Regular paper

Journal of the KIECS. pp. 621-628, vol. 12, no. 4, Aug. 31 2017, t. 84, pISSN 1975-8170 | eISSN 2288-2189
http://dx.doi.org/10.13067/JKIECS.2017.12.4.621

유한체위에서 정규기저의 고속생성과 저비용 연산 알고리즘의 구현에 관한 연구

김용태

On Implementations of Algorithms for Fast Generation of Normal Bases and Low Cost Arithmetics over Finite Fields

Yong-Tae Kim

요 약

유한체위에서 사칙연산의 H/W 구현의 효율성은 사용하는 유한체의 기저 선택에 의해서 크게 좌우된다. 그러한 H/W 구현의 효율성의 관점에서 보면, 정규기저가 가장 적절한 이유는, 표수가 2인 유한체 $GF(2^n)$의 원소를 $GF(2)$위에서 정규기저로 표현하면, 원소의 제곱은 단순하게 좌표의 순환이동이 되기 때문이다. 본 논문에서는, 모든 유한체에서 관용기저로 부터 정규기저로 고속으로 변환하는 알고리즘을 소개하였으며 그 알고리즘을 이용한 H/W 구현결과와 우리의 방법으로 구현한 정규기저를 이용하여, 유한체 $GF(2^n)$위에서 두 원소의 곱셈과 역원을 구하는 효율적인 알고리즘에 따른 프로그램과 H/W 구현결과를 제시하였다.

ABSTRACT

The efficiency of implementation of the arithmetic operations in finite fields depends on the choice representation of elements of the field. It seems that from this point of view normal bases are the most appropriate, since raising to the power 2 in $GF(2^n)$ of characteristic 2 is reduced in these bases to a cyclic shift of the coordinates. We, in this paper, introduce our algorithm to transform fastly the conventional bases to normal bases and present the result of H/W implementation using the algorithm. We also propose our algorithm to calculate the multiplication and inverse of elements with respect to normal bases in $GF(2^n)$ and present the programs and the results of H/W implementations using the algorithm.

키워드

Finite Field, Normal Basis, Multiplication Function, Boolean Matrix, Multiplication, Inverse
유한체, 정규 기저, 곱셈 함수, 부울 행렬, 곱셈, 역원

Ⅰ. 서 론

유한체 이론은 전환(switching)이론, 컴퓨터 H/W 상에서의 연산, 오류 정정 부호이론과 암호학 등에 폭넓게 응용되기 때문에 꾸준히 주목을 받고 있다. 그런데 유한체위에서 사칙연산의 H/W 구현의 효율성은 사용하는 유한체의 기저(basis)의 선택에 의해서 크게 좌우된다. 가장 큰 이유는 유한체의 원소를 곱하거나 역

* 교신저자: 광주교육대학교 수학교육과
• 접 수 일 : 2017. 04. 27
• 수정완료일 : 2017. 07. 13
• 게재확정일 : 2017. 08. 01
• Received : Apr 27, 2017, Revised : July 13, 2017, Accepted : Aug 01, 2017
• Corresponding Author : Yongtae Kim
Dept. of Mathematics Education, Gwangju National University of Education
Email : ytkim@gnue.ac.kr

원을 구할 경우에 유한체의 원소를 관용기저(conventional basis)로 표현하면 계수의 계산이 매우 복잡하지만, 원소를 정규기저(normal basis)로 표현하면, 원소의 제곱은 단순하게 좌표의 순환이동(cyclic shift)이 되기 때문이다. 따라서 유한체에서의 곱셈이나 역원을 구하는 경우, 관용기저보다 정규기저를 사용하면 H/W 실행시간이 현저하게 단축된다[1]. 모든 유한체가 정규기저를 갖는다는 사실은 1888년에 K. Hensel에 의해서 밝혀졌으나[2], 실제로 유한체위에서 정규기저를 이용한 컴퓨터상에서의 연산 스킴이 제안된 것은 그 후 100년 뒤의 일이다[3]. 그 후, 유한체의 다양한 정규기저[4]가 발견되었으며 그 중에서 가장 효율성이 높은 최적정규기저를 이용한 곱셈기 [5]도 등장하였다. 그러나 최적정규기저는 매우 제한된 유한체에서만 존재하기 때문에 다양한 유한체를 계속적으로 바꾸어가면서 사용하는 타원곡선 암호법이나 공개키 암호법에서는 최적정규기저보다는 저 복잡도 정규기저를 가지는 유한체가 가장 많이 사용되고 있다. 본 논문에서는, 실행속도의 효율성을 위하여 최근의 수열 고속생성에 관한 연구결과[6-7]를 적용한 표수(characteristic)가 2인 유한체 $GF(2^n)$의 정규기저를 고속으로 생성하는 알고리즘과 유한체를 생성하는 기약다항식을 이용하여 정규기저로 표현된 $GF(2^n)$의 두 원소의 곱셈과 역수를 구하는 효율적인 알고리즘을 제안하기로 한다. 또한 그 알고리즘에 따른 프로그램과 구체적인 H/W 구현결과를 제시하기로 한다.

