New Bit-Parallel Systolic Multiplier over GF(2<sup>m</sup>)
Using The Modified Booth’s Algorithm

Chiou-Yng Lee*
LungHwa University
Email: lchiou@computer.org

Hsin-Chiu Yu
Chang Gung University, Hsing Wu College
Email: 092040@mail.hwc.edu.tw

Che Wun Chiou
Ching Yun University
Email: cwchiou@cyu.edu.tw

Abstract—A new algorithm for the multiplication of two elements in GF(2<sup>m</sup>) based on the modified Booth’s algorithm is presented. The proposed algorithm permits efficient realization of the multiplexer-based bit-parallel multiplication using iterative arrays. The latency of the multiplier has [3m/2] clock cycles. For the estimated complexity of the proposed multiplier, we take into the transistor count using a standard CMOS VLSI realization. As <i>m</i> increase, one can find that our proposed multipliers have less complex than the other multipliers. Our analysis shows that, in terms of the time and the space complexities, the multiplexer-based array architecture is the better choice for our proposed bit-parallel systolic multiplier.

I. Introduction

Many important applications, such as error-correcting codes [1] and cryptography [2], are based on GF(2<sup>m</sup>) arithmetic operations. Hence, the design of efficient GF(2<sup>m</sup>) arithmetic with low circuit complexity, high throughout rate and short computation delay is required. Since Yeh, Reed, and Truong [3] first suggested a bit-parallel systolic multiplier over GF(2<sup>m</sup>), various bit-parallel systolic multipliers have been reported [4-6]. Fenn et al. in [4] proposed a systolic multiplier using the dual basis representation. Two parallel systolic multipliers using a unidirectional data flow were suggested by Wang-Lin [5]. A power-sum circuit systolic multiplier for computing AB<sup>2</sup> + C over GF(2<sup>m</sup>) was proposed by Wei [6]. However, the previous existing multipliers require at least the latency of 3m clock cycles.

On the other hand, the key idea of the "divide-and-conquer" methodology was first introduced by Booth in [7]. In this technique, intermediate results are always in a redundant form of two integer numbers. Pekmemstzi in 1999 [8] presented a multiplexer-based array multiplier which efficiently reduced complexities by using the modified Booth’s algorithm. From the basic idea of the multiplexer-based implementation, this article aims to implement the GF(2<sup>m</sup>) multiplier in contrast to previous GF(2<sup>m</sup>) multipliers using AND and XOR gates.

In this paper, we reduce the number of the latency in the proposed multiplication algorithm using the modified Booth’s algorithm. The algorithm can be implemented using a multiplexer-based bit-parallel systolic multiplier. The multiplier has the latency of [3m/2] clock cycles, as opposed to 3m in existing multipliers in [3,5]. Therefore, the speed of the proposed multiplier is twofold. For the estimated complexity of the proposed multiplier, we will take into the transistor count using a standard CMOS VLSI realization. As <i>m</i> increase, one can find that our proposed multipliers have less complex than the other multipliers. Our analysis shows that, in terms of the time and the space complexities, the multiplexer-based array architecture is the better choice for our proposed bit-parallel systolic multipliers.

II. Finite Field Representation

Suppose the reader is already very familiar with the basic notion and theory of the finite field, the characteristic of finite field refers to [1-2]. In the following paragraphs, we will briefly introduce some basic operations of finite fields.

