Digit-Serial Systolic Karatsuba Multiplier for Special Classes over GF(2^m)

Che Wun Chiou[1*] Chiou-Yng Lee[2] Jim-Min Lin[3] Yun-Chi Yeh[4]

Hung Wei Chang[5] Liuh-Chii Lin[4]

[1] Department of Computer Science and Information Engineering, Chien Hsin University of Science and Technology
Taoyuan City 32097, Taiwan
cwchiou@uch.edu.tw

[2] Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology
Taoyuan City 33306, Taiwan
PP010@mail.lhu.edu.tw

[3] Department of Information Engineering and Computer Science, Feng Chia University
Taichung City 407, Taiwan
jimmy@fcu.edu.tw

[4] Department of Electronic Engineering, Chien Hsin University of Science and Technology
Jhong-Li 32097, Taiwan
{yunchi, chiilin}@uch.edu.tw

[5] SW Architect, Top Victory Electronics (Taiwan) Co., Ltd.
New Taipei City 23553, Taiwan
marvin1117@gmail.com

Received 29 August 2014; Revised 23 September 2014 ; Accepted 20 March 2015

Abstract. Finite field multiplication over GF(2^m) is one of the most important arithmetic operations for Elliptic Curve Cryptosystem (ECC). Polynomial basis multipliers over GF(2^m) are widely applied in ECC due to its regular, modular, easily expansible benefits and the high suitability for VLSI implementation. This study will present a novel digit-serial polynomial basis multiplier using Karatsuba algorithm representation. To achieve efficient architectures, our proposed digit-serial architecture is different from existing digit-serial polynomial basis multipliers that use cut-set algorithm. The proposed digit-serial polynomial basis multiplier saves 90% space complexity as compared to existing similar studies. Existing digit-serial polynomial basis multipliers employ one dimensional array of digit cells, but our proposed digit-serial polynomial basis multiplier uses only one digit cell.

Keywords: Karatsuba algorithm, elliptic curve cryptosystem, finite field multiplication, digit-serial multiplier, systolic.

1 Introduction

Finite field arithmetic operations have played an important role in many applications, e.g., error-correcting code [1], digital signal processing [2], and cryptography [3]. Public-key cryptosystems such as elliptic curve cryptosystem (ECC) [4,5], hyperelliptic curve cryptosystem (HECC) [6], and pairing based cryptosystem [7] have become increasingly popular in the last few years. Elliptic curve cryptosystem was suggested in 1985 by Victor Miller [4] and Neil Koblitz [5] as an alternative mechanism for implementing public-key cryptosystem. Elliptic curve cryptosystem relies on the believed difficulty of the elliptic curve discrete logarithm for its security. Today, due to the high level of security with relatively small keys provided by ECC, ECC has gained increasing acceptance and has been the subject of several standards in the industry and the academic community. The performance of these public-key cryptosystems are highly dependent on the efficiency of finite field arithmetic over prime field GF(p), characteristic two field GF(2^m), and characteristic three field GF(3^m). Due to advantages of low hardware cost and fast execution time, GF(2^m) arithmetic is often chosen for realizing these public-key cryptosystems. Finite field arithmetic operations in GF(2^m) may generally include addition, multiplication, mul-

tiplicative inversion, division, and exponentiation. Addition is actually a simple bit independent XOR operation. The other operations, i.e., multiplicative inversion, division, and exponentiation, are much more sophisticated. Fortunately, these operations could be performed by repeating a multiply-square algorithm. Thus, finite field multiplication is actually the most critical arithmetic operation in $GF(2^m)$.

The efficiency of finite field multiplication in $GF(2^m)$ is deeply relied on how elements are represented. There are three major basis representations: polynomial basis (PB) [8-22], dual basis (DB) [23-27], and normal basis (NB) [10, 27-35]. Each basis has its own features. DB multipliers have smaller chip area than that of the multipliers of the other two bases. The major merit of NB architectures is that squaring could be simply performed just by cyclically shifting its binary form. Therefore, NB multipliers are effective and efficient for performing multiplicative inversion, squaring, and exponentiation operations. PB multipliers own the major features of simplicity, regularity, and modularity. Thus, PB multipliers are particularly suited to VLSI implementation.

Typically, multipliers in $GF(2^m)$ can be classified into four types: bit-serial [25,26], bit-parallel [10,12,14,23,27,35], hybrid [13,15], and digit-serial [16-19]. Bit-serial multipliers iteratively generate a result bit per clock cycle and thus have the advantage of low hardware cost. Bit-parallel multipliers generate all result bits in parallel in the same single clock cycle and therefore have higher hardware cost. Hybrid multipliers in [15] can be used to design subquadratic space complexity multipliers using various bases. Digit-serial multipliers give a designer the flexibility of making trade-offs between speed and space. Digit-serial multipliers are practical for resource constrained devices, like smart phones. However, existing digit-serial PB multipliers [16-19] realized digit-serial architecture with one dimensional array of digit cells. To overcome this problem, our proposed digit-serial multiplier uses only one digit cell. Therefore, the proposed digit-serial multiplier has lower space complexity than the existing similar multipliers.

Karatsuba method [36] is a fast multiplication algorithm for multi-precision numbers with $O(m^{1.58})$ asymptotic complexity as compared to the schoolbook multiplication method with $O(m^2)$ complexity [37]. Applying the concept of Karatsuba algorithm, finite field multiplications in $GF(2^m)$ were proposed in [20,38,39,40]. Zhou *et al.* [20] applied Karatsuba-Ofman algorithm to give efficient bit-parallel polynomial basis multipliers. Beuchat *et al.* [41] developed Karatsuba-Ofman multipliers over $GF(3^m)$ for accelerating the Tate Pairing in supersingular elliptic curves. Ghosh *et al.* [42] proposed a first 128-bit secure η_T pairing over $GF(2^m)$ for supersingular elliptic curves. Morales-Sandoval [43] utilized linear feedback shift registers for designing digit-serial $GF(2^m)$ Montgomery multipliers. Chen *et al.* [44] employed Toeplitz matrix for developing scalable and systolic Montgomery multipliers. These bit-parallel Karatsuba multipliers suffer from the problem of long gate delay. This study presents a novel systolic Karatsuba digit-serial PB multiplier with the features of low hardware cost and short gate delay. The proposed digit-serial PB multiplier will reduce a $2d \times 2d$ array to a $d \times d$ array by using the Karatsuba algorithm for further reduction of space complexity.

Kim et al. [17] proposed a systolic digit-serial multiplier for finite field $GF(2^m)$ by applying the cut-set systolization technique for obtaining less delay time than previously proposed similar multipliers. Talapatra et al. [19] presented an efficient digit-serial Montgomery multiplier for all-one polynomial over $GF(2^m)$. These existing digit-serial multipliers employ one-dimensional array of digit cells. However, low-hardware cost design of multipliers in $GF(2^m)$ is very important in resource-limited mobile devices such as smart phones for E-commerce. Thus, the motivation of this study is to develop a low-cost multiplier for resource-limited mobile devices. In this paper, the proposed digit-serial multiplier uses one digit cell other than one-dimensional array of digit cells in existing similar multipliers for achieving low hardware cost design.

Two major contributions of this study are listed as follows:

(a) It is the first digit-serial PB multiplier that uses only one digit cell, while the existing similar architectures employ one-dimensional array of digit cells. Obviously, the proposed digit-serial multiplier has lower space complexity.
(b) It is the first digit-serial PB multiplier that uses the Karatsuba algorithm to reduce $2d \times 2d$ array to a $d \times d$ array.

The rest of this paper is organized as follows. Section 2 describes a systolic bit-parallel PB multiplier using Karatsuba algorithm. Section 3 proposes the novel systolic digit-serial PB multiplier using Karatsuba algorithm. The comparing results are then discussed in Section 4. A brief conclusion is finally made in Section 5.

2 Finite Field Multiplication and Systolic Bit-Parallel PB Multiplier Using Karatsuba Algorithm

This section will briefly review the traditional finite field polynomial basis multiplication over $GF(2^m)$, and will propose a systolic bit-parallel PB multiplier using Karatsuba algorithm. Based on such bit-parallel PB multiplier, the proposed digit-serial PB multiplier will be introduced in the next section.

2.1 Finite Field Multiplication

The finite field GF(2^m) contains 2^m elements. GF(2^m) is an extension field of the ground field GF(2) of 2 elements, i.e., GF(2)={0,1}. GF(2^m) is a vector space over GF(2). All arithmetic operations over GF(2^m) are carried out by taking the results modulo 2. Suppose that the finite field GF(2^m) is generated by the irreducible polynomial $P(x) = p_0 + p_1x^1 + p_2x^2 + ... + p_{m-1}x^{m-1} + x^m$ of degree m over GF(2).

