basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ and it is called as a polynomial basis.*

If a polynomial basis is chosen to represent the finite field $\mathbb{F}_{q^n}$, then field elements are polynomials of degree at most $n-1$. Therefore, addition and multiplication operations of field elements are implemented in the usual way of polynomial addition and polynomial multiplication in $\mathbb{F}[x]$.

The other important type of basis is a normal basis.

Definition 2.2.6. *Let $\alpha \in \mathbb{F}_{q^n}$ be a suitable element such that the set $\left\{\alpha, \alpha^q, .., \alpha^{q^{n-1}}\right\}$ is a basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. This basis is called as a normal basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.*

Using normal basis to represent the elements of finite fields is practical in some implementations such as exponentiation. Namely, computing the q-th power of an element in $\mathbb{F}_{q^n}$ which is represented by a normal basis is just a shifting operation.

Theorem 2.2.7. *For any $n \geq 2$, there is a normal basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.*

Theorem 2.2.7 says that every finite field has a normal basis, so as an alternative to the polynomial bases, one can use normal bases to represent finite fields. Therefore we can note that

there are many bases of an extension field over its any subfield. For a given set containing n elements of $\mathbb{F}_{q^n}$ one can test whether this set is a basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ or not. The following result is one of the methods to determine a basis of $\mathbb{F}_{q^n}$.

Theorem 2.2.8. *The set $\{\beta_1, .., \beta_n\}$ is a basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ if and only if the determinant*

$$\begin{vmatrix} \beta_1 & \dots & \beta_n \\ \beta_1^q & \dots & \beta_n^q \\ \vdots & \ddots & \vdots \\ \beta_1^{q^{n-1}} & \vdots & \beta_n^{q^{n-1}} \end{vmatrix}$$

is nonzero.

2.3 Binary Field Arithmetic

Basic arithmetic operations in binary extension fields $\mathbb{F}_{2^n}$ are commonly used in the applications of cryptography and coding theory [31]. Their efficient implementations are desired for the efficiency of the cryptosystems. In the previous section, we give the bases that are used to represent the finite fields. The choice of the basis has a significant role on the performance of the field arithmetic. Multiplication is the most time consuming operation in field arithmetic. Polynomial basis representation is very efficient for the multiplication methods in comparison to the other basis representations.

We have seen that an element in $\mathbb{F}_{2^n}$ can be represented as a polynomial of degree at most $n-1$. The addition of field elements is performed as polynomial addition or if we write elements as a sequence in $\mathbb{F}_2$ then addition is a bitwise operation, named as XOR operation. Therefore it is a fast operation, however multiplication is not a direct operation as addition. It has a carry propagation and this increases the complexity. The hardware and software implementations of the polynomial basis multiplication are studied extensively such as in [1], [20], [30], [37]. In these studies, irreducible polynomials with low hamming weights and special structures have been chosen. The multiplication of two field elements is performed as polynomial multiplication modulo an irreducible polynomial, so the polynomial basis multiplication has two steps: polynomial multiplication and modular reduction [27].

In hardware implementations, multiplication of finite field elements are processed by serial, digit or parallel multipliers. Space and time complexities are two important criteria to evaluate and compare these multipliers. Space complexity includes the number of coefficient multiplications (ANDs) and the number of coefficient additions (XORs).

2.3.1 Multiplication

There are several methods for multiplication operation. The standart polynomial multiplication is the direct method which is also called as the *schoolbook method*. The following example gives the addition and multiplication of two elements by using schoolbook method, where the elements are represented as sequences in $\mathbb{F}_2$.

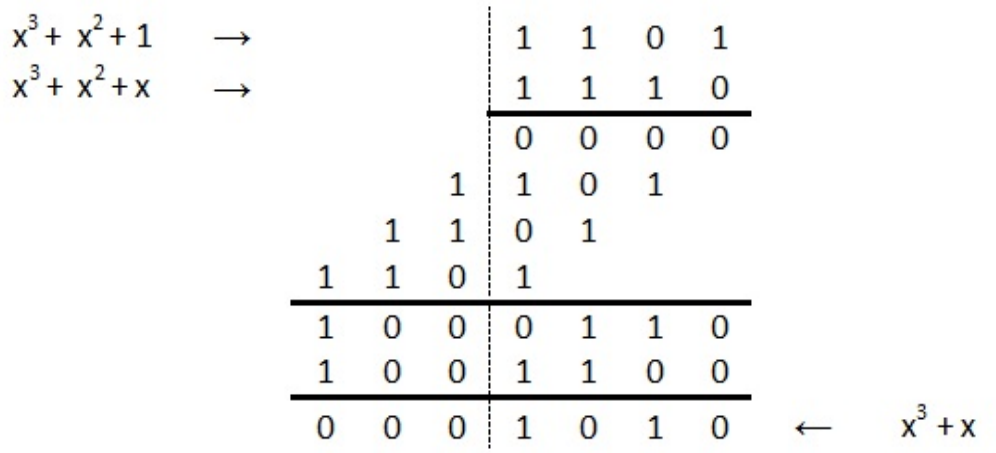

Figure 2.1: Multiplication of two elements in $\mathbb{F}_{2^4}$ by using schoolbook method

Example 2.3.1. *Let $f(x) = x^4 + x + 1 \in \mathbb{F}_2[x]$. Since $f(x)$ is irreducible, then $\mathbb{F}_2[x]/(f(x))$ is isomorphic to $\mathbb{F}_{2^4}$. Let $x^3 + x^2 + 1$ and $x^3 + x^2 + x$ be two elements in $\mathbb{F}_2[x]/(f(x))$. These elements are also written as sequences: (1101) and (1110). The addition of these elements is just the XOR of two sequences which is (0011). The multiplication by the schoolbook method is accomplished by using simply left-shifts and XORs. The resulting sequence is (1010) which gives the polynomial $x^3 + x$. The schoolbook method is illustrated in Figure 2.1.*

Let $A(x)$ and $B(x)$ be two polynomials of degree $n-1$. The schoolbook method has the space complexity to compute $C(x) = A(x)B(x)$ as

- n^2 coefficient multiplications (ANDs)
- $(n-1)^2$ coefficient additions (XORs).

In Example 2.3.1 the multiplication of $x^3 + x^2 + 1$ and $x^3 + x^2 + x$ requires 16 ANDs and 9 XORs.

The complexity of the multiplication of two polynomials is independent of the choice of the irreducible polynomial. There are several different multiplication methods with different complexities. The Karatsuba algorithm has less coefficient multiplications but has more extra additions compared to the schoolbook method. Since multiplication costs more than addition, Karatsuba algorithm is considered to be more efficient than the schoolbook method.

The Karatsuba algorithm is applied in a recursive way and it has a divide-conquer idea [39]. If the degree n of the polynomials is a power of 2, then the space complexity of the algorithm is as follows:

- $n^{log_2 3}$ multiplications,
- $6n^{log_2 3} - 8n + 2$ addititons.

The second step of the multiplication of polynomials is reduction. The reduction complexity is dependent to the irreducible polynomial. In [40], an upper bound for the complexity of the modular reduction is given as $(r-1)(n-1)$ if the irreducible polynomial has degree n and r terms.

In binary extension fields, there is another method proposed alternatively for polynomial multiplication which is described as matrix-vector operations. In one approach, polynomial multiplication part is computed by any multiplication method and the reduction part is done by using a reduction matrix. In another approach, these two parts are performed in one step by using a single matrix which is called Mastrovito matrix.

Mastrovito proposed a bit-parallel multiplier combining the two steps of polynomial multiplication [30]. The complexity of computing the product matrix is proportional to the number of nonzero elements of irreducible polynomials. In [38], it is shown that the general Mastrovito multiplication requires (n^2-1) XORs and n^2 ANDs if the irreducible polynomial is a trinomial $x^n + x^k + 1$. This type of parallel multipliers is the quadratic complexity architectures that requires quadratic space complexity, $O(n^2)$. There is also another type of bit parallel multiplier designs that has subquadratic space complexity, $O(n^m), m < 2$. For large field sizes of $\mathbb{F}_{2^n}$, these architectures that have subquadratic space complexity are practical in hardware implementations of elliptic curve cryptosystems. As we have given the space complexity of the Karatsuba algorithm, it is the most well known algorithm which is used to design subquadratic space complexity multipliers.

In the literature, subquadratic space complexity multipliers designed by using different polynomial bases are proposed for binary extension fields. In [23] and [24], Dickson polynomials are used to represent the finite fields $\mathbb{F}_{2^n}$ and a new scheme is proposed for the design of the subquadratic space complexity parallel multiplier. They use Toeplitz matrix-vector product instead of the polynomial multiplication algorithms and design a new multiplier with lower subquadratic space complexity.

There are also other orthogonal polynomials that are used to represent binary extension fields. In [4], Charlier and Hermite polynomials are presented as the bases of $\mathbb{F}_{2^n}$ to get subquadratic space complexity multipliers.

2.3.2 Squaring

Squaring a polynomial in $\mathbb{F}_2[x]$ is much faster than multiplying two arbitrary polynomials, since it is a linear operation in $\mathbb{F}_{2^n}$, ie. let $A(x) = \sum_{i=0}^{n-1} a_i x^i$, then $A(x)^2 = \sum_{i=0}^{n-1} a_i x^{2i}$. The binary representation of $A(x)^2$ can be obtained by inserting a 0 bit between each consecutive bits of the binary representation of $A(x)$. Then the reduction operation is performed.

In normal basis representation of $\mathbb{F}_{2^n}$, squaring of an element is just a cyclic shift of its coefficients. Let $\left\{\beta, \beta^2, .., \beta^{2^{n-1}}\right\}$ be a normal basis of $\mathbb{F}_{2^n}$, then $A = \sum_{i=0}^{n-1} a_i \beta^{2^i}$ and

$A^2 = \sum_{i=0}^{n-1} a_i \beta^{2^{i+1}}$. Viewing the coefficients of A as a bilinear form $A = (a_0, a_1, .., a_{n-1})$ then A^2 has bilinear form $(a_{n-1}, a_0, .., a_{n-2})$.

2.3.3 Inversion

Inversion in binary extension fields can be performed by using the extended Euclidean algorithm or exponentiation based techniques. In the first inversion approach, the extended Euclidean algorithm is used for computing the greatest common divisor of binary field elements in standart polynomial representation. Let $a(x)$ be a polynomial over $\mathbb{F}_2$. Then $a(x)a^{-1}(x) = 1 \pmod{f(x)}$ where $f(x)$ is the irreducible polynomial of degree n. By using $a(x)$ and $f(x)$ as inputs in the extended Euclidean algorithm, one can find the inverse of $a(x)$, ie. $a^{-1}(x)$ [21]. By this method, inverses of elements in $\mathbb{F}_{2^n}$ can be efficiently computed.

The second inversion approach is based on Fermat's little theorem and some efficient algorithms are given in [1] and [25]. In this method, exponentiation is performed to compute the inverse of a field element. Let α be an element of $\mathbb{F}_{2^n}$, the inverse α^{-1} is computed as follows,

$$\alpha^{-1} = \alpha^{2^n-2} = \alpha^{2^1+2^2+...+2^{n-2}+2^{n-1}}$$

Normal basis representation is prefered in this method, because squaring in normal basis is a cyclic shift in binary extension fields.

CHAPTER 3

CHARLIER POLYNOMIAL REPRESENTATION

In this chapter, we give the Charlier polynomial representation in finite fields of characteristic three. Charlier polynomials are used in the representation of binary fields in [5]. We modify this idea for $\mathbb{F}_{3^n}$. We compute the complexities of multiplication and reduction operations. We design a sequential multiplier and we investigate the cubing operation for finite fields $\mathbb{F}_{3^n}$, where $n \equiv 2 \pmod 3$. We also give the inversion operation in $\mathbb{F}_{3^n}$. We show that by using some specific irreducible Charlier binomials in multiplication and cubing operations we have a lower reduction complexity in Charlier polynomial representation than in standart polynomial representation.

This chapter was presented in [6].

3.1 Charlier Polynomials

In this section, we give preliminaries about Charlier polynomial representation in $\mathbb{F}_{3^n}$.

Definition 3.1.1. *[18] The Charlier polynomials are the monic orthogonal polynomials where* $C_0(x) = 1$, $C_1(x) = x$ *and for* $n \geq 2$

$$C_n(x) = (x - n + 1)C_{n-1}(x)$$

We give the Charlier polynomials in $\mathbb{F}_3[x]$ for $n \leq 10$ in Table 3.1.

Remark 3.1.2. *We note that* $deg(C_n(x)) = n$ *and Charlier polynomials have a recursive structure. By computing* $C_n(x)$*'s for* $n \leq 10$*, one can see that all Charlier polynomials in* $\mathbb{F}_3$ *have the forms where* $k \in \mathbb{N}$ *as follows:*

$$\begin{aligned} C_{3k}(x) &= x^k \cdot (x+2)^k \cdot (x+1)^k \\ C_{3k+1}(x) &= x^{k+1} \cdot (x+2)^k \cdot (x+1)^k \\ C_{3k+2}(x) &= x^{k+1} \cdot (x+2)^{k+1} \cdot (x+1)^k \end{aligned}$$

For notational simplicity, let $C_n(x) = \beta_n$ be the n-th Charlier polynomial in $\mathbb{F}_3[x]$. The multiplication of Charlier polynomials in $\mathbb{F}_3[x]$ is given in Theorem 3.1.3.

