Low complexity bit parallel multiplier for GF($2^m$) generated by equally-spaced trinomials

Haibin Shen a,*, Yier Jin a, b

a Institute of VLSI Design, Zhejiang University, Hangzhou, China
b Department of Electrical Engineering, Yale University, 10 Hillhouse Avenue, New Haven, CT 06520, USA

Received 20 February 2007; received in revised form 14 May 2007; accepted 3 January 2008
Available online 16 March 2008
Communicated by D. Pointcheval

Abstract

Based on the shifted polynomial basis (SPB), a high efficient bit-parallel multiplier for the field GF($2^m$) defined by an equally-spaced trinomial (EST) is proposed. The use of SPB significantly reduces time delay of the proposed multiplier and at the same time Karatsuba method is combined with SPB to decrease space complexity. As a result, with the same time complexity, approximately 3/4 gates of previous multipliers are used in the proposed multiplier.

© 2008 Published by Elsevier B.V.

Keywords: Shifted polynomial basis (SPB); Equally-spaced trinomial (EST); Karatsuba method; Bit-parallel multiplier; Cryptography

1. Introduction

Finite filed operations are used in many areas such as coding theory, computers algebra, combinatorial designs and cryptography [3,6,13]. Among these operations, multiplication is of the most importance because other complex operations such as exponentiation, division, etc. can be carried out through iterative multiplications. Based on the advanced design technology nowadays, more and more logic gates can be located on a single chip which makes the implementations of parallel architectures possible and reasonable. In order to improve the efficiency of cryptographic system and coding system, many bit-parallel multiplier architectures have been proposed recently to achieve high computation speed [2,5,8,10–12,14,15]. Also, various basis except for polynomial basis (PB), dual basis (DB), and normal basis (NB) are developed in order to further reduce the critical path of multipliers with even fewer logic gates. And certain types of irreducible polynomials are used to improve the performance of multipliers in which trinomial is one of the best choices [12].

Although different architectures can be evaluated from several points of view, time complexity and space complexity are often the two most important parameters. The former is defined as the elapsed time between input and output of the circuit implementing the multiplier, and it is usually expressed as the sum of $T_A$ (the delay of a two input AND gate) and $T_X$ (the delay of a two input XOR gate) with corresponding coefficients. The latter is weighed by the numbers of AND gates and XOR gates used in multiplier denoted as #AND
and #XOR. In the finite field generated by trinomials, the most efficient multiplier architecture nowadays contains \( m^2 \) AND gates and \( m^2 - 1 \) XOR gates with time delay of \( T_A + (1 + \lceil \log_2(m) \rceil) T_X \) [5]. If the trinomial is an equally-spaced trinomial (EST) in the form of \( f(x) = x^m + x^\frac{m}{2} + 1 \) (\( m \) is even), the best result is \( \text{TimeDelay} = T_A + (1 + \lceil \log_2(m-1) \rceil) T_X \). #AND = \( m^2 \), #XOR = \( m^2 - m \).

As mentioned above, although we can put much more logic gates in a single chip than ever before, the \( O(m^2) \) complexity of AND and XOR gates still costs considerable chip area. Many researchers are devoted to the reduction of multipliers’ space complexity without increasing their time complexity. M. Elia et al. [1] use Karatsuba–Ofman multiplication and achieve even lower space complexity but their method requires two more \( T_X \) delays.

In this paper, we propose a new bit-parallel multiplier for \( \text{GF}(2^m) \) defined by EST \( f(x) = x^m + x^\frac{m}{2} + 1 \) using shifted polynomial basis which can significantly reduce the critical path. Also, we use the well-known Karatsuba–Ofman multiplication [4] to decrease the space complexity of the proposed bit-parallel multiplier. Based on these two methods, an high efficient architecture is constructed. The space complexity of the proposed multiplier is about \( \frac{3}{4} \) of the previous result while the time complexity matches the best efficient multipliers ever known of \( T_A + (1 + \lceil \log_2(m-1) \rceil) T_X \). The irreducible EST in the form of \( x^m + x^\frac{m}{2} + 1 \) exist when \( m = 2 \times 3^i \) where \( i \) is a non-negative integer.