Let *A(x)*, B*(x)*, *C(x)* be elements over GF(2^m), where *C(x)* is the product of *A(x)* and *B(x)*, i.e., *C(x)=A(x)B(x)* mod *P(x)*. Then *A(x)*, B*(x)*, *C(x)* can be expressed as follows:

$$\begin{aligned} A(x) &= a_0 + a_1x + ... + a_{m-1}x^{m-1}, \\ B(x) &= b_0 + b_1x + ... + b_{m-1}x^{m-1}, \\ C(x) &= c_0 + c_1x + ... + c_{m-1}x^{m-1}. \end{aligned} \tag{1}$$

C(x) is the product of *A(x)* and *B(x)*. Then, we have

$$\begin{aligned} & C(x) \\ &= A(x) \times B(x) \bmod P(x) \\ &= \left(a_0 + a_1x + a_2x^2 + ... + a_{m-1}x^{m-1}\right)B \\ &= a_0B + a_1xB + a_2x^2B + ... + a_{m-1}x^{m-1}B. \end{aligned} \tag{2}$$

For clarity, let us note that *A(x)*, *B(x)*, *C(x)*, and *P(x)* are simplified to *A*, *B*, *C*, and *P* in the remaining of the paper. Finite field multiplication over GF(2^m) is different from standard integer multiplication. There are a limited number of elements in the finite field and all operations performed in the finite field result in an element within that field. Finite field multiplication is multiplication modulo *P* used to define the finite field.

2.2 Systolic Bit-Parallel PB Multiplier Using Karatsuba Algorithm

Using summation equation, *A* and *B* are expressed as

$$A = \sum_{i=0}^{m-1} a_i x^i, \; B = \sum_{i=0}^{m-1} b_i x^i. \tag{3}$$

Assume that, both elements *A* and *B* can be subdivided into two parts as follows.

$$\begin{aligned} A &= A_L + x^{\frac{m}{2}} A_H, \\ B &= B_L + x^{\frac{m}{2}} B_H. \end{aligned} \tag{4}$$

By using Karatsuba algorithm, the product *C* is computed as follows.

$$\begin{aligned} & C \\ &= AB \\ &= A_LB_L + \left(A_LB_H + B_LA_H\right)x^{\frac{m}{2}} + A_HB_Hx^m \\ &= A_LB_L + \left(\left(A_L + A_H\right)\left(B_L + B_H\right) + A_LB_L + A_HB_H\right)x^{\frac{m}{2}} + A_HB_Hx^m \\ &= C_{LL} + C_{LH}x^{\frac{m}{2}} + C_{HH}x^m \\ &= C_1 + C_2x^{\frac{m}{2}} + C_3x^m + C_4x^{\frac{3m}{2}} \\ &= C_1 + C_2x^{\frac{m}{2}} + \text{Reduction}\left(C_3x^m + C_4x^{\frac{3m}{2}}\right) \\ &= C_{12} + C_{34}, \end{aligned} \tag{5}$$

where

$C_{LL} = A_LB_L$,

$C_{LH} = \left(A_L + A_H\right)\left(B_L + B_H\right) + A_LB_L + A_HB_H$,

$C_{HH} = A_HB_H$,

$C_1 = \text{low } \frac{\text{m}}{2} - \text{bit of } C_{LL}$,

$C_2 = \left(\text{high } \frac{\text{m}}{2} - \text{bit of } C_{LL}\right) \oplus \left(\text{low } \frac{\text{m}}{2} - \text{bit of } C_{LH}\right)$,

$C_3 = \left(\text{high } \frac{\text{m}}{2} - \text{bit of } \text{C}_{\text{LH}}\right) \oplus \left(\text{low } \frac{\text{m}}{2} - \text{bit of } C_{HH}\right)$,

$C_4 = \text{high } \frac{\text{m}}{2} - \text{bit of } C_{HH}$,

$C_{12} = C_1 + C_2 x^{\frac{m}{2}}$, and

$C_{34} = \text{Reduction}\left(C_3 x^m + C_4 x^{\frac{3m}{2}}\right)$.

The Reduction(H) operation denotes H mod P. According to Eq. (5), the Karatsuba algorithm for polynomial basis multiplication is described as follows:

Algorithm KA(*A(x),B(x),h*)

INPUT: Polynomials $A(x)$, $B(x)$, $P(x)$

OUTPUT: $C(x)=A(x)B(x) \bmod P(x)$

If $h<m/2$ then return 0

If h is even then $h=h+1$

Let $A(x)=A_L(x)+A_H(x)x^{h/2}$ and $B(x)=B_L(x)+B_H(x)x^{h/2}$

$D_1=KA(A_L(x),B_L(x),h/2)$

$D_2=KA(A_L(x)+A_H(x),B_L(x)+B_H(x),h/2)$

$D_3=KA(A_H(x),B_H(x),h/2)$

Return $(D_1+(D_2+D_1+D_3)x^{h/2}+D_3x^h) \bmod P(x)$

Algorithm KA(*A(x),B(x),m*) is applied for giving the product *C(x)*. Based on Eq. (5), an m×m multiplication can be performed by the following operations:

(1) Two $\frac{m}{2}\times\frac{m}{2}$ multiplications for $A_L B_L$ and $A_H B_H$.

(2) Two XOR operations for $\left(A_L + A_H\right)$ and $\left(B_L + B_H\right)$.

(3) One $\frac{m}{2}\times\frac{m}{2}$ multiplication for $\left(A_L + A_H\right)$ and $\left(B_L + B_H\right)$.

(4) Five XOR operations for summing partial results.

Three multiplications $A_L B_L$, $A_H B_H$, and $\left(A_L + A_H\right)\left(B_L + B_H\right)$ can sequentially employ the same multiplier for saving hardware cost and area. The hardware architecture for Eq. (5) is shown in Fig.1. It is noted that only the $\frac{m}{2}\times\frac{m}{2}$ multiplier is used in bit-parallel PB multiplier using Karatsuba algorithm while the m×m multiplier is utilized in a traditional bit-parallel multiplier. The bit-parallel PB multiplier using Karatsuba algorithm requires seven XOR operations, but traditional bit-parallel multipliers do not need. However, XOR operation is much simpler than multiplication. Thus, bit-parallel PB multiplier using Karatsuba algorithm could have lower space and time complexities than traditional ones if systolic array architecture is used in the $\frac{m}{2}\times\frac{m}{2}$ multiplier in Fig.1.

3 Proposed Systolic Digit-Serial PB Multiplier Using Karatsuba Algorithm

Considering the trade-offs between area and speed, digit-serial PB multiplier gives a proper solution for implementing cryptosystem in a hardware resource constrained environment, such as handheld devices. Traditional digit-serial PB multipliers can be further reduced on both space and time complexities by using Karatsuba algorithm. This novel digit-serial PB multiplier using Karatsuba algorithm will be presented in this section.

3.1 The Proposed Multiplier Using Karatsuba Algorithm

Elements *A* and *B* are represented in digit-serial form as follows. If each digit is represented with 2d bits and thus n (n=$\lceil$m/2d$\rceil$) digits are obtained.