Table 3.1: Charlier polynomials in $\mathbb{F}_3[x]$

$C_0(x)$	1
$C_1(x)$	x
$C_2(x)$	$x^2 + 2x$
$C_3(x)$	$x^3 + 2x$
$C_4(x)$	$x^4 + 2x^2$
$C_5(x)$	$x^5 + 2x^4 + 2x^3 + x^2$
$C_6(x)$	$x^6 + x^4 + x^2$
$C_7(x)$	$x^7 + x^5 + x^3$
$C_8(x)$	$x^8 + 2x^7 + x^6 + 2x^5 + x^4 + 2x^3$
$C_9(x)$	$x^9 + 2x^3$
$C_{10}(x)$	$x^{10} + 2x^4$

Theorem 3.1.3. *Let β_n be the n-th Charlier polynomial in $\mathbb{F}_3[x]$, where $n \geq 0$. Then for all $i, j \geq 0$ the Charlier polynomials $\{\beta_0, \beta_1, ..., \beta_{n-1}, ...\}$ satisfy the following equation*

$$\beta_i \cdot \beta_j = \beta_{i+j} + l \cdot (k \cdot \beta_{i+j-1} + 2 \cdot m \cdot \beta_{i+j-2}) \tag{3.1}$$

where $l, k, m \in \mathbb{F}_3$ is defined as

$$l = \begin{cases} 0 & \text{if } i \text{ or } j \equiv 0 \text{ mod } 3 \\ 1 & \text{otherwise.} \end{cases}$$

$$k = \begin{cases} 1 & \text{if } i \equiv j \text{ mod } 3 \\ 2 & \text{otherwise.} \end{cases}$$

$$m = \begin{cases} 1 & \text{if } i, j \equiv 2 \text{ mod } 3 \\ 0 & \text{otherwise.} \end{cases}$$

Proof. We have formalized the Charlier polynomials in Remark 3.1.2. Now, we compute all the cases with respect to the residues of i and j (mod 3), respectively. Since $\beta_{3k} = x^k \cdot (x+2)^k \cdot (x+1)^k$ and the powers of x, $(x+2)$, $(x+1)$ are same, then it is obvious that by multiplying any β_i and β_{3k} we get β_{i+3k}. If we compute the other cases, ie. $\beta_i \cdot \beta_j$ where $i, j \equiv 1$ or 2 (mod 3), we get the following equations:

$$\begin{aligned} \beta_{3k+1} \cdot \beta_{3k+1} &= \beta_{6k+2} + \beta_{6k+1} \\ \beta_{3k+1} \cdot \beta_{3k+2} &= \beta_{6k+3} + 2\beta_{6k+2} \\ \beta_{3k+2} \cdot \beta_{3k+2} &= \beta_{6k+4} + \beta_{6k+3} + 2\beta_{6k+2} \end{aligned}$$

If we combine all these cases, then we get the Equation 3.1. As a result, we can say that the multiplication of Charlier polynomials, β_i and β_j changes with respect to the indices i,j (mod 3). One can also use the induction method to prove the theorem. ∎

Now, we represent the elements of finite field $\mathbb{F}_{3^n}$ by using Charlier polynomials in $\mathbb{F}_3[x]$. We first use the elements represented with standart polynomial representation in $\mathbb{F}_{3^n}$. We recall the standart polynomial basis representation in Section 2.2. Let $a(x) = a'_{n-1}x^{n-1} + a'_{n-2}x^{n-2} + .. + a'_0$, where $a'_i \in \mathbb{F}_3$. $a(x)$ can be represented by using Charlier polynomials as $a = a_{n-1}\beta_{n-1} + .. + a_0\beta_0$, where $a_i \in \mathbb{F}_3$. Algorithm 3 in Appendix A gives the way of conversion of the coefficients from standart polynomial representation to Charlier polynomial representation.

3.1.1 Charlier Basis

In Section 2.2 we mention that the construction of any extension fields can be done by using irreducible polynomials. Similarly, we can say that any finite field $\mathbb{F}_{3^n}$ is isomorphic to $\mathbb{F}_3[x]/(f(x))$, where $f(x)$ is an irreducible polynomial of degree n in $\mathbb{F}_3[x]$. The standart representation of elements of $\mathbb{F}_3[x]/(f(x))$ is done by using the polynomial basis $\{1,\ x,\ x^2, ..., x^{n-1}\}$. $\mathbb{F}_{3^n}$ is an extension field of $\mathbb{F}_3$ and also considered as a vector space over $\mathbb{F}_3$, therefore it should contain n basis elements.

The Charlier polynomial representation is done by using $\{\beta_0, \beta_1, ..., \beta_{n-1}\}$ in the same way. It is obvious that this set is linearly independent. By using Algorithm 3, we can convert elements from standart polynomial representation into Charlier polynomial representation and we write each element uniquely as a linear combination of $\{\beta_0, \beta_1, .., \beta_{n-1}\}$. As a result, we write down the following remark.

Remark 3.1.4. *Let $f = f_n\beta_n + .. + f_0\beta_0$ be an irreducible polynomial of degree n where each $f_i \in \mathbb{F}_3$. The set $\{\beta_0, \beta_1, .., \beta_{n-1}\}$ is a basis of $\mathbb{F}_{3^n} \cong \mathbb{F}_3[x]/f(x)$.*

3.2 Multiplication in Charlier Representation

In this section, we describe the multiplication of field elements represented with Charlier polynomials and explore the complexity of the multiplication. In Section 2.3 we give the multiplication in binary fields. In the same way, multiplication of finite field elements in $\mathbb{F}_{3^n}$ can be performed in two steps: multiplication of polynomials and then modular reduction with respect to the irreducible polynomial that is chosen before. We give the multiplication and the reduction operations, respectively. Theorem 3.2.1 gives the required number of multiplications and additions to multiply polynomials in Charlier basis where $M(n)$ and $A(n)$ denote the minimum number of multiplications and the minimum number of additions for the corresponding algorithm for multiplication of two n-term polynomials.

Theorem 3.2.1. *Let $a = a_{n-1}\beta_{n-1} + ... + a_0\beta_0$ and $b = b_{n-1}\beta_{n-1} + ... + b_0\beta_0$ be n-term polynomials over $\mathbb{F}_3$ and $a \cdot b = c = c_{2n-2}\beta_{2n-2} + ... + c_0\beta_0$. By using any multiplication method, the coefficients of the polynomial c are computed with*

$$M(n) \quad + \quad M(n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + 3 \cdot M(\ \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor\ )$$

$$+ \quad 4 \cdot \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor \cdot (n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor)$$

multiplications and

$$\begin{aligned} A(n) \quad &+ \quad A(n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + 2 \cdot A(\lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) \\ &+ \quad 2 \cdot \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor \cdot (n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + 3 \end{aligned}$$

additions.

Proof. By using Theorem 3.1.3, the coefficients are computed as follows,

$$\begin{aligned} c_0 &= a_0 b_0 \\ c_1 &= a_0 b_1 + a_1 b_0 + a_1 b_1 \\ c_2 &= a_0 b_2 + a_2 b_0 + a_1 b_1 + 2a_2 b_1 + 2a_1 b_2 + 2a_2 b_2 \\ &\vdots \\ c_{2n-3} &= a_{n-2} b_{n-1} + a_{n-1} b_{n-2} + a_{n-1} b_{n-1} \\ c_{2n-2} &= a_{n-1} b_{n-1} \end{aligned}$$

If we compare this multiplication with standard polynomial basis representation, there are some extra terms. All these terms come from the multiplication of the basis elements $\beta_i \cdot \beta_j$, where $0 \leq i, j \leq n-1$ and $i, j \not\equiv 0 \pmod 3$. The multiplication differs with respect to the values of $i \pmod 3$ and $j \pmod 3$, i.e., if $i, j \equiv 1 \pmod 3$ then $\beta_i \cdot \beta_j = \beta_{i+j} + \beta_{i+j-1}$ or if $i, j \equiv 2 \pmod 3$ then $\beta_i \cdot \beta_j = \beta_{i+j} + \beta_{i+j-1} + 2 \cdot \beta_{i+j-2}$. Therefore, the number of the extra terms is related to the number of the indices which are smaller than n and equal to 1 or 2 (mod 3). The number of indices that are equal to 2 (mod 3) is $\lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor$ and the number of indices that are equal to 1 (mod 3) is $n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor$. The extra terms are computed with the multiplication of the polynomials contains those numbers of terms, so the total multiplication complexity is determined as the sum of these multiplications.

Similarly, in the total addition complexity, we have additions to combine these extra terms to the ordinary multiplication terms. These are related to the indices of c_i, where $0 \leq i \leq 2n-4$ and the values of these indices modulo 3. ■

Remark 3.2.2. *Some of this multiplication complexity comprises scalar multiplication, ie. multiplication by two. The number of scalar multiplication is:*

$$2 \cdot \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor \cdot (n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + M(\lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor)$$

In hardware implementations of finite fields of characteristic three, multiplication by two is equivalent to the negation operation, so we can perform a subtraction operation in place of multiplication by two [35].

By the choice of the multiplication method, some or all elements of extra terms may be computed in the first part of the algorithm, ie. in n-term polynomial product, therefore the complexities added to $M(n)$ and $A(n)$ may be smaller than the given ones in Theorem 3.2.1. Hence,

Theorem 3.2.1 gives the upper bound for the complexity of the multiplication of two elements in Charlier basis representation. We explain the theorem with an example by using the Karatsuba multiplication method. We give the complexity of the Karatsuba algorithm in Section 2.3. Recall that Karatsuba algorithm decreases the number of coefficient multiplications compared to the schoolbook or ordinary multiplication method by using the divide-conquer idea recursively. Example 3.2.3 shows the required multiplications for 4-term polynomials over $\mathbb{F}_p$ where $p \geq 2$.

Example 3.2.3. *Let $a(x) = a_3x^3 + ... + a_0$ and $b(x) = b_3x^3 + ... + b_0$ be 4-term polynomials over $\mathbb{F}_p$ where $p \geq 2$. Karatsuba algorithm computes the product $c(x) = c_6x^6 + ... + c_0$ with the following multiplications:*

$$\begin{aligned}
m_0 &= a_0b_0 \\
m_1 &= a_1b_1 \\
m_2 &= a_2b_2 \\
m_3 &= a_3b_3 \\
m_4 &= (a_0 + a_1)(b_0 + b_1) \\
m_5 &= (a_0 + a_2)(b_0 + b_2) \\
m_6 &= (a_1 + a_3)(b_1 + b_3) \\
m_7 &= (a_2 + a_3)(b_2 + b_3) \\
m_8 &= (a_0 + a_1 + a_2 + a_3)(b_0 + b_1 + b_2 + b_3)
\end{aligned}$$

By using appropriate additions of m_i's the coefficients of c are obtained. In this example, Karatsuba algorithm needs 9 multiplications and 24 additions [39].

Now, in Example 3.2.4, we multiply two 4-term polynomials in Charlier basis representation by using Karatsuba multiplication method.

Example 3.2.4. *Let $a = a_3\beta_3 + a_2\beta_2 + a_1\beta_1 + a_0\beta_0$ and $b = b_3\beta_3 + b_2\beta_2 + b_1\beta_1 + b_0\beta_0$ be 4-term polynomials over $\mathbb{F}_3$. Let $a \cdot b = c = c_6\beta_6 + c_5\beta_5 + ... + c_0\beta_0$. Then*

$$\begin{aligned}
c_0 &= a_0b_0 \\
c_1 &= a_0b_1 + a_1b_0 + \underline{a_1b_1} \\
c_2 &= a_0b_2 + a_2b_0 + a_1b_1 + \underline{2a_2b_1} + \underline{2a_1b_2} + \underline{2a_2b_2} \\
c_3 &= a_0b_3 + a_3b_0 + a_1b_2 + a_2b_1 + \underline{a_2b_2} \\
c_4 &= a_1b_3 + a_3b_1 + a_2b_2 \\
c_5 &= a_2b_3 + a_3b_2 \\
c_6 &= a_3b_3
\end{aligned}$$

The extra terms in this multiplication are a_1b_1, $2a_2b_1 + 2a_1b_2 + 2a_2b_2$ and a_2b_2. By applying Karatsuba algorithm for performing an ordinary polynomial multiplication as in Example

3.2.3, we compute the terms except these extra terms by using 9 *multiplications and* 24 *additions. Now we compute the cost of the extra terms.*

We already have $m_1 = a_1b_1$ *and* $m_2 = a_2b_2$*. We compute* $2m_2$ *to obtain* $2a_2b_2$*, so we need one multiplication. To compute* $2a_2b_1 + 2a_1b_2$*, we do*

$$2 \cdot [(a_1 + a_2) \cdot (b_1 + b_2) - a_1b_1 - a_2b_2] = 2 \cdot [(a_1 + a_2) \cdot (b_1 + b_2) - m_1 - m_2]$$

The cost is 2 *multiplications and* 4 *additions. Also we need* 4 *additions to add these terms to the general result. So we need totally* 3 *multiplications and* 8 *additions as an extra cost. As a result, by using Karatsuba algorithm we need* $9 + 3 = 12$ *multiplications and* $24 + 8 = 32$ *additions to multiply* $c = a \cdot b$ *in Charlier polynomial representation.*

In Theorem 3.2.1, we have given the upper bound for the complexity of the multiplication of two elements in Charlier basis representation. Now, we explore the reduction part.

3.2.1 Irreducible Charlier Binomials

Recall that the Charlier polynomials in $\mathbb{F}_3$ are given in Table 3.1 for $n \leq 10$. Because of the recursive structure of the Charlier polynomials, we can say that the polynomials including constant terms are only β_0. We explore low weight irreducible Charlier polynomials for the performance of the reduction operation, so we look for the Charlier binomials. Since irreducible polynomials should include the constant term, there is only one form of the Charlier binomials that is $\beta_n + \beta_0$ for some integer $n \geq 1$. We give some irreducible Charlier binomials in Table $B.1$ in Appendix B.

3.2.1.1 Reduction

We give the reduction operations with respect to each form of the irreducible Charlier binomials in Appendix B. They are different due to the residues of the indices n in mod 3, since the multiplication of Charlier polynomials differs with respect to the values of the indices mod 3. Due to the reduction complexities in Appendix B, the binomials $\beta_n + \beta_0$, where $n \equiv 0 \pmod 3$ has the least number of multiplications and additions. However there is only one binomial in this form which is $\beta_3 + \beta_0$. From the implementation point of view, one cannot find an irreducible Charlier binomial in this form to represent the finite fields $\mathbb{F}_{3^n}$ except $\beta_3 + \beta_0$. Therefore we choose the binomials $\beta_n + \beta_0$, where $n \equiv 2 \pmod 3$ from Table $B.2$ in Appendix B since they have the least number of multiplications and additions after the binomials $\beta_n + \beta_0$, where $n \equiv 0 \pmod 3$. By the following theorem we give the total reduction complexity by using the binomial $\beta_n + \beta_0$, where $n \equiv 2 \pmod 3$.

Theorem 3.2.5. *Let* $f = \beta_n + \beta_0$*, where* $n \equiv 2$ *(mod* 3*) and* $a \cdot b = c' = c'_{2n-2}\beta_{2n-2} + c'_{2n-3}\beta_{2n-3} + ... + c'_0\beta_0$ *be the product of* a *and* b *in* $\mathbb{F}_{3^n}$*, where* $c'_i \in \mathbb{F}_3$*. Then the reduction of* $c = c'$ *(mod* f*) requires* $\frac{5}{3}n - \frac{7}{3}$ *additions and* $\frac{4}{3}n - \frac{5}{3}$ *scalar multiplications.*

Proof. The reduction operation is performed due to the terms of c' with indices $n \leq i \leq 2n-2$. We compute the reduced forms of β_i's for $n \leq i \leq 2n - 2$ by using the reduction formula that

we give in Appendix B.

$$\begin{aligned}
\beta_n &= 2\beta_0 \\
\beta_{n+1} &= 2\beta_1 + 2\beta_0 \\
\beta_{n+2} &= 2\beta_2 + \beta_1 \\
&\vdots \\
\beta_{2n-3} &= 2\beta_{n-3} + \beta_{n-4} \\
\beta_{2n-2} &= 2\beta_{n-2}
\end{aligned}$$

The coefficients c_i' of each β_i on the left of the equations are added to the coefficients c_j' of each β_j's on the right, where $0 \leq j \leq n-1$. So the number of additions is equal to the total number of β_i's on the right side of the equations. There are $n-1$ equations. Starting from the first equation, for each three equations there are five β_j's on the right side except the last equation. The number of these three iterative equations is $\frac{n-2}{3}$, then the number of terms on the right hand side is $5\left(\frac{n-2}{3}\right)$. Also there is one more term in the last equation. So the total number of terms is $5\left(\frac{n-2}{3}\right)+1$. In each three iterative equations there are four scalar multiplications, so from all of three iterative equations, we have $4\left(\frac{n-2}{3}\right)$ scalar multiplications. Also we have one multiplication in the last equation.