II. Massey-Omura 스킴

이 장에서는 Massey-Omura가 제안한 유한체위에서의 연산 스킴을 설명하면서 유한체 연산에서 정규기저의 특성을 알아보기로 한다.

2.1 Massey-Omura 스킴의 개요

2^n 개의 원소로 이루어지는 유한체 $GF(2^n)$는 유한체 $GF(2)$ 위에서의 기약다항식 $p(x)$를 찾아 $GF(2)[x]/(p(x))$로 나타내 진다. 따라서 유한체 $GF(2^n)$의 모든 원소는 $GF(2)$위에서 차수가 $n-1$인 다항식으로 나타내지며, $GF(2^n)$는 $GF(2)$위에서 관용기저$\{1, x, x^2, \cdots, x^{n-1}\}$를 갖게 된다. 잘 알려진 바와 같이 관용기저는 유한체의 원소를 가장 간단하게 표현할 수 있으나, 두 원소의 곱셈 또는 어떤 원소의 역원을 계산할 때에는 초등대수학에서 알 수 있는 바와 같이 각 기저의 계수 계산이 매우 복잡하다는 단점이 있다. 이러한 단점을 극복하기 위해서 Massey-Omura[3]는 관용기저 대신 정규기저를 사용하는 방안을 소개하였다. 지금부터 그 방법의 개요를 설명하기로 한다.

$\{w, w^2, \cdots, w^{2^{n-1}}\}$이 $GF(2)$위에서 유한체 $GF(2^n)$의 정규기저이고 α, β를 $GF(2^n)$의 임의의 두 원소라고 하자. 그러면

$$\alpha = a_0 w + a_1 w^2 + \cdots + a_{n-1} w^{2^{n-1}}, \quad (1)$$

$$\beta = b_0 w + b_1 w^2 + \cdots + b_{n-1} w^{2^{n-1}}. \quad (2)$$

단 $a_i, b_i \in GF(2)$이다. 이 경우에 표기를 간단하게 하기 위해서 α, β를 계수만을 이용하여 다음과 같이 표현하기로 한다.

$$\alpha = (a_0, a_1, \cdots, a_{n-1}),\ \beta = (b_0, b_1, \cdots, b_{n-1}). \quad (3)$$

그러면 유한체 $GF(2^n)$는 표수가 2이므로,

$(\alpha+\beta)^2 = \alpha^2 + \beta^2$, $\alpha^{2^n} = \alpha$ 인 성질을 이용하면

$$\begin{aligned}\alpha^2 &= (a_0 w + a_1 w^2 + \cdots + a_{n-1} w^{2^{n-1}})^2 \\ &= a_{n-1}\, w + a_0 w^2 + \cdots + a_{n-2} w^{2^{n-1}} \\ &= (a_{n-1}, a_0, \cdots, a_{n-2}). \end{aligned} \quad (4)$$

만일

$$\gamma = \alpha\beta = (c_0, c_1, \cdots, c_{n-1}) \quad (5)$$

라 하고, 곱셈함수(Bool 행렬) f를 이용하여 γ의 마지막 계수를 다음과 같이 표기하기로 하자.

$$c_{n-1} = f(a_0, a_1, \cdots, a_{n-1}; b_0, b_1, \cdots, b_{n-1}). \quad (6)$$

그러면 식 (4)에 의해서

$$\begin{aligned}\gamma^2 &= \alpha^2\beta^2 \\ &= (a_{n-1}, a_0, \cdots, a_{n-2})(b_{n-1}, b_0, \cdots, b_{n-2}) \\ &= (c_{n-1}, c_0, \cdots, c_{n-2}) \end{aligned} \quad (7)$$

이 되며, 식 (6)에 의해서

$$c_{n-2} = f(a_{n-1}, a_0, \cdots, a_{n-2}; b_{n-1}, b_0, \cdots, b_{n-2}). \quad (8)$$

계속해서 γ의 지수를 올려 가면 결국 다음과 같은 결과를 얻게 된다.

$$c_{n-1}=f(a_0,a_1,\cdots,a_{n-1};b_0,b_1,\cdots,b_{n-1})$$
$$c_{n-2}=f(a_{n-1},a_0,\cdots,a_{n-2};b_{n-1},b_0,\cdots,b_{n-2})$$
$$\vdots$$
$$c_0=f(a_1,a_2,\cdots,a_0;b_1,b_2,\cdots,b_0). \quad (9)$$

곱셈함수 f의 이러한 성질을 이용하면 유한체의 두 원소의 곱셈한 결과의 계수들을 아주 빠르게 구할 수 있다. 따라서 유한체의 관용기저를 알고 있는 경우에, 그 관용기저를 정규기저로 변환시키는 일이 중요하게 된다.