Although the number of irreducible EST is not that redundant, they can be used in source critical area, e.g., smartcard where field polynomials are often fixed for the sake of lowering chip area.

The rest of paper is organized as follows: Section 2 introduces the representation of shift polynomial basis (SPB). Based on this representation a new bit-parallel multiplier architecture is proposed in Section 3. Section 4 presents the comparison between the proposed multiplier and some others. Finally the conclusions are drawn in Section 5.

2. Shifted polynomial basis (SPB)

Shifted polynomial basis (SPB) is first introduced by H. Fan and Y. Dai [5] which is derived from polynomial basis (PB) by adding a shift variable into each field element in order to improve the efficiency of multiplier. In PB representation, each element of \( \text{GF}(2^m) \) is represented by a different binary polynomial of degree less than \( m \). More explicitly, a bit string \((a_{m-2}, \ldots, a_1, a_0)\) is taken to represent binary polynomial as

\[
a(x) = \sum_{i=0}^{m-1} a_i x^i = a_{m-1} x^{m-1} + a_{m-2} x^{m-2} + \ldots + a_1 x + a_0, \quad a_j \in \text{GF}(2).
\]

Here the set \( M = \{x^{m-1}, x^{m-2}, \ldots, x, 1\} \) represents a polynomial basis.

The addition of bit strings corresponds to addition of binary polynomials. Multiplication is defined in terms of an irreducible binary polynomial \( f(x) \) of degree \( m \), called the field polynomial for the representation. The product of two elements is simply the product of the corresponding polynomials, reduced modulo \( f(x) \).

Here goes the definition of SPB over \( \text{GF}(2^m) \) in \( \text{GF}(2) \).

**Definition 1.** (See [5].) Let \( v \) be an integer and the ordered set \( M = \{x^i \mid 0 \leq i \leq m-1\} \) be a polynomial basis of \( \text{GF}(2^m) \) over \( \text{GF}(2) \). The ordered set \( x^v M := \{x^i + v \mid 0 \leq i \leq m-1\} \) is called the shifted polynomial basis (SPB) with respect to \( M \).

In reality, let \( f(x) = x^m + x^k + 1 \) be an irreducible trinomial over \( \text{GF}(2) \), \( M = \{x^i \mid 0 \leq i \leq m-1\} \) be a PB and \( x^{-v} M := \{x^{i-v} \mid 0 \leq i \leq m-1\} \) be an SPB, where \( 0 \leq v \leq m-1 \) and \( x \) is a root of \( f(x) = 0 \). It has been proved that the best value of \( v \) is \( k \) or \( k-1 \) with which the multiplier has lowest complexities [5]. From now on, we denote that \( v \) equals to \( k \) and \( x^{-k} M := \{x^{i-k} \mid 0 \leq i \leq m-1\} \) is the SPB. A field element \( a(x) \) can be uniquely represented as \( a(x) = (a_{m-1}, a_{m-2}, \ldots, a_1, a_0) = x^{-k} \sum_{i=0}^{m-1} a_i x^i \) with respect to SPB.

It is easy to transform the elements between PB and SPB representations. Let \( d(x) = \sum_{i=0}^{m-1} d_i x^i \) and \( a(x) = \sum_{i=-v}^{m-1-v} a_{v+i} x^i \) be two elements represented in PB and SPB. The conversions between these two representations are showed by the following two formulæ:

\[
d(x) = \sum_{i=0}^{m-1} d_i x^i
\]

\[
= \sum_{i=0}^{m-1-v} d_i x^i + \sum_{i=m-v}^{m-1} d_i (x^{i+v-m} + x^{i-m})
\]

\[
= \left( \sum_{i=0}^{m-1-v} d_i x^i + \sum_{i=-v}^{v-1} d_{m+i} x^i \right) + \sum_{i=0}^{v-1} d_{m+i-v} x^i,
\]
\[ a(x) = \sum_{i=-v}^{m-1-v} a_{v+i} x^i = \sum_{i=0}^{m-1-v} a_{v+i} x^i + \sum_{i=-v}^{-1} a_{v+i} (x^{m+i} + x^{v+i}) \]
\[ = \left( \sum_{i=0}^{m-1-v} a_i x^i + \sum_{i=m-v}^{m-1} a_{v-m+i} x^i \right) + \sum_{i=0}^{v-1} a_i x^i. \]

According to the above formulae, the conversion between these two representations only needs \( v \) XOR gates and \( 1T_X \) delay with parallel computing.

