情報保護學會論文誌
第19卷 第6號, 2009. 12

반복 기약다항식 기반의 효율적인 비트-병렬 다항식 기저 곱셈기*

장 남 수,[1†] 김 창 한,[2‡] 홍 석 희[1]
[1]고려대학교, [2]세명대학교

Efficient Bit-Parallel Polynomial Basis Multiplier for Repeated Polynomials*

Nam Su Chang,[1†] Chang Han Kim,[2‡] Seokhie Hong[1]
[1]Korea University, [2]Semyung University

요 약

최근 Wu는 효율적인 비트-병렬 곱셈기를 위한 세 가지 종류의 이진체 제안하였다. 제안된 곱셈기는 오항 기약다항식을 사용하는 기존의 결과보다 효율적이다. 본 논문에서는 비트-병렬 곱셈에서 효율적인 이진체 위의 새로운 반복 다항식(Repeated Polynomial:RP)을 제안한다. 제안하는 RP를 case 1, case 2와 case 3 3가지로 구분할 때, 제안하는 RP를 위한 비트-병렬 곱셈기는 기존의 오항 기약다항식의 결과보다 효율적이다. 유한체의 차수가 1,000이하에서 EPS 또는 삼항 기약다항식이 없는 차수를 고려할 때, Wu의 단지 11개의 유한체만 존재한다. 그러나 제안하는 결과는 case 1에서 181, case 2에서 232 그리고 case 3에서 443개의 유한체가 존재한다.

ABSTRACT

Recently, Wu proposed a three small classes of finite fields F_{2^n} for low-complexity bit-parallel multipliers. The proposed multipliers have low-complexities compared with those based on the irreducible pentanomials. In this paper, we propose a new Repeated Polynomial(RP) for low-complexity bit-parallel multipliers over F_{2^m}. Also, three classes of Irreducible Repeated polynomials are considered which are denoted, respectively, by case 1, case 2 nad case3. The proposed RP bit-parallel multiplier has lower complexities than ones based on pentanomials. If we consider finite fields that have neither a ESP nor a trinomial as an irreducible polynomial when $n \leq 1{,}000$. Then, in Wu''s result, only 11 finite fields exist for three types of irreducible polynomials when $n \leq 1{,}000$. However, in our result, there are 181, 232, and 443 finite fields of case 1, 2 and 3, respectively.

Keywords: Finite field, irreducible polynomial, polynomial basis, bit-parallel multiplier

I. 서 론

부호이론, 암호 알고리즘 등에서 유한체 연산은 매우 중요한 연구 분야이다. 특히, 타원곡선 암호 알고리즘이 주목을 받으면서 유한체 연산은 많이 이슈화되었다. 이러한 유한체 연산 중 곱셈연산이 가장 중요한 비중을 차지하며, 곱셈 연산의 설계는 기저, 알고리즘과 기약다항식 등에 의하여 고려된다.

이진체에서의 연산은 대부분 덧셈(뺄셈) 연산과 곱셈 연산에 의하여 계산된다. 본 논문에서는 이진체에서의 곱셈 연산에 대하여 논하므로 F_2에서의 덧셈과 곱셈 연산은 비트 단위의 AND와 XOR 연산에 의하여 표현된다. 이는 하드웨어에서 2-입력 1-출력의 AND와 XOR 게이트(gate)이다. 또한 하드웨어에서 연산의 효율성은 시간 및 공간 복잡도에 의하여 정의되므로 본

접수일(2009년 4월 10일), 게재확정일(2009년 11월 1일)
* 이 연구에 참여한 연구자(의 일부)는 '2단계 BK21사업'의 지원비를 받았습니다.
† 주저자, ns-chang@korea.ac.kr
‡ 교신저자, chkim@semyung.ac.kr

논문에서는 공간 복잡도 계산을 위하여 AND와 XOR 게이트 수를 각각 #AND와 #XOR로 표기하고 시간 복잡도를 위하여 AND와 XOR 게이트 시간 지연(time delay)을 각각 T_A와 T_X로 표기한다.

이진체에서 주로 사용되는 효율적인 기약다항식은 순서별로 All One Polynomial(AOP) [1], [2], [3], Equally Spaced Polynomial(ESP) [1], [2], [3], 삼항 기약다항식(Trinomial) [1], [2], [4], 오항 기약다항식(Pentanomial) [2], [5], [6]이 있다. 정해진 유한체 차수에서 가장 효율적인 기약다항식은 위와 같은 순서로 찾는다. 그러나 암호 알고리즘에서 사용되는 유한체의 차수는 대부분 소수이므로 AOP와 ESP는 거의 사용하지 않으며 대부분의 표준에서는 삼항 기약다항식과 오항 기약다항식을 사용한다.

최근 [7]에서 AOP, ESP, 삼항 기약다항식이 존재하지 않는 차수(443개)에서 일부 사용할 수 있는 세 가지 종류의 기약다항식을 제안하였다. 제안된 새로운 기약다항식의 효율성은 공간복잡도의 경우 $\#AND = n^2$, $\#XOR = n^2+n-3$이고 시간복잡도의 경우 $\le T_A + (\lceil \log_2(n-1) \rceil + 3)T_X$이므로 오항 기약다항식보다 효율적이다. 그러나 443개의 차수 중 10개의 차수에만 존재한다.

본 논문에서는 새로운 반복 다항식(Repeated Polynomial:RP)를 제안한다. 제안하는 RP 기약다항식은 [7]에서 제안된 세 가지 기약다항식과 같이 차수에 따라 오항 기약다항식보다 효율적이기 위하여 세 가지 case로 구분한다. 이는 [7]의 결과를 포함한다. 차수가 $1 \le n \le 1{,}000$인 경우 ESP, 삼항 기약다항식이 존재하지 않는 차수(443개)에서 case 1의 경우 181(40.8%)개가 존재하며, case 2의 경우 232(52.4%)개, case 3의 경우 443 (100%)개 존재한다. 또한, 타원곡선 암호시스템의 표준(ANSI X9.62, IEEE P1363, SEC2)에서 제안되어 사용되는 이진체 중 오항 기약다항식을 사용하는 차수인 n=131, 163, 283, 571 모두에서 RP가 존재한다. 본 논문의 결과는 [7]의 결과를 포함한다.

본 논문의 구성은 다음과 같다. 2절에서는 기본적인 다항식 곱셈과 [7]에서 제안된 새로운 세가지 기약다항식을 소개한다. 3절에서는 제안하는 반복 기약다항식과 이를 이용한 비트-병렬 곱셈의 시간 및 공간 복잡도를 기술한다. 4절에서는 제안하는 방법의 결과와 기존 결과를 비교하고 결론을 내린다.

Ⅱ. F_{2^n} 위의 다항식기저 기반의 곱셈

유한체 F_{2^n}는 기약다항식 $f(x) = x^n + \sum_{i=1}^{n-1} f_i x^i + 1$에 의하여 생성되며, 이때 $f(x)$의 근을 α라 정의하자. ($f_i \in \{0,1\}$이다.) 그러면 F_{2^n}의 임의의 두 원소 $a(\alpha) = \sum_{i=0}^{n-1} a_i \alpha^i$, $b(\alpha) = \sum_{i=0}^{n-1} b_i \alpha^i$의 곱 $c(\alpha) = \sum_{i=0}^{n-1} c_i \alpha^i$는 다음과 같은 두 과정에 의하여 계산된다.(이때 $a_i, b_i \in F_2$이다.)

1. 다항식 곱셈:

$$s(\alpha) = a(\alpha)b(\alpha) = \sum_{i=0}^{2n-2} s_i \alpha^i, \quad s_i = \sum_{\substack{u+v=i \\ 0 \le u,v \le n-1}} a_u b_v.$$

2. $f(x)$에 의한 모듈러 감산 연산:

$$c(\alpha) \equiv s(\alpha) \bmod f(x) \equiv \sum_{i=0}^{n-1} c_i \alpha^i, \quad c_i \in F_2.$$

다항식 곱셈의 공간 복잡도는 다음과 같다. $0 \le i \le n-1$인 경우 s_i의 계산에서 $n(n+1)/2$번의 AND 연산과 $n(n-1)/2$번의 XOR 연산이 소요되며, $n \le i \le 2n-2$인 경우 $n(n-1)/2$번의 AND 연산과 $(n-1)(n-2)/2$번의 XOR 연산이 소요되므로 다항식 곱셈의 공간 복잡도는 n^2번의 AND와 $(n-1)^2$번의 XOR 연산이다. 그리고 $i = n-1$일 때 s_i의 계산에서 $a_u b_v$가 가장 많으므로 차수 $n-1$에서 최대 시간 지연(critical path delay)을 가지며 $T_A + \lceil \log_2 n \rceil T_X$이다.