III. 정규기저 생성 알고리즘

이 장에서는 주어진 유한체의 관용기저로부터 정규기저를 고속으로 생성하는 알고리즘을 제안하고 그 알고리즘에 의한 H/W상에서의 수치실험의 예를 제시하기로 한다.

3.1 정규기저의 생성과정

정수 n은 홀수이고, $GF(2)$위에서 유한체 $GF(2^n)$의 관용기저를 $\{1,x,x^2,\cdots,x^{n-1}\}$이라고 하자. 또한 다음의 n개의 등식

$$(x^{i-1})^2=m_{i1}1+m_{i2}x+\cdots+m_{in}x^{n-1}, i=1,\cdots,n. \quad (10)$$

의 n차 계수행렬을 $M=(m_{ij})$라고 하자. 그러면

$$(1,x^2,\cdots,x^{2^{n-1}})^T=M\bullet(1,x,\cdots,x^{n-1})^T \quad (11)$$

이 성립한다. 단, T는 행렬의 전치를 의미한다.

이제, 식 (10)의 양변을 2^{j-1} 제곱으로 올리면 다음의 식을 얻게 된다.

$$(x^{i-1})^{2^{j-1}}=m_{i1}1+m_{i2}x^{2^{j-1}}+\cdots+m_{in}(x^{n-1})^{2^{j-1}} \quad (12)$$
$$, i=1,\cdots,n.$$

그러면 식 (11)과 마찬가지로 다음 식이 성립한다.

$$(1,x^{2^j},\cdots,(x^{n-1})^{2^j})^T=M^j\bullet(1,x,\cdots,x^{n-1})^T. \quad (13)$$

그런데, 모든 $x\in GF(2^n)$에 대하여 $x^{2^n}=x$ 이고 $\{1,x,x^2,\cdots,x^{n-1}\}$이 관용기저이므로, $M^n=I_n$(n차 단위행렬) 이므로 $g(\eta)=\eta^n-1$ 이 n차 계수행렬 M의 최소다항식이다. 지금,

$$g(\eta)=\Pi_{s=1}^{d}g_s(\eta) \quad (14)$$

을 $g(\eta)$의 $GF(2)$위에서의 기약 인수분해이고 $\deg g_i=e_i$라고 하자. 그러면 함수 $g(\eta)$는 중근을 갖지 않으며, 다항식 $g_s(\eta)$의 근인 $\eta_{s1},\cdots,\eta_{se_s}$는 행렬 M의 고유치(eigenvalues)가 된다. 그러면

[2]의 정리 2.14에 의해서 다음 식이 성립한다.

$$\eta_{si}=(\eta_{s1})^{2^{i-1}}, i=1,\cdots,e_s. \quad (15)$$

3.2 정규기저의 생성 알고리즘

3.1 절에서 유한체의 관용기저를 정규기저로 변환하는 과정을 제안하였다. 그 과정을 알고리즘화하면 다음과 같다.

Algorithm

Step 1. 다음의 연립방정식의 근을 구한다.

$$u_{s1}(M-\eta_{s1}I_n)=0. \quad (16)$$

편의상 $u_{s1}=(u_{s1}(1),\cdots,u_{s1}(n))$이라고 하자.

Step 2.

$$u_{sr}=(u_{s1}(1)^{2^{r-1}},\cdots,u_{s1}(n)^{2^{r-1}}), r=1,\cdots,e_s. \quad (17)$$

Step 3.

$$u_s=u_{s1}+\cdots+u_{se_s}, s=1,\cdots,d \quad (18)$$

라고 하면 u_s의 모든 성분은 $GF(2)$의 원소이다.

Step 4.

$$z=u_1+\cdots+u_d=(z_1,\cdots,z_n) \quad (19)$$

로 놓으면 모든 z_i 역시 $GF(2)$의 원소이다.

Step 5.

$$\theta=z_1 1+z_2x+\cdots+z_nx^{n-1} \quad (20)$$

로 놓자.

그러면 위의 알고리즘에서 다음과 같은 정리를 얻게 된다.

정리 1. 위의 알고리즘에서 구한 θ는 $GF(2)$위에서 유한체 $GF(2^n)$의 정규기저를 n과 2의 다항식 시간(polynomial time) 내에 생성된다.