Totally, the number of additions is $\frac{5}{3}n - \frac{7}{3}$ and the number of scalar multiplications is $\frac{4}{3}n - \frac{5}{3}$. ■

Now, we give the general reduction complexity in standart polynomial representation. In [15], the reduction complexity in standart polynomial representation is given for binary extension fields and in [40] it is given for general extension fields. The following theorem gives the reduction complexity in standart polynomial representation for $\mathbb{F}_{q^n}$.

Theorem 3.2.6. *[40] Let $f(x)$ be a degree n monic irreducible polynomial with r nonzero terms in $\mathbb{F}_q[x]$. Then the reduction with respect to $f(x)$ in standart polynomial representation requires $(r-1)(m-1)$ additions and at most $(r-1)(m-1)$ scalar multiplications over $\mathbb{F}_q$.*

Remark 3.2.7. *In Charlier polynomial representation, we deal with irreducible binomials $\beta_n + \beta_0$, where $n \equiv 2 \pmod 3$ to decrease the reduction complexity. For the comparison, we choose irreducible trinomials with degree n as low weight polynomials in standart polynomial representation. Recall that in [17], it's proved that there is always a trinomial or quadrinomial for the extension degree $n \leq 539$.*

We give Example 3.2.8 to compare the reduction operation in Charlier polynomial representation with the reduction operation in standart polynomial representation.

Example 3.2.8. *Let $f = \beta_5 + \beta_0$ be an irreducible Charlier binomial over $\mathbb{F}_3$. We choose $x^5 + 2x + 1$ over $\mathbb{F}_3$ as irreducible trinomial in standart polynomial representation. Now, we compute the reduction complexities with respect to these polynomials in each representation, respectively.*

We reduce the elements β_i, where $5 \leq i \leq 8$ as follows:

$$\begin{aligned}\beta_5 &= 2\beta_0 \\ \beta_6 &= 2\beta_1 + 2\beta_0 \\ \beta_7 &= 2\beta_2 + \beta_1 \\ \beta_8 &= 2\beta_3\end{aligned}$$

Then we have 6 additions and 5 scalar multiplications as total reduction complexity. By using Theorem 3.2.5 we can also compute the reduction complexity. The number of additions is $\frac{5}{3}5 - \frac{7}{3} = 6$ and the number of multiplications is $\frac{4}{3}5 - \frac{5}{3} = 5$.

We reduce the elements x^i, where $5 \leq i \leq 8$ as follows:

$$\begin{aligned}x^5 &= x + 2 \\ x^6 &= x^2 + 2x \\ x^7 &= x^3 + 2x^2 \\ x^8 &= x^4 + 2x^3\end{aligned}$$

Then we have 8 additions and 4 scalar multiplications as total reduction complexity. By using Theorem 3.2.6 we compute the reduction complexity as $(3-1)(5-1) = 8$ additions and at most 8 scalar multiplications.

Remark 3.2.9. *For the comparison, we choose an irreducible trinomial $x^n + ax + b$ in $\mathbb{F}_3[x]$. By using the Theorem 3.2.6, the reduction with respect to this trinomial requires $2n-2$ additions and at most $2n-2$ multiplications. In Table 3.2, we give the comparison of the reduction complexities in Charlier polynomial representation and standart polynomial representation.*

Table 3.2: Reduction Complexity

Form	# Additions	# Constant Multiplications
Charlier Binomial, $\beta_n + \beta_0$ ($n \equiv 2 \bmod 3$)	$\frac{5}{3}n - \frac{7}{3}$	$\frac{4}{3}n - \frac{5}{3}$
Polynomial Basis, $x^n + ax + b$	$2n-2$	$2n-2$

3.3 Design of Arithmetic Operations in Charlier Polynomial Representation

In previous section, we have shown that if we choose an irreducible Charlier binomial $\beta_n + \beta_0$ where $n \equiv 2 \pmod 3$, we have a lower reduction complexity than an irreducible trinomial in standart polynomial representation. Therefore, in this section we study the arithmetic operations in finite fields $\mathbb{F}_{3^n}$, where $n \equiv 2 \pmod 3$ and we use the irreducible Charlier binomials $\beta_n + \beta_0$, where $n \equiv 2 \pmod 3$ to construct these fields.

We give the algorithms of multiplication and reduction, then we indicate the computations for cubing of the elements of $\mathbb{F}_{3^n}$ represented with Charlier polynomial representation.

3.3.1 A Sequential Multiplier

We give the multiplication of the elements of $\mathbb{F}_{3^n}$ where $n \equiv 2 \pmod 3$ with a sequential model in Algorithm 1. It is similar to the left-to-right serial multiplication in an algebraic field. By Equation 3.1, multiplication of two basis elements β_i and β_j produces up to three terms with respect to the values i and j modulo 3. The first term in all forms is β_{i+j} which is similar to the multiplication algorithm using standart polynomial basis. We give the product of two polynomials in standart polynomial representation in Section 2.1. In Charlier representation the product of two elements a and b can be written as

$$c = a \cdot b = \left(\sum_{i=0}^{n-1} a_i \beta_i\right) \cdot \left(\sum_{j=0}^{n-1} b_j \beta_j\right) = \sum_{i=0}^{n-1} a_i \cdot \left(\sum_{j=0}^{n-1} b_j \beta_i \beta_j\right)$$

As we have noted that we take the elements of $\mathbb{F}_{3^n}$ where $n \equiv 2 \pmod 3$. We can express the product of two elements with respect to the multiplication of Charlier polynomials, where $n \equiv 2 \pmod 3$ as follows:

$$\begin{aligned}
c &= \sum_{i=0}^{n-1} a_i \cdot \left(\sum_{j=0}^{n-1} b_j \beta_{i+j}\right) \\
&+ \sum_{s=0}^{\left(\frac{n-2}{3}\right)} a_{3s+1} \cdot \left(\sum_{t=0}^{\left(\frac{n-2}{3}\right)} b_{3t+1}\beta_{(3s+1)+(3t+1)-1} + 2\sum_{t=0}^{\lfloor\frac{n-3}{3}\rfloor} b_{3t+2}\beta_{(3s+1)+(3t+2)-1}\right) \\
&+ \sum_{s=0}^{\lfloor\frac{n-3}{3}\rfloor} a_{3s+2} \cdot \left(2\sum_{t=0}^{\left(\frac{n-2}{3}\right)} b_{3t+1}\beta_{(3s+2)+(3t+1)-1}\right) \\
&+ \sum_{s=0}^{\lfloor\frac{n-3}{3}\rfloor} a_{3s+2} \cdot \left(\sum_{t=0}^{\lfloor\frac{n-3}{3}\rfloor} b_{3t+2}\beta_{(3s+2)+(3t+2)-1} + 2\sum_{t=0}^{\lfloor\frac{n-3}{3}\rfloor} b_{3t+2}\beta_{(3s+2)+(3t+2)-2}\right)
\end{aligned}$$

To implement the algorithm, we take four registers B, B^1, B^{20} and B^{21} initialized with the coefficients of b_i's. In B all coefficients of b are taken, this register corresponds to the first summand of the sum. Then the register B is shifted by one to the right. The coefficients of b_i with indices $i \equiv 1 \pmod 3$ in the shifted register B remain same and the other coefficients vanish. By this way we obtain the register B^1. Similarly, to generate B^{20} the coefficients of b with indices $i \equiv 2 \pmod 3$ are taken in the shifted register B and the rest of them vanish. The last register B^{21} is obtained by shifting the register B^{20} by one to the right. In Algorithm 1, we get the coefficients of the extra terms that come from the Charlier polynomial multiplication by multiplying the coefficients of a and the registers B^1, B^{20} and B^{21}.

We note that the registers B^1, B^{20} and B^{21} given in Algorithm 1 are initialized for the multiplication of elements in $\mathbb{F}_{3^n}$ where $n \equiv 2 \pmod 3$. For other fields, the initialization of the registers B^1, B^{20} and B^{21} only differ in Algorithm 1.

The second part of the multiplication is the reduction operation. Algorithm 2 gives the reduction with respect to $f = \beta_n + \beta_0$ where $n \equiv 2 \pmod 3$. In this algorithm, we take three registers

Algorithm 1 Sequential Multiplication of Elements in Charlier Representation

Input: $a = \sum_{i=0}^{n-1} a_i \beta_i$ and $b = \sum_{j=0}^{n-1} b_j \beta_j$
Output: $C = ab$
Variables: $B[2n-2...0], B^1[2n-2...0], B^{20}[2n-2...0], B^{21}[2n-2...0]$, $C[2n-2...0]$
$\{initialization\}$
$B[2n-2...n-1] \leftarrow (b_{n-1}, .., b_1, b_0)$
$B^1[2n-2...n-1] \leftarrow (0, b_{n-1}, 0, 0, b_{n-4}, 0, 0, ..., b_1)$
$B^{20}[2n-2...n-1] \leftarrow (0, 0, 0, b_{n-3}, 0, 0, b_{n-6}, ..., b_2, 0)$
$B^{21}[2n-2...n-1] \leftarrow (0, 0, 0, 0, b_{n-3}, 0, 0, b_{n-6}, ..., 0, b_2)$

$\{main\ loop\}$

for $i = n-1$ downto 0 **do**
$C[2n-2...0] \leftarrow C[2n-2...0] + a_i \cdot B[2n-2...0]$
  if $i \equiv 1$ (mod 3) then
$C[2n-2...0] \leftarrow C[2n-2...0] + a_i \cdot B^1[2n-2...0]$
$C[2n-2...0] \leftarrow C[2n-2...0] + 2a_i \cdot B^{20}[2n-2...0]$
  end if
  if $i \equiv 2$ (mod 3) then
$C[2n-2...0] \leftarrow C[2n-2...0] + 2a_i \cdot B^1[2n-2...0]$
$C[2n-2...0] \leftarrow C[2n-2...0] + a_i \cdot B^{20}[2n-2...0]$
$C[2n-2...0] \leftarrow C[2n-2...0] + 2a_i \cdot B^{21}[2n-2...0]$
  end if
shift right B,
shift right B^1,
shift right B^{20},
shift right B^{21},
end for
return C

R, R^0 and R^1 initialized with the coefficients c_i where $n \leq i \leq 2n-2$. The register R contains all these coefficients. Then the register R is shifted by one to the right. The coefficients c_i with indices $i \equiv 0 \pmod 3$ in the shifted register R remain same and the other coefficients vanish. By this way we obtain the register R^0. Similarly, to generate R^1 the coefficients of c with indices $i \equiv 1 \pmod 3$ are taken in the shifted register R and the rest of them vanish. Recall that the arithmetic complexities of these algorithms are given in Section 3.2.

Algorithm 2 Reduction

Input: $c \in \mathbb{F}_{3^n}$ where $c = \sum_{i=0}^{2n-2} c_i \beta_i$
Output: $c \pmod f$
Variables: $B[n-2...0], C[2n-2...0]$
$\{initialization\}$
$R[n-2...0] \leftarrow (c_{2n-2}, ..., c_{n+1}, c_n)$
$R^0[n-2...0] \leftarrow (0, 0, 0, c_{2n-4}, 0, 0, c_{2n-7}, ..., 0, c_{n+1})$
$R^1[n-2...0] \leftarrow (0, 0, c_{2n-3}, 0, 0, c_{2n-6}, ..., c_{n+2}, 0)$

1: $C[n-2...0] \leftarrow C[n-2...0] + 2R[n-2...0] + 2R^0[n-2...0] + R^1[n-2...0]$
2: **return** $C[n-1...0]$

3.3.2 Cubing

Cubing is a linear operation in finite fields of characteristic three. In standart polynomial representation, cubing over $\mathbb{F}_{3^n}$ consists in reducing the following expression modulo $f(x)$:

$$c(x) = a(x)^3 \ mod \ (f(x)) = \sum_{i=0}^{n-1} a_i x^{3i} \ mod \ (f(x))$$

In Charlier polynomial representation, the cubing of an element $a = a_{n-1}\beta_{n-1} + .. + \beta_0$ in $\mathbb{F}_{3^n}$ can be computed as,

$$c = a^3 \ mod \ (f) = \sum_{i=0}^{n-1} a_i \beta_i^3 \ mod \ (f)$$

where the reduction polynomial is f. By using the Equation 3.1, we compute the cubes of the basis elements in three equations with respect to their indices. We find the following results for $0 \leq k \leq \lfloor \frac{n-1}{3} \rfloor$:

$$\begin{aligned} \beta_{3k}^3 &= \beta_{9k} \\ \beta_{3k+1}^3 &= \beta_{9k+3} + \beta_{9k+1} \\ \beta_{3k+2}^3 &= \beta_{9k+6} + 2\beta_{9k+4} + 2\beta_{9k+3} + \beta_{9k+2} \end{aligned}$$

In cubing operation multiplication is performed iteratively, therefore we have extra terms coming from Charlier polynomial multiplication as it can be seen in above equations. By using an appropriate multiplication method the number of these extra terms can be decreased. However,

the current multiplication methods cannot completely reduce the extra cost in the multiplication or cubing operations in Charlier polynomial representation. These operations have quite fewer complexities than the same operations in standart polynomial representation.

Now, we examine the reduction part of the cubing operation in Charlier polynomial representation. Different than the reduction operation in multiplication, in cubing we reduce some terms into the interval $[0, n-1]$ by applying reduction two times. In this part, we again use $f = \beta_n + \beta_0$, where $n \equiv 2 \pmod 3$ as the reduction polynomial since it has a better reduction complexity. By using the reduction formula two times for $\beta_n + \beta_0$, where $n \equiv 2 \pmod 3$ given in Appendix B we get the following equations for each β_i where $2n \leq i \leq 3n-3$ and i is equal to zero, one and two (mod 3), respectively:
If $i \equiv 0 \pmod 3$:

$$\beta_i \ (mod\ f) = \beta_{i-2n} + \beta_{i-2n-2} \tag{3.2}$$

If $i \equiv 1 \pmod 3$:

$$\beta_i \ (mod\ f) = \beta_{i-2n} + 2\beta_{i-2n-1} + \beta_{i-2n-2} \tag{3.3}$$

If $i \equiv 2 \pmod 3$:

$$\beta_i \ (mod\ f) = \beta_{i-2n} + \beta_{i-2n-1} \tag{3.4}$$

To compare the reduction with standart polynomial representation by using an irreducible trinomial f and applying reduction two times with respect to this trinomial, we give the reduced form of x^i where $2n \leq i \leq 3n-3$ as follows:

$$x^i \ (mod\ f) = x^{i-2n+2} + 2x^{i-2n+1} + x^{i-2n} \tag{3.5}$$

Apart from the extra terms coming from the cubing operation of an element, the terms β_i in the cube of an element have indices $i \equiv 0 \pmod 3$ and in standart polynomial representation the terms in the cube of an element have exponents $i \equiv 0 \pmod 3$. Therefore, if we compare the reductions in Equation 3.2 and Equation 3.5 we can see that in Charlier representation the reduction of one term requires one less addition and one less scalar multiplication. If we evalute the total reduction complexity, since in each representation the number of reduced terms will be equal then we can say that the total reduction complexity in Charlier representation will be less than the total reduction complexity in standart polynomial representation.