모듈러 감산 연산의 경우

$$\begin{aligned} c(\alpha) &\equiv s(\alpha) \bmod f(x) \\ &= \sum_{i=0}^{n-1} s_i \alpha^i + \left(\sum_{i=n}^{2n-2} s_i \alpha^i \bmod f(x) \right) \end{aligned} \tag{1}$$

이고, 모듈러 감산 부분은 $\sum_{i=n}^{2n-2} s_i \alpha^i \bmod f(x)$이므로

$$\alpha^{n+i} \bmod f(x) = \sum_{j=0}^{n-1} t_{i,j} \alpha^{n-1-j}, \quad 0 \le i \le n-2 \tag{2}$$

이다. $(n-1) \times n$ 곱셈 행렬 $T = (t_{i,j})_{(n-1)\times n}$를

$$\begin{bmatrix} \alpha^n \\ \alpha^{n+1} \\ \vdots \\ \alpha^{2n-2} \end{bmatrix} = T \times \begin{bmatrix} \alpha^{n-1} \\ \alpha^{n-2} \\ \vdots \\ 1 \end{bmatrix}$$

라 정의하면 모듈러 감산은 곱셈 행렬 T에 의하여 계

산된다. 이는 (2)를 (1)에 대입하여

$$\begin{aligned} c(\alpha) &\equiv \sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=n}^{2n-2} s_i\alpha^i \bmod f(x)\right) \\ &= \sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{n-2} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-1-j} \bmod f(x)\right) \\ &= \sum_{i=0}^{n-1}\left(s_i + \sum_{j=0}^{n-2} t_{j,m-1-i}s_{n+j}\right)\alpha^i \end{aligned}$$

와 같이 계산한다. 곱셈 행렬 T의 효율성은 1의 개수에 의하여 정의되며 이는 기약다항식에 의존한다. F_{2^n}에서 효율적인 기약다항식은 self-reciprocal인 Equally Spaced Polynomial(ESP)와 0이 아닌 항의 개수가 작은 삼항 기약다항식 또는 오항 기약다항식이 있다.

2.1 ESP 기반의 곱셈

정의 1. F_2 위에서 $n=ms$를 만족하는 다항식

$$f(x) = x^{ms} + x^{(m-1)s} + x^{(m-2)s} + x^s + 1$$

를 n차 s-ESP(Equally Spaced Polynomial)라 한다.

s-ESP는 정의 1에서와 같이 상수항부터 n차항 까지 s의 배수 차수만 계수가 1인 다항식이다. s가 1인 경우를 AOP(All One Polynomial)라 하고 $s=n/2$인 경우를 EST(Equally Spaced Trinomial)라 한다. 예를 들어, 3-ESP인 $f(x)=x^6+x^3+1$의 경우 감산 행렬 T는 다음과 같다.

$$\begin{array}{l} x^6 = x^3+1 \\ x^7 = x^4+x \\ x^8 = x^5+x^2 \\ x^9 = 1 \\ x^{10} = x \end{array} \quad \Rightarrow \quad T = \begin{bmatrix} 0&0&1&0&0&1 \\ 0&1&0&0&1&0 \\ 1&0&0&1&0&0 \\ 0&0&0&0&0&1 \\ 0&0&0&0&1&0 \end{bmatrix}$$

예제에서 알 수 있듯이 α^n 부터 α^{n+s-1}까지는 감산 연산하면 m개의 항을 가진다. 따라서 위의 예제의 행렬 T의 s개의 행(0행~$s-1$행)은 각각 m개의 1을 가지며 겹치지 않고 행마다 1이 하나만 있다. 따라서 $1T_X$ 시간에 n번의 XOR 연산이 소요된다. 그리고 α^{n+s} 부터 α^{2n-2}항까지는

$$\begin{aligned} & s_{n+s}\alpha^{n+s} + \cdots + s_{2n-3}\alpha^{2n-3} + s_{2n-2}\alpha^{2n-2} \\ = \quad & s_{n+s} + \cdots + s_{2n-3}\alpha^{n-s-3} + s_{2n-2}\alpha^{n-s-2} \end{aligned}$$

이므로 $n-s-1$번의 XOR 연산이 소요되며 (s_0+s_n+s), $\cdots,(s_{n-s-2}+s_{2n-2})$는 $\lceil \log_2(n-s-1) \rceil T_X$ 시간이 소요되므로

$$\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=n+s}^{2n-2} s_i\alpha^i \bmod f(x)\right)$$

는 $\lceil \log_2(n) \rceil T_X$ 시간이 소요된다. 따라서 s-ESP 기반의 곱셈기 복잡도는 다음과 같다[2].

$$\begin{aligned} &\#AND = n^2, \\ &\#XOR = n^2 - s, \\ &T_{TOT} = 1T_A + (1 + \lceil \log_2 n \rceil)T_X. \end{aligned}$$

2.2 [7]에서 제안된 기약다항식 기반의 곱셈

2008년 [7]에서 Wu는 기존의 오항 기약다항식 보다 효율적인 세 가지 타입의 기약다항식을 제안하였고 이를 이용한 비트-병렬 곱셈기를 제시하였다. 본 소절에서는 [7]의 기약다항식과 이를 이용한 곱셈기를 간략히 소개한다.

2.2.1 Type-I 기약다항식

이진체의 차수 n이 $n=3h+2$이고 다항식이 다음과 같은 성질을 가지고 기약일 때 Type-I 기약다항식이라 한다.

$$\begin{aligned} f(x) &= \left(\sum_{i=0}^{h}(x^{3i+1}+x^{3i+2})\right)+1 \\ &= \sum_{i=0}^{n-1} f_i x^i \end{aligned}$$

이때 $f_i = \begin{cases} 0 & , \text{만약 } i \neq 0 \text{이고 } i = 0 \bmod 3 \text{이면} \\ 1 & , \text{그렇지 않으면} \end{cases}$

이다. 따라서 x^{n+i}는

$$\begin{aligned} x^n &= x^{3h+1} + \sum_{i=0}^{h-1}(x^{3i+1}+x^{3i+2})+1, \\ x^{n+1} &= \sum_{i=1}^{h}(x^{3i}+x^{3i+1})+1, \\ x^{n+k} &= x^k + x^{k-2} \end{aligned}$$

와 같으며, 이때 k는 $2 \le k \le n-2$이다. 따라서 Type-I 기양다항식에 의하여 $c(\alpha)$는 다음과 같이 계산된다.

$$
\begin{aligned}
c(\alpha) &\equiv \left(\sum_{i=0}^{n-1} s_i\alpha^i\right)+\left(\sum_{i=n}^{2n-2} s_i\alpha^i \bmod f(x)\right)\\
&= \left(\sum_{i=0}^{n-1} s_i\alpha^i\right) && \Rightarrow(3.1)\\
&+ s_m \cdot \left(x^{3h+1}+\sum_{i=0}^{h-1}\left(x^{3i+1}+x^{3i+2}\right)+1\right) && \Rightarrow(3.2)\\
&+ s_{m+1} \cdot \left(\sum_{i=1}^{h}\left(x^{3i}+x^{3i+1}\right)+1\right) && \Rightarrow(3.3)\\
&+ \left(\sum_{i=2}^{m-2} s_{m+i}x^{i-2}\right) && \Rightarrow(3.4)\\
&+ \left(\sum_{i=2}^{m-2} s_{m+i}x^{i}\right) && \Rightarrow(3.5)
\end{aligned}
\tag{3}
$$

다항식 곱셈에서 n^2번의 AND 연산과 $(n-1)^2$번의 XOR 연산을 수행하고, 식 (3)에 의하여 모듈러 감산 연산을 수행한다. 식 (3.2)와 (3.3)의 합은 s_m+s_{m+1} 1번의 XOR 연산으로 계산되며 식 (3.4)와 (3.5)의 합은 각각 차수가 $0\sim m-4$, $2\sim m-2$이므로 $m-5$번의 XOR 연산으로 계산되고, 이와 같은 두 번의 덧셈 결과와 (3.1)의 합에서 2m번의 XOR 연산이 소요되므로 공간복잡도는

$$\#\mathrm{AND}=\mathrm{n}^2,$$
$$\#\mathrm{XOR}=(\mathrm{n}-1)^2+1+(\mathrm{n}-5)+2\mathrm{n}=\mathrm{n}^2+\mathrm{n}-3$$

이다. 시간복잡도의 경우 다항식 곱셈 계산에서 $1\mathrm{T_A}+\lceil\log_2 n\rceil \mathrm{T_X}$의 시간 지연이 소요되며 (3.2)+(3.3)과 (3.4)+(3.5)에서 $1\mathrm{T_X}$, 두 번의 덧셈 결과와 (3.1)의 합에서 $2\mathrm{T_X}$ 소요되므로

$$\mathrm{T_{TOT}}=1\mathrm{T_A}+(\lceil\log_2 n\rceil+3)\mathrm{T_X}$$

이다.

2.2.2 Type-II 기약다항식

이진체의 차수 n이 $n=3h$이고 다항식이 다음과 같은 성질을 가지고 기약일 때 Type-II 기약다항식이라 한다.

$$
\begin{aligned}
f(x) &= x^{3h}+\sum_{i=0}^{h-1}\left(x^{3i+2}+x^{3i}\right)\\
&= \sum_{i=0}^{n-1} f_i x^i
\end{aligned}
$$

이때, $f_i=\begin{cases}0 & ,\text{만약 } i=1 \bmod 3\text{이면}\\ 1 & ,\text{그렇지 않으면}\end{cases}$

이다. 따라서 x^{n+i}는

$$
\begin{aligned}
x^n &= \sum_{i=0}^{h-1}\left(x^{3i+2}+x^{3i}\right),\\
x^{n+1} &= \sum_{i=1}^{h-1}\left(x^{3i+2}+x^{3i+1}\right)+1,\\
x^{n+k} &= x^{k-1}+x^{k-2}
\end{aligned}
$$

와 같으며, 이때 k는 $2\le k\le n-2$이다. 식의 전개와 복잡도 계산은 Type-I과 유사하므로 생략하고 복잡도만 정리하면 다음과 같다.

$$\#\mathrm{AND}=\mathrm{n}^2,$$
$$\#\mathrm{XOR}=\mathrm{n}^2+\mathrm{n}-3,$$
$$\mathrm{T_{TOT}}=1\mathrm{T_A}+(\lceil\log_2 n\rceil+3)\mathrm{T_X}.$$

2.2.3 Type-III 기약다항식

이진체의 차수 n이 $n=3h$이고 다항식이 다음과 같은 성질을 가지고 기약일 때 Type-III 기약다항식이라 한다.