(증명) 위의 3.1절의 정규기저의 생성과정에서

$$\theta^{2^j}=zM^j(1,x,\cdots,x^{n-1}), j=0,1,\cdots,n-1. \quad (21)$$

그런데 Step 4에 의하면, z는 식 (10)에서 정의

한n차 계수행렬을 $M=(m_{ij})$의 서로 다른 고유치 $\alpha_1,\cdots,\alpha_n$에 대응하는 고유벡터 $y_1,y_2,\cdots,y_n$의 합 즉, $z=y_1+y_2+\cdots+y_n$으로 표현할 수 있다. 그렇다면,

$$zM^j=\alpha_1^j y_1+\cdots+\alpha_n^j y_n, j=0,1,\cdots,n-1. \quad (22)$$

이때 만일,

$$\sum_{i=0}^{n-1} a_i zM^i=0,\ a_i \in GF(2) \quad (23)$$

이라면

$$\sum_{v=1}^{n}(\sum_{i=0}^{n-1} a_i\alpha_v^i)y_v=0. \quad (24)$$

그런데 $y_1,y_2,\cdots,y_n$는 일차독립이므로

$$\sum_{i=0}^{n-1} a_i\alpha_v^i=0,\ v=1,2,\cdots,n \quad (25)$$

이다. 따라서 고유치 $\alpha_1,\cdots,\alpha_n$와 그들의 Vandermonde행렬식의 성질에 의해서 모든 i에 대해서 계수 $a_i=0$이 된다. 그러므로 $z,zM,\cdots,zM^{n-1}$은 일차독립이고, 그에 따라서 위의 Algorithm에서 구한 $\theta,\theta^2,\cdots,\theta^{2^{n-1}}$ 역시 일차독립이다. 즉, $\{\theta,\theta^2,\cdots,\theta^{2^{n-1}}\}$는 $GF(2)$위에서 유한체 $GF(2^n)$의 정규기저이다.

3.3 정규기저의 H/W 구현결과

Pentium 166 MHZ CPU에서 Mathematica 4.0[8]을 이용한 프로그램을 사용하여 얻은 H/W 구현 결과는 다음과 같다. 단, p는 $GF(2)$위에서의 유한체 $GF(2^{21})$의 기약다항식, f는 $GF(2)$위에서의 행렬 M의 최소다항식의 기약인수분해, qq는 Algorithm의 1단계에서의 고유벡터, w는 Algorithm의 3단계에서의 u_s, z는 Algorithm의 4단계에서의 벡터이다.

$p=1+x^2+x^3+x^6+x^9+x^{12}+x^{15}+x^{18}+x^{21}$

$f=(1+x)(1+x+x^2)(1+x+x^3)(1+x^2+x^3)(1+x+x^2+x^4+x^6)(1+x^2+x^4+x^5+x^6)$

$qq1=(1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$

$w1=(1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$

$qq2=(0,1,1+x,x,0,1+x,1+x,1,1+x,0,x,x,1,0,1,1+x,x,0,0,1+x,1)$

$w2=(0,0,1,1,0,1,1,0,1,0,1,1,0,0,0,1,1,0,0,1,0)$

$qq3=(x^2,1+x+x^2,1,x,1,1+x,1+x,x+x^2,1+x^2,1+x+x^2,x,1+x^2,x+x^2,x,x^2,1,1,1+x,x+x^2,1+x,1)$

$w3=(0,1,1,0,1,1,1,0,1,1,0,1,0,0,0,1,1,1,0,1,1)$

$qq4=(x+x^2,x+x^2,0,0,1+x^2,1+x,x,0,x,1+x,x,x+x^2,1,1,1+x+x^2,x,1+x,1+x^2,1+x+x^2,x^2,1)$

$w4=(0,0,0,0,0,0,1,0,1,0,1,1,0,1,1,10,0,1,1,1)$

$qq5=(x+x^2+x^5,x+x^2+x^3,1+x^2+x^3+x^4+x^5,1+x^2+x^5,1+x^3+x^5,x+x^2+x^3+x^4,1+x+x^4,1,1+x+x^4+x^5,x+x^2+x^4+x^5,x,x^2+x^3+x^4,x^2+x^3+x^4+x^5,x+x^2+x^3+x^4,x+x^3+x^4+x^5,x^3+x^4+x^5,x^5,x+x^2+x^5,1+x+x^3+x^4,1+x^3+x^4+x^5,1)$

$w5=(1,0,1,1,1,0,0,0,1,1,0,0,1,0,1,1,1,1,0,1,0)$

$qq6=(x+x^4,x^4,x+x^3+x^4+x^5,x+x^3+x^5,1+x+x^4,1+x^5,x^3+x^4+x^5,1+x^3+x^4+x^5,x^2+x^5,1+x^2+x^3,x,x^5,1+x+x^2+x^4+x^5,1+x+x^2+x^5,1+x+x^3+x^4,1+x^2+x^4+x^5,1+x+x^3+x^4,x+x^2+x^3+x^4,x^2+x^3+x^4+x^5,1+x+x^4+x^5,1)$