$$A = \sum_{i=0}^{n-1} A_i x^{2id}, \qquad (6)$$

$$B = \sum_{j=0}^{n-1} B_j x^{2jd},$$

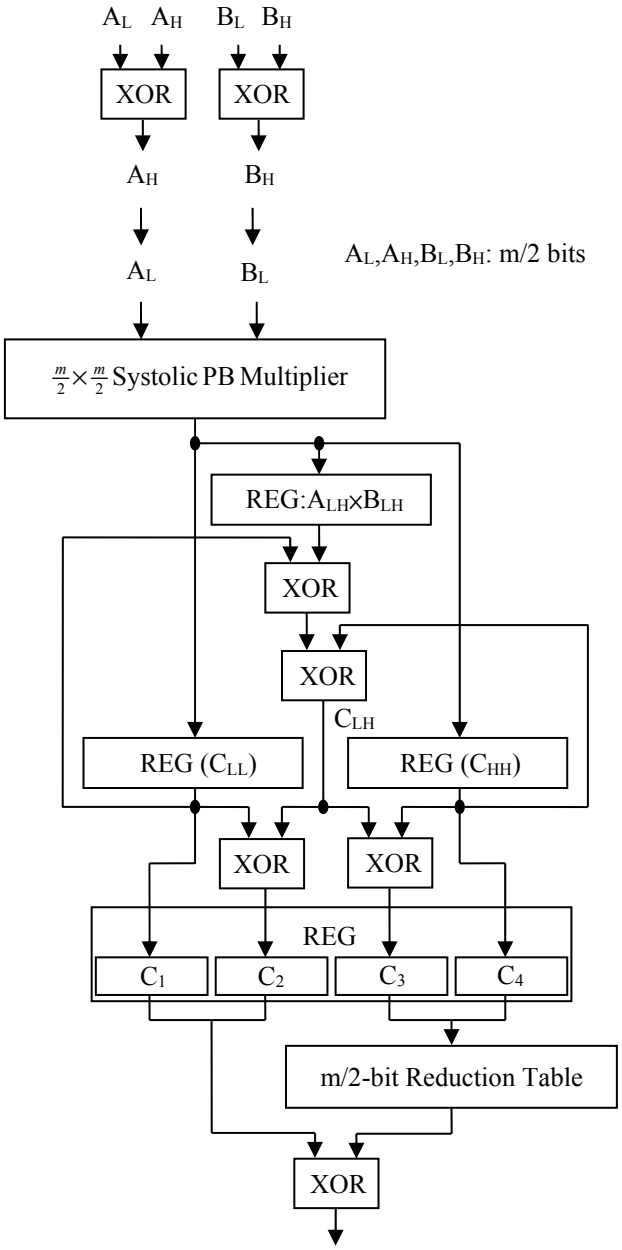


Fig.1. Systolic bit-parallel PB multiplier using Karatsuba algorithm

where $A_i = \sum_{k=0}^{2d-1} a_{2id+k} x^k$, $B_j = \sum_{k=0}^{2d-1} b_{2jd+k} x^k$.

The product C is computed as follows.

$$\begin{aligned} & C \\ &= AB \\ &= \sum_{i=0}^{n-1} A_i x^{2id} \times \sum_{j=0}^{n-1} B_j x^{2jd} \\ &= \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} A_i B_j x^{2(i+j)d} \\ &= \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} C_{ij} x^{2(i+j)d}, \end{aligned} \tag{7}$$

where $C_{ij} = A_i B_j$.

Eq.(7) can be rewritten as follows.

$$C = \left(\left(\left(AB_{n-1}\right)x^{2d} + AB_{n-2}\right)x^{2d} + ...\right)x^{2d} + AB_0 \mod P \tag{8}$$

Each sub-product term AB_j ($0 \le j \le n-1$) in Eq.(8) can be computed in same way as follows.

$$\begin{aligned} & AB_j \\ &= \left(A_{n-1}x^{2(n-1)d} + A_{n-2}x^{2(n-2)d} + ... + A_1 x^{2d} + A_0\right)B_j \\ &= \left(\left(\left(\left(A_{n-1}B_j\right)x^{2d} + A_{n-2}B_j\right)x^{2d} + A_{n-3}B_j\right)x^{2d} + ...\right)x^{2d} + A_0 B_j \end{aligned} \tag{9}$$

Subsequently, the computation form of C_{ij} can be simplified by the Karatsuba algorithm as follows. Firstly, both A_i and B_j are subdivided into two parts,

$$\begin{aligned} A_i &= A_{iL} + A_{iH}x^d, \\ B_j &= B_{jL} + B_{jH}x^d, \end{aligned} \tag{10}$$

where $A_{iL} = \sum_{k=0}^{d-1} a_{2id+k}x^k$, $A_{iH} = \sum_{k=0}^{d-1} a_{2id+d+k}x^k$, $B_{jL} = \sum_{k=0}^{d-1} b_{2jd+k}x^k$, and $B_{jH} = \sum_{k=0}^{d-1} b_{2jd+d+k}x^k$.

$$\begin{aligned} & C_{ij} \\ &= A_i B_j \\ &= \left(A_{iL} + A_{iH}x^d\right)\left(B_{jL} + B_{jH}x^d\right) \\ &= A_{iL}B_{jL} + \left(\left(A_{iL} + A_{iH}\right)\left(B_{jL} + B_{jH}\right) + A_{iL}B_{jL} + A_{iH}B_{jH}\right)x^d + A_{iH}B_{jH}x^{2d} \\ &= C_{ij}^1 + C_{ij}^2 x^d + C_{ij}^3 x^{2d}, \end{aligned} \tag{11}$$

where $C_{ij}^1 = A_{iL}B_{jL}$, $C_{ij}^2 = \left(A_{iL} + A_{iH}\right)\left(B_{jL} + B_{jH}\right) + C_{ij}^1 + C_{ij}^3$, and $C_{ij}^3 = A_{iH}B_{jH}$.

One systolic bit-parallel d×d multiplier is then designed for implementing Eq.(11). Let $\overline{A}$ and $\overline{B}$ be d-bit elements and $\overline{C}$ be their 2d-bit product. Therefore, this systolic bit-parallel d×d multiplier could be designed according to the following equations.

$$\overline{A} = \sum_{i=0}^{d-1} \overline{a_i}x^i, \overline{B} = \sum_{i=0}^{d-1} \overline{b_i}x^i, \quad \overline{C} = \sum_{i=0}^{2d-1} \overline{c_i}x^i \text{ , and } \overline{C} = \overline{A} \times \overline{B} = \sum_{i=0}^{d-1} \overline{a_i}\overline{B}x^i. \tag{12}$$

Let $\overline{B}^i = \overline{B}x^i$, thus

$$\overline{B}^i = \overline{B}^{i-1}x \text{ .} \tag{13}$$

The representation of $\overline{B}^i$ is $\overline{B}^i = \sum_{k=0}^{d-1} \overline{b_k}^i x^k$. As a result, $\overline{B}^i$ can be obtained from $\overline{B}^{i-1}$ as follows.

$$\overline{B}^i = \overline{B}^{i-1}x = \left(\sum_{k=0}^{d-1} \overline{b_k}^{i-1}x^k\right)x = \sum_{k=0}^{d-1} \overline{b_k}^{i-1}x^{k+1} = \sum_{k=1}^{d} \overline{b_k}^i x^k, \tag{14}$$

where $\overline{b_k}^i = \overline{b_{k-1}}^{i-1}$ for $1 \le k \le d$.

Based on Eqs. (12) and (14), the semi-systolic d×d bit-parallel PB multiplier is shown in Fig.2. The circuit for realizing U cell is drawn in Fig.3. By employing d×d bit-parallel PB multiplier in Fig.3, the proposed systolic 2d×2d bit-parallel PB multiplier based on Eq.(11) is shown in Fig.4. According to Eqs. (8) and (9), the proposed systolic digit-serial PB multiplier using 2d×2d bit-parallel PB multiplier is depicted in Fig.5. The Feedback barrel shifter in Fig.5 performs multiplication-summation-shift operation such as $Hx^{2d} + K$ and is shown in Fig.6. The Mod Function in Fig.6 has 2d inputs and obtains m outputs after carrying out the function: $\left(c_m x^m + c_{m+1}x^{m+1} + ... + c_{m+2d-1}x^{m+2d-1}\right) \mod P$. The Mod Function depends on P. Based on the multiplier in Fig.5, the digit-serial PB multiplication algorithm using Karatsuba algorithm is illustrated in the Algorithm-DSMK. The multiplication operation A[i]×B[j] is carried out on the systolic 2d×2d PB multiplier in Fig.4. The Shift

function performs the operation $\times x^{2d}$ appeared in Eqs. (8) and (9), and is applied to the Feedback barrel shifter in Fig.6. The calculation of mod P is carried out by the Mod Function.

Algorithm-DSMK

```
C1=0;
For j=n-1 To 0 Do
  Begin
    C2=0;
    For i=n-1 To 0 Do
      Begin
        C2=(Shift(C2)+ A[i]*B[j]) mod P;
      End;
    C1=(Shift(C1)+C2) mod P;
  End;
```

The computation of Mod Function is dependent on the module, P. Inputs of the Mod Function are weighted with the coefficients: $\sum_{i=0}^{2d-1} t_i x^{m+i}$, and then the Mod Function performs $\left(\sum_{i=0}^{2d-1} t_i x^{m+i}\right) \bmod P$. Three popular irreducible polynomials for P: all one polynomial (AOP), trinomial, and pentanomial, will be discussed in the following subsection.