$$
\begin{aligned}
f(x) &= x^{3h}+\sum_{i=1}^{h-1}\left(x^{3i+2}+x^{3i}\right)+x+1\\
&= \sum_{i=0}^{n-1} f_i x^i
\end{aligned}
$$

이때, $f_i=\begin{cases}0 & ,\text{만약 } i=2 \text{ 또는 } 3j+1\text{이면}, j=1,\ldots,h-1\\ 1 & ,\text{그렇지 않으면}\end{cases}$

이다. 따라서 의하여 x^{n+i}는

$$
\begin{aligned}
x^n &= \sum_{i=0}^{h-1}\left(x^{3i+2}+x^{3i}\right)+x+1,\\
x^{n+1} &= \sum_{i=1}^{h-1}\left(x^{3i+2}+x^{3i+1}\right)+x^3+x^2+1,\\
x^{n+k} &= x^{k+2}+x^{k-2},\\
x^{2n-2} &= x^{3h-2}+x^{3h-3}+x^{3h-6}+\sum_{i=1}^{h-3}\left(x^{3i+2}+x^{3i}\right)+x+1
\end{aligned}
$$

와 같으며, 이때 k는 $2\le k\le n-3$이다. 식의 전개와 복잡도 계산은 Type-I과 유사하므로 생략하고 복잡도만 정리하면 다음과 같다.

$$\#\mathrm{AND}=\mathrm{n}^2,$$
$$\#\mathrm{XOR}=\mathrm{n}^2+\mathrm{n}-3,$$
$$\mathrm{T_{TOT}}=1\mathrm{T_A}+(\lceil\log_2 n\rceil+3)\mathrm{T_X}.$$

III. 제안하는 반복 기약다항식과 이를 이용한 비트-병렬 곱셈

3.1 반복 다항식

이진체에서 효율적인 기약다항식의 순서는 ESP, 삼항 기약다항식, 오항 기약다항식 등으로 구분된다. 따라서 n이 정해지면 효율성의 순으로 기약다항식을 찾는다. 그러나 $n\le 1{,}000$를 고려하여 조사하면 77개의 n에서만 ESP가 존재하며 543개의 n에서만 삼항 기약다항식이 존재한다. 그리고 이 둘을 합하여도 557개 n

에서만 ESP 또는 삼항 기약다항식 존재하므로 나머지 n에서는 오항 기약다항식을 사용하여야 한다. 본 절에서는 오항 기약다항식보다 효율적인 새로운 타입의 기약다항식을 제안하고 이를 이용한 비트 병렬 곱셈기와 복잡도를 제시한다. 제안하는 새로운 타입의 기약다항식은 [7]의 결과를 포함한다.

정의 2. F_2 위에서 임의의 두 다항식 $p(x)$와 $l(x)$에 대하여

$$f(x)=\sum_{i=0} p(x)x^{i\cdot s+v}+l(x),\ \deg(l(x))\le v,\ \deg(p(x))<s$$

를 만족하면 $f(x)$를 반복 다항식(RP:Repeated Polynomial)이라 한다.

정의 2의 반복 다항식은 s-ESP를 포함하며 임의의 다항식 $h(x)$가 주어졌을 때 $x^{n+r}=h(x)\bmod f(x)$을 만족하는 $f(x)$가 RP임을 보이기 위하여 다음 정리를 이용한다.

정리 1. F_2 위에서 임의의 다항식 $q(x)=x^r+1+\sum_{i=1}^{r-1}q_i x^i (q_i\in F_2)$에 대하여 $x^s=q(x)p(x)+1$을 만족하는 최소의 양의 정수 s와 다항식 $p(x)$가 존재할 때, 임의의 양의 정수 m에 대하여 x^m이 $x^m=q(x)p_m(x)+1$을 만족하면 양의 정수 i에 대하여 $m=i\cdot s$이다.

증명. $x^m=q(x)p_m(x)+1$을 만족하는 m의 순서집합(ordered set)을 $\Gamma=\{m_i\in N \mid x^{m_i}=q(x)p_{m_i}(x)+1$, 만약 $i>j$이면 $m_i>m_j\}$라 하자. 그러면 가정에 의하여 순서집합 Γ의 최소값은 $m_0=s$이다. 다음으로 s보다 큰 순서집합 Γ의 최소값을 m_1라 하면 $x^{m_1}=q(x)p_{m_1}(x)+1$이므로 $F_2[x]$에서

$$\begin{aligned} x^{m_1}-x^s &= (q(x)p_{m_1}(x)+1)-(q(x)p(x)+1)\\ \Rightarrow x^s(x^{m_1-s}+1) &= q(x)[p_{m_1}(x)+p(x)] \end{aligned}$$

이다. 이때 가정에 의하여 $q(x)|(x^{m_1-s}+1)$이므로 m_1-s는 Γ의 원소이다. m_1은 s보다 큰 순서집합 Γ의 최소값이고 $m_1>m_1-s=s$이므로 $m_1=2s$이고 이를 일반화하면 $m_i=i\cdot s$이다. 즉, 순서집합 Γ의 원소는 양의 정수 i에 대하여 $m=i\cdot s$이다. □

정리 2. 정리 1의 다항식 $q(x)$와 임의의 다항식 $h(x)\in F_2[x]$에 대하여 $x^{n_0+r}=q(x)f_{n_0}(x)+h(x)$을 만족하는 최소 양의 정수 n_0와 다항식 $f_{n_0}(x)$ $(n_0=\deg(f_{n_0}(x)))$가 존재하면, 임의의 n에 대하여 $x^{n+r}=q(x)f(x)+h(x)$를 만족하는 $f(x)$는 $f_{n_0}(x)$ 또는 $l(x)=f_{n_0}(x)$인 반복 다항식이다.

증명. $x^{n_i+r}=q(x)f_{n_i}(x)+h(x)$를 만족하는 n_i의 순서집합(ordered set)을 $\Gamma=\{n_i\in N \mid x^{n_i+r}=q(x)\cdot f_{n_i}(x)+h(x)$, 만약 $i_j>i_g$이면 $m_{i_j}>m_{u_g}\}$라 하자. 가정에 의하여 $x^{n_0+r}=q(x)f_{n_0}(x)+h(x)$을 만족하는 최소 양의 정수 n_0와 다항식 $f_{n_0}(x)$가 존재하고, Γ의 원소인 n_0보다 큰 최소값을 n_1이라 하면 $x^{n_1+r}=q(x)f_{n_1}(x)+h(x)$이다. 그러면

$$x^{n_1+r}+x^{n_0+r}=x^{n_0+r}(x^{n_1-n_0}+1)=q(x)(f_{n_1}(x)-f_{n_0}(x))$$

이고 정리 1에서 $q(x)=x^r+1+\sum_{i=1}^{r-1}q_0x^i$이므로 $q(x)\mid(x^{n_1-n_0}+1)$이다. 따라서 정리 1에 의하여 n_1-n_0는 $i\cdot s$이며 n_1이 n_0보다 큰 최소값이므로 $n_1-n_0=s$이다. 실제로

$$\begin{aligned} x^{n_0+r} &= q(x)f_{n_0}(x)+h(x)\\ (x^{n_0+r})x^s &= (q(x)f_{n_0}(x)+h(x))\cdot x^s\\ &= (q(x)f_{n_0}(x)+h(x))\cdot(q(x)p(x)+1)\\ &= q(x)\{(q(x)f_{n_0}(x)+h(x))p(x)+f_{n_0}(x)\}+h(x)\\ &= q(x)\{x^{n_0+r}p(x)+f_{n_0}(x)\}+h(x)\\ &= q(x)f_1(x)+h(x) \end{aligned}$$

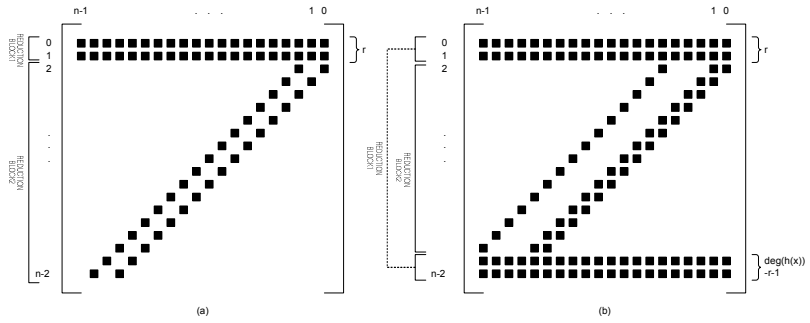
이고, 순서집합 Γ의 원소에 대하여 일반화하면 $n_{i+1}-n_i=s$이다. 따라서 $x^{n_i+r}=x^{i\cdot s+n_0+r}=q(x)\cdot f_{n_i}(x)+h(x)$이다. 다음으로 $f_{n_{i+1}}(x)=x^{i\cdot s+n_0+r}\cdot p(x)+f_{n_i}(x)$이므로 $f_{n_0}(x)$와 n_0+r을 각각 $l(x)=f_{n_0}(x)$, $v=n_0+r$라 하면

$$f_{n_{i+1}}(x)=\left(\sum_{j=0}^{i}p(x)x^{j\cdot s+v}\right)+l(x)$$

이다. 따라서 정리 1의 다항식 $q(x)$와 임의의 다항식 $h(x)\in F_2[x]$에 대하여, $x^{n+r}=q(x)f(x)+h(x)$를 만족하는 임의의 n이 존재할 때 $f(x)$는 $f_{n_0}(x)$ 또는 $l(x)=f_{n_0}(x)$인 반복 다항식이다. □