It is well known that the finite field GF(2<sup>m</sup>) can be viewed as a vector space of dimension <i>m</i> over GF(2). A basis of the form 1,α,α<sup>2</sup>,...,α<sup>m-1</sup> is called a polynomial basis of GF(2<sup>m</sup>) and α is called a primitive element of GF(2<sup>m</sup>). A polynomial <i>P</i>(x) of degree <i>m</i> over GF(2) is primitive if <i>P</i>(x) = p<sub>0</sub> + p<sub>1</sub>x + ··· + p<sub>m-1</sub>x<sup>m-1</sup> + x<sup>m</sup> over GF(2) is irreducible and has period 2<sup>m</sup>-1. If α ∈GF(2<sup>m</sup>) is a root of a primitive polynomial <i>P</i>(x), then α<sup>m</sup> = p<sub>0</sub> + p<sub>1</sub>α + ··· + p<sub>m-1</sub>α<sup>m-1</sup>. Let the set {1,α,α<sup>2</sup>,...,α<sup>m-1</sup>} be a polynomial basis, every element A in GF(2<sup>m</sup>) can be represented by A = a<sub>0</sub> + a<sub>1</sub>α + ··· + a<sub>m-1</sub>α<sup>m-1</sup>, where a<sub>i</sub> ∈GF(2) (0 ≤ i ≤ m-1) is the <i>i</i>th coordinate of A. Each element in GF(2<sup>m</sup>) has a unique representation as a linear combination of the polynomial. The multiplication of two elements in GF(2<sup>m</sup>) is uniquely determined by α<sup>m</sup> = ∑<sub>j=0</sub><sup>m-1</sup> p<sub>j</sub>α<sup>j</sup>, p<sub>j</sub> ∈GF(2), since P(α) = 0. Namely, multiplication in GF(2<sup>m</sup>) can be performed by polynomial modulo <i>P</i>(x) on the field elements represented as polynomials of degree <i>m</i> - 1 or less.

III. Proposed Multiplexer-Based Multiplication over GF(2<sup>m</sup>) Using Modified Booth’s Algorithm

Booth [7] firstly introduced the key idea of the "divide-and-conquer" methodology. The major concept is that intermediate results are always in a redundant form of two integer numbers. Since Pekmemstzi in 1999 [8] presented a multiplexer-based array multiplier whose complexities were efficiently reduced by using the modified Booth’s algorithm, this section aims to implement the GF(2<sup>m</sup>)
multiplier using multiplexers other than previous GF($2^m$) multipliers using AND and XOR gates.

Assume that the finite field GF($2^m$) is generated by a primitive polynomial of the form $P(x) = p_0 + p_1 x + p_2 x^2 + \cdots + p_{m-1} x^{m-1} + x^m$ over GF(2). Let $\alpha$ be a root of $P(x)$, i.e., $P(\alpha) = 0$, we have

$$\alpha^m = p_0 + p_1 \alpha + p_2 \alpha^2 + \cdots + p_{m-1} \alpha^{m-1}$$  \hspace{0.5cm} (1)

$$\alpha^{m+1} = p_0 \alpha + p_1 \alpha^2 + p_2 \alpha^3 + \cdots + p_{m-1} \alpha^m$$  \hspace{0.5cm} (2)

where

$$p_j = \begin{cases} p_{j+1} + p_{m-1} p_j & \text{for } 1 \leq j \leq m-1 \\ p_{m-1} p_j & \text{for } j=0 \end{cases}$$

Given the element $A = a_0 + a_1 \alpha + a_2 \alpha^2 + \cdots + a_{m-1} \alpha^{m-1}$ over GF(2) in GF($2^m$), a common computation in both types of multiplications is the multiply-by-$\alpha$, which can be done by the following rules in Eq.(4). Let us define that $A' = A \alpha$, we obtain

$$A' = A \alpha$$

$$= a_0 \alpha + a_1 \alpha^2 + a_2 \alpha^3 + \cdots + a_{m-1} \alpha^m$$

Substituting (1) into (3), we have

$$A' = \left( a_0 + a_1 \alpha^2 + a_2 \alpha^3 + \cdots + a_{m-1} \alpha^m \right) \alpha = a_0 \alpha + a_1 \alpha^2 + a_2 \alpha^3 + \cdots + a_{m-1} \alpha^m$$

where

$$\alpha' = \{ a_j - \alpha a_{m-1} \alpha^j \} \text{ for } 1 \leq j \leq m-1$$

Next, let $A = a_0 + a_1 \alpha + a_2 \alpha^2 + \cdots + a_{m-1} \alpha^{m-1}$, $B = b_0 + b_1 \alpha + b_2 \alpha^2 + \cdots + b_{m-1} \alpha^{m-1}$ and $C = c_0 + c_1 \alpha + c_2 \alpha^2 + \cdots + c_{m-1} \alpha^{m-1}$ be three elements in GF($2^m$) with a primitive polynomial $P(x)$, where $C$ is the multiplication of $A$ and $B$. Assume that $A'$ is pre-computed using Eq. (4), the product $C$ can be represented by

$$C = AB$$

$$= A \sum_{j=0}^{m-1} b_j \alpha^j$$

$$= \alpha^{m-2} \left( b_{m-1} \alpha A + b_{m-2} A \right) + A \sum_{j=0}^{m-3} b_j \alpha^j$$

$$= \alpha^{m-2} \left( b_{m-1} A' + b_{m-2} A \right) + A \sum_{j=0}^{m-3} b_j \alpha^j$$

$$= \alpha^{m-2} \left( b_{m-1} A' + b_{m-2} A \right) + A \left( \sum_{j=0}^{m-5} b_j \alpha^j \right)$$