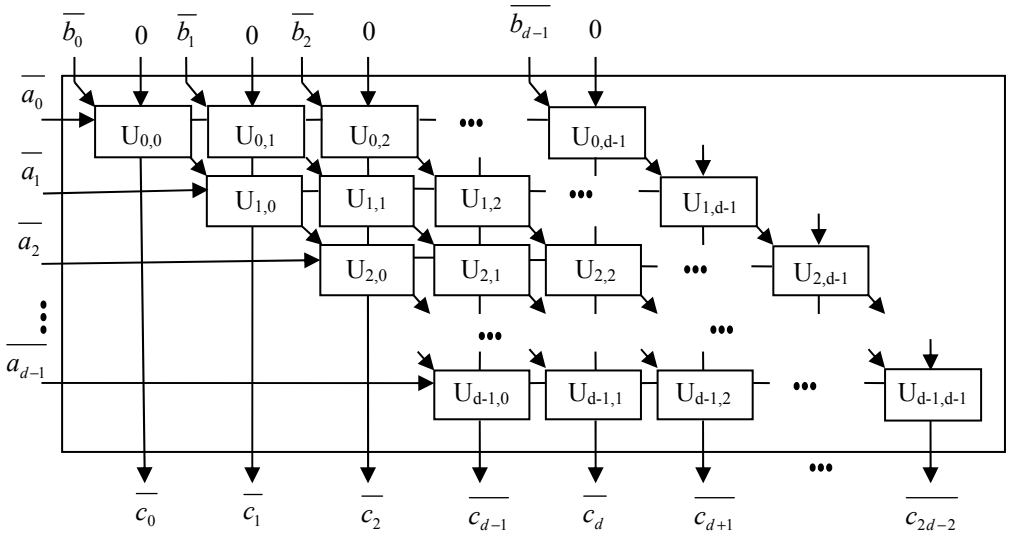
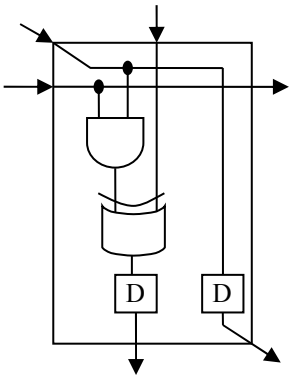


Fig.2. The proposed semi-systolic d×d bit-parallel PB multiplier.



Note: D represents D flip-flop

Fig.3. The detailed circuit of U cell

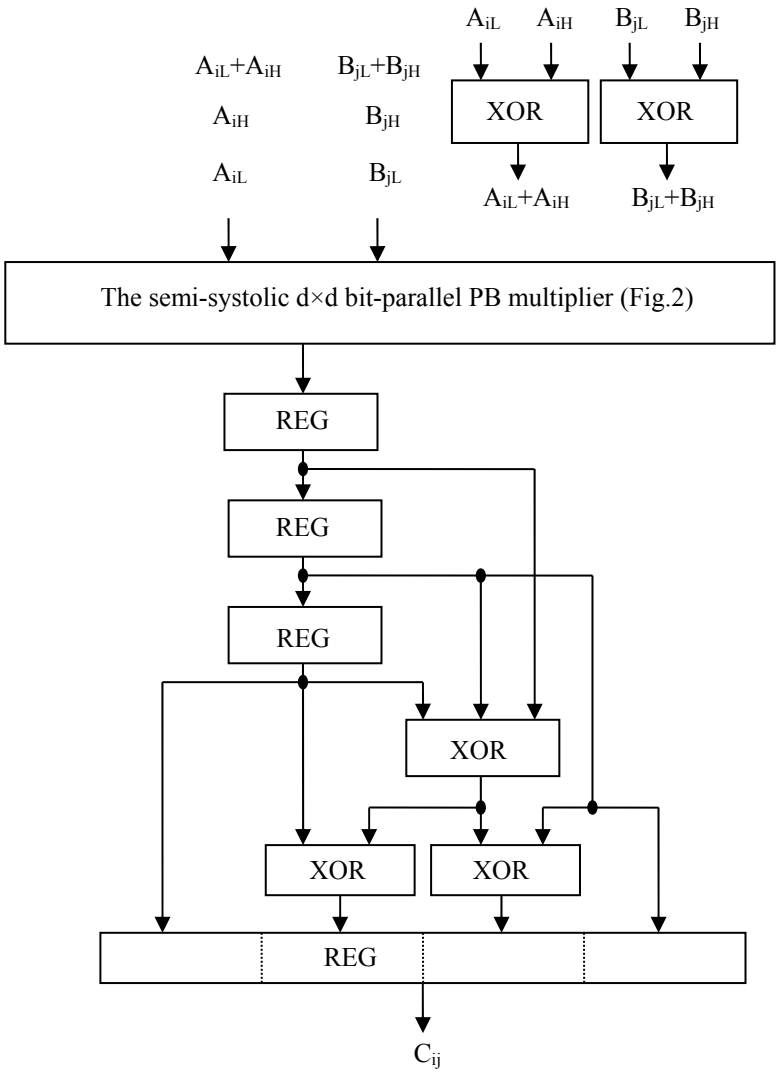


Fig.4. The proposed systolic 2d×2d PB multiplier

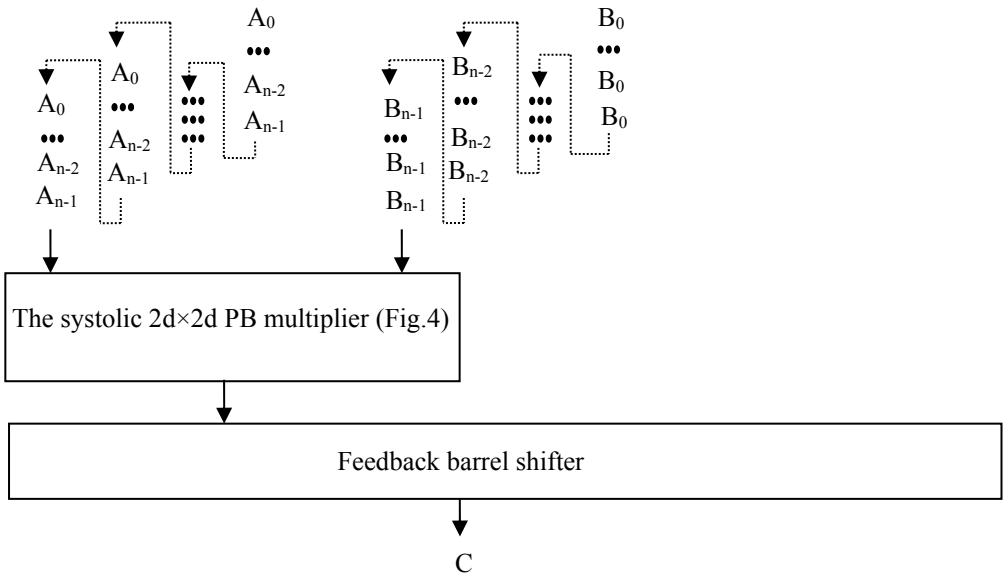


Fig.5. The proposed systolic digit-serial PB multiplier with each digit 2d bits.

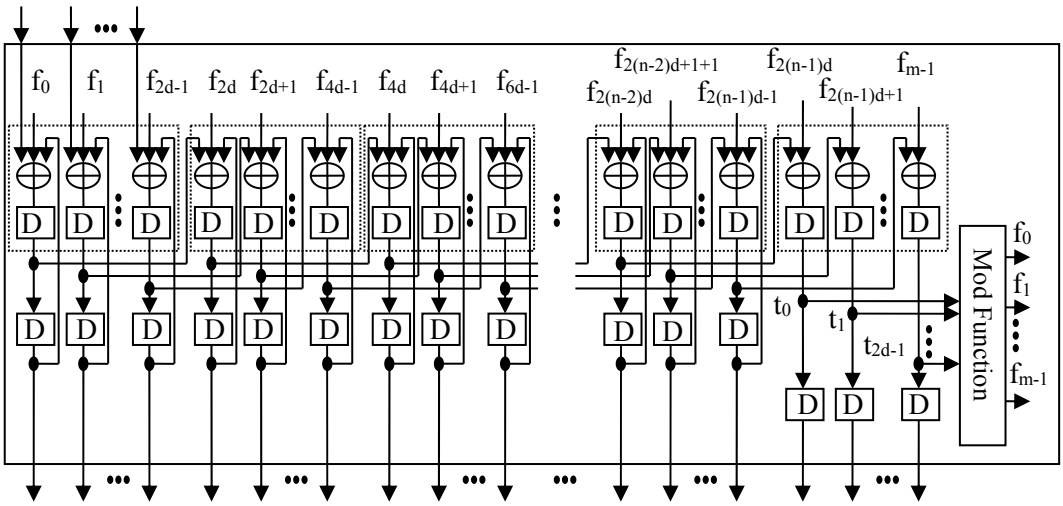


Fig.6. Feedback barrel shifter

3.2 The Mod Function for AOP, Trinomial, and Pentanomial Classes

Three cases are discussed separately.