3.2 RP 비트-병렬 곱셈

정리 2로부터 임의의 $q(x)$, $h(x)$에 대하여



[그림 1] 반복 기약다항식 $f(x)$의 감산행렬 T: a) $\deg(h(x)) < r-2$ (b) $\deg(h(x)) \geq r-2$

$x^{n+r} = q(x)f(x)+h(x) \Rightarrow x^{n+r} = h(x) \mod f(x)$를 만족하는 최소 차수의 $f_{n_0}(x)$가 주어졌을 때, $x^{n+r} = q(x)f(x)+h(x)$를 만족하는 나머지 $f(x)$는 모두 $f_{n_0}(x)$와 $p(x)$를 이용한 반복 다항식임을 알 수 있다. 기약다항식 $f(x)$에 의하여 정의된 F_{2^n}의 임의의 두 원소를 $a(\alpha)$, $b(\alpha)$라 하고 $f(x) \in F_2[x]$는 정리 2를 만족한다면 $x^{n+r} \equiv h(x) \bmod f(x)$이다. 따라서 식 (2)는

$$\alpha^{n+i} \bmod f(x) = \begin{cases} \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-1-j}, & 0 \leq i \leq r-1 \\ h(x)\alpha^{i-r}, & r \leq i \leq n-2, \\ & \deg(h(x))+i-r \geq n \\ \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-1-j}, & r \leq i \leq n-2, \\ & \deg(h(x))+i-r < n \end{cases}$$

이다. 이때, 만약 $\deg(h(x)\alpha^{i-r}) \geq n$이면, $h(x)\alpha^{i-r}$는 모듈러 감산되어야 한다. 따라서 식 (1) $c(\alpha) = a(\alpha) \cdot b(\alpha)$은 $f(x)$에 의하여 다음과 같다.

$$\begin{aligned} c(\alpha) &\equiv \sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=n}^{2n-2} s_i\alpha^i \bmod f(x)\right) \\ &= \sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=n}^{n+r-1} s_i\alpha^i + \sum_{i=n+r}^{2n-2} s_i\alpha^i \bmod f(x)\right) \\ &= \sum_{i=0}^{n-1} s_i\alpha^i + \\ &\quad \left(\sum_{i=n}^{n+r-1} s_i\alpha^i + \left(\sum_{i=n+r}^{2n-2} s_i\alpha^{i-n-r}\right)h(x) \bmod f(x)\right) \\ &= \sum_{i=0}^{n-1} s_i\alpha^i \\ &\quad + \left(\sum_{i=0}^{r-1} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-1-j}\right) \quad \Rightarrow (4.1) \\ &\quad + \left\{\left(\sum_{i=n+r}^{2n-2} s_i\alpha^{i-n-r}\right)h(x) \bmod f(x)\right\} \Rightarrow (4.2) \end{aligned} \tag{4}$$

이때, 만약 $\deg(h(x)) \geq r+2$이면 식 (4.2)은 다시 모듈러 감산되어야 한다. 감산행렬 T에서 (4.2)와 같이 $h(x)$로 표현되는 행을 REDUCTION BLOCK2라 하고 나머지 행을 REDUCTION BLOCK1라 정의하면 RP의 감산행렬 T는 [그림 1]과 같다. $c(\alpha)$의 계산에서 REDUCTION BLOCK1의 효율성은 $h(x)$의 0이 아닌 비트수에 의존하고 REDUCTION BLOCK2는 $h(x)$와 r에 의존한다.

제안하는 반복 기약다항식을 이용하여 식 (4)의 간단한 예를 들면 다음과 같다. $q(x), h(x) \in F_2[x]$를 각각 $q(x) = x^3+x^2+x+1$, $h(x) = x^5+x^2+1$이라 하면, $x^{4+3} = q(x)(x^4+x^3+x^2+x+1)+h(x)$이므로 정리 2에 의하여 $f_{n_0}(x) = x^4+x^3+x^2+x+1$이고 $x^n = q(x)f(x)+h(x)$을 만족하는 반복 다항식은 $f(x) = \left(\sum_{i=0}(x+1)x^{i \cdot 4+7}\right) + x^4+x^3+x^2+x+1$이다. 이때 $i=1$인 $f(x) = x^8+x^7+x^4+x^3+x^2+x+1$은 기약다항식이고 이를 이용하여 F_{2^8}에서 $c(\alpha) = a(\alpha) \cdot b(\alpha)$를 계산하면

$$\begin{aligned} c(\alpha) &\equiv \sum_{i=0}^{7} s_i\alpha^i + \left(\sum_{i=8}^{14} s_i\alpha^i \bmod f(x)\right) \\ &= \sum_{i=0}^{7} s_i\alpha^i + \left(\sum_{i=0}^{2} s_{n+i}\sum_{j=0}^{7} t_{i,j}\alpha^{n-1-j}\right) \\ &\quad + \left(\sum_{i=11}^{14} s_i\alpha^{i-n-r}\right)h(x) \bmod f(x) \\ &= \sum_{i=0}^{7} s_i\alpha^i + \left(\sum_{i=11}^{13} s_i\alpha^{i-11}\right)h(x) \\ &\quad + \left(\sum_{i=0}^{2} s_{8+i}\sum_{j=0}^{7} t_{i,j}\alpha^{7-j}\right) + \left(s_{14}\sum_{j=0}^{n-1} t_{14,j}\alpha^{7-j}\right) \end{aligned}$$

이다. 그리고 이때 7×8 곱셈 행렬 $T = (t_{i,j})_{7 \times 8}$는 다음과 같다.

$$T = \begin{bmatrix} 10011111 \\ 10100001 \\ 11011101 \\ 00100101 \\ 01001010 \\ 10010100 \\ 10110111 \end{bmatrix}$$

3.3 효율적인 RP와 그 비트-병렬 곱셈기 복잡도

오항 기약다항식 $f(x)=x^n+x^{k_3}+x^{k_2}+x^{k_1}+1$에 대한 [2]의 결과를 정리하면 [표 1]과 같다. [표 1]에서 알 수 있듯이 AND 게이트 수는 모두 n^2으로 같으며 $k_1=1$, $k_3-k_2=1$인 경우를 제외하면 XOR 게이트와 시간 지연은 3가지로 구분된다. 따라서 제안하는 RP가 오항 기약다항식보다 차수에 따라 시간 또는 공간복잡도 측면에서 효율적이기 위해서 RP에 요구되는 XOR 게이트수와 시간복잡도 영역을 대략적으로 구분하면 다음과 같다.

- Case 1 : $\#\mathrm{XOR} \approx \mathrm{n}^2+\mathrm{n}$, $\mathrm{Time\ Delay} \approx \mathrm{T_A}+(\lceil \log_2(n) \rceil+3)\mathrm{T_X}$
- Case 2 : $\#\mathrm{XOR} \approx \mathrm{n}^2+\mathrm{n}$, $\mathrm{Time\ Delay} \approx \mathrm{T_A}+(\lceil \log_2(n) \rceil+4)\mathrm{T_X}$
- Case 3 : $\#\mathrm{XOR} \approx \mathrm{n}^2+2\mathrm{n}$, $\mathrm{Time\ Delay} \approx \mathrm{T_A}+(\lceil \log_2(n) \rceil+3)\mathrm{T_X}$

이때, [7]에서 제시한 세 가지 타입의 RP는 Case 1에 속한다. (이때, $2r+2<k_1<(n+3r+1)/2$)

본 소절에서는 제안하는 RP를 이용한 비트-병렬 곱셈기의 시간 및 공간 복잡도를 제시한다. $h(x)$의 차수와 0이 아닌 항(non-zero term)의 개수를 $\mathbf{H}(\mathrm{h(x)})$하 하면, 식 (4)에서와 같이 RP 기반의 곱셈은 다항식 $q(x)$의 차수 r, $\mathbf{H}(\mathrm{h(x)})$에 의존한다. 따라서 효율적인 r과 $h(x)$에 대하여 고려하여야 한다.

우선 임의의 r에 대하여 $h(x)=1$인 경우를 살펴보자. 이 경우 RP $f(x)=\sum_{i=0} p(x)x^{i\cdot s+v}+l(x)$는 정리 2의 증명에서 $f_{n_0}(x)$는 정리 1의 $p(x)$가 된다. 따라서 $p(x)=l(x)$이고 항상 $f(x)=p(x)\cdot\left(\sum_{i=0} x^{i\cdot s}+1\right)$이다. 결국 $p(x)\neq 1$인 경우 $f(x)$는 기약다항식이 될 수 없으며, $p(x)=1$인 경우 $f(x)$는 ESP가 된다. 따라서 이를 제외하면 임의의 r에 대하여 최소 헤밍웨이트인 $\mathbf{H}(\mathrm{h(x)})=2$인 경우가 가장 효율적이다. 다음으로 $\mathbf{H}(\mathrm{h(x)})=2$인 경우 r을 살펴보자. $r=1$인 경우 정리 2에서 $q(x)$는 가정에 의하여 $q(x)=x+1$이다. 따라서 $x^{n+r}+h(x)=q(x)f(x)$에서 $q(x)f(x)$의 해밍 웨이트는 항상 짝수이다. 그러나 $x^{n+r}+h(x)$의 헤밍 웨이트는 항상 홀수이므로 $r=1$와 $\mathbf{H}(\mathrm{h(x)})=2$를 만족하는 $f(x)$는 존재하지 않는다. 따라서 $\mathbf{H}(\mathrm{h(x)})=2$인 경우 $r\geq 2$인 경우만 RP $f(x)$가 존재한다. 마지막으로 r과 $\mathbf{H}(\mathrm{h(x)})$의 최대값을 살펴보자. $r\geq 9$인 경우는 식 (4.1)의 계산에서만 $3\mathrm{T_X}$가 소요되어 위에서 정의한 case를 만족할 수 없다. 또한 $\mathbf{H}(\mathrm{h(x)})\geq 4$인 경우는 식 (4.2)가 다시 모듈러 감산되지 않는다 하여도 $\#\mathrm{XOR}\geq \mathrm{n}^2+3\mathrm{n}$이 되어 역시 위에서 정의한 case를 만족할 수 없다. 따라서 이를 모두 정리하면 case1~case3을 만족하는 조건은 다음과 같다.