$w6=(0,1,0,1,0,01,1,1,1,1,0,1,0,0,0,0,1,0,0,0)$

$z=(0,0,1,1,0,0,0,1,1,1,1,1,0,1,0,0,1,1,1,0,0)$

IV. 정규기저와 곱셈함수를 이용한 곱셈과 역원의 구현

최근에 초대규모집적회로(VLSI)를 내장한 컴퓨터에서의 유한체 연산의 효율적인 H/W 구현 방안에 대한 연구[9]와 함께, network 공학 등의 다양한 응용분야에서 유한체위에서의 고속연산에 대한 연구[10]가 활발하게 수행되고 있다. 또한 유한체위에서의 효율적인 연산에 관한 알고리즘 또는 프로토콜[11]에 관한 연구 결과도 제시되고 있다. 이장에서는, 그러한 연구 결과를 참고하여, 유한체 $GF(2)$ 위에서 주어진 기약다항식 $p(x)$에 의해서 생성되는 Ⅱ장의 식(6)의 곱셈함수 f를 구축하기로 한다. 그런 다음, Ⅲ장의 방법을 적용하여 얻은 $GF(2^n)$의 정규기저와 곱셈함수 f를 이용하여 $GF(2^n)$의 원소들의 곱셈과 원소의 역원을 구하는 알고리즘을 구축한다. 또한 그 알고리즘을 적용한 곱셈과 역원의 H/W 구현결과를 제시하기로 한다.

4.1 곱셈함수의 구축

다항식

$$p(x)=x^n+c_1x^{n-1}+c_2x^{n-2}+\cdots+c_{n-1}x+c_n \quad (26)$$

을 $GF(2)$위에서 유한체 $GF(2^n)$의 n차 기약다항식이라고 하자. 그리고 $\{1,x,x^2,\cdots,x^{n-1}\}$, $\{\theta,\theta^2,\cdots,\theta^{2^{n-1}}\}$을 각각 $GF(2)$위에서 유한체 $GF(2^n)$의 관용기저, Ⅲ장의 방법에 의해서 구한 정규기저라고 하자. 그러면 임의의 $\alpha,\beta\in GF(2^n)$는 다음과 같이 표현된다.

$$\alpha=a_0\theta+a_1\theta^2+\cdots+a_{n-1}\theta^{2^{n-1}}, \quad (27)$$

$$\beta=b_0\theta+b_1\theta^2+\cdots+b_{n-1}\theta^{2^{n-1}}, \quad (28)$$

$a_i,b_i\in GF(2)$이다. 이 경우에 표기를 간단하게 하기 위해서 α,β를 계수만을 이용하여 다음과 같이 표현하기로 한다.

$$\alpha=(a_0,a_1,\cdots,a_{n-1}),\ \beta=(b_0,b_1,\cdots,b_{n-1}). \quad (29)$$

그러면

$$\begin{aligned}\alpha^2&=(a_0\theta+a_1\theta^2+\cdots+a_{n-1}\theta^{2^{n-1}})^2\\&=a_{n-1}\theta+a_0\theta^2+\cdots+a_{n-2}\theta^{2^{n-1}}\\&=(a_{n-1},a_0,\cdots,a_{n-2})\end{aligned} \quad (30)$$

만일

$$\gamma=\alpha\beta=(c_0,c_1,\cdots,c_{n-1}) \quad (31)$$

라 하고, 곱셈함수 f를 이용하여 γ의 마지막 계수를 다음과 같이 표기하기로 하자.

$$c_{n-1}=f(a_0,a_1,\cdots,a_{n-1};b_0,b_1,\cdots,b_{n-1}). \quad (32)$$

그러면 식 (4)에 의해서

$$\begin{aligned}\gamma^2&=\alpha^2\beta^2\\&=(a_{n-1},a_0,\cdots,a_{n-2})(b_{n-1},b_0,\cdots,b_{n-2})\\&=(c_{n-1},c_0,\cdots,c_{n-2})\end{aligned} \quad (33)$$

이므로 정규기저로 표현된 유한체 $GF(2^n)$의 원소의 곱셈이 훨씬 쉬워진다.

이제, 유한체 $GF(2^n)$의 정규기저 $\{\theta,\theta^2,\cdots,\theta^{2^{n-1}}\}$로 표현한 식(27), (28) 또는 (29)의 임의의 두 원소 α, β를 효율적으로 곱하는 식(32)의 곱셈함수 f를 구축하기로 한다.

$$\alpha\beta=\sum_{i,j=0}^{n-1}a_ib_j\theta^{q^i+q^j}=\sum_{i,j=0}^{n-1}a_ib_j\theta^{(q^{i-j}+1)q^j}, \quad (34)$$

여기에서 $i-j$는 n을 법으로 계산한 값이며, 식 (34)의 $\theta^{(q^{i-j}+1)q^j}$를 정규기저 $\{\theta,\theta^2,\cdots,\theta^{2^{n-1}}\}$로 표현하고, 그 계수를 $t_{i,j}$를 표현하면 다음과 같게 된다.

$$\begin{aligned}\theta^{(q^{i-j}+1)q^j}&=((\theta^{q^{i-j}+1})^{q^j}=(\sum_{k=0}^{n-1}t_{i-j,k}\theta^{q^k})^{q^j}\\&=\sum_{k=0}^{n-1}t_{i-j,k}\theta^{q^{k+j}}=\sum_{k=0}^{n-1}t_{i-j,m-j}\theta^{q^m},\end{aligned} \quad (35)$$

단, $m-j$와 $k+j$는 n을 법으로 계산한 값이다.