(1) AOP

If P has the form: $P = 1 + x + x^2 + ... + x^{m-1} + x^m$, it is termed all one polynomial. In this case, one has the following properties:

(a) $x^m = 1 + x + x^2 + ... + x^{m-1}$,

(b) $x^{m+1} = 1$.

Based on the above properties, the outputs (f_i for $0 \le i \le m-1$) and the inputs (t_i for $0 \le i \le 2d-1$) will hold the following relations:

$$f_i = \begin{cases} t_0 + t_{i+1} & \text{if } 0 \le i \le 2d-2, \\ t_0 & \text{if } 2d-1 \le i \le m-1. \end{cases} \quad (15)$$

The circuit for realizing the above equation is drawn in Fig.7. It requires 2d-1 XOR gates.

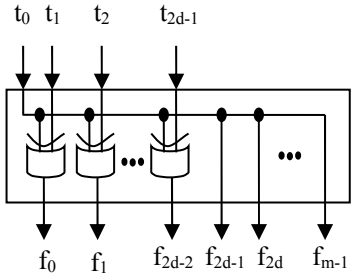


Fig. 7. Circuit of the Mod Function for AOP

(2) Trinomial

The P with the form: $P(x) = x^m + x^k + 1$ ($1 \le k \le m-1$) is called trinomial. In this case, one has the following properties:

(a) $x^m = x^k + 1$,

(b) $x^{m+i} = x^{k+i} + x^i$ for $0 \le i \le 2d-1$.

Depending on value of k, the following cases will be discussed.

(i) $k < 2d$

$$f_i = \begin{cases} t_i & \text{for } 0 \le i \le k-1, \\ t_i + t_{i-k} & \text{for } k \le i \le 2d-1, \\ t_{i-k} & \text{for } 2d \le i \le k+2d-1, \\ 0 & \text{for } k+2d \le i \le m-1. \end{cases} \quad \textbf{(16)}$$

The circuit for the Mod Function in this case is shown in Fig.8(a).

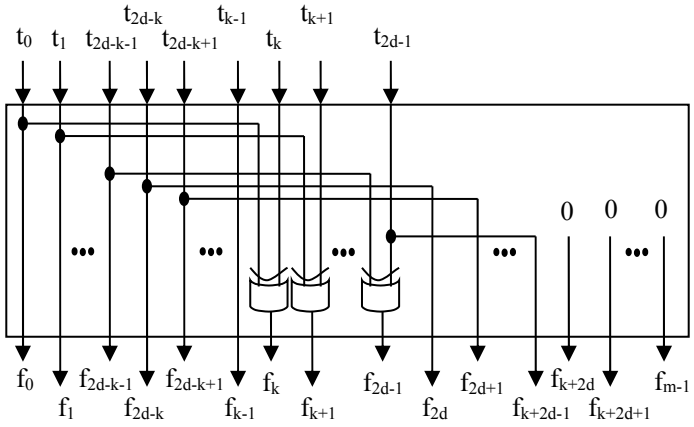


Fig. 8(a). Circuit of the Mod Function for trinomial with k<2d

(ii) $2d \le k \le m-2d-1$

$$f_i = \begin{cases} t_i & \text{for } 0 \le i \le 2d-1, \\ 0 & \text{for } 2d \le i \le k-1, \\ t_{i-k} & \text{for } k \le i \le k+2d-1, \\ 0 & \text{for } k+2d \le i \le m-1. \end{cases} \quad \textbf{(17)}$$

The circuit of the Mod Function in this case is depicted in Fig.8(b).

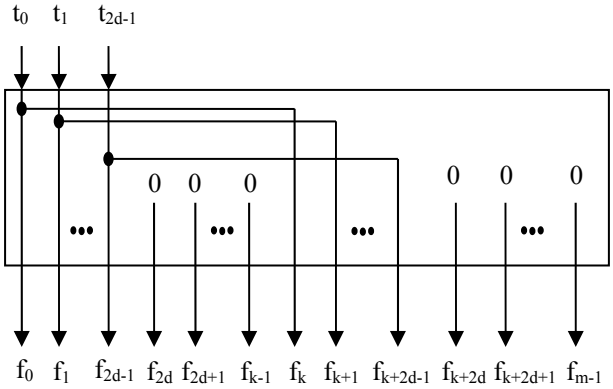


Fig. 8(b). Circuit of Mod Function for trinomial with $2d \le k \le m-2d-1$

(iii) $m-2d \le k < m-1$

Let $h = m-1-k$, two sub-cases will be further discussed.

(a) $x^{m+i} = x^{k+i} + x^i$ for $0 \le i \le h$

$f_i = t_i$ for $0 \le i \le h$, and $f_{k+i} = t_i$ for $0 \le i \le h$.

(b) $x^{m+i} = x^{k+1} + x^i$ for $h+1 \le i \le 2d-1$, and $x^{m+i+h+1} = x^{k+i+h+1} + x^{i+h+1}$ for $0 \le i \le 2d-1-h-1$.

$$f_i = \begin{cases} t_i + t_{i+m-k} & \text{for } 0 \le i \le 2d + k - m - 1, \\ t_i & \text{for } 2d + k - m \le i \le 2d - 1, \\ 0 & \text{for } 2d \le i \le k - 1, \\ t_{i+m-2k} + t_{i-k} & \text{for } k \le i \le 2d + 2k - m - 1, \\ t_{i-k} & \text{for } 2d + 2k - m \le i \le m - 1. \end{cases} \quad (18)$$

The circuit of the Mod Function in case (iii) is shown in Fig.8(c).

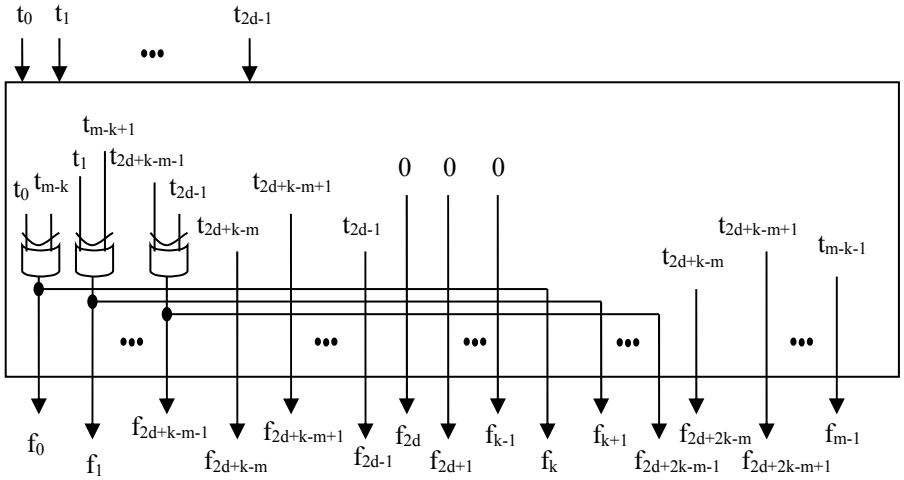


Fig. 8(c). Circuit of the Mod Function for trinomial with $m - 2d \le k < m - 1$.

(3) Pentanomial

If $P = x^m + x^{k3} + x^{k2} + x^{k1} + 1$ ($k3 > k2 > k1 > 0$), it is called pentanomial. In this case, the property: $x^m = x^{k3} + x^{k2} + x^{k1} + 1$ is held. For saving space complexity, proper selection of *k3* with *m-k3<2d* is employed in this study. The relations between outputs and inputs of the Mod Function are described as follows.

$$f_i = \begin{cases} t_i & \text{if } i < 2d \text{ and } i < k1, \\ t_i + t_{i-k1} & \text{if } i < 2d \text{ and } k1 \le i < k1 + 2d, \text{ and } i < k2 \\ t_i + t_{i-k1} + t_{i-k2} & \text{if } i < 2d, k1 \le i < k1 + 2d, k2 \le i < k2 + 2d, \text{ and } i < k3 \\ t_i + t_{i-k1} + t_{i-k2} + t_{i-k3} & \text{if } i < 2d, k1 \le i < k1 + 2d, \text{ and } k2 \le i < k2 + 2d, k3 \le i < k3 + 2d, \\ t_{i-k1} & \text{if } i \ge 2d, k1 \le i < k1 + 2d, \text{ and } i < k2, \\ t_{i-k1} + t_{i-k2} & \text{if } i \ge 2d, k1 \le i < k1 + 2d, k2 \le i < k2 + 2d, \text{ and } i < k3, \\ t_{i-k1} + t_{i-k2} + t_{i-k3} & \text{if } i \ge 2d, k1 \le i < k1 + 2d, k2 \le i < k2 + 2d, \text{ and } k3 \le i < k3 + 2d, \\ t_{i-k2} & \text{if } i \ge 2d, i > k1, k2 \le i < k2 + 2d, \text{ and } i < k3, \\ t_{i-k2} + t_{i-k3} & \text{if } i \ge 2d, i > k1, k2 \le i < k2 + 2d, \text{ and } k3 \le i < k3 + 2d, \\ t_{i-k3} & \text{if } i \ge 2d, i > k1, i > k2, \text{ and } k3 \le i < k3 + 2d, \\ 0 & \text{if others} \end{cases} \quad (19)$$

To illustrate the circuit of the Mod Function in this case, an example with the proper $P=x^{16}+x^5+x^3+x+1$ and 2d=4 is selected. The circuit for this example is shown in Fig. 9.