3.3.3 Inversion

We have given inversion operation for binary extension fields in Section 2.3.3. The same methods are used in finite fields of characteristic three. The extended Euclidean algorithm is performed for the inversion of an element in standart polynomial representation of $\mathbb{F}_{3^n}$ and as an alternative way the exponentiation based algorithms are used in normal basis representation.

Since we deal with the Charlier polynomial representation, we investigate the Euclidean algorithm approach for the inversion in this representation. The extended Euclidean algorithm for binary polynomials in [21] is adapted to be used in finite fields of characteristic three [3]. In this algorithm the most used operations are addition and multiplication. As we give the multiplication complexity in Theorem 3.2.1, we have extra cost in multiplication of elements

in Charlier polynomial representation than in standart polynomial representation. Let $a(x)$ be a polynomial over $\mathbb{F}_3$ in standart polynomial representation, to compute the inverse of $a(x)$ which is denoted by $a^{-1}(x)$ we propose that the extended Euclidean algorithm is implemented in standart polynomial representation and then the inverse $a^{-1}(x)$ can be converted to the Charlier polynomial representation. By this way, we can prevent the complexity of the inversion to be increased by the Charlier polynomial multiplication.

Finally in this chapter, we emphasize that if we use an appropriate multiplication method and reduce the number of extra terms coming from the multiplication of Charlier polynomials, then we also reduce the complexities in the cubing and inversion operations. Apart from this, in the reduction part of both multiplication and cubing operations by using the irreducible Charlier binomial $\beta_n + \beta_0$, where $n \equiv 2$, we have already less reduction complexity than the standart polynomial representation where an irreducible trinomial is used.

CHAPTER 4

HERMITE POLYNOMIAL REPRESENTATION

In this chapter, we present Hermite polynomials to represent finite field elements in $\mathbb{F}_{3^n}$. We obtain a set of irreducible binomials in Hermite polynomial representation to get faster modular reduction. In this representation, we give the multiplication of two elements in $\mathbb{F}_{3^n}$ by using matrix-vector multiplication design and we construct reduction matrix.

The work done in this chapter is partially presented in [7] and [8] and included in [9].

4.1 Hermite Polynomials and Hermite Basis in $\mathbb{F}_{3^n}$

In this section, we give some preliminaries and Hermite polynomial representation of finite fields of characteristic three.

Definition 4.1.1. *[14] The probabilists Hermite polynomials are $H_0(x) = 1$, $H_1(x) = x$ and for $n \geq 2$*

$$H_n(x) = x \cdot H_{n-1}(x) - (n-1) \cdot H_{n-2}(x)$$

We give the Hermite polynomials in $\mathbb{F}_3[x]$ for $n \leq 10$ in Table 4.1.

Remark 4.1.2. *We note that $deg(H_n(x)) = n$. Since Hermite polynomials have a recursive structure, it is easy to show that all Hermite polynomials in $\mathbb{F}_3$ have the forms*

$$\begin{aligned} H_{3k}(x) &= x^{3k} \\ H_{3k+1}(x) &= x^{3k+1} \\ H_{3k+2}(x) &= x^{3k+2} + 2x^{3k} \end{aligned}$$

for $k \in \mathbb{N}$.

Let $H_n(x) = \beta_n$ be the n-th Hermite polynomial in $\mathbb{F}_3[x]$ with degree n. Now, we define the multiplication operation on Hermite polynomials over $\mathbb{F}_{3^n}$.

Theorem 4.1.3. *Let $H_n(x) = \beta_n$ be the n-th Hermite polynomial in $\mathbb{F}_3[x]$, where $n \geq 0$. Then for all $i, j \geq 0$ the Hermite polynomials $\{\beta_0, \beta_1, ..., \beta_{n-1}, ..\}$ satisfy the following equation*

$$\beta_i \cdot \beta_j = \beta_{i+j} + l \cdot (k \cdot \beta_{i+j-2} + 2 \cdot m \cdot \beta_{i+j-4}) \tag{4.1}$$

Table 4.1: Hermite polynomials in $\mathbb{F}_3[x]$

$H_0(x)$	1
$H_1(x)$	x
$H_2(x)$	x^2+2
$H_3(x)$	x^3
$H_4(x)$	x^4
$H_5(x)$	x^5+2x^3
$H_6(x)$	x^6
$H_7(x)$	x^7
$H_8(x)$	x^8+2x^6
$H_9(x)$	x^9
$H_{10}(x)$	x^{10}

where $l,k,m \in \mathbb{F}_3$ *is defined as*

$$l = \begin{cases} 0 & \textit{if } i \textit{ or } j \equiv 0 \textit{ mod } 3 \\ 1 & \textit{otherwise.} \end{cases}$$

$$k = \begin{cases} 1 & \textit{if } i \equiv j \textit{ mod } 3 \\ 2 & \textit{otherwise.} \end{cases}$$

$$m = \begin{cases} 1 & \textit{if } i,j \equiv 2 \textit{ mod } 3 \\ 0 & \textit{otherwise.} \end{cases}$$

Proof. As noted in Remark 4.1.2, $\forall k \in \mathbb{N}$, $\beta_{3k} = x^{3k}$, $\beta_{3k+1} = x^{3k+1}$ and $\beta_{3k+2} = x^{3k+2} + 2x^{3k+2}$. Now, we compute all the cases with respect to the residues of i and j (mod 3), respectively. It is obvious that by multiplying any β_i and β_{3k} we get β_{i+3k}. If we compute the other cases, i.e., $\beta_i \cdot \beta_j$, where $i,j \equiv 1$ or 2 (mod 3), we get the following equations:

$$\begin{aligned} \beta_{3k+1} \cdot \beta_{3k+1} &= \beta_{6k+2} + \beta_{6k} \\ \beta_{3k+1} \cdot \beta_{3k+2} &= \beta_{6k+3} + 2\beta_{6k+1} \\ \beta_{3k+2} \cdot \beta_{3k+2} &= \beta_{6k+4} + \beta_{6k+2} + 2\beta_{6k} \end{aligned}$$

If we combine all these cases with respect to the residues of i and j (mod 3) respectively, then we get the Equation 4.1. As a result, the multiplication of Hermite polynomials, β_i and β_j changes with respect to the indices i,j (mod 3). One can also use the induction method to prove the theorem. ■

Our aim is to represent the elements of finite field $\mathbb{F}_{3^n}$ by using Hermite polynomials in $\mathbb{F}_3[x]$. As in Section 3.1, we take the elements in the standart polynomial representation and convert them into the Hermite polynomial representation. We use the Algorithm 3 in Appendix A.

Remark 4.1.4. *The Hermite polynomial representation is done by using* $\{\beta_0, \beta_1, ..., \beta_{n-1}\}$. *It is obvious that this set is linearly independent. By using Algorithm* 3*, we can convert the elements into this representation and we can write each element of* $\mathbb{F}_{3^n}$ *uniquely as a linear combination of* $\{\beta_0, \beta_1, .., \beta_{n-1}\}$. *Let* $f = f_n\beta_n + .. + f_0\beta_0$ *be an irreducible polynomial of degree* n*, where each* $f_i \in \mathbb{F}_3$. *The set* $\{\beta_0, \beta_1, .., \beta_{n-1}\}$ *constitutes a basis of* $\mathbb{F}_{3^n} \cong \mathbb{F}_3[x]/f(x)$.

4.2 Multiplication of Polynomials in Hermite Representation

In this section, we describe the multiplication of field elements represented by Hermite polynomials and explore the complexity of the multiplication. As in Section 3.2 we give the multiplication and the reduction operations respectively. Theorem 4.2.1 gives the required number of multiplications and additions to multiply polynomials in Hermite basis where $M(n)$ and $A(n)$ denote the minimum number of multiplications and the minimum number of additions for the corresponding algorithm for multiplication of two n-term polynomials.

Theorem 4.2.1. *Let* $a = a_{n-1}\beta_{n-1} + ... + a_0\beta_0$ *and* $b = b_{n-1}\beta_{n-1} + ... + b_0\beta_0$ *be* n*-term polynomials over* $\mathbb{F}_3$ *and* $a \cdot b = c_{2n-2}\beta_{2n-2} + ... + c_0\beta_0$. *By using any multiplication method, the coefficients of the polynomial* c *are computed with*

$$\begin{aligned} M(n) \quad &+ \quad M(n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + 3 \cdot M(\ \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor\ ) \\ &+ \quad 4 \cdot \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor \cdot (n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) \end{aligned}$$

multiplications and

$$\begin{aligned} A(n) \quad &+ \quad A(n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + 2 \cdot A(\lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) \\ &+ \quad 2 \cdot \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor \cdot (n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + 3 \end{aligned}$$

additions.

Proof. By using Theorem 4.1.3, the coefficients are computed as follows,

$$\begin{aligned} c_0 &= a_0b_0 + a_1b_1 + 2a_2b_2 \\ c_1 &= a_0b_1 + a_1b_0 + 2a_2b_1 + 2a_1b_2 \\ c_2 &= a_0b_2 + a_2b_0 + a_1b_1 + a_2b_2 \\ &\vdots \\ c_{2n-3} &= a_{n-2}b_{n-1} + a_{n-1}b_{n-2} \\ c_{2n-2} &= a_{n-1}b_{n-1} \end{aligned}$$

If we compare this multiplication with standard polynomial basis representation, we can see extra terms. All these terms come from the multiplication of the basis elements $\beta_i \cdot \beta_j$, where

$0 \leq i, j \leq n-1$ and $i, j \not\equiv 0 \pmod 3$. The multiplication differs with respect to the values of $i \pmod 3$ and $j \pmod 3$, ie. if $i, j \equiv 1 \pmod 3$ then $\beta_i \cdot \beta_j = \beta_{i+j} + \beta_{i+j-2}$ or if $i, j \equiv 2 \pmod 3$ then $\beta_i \cdot \beta_j = \beta_{i+j} + \beta_{i+j-2} + 2 \cdot \beta_{i+j-4}$. Therefore, the number of the extra terms are related to the number of indices which are smaller than n and equal to one or two (mod 3). The number of indices that are equal to two (mod 3) is $\lfloor \frac{n-\lceil \frac{n}{3} \rceil}{2} \rfloor$ and the number of indices that are equal to one (mod 3) is $n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n-\lceil \frac{n}{3} \rceil}{2} \rfloor$. The extra terms are computed with the multiplication of the polynomials contains those numbers of terms, so the total multiplication complexity is determined as the sum of these multiplications.

Similarly, in the total addition complexity, we have additions to combine these extra terms to the ordinary multiplication terms. These are related to the indices of c_i, where $0 \leq i \leq 2n-4$ and the remainder of these indices from the division with three. ■

Remark 4.2.2. *Some of this multiplication complexity comprises scalar multiplication, ie. multiplication by two. The number of scalar multiplication is:*

$$2 \cdot \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor \cdot (n - \lceil \frac{n}{3} \rceil - \lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor) + M(\lfloor \frac{n - \lceil \frac{n}{3} \rceil}{2} \rfloor)$$

In hardware implementations of finite fields of characteristic three, multiplication by two is equivalent to the negation operation, so we can perform a subtraction operation in place of multiplication by two [35].

By the choice of the multiplication method, some or all elements of extra terms may be computed in the first part of the algorithm, i.e., in n-term polynomial product, so the complexities added to $M(n)$ and $A(n)$ may be smaller than the ones given in Theorem 4.2.1. We explain the theorem with an example by using the Karatsuba multiplication method as in Section 3.2. In Example 4.2.3, we multiply two 4-term polynomials in standart polynomial representation and in Hermite polynomial representation by using Karatsuba multiplication method and we give the complexities in each representation.

Example 4.2.3. *Let $a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$, $b(x) = b_3x^3 + b_2x^2 + b_1x + b_0$ be 4-term polynomials in standart polynomial representation over $\mathbb{F}_3$ and let $a = a_3\beta_3 + a_2\beta_2 + a_1\beta_1 + a_0\beta_0$, $b = b_3\beta_3 + b_2\beta_2 + b_1\beta_1 + b_0\beta_0$ be 4-term polynomials in Hermite polynomial representation over $\mathbb{F}_3$. We get $a(x) \cdot b(x) = c(x) = c_6x^6 + c_5x^5 + ... + c_0$ and $a \cdot b = c = c_6\beta_6 + c_5\beta_5 + ... + c_0\beta_0$ respectively from the multiplications of given polynomials in each representation where the coefficients can be written as:*

Coefficients in Standart polynomial representation			*Coefficients in Hermite polynomial representation*		
c_0	$=$	a_0b_0	c_0	$=$	$a_0b_0 + \underline{a_1b_1} + \underline{2a_2b_2}$
c_1	$=$	$a_0b_1 + a_1b_0$	c_1	$=$	$a_0b_1 + a_1b_0 + \underline{2a_2b_1} + \underline{2a_1b_2}$
c_2	$=$	$a_0b_2 + a_2b_0 + a_1b_1$	c_2	$=$	$a_0b_2 + a_2b_0 + a_1b_1 + \underline{a_2b_2}$
c_3	$=$	$a_0b_3 + a_3b_0 + a_1b_2 + a_2b_1$	c_3	$=$	$a_0b_3 + a_3b_0 + a_1b_2 + a_2b_1$
c_4	$=$	$a_1b_3 + a_3b_1 + a_2b_2$	c_4	$=$	$a_1b_3 + a_3b_1 + a_2b_2$
c_5	$=$	$a_2b_3 + a_3b_2$	c_5	$=$	$a_2b_3 + a_3b_2$
c_6	$=$	a_3b_3	c_6	$=$	a_3b_3

We first examine the complexity of multiplication in standart polynomial representation. In Example 3.2.3 we see that Karatsuba algorithm needs 9 *multiplications and* 24 *additions to find the coefficients of* $c(x)$.