이때, $p_m=\sum_{i,j=0}^{n-1}t_{i-j,m-j}a_ib_j$라 놓으면 식 (34)는 다음과 같게 된다.

$$\alpha\beta=\sum_{m=0}^{n-1}p_m\theta^{q^m}. \quad (36)$$

지금, $f_{i,j}=t_{i-j,k-j}$ 로 놓고 $i-j$와 $k-j$를 n을 법으로 계산하면 다음과 같은 식을 얻게 된다.

$$\begin{aligned}p_m&=\sum_{i,j=0}^{n-1}t_{i-j,m-j}a_ib_j=\sum_{k,l=0}^{n-1}t_{k-l,m-l}a_{k+m}b_{l+m}\\&=\sum_{i,j=0}^{n-1}f_{i,j}a_{i+m}b_{j+m}=\sum_{i,j=0}^{n-1}f_{i,j}S^m(a_i)S^m(b_j),\end{aligned} \quad (37)$$

단, S^m은 좌표벡터의 m개의 좌표를 순환시키는 연산을 뜻한다. 이때. 다항식

$$f(x,y)=\sum_{i,j=0}^{n-1}f_{i,j}a_ib_j \quad (38)$$

이 우리가 찾는 곱셈함수이다.

4.2 프로그램과 H/W 구현결과

Pentium 166 MHZ CPU에서 Mathematica 4.0[8]을 이용한 프로그램을 사용하여 얻은 H/W 구현 결과는 다음과 같다. 단, p는 $GF(2)$위에서의 105차 기약다항식, f는 곱셈함수를 생성하는 Mathematica 프로그램, invv는 역원을 구하는 Mathematica 프로그램, xx와 yy는 기약다항식 p에 의해서 생성된 정규기저로 표현된 $GF(2^{105})$의 두 원소, zz는 xx와 yy의 곱, hh는 xx의 역원이다.