$r=1$이고 $\mathbf{H}(\mathrm{h(x)})=3$,
$2\leq r\leq 8$이고 $2\leq \mathbf{H}(\mathrm{h(x)})\leq 3$.

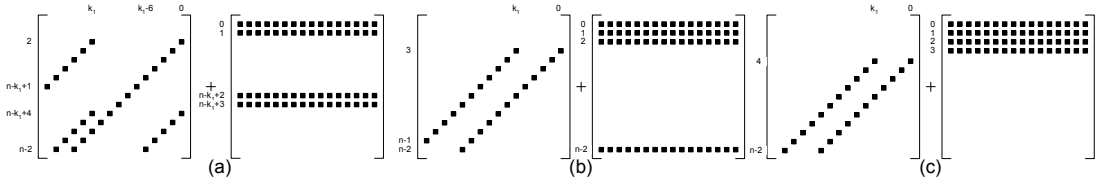
따라서 본 소절에서는 r과 $\mathbf{H}(\mathrm{h(x)})$에 따른 RP와 그 비트-병렬 곱셈기의 효율성을 기술한다. 이때, $\sum_{i=u}^{v}()$에서 $u>v$ 또는 $u<0$인 경우 $\sum_{i=u}^{v}()=0$이다.

3.3.1 case 1인 경우

우선 $r=2$와 $\mathbf{H}(\mathrm{h(x)})=2$라 하면 $h(x)=x^{k_1}+1$이다.

[표 1] [2]의 오항 기약다항식 $f(x)$ 기반의 비트-병렬 곱셈기 복잡도 비교

Special Case		Space Complexity		Time Complexity
		#AND	#XOR	Time Delay
$f(x)=x^n+x^{k_3}+x^{k_2}+x^{k_1}+1,\ 1<k_1<k_2<k_3\leq n/2$				
$k_1=1$	$k_3-k_2\neq 1$	n^2	n^2+2n-3	$\mathrm{T_A}+(3+\lceil \log_2(n-1) \rceil)\mathrm{T_X}$
	$k_3-k_2=1$	n^2	n^2+n	$\mathrm{T_A}+(3+\lceil \log_2(n-1) \rceil)\mathrm{T_X}$
$k_1>1$	$k_3-k_2\neq k_1$	n^2	n^2+2n-3	$\mathrm{T_A}+(4+\lceil \log_2(n-1) \rceil)\mathrm{T_X}$
	$k_3-k_2=k_1$	n^2	n^2+n+k_1-2	$\mathrm{T_A}+(4+\lceil \log_2(n-1) \rceil)\mathrm{T_X}$
$f(x)=x^n+x^{n-s}+x^{n-2s}+x^{n-3s}+1$				
$k_1>1$	$\frac{n-1}{8}\leq s\leq \frac{m-1}{3}$	n^2	$\leq n^2+n$	$\mathrm{T_A}+(4+\lceil \log_2(n-1) \rceil)\mathrm{T_X}$



[그림 2] 반복 기약다항식 $f(x)$의 감산행렬 T: (a) $r=2$ (b) $r=3$ (c) $r=4$

[그림 2(a)]와 같이 REDUCTION BLOC K1의 임의의 열에서 1의 개수가 4이하가 되어야 $3T_X$에 계산 가능하므로 k_1의 범위는 $2k_1 \le n+5$이다. 따라서 최대 $k_1 = \lfloor (n+5)/2 \rfloor$ 일 때, 식 (4)는 다음과 같다.

$$
\begin{aligned}
&\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{1} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \\
&+\left\{\left(\sum_{i=n+2}^{2n-2} s_i\alpha^{-n+i-2}\right)(x^{k_1}+1) \bmod f(x)\right\} \\
&=\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{1} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \\
&\quad+\left\{\left(\sum_{i=0}^{n-4} s_{n+i+2}\alpha^i\right)+\left(\sum_{i=k_1}^{n-1} s_{n-k_1+i+2}\alpha^i\right.\right. \\
&\quad+\sum_{i=0}^{1} s_{2n-k_1+i+2}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1} \\
&\quad\left.\left.+\sum_{i=n+2}^{n+k_1-4} s_{n-k_1+i+2}\alpha^i\right) \bmod f(x)\right\} \qquad (5) \\
&=\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=k_1-5}^{n-4} s_{n+i+2}\alpha^i\right) \qquad \Rightarrow(5.1) \\
&\quad+\left(\sum_{i=0}^{1}(s_{n+i}+s_{2n-k_1+i+2})\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \Rightarrow(5.2) \\
&\quad+\left\{(1+\alpha^{k_1})\left(\sum_{i=0}^{k_1-6}(s_{n+i+2}+s_{2n-k_1+i+4})\alpha^i\right)\right. \\
&\quad\left.+\left(\sum_{i=2k_1-5}^{n-1} s_{n-k_1+i+2}\alpha^i\right)\right\} \qquad \Rightarrow(5.3)
\end{aligned}
$$

따라서 모듈러 감산의 XOR 게이트 수는 (5.1)에서 $n-k_1+2$개, (5.2)에서 2개, (5.3)에서 k_1-3개, (5.1)+(5.3)에서 $n-3$개, ((5.1)+(5.3))+(5.2)에서 최대 n개 이므로 전체 $3n-2$이다. 시간 복잡도의 경우 (5.1)+(5.3) 계산에서 $2T_X$가 필요하고, (5.2)의 $(s_{n+i}+s_{2n-k_1+i+2})$에서 $1T_X$와 $t_{0,j}+t_{1,j}$에서 $1T_X$가소요되므로 $2T_X$가 소요된다. 따라서 모듈러 감산에서 최대 $3n-5$ XOR 게이트와 $3T_X$가 추가적으로 필요하다.

다음으로 $r=3$인 경우를 고려하자. [그림 2(a)]와 같이 REDUCTION BLOCK1의 임의의 열에서 1의 개수가 4이하가 되어야 $3T_X$에 계산 가능하므로 k_1의 범위는 $k_1<6$이다. 따라서 $\delta=\begin{cases}1, k_1=5\\ 0\ , k_1<5\end{cases}$ 라 하고 최대 $k_1=5$을 가정하면 식 (4)는 다음과 같다.

$$
\begin{aligned}
&\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{2} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \\
&+\left\{\left(\sum_{i=n+3}^{2n-2} s_i\alpha^{-n+i-2}\right)(x^5+1) \bmod f(x)\right\} \\
&=(s_n+\delta\cdot s_{2n-2})\sum_{j=0}^{n-1} t_{0,j}\alpha^{n-j-1} \\
&\quad+\left(\sum_{i=1}^{2} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \qquad \Rightarrow(6.1) \\
&\quad+\left\{\sum_{i=0}^{n-1} s_i\alpha^i+\left(\sum_{i=0}^{n-5} s_{n+i+3}\alpha^i\right)\right. \qquad (6) \\
&\quad+\delta\cdot\left(\sum_{i=k_1}^{n+k_1-4} s_{n-k_1+i+3}\alpha^i\right) \\
&\quad\left.+(1-\delta)\cdot\left(\sum_{i=k_1}^{n+k_1-5} s_{n-k_1+i+3}\alpha^i\right)\right\}\Rightarrow(6.2)
\end{aligned}
$$

식 (10.1)과 같이 REDUCTION BLOCK1 $\left(\sum_{i=0}^{2} s_{n+i}\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right)$의 XOR 게이트수를 ε이라 하면

$$0<\varepsilon<2^r-r \qquad (7)$$

이다. 따라서 (6.1)은 최대 $\varepsilon=4(\delta=1$인 경우는 5) XOR 게이트로 계산되며 (6.2)에서는 $n-4+(n-3)$ ($\delta=1$인 경우$n-4$) XOR 게이트로 계산되고, (6.1)+(6.2) 는 n XOR 게이트로 계산되므로 모듈러 감산에서 최대 3n-3 XOR 게이트가 필요하다. 시간복잡도의 경우 (6.1)와 (6.2)는 $2T_X$에 계산되므로 $3T_X$의 추가 XOR delay가 증가한다. $r=4$인 경우 또한 유사하게 전개되므로 결과만 정리하면

$r=3$, $k_1<6$인 경우 #AND $=n^2$,
#XOR $=n^2+n+\varepsilon-10$,
Delay $=1T_A+\lceil \log_{2(n)}+3 \rceil T_X$

이다. $r>4$인 경우 REDUCTION BLOCK1의 시간지

연만 $3T_X$가 되어 case 1에 해당하지 않는다. 따라서 다음 RP만 case 1을 만족한다.

1) $r=2$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2, \mathrm{k}_1<(\mathrm{n}+6)/2$,
2) $r=3$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2, \mathrm{k}_1<6$,
3) $r=4$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2, \mathrm{k}_1<6$.