It is obvious that the coefficients in Hermite polyomial representation are computed by using more terms. We underline the extra terms: $a_1b_1 + 2a_2b_2$, $2a_2b_1 + 2a_1b_2$ *and* a_2b_2. *The terms which are not underlined can be similarly obtained by using Karatsuba algorithm with a cost of* 9 *multiplications and* 24 *additions. Now we compute the extra cost.*

We already have a_1b_1 *and* a_2b_2. *We compute* $2a_2b_2$ *by using one multiplication. To compute* $2a_2b_1 + 2a_1b_2$, *we do*

$$2 \cdot [(a_1 + a_2) \cdot (b_1 + b_2) - a_1b_1 - a_2b_2]$$

The cost is 2 *multiplications and* 4 *additions. Also we need* 4 *additions to add these terms to the general result. So we totally need* 3 *multiplications and* 8 *additions as an extra cost. If we add this to the previous result, we can say that by using Karatsuba algorithm we need* $9 + 3 = 12$ *multiplications and* $24 + 8 = 32$ *additions to multiply* $c = a \cdot b$.

By this way, we compare the complexities of the first parts of the polynomial multiplications in the Hermite polynomial representation and the standart polynomial representation.

4.2.1 Irreducible Hermite Binomials

The Hermite polynomials in $\mathbb{F}_3$ are given in Table 4.1 for $n \leq 10$. Because of the recursive structure of the Hermite polynomials, we can say that the polynomials including constant terms are only β_0 and β_2. We try to find irreducible Hermite binomials as low weight polynomials for the performance of the reduction operation. Since irreducible polynomials should include the constant term, there are two forms of the Hermite binomials that are $\beta_n + \beta_0$ and $\beta_n + \beta_2$. We give some irreducible Hermite binomials in Table $C.1$ in Appendix C.

4.2.1.1 Reduction

We perform the reduction operations with respect to each form of the binomials and give the results in Appendix C. From Table $C.2$ in Appendix C, we can see $\beta_n + \beta_2$, where $n \equiv 0 \pmod 3$ has the least number of constant multiplications and the least number of additions. Therefore, we compute the total reduction complexity due to the binomial $\beta_n + \beta_2$, where $n \equiv 0 \pmod 3$. In the rest of the study, we use this binomial so we call it as $f = \beta_n + \beta_2$, where $n \equiv 0 \pmod 3$ and we call the corresponding polynomial of this binomial in the standart polynomial representation as $f(x) = x^n + x^2 + 2$, where $n \equiv 0 \pmod 3$.

Theorem 4.2.4. *Let* $f = \beta_n + \beta_2$, *where* $n \equiv 0 \pmod 3$ *and* $a \cdot b = c'_{2n-2}\beta_{2n-2} + c'_{2n-3}\beta_{2n-3} + ... + c'_0\beta_0$, *where* $c'_i \in \mathbb{F}_3$ *be the product of* a *and* b *in* $\mathbb{F}_{3^n}$. *Then the reduction of* $c = c' \pmod f$ *requires* $2n - 3$ *additions and* $\frac{4}{3}n - 3$ *scalar multiplications.*

Proof. The reduction operation is performed due to the terms of c' with indices $n \leq i \leq 2n-2$. We compute the reduced forms of β_i's for $n \leq i \leq 2n - 2$ by using the reduction formula that

we give in Appendix C.

$$\begin{aligned}
\beta_n &= 2\beta_2 \\
\beta_{n+1} &= 2\beta_3 + \beta_1 \\
\beta_{n+2} &= 2\beta_4 + 2\beta_2 + \beta_0 \\
\beta_{n+3} &= 2\beta_5 \\
\beta_{n+4} &= 2\beta_6 + \beta_4 \\
\beta_{n+5} &= 2\beta_7 + 2\beta_5 + \beta_3 \\
&\vdots \\
\beta_{2n-3} &= 2\beta_{n-1} \\
\beta_{2n-2} &= \beta_{n-2} + \beta_2
\end{aligned}$$

The coefficients c_i' of each β_i on the left of the equations are added to the coefficients c_j' of each β_j's on the right, where $0 \leq j \leq n-1$. So the number of additions is equal to the total number of β_i's on the right side of the equations. Starting from the first equation, for each three equations there are 6 β_j's on the right side except the last two equations. The number of these three iterative equations is $\left(\frac{n-3}{3}\right)$, then the number of terms on the right hand side is $6\left(\frac{n-3}{3}\right)$. Also there are 3 more terms in the last two equations. So the total number is $6\left(\frac{n-3}{3}\right)+3$ and it is equal to $2n-3$.

In each three iterative equations there are 4 scalar multiplications, so from all of three iterative equations, we have $4\left(\frac{n-3}{3}\right)$ scalar multiplications. Also we have one multiplication in the last two equation. Totally, we have $\frac{4}{3}n-3$ scalar multiplications. ■

Remark 4.2.5. *The standart polynomial representation of $\beta_n+\beta_2$ is the trinomial x^n+x^2+2. The reduction by using this trinomial is performed as*

$$\begin{aligned}
x^n &= 2x^2 + 1 \\
x^{n+1} &= 2x^3 + x \\
x^{n+2} &= 2x^4 + x^2 \\
x^{n+3} &= 2x^5 + x^3 \\
x^{n+4} &= 2x^6 + x^4 \\
x^{n+5} &= 2x^7 + x^5 \\
&\vdots \\
x^{2n-2} &= x^{n-2} + x^2 + 2
\end{aligned}$$

After the reduction operation, the coefficients of c in $\mathbb{F}_3$ are given in both Hermite polynomial representation and standart polynomial representation as

Coefficients in Hermite polynomial representation

$$
\begin{aligned}
c_0 &= c'_0 + c'_{n+2} \\
c_1 &= c'_1 + c'_{n+1} \\
c_2 &= c'_2 + 2c'_n + 2c'_{n+2} + c'_{2n-2} \\
c_3 &= c'_3 + 2c'_{n+1} + c'_{n+5} \\
c_4 &= c'_4 + 2c'_{n+2} + c'_{n+4} \\
c_5 &= c'_5 + 2c'_{n+3} + 2c'_{n+5} \\
c_6 &= c'_6 + 2c'_{n+4} + c'_{n+8} \\
&\vdots \\
c_{n-1} &= c'_{n-1} + 2c'_{2n-3}
\end{aligned}
$$

Coefficients in Standart polynomial representation

$$
\begin{aligned}
c_0 &= c'_0 + c'_n + 2c'_{2n-2} \\
c_1 &= c'_1 + c'_{n+1} \\
c_2 &= c'_2 + 2c'_n + c'_{n+2} + c'_{2n-2} \\
c_3 &= c'_3 + 2c'_{n+1} + c'_{n+3} \\
c_4 &= c'_4 + 2c'_{n+2} + c'_{n+4} \\
c_5 &= c'_5 + 2c'_{n+3} + c'_{n+5} \\
c_6 &= c'_6 + 2c'_{n+4} + c'_{n+6} \\
&\vdots \\
c_{n-1} &= c'_{n-1} + 2c'_{2n-3}
\end{aligned}
$$

Now, we compute the reduction complexity in standart polynomial representation by using $x^n + x^2 + 2$ as the irreducible polynomial. We use the reduction complexity results in standart polynomial representation in Theorem 3.2.6. However, in our case because of the chosen trinomial $x^n + x^2 + 2$, the number of additions is slightly different than the one given in Theorem 3.2.6 which is obviously seen in Remark 4.2.5.

Remark 4.2.6. *By using the trinomial $x^n + ax^2 + b$ in $\mathbb{F}_3[x]$, the polynomial modular reduction can be done with $2(n-1)+1 = 2n-1$ additions and at most $2(n-1)$ multiplications. From Remark 4.2.5, we can see that there are $2(n-1)+1$ terms on the right sides of the equations and in each equation there is one scalar multiplication, so there are $n-1$ scalar multiplications.*

Recall that, in the Hermite polynomial representation the reduction with respect to the binomial $\beta_n + \beta_2$ requires $2n-3$ additions and $\frac{4}{3}n - 3$ multiplications. In Table 4.2, we give the comparison of the reduction complexities in Hermite polynomial representation and standart polynomial representation.

Table 4.2: Reduction Complexity

Form	# Additions	# Constant Multiplications
Hermite Binomial, $\beta_n + \beta_2$ ($n \equiv 0 \bmod 3$)	$2n-3$	$\frac{4}{3}n - 3$
Polynomial Basis, $x^n + ax^2 + b$	$2n-1$	$2n-1$

4.3 Matrix vector product for Hermite Basis

In this section, we express the product of two elements in $\mathbb{F}_{3^n}$ as a matrix vector product. Let $u(x) = u_{n-1}x^{n-1} + ... + u_1x + u_0$ be a polynomial representing an element in $\mathbb{F}_{3^n}$. The coefficient vector of $u(x)$ is given by $u = [u_0, u_1, ..., u_{n-1}]^T$. Let a, b and c are such vectors including the coefficients of $a(x), b(x)$ and $c(x)$ in $\mathbb{F}_{3^n}$ and let $f(x)$ be an irreducible polynomial with degree n. We now compute $a(x) \cdot b(x) = c(x) \pmod{f(x)}$ by using the matrix

vector product method. First we construct a $2n-1 \times n$ matrix including the coefficients of $a(x)$ and denote it by M'.

$$\mathbf{M'} = \left[\begin{array}{cccccc} a_0 & 0 & 0 & \dots & 0 & 0 \\ a_1 & a_0 & 0 & \dots & 0 & 0 \\ \vdots & \vdots & & & \vdots & \vdots \\ a_{n-2} & a_{n-3} & a_{n-4} & \dots & a_0 & 0 \\ a_{n-1} & a_{n-2} & a_{n-3} & \dots & a_1 & a_0 \\ \hline 0 & a_{n-1} & a_{n-2} & \dots & a_2 & a_1 \\ 0 & 0 & a_{n-1} & \dots & a_3 & a_2 \\ \vdots & \vdots & & & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a_{n-1} & a_{n-2} \\ 0 & 0 & 0 & \dots & 0 & a_{n-1} \end{array}\right]$$

Let us call the upper part of the matrix M' as L which is a $n \times n$ matrix and the lower part of the matrix M' as U which is a $n-1 \times n$ matrix, i.e.,

$$\mathbf{M'} = \left[\frac{L}{U}\right]$$

The reduction operation is performed by using an $n \times n-1$ reduction matrix which is defined in terms of the irreducible polynomial $f(x)$ and called as Q. After the reduction, one can get an $n \times n$ matrix denoted by M. This matrix is called the Mastrovito matrix in binary fields. It is computed as follows,

$$M = L + Q \cdot U$$

Thus, the multiplication $a(x) \cdot b(x) = c(x) \bmod f(x)$ can be done by the product of matrix M and vector b, i.e.,

$$c = M \cdot b$$

Remark 4.3.1. *In the reduction part, the rows $n, (n+1), .., (2n-1)$ of the matrix M' are added to the first n rows of M' with respect to the reduction modulo $f(x)$. The number of nonzero entries in the reduction matrix Q is equal to the number of additions and the number of nonzero entries different than one is equal to the number of scalar multiplications in reduction complexity.*

In [15], they survey the matrix vector product techniques for binary fields. There are two different approaches in the application of the matrix vector product. One is that the polynomial multiplication part is done by any multiplication method and then the reduction part is performed by the matrix vector product. The other approach is that the two steps of multiplication are performed by the matrix vector product using the Mastrovito matrix. Recall that we give the complexity results of the Mastrovito multiplication in Section 2.3. In Table 4.2, we show that in reduction part of the polynomial multiplication, in some cases Hermite polynomial representation is better than the standart polynomial representation in the case of the number of additions. Therefore in this section, we use matrix vector product method only in the reduction

part and we explain how the reduction operation in Hermite polynomial representation can be computed by using matrix vector operations.

The reduction matrix is constructed by using the irreducible reduction polynomial $f(x)$. Let the jth column of reduction matrix Q be denoted by $q_j = [q_{0,j}, q_{1,j}, ..., q_{n-1,j}]^T$, where the entries of this column vector correspond to the coefficients of $q_j(x) = q_{0,j} + q_{1,j}x + ... + q_{n-1,j}x^{n-1}$. In [15], this $q_j(x)$ is defined by

$$q_j(x) = \begin{cases} x^n \bmod f(x), & j = 0 \\ xq_{j-1}(x) \bmod f(x), & j = 1, ..., n-2, \end{cases} \tag{4.2}$$

In the Hermite polynomial representation, let the entries of the column vector q_j correspond to the coefficients of $q_j = q_{0,j}\beta_0 + q_{1,j}\beta_1 + ... + q_{n-1,j}\beta_{n-1}$ and we define q_j by

$$q_j = \begin{cases} \beta_n \bmod f, & j = 0 \\ \beta_1 q_{j-1} + l \cdot q_{j-2} \bmod f, & j = 1, ..., n-2, \end{cases} \tag{4.3}$$

where

$$l = \begin{cases} 0 & \text{if } j \equiv 1 \bmod 3 \\ 1 & \text{if } j \equiv 2 \bmod 3 \\ 2 & \text{if } j \equiv 0 \bmod 3 \end{cases}$$

Now, we give an example to show the difference between the reduction matrices in standart polynomial representation and Hermite polynomial representation.

Example 4.3.2. *We choose an irreducible Hermite binomial $\beta_n + \beta_2$, where $n \equiv 0$ (mod 3). Let's take $\beta_{12} + \beta_2$ from Table C.1. It is equal to $x^{12} + x^2 + 2$ in standart polynomial representation. We compute the reduction matrices by using Equation 4.2 and Equation 4.3. The sizes of the matrices are 12×11. We denote the reduction matrix of $x^{12} + x^2 + 2$ by Q_S and the reduction matrix of $\beta_n + \beta_2$ by Q_H.*

$$\mathbf{Q_S} = \left[\begin{array}{ccc|ccc|ccc|cc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \end{array}\right]$$

$$\mathbf{Q_H} = \left[\begin{array}{ccc|ccc|ccc|cc} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 2 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \end{array}\right]$$

We count the nonzero entries of each matrices. The matrix Q_S has 23 nonzero entries which is also equal to the number of additions in reduction complexity and the number of 2's in the matrix is 11 which is equal to the number of scalar multiplications. The matrix Q_H has 21 nonzero entries and the number of 2's in the matrix is 13.

By using Theorem 4.2.4, the reduction complexity of $\beta_{12}+\beta_2$ contains $2n-3 = 2.12-3 = 21$ additions and $\frac{4}{3}n - 3 = \frac{4}{3}12 - 3 = 13$ scalar multiplications. In the standart polynomial representation, as we stated in Remark 4.2.6, we can compute the reduction complexity of $x^{12} + x^2 + 2$ as $(3-1)(12-1)+1 = 23$ additions and $12 - 1 = 11$ scalar multiplications.

In Example 4.3.2, we divide the matrices into 3×3 block matrices to simplify the observation. As n increases the entries of the matrices do not change, only they expand diagonally. Both matrices have a recursive structure. The 3×3 block matrices repeat down the diagonals of the matrix. Therefore, we can compute the general reduction complexity for any n, where $n \equiv 0 \pmod 3$ by using the matrices in Example 4.3.2.