$$\begin{aligned}p(x)=1&+x+x^4+x^5+x^6+x^8+x^9+x^{16}+x^{17}+\\&x^{20}+x^{21}+x^{22}+x^{64}+x^{65}+x^{68}+x^{69}+x^{70}+x^{72}+\\&x^{73}+x^{80}+x^{81}+x^{84}+x^{85}+x^{86}+x^{96}+x^{97}+\\&x^{100}+x^{101}+x^{102}+x^{104}+x^{105}.\end{aligned}$$

```
f[xx_,yy_]:=Block[{},Table[cc[k-1]=0,{k,1,105}];
      aa=Table[b[k-1]=xx[[1,k]],{k,1,105}];
      bb=Table[c[k-1]=yy[[1,k]],{k,1,105}];
      d=Table[0,{1},{105}];
      For[l=1,l<106,l++,a1=RotateLeft[aa,l+1];
        b1=RotateLeft[bb,l+1];
      Table[b[k-1]=Take[a1,{k,k}][[1]],{k,1,105}];
      Table[c[k-1]=Take[b1,{k,k}][[1]],{k,1,105}];
      cc[105-l]=Mod[b[42]*c[0]+b[104]*c[0]+
        b[26]*c[1]+b[42]*c[1]+b[33]*c[2]+
b[85]*c[2]+b[69]*c[3]+b[93]*c[3]+b[9]*c[4]+
b[99]*c[4]+b[14]*c[5]+b[65]*c[5]+b[17]*c[6]+
b[78]*c[6]+b[21]*c[7]+b[58]*c[7]+b[72]*c[8]+
b[98]*c[8]+b[4]*c[9]+b[70]*c[9]+b[14]*c[10]+
b[97]*c[10]+b[59]*c[11]+b[80]*c[11]+b[49]*c[12]+
b[87]*c[12]+b[64]*c[13]+b[96]*c[13]+b[5]*c[14]+
b[10]*c[14]+b[78]*c[15]+b[80]*c[15]+b[29]*c[16]+
b[77]*c[16]+b[6]*c[17]+b[66]*c[17]+b[21]*c[18]+
b[46]*c[18]+b[52]*c[19]+b[75]*c[19]+b[45]*c[20]+
b[57]*c[20]+b[7]*c[21]+b[18]*c[21]+b[52]*c[22]+
b[72]*c[22]+b[35]*c[23]+b[39]*c[23]+b[83]*c[24]+
b[102]*c[24]+b[32]*c[25]+b[41]*c[25]+b[1]*c[26]+
b[43]*c[26]+b[56]*c[27]+b[85]*c[27]+b[48]*c[28]+
b[76]*c[28]+b[16]*c[29]+b[81]*c[29]+b[66]*c[30]+
b[101]*c[30]+b[40]*c[31]+b[54]*c[31]+b[25]*c[32]
+b[84]*c[32]+b[2]*c[33]+b[43]*c[33]+b[38]*c[34]+
b[69]*c[34]+b[23]*c[35]+b[73]*c[35]+b[83]*c[36]+
b[91]*c[36]+b[55]*c[37]+b[68]*c[37]+b[34]*c[38]+
b[44]*c[38]+b[23]*c[39]+b[53]*c[39]+b[31]*c[40]+
b[102]*c[40]+b[25]*c[41]+b[103]*c[41]+b[0]*c[42]
+b[1]*c[42]+b[26]*c[43]+b[33]*c[43]+b[38]*c[44]+
b[56]*c[44]+b[20]*c[45]+b[53]*c[45]+b[18]*c[46]+
b[67]*c[46]+b[75]*c[47]+b[92]*c[47]+b[28]*c[48]+
b[86]*c[48]+b[12]*c[49]+b[81]*c[49]+b[90]*c[50]+
b[96]*c[50]+b[71]*c[51]+b[74]*c[51]+b[19]*c[52]+
b[22]*c[52]+b[39]*c[53]+b[45]*c[53]+b[31]*c[54]+
b[67]*c[54]+b[37]*c[55]+b[84]*c[56]+b[27]*c[56]+
b[44]*c[56]+b[20]*c[57]+b[76]*c[57]+b[7]*c[58]+
b[79]*c[58]+b[11]*c[59]+b[98]*c[59]+b[87]*c[60]+
b[94]*c[60]+b[62]*c[61]+b[63]*c[61]+b[61]*c[62]+
b[88]*c[62]+b[61]*c[63]+b[95]*c[63]+b[13]*c[64]+
b[88]*c[64]+b[5]*c[65]+b[100]*c[65]+b[17]*c[66]+
b[30]*c[66]+b[46]*c[67]+b[54]*c[67]+b[37]*c[68]+
b[92]*c[68]+b[3]*c[69]+b[34]*c[69]+b[9]*c[70]+
b[73]*c[70]+b[51]*c[71]+b[97]*c[71]+b[8]*c[72]+
b[22]*c[72]+b[35]*c[73]+b[70]*c[73]+b[51]*c[74]+
b[91]*c[74]+b[19]*c[75]+b[47]*c[75]+b[28]*c[76]+
b[57]*c[76]+b[16]*c[77]+b[79]*c[77]+b[6]*c[78]+
b[15]*c[78]+b[58]*c[79]+b[77]*c[79]+b[11]*c[80]+
b[15]*c[80]+b[29]*c[81]+b[49]*c[81]+b[90]*c[82]+
b[101]*c[82]+b[24]*c[83]+b[36]*c[83]+b[32]*c[84]
+b[55]*c[84]+b[2]*c[85]+b[27]*c[85]+b[48]*c[86]+
b[93]*c[86]+b[12]*c[87]+b[60]*c[87]+b[62]*c[88]+
b[64]*c[88]+b[95]*c[89]+b[100]*c[89]+b[50]*c[90]
+b[82]*c[90]+b[36]*c[91]+b[74]*c[91]+b[47]*c[92]
+
b[68]*c[92]+b[3]*c[93]+b[86]*c[93]+b[60]*c[94]+
b[99]*c[94]+b[63]*c[95]+b[89]*c[95]+b[13]*c[96]+
b[50]*c[96]+b[10]*c[97]+b[71]*c[97]+b[8]*c[98]+
b[59]*c[98]+b[4]*c[99]+b[94]*c[99]+b[65]*c[100]+
b[89]*c[100]+b[30]*c[101]+b[82]*c[101]+
b[24]*c[102]+b[40]*c[102]+b[41]*c[103]+
b[103]*c[103]+b[0]*c[104],2]];
Do[d[1,k]=cc[k-1],{k,1,105}];Return[d]];

invv[a9]:=Block[{},
          d9=a9[[1]];Print[d9];
          d9=RotateRight[d9,1];d10=d9;
  d9=f[f[{RotateRight[d9,2]},RotateRight[d9,1]}],
      {d9}][[1]];
  d9=f{RotateRight[d9,3]},{d9}][[1]];
  d9=f[{d10},f[{RotateRight[d9,1]},RotateRight
      [d9,7]}]][[1]];
  d9=f[{RotateRight[d9,13]},{d9}][[1]];
  d9=f[{RotateRight[d9,26]},{d9}][[1]];
  d9=f[{RotateRight[d9,52]},{d9}][[1]];
```

```
    Return[d9]];

xx={{1,1,1,1,0,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1}};
yy={{1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,
     0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,1}};
zz=xx*yy;
zz={{1,0,1,1,1,0,1,1,0,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,
     1,1,1,1,0,1,1,1,1,0,1,1,1,0,1,1,0,1,1,1,1,1,1,1,0,1,
     0,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,
     1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,0,1,
     1,1,1,1,1}};
(*xx^(-1)*)
hh={{0,0,0,0,0,1,1,1,1,0,0,1,0,1,1,0,0,1,1,1,1,1,0,1,0,0,
     0,0,1,0,0,1,1,1,1,1,0,0,1,0,0,0,1,1,1,1,1,1,1,1,1,1,
     1,0,0,0,1,0,1,1,0,0,1,1,1,1,1,1,0,1,1,1,0,0,1,0,1,0,
     0,1,1,1,0,1,0,1,1,0,1,0,1,1,0,1,1,1,0,0,1,0,0,1,1,
     0,0,1,0,1}};
```