3.3.2 case 2인 경우

$r=2$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2$라 하면, $h(x)=x^{k_1}+1$이다. $\lfloor (n+5)/2 \rfloor < k_1 \le \lfloor (2n+7)/3 \rfloor$을 가정하자. 그러면 최대 k_1에서 식 (4)는 다음과 같다.

$$
\begin{aligned}
&\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{1} s_{n+i} \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \\
&\quad + \left\{\left(\sum_{i=n+2}^{2n-2} s_i\alpha^{-n+i-2}\right)(x^{k_1}+1) \bmod f(x)\right\} \\
&= \sum_{i=0}^{1}(s_{n+i} + \underbrace{(s_{2n-k_1+i+2}+s_{3n-2k_1+i+4})}_{(a)}) \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1} \quad \Rightarrow(8.1) \\
&\quad + \left(\sum_{i=0}^{n-1} s_i\alpha^i + \sum_{i=k_1-5}^{n-4} s_{n+i+2}\alpha^i\right) \quad \Rightarrow(8.2) \\
&\quad + \left(\sum_{i=0}^{-n+2k_1-8} \underbrace{(s_{n+i+2}+\underbrace{(s_{2n-k_1+i+4}+s_{3n-2k_1+i+6})}_{(b)})\alpha^i}_{(c)}\right) \\
&\quad + \left(\sum_{i=-n+2k_1-7}^{k_1-6} \underbrace{(s_{n+i+2}+s_{2n-k_1+i+4})\alpha^i}_{(d)}\right) \quad \Rightarrow(8.3) \\
&\quad + \left(\sum_{i=k_1}^{-n+3k_1-8} \underbrace{(s_{n-k_1+i+2}+\underbrace{(s_{2n-2k_1+i+4}+s_{3n-3k_1+i+6})}_{(f)})\alpha^i}_{(e)}\right) \\
&\quad + \left(\sum_{i=-n+3k_1-7}^{n-1} \underbrace{(s_{n-k_1+i+2}+s_{2n-2k_1+i+4})\alpha^i}_{(g)}\right) \quad \Rightarrow(8.4)
\end{aligned} \tag{8}
$$

우선 공간복잡도를 고려해보자. 식 (8)에서 (d)의 $n-k_1+2$ XOR 게이트 계산에서 (a), (b), (f), (g)는 모두 계산되고 (c) $-n+2k_1-7$ XOR 게이트 계산에서 (e)가 계산된다. (8.1)는 3 XOR 게이트로 계산되며 (8.2)는 $n-k_1+2$ XOR 게이트로 계산된다. 따라서 (8.3)+(8.4)는 연산량이 없으며, (8.1)+ (8.2)는 n XOR 게이트, ((8.1)+(8.2))+ ((8.3)+ (8.4)) 는 $n-5$ XOR 게이트로 계산되므로 모듈러 감산에서 $3n-5$ XOR 게이트로 계산되며 전체 $n^2\mathrm{AND}+(\mathrm{n}^2+\mathrm{n}-4)\mathrm{XOR}$가 소요된다. 다음으로 시간복잡도를 살펴보자. (8.1) 계산에서 3 $\mathrm{T_X}$, (8.2) 계산에서 1 $\mathrm{T_X}$, (8.3)+(8.4) 계산에서 2$\mathrm{T_X}$가 소요되므로 모듈러 감산에서 $4T_X$가 추가적으로 필요하다.

다음으로 $r=3$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2$, $6 \le k_1 \le (n+7)/2$를 가정하자. 그러면 최대인 $k_1=\lfloor (n+7)/2 \rfloor$의 경우 식 (4)는 다음과 같다.

$$
\begin{aligned}
&\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{2} s_{n+i} \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \\
&\quad + \left\{\left(\sum_{i=n+3}^{2n-2} s_i\alpha^{-n+i-2}\right)(x^{k_1}+1) \bmod f(x)\right\} \\
&= \sum_{i=0}^{2}(s_{n+i}+s_{2n-k_1+3}) \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1} \quad \Rightarrow(9.1) \\
&\quad + \underbrace{\sum_{i=0}^{n-1} s_i\alpha^i + \sum_{i=k_1-7}^{n-5} s_{n+i+3}\alpha^i + \sum_{i=2k_1-7}^{n-1} s_{n-k_1+i+3}\alpha^i}_{(a)} \\
&\quad + \underbrace{\sum_{i=0}^{k_1-8}(s_{n+i+3}+s_{2n-k_1+i+5})\alpha^i}_{(c)} \\
&\quad + \underbrace{\sum_{i=k_1}^{2k_1-8}(s_{n-k_1+i+3}+s_{2n-2k_1+i+5})\alpha^i}_{(d)}
\end{aligned} \tag{9}
$$

(c)+(d)를 (b)라고 정의하자. (9.1) 계산에서 $\varepsilon+3$ XOR 게이트, (a) 계산에서 $2n-3k_1+10$ XOR 게이트, (c)의 k_1-7 XOR 게이트 계산에서 (d)는 계산되었으므로 연산량이 없다. 그리고 (a)+(b)에서 $2k_1-14$ XOR 게이트, (9.1)+((a) +(b))에서 n XOR 게이트로 계산되므로 모듈러 감산에서 $3n+\varepsilon-8$ XOR 게이트가 소요된다. 또한 (9.1)에서 $3\mathrm{T_X}$, (a)+(b)에서 $3\mathrm{T_X}$ 시간지연이 소요되므로 모듈러 감산에서 $4\mathrm{T_X}$가 소요된다.

다음으로 $r=4$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2$, $6 \le k_1 \le (n+9)/2$를 가정하자. 그러면 최대인 $k_1=\lfloor (n+9)/2 \rfloor$의 경우 식 (4)는 다음과 같다.

$$
\begin{aligned}
&\sum_{i=0}^{n-1} s_i\alpha^i + \left(\sum_{i=0}^{3} s_{n+i} \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1}\right) \\
&\quad + \left\{\left(\sum_{i=n+4}^{2n-2} s_i\alpha^{-n+i-2}\right)(x^{k_1}+1) \bmod f(x)\right\} \\
&= \sum_{i=0}^{3}(s_{n+i}+s_{2n-k_1+4}) \sum_{j=0}^{n-1} t_{i,j}\alpha^{n-j-1} \quad \Rightarrow(10.1) \\
&\quad + \underbrace{\sum_{i=0}^{n-1} s_i\alpha^i + \sum_{i=k_1-9}^{n-6} s_{n+i+4}\alpha^i + \sum_{i=2k_1-9}^{n-1} s_{n-k_1+i+4}\alpha^i}_{(a)} \\
&\quad + \underbrace{\sum_{i=0}^{k_1-10}(s_{n+i+4}+s_{2n-k_1+i+8})\alpha^i}_{(c)} \\
&\quad + \underbrace{\sum_{i=k_1}^{2k_1-10}(s_{n-k_1+i+4}+s_{2n-2k_1+i+8})\alpha^i}_{(d)}
\end{aligned} \tag{10}
$$

(c)+(d)를 (b)라고 정의하자. (10.1) 계산에서 $\varepsilon+4$ XOR 게이트, (a) 계산에서 $2n-3k_1+13$ XOR 게이트, (c)의 k_1-9 XOR 게이트 계산에서 (d)는 계산되

었으므로 연산량이 없다. 그리고 (a)+(b)에서 $2k_1-18$ XOR 게이트, (10.1)+ ((a)+(b))에서 n XOR 게이트로 계산되므로 모듈러 감산에서 $3n+\varepsilon-10$ XOR 게이트가 소요된다. 또한 (10.1)에서 $3T_X$, (a)+(b)에서 $3T_X$ 시간지연이 소요되므로 모듈러 감산에서 $4T_X$가 소요된다.

위에서 가정한 경우 보다 k_1이 큰 경우 REDUCTION BLOCK 1의 계산에만 $4T_X$가 소요되므로 어느 case에도 해당하지 않는다. case 2에 해당하기 위해서는 REDUCTION BLOCK 1의 행의 개수가 8이하가 되어야 한다. 식 (4.1)의 REDUCTION BLOCK 1에서 행 개수는 r이며, 식 (4.2)는

$$\left\{\left(\sum_{i=0}^{n-r-2} s_{n+r+i}\alpha^i\right)(x^{k_1}+1) \bmod f(x)\right\}$$

이므로 $(n-r-2+k_1)-(n-1)+r \le 8$이다. 따라서 $r>4$인 경우 case 2를 만족하기 위해서는 $r<9$이고 $k_1 \le 9$이어야 한다. 이 경우 모듈러 감산의 공간복잡도는 $2(n-r-2)+\varepsilon+n=3n-2r+\varepsilon-4$ XOR 게이트이고 추가되는 시간 복잡도는 $4T_X$이다. 따라서 다음 RP만 case 2을 만족한다.

1) $r=2$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2$, $\lfloor (n+5)/2 \rfloor < k_1 \le \lfloor (2n+7)/3 \rfloor$,
2) $r=3$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2, 6 \le \mathrm{k}_1 \le \lfloor (\mathrm{n}+7)/2 \rfloor$,
3) $r=4$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2, 6 \le \mathrm{k}_1 \le \lfloor (\mathrm{n}+9)/2 \rfloor$,
4) $4<r<9$, $\mathbf{H}(\mathrm{h}(\mathrm{x}))=2, \mathrm{k}_1 \le 9$.