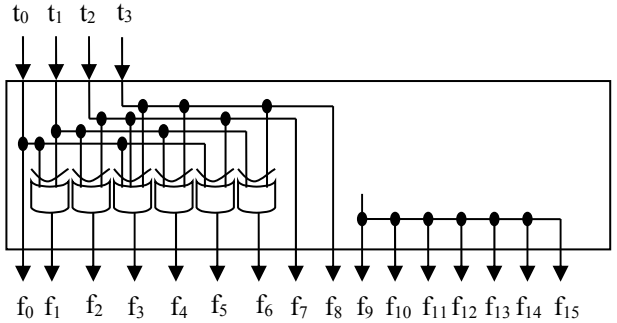


Fig. 9. Circuit of the Mod Function for the pentanomial $P=x^{16}+x^5+x^3+x+1$ and 2d=4

4 **Comparison Results**

For comparison with other similar studies, the following transistor-count assumptions were made for VLSI implementation: w-input AND gate, D flip-flop, 2-to-1 multiplexer, and 2-input XOR gate are composed of 2w+2, 18, 6, and 12 transistors, respectively [45]. The w-input XOR gate (w>2) is assumed to be realized by a binary 2-input XOR tree.

Comparing results of our proposed multiplier with similar studies for space complexity are listed in Table 1. Comparisons for the m values suggested by NIST for space complexity are depicted in Table 2. Results show that our proposed digit-serial PB multiplier can save 90% space complexity in average while comparing with the Talapatra multiplier in [19]. Comparison of time complexity is illustrated in Table 3. The proposed multiplier is simulated on Stratix II F1508 ASIC Prototyping Board which uses Altera EP2S180F1508 FPGA chip. The simulated results are listed in Table 4. Due to the limited resource of EP2S180F1508C5 chip, only cases of m=163 and m=233 have been completed, other cases are failed to compile. The simulated results show that our proposed multiplier saves about 58% space complexity but has same time complexity. Fig.10 shows the numbers of consumed ALUTs and pins of EP2S180F1508C5 for Talapatra multiplier [19] and our proposed multiplier. The total pin number of EP2S180F1508C5 is 1171 and the proposed multiplier saves about 20% pins as compared to the Talapatra multiplier [19]. Figure 11 shows that our proposed multiplier has lower space complexity than Talapatra multiplier [19] as m is ranged from 55 up to 955 for digit size being16 bits.

Table 1. Comparisons on space complexities of various systolic digit-serial PB multipliers with digit size=2d.

Multipliers	Kim *et al.* [17]	Talapatra *et al.* (Figs.4 & 5 in [19])	The proposed multiplier (Fig.5)
One-dimensional array	Yes	Yes	No
Generating polynomial	General polynomial	Special polynomial:AOP	Special polynomials: AOP, trinomial, pentanomial
Number of digit cells	n	n	1
Array type of digit cell	Systolic	Systolic	Semi-systolic
Number of bit cells in digit cell	2d×2d	2d×2d	d×d
Space complexity of digit cells	$8nd^2$ AND_2	$4nd^2$ AND_2	d^2 AND_2
	$8nd^2$ XOR_2	$4nd^2$ XOR_2	(d^2+4d) XOR_2+ d XOR_3
	$12nd^2$ D F-Fs	$(8nd^2+n)$ D F-Fs	$(2d^2+7d)$ D F-Fs
		4nd MUX_2	
Space complexity	$(8nd^2+2nd)$ AND_2	$4nd^2$ AND_2	d^2 AND_2
	$8nd^2$ XOR_2	$4nd^2$ XOR_2	(d^2+4d) XOR_2+ $(m+d)$ XOR_3
	$12nd^2$ D F-Fs	$(8nd^2+n)$ D F-Fs	$(2m+2d^2+7d)$ D F-Fs
	4nd MUX_2	4nd MUX_2	1 Mod Function: AOP: (2d-1) XOR_2 Trinomial: $\leq 2d$ XOR_2 Pentanomial: $\leq (d-1)^2$ XOR_2
Transistor count	$376nd^2+40nd$	$224nd^2+24nd+18n$	AOP: $56d^2+60m+222d-12$ Trinomial: $56d^2+60m+222d$ Pentanomial: $68d^2+60m+174d+12$

Table 2. Space complexity comparisons for the m values suggested by NIST.

Multipliers		Talapatra et al. (Figs.4 & 5 in [19])	The proposed multiplier (Fig.5)	Comparing results
m	2d	Transistors (a)	Transistors (b)	(b)/(a)
163	8	77658	11576	15%
	16	160006	15536	10%
	32	346476	29984	9%
233	8	110940	15764	15%
	16	218190	19340	9%
	32	461968	31868	7%
283	8	133128	18776	15%
	16	261828	22736	9%
	32	519714	37184	8%
409	8	192296	26324	14%
	16	378196	29900	8%
	32	750698	42428	6%
571	8	266256	36056	15%
	16	523656	40016	8%
	32	1039428	54464	6%
Average				10%
Saved space complexity by the proposed multiplier as compared to Talapatra multiplier				90%

Table 3. Comparisons on time complexity of various systolic digit-serial PB multipliers with digit size=2d.

Multipliers	Kim *et al.* [17]	Talapatra *et al*. (Figs.4 & 5 in [19])	The proposed multiplier (Fig.5)
Latency (One product)	3n	2n-1	n
Cell delay	$T_A+T_X+T_L$	$T_A+T_X+T_L$	$T_A+T_X+T_L$
Digit cell delay	$2d\times(T_M+T_A+T_X+T_L)$	$2d\times(T_M+T_A+T_X+T_L)$	$(d+2)\times(T_A+T_X+T_L)+2T_X+T_L$
Latency (one multiplication)	(n^2+3n-1)	(n^2+2n-2)	n^2

Note: T_A, T_X, T_M, T_L denote the propagation delays of a 2-input AND gate, a 2-input XOR gate, a 2-to-1 Multiplexer, and a 1-bit Latch, respectively.

Table 4. Simulation results for space complexity

Multipliers		Talapatra *et al.* (Figs.4 & 5 in [19])			The proposed multiplier (Fig.5)		
m	2d	ALUTs	pins	t_{pd}/f_{max}	ALUTs	pins	t_{pd}/f_{max}
163	32	26569	816	25.229 ns/ 263.57MHz	11307	655	24.249 ns/ 263.57 MHz
233	32	54290	1166	30.822 ns/ 141.60 MHz	22716	935	30.812 ns/ 141.60 MHz
Saved space complexity as compared to [19] in average							58.5%
Saved pins as compared to [19] in average							20%
Saved time complexity as compared to [19] in average							1.5%

Note: ALUTs: Adaptive look-up tables in Altera chips. t_{pd}: pin-to-pin delay.