Let $T_i = t_{i,0} + t_{i,1} \alpha + t_{i,2} \alpha^2 + \cdots + t_{i,m-1} \alpha^{m-1}$ be the $i^{th}$ intermediate multiplication, then the product $C$ can be computed recursively as follows:

$$T_0 = 0$$

$$T_i = T_{i-1} \alpha + A' b_{m-2i+1} + A b_{m-2i}$$

$$C = T_{m/2}$$

Substituting Eq.(1) and Eq.(2) into Eq.(7), $T_i$ yields

$$T_i = t_{i-1,m-3} \alpha^{m-1} + \cdots + t_{i-1,1} \alpha^3 + t_{i-1,0} \alpha^2$$

$$+ t_{i-1,m-1} (p'_m - \alpha^m + \cdots + p'_1 \alpha + p'_0)$$

$$+ t_{i-1,m-2} (p_m - \alpha^{m-1} + \cdots + p_1 \alpha + p_0)$$

$$+ b_{m-2i+1} \alpha^{m-1} + \cdots + a'_i \alpha + a_0$$

$$+ b_{m-2i} (a_{m-1} \alpha^{m-1} + \cdots + a_1 \alpha + a_0)$$

where

$$t_{i,j} = \begin{cases} v_{i,j} + k_{i,j} & \text{for } j = 0, 1 \\ t_{i,j-2} + v_{i,j} + k_{i,j} & \text{for } 2 \leq j \leq m-1 \end{cases}$$

Since the modified Booth's recoding is considered, the quantity $v_{i,j}$ can be determined by the values of $p_j$ and $p'_j$, as shown in Table 1. The quantity $v_{i,j}$ requires computation when both $p_j$ and $p'_j$ have logical value 1s. Hence, the quantity $v_{i,j}$ can be selected from 0, $t_{i-1,m-2}$, $t_{i-1,m-1}$ and $t_{i-1,m-1}$. Similarly, the quantity $k_{i,j}$ can also be selected from 0, $b_{m-2i+1}$, $b_{m-2i}$ and $t_{i} = t_{i-1,m-2} + t_{i-1,m-1}$ depending on the values of $p_j$ and $p'_j$. Similarly, the quantity $k_{i,j}$ can also be selected from 0, $b_{m-2i+1}$, $b_{m-2i} +$ and $t_{i} = t_{i-1,m-2} + t_{i-1,m-1}$ depending on the values of $a_j$ and $a'_j$, as shown in Table 2. Therefore, the computation of $t_{i,j}$ in Eq.(10) can use two 4x1 multiplexers and one 3-input XOR gate to determine the value of $t_{i,j}$ when the values of $t_{i}$ and $t_{i}$ are pre-computed.

| $T_i$ | | $v_{i,j}$ | | $k_{i,j}$ |
|---|---|---|
| 1 | 1 | 0 | $t_{i-1,m-2}$ |
| 0 | 1 | $t_{i-1,m-1}$ |
| 0 | 0 | 0 |

<table>
<thead>
<tr>
<th>$a_j$</th>
<th>$a'_j$</th>
<th>$k_{i,j}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>$b_i$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$b_{m-2i+1}$</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
finite field GF(2^4). An analogous development can be constructed for any finite field GF(2^m).