3.3.3 case 3인 경우

우선 $r=1$인 경우는 $\mathbf{H}(\mathrm{h}(\mathrm{x}))=3$이므로 $h(x)=x^{k_1}+x^{k_2}+1$라 하자($k_1>k_2>1$). 우선 $2k_1 \le n+3$라 가정하면 최대 $k_1=\lfloor (n+3)/2 \rfloor$에서 식 (4)는 다음과 같다.

$$\begin{aligned}
&\sum_{i=0}^{n-1} s_i\alpha^i+\left(s_n\sum_{j=0}^{n-1} t_{i,j}\alpha^{n-1-j}\right)\\
&+\left\{\left(\sum_{i=n+1}^{2n-2} s_i\alpha^{-n+i-1}\right)(x^{k_1}+x^{k_2}+1) \bmod f(x)\right\}\\
&=\sum_{i=0}^{n-1} s_i\alpha^i+\left((s_n+(s_{2n-k_1+1}+s_{2n-k_2+1}))\sum_{j=0}^{n-1} t_{0,j}\alpha^{n-1-j}\right)\\
&+\underbrace{\left(\sum_{i=0}^{n-3} s_{n+i+1}\alpha^i\right)}_{(b1)}+\underbrace{\left(\sum_{i=k_1}^{n-1} s_{n-k_1+i+1}\alpha^i\right)}_{(c1)}+\underbrace{\left(\sum_{i=k_2}^{n-1} s_{n-k_2+i+1}\alpha^i\right)}_{(d)}\\
&+\underbrace{\left(\sum_{i=0}^{k_1-4} s_{2n-k_1+i+2}\alpha^i\right)}_{(b2)}+\underbrace{\left(\sum_{i=k_2}^{k_1+k_2-4} s_{2n-k_1-k_2+i+2}\alpha^i\right)}_{(a1)}\\
&+\underbrace{\left(\sum_{i=k_1}^{2k_1-4} s_{2n-2k_1+i+2}\alpha^i\right)}_{(c2)}+\left(\sum_{i=0}^{k_2-4} s_{2n-k_2+i+2}\alpha^i\right) \qquad (11)\\
&+\underbrace{\left(\sum_{i=k_2}^{2k_2-4} s_{2n-2k_2+i+2}\alpha^i\right)}_{(e)}+\underbrace{\left(\sum_{i=k_1}^{k_1+k_2-4} s_{2n-k_1-k_2+i+2}\alpha^i\right)}_{(a2)}
\end{aligned}$$

식 (5)에서 (a1)+(a2)는 $k_1>k_2$이므로

$$\begin{aligned}
&\underbrace{\left(\sum_{i=k_2}^{k_1+k_2-4} s_{2n-k_1-k_2+i+2}\alpha^i\right)}_{(a1)}+\underbrace{\left(\sum_{i=k_1}^{k_1+k_2-4} s_{2n-k_1-k_2+i+2}\alpha^i\right)}_{(a2)}\\
&=\sum_{i=k_2}^{k_1-1} s_{2n-k_1-k_2+i+2}\alpha^i \qquad (12)
\end{aligned}$$

이고, (b1)+(b2)와 (c1)+(c2)는

$$\begin{aligned}
&\underbrace{\left(\sum_{i=0}^{n-3} s_{n+i+1}\alpha^i\right)}_{(b1)}+\underbrace{\left(\sum_{i=0}^{k_1-4} s_{2n-k_1+i+2}\alpha^i\right)}_{(b2)}\\
&=\left(\sum_{i=0}^{k_1-4}(s_{n+i+1}+s_{2n-k_1+i+2})\alpha^i\right)+\left(\sum_{i=k_1-3}^{n-3} s_{n+i+1}\alpha^i\right),\\
&\underbrace{\left(\sum_{i=k_1}^{n-1} s_{n-k_1+i+1}\alpha^i\right)}_{(c1)}+\underbrace{\left(\sum_{i=k_1}^{2k_1-4} s_{2n-2k_1+i+2}\alpha^i\right)}_{(c2)}\\
&=\left(\sum_{i=k_1}^{2k_1-4}(s_{n-k_1+i+1}+s_{2n-2k_1+i+2})\alpha^i\right)\\
&\quad+\left(\sum_{i=2k_1-3}^{n-1} s_{n-k_1+i+1}\alpha^i\right)
\end{aligned}$$

이므로 $(s_{n-k_1+i+1}+s_{2n-2k_1+i+2})$는 의 $(s_{n+i+1}+s_{2n-k_1+i+2})$과 중복 연산이므로 연산량이 없다. 또한 식 (12)와 (d)의 합은

$$\begin{aligned}
&\underbrace{\left(\sum_{i=k_2}^{n-1} s_{n-k_2+i+1}\alpha^i\right)}_{(d)}+\sum_{i=k_2}^{k_1-1} s_{2n-k_1-k_2+i+2}\alpha^i=\\
&\left(\sum_{i=k_2}^{k_1-1}(s_{n-k_2+i+1}+s_{2n-k_1-k_2+i+2})\alpha^i\right)+\sum_{i=k_1}^{n-1} s_{n-k_2+i+1}\alpha^i
\end{aligned}$$

이므로 이 또한 $(s_{n+i+1}+s_{2n-k_1+i+2})$과 중복연산으로 연산량이 없다. 마지막으로 (e)=$\delta_1+\delta_2$를 위하여 δ_1,δ_2를 정리하면

$$\delta_1 = \begin{cases} \sum_{i=k_1-3}^{2k_2-4} s_{2n-2k_2+i+2}\alpha^i, & 2k_2-4 \geq k_1-3 \\ 0, & 2k_2-4 < k_1-3 \end{cases},$$

$$\delta_2 = \begin{cases} \sum_{i=k_2}^{k_1-4} s_{2n-2k_2+i+2}\alpha^i, & 2k_2-4 \geq k_1-3 \\ \sum_{i=k_2}^{2k_2-4} s_{2n-2k_2+i+2}\alpha^i, & 2k_2-4 < k_1-3 \end{cases}$$

이고, 위의 내용을 적용하여 식 (11)을 정리하면 다음과 같다.

$$\begin{aligned} &\left(\left(s_n+(s_{2n-k_1+1}+s_{2n-k_2+1})\right)\sum_{j=0}^{n-1} t_{0,j}\alpha^{n-1-j}\right) && \Rightarrow(13.1)\\ &+\left\{\sum_{i=0}^{n-1} s_i\alpha^i+\left(\sum_{i=k_1-3}^{n-3} s_{n+i+1}\alpha^i+\delta_2+\sum_{i=0}^{k_2-4} s_{2n-k_2+i+2}\alpha^i\right)\right\} && \Rightarrow(13.2)\\ &+\left\{\sum_{i=k_1}^{n-1} s_{n-k_2+i+1}\alpha^i+\left(\sum_{i=2k_1-3}^{n-1} s_{n-k_1+i+1}\alpha^i+\delta_1\right)\right\} \\ &+\sum_{i=0}^{k_1-4}(s_{n+i+1}+s_{2n-k_1+i+2})\alpha^i && \Rightarrow(13.3)\\ &+\sum_{i=k_1}^{2k_1-4}(s_{n-k_1+i+1}+s_{2n-2k_1+i+2})\alpha^i \\ &+\sum_{i=k_2}^{k_1-1}(s_{n-k_2+i+1}+s_{2n-k_1-k_2+i+2})\alpha^i && \Rightarrow(13.4) \end{aligned} \quad (13)$$

식 (13.1)의 에서 s_{2n-k_1+1}와 s_{2n-k_2+1}는 각각 $\log_2(k_1-2)T_X$, $\log_2(k_2-2)T_X$ 시간지연을 가지므로 가정 $2k_1 < n+4$에 의하여 모두 $\log_2(n/2)T_X$보다 작은 계산시간을 가지며 $(s_{2n-k_1+1}+s_{2n-k_2+1})$는 s_{n-1}과 동시에 계산되므로 식 (13.1)의 계산은 $1T_X$에 계산 가능하다. 또한(13.2)~(13.4)는 모두 $1T_X$에 계산되므로 식 (13)의 시간지연은 $1T_A+(\lceil \log_2 n \rceil+3)T_X$이다. 다음으로 공간복잡도를 살펴보자. (13.2)+(13.3)은 $(n-k_1+1)+(k_2-3)+(n-k_1)+(n-2k_1+3)+(k_2-3)$와 $2(k_1-3)$의 합이므로 전체 $3n-2k_1+2k_2-8$ XOR 게이트로 연산된다. 또한 (13.1)은 2 XOR 게이트로 연산되며, (13.4)는 (13.3)에서 계산되어 연산량이 없으므로 ((13.2)+ (13.3))+(13.4)에 $2k_1-k_2-3$ XOR 게이트로 연산되며, ((13.2)+(13.3)+(13.4)) +(13.1)은 $\mathbf{H}(f(x))$와 같으므로 이를 정리하면 전체 n^2AND+$\left((n-1)^2+(3n+k_2+\boldsymbol{H}(f(x))-9)\right)$XOR이고 n^2AND+$\left((n)^2+(n+k_2+\boldsymbol{H}(f(x))-8)\right)$XOR으로 #XOR $\approx n^2+2n$이므로 case 3이다. 그리고 가정과 달리 $2k_1 \geq n+4$인 경우 (13.1)이 $2T_X$가 되어 어느 case에도 해당하지 않는다.

case 3의 경우 REDUCTION BLOCK2 계산에서 $2T_X$가 소요되므로 REDUCTION BLOCK1의 행의 개수가 4보다 작은 $r<5$이다. 따라서 $1 \leq r \leq 4$인 경우 식 (4.2)를 고려했을 때 $(n-r-2+k_1)-(n-1)+r \leq 4$이므로 $k_1 \leq 5$이다. 이 경우 모듈러 감산의 공간복잡도는 $3(n-r-2)+\varepsilon+n=4n-2r+\varepsilon-4$XOR 게이트이고 추가되는 시간 복잡도는 $3T_X$이다. 따라서 다음 RP만 case 3을 만족한다.