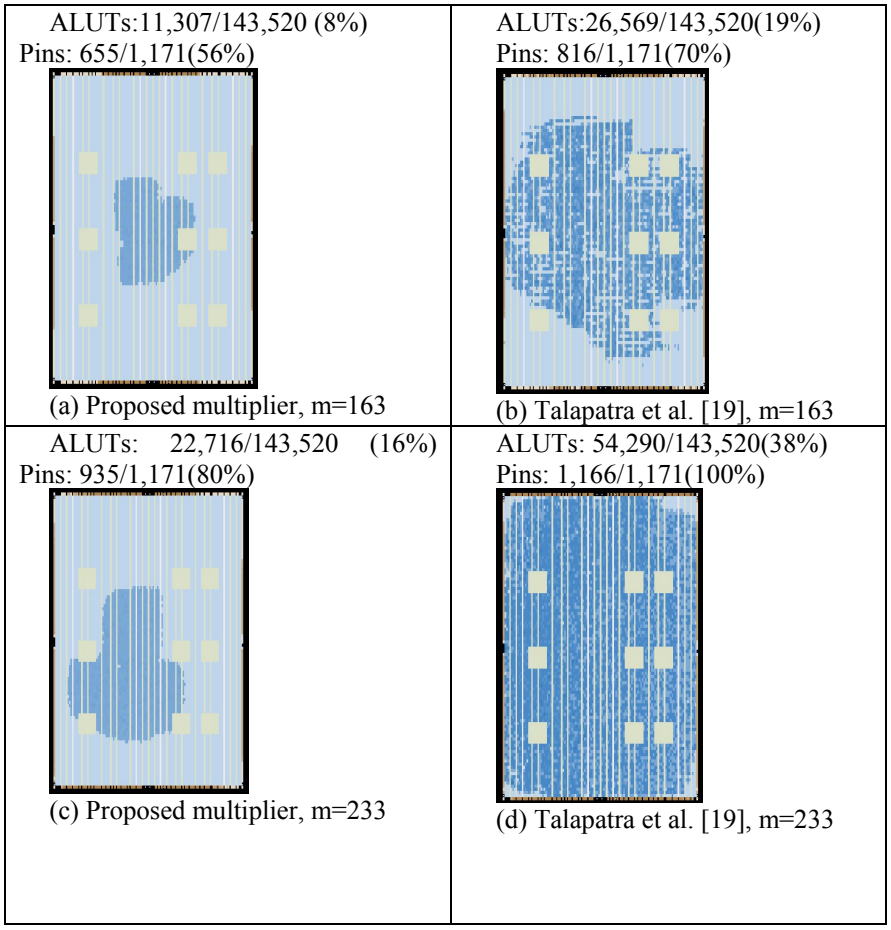


Fig. 10. Used ALUTs and pins of EP2S180F1508C5 for various multipliers.

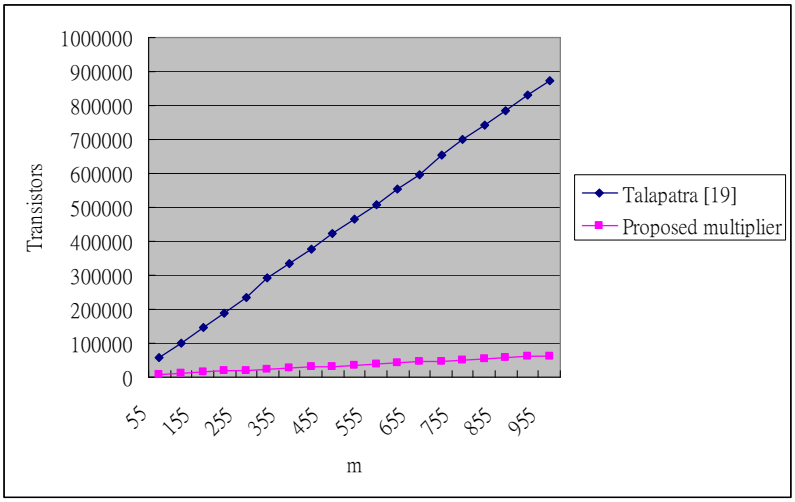


Fig.11. Space complexity comparisons for various m values with digit size=16 bits

5 Conclusions

Traditional digit-serial polynomial basis multipliers employ one-dimensional systolic array architecture, such as Guo-Wang multiplier [16], Kim multiplier [17], and Talapatra multiplier [19]. The proposed digit-serial polynomial basis multiplier uses only one digit cell to realize the multiplier. Furthermore, the Karatsuba algorithm is employed for reducing a digit cell size from 2d×2d bits to d×d bits. Analysis results show that our proposed digit-serial polynomial basis multiplier saves 90% space complexity as compared to existing similar studies.

Acknowledgment

The authors would like to thank anonymous referees and the editor for their carefully reading the paper and for their great help in improving the paper. The authors would also like to thank the National Science Council of the Republic of China, Taiwan, for partly financial supporting this research under Contract No. NSC 102-2221-E-231-008.

Appendix: Abbreviations and Symbols

Abbreviations/Symbols	Definitions
DB	Dual Basis
PB	Polynomial basis
NB	Normal basis
ECC	Elliptic curve cryptosystem
GF	Galois Field/Finite Field
mod	Modulo, find a remainder of a number
NIST	National Institute of Standards and Technology, USA
GF(2^m)	Extension binary field of GF(2) with m-bit strings.
ASIC	Application-specific integrated circuit
VLSI	Very-large-scale integration
FPGA	Field-programmable gate array
P(x)	An irreducible polynomial of degree m over GF(2)
A(x), *B(x)*, *C(x)*	Elements over GF(2^m) generated by *P(x)*
A_L (B_L)	Denotes low half part of A (B) with degree lower than m/2, $A_L = \sum_{i=0}^{m/2-1} a_i x^i$ ($B_L = \sum_{i=0}^{m/2-1} b_i x^i$)
A_H (B_H)	Denotes high half part of A (B) with degree larger than or equal to m/2, $A_H = \sum_{i=m/2}^{m-1} a_i x^{i-m/2}$ ($B_H = \sum_{i=m/2}^{m-1} b_i x^i$)
C_{LL}	The product of A_L and B_L
C_{LH}	Denotes $(A_L + A_H)(B_L + B_H) + A_L B_L + A_H B_H$
C_{HH}	The product of A_H and B_H
Reduction(H)	Denotes H mod P
REG	Register
AOP	All one polynomial, for example, $P = \sum_{i=0}^{m} x^i$
Trinomial	The polynomial has the form: $P = x^m + x^k + 1, 1 \le k \le m-1$
Pentanomial	The polynomial has the form: $P = x^m + x^{k3} + x^{k2} + x^{k1} + 1, 1 \le k1 < k2 < k3 \le m-1$
d	Each digit with 2×d bits
n	Each element with n digits
A_i (B_j)	The (i+1)th digit of A with 2d bits
C_{ij}	The product of A_i and B_j
A_{iL} (B_{jL})	The low half part of A_i (B_j), $A_{iL} = \sum_{k=0}^{d-1} a_{2id+k} x^k$ ($B_{jL} = \sum_{k=0}^{d-1} b_{2jd+k} x^k$)
A_{iH} (B_{jH})	The high half part of A_i (B_j), $A_{iH} = \sum_{k=0}^{d-1} a_{2id+d+k} x^k$ ($B_{jH} = \sum_{k=0}^{d-1} b_{2jd+d+k} x^k$)
C_{ij}^1	The product of A_{iL} and B_{jL}
C_{ij}^2	Denotes $(A_{iL} + A_{iH})(B_{jL} + B_{jH}) + C_{ij}^1 + C_{ij}^3$
C_{ij}^3	The product of A_{iH} and B_{jH}
$\overline{A}, \overline{B}$	Any element with d bits
$\overline{C}$	The product of $\overline{A}$ and $\overline{B}$
T_A	The delay of a 2-input AND gate
T_X	The delay of a 2-input XOR gate

T_L	The delay of 1-bit Latch
T_M	The delay of 2-to-1 multiplexer

References

[1] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Pub. Co., Amsterdam, 1977.

[2] R. E. Blahut, *Fast algorithms for digital signal processing*, Addison-Wesley Longman Publishing Co., Boston, 1985.

[3] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and Their Applications*, Cambridge Univ. Press, New York, 1994.

[4] V.S. Miller, "Use of elliptic curves in cryptography," in *Proc. of Crypto 85*, LNCS 218, Springer, pp.417-426, 1986.

[5] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, Vol. 48, pp. 203-209, 1987.

[6] N. Koblitz, "A family of Jacobians suitable for discrete log cryptosystems," Advances in Cryptology, *Proc. Crypto'88*, S. Goldwasser, ed., pp.94-99, 1988.

[7] R. Dutta, R. Barua, and P. Sarkar, *Pairing-based Cryptographic Protocols: A Survey*, Cryptology ePrint Archive, Report 064/2004, 2004.

[8] T. C. Bartee and D. J. Schneider, "Computation with finite fields," *Information and Computing*, Vol. 6, pp. 79-98, 1963.

[9] E. D. Mastrovito, "VLSI architectures for multiplication over finite field GF(2^m)," in *Proc. Sixth Int'l Conf. on Applied Algebra, Algebraic Algorithms, and Error-Correcting Codes, AAECC-6*, T. Mora, ed., Rome, pp. 297-309, 1988.