In each reduction matrices, there are two different 3×3 nonzero block matrices and they appear in each three columns. The number of 3×3 block matrices in each reduction matrices is $\frac{(n-3)}{3}$. First, we compute the reduction complexity for Q_S. These two block matrices in Q_S contains 6 nonzero terms and 3 scalar multiplications and in the last two columns there are 5 nonzero terms and 2 scalar multiplications. Totally, Q_S has $6\frac{(n-3)}{3} + 5 = 2n - 1$ nonzero terms and $3\frac{(n-3)}{3} + 2 = n - 1$ scalar multiplications. For Q_H, these two block matrices contains 6 nonzero terms and 4 scalar multiplications and in the last two columns there are 3 nonzero terms and 1 scalar multiplications. Totally, Q_S has $6\frac{(n-3)}{3} + 3 = 2n - 3$ nonzero terms and $4\frac{(n-3)}{3} + 1 = \frac{4}{3}n - 3$ scalar multiplications. These results are the same with the reduction complexities given in Table 4.2. By this way, we give the matrix vector product method for the reduction part of the polynomial multiplication in Hermite polynomial representation of $\mathbb{F}_{3^n}$.

CHAPTER 5

NORMAL BASIS REPRESENTATION

In this chapter, we give some structural properties of normal bases and we examine the relationship between normal bases and orthogonal polynomials.

We have briefly given the definition of normal bases in Section 2.2. Let $g(x)$ be an irreducible polynomial of degree n. Recall that, if α is a root of $g(x)$ then the n distinct roots of $g(x)$ in $\mathbb{F}_{q^n}$ is given by $B = \left\{\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}\right\}$. If the elements of B are linearly independent, then B is called a *normal basis* for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ and α is named as a *normal element* of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. Also, $g(x)$ is called a *normal polynomial*. The elements in a normal basis are exactly the roots of a normal polynomial. Normal polynomials exist for every degree n [32].

Let $\alpha = \alpha_0$ generate $B = \{\alpha_0, \alpha_1, \alpha_2, ..., \alpha_n\}$, a normal basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ and let $\alpha_i = \alpha^{q^i}$ for $0 \leq i \leq n-1$. Then $\alpha_i\alpha_j$ is a linear combination of $\alpha_0, \alpha_1, \alpha_2, ..., \alpha_n$ with coefficients in F_q for any $0 \leq i, j \leq n-1$, ie. $\alpha_i\alpha_j = \sum_{k=0}^{n-1} t_{ij}^{(k)}\alpha_k$ where $t_{ij}^{(k)} \in \mathbb{F}_q$. For an element $a \in \mathbb{F}_{q^n}$, let $a_i \in \mathbb{F}_q$ and $a = \sum_{i=0}^{n-1} a_i\alpha_i$ and let $A = (a_0, ..., a_{n-1})$. The multiplication of two elements $a, b \in \mathbb{F}_{q^n}$ is given by $c = ab$ where $c_k = \sum_{i,j} a_i b_j t_{ij}^k = AT_kB'$ where the collection of matrices $\left\{T_k = (t_{ij}^{(k)})\right\}$ is known as a multiplication table for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. $(t_{ij}^{(k)})$ denotes the left cyclic shift of the vector $t^{(0)}$ by k positions, ie. $t_{ij}^{(k)} = t_{i-k,j-k}^{(0)}$. Let $(t_{ij}^{(0)})$ be the matrix denoted by T_0. Then, T_0 and T_k have the same nonzero entries. The number of non-zero entries in T is called as the *complexity of the normal basis* B, denoted by c_B [33]. The following theorem gives the lower bound of c_B.

Theorem 5.0.3. *For any normal basis B of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$, $c_B \geq 2n-1$.*

5.1 Optimal Normal Bases

A normal basis B is called *optimal* if $c_B = 2n-1$. Optimal normal bases are normal bases with low complexity. For the efficiency of hardware and software implementations of finite fields the normal bases are required to have low complexities. Therefore, existence of optimal normal bases in finite fields is an important issue. For finding normal bases of a required complexity, we don't have many techniques. However, in [33] two general constructions that give all the optimal normal bases and a large family of normal bases of low complexity are given. We begin with these two constructions presented in [33].

Theorem 5.1.1. *Suppose $n+1$ is a prime and q is primitive in $\mathbb{Z}_{n+1}$, where q is a prime or prime power. Then the n nonunit $n+1$th roots of unity are linearly independent and they form an optimal normal basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.*

Theorem 5.1.2. *Let $2n+1$ be a prime and assume that either*

- **(i)** *2 is primitive in $\mathbb{Z}_{n+1}$, or*
- **(ii)** *$2n+1 \equiv 3$ (mod 4) and 2 generates the quadratic residues in $\mathbb{Z}_{2n+1}$. Then $\alpha = \gamma + \gamma^{-1}$ generates an optimal normal basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$, where γ is a primitive $(2n+1)$th root of unity.*

The optimal normal basis constructed by using Theorem 5.1.1 is called *type I optimal normal basis*, and the basis constructed by using Theorem 5.1.2 is called *type II optimal normal basis*. In [33], for binary extension fields it is proved that all the optimal normal bases in finite fields are completely determined by these two theorems. If n does not satisfy the criteria in these theorems then $\mathbb{F}_{2^n}$ does not contain an optimal normal basis. Later, in [16] this result is extended for any arbitrary finite field. The following theorem is used to determine that a given basis is an optimal normal basis or not.

Theorem 5.1.3. *[32] Let $B = \left\{\alpha, \alpha^q, \alpha^{q^2}, ..., \alpha^{q^{n-1}}\right\}$ be an optimal normal basis of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. Let $z = Tr_{q^n|q}(\alpha)$, the trace of α in $\mathbb{F}_q$. Then either*

- **(i)** *$n+1$ is a prime, q is primitive in $\mathbb{Z}_{n+1}$ and $-\alpha/z$ is a primitive $(n+1)$th root of unity or*
- **(ii)**
  - **(a)** *$q = 2^m$ for some integer m such that $gcd(m, n) = 1$,*
  - **(b)** *$2n+1$ is a prime, 2 and -1 generate the multiplicative group $\mathbb{Z}^*_{2n+1}$ and*
  - **(c)** *$\alpha/z = \zeta + \zeta^{-1}$ for some primitive $(2n+1)$th root ζ of unity.*

5.2 Dickson Polynomials

Dickson polynomials are first introduced by L.E.Dickson in 1896, see Lidl et al. [28]. They are an important class of permutation polynomials. A polynomial $p(x)$ over $\mathbb{F}_q$ is a permutation polynomial if $p(x)$ induces an injective mapping on the field $\mathbb{F}_q$. Dickson polynomials with this important permutation property have several applications in combinatorics and cryptography [23], [24]. Over the complex numbers, Dickson polynomials are essentially equivalent to orthogonal Chebyshev polynomials with a change of variable. Dickson polynomials are mainly studied over finite fields, when they are not equivalent to Chebyshev polynomials.

There are two kinds of Dickson polynomials. In this study, we consider the first kind of Dickson polynomials.

Definition 5.2.1. *Let $a \in \mathbb{F}_q$ and $n \geq 2$ be an integer. The Dickson polynomial of the first kind with degree n and parameter a is defined as*

$$D_n(x,a) = \sum_{i=0}^{\lfloor n/2 \rfloor} \frac{n}{n-i} \binom{n-i}{i} (-a)^i x^{n-2i}$$

For $n = 0$ we define $D_0(x, a) = 2$ and similarly we define $D_1(x, a) = x$ for $n = 1$.

Remark 5.2.2. *Dickson polynomials satisfy a second order recurrence, i.e. for $n \geq 0$,*

$$D_{n+2}(x, a) = xD_{n+1}(x, a) - aD_n(x, a).$$

We are interested in the first kind of Dickson polynomials where the parameter $a = 1$ and we denote them shortly, by $D_n(x)$. Then in this case, we can express the recurrence as $D_{n+2}(x) = xD_{n+1}(x) - D_n(x)$. In [23], they use $D_n(x)$ to represent binary extension fields for efficient field multiplication. Using the relation the Dickson polynomials $D_n(x)$ in $\mathbb{F}_2[x]$ are obtained for $n \leq 10$ and given in Table 5.1.

Table 5.1: Dickson polynomials in $\mathbb{F}_2[x]$

$D_0(x)$	0
$D_1(x)$	x
$D_2(x)$	x^2
$D_3(x)$	$x^3 + x$
$D_4(x)$	x^4
$D_5(x)$	$x^5 + x^3 + x$
$D_6(x)$	$x^6 + x^2$
$D_7(x)$	$x^7 + x^5 + x$
$D_8(x)$	x^8
$D_9(x)$	$x^9 + x^7 + x^5 + x$
$D_{10}(x)$	$x^{10} + x^6 + x^2$

Dickson polynomials are also used for the construction of optimal normal bases. We give the conditions for optimal normal bases in Theorem 5.1.2. In [28], they give the connection between this theorem and Dickson polynomials. They show that the minimal polynomial of such an α over $\mathbb{F}_q$ is related to the Dickson polynomials D_{n+1} and D_n.

Theorem 5.2.3. *[28] If $2n + 1$ is a prime and α as in Theorem 5.1.2 (ii), generates an optimal normal basis then the minimal polynomial of α over $\mathbb{F}_q$ is given by*

$$(D_{n+1}(x) - D_n(x))/(x - 2).$$

Proof. By assumption let $2n+1$ be prime and $\alpha = \zeta + \zeta^{-1}$ generates an optimal normal basis, i.e. $B = \left\{\alpha, \alpha^2, \alpha^{2^2}, ..., \alpha^{2^{n-1}}\right\}$.

Since $2n + 1 \equiv 3 \pmod 4$ and 2 generates the quadratic residues in $\mathbb{Z}_{2n+1}$, then the basis can be written as $B = \left\{\alpha, \alpha^2, ..., \alpha^n\right\} = \left\{\zeta + \zeta^{-1}, \zeta^2 + \zeta^{-2}, ..., \zeta^n + \zeta^{-n}\right\}$.

Since it is given that ζ is a primitive $(2n + 1)$th root of unity, i.e. $\zeta^{2n+1} = 1 \pmod{q^n}$. Then $\zeta^{n+1} = \zeta^{-n} \pmod{q^n}$. For any $0 \leq j \leq n$, ζ^j is also a $(2n + 1)$th root of unity. This gives us

the following equation:

$$(\zeta^j)^n + (\zeta^j)^{-n} = (\zeta^j)^{n+1} + (\zeta^j)^{-(n+1)}. \tag{5.1}$$

Let $m_n(x) \in \mathbb{F}_q[x]$ be the minimal polynomial of $\alpha = \zeta + \zeta^{-1}$, then all the roots of $m_n(x)$ are the elements of the basis $B = \left\{\zeta + \zeta^{-1}, \zeta^2 + \zeta^{-2}, ..., \zeta^n + \zeta^{-n}\right\}$. We can express the polynomial as

$$m_n(x) = \prod_{j=1}^{n}(x - \zeta^j - \zeta^{-j}).$$

By Waring's formula, for any positive integer t,

$$(\zeta^j)^t + (\zeta^j)^{-t} = \sum_{i=0}^{[t/2]} \frac{t}{t-i}\binom{t-i}{i}(-1)^i(\zeta^j - \zeta^{-j})^{t-2i}.$$

If, we write the first kind of Dickson polynomials,

$$D_t(x) = \sum_{i=0}^{[t/2]} \frac{t}{t-i}\binom{t-i}{i}(-1)^i x^{t-2i}$$

it is obvious by 5.1 that $\zeta^j + \zeta^{-j}$ is a root of $D_{n+1}(x) - D_n(x)$ for $j = 0, 1, ..., n$. Then the minimal polynomial $m_n(x)$ divides $D_{n+1}(x) - D_n(x)$. The degree of $D_{n+1}(x) - D_n(x)$ is $n + 1$ and $m_n(x)$ is n. If we divide $D_{n+1}(x) - D_n(x)$ by $m_n(x)$, we get a linear polynomial,

$$D_{n+1}(x) - D_n(x) = m_n(x)(x - 2).$$

■

5.3 Hermite Polynomials in $\mathbb{F}_2[x]$

In Chapter 4 we use the recurrence relation of probabilists Hermite polynomials and we generate them in $\mathbb{F}_3[x]$. In this section, we give more details about these polynomials and our aim is to find a relationship between Hermite polynomials and optimal normal basis elements as in Theorem $5.2.3$ over $\mathbb{F}_2$. Therefore in this section, we deal with Hermite polynomials over $\mathbb{F}_2$.

Hermite polynomials are a classical orthogonal polynomial sequence that arise in probability, combinatorics and physics. The probabilists' Hermite polynomials are defined by

$$H_n(x) = (-1)^n e^{x^2/2}\frac{d^n}{dx^n}e^{-x^2/2}$$

They can be written explicitly as:

$$H_n(x) = \frac{n!}{(\sqrt{2})^n} \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} \frac{(-1)^i}{i!(n-2i)!} (\sqrt{2}x)^{n-2i}$$

As it is given in Section 4.1, Hermite polynomials satisfy the recurrence relation $H_n(x) = x \cdot H_{n-1}(x) - (n-1) \cdot H_{n-2}(x)$. We give the Hermite polynomials in $\mathbb{F}_2[x]$ for $n \leq 10$ in Table 5.2.

Table 5.2: Hermite polynomials in $\mathbb{F}_2[x]$

$H_0(x)$	1
$H_1(x)$	x
$H_2(x)$	x^2+1
$H_3(x)$	x^3+x
$H_4(x)$	x^4+1
$H_5(x)$	x^5+x
$H_6(x)$	$x^6+x^4+x^2+1$
$H_7(x)$	$x^7+x^5+x^3+x$
$H_8(x)$	x^8+1
$H_9(x)$	x^9+x
$H_{10}(x)$	$x^{10}+x^8+x^2+1$

For finding a relationship between optimal normal basis elements and Hermite polynomials, we examine that if we can get a connection between Hermite polynomials and Dickson polynomials over $\mathbb{F}_2$. We use the recurrence relations of polynomials and we compare them over $\mathbb{F}_2$.

$$\begin{aligned} D_n(x) &= xD_{n-1}(x) + D_{n-2}(x), \;\; D_0(x) = 0, \;\; D_1(x) = x \\ H_n(x) &= xH_{n-1}(x) + H_{n-2}(x) + nH_{n-2}(x), \;\; H_0(x) = 1, \;\; H_1(x) = x \end{aligned}$$

Different than Dickson recurrence relation, Hermite relation has a term $nH_{n-2}(x)$ and the first Hermite polynomial, $H_0(x)$ is 1. We define a new polynomial sequence generated by using Hermite polynomials over $\mathbb{F}_2$ to obtain Dickson polynomials from Hermite polynomials.

Definition 5.3.1. *Let the polynomial sequence be denoted by $A_n(x)$ and $A_0(x) = 1$, $A_1(x) = 0$. The recurrence relation of $A_n(x)$ connected to Hermite polynomials for $n \geq 2$ is*

$$A_n(x) = xA_{n-1}(x) + A_{n-2}(x) + nH_{n-2}(x) \tag{5.2}$$

In Table 5.3, we give the polynomials generated by $A_n(x)$ for $n \leq 10$.