1) $r=1, \mathbf{H}(h(x))=3, 2k_1 \leq n+3,$
2) $1<r<5, \mathbf{H}(h(x))=3, k_1 \leq 5.$

IV. 비교 및 결론

본 절에서는 제안하는 RP 기약다항식과 기존의 기약다항식(ESP, Trinomial, Pentanomial)과 효율성을 비교하교 결론을 내린다. 본 논문에서는 새로운 기약다항식인 RP 기약다항식을 제안하였으며, [표 1]에서 보인 기존의 오항 기약다항식 보다 차수에 따라 효율적인 세 가지 경우의 RP를 제안하였다. 제안하는 세 가지 경우의 기약다항식의 시간 및 공간 복잡도를 r에 따라 정리하면 [표 2]와 같다. 또한 차수가 $1 \leq n \leq 1{,}000$인 경우의 RP를 정리하면 [표 3]과 같다. 가장 효율적인 기약다항식인 ESP와 Trinomial이 없는 443개의 차수 중 [표 3]에서 알 수 있듯이 case 1의 경우 181(40.8%)개가 존재하며, case 2의 경우 232(52.4%)개, case 3의 경우 443(100%)개 존재한다. 또한 case 3 $r=1$의 경우 $n=1,2,16,515$를 제외한 모든 차수에서 존재한다.

타원곡선 암호시스템의 표준(ANSI X9.62, IEEE P1363, SEC2)에서 제안되어 사용되는 이진체 중 오항 기약다항식을 사용하는 차수인 n=131, 163, 283, 571 모두에서 RP가 존재하며 이를 정리하면

- n=131 : $f(x)=\left(\sum_{i=67}^{131} x^i\right)+x+1$
- n=163 : $f(x)=\left(\sum_{i=33}^{163} x^i\right)+\left(\sum_{i=0}^{21} x^i\right),$

$f(x)=\left(\sum_{i=63}^{163} x^i\right)+\left(\sum_{i=0}^{5} x^i\right),\ f(x)=\left(\sum_{i=63}^{163} x^i\right)+\left(\sum_{i=0}^{41} x^i\right),$

$f(x)=\left(\sum_{i=71}^{163} x^i\right)+\left(\sum_{i=0}^{1} x^i\right),\ f(x)=\left(\sum_{i=75}^{163} x^i\right)+\left(\sum_{i=0}^{29} x^i\right),$

$f(x)=\left(\sum_{i=77}^{163} x^i\right)+\left(\sum_{i=0}^{21} x^i\right)$

[표 2] 제안하는 RP 기반의 비트-병렬 곱셈기 복잡도 비교

$x^{n+r} = h(x) \bmod f(x)$			Space Complexity		Time Complexity Upper Bound
r	$\deg(h(x)) = k_1$	$\mathbf{H}(\mathrm{h(x)})$	#AND	#XOR	
2	$\leq \lfloor (n+5)/2 \rfloor$	2	n^2	$\leq n^2+n-6$	$1T_A + \lceil \log_2(n)+3 \rceil T_X$
3	≤ 5	2	n^2	$\leq n^2+n-2$	$1T_A + \lceil \log_2(n)+3 \rceil T_X$
4	≤ 5	2	n^2	$\leq n^2+n-1$	$1T_A + \lceil \log_2(n)+3 \rceil T_X$
2	$\lfloor (n+5)/2 \rfloor < k_1 \leq \lfloor (2n+7)/3 \rfloor$	2	n^2	$\leq n^2+n-4$	$1T_A + \lceil \log_2(n)+4 \rceil T_X$
3	$6 \leq k_1 \leq \lfloor (n+7)/2 \rfloor$	2	n^2	$\leq n^2+n-3$	$1T_A + \lceil \log_2(n)+4 \rceil T_X$
4	$6 \leq k_1 \leq \lfloor (n+9)/2 \rfloor$	2	n^2	$\leq n^2+n+3$	$1T_A + \lceil \log_2(n)+4 \rceil T_X$
5,6,7,8	≤ 9	2	n^2	$\leq n^2+n-2r+\varepsilon-3$	$1T_A + \lceil \log_2(n)+4 \rceil T_X$
1	$\leq \lfloor (n+3)/2 \rfloor$	3	n^2	$\leq n^2+n+k_2+\mathbf{H}(\mathrm{f(x)})-$	$1T_A + \lceil \log_2(n)+3 \rceil T_X$
2,3,4	≤ 5	3	n^2	$\leq n^2+2n-2r+\varepsilon-3$	$1T_A + \lceil \log_2(n)+3 \rceil T_X$
※ 만약 $\mathbf{H}(\mathrm{h(x)})=2$인 경우, $h(x)=x^{k_1}+1\ (k_1>0)$. 만약 $\mathbf{H}(\mathrm{h(x)})=3$인 경우, $h(x)=x^{k_1}+x^{k_2}+1(k_1>k_2>0)$					

- n=283 : $f(x)=\left(\sum_{i=0}^{31}(x^4+x^2+x+1)x^{(62+7\cdot i)}\right)+(x^4+x^2+x+1)x^{55}+\left(\sum_{i=0}^{6}(x^4+x^2+x^1+1)x^{(7\cdot i)}\right)$

$f(x)=\left(\sum_{i=15}^{283}x^i\right)+\left(\sum_{i=0}^{9}x^i\right)$, $f(x)=\left(\sum_{i=97}^{283}x^i\right)+\left(\sum_{i=0}^{1}x^i\right)$,

$f(x)=\left(\sum_{i=105}^{283}x^i\right)+\left(\sum_{i=0}^{29}x^i\right)$, $f(x)=\left(\sum_{i=123}^{283}x^i\right)+\left(\sum_{i=0}^{5}x^i\right)$

- n=571 : $f(x)=\left(\sum_{i=207}^{571}x^i\right)+\left(\sum_{i=0}^{45}x^i\right)$

이다. n=283인 경우 case 2 $r=3$에서 하나 존재하며 나머지는 모두 case3 $r=1$인 경우에서 존재한다. 따라서 제안하는 RP 기약다항식은 이진체 차수 $1 \leq n \leq 1{,}000$에서 모두 존재하며 EPS 또는 삼항 기약다항식이 없는 차수에서 차수에 따라 오항 기약다항식 기반의 비트-병렬 곱셈기 보다 효율적이다.

[표 3] 경우에 따른 RP 기약다항식 비교

기약다항식		기약다항식 개수	$1 \leq n \leq 1{,}000$ 기약다항이 존재하는 차수의 개수			ESP 또는 Trinomial이 없는 차수의 개수		
case	r							
ESP		80	77			557 (ESP 또는 Trinomial이 존재하는 차수의 개수)		
Trinomial		1,513	543					
1	2	504	279	315	1,000	178	181	443
	3	43	40			5		
	4	11	11			3		
2	2	150	128	592		86	232	
	3	482	300			37		
	4	273	192			128		
	5	51	42			18		
	6	29	23			3		
	7	55	42			16		
	8	40	33			7		
3	1	117,858	996	999		442	443	
	2	71	65			23		
	3	70	59			27		
	4	82	68			29		

참 고 문 헌

[1] A. Halbutogullari and C.K. Koc, "Mastrovito Multiplier for General Irreducible Polynomials," IEEE Trans. Computers, vol. 49, no. 5, pp. 503-518, May 2000.

[2] 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. 945-958, Aug. 2004.

[3] E.D. Mastrovito, "VLSI Architectures for Computation in Galois Fields," Ph.D. Thesis, Linkoping, Sweden, 1991.

[4] H. Wu, "Bit-Parallel Finite Field Multiplier and Squarer Using Polynomial Basis," IEEE Trans. Computers, vol. 51, no. 7, pp. 750-758, July 2002.

[5] F. Rodriguez and C.K. Koc, "Parallel Multipliers Based on Special Irreducible Pentanomials," IEEE Trans. Computers, vol. 52, no. 12, pp. 1535-1542, Dec. 2003.

[6] T. Zhang and K.K. Parhi, "Systematic Design of Original and Modified Mastorvito Multipliers for General Irreducible Polynomials," IEEE Trans. Computers, vol. 50, no. 7, pp. 734-748, July 2001.

[7] H. Wu, "Bit-Paralle polynomial Basis Multiplier for New Classes of Finite Fields," IEEE Trans. Computers, vol. 57, no. 8, pp. 1023-1031, Aug. 2008.

<著 者 紹 介>

장 남 수 (Nam Su Chang) 학생회원
2002년 2월: 서울 시립대학교 수학과 이학사
2004년 8월: 고려대학교 정보보호 대학원 공학석사
2005년 2월 ~ 현재: 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 암호칩 설계 기술, 부채널 공격, 공개키 암호 알고리즘, 공개키 암호 암호분석

김 창 한 (Chang Han Kim) 정회원
1985년 2월: 고려대학교 수학과 학사
1987년 2월: 고려대학교 수학과 석사
1992년 2월: 고려대학교 수학과 박사
1992년 3월 ~ 현재: 세명대학교 정보통신학부 교수
<관심분야> 정수론, 공개키암호, 암호프로토콜

홍 석 희 (Seokhie Hong) 정회원
1995년: 고려대학교 수학과 학사
1997년: 고려대학교 수학과 석사
2001년: 고려대학교 수학과 박사
1999년 8월 ~ 2004년 2월: (주)시큐리티 테크놀로지스 선임연구원
2003년 3월 ~ 2004년 2월: 고려대학교 시간강사
2004년 4월 ~ 2005년 2월: K.U. Leuven 박사후연구원
2005년 3월 ~ 현재: 고려대학교 정보경영전문대학원 부교수
<관심분야> 대칭키 암호 알고리즘, 공개키 암호 알고리즘, 포렌식