Multiplication on SPB is the same as that on PB except that reduction step abides by two formulae:

\[ x^i = x^{k+i-m} + x^{i-m}, \quad \text{where} \quad m - v \leq i \leq 2m - 2 - 2v, \]
\[ x^i = x^{m+i} + x^{k+i}, \quad \text{where} \quad -2v \leq i \leq -(v + 1). \]

If the irreducible trinomial is \( f(x) = x^m + x^{\frac{m}{2}} + 1 \) where \( k = v = m_2 \), we have:

\[ x^i = x^{i-m} + x^{i-m}, \quad \text{where} \quad \frac{m}{2} \leq i \leq m - 2, \]
\[ x^i = x^{m+i} + x^{m+i}, \quad \text{where} \quad -m \leq i \leq -(m + 1). \]

### 3. Multiplier based on SPB

Karatsuba method has been used to improve the efficiency of bit-parallel multiplier for GF(2\(^m\)) generated by an AOP (All-One Polynomial) and a trinomial in [1,7,9]. This method can reduce the space complexity by approximately a factor of \( 3/4 \) because it replaces the multiplication by three half-sized integers multiplications. This method, however, will increase the time delay which makes the decrease of space complexity less attractive. Here, by using SPB representation, we modify the Karatsuba method and propose a new multiplier architecture with significantly low space complexity and time delay in the fields generated by EST.

Assume that \( a(x) = x^{-m_2} \sum_{i=0}^{m-1} a_i x^i \), \( b(x) = x^{-m_2} \sum_{i=0}^{m-1} b_i x^i \) are two elements in SPB representation. We partition \( a(x) = A \cdot x^{-m_2} + B \) and \( b(x) = C \cdot x^{-m_2} + D \), where

\[ A = \sum_{i=0}^{m_2-1} a_i x^i, \quad B = \sum_{i=0}^{m_2-1} a_{i+m_2} x^i, \]
\[ C = \sum_{i=0}^{m_2-1} b_i x^i, \quad D = \sum_{i=0}^{m_2-1} b_{i+m_2} x^i. \]

Then, we multiply \( a(x) \) and \( b(x) \) with Karatsuba method and do some transformations as follows:

\[ S = a(x) \cdot b(x) \]
\[ = (A \cdot x^{-m_2} + B)(C \cdot x^{-m_2} + D) \]
\[ = AC \cdot x^{-m_2} + BD + (AC + BD)x^{-m_2} \]
\[ + (A + B)(C + D)x^{-m_2} \]
\[ = (AC \cdot x^{-m_2} + BD)x^{-m_2} \]
\[ + (AC \cdot x^{-m_2} + BD) + (A + B + (C + D)x^{-m_2}. \quad (1) \]

The right side of (1) can be divided into two parts: \( S_{\text{re}} \), which needs further reductions modulo \( f(x) \) and \( S_{\text{nore}} \), which does not need any reductions because the exponents of all elements in \( S_{\text{nore}} \) are located in the interval of \( [-m_2, m_2 - 1] \). These two parts are listed separately as follow:

\[ S_{\text{re}} = (AC \cdot x^{-m_2} + BD)x^{-m_2} + (AC \cdot x^{-m_2} + BD), \]
\[ S_{\text{nore}} = (A + B + (C + D)x^{-m_2}. \]