Proposition 5.3.2. *If $2n+1$ is a prime and α as in Theorem 5.1.2 (ii), generates an optimal normal basis then the minimal polynomial of α over $\mathbb{F}_2$ is given by*

$$\left((A_{n+1}(x) + H_{n+1}(x)) + (A_n(x) + H_n(x))\right)/x.$$

Table 5.3: Polynomials generated by $A_n(x)$ over $\mathbb{F}_2$

$A_0(x)$	1
$A_1(x)$	0
$A_2(x)$	1
$A_3(x)$	0
$A_4(x)$	1
$A_5(x)$	x^3
$A_6(x)$	x^4+1
$A_7(x)$	x^3
$A_8(x)$	1
$A_9(x)$	x^7+x^5
$A_{10}(x)$	x^8+x^6+1

Proof. If we sum up the recurrence relations of two polynomials $A_n(x)$ and $H_n(x)$, we get

$$\begin{aligned} A_n(x)+H_n(x) &= xA_{n-1}(x)+A_{n-2}(x)+nH_{n-2}(x)+xH_{n-1}(x)+H_{n-2}(x)+nH_{n-2}(x) \\ &= xA_{n-1}(x)+A_{n-2}(x)+xH_{n-1}(x)+H_{n-2}(x) \\ &= x\left(A_{n-1}(x)+H_{n-1}(x)\right)+\left(A_{n-2}(x)+H_{n-2}(x)\right) \end{aligned}$$

and the first two polynomials of this recurrence relation are $A_0(x)+H_0(x)=0$ and $A_1(x)+H_1(x)=x$ which are same with Dickson polynomials. By these results, we can write $D_n(x)=A_n(x)+H_n(x)$. Therefore we can rearrange the result in Theorem 5.2.3 by using the connection between Dickson polynomials and Hermite polynomials. ■

Remark 5.3.3. *As we have given in Definition 5.3.1, the polynomial sequence $A_n(x)$ is generated by using Hermite polynomials over $\mathbb{F}_2$. Firstly by constructing $A_n(x)$ from Hermite polynomials, we can obtain Dickson polynomials from Hermite polynomials over $\mathbb{F}_2$. Therefore we can conclude that the minimal polynomial of a normal element generating an optimal normal basis of $\mathbb{F}_{2^n}$ over $\mathbb{F}_2$ is related to the Hermite polynomials.*

CHAPTER 6

CONCLUSION

In this thesis, Charlier and Hermite polynomial representations over finite fields of characteristic three are presented and in addition to this, optimal normal basis elements in binary fields are investigated.

In Chapter 3, we propose the Charlier polynomial representation for finite fields of characteristic three. Charlier polynomials are used in the representation of binary fields in [5]. We modify the idea given in [5] for $\mathbb{F}_{3^n}$. We give the multiplication method in two steps and compute the multiplication and reduction complexites. In this representation, we show that one can obtain an irreducible binomial $\beta_n + \beta_0$, where $n \equiv 2 \pmod 3$, which allows us an efficient modular reduction. For these Charlier binomials, we have better results in reduction complexity according to the standart polynomial representation. We design a sequential multiplier for the elements of $\mathbb{F}_{3^n}$, where $n \equiv 2 \pmod 3$ and we give the cubing operation for these elements.

In Chapter 4, we modify the Hermite polynomial representation given in [4] for finite fields of characteristic three. We give the multiplication method in Hermite polynomial representation. In this representation, we get an efficient modular reduction if we choose an irreducible binomial $\beta_n + \beta_2$, where $n \equiv 0 \pmod 3$. We also give the matrix vector product method for the multiplication of field elements in this representation and we construct the reduction matrix for the binomial $\beta_n + \beta_2$, where $n \equiv 0 \pmod 3$. Furthermore, these proposed methods presented in Chapter 3 and Chapter 4 bring a new approach to the representation of finite fields of characteristic three.

In Chapter 5, we focus on the relation between the optimal normal bases elements in binary fields and permutation polynomials. In [33] the constructions that give all the optimal normal bases are presented. Dickson polynomials are also used for the construction of optimal normal bases. In [28], the connection between optimal normal bases and Dickson polynomials are given. By using this connection and constructions of optimal normal bases, we find a relationship between optimal normal basis elements and Hermite polynomials over $\mathbb{F}_2$.

REFERENCES

[1] G. Agnew, T. Beth, R. Mullin and S. Vanstone, *Arithmetic operations in $GF(2^m)$*, Journal of Cryptology, 6:3-13, 1993.

[2] O. Ahmadi, F. Rodriguez-Henriquez, *Low Complexity Cubing and Cube Root Computation over $\mathbb{F}_{3^m}$ in Polynomial Basis*. IEEE Transactions on Computers, 59:1297-1308.

[3] O. Ahmadi, D. Hankerson and A. Menezes, *Software Implementation of Arithmetic in $\mathbb{F}_{3^m}$*, International Workshop on Arithmetic of Finite Fields (WAIFI 2007), LNCS 4547, 85-102, 2007.

[4] S. Akleylek, *On the Representation of Finite Fields*, Phd. Thesis, 2010.

[5] S. Akleylek, M. Cenk, F. Özbudak, *Polynomial Multiplication over Binary Fields Using Charlier Polynomial Representation with Low Space Complexity*, INDOCRYPT 2010: 227-237.

[6] S. Akleylek, F. Özbudak, C. Özel, *Charlier Polynomial Representation for Finite Fields of Characteristic Three*, 18th International Conference on Applications of Computer Algebra (ACA 2012), Sofia, Bulgaria, 2012.

[7] S. Akleylek, F. Özbudak, C. Özel, *Hermite Polynomial Representation for Finite Fields of Characteristic Three*, 5th International Information Security and Cryptology Conference (ISCTURKEY 2012), vol.5 pp.155-159 Ankara, 2012.

[8] S. Akleylek, F. Özbudak, C. Özel, *On the Multiplication over Finite Fields of Characteristic Three in Hermite Polynomial Representation*, International Conference on Applied and Computational Mathematics (ICACM), Ankara, 2012.

[9] S. Akleylek, F. Özbudak, C. Özel, *On the Arithmetic Operations over Finite Fields of Characteristic Three with Low Complexity*, submitted, 2013.

[10] P.S.L.M. Barreto, H.Y. Kim, B. Lynn, M. Scott, *Efficient Algorithms for Pairing-Based Cryptosystems*, Proc. 22nd Ann. Int'l Cryptology Conf. Advances in Cryptology (CRYPTO'02), M. Yung, ed., pp. 354-368, 2002.

[11] J.-L. Beuchat, M. Shirase, T. Takagi and E. Okamoto, *An algorithm for the η_T pairing calculation in characteristic three and its hardware implementation*, Proceedings of the 18th IEEE Symposium on Computer Arithmetic, pages 97-104, IEEE Computer Society, 2007.

[12] I.F. Blake, G. Seroussi, N.P. Smart, *Elliptic curves in cryptography*, London Mathematical Society Lecture Note Series 265, Cambridge Univ. Press, 1999.

[13] H. Cohen, G. Frey, *Handbook of Elliptic and Hyperelliptic Curve Cryptography*, Discrete Math. Appl., Chapman Hall/CRC, 2006.

[14] D. Drake, *The Combinatorics of Associated Hermite Polynomials*, European Journal of Combinatorics, vol.30 no.4 pp. 1005-1021, 2009.

[15] S. S. Erdem, T. Yanik, and C. K. Koc, *Polynomial Basis Multiplication over* $GF(2^m)$, Acta Applicandae Mathematicae, vol. 93, nos. 1-3, pp. 33-55, 2006.

[16] S. Gao and H. W. Lenstra, *Optimal normal basis*, Designs, Codes and Cryptography, 2:315-323, 1992.

[17] J. von zur Gathen, *Irreducible Trinomials over Finite Fields*, In B. Mourrain, editor, Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation-ISSAC2001, pp. 332-336. ACM Press, 2001.

[18] C.D. Godsil, *Algebraic Combinatorics*, Chapman Hall/CRC Mathematics Series, Boca Raton, 1993.

[19] R. Granger, D. Page and M. Stam, *Hardware and software normal basis arithmetic for pairing-based cryptography in characteristic three*, IEEE Transactions on Computers, pp. 852-860, 54, 2005.

[20] D. Hankerson, J. Lopez Hernandez and A. Menezes, *Software Implementation of Elliptic Curve Cryptography over Binary Fields*, Proc. of CHES 2000, LNCS, Vol. 1965, pp. 1-24, Springer-Verlag, 2000.

[21] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*, Springer-Verlag, 2004.

[22] K. Harrison, D. Page and N. Smart, *Software implementation of finite fields of characteristic three, for use in pairing-based cryptosystems*, LMS Journal of Computation and Mathematics, pp. 181-193, 5, 2002.

[23] M.A. Hasan and C. Negre, *Subquadratic Space Complexity Multiplication over Binary Fields with Dickson Polynomial Representation*, WAIFI 2008, LNCS 5130, pp.88-102, 2008.

[24] M.A. Hasan and C. Negre, *Low Space Complexity Multiplication over Binary Fields with Dickson Polynomial Representation*, IEEE Trans. on Computers, vol.60, no.4, pp.602-607, 2011.

[25] T. Itoh and S. Tsujii, *A Fast Algorithm for Computing Multiplicative Inverse in* $GF(2^m)$ *Using Normal Bases*, Information and Computer, vol. 78, pp. 171-177, 1988.

[26] T. Kerins, W.P. Marnane, E.M. Popovici and P.S.L.M. Barreto, *Efficient Hardware for the Tate Pairing Calculation in Characteristic Three*, CHES 2005, LNCS 3659, pp. 412-426, 2005.

[27] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge University, 1997.

[28] R. Lidl, G.L. Mullen, G. Turnwald, *Dickson Polynomials*, Longman Scientific and Technical, Essex, United Kingdom, 1993.

[29] Maplesoft, a division of Waterloo Maple Inc., Waterloo, Ontario. http://www.maplesoft.com

[30] E.D. Mastrovito, *VLSI Architectures for Multiplication over Finite Field* $GF(2^m)$, Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes, T. Mora, ed., pp. 297-309, Berlin, Springer-Verlag, 1988.

[31] A.J. Menezes, *Handbook of Applied Cryptography*, CRC, 1997.

[32] A.J. Menezes, I.F. Blake, X. Gao, R. C. Mullin, S.A. Vanstone, T. Yaghoobian, *Application of Finite Fields*, Kluwer Academic Publishers, 1993.

[33] R.C. Mullin, I.M. Onyszchuk, S.A. Vanstone and R.M. Wilson, *Optimal normal bases in* $GF(p^n)$, Discrete Appl. Math. 22, pp. 149-161, 1988/89.

[34] R. Mullin, I.Onyszchuk and S. Vanstone, *Computational method and apparatus for multiplication*, U.S patent 4,745,568, May 1988.

[35] D.Page, N.Smart, *Hardware Implementation of Finite Fields of Characteristic Three*, In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002, LNCS, vol. 2523, pp. 529-539, Springer, Heidelberg, 2003.

[36] D.Panairo and D.Thompson, *Efficient pth Root Computations in Finite Fields of Characteristic p*, Designs, Codes and Cryptography, vol. 50, pp. 351-358, 2009.

[37] A. Reyhani-Masoleh and M. A. Hasan, *Low complexity bit parallel architectures for polynomial basis multiplication over* $GF(2^m)$, IEEE Trans. Computers, vol. 53, no. 8, pp. 945-959, 2004.

[38] B.Sunar, Ç.K.Koç, *Mastrovito Multiplier for All Trinomials*, IEEE Trans. Computers, vol. 48, no. 5, pp. 522-527, May 1999.

[39] A.Weimerskirch, C.Paar, *Generalizations of the Karatsuba Algorithm for Efficient Implementations*, http://eprint.iacr.org/2006/224, 2006.

[40] H. Wu, *Bit-parallel Finite Field Multiplier and Squarer Using Polynomial Basis*, IEEE Trans. Computers, vol. 51, no. 7, pp. 750-758, 2002.

Appendix A

CONVERSION FROM STANDART POLYNOMIAL REPRESENTATION TO A MODIFIED POLYNOMIAL REPRESENTATION

Algorithm 3 Conversion of Coefficients From Standart Polynomial Representation to a Modified Polynomial Representation

Input: $a(x) = \sum_{i=0}^{n-1} a'_i x^i$
Output: $(a_0, a_1, ..., a_{n-1})$ where $a = \sum_{i=0}^{n-1} a_i \beta_i$

1: $T \leftarrow a$
2: **for** $i = n$ downto 1 **do**
3: **if** $deg(T) = i$ then
4: **if** $a'_i = 1$ then $a_i \leftarrow 1, T \leftarrow T + 2\beta_i$
5: **else if** $a'_i = 2$ then $a_i \leftarrow 2, T \leftarrow T + \beta_i$
6: **end if**
7: **else** $a_i \leftarrow 0$
8: **end if**
9: **end for**
10: $a_0 \leftarrow T$

Appendix B

IRREDUCIBLE CHARLIER BINOMIALS AND REDUCTION

Table B.1: Irreducible Charlier Binomials

$\beta_3 + \beta_0$	$\beta_{37} + \beta_0$	$\beta_{118} + \beta_0$
$\beta_5 + \beta_0$	$\beta_{46} + \beta_0$	$\beta_{133} + \beta_0$
$\beta_{11} + \beta_0$	$\beta_{47} + \beta_0$	$\beta_{158} + \beta_0$
$\beta_{14} + \beta_0$	$\beta_{74} + \beta_0$	$\beta_{179} + \beta_0$
$\beta_{26} + \beta_0$	$\beta_{77} + \beta_0$	$\beta_{242} + \beta_0$
$\beta_{31} + \beta_0$	$\beta_{83} + \beta_0$	$\beta_{386} + \beta_0$

Let f be the irreducible Charlier binomial f. Reduction operation with respect to modulo f can be performed as follows:

We take each binomial form respectively, let $f = \beta_n + \beta_0$ where $n \equiv 0 \pmod 3$ and let $n \leq i \leq 2n-2$. Then,

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i \\ 2\beta_0 \cdot \beta_{i-n} &= \beta_i \\ \beta_i &= 2\beta_{i-n} \end{aligned} \tag{B.1}$$

Now let $f = \beta_n + \beta_0$ where $n \equiv 1 \pmod 3$. The reduction of β_i where $n \leq i \leq 2n-2$ differs with respect to the value of $i - n \pmod 3$. We give the reduction formulas for the values zero, one and two, respectively.