[10] Ç. K. Koç and B. Sunar, "Low-complexity bit-parallel canonical and normal basis multipliers for a class of finite fields," *IEEE Trans. Computers*, Vol. 47, No. 3, pp. 353-356, 1998.

[11] T. Itoh and S. Tsujii, "Structure of parallel multipliers for a class of fields GF(2^m)," *Information and Computation*, Vol. 83, pp.21-40, 1989.

[12] C. Y. Lee, E. H. Lu, and J. Y. Lee, "Bit-parallel systolic multipliers for GF(2^m) fields defined by all-one and equally-spaced polynomials," *IEEE Trans. Computers*, Vol. 50, No. 5, pp. 385-393, 2001.

[13] C. Paar, P. Fleischmann, and P. Roelse, "Efficient multiplier architectures for Galois Fields GF(2^{4n})," *IEEE Trans. Computers*, Vol. 47, No. 2, pp. 162-170, 1998.

[14] H. Wu, "Bit-parallel finite field multiplier and squarer using polynomial basis," *IEEE Trans. Computers*, Vol. 51, No. 7, pp.750-758, 2002.

[15] H. Fan, M.A. Hasan, "A new approach to subquadratic space complexity parallel multipliers for extended binary fields," *IEEE Trans. Computers*, Vol. 56, No. 2, pp.224-233, 2007.

[16] J.-H. Guo and C.-L. Wang, "Digit-serial systolic multiplier for finite fields GF(2^m)," *IEE Proc. Comput. Digit. Tech.*, Vol. 145, No. 2, pp. 143-148, 1998.

[17] C.H. Kim, C.P. Hong, and S. Kwon, "A digit-serial multiplier for finite field GF(2^m)," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, Vol. 13, No. 4, pp. 476-483, 2005.

[18] S. Kumar, T. Wollinger, and C. Paar, "Optimum digit-serial GF(2^m) multipliers for curve-based cryptography," *IEEE Trans. Computers*, Vol. 55, No. 10, pp. 1306-1311, 2006.

[19] S. Talapatra, H. Rahaman, and J. Mathew, "Low complexity digit serial systolic Montgomery multipliers for special class of GF(2^m)," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, Vol. 18, No. 5, pp. 847-852, 2010.

[20] G. Zhou, H. Michalik, L. Hinsenkamp, "Complexity analysis and efficient implementations of bit parallel finite field multipliers based on Karatsuba-Ofman algorithm on FPGAs," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, Vol. 18, No. 7, pp. 1057-1066, 2010.

[21] X.-N. Xie, G.-L. Chen, Y. Li, "Novel bit-parallel multiplier for GF(2^m) defined by all-one polynomial using generalized Karatsuba algorithm, " *Information Processing Letters*, Vol. 114, pp. 140-146, 2014.

[22] C.-Y. Lee, W.-Y. Lee, C.-T. Wu, C.-C. Yang, "Scalable systolic multiplier over binary extension fields based on two-level Karatsuba decompoition," *International Journal of Computer, Systems and Control Engineering*, Vol. 8, No. 5, pp. 737-743, 2014.

[23] H. Wu, M. A. Hasan, and I. F. Blake, "New low-complexity bit-parallel finite field multipliers using weakly dual bases," *IEEE Trans. Computers*, Vol. 47, No. 118, pp. 1223-1234, 1998.

[24] S. T. J. Fenn, M. Benaissa, and D. Taylor, "GF(2^m) multiplication and division over the dual basis," *IEEE Trans. Computers*, Vol. 45, No. 3, pp. 319-327, 1996.

[25] M. Wang and I.F. Blake, "Bit serial multiplication in finite fields," *SIAM J. Disc. Math.*, Vol. 3, No. 1, pp. 140-148, 1990.

[26] E.R. Berlekamp, "Bit-serial Reed-Solomon encoder," *IEEE Trans. Inform. Theory*, Vol. IT-28, pp. 869-874, 1982.

[27] C.Y. Lee and C.W. Chiou, "Efficient design of low-complexity bit-parallel systolic Hankel multipliers to implement multiplication in normal and dual bases of GF(2^m)," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science*, Vol. E88-A, No. 11, pp. 3169-3179, 2005.

[28] J. L. Massey and J. K. Omura, *Computational Method and Apparatus for Finite Field Arithmetic*, U.S. Patent Number 4,587,627, 1986.

[29] C. C. Wang, T. K. Truong, H. M. Shao, L. J. Deutsch, J. K. Omura, and I. S. Reed, "VLSI architectures for computing multiplications and inverses in GF(2^m)," *IEEE Trans. Computers*, Vol. C-34, No. 8, pp. 709-717, 1985.

[30] A. Reyhani-Masoleh, "Efficient algorithms and architectures for field multiplication using Gaussian normal bases," *IEEE Trans. Computers*, Vol. 55, No. 1, pp. 34-47, 2006.

[31] C.W. Chiou, C.Y. Lee, "Multiplexer-based double-exponentiation for normal basis of GF (2^m)," *Computers & Security*, Vol. 24, No. 1, pp. 83-86, 2005.

[32] G.B. Agnew, R.C. Mullin, I.M. Onyszchuk, S.A. Vanstone, "An implementation for a fast public-key cryptosystem," *Journal of Cryptology*, Vol. 3, pp. 63-79, 1991.

[33] M.A. Hasan, M.Z. Wang, V.K. Bhargava, "A modified Massey-Omura parallel multiplier for a class of finite fields," *IEEE Trans. Computers*, Vol. 42, No. 10, pp. 1278-1280, 1993.

[34] S. Kwon, "A low complexity and a low latency bit parallel systolic multiplier over GF(2^m) using an optimal normal basis of type II," *Proc. of the 16th IEEE Symposium on Computer Arithmetic*, Santiago de Compostela, Spain, pp. 196-202, 2003.

[35] H. Fan, M.A. Hasan, "Subquadratic computational complexity schemes for extended binary field multiplication using optimal normal bases," *IEEE Trans. Computers*, Vol. 56, No. 10, pp. 1435-1437, 2007.

[36] A. Karatsuba and Y. Ofman, "Multiplication of many-digital numbers by automatic computers," in *Proceedings of the USSR Academy of Sciences*, pp. 293-294, 1962.

[37] B. Sunar, "A generalized method for constructing subquadratic complexity GF(2^k) multipliers," *IEEE Trans. Computers*, Vol. 53, No. 9, pp. 1097-1105, 2004.

[38] F. Rodriguez-Henriquez and C. K. Koc, "On Fully Parallel Karatsuba Multipliers for GF(2^m)," *Proc. of the International Conference on Computer Science and Technology, CST 2003*, pp. 405-410, Acta Press, Cancun, Mexico, 2003.

[39] H. Fan and M.A. Hasan, "Alternative to the Karatsuba algorithm for software implementations of GF(2^n) multiplications," *IET Information Security*, Vol. 3, No. 2, pp. 60-65, 2009.

[40] Y. Li, G.-L. Chen, J.-H. Li, "Speeding of bit-parallel Karatsuba multiplier in GF(2^m) generated by trinomials," *Information Processing Letters*, Vol. 111, No. 8, pp. 390-394, 2011.

[41] J.-L. Beuchat, J. Detrey, N. Estibals, E. Okamoto, F. Rodríguez-Henríquez, "Hardware accelerator for the Tate pairing in characteristic three based on Karatsuba-Ofman multipliers," in *Proc. Workshop on Cryptographic Hardware and Embedded Systems-CHES 2009*, Lausanne, Switzerland, September 6-9, 2009, LNCS 5747, pp. 225-239, 2009.

[42] S. Ghosh, D. Roychowdhury, A. Das, "High speed cryptoprocessor for η_T pairing on 128-bit secure supersingular elliptic curves over characteristic two fields," in *Proc. Workshop on Cryptographic Hardware and Embedded Systems-CHES 2011*, Nara, Japan, Sep. 28~Oct. 1, 2011, LNCS 6917, Springer-Verlag, pp. 442-458, 2011.

[43] M. Morales-Sandoval, C. Feregrino-Uribe, and P. Kitsos, "Bit-serial and digit-serial GF(2^m) Montgomery multipliers using linear feedback shift registers," *IET Computers & Digital Techniques Journal*, Vol. 5, No. 2, pp. 86-94, 2011.

[44] C.-C. Chen, C.-Y. Lee, E.-H. Lu, "Scalable and Systolic Montgomery Multipliers over GF(2^m)," *IEICE Trans. Fundamentals*, Vol. E91-A, No. 7, pp. 1763-1771, 2008.

[45] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A System Perspective*, Addison-Wesley, Boston, 1985.