(i) We consider \( S_{\text{re}} \) in detail first. Let

\[ AC = \left( \sum_{i=0}^{m_2-1} a_i x^i \right) \cdot \left( \sum_{i=0}^{m_2-1} b_j x^i \right) = \sum_{i=0}^{m_2-1} p_i x^i. \]

These \( p_i \)'s can be computed as follows:

\[ p_i = \left\{ \begin{array}{ll}
\sum_{j=0}^{i} a_j b_{i-j} & , 0 \leq i \leq m_2 - 1,
\sum_{j=i-m_2}^{m_2-i} a_j b_{i-j} & , m_2 \leq i \leq m - 2.
\end{array} \right. \quad (2) \]

Similarly, we get the coefficients \( q_i \) of \( BD = \sum_{i=0}^{m_2-1} q_i x^i \) as:

\[ q_i = \left\{ \begin{array}{ll}
\sum_{j=0}^{i} a_j b_{i-j} + q_i & , 0 \leq i \leq m_2 - 1,
\sum_{j=i-m_2}^{m_2-i} a_j b_{i-j} + q_i & , m_2 \leq i \leq m - 2.
\end{array} \right. \quad (3) \]

According to (2) and (3), the result of the expression \( AC \cdot x^{-m_2} + BD \) can be computed quickly.

\[ AC \cdot x^{-m_2} + BD = \sum_{i=0}^{m_2-2} z_i x^i \]
\[ = \sum_{i=0}^{m_2-2} \left\{ \begin{array}{ll}
p_{i-m_2} & , -m_2 \leq i \leq -1,
p_{i-m_2} + q_i & , 0 \leq i \leq m_2 - 1,
q_i & , m_2 \leq i \leq m - 2.
\end{array} \right. \quad (4) \]

From (4), we can find that, for \( -m_2 \leq i \leq -1 \), \( z_i \) contains \((i+m_2+1)\) elements, for \( 0 \leq i \leq m_2 - 1 \), \( z_i \) contains \( m_2 - i \) elements and for \( m_2 \leq i \leq m - 2 \), \( z_i \) contains \((m-1-i)\) elements. Thus, circuit implementation of \( (AC \cdot x^{-m_2} + BD) \) requires \( \frac{m_2}{2}, \frac{m_2}{2} - m + \frac{m_2}{2}, \frac{m_2}{2} - m \) XOR gates. The calculations of \( AC \) and \( BD \) both need
The space and time complexities of $S_{re} = (A + B)(C + D)x^{-\frac{m}{2}}$ can be calculated from Tables 1, 2 and extra gates on adding $S_{re}$ and $S_{nore}$ together:

$$\text{#AND} = \frac{m^2}{2} + \frac{m^2}{4} - \frac{m^2}{2} = \frac{3}{4}m^2,$$

$$\text{#XOR} = \frac{m^2}{2} + \frac{m^2}{4} + 1 + m = \frac{3}{4}m^2 + m + 1,$$

Time delay $= T_A + (1 + [\log_2 m])T_X$.

Because $m$ is even, $[\log_2 m] = [\log_2 (m-1)]$, the time complexity can be rewritten as $T_A + (1 + [\log_2 (m-1)])T_X$.

4. Comparison

In the fields generated by trinomials, low complexity multipliers mainly use polynomial basis. Table 3 gives a comparison of four different implementations of bit-parallel multipliers in the class of fields generated by an equally-spaced trinomial $x^m + x^{\frac{m}{2}} + 1$ according to space complexity and time complexity. From Table 3, the proposed multiplier requires about 25 percent fewer circuit gates than the previous best architectures while with the same time complexity of $T_A + (1 + [\log_2 (m-1)])T_X$. This merit enables the pro-
posed multiplier to be used in space critical area such as smartcard, RFID tags, etc.

5. Conclusion

In this paper, a new bit-parallel multiplier architecture is proposed. In this architecture, SPB and Karatsuba method are combined which can reduce the time complexity and space complexity, respectively. This multiplier can be used in area-critical occasion because of its low space complexity in GF($2^m$) defined by EST. To find more efficient polynomials which can use the method proposed in this paper should be the future work.

References