If $i - n \equiv 0 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i \\ \beta_i &= 2\beta_0 \cdot \beta_{i-n} \\ \beta_i &= 2\beta_{i-n} \end{aligned}$$

If $i - n \equiv 1 \pmod 3$:

$$\begin{aligned}
\beta_n \cdot \beta_{i-n} &= \beta_i + \beta_{i-1} \\
2\beta_0 \cdot \beta_{i-n} &= \beta_i + \beta_{i-1} \\
\beta_i &= 2\beta_{i-n} + 2\beta_{i-1} \\
\beta_i &= 2\beta_{i-n} + 2(2\beta_{i-1-n}) \\
\beta_i &= 2\beta_{i-n} + \beta_{i-1-n}
\end{aligned}$$

If $i - n \equiv 2 \pmod 3$:

$$\begin{aligned}
\beta_n \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-1} \\
2\beta_0 \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-1} \\
\beta_i &= 2\beta_{i-n} + \beta_{i-1} \\
\beta_i &= 2\beta_{i-n} + 2\beta_{i-1-n} + \beta_{i-2-n}
\end{aligned}$$

If we combine these, we get the equation:

$$\beta_i = 2\beta_{i-n} + u \cdot (v \cdot \beta_{i-1-n} + w \cdot \beta_{i-2-n}) \tag{B.2}$$

$$u = \begin{cases} 0 & \text{if } i - n \equiv 0 \bmod 3 \\ 1 & \text{otherwise.} \end{cases}$$

$$v = \left\{ \begin{array}{ll} 1 & \text{if } i - n \equiv 1 \bmod 3 \\ 2 & \text{if } i - n \equiv 2 \bmod 3 \end{array} \right\}, w = \begin{cases} 0 & \text{if } i - n \equiv 1 \bmod 3 \\ 1 & \text{if } i - n \equiv 2 \bmod 3 \end{cases}$$

Now let $f = \beta_n + \beta_0$, where $n \equiv 2 \pmod 3$. Since the value of $n \equiv 2 \pmod 3$, the multiplication changes with respect to the values of $i - n \pmod 3$, where $n \leq i \leq 2n - 2$. We write this seperately.

If $i - n \equiv 0 \pmod 3$:

$$\begin{aligned}
\beta_n \cdot \beta_{i-n} &= \beta_i \\
2\beta_0 \cdot \beta_{i-n} &= \beta_i \\
\beta_i &= 2\beta_{i-n}
\end{aligned}$$

If $i - n \equiv 1 \pmod 3$:

$$\begin{aligned}
\beta_n \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-1} \\
2\beta_0 \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-1} \\
\beta_i &= 2\beta_{i-n} + \beta_{i-1} \\
\beta_i &= 2\beta_{i-n} + 2\beta_{i-1-n}
\end{aligned}$$

If $i - n \equiv 2 \pmod 3$:

$$\begin{aligned}
\beta_n \cdot \beta_{i-n} &= \beta_i + \beta_{i-1} + 2\beta_{i-2} \\
2\beta_0 \cdot \beta_{i-n} &= \beta_i + \beta_{i-1} + 2\beta_{i-2} \\
\beta_i &= 2\beta_{i-n} + 2\beta_{i-1} + \beta_{i-2} \\
\beta_i &= 2\beta_{i-n} + 2(2\beta_{i-1-n} + 2\beta_{i-2-n}) + 2\beta_{i-2-n} \\
\beta_i &= 2\beta_{i-n} + \beta_{i-1-n}
\end{aligned}$$

If we combine these in a formula:

$$\beta_i = 2\beta_{i-n} + r \cdot \beta_{i-1-n} \tag{B.3}$$

$$r = \begin{cases} 0 & \text{if } i - n \equiv 0 \bmod 3 \\ 2 & \text{if } i - n \equiv 1 \bmod 3 \\ 1 & \text{if } i - n \equiv 2 \bmod 3 \end{cases}$$

We perform the reduction operations by choosing all possible irreducible Charlier binomials as some of these are given in Table $B.1$. Since we have three kinds of irreducible Charlier binomials, we get three different reduction formulas. Each reduction formulas changes due to the difference $i - n$ and so the value of the index i (mod 3), which is the index of reduced element β_i. Table $B.2$ gives the reduction complexity of β_i for each form of the irreducible Charlier binomials.

Table B.2: Reduction Complexity of β_i

Form	# Constant Multiplications	# Additions
$\beta_n + \beta_0$ ($n \equiv 0 \bmod 3$)	1	0
$\beta_n + \beta_0$ ($n \equiv 1 \bmod 3$)	2	2
$\beta_n + \beta_0$ ($n \equiv 2 \bmod 3$)	2	1

There are also irreducible Charlier binomials in the form $\beta_n + 2\beta_0$. In the reduction operations, they have the same number of additions with $\beta_n + \beta_0$. In hardware implementations, there is no difference in the selection of binomials $\beta_n + \beta_0$ or $\beta_n + 2\beta_0$ in the way of reduction complexity.

Appendix C

IRREDUCIBLE HERMITE BINOMIALS AND REDUCTION

Table C.1: Irreducible Hermite Binomials

$\beta_3 + \beta_2$	$\beta_4 + \beta_2$	$\beta_{11} + \beta_0$
$\beta_{12} + \beta_2$	$\beta_7 + \beta_2$	$\beta_{26} + \beta_0$
$\beta_{15} + \beta_2$	$\beta_{19} + \beta_2$	$\beta_{35} + \beta_0$
$\beta_{60} + \beta_2$	$\beta_{28} + \beta_2$	$\beta_{119} + \beta_0$
$\beta_{111} + \beta_2$	$\beta_{67} + \beta_2$	$\beta_{146} + \beta_0$
$\beta_{183} + \beta_2$	$\beta_{151} + \beta_2$	$\beta_{242} + \beta_0$

We list the irreducible binomials with respect to the indices n (mod 3). The reduction operations are different due to the chosen binomials from each column of the Table $C.1$, since the multiplication of Hermite polynomials differs with respect to the values of the indices in mod 3, as it is given in Theorem 4.1.3.

Let f be the irreducible Hermite binomial. Reduction operation with respect to modulo f can be performed as follows:

We deal with each binomial form respectively, so first let's take $f = \beta_n + \beta_2$ where $n \equiv 0 \pmod 3$ and let $n \leq i \leq 2n - 2$. Then,

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i \\ 2\beta_2 \cdot \beta_{i-n} &= \beta_i \\ \beta_i &= 2\beta_{i-n}.\beta_2 \end{aligned}$$

We formulate this by using Theorem 4.1.3 as follows:

$$\beta_i = 2\beta_{i-n+2} + r \cdot (s \cdot \beta_{i-n} + t \cdot \beta_{i-n-2}) \tag{C.1}$$

$$r = \begin{cases} 0 & \text{if } i - n \equiv 0 \bmod 3 \\ 1 & \text{otherwise.} \end{cases}$$

$$s = \begin{cases} 1 & \text{if } i - n \equiv 1 \bmod 3 \\ 2 & \text{if } i - n \equiv 2 \bmod 3 \end{cases}$$

$$t = \begin{cases} 1 & \text{if } i - n \equiv 2 \bmod 3 \\ 0 & \text{otherwise.} \end{cases}$$

Remark C.0.4. *If $n \equiv 0$ (mod 3) then by Theorem 4.1.3, l is zero and note that $\beta_n = 2 \cdot \beta_2$.*

Now, let $f = \beta_n + \beta_2$, where $n \equiv 1 \pmod 3$. The reduction of β_i where $n \leq i \leq 2n-2$ differs with respect to the value of $i - n \pmod 3$. Let's give the reduction formulas for the values zero, one and two, respectively.

If $i - n \equiv 0 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i \\ \beta_i &= 2\beta_2 \cdot \beta_{i-n} \\ \beta_i &= 2\beta_{i-n+2} \end{aligned}$$

If $i - n \equiv 1 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i + \beta_{i-2} \\ 2\beta_2 \cdot \beta_{i-n} &= \beta_i + \beta_{i-2} \\ \beta_i &= 2\beta_2 \cdot \beta_{i-n} + 2\beta_{i-2} \\ \beta_i &= 2\beta_{i-n+2} + \beta_{i-n} + 2\beta_{i-2} \\ \beta_i &= 2\beta_{i-n+2} + \beta_{i-n} + 2(2\beta_{i-2-n+2} + 2\beta_{i-2-n} + \beta_{i-2-n-2} + 2\beta_{i-4-n+2}) \\ \beta_i &= 2\beta_{i-n+2} + 2\beta_{i-n} + 2\beta_{i-n-2} + 2\beta_{i-n-4} \end{aligned}$$

If $i - n \equiv 2 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-2} \\ 2\beta_2 \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-2} \\ \beta_i &= 2\beta_2 \cdot \beta_{i-n} + \beta_{i-2} \\ \beta_i &= 2\beta_{i-n+2} + 2\beta_{i-n} + \beta_{i-n-2} + \beta_{i-2} \\ \beta_i &= 2\beta_{i-n+2} + 2\beta_{i-n} + \beta_{i-n-2} + 2\beta_{i-2-n+2} \\ \beta_i &= 2\beta_{i-n+2} + \beta_{i-n} + \beta_{i-n-2} \end{aligned}$$

If we combine these, we get the equation:

$$\beta_i = 2\beta_{i-n+2} + u \cdot [v \cdot (\beta_{i-n} + \beta_{i-n-2}) + w \cdot \beta_{i-n-4}] \tag{C.2}$$

$$u = \begin{cases} 0 & \text{if } i - n \equiv 0 \bmod 3 \\ 1 & \text{otherwise.} \end{cases}$$

$$v = \begin{cases} 2 & \text{if } i - n \equiv 1 \bmod 3 \\ 1 & \text{if } i - n \equiv 2 \bmod 3 \end{cases}$$

$$w = \begin{cases} 2 & \text{if } i-n \equiv 1 \bmod 3 \\ 0 & \text{otherwise.} \end{cases}$$

Now let $f = \beta_n + \beta_0$, where $n \equiv 2 \pmod 3$. Since the value of $n \equiv 2 \pmod 3$, the multiplication changes with respect to the values of $i - n \pmod 3$, where $n \leq i \leq 2n-2$. We compute the reduction respectively.

If $i - n \equiv 0 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i \\ 2\beta_0 \cdot \beta_{i-n} &= \beta_i \\ \beta_i &= 2\beta_{i-n} \end{aligned}$$

If $i - n \equiv 1 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-2} \\ 2\beta_0 \cdot \beta_{i-n} &= \beta_i + 2\beta_{i-2} \\ \beta_i &= 2\beta_{i-n} + \beta_{i-2} \end{aligned}$$

If $i - n \equiv 2 \pmod 3$:

$$\begin{aligned} \beta_n \cdot \beta_{i-n} &= \beta_i + \beta_{i-2} + 2\beta_{i-4} \\ 2\beta_0 \cdot \beta_{i-n} &= \beta_i + \beta_{i-2} + 2\beta_{i-4} \\ \beta_i &= 2\beta_{i-n} + 2\beta_{i-2} + \beta_{i-4} \end{aligned}$$

If we combine these in a formula:

$$\beta_i = 2\beta_{i-n} + h \cdot (y \cdot \beta_{i-2} + z \cdot \beta_{i-4}) \tag{C.3}$$

$$h = \begin{cases} 0 & \text{if } i-n \equiv 0 \bmod 3 \\ 1 & \text{otherwise.} \end{cases}$$

$$y = \begin{cases} 1 & \text{if } i-n \equiv 1 \bmod 3 \\ 2 & \text{if } i-n \equiv 2 \bmod 3 \end{cases}$$

$$z = \begin{cases} 1 & \text{if } i-n \equiv 2 \bmod 3 \\ 0 & \text{otherwise.} \end{cases}$$

We perform the reduction operations by choosing all possible irreducible Hermite binomials as some are given in the Table $C.1$. Since we have three kinds of irreducible Hermite binomials, we get three reduction formulas in Equations C.1, C.2 and C.3. In the form $\beta_n + \beta_2$ where $n \equiv 0 \pmod 3$, the multiplication of $\beta_n \cdot \beta_{i-n}$ directly gives β_i, so we can reduce β_i in one step for

this form. In the other two forms, each reduction formulas changes due to the difference $i - n$ and so the value of the index i (mod 3), which is the index of reduced element β_i. Reduction in Equation C.3 is not completed in one step. The terms β_{i-2}, β_{i-4} should be reduced until the indices drop into the interval $[0, n-1]$. We do not carry on these reductions, since it depends on the value of the index i. Table $C.2$ gives the upper bound of the reduction complexity for each form of the irreducible Hermite binomials. Since the reduction in Equations C.3 is not completed, we give the signed numbers in Table $C.2$ which are the numbers of constant multiplications and additions for one step of the reduction. The total number of multiplications and additions for this binomial increases by the number of reduction steps.

Table C.2: Reduction Complexity of β_i

Form	# Constant Multiplications	# Additions
$\beta_n + \beta_2$ ($n \equiv 0$ mod 3)	2	2
$\beta_n + \beta_2$ ($n \equiv 1$ mod 3)	4	3
$\beta_n + \beta_0$ ($n \equiv 2$ mod 3)	2^*	2^*

There are also irreducible Hermite binomials in the form $\beta_n + 2\beta_0$ and $\beta_n + 2\beta_2$. In the reduction operations, they have the same number of additions with $\beta_n + \beta_0$ and $\beta_n + \beta_2$, respectively. Therefore in hardware implementations, the choice of the binomials $\beta_n + \beta_0$ or $\beta_n + 2\beta_0$ and the choice of $\beta_n + \beta_2$ or $\beta_n + 2\beta_2$ do not effect the reduction complexity.

CURRICULUM VITAE

PERSONAL INFORMATION

Surname, Name: Özel, Canan
Nationality: Turkish
Date and Place of Birth: 1982 - Manisa
Marital Status: Married
email: canancozel@gmail.com

EDUCATION

Degree	Institution	Year of Graduation
M.S.	METU, Department of Cryptography	2008
B.S.	METU, Department of Mathematics	2005
High School	Eskişehir Fatih Science High School	2000

PROFESSIONAL EXPERIENCE

Year	Place	Enrollment
2006 - 2013	Institute of Applied Mathematics, METU	Research Assistant

PUBLICATIONS

Papers in Progress

S. Akleylek, F. Özbudak, C. Özel, On the Arithmetic Operations over Finite Fields of Characteristic Three with Low Complexity, submitted, 2013.

Papers published in International Conference Proceedings

S. Akleylek, F. Özbudak, C. Özel, Hermite Polynomial Representation for Finite Fields of Characteristic Three, 5th International Information Security and Cryptology Conference (ISC-TURKEY 2012), vol.5 pp.155-159 Ankara, 2012.

Presentations in International Conferences

S. Akleylek, F. Özbudak, C. Özel, On the Multiplication over Finite Fields of Characteristic Three in Hermite Polynomial Representation, International Conference on Applied and Computational Mathematics (ICACM), Ankara, 2012.

S. Akleylek, F. Özbudak, C. Özel, Charlier Polynomial Representation for Finite Fields of Characteristic Three, 18th International Conference on Applications of Computer Algebra (ACA 2012), Sofia, Bulgaria, 2012.