V. 결 론

유한체위에서 사칙연산의 H/W 구현의 효율성은 사용하는 유한체의 기저의 선택에 의해서 크게 좌우된다. 그런데 유한체의 원소를 정규기저로 표현하면, 원소의 제곱은 단순하게 좌표의 순환이동이 된다. 따라서 유한체가 응용되는 전환(switching)이론, VLSI를 내장한 컴퓨터 H/W, network 공학 과 암호학 등에 곱셈이나 역원을 구하는 경우, 제한된 유한체에서만 존재하는 최적정규기저보다는 저 복잡도 정규기저를 가지는 유한체가 가장 많이 사용되고 있다. 이러한 정규기저의 중요성을 고려하여, 본 논문의 Ⅲ장에서는 모든 유한체에서 관용기저로 부터 정규기저로 고속으로 변환하는 알고리즘을 소개하였으며 그 알고리즘을 이용한 H/W 구현결과를 제시하였다. 또한 Ⅳ장에서는 Ⅲ장의 방법으로 구현한 정규기저를 이용하여, 지금도 연구가 진행[12]되고 있는 표수가 2인 유한체 $GF(2^n)$위에서 두 원소의 곱셈과 역원을 구하는 효율적인 알고리즘에 따른 프로그램과 H/W 구현결과를 제시하였다.

감사의 글

본 논문은 2017년도 광주교육대학교 학술연구비 지원에 의한 것임

References

[1] G. Drolet, “A new representation of elements of finite fields $GF(2^m)$ yielding small complexity arithmetic circuit,” *IEEE Trans. Computers*, vol. 47, no. 9, 1998, pp.938-946.

[2] R. Lidl and H. Niederreiter, *Finite Fields*. Cambrideg: Cambridge University Press, 1984.

[3] J. Massey and J. Omura, “Computational method and apparatus for finite field arithmetic,” *United States Patent* vol. 4, Patent Number 4,587,627, May 1986, pp.587-627.

[4] C. Chiou, C. Chang, C. Lee, and T. Hou, “Concurrent Error Detection and Correction in Gaussian Normal Basis Multiplier over $GF(2^m)$,” *IEEE Trans. Computers*, vol. 58, no. 6, 2009, pp. 851-857.

[5] C. Kim and S. Chang, “Type II Optimal Normal Basis Multipliers in $GF(2^n)$,” *J. of the Korea Institute of Information Security and Cryptology*, vol. 25, no. 5, 2015, pp. 979-984.

[6] S. Kwon, S. Cho, and H. Kim, “Reachable table of non-linear cellular automata”, *J. of the Korea Institute of Electronic Communication Sciences*, vol. 10, no. 5, 2015, pp. 593-598.

[7] U. Choi, S. Cho, H. Kim, M. Kwon, and S. Kim,“Synthesis of 90/102(170)/150 linear CA using 90/150 linear CA,” *J. of the Korea Institute of Electronic Communication Sciences*, vol. 11, no. 9, 2016, pp. 885-891.

[8] S. Wolfram, *Mathematica. 4th Ed.* New York: Wolfram Champaign Research, Inc., 1999.

[9] M. Olofsson, "*VLSI Aspects on Inversion in Finite Fields,*" Ph.D's Thesis, Linkőpings University, 2002.
[10] E. Moreno, "*Acceleration of Finite Field Arithmetic with an Application to Reverse Engineering Genetic Network,*" Ph.D's Thesis, University of Puerto Rico Mayagüez Campus, 2008.
[11] J. von zur Gathen and J. Garhard, *Modern Computer Algebra, 3rd Ed.* Cambridge: Cambridge University Press, 2013.
[12] H. Kim, S. Cho, M. Kwon, and H. An, "A study on the cross sequences," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 7, no. 1, 2012, pp. 61-67.

저자 소개

김용태(Yong-Tae Kim)

1976년 : 공주사범대학 수학교육과(이학사)
1986년 : 고려대학교 대학원 수학과 (이학석사)
1991년 : 고려대학교대학원 수학과(이학박사)
2000년 : 서울대학교 대학원 수학교육과(교육학석사)
2008년 : 서울대학교 대학원 수학교육과(박사과정수료)
1992년 ~ 현재 : 광주교육대학교 수학교육과 교수
※ 주관심분야 : ECC, 정수론적 암호학, 공개키암호학