Fig. 1 shows a signal flow graph (SFG) array for realizing the recursive given in Eqs. (6-10) with m = 4. It consists of [m/2]m U-cells, m V-cells and [m/2] Q-cells. Assume that the cell is located at the ith row and the jth column of the SFG array; we shall then refer to it as the (i,j) cell. From Fig. 1, the first row and the column cells are identical of the V-cell and Q-cell, respectively; other cells are identical of the U-cell. Each V-cell in the position (0,j), for 0 ≤ j ≤ m − 1, uses one 2-input AND and one 2-input XOR gates to perform the computation of \( a'_j = a_{m-1}p_j + a_{j-1} \), as shown in Fig.3. Applying the basic concept of the modified Booth’s algorithm, each Q-cell in Fig.4 generates two sets, \( (t_{i-1,m-1}, t_{i-1,m-2}, t_i) \) and \( (b_{m-2i+1}, b_{m-2i}, b_i) \) to provide the U-cell to determine the values \( v_{i,j} \) and \( k_{i,j} \), respectively. While two sets, \( (t_{i-1,m-1}, t_{i-1,m-2}, t_i) \) and \( (b_{m-2i+1}, b_{m-2i}, b_i) \), are generated in Q cells, and U \( i,j \)-cell in Fig.2, for \( 0 ≤ j ≤ m − 1 \) and \( 1 ≤ i ≤ [m/2] \), is performed by the following operations:

1) MUX1: Determine the value \( v_{i,j} \) which is selected from 0, \( t_{i-1,m-2}, t_{i-1,m-1} \) and \( t_i \) by the values of \( p_j \) and \( p'_j \);  
2) MUX2: Determine the value \( k_{i,j} \) which is selected from 0, \( b_{m-2i-1}, b_{m-2i} \) and \( b_i \) by the values of \( a_j \) and \( a'_j \);  
3) 3-input XOR: Perform the computation of \( t_{i,j} = t_{i-1,j-2} + v_{i,j} + k_{i,j} \).

Applying the cut-set systolization techniques in [9], support that two adjacent cells in the horizontal direction are combined into one cell to get a modified SFG array, we can derive a new parallel-in parallel-out systolic multiplier for computing \( AB \) in GF(2^m), as shown in Fig.5. The circuit consists of \( [m/2]^2 \) U cells, \([m/2]\) V cells and \([m/2]\) Q-cells. Each U cell is composed of two 3-input XOR gates, two MUX gates and 24 1-bit latches, as depicted in Fig.6. Each V cell is identical of two 2-input AND gates, two 2-input XOR gates and 10 1-bit latches, as shown in Fig.7. The longest propagation delay time for each cell is the total of the delay due to one MUX and one 3-input XOR gates. The proposed systolic multiplier over GF(2^m) can produce one result every clock cycle with an initial delay of \([3m/2]\) clock cycles.

Several bit-parallel systolic multipliers have been reported for performing the multiplication in GF(2^m) [3-6]. Their architectures use AND and XOR gates to implement various bit-parallel systolic multipliers. The architecture for computing multiplication in GF(2^m) [5] require at least the latency of 3m clock cycles, and incorporate \( m^2 \) identical cells. Each cell is composed of two 2-input AND...
The total number of transistors for the proposed multipliers has attractive features for high-speed VLSI system design, such as simplicity, regularity and modularity.

V. Conclusions

In this paper, we have proposed a novel multiplication algorithm using the modified Booth’s algorithm that permits efficient VLSI realizations. The proposed algorithm can use a multiplexer-based array to implement a new bit-parallel systolic multiplier. The multiplier has the latency of $3m/2$ clock cycles, as opposed to $3m$ in previous works used AND and XOR gates. Therefore, the speed of the proposed multiplier is twofold. For the estimated complexity of the proposed multiplier, we will take into the transistor count using a standard CMOS VLSI realization. Table 4 shows that our proposed multiplier has less complex than other bit-parallel systolic multipliers. Therefore, one can find that, in terms of the time and the space complexities, the multiplexer-based multiplier is the better choice for our proposed bit-parallel systolic multipliers. Moreover, our proposed multipliers have attractive features for high-speed VLSI system design, such as simplicity, regularity and modularity.

References


TABLE III

Comparison of the related systolic multipliers over GF(2^m)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#cells</td>
<td>m^2</td>
<td>m</td>
<td>m</td>
</tr>
<tr>
<td>Throughput</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>cell complexity</td>
<td>2 AND</td>
<td>1 XOR</td>
<td>7 latches</td>
</tr>
<tr>
<td>Computation time per cell</td>
<td>T_{AND}</td>
<td>+ T_{XOR}</td>
<td>T_{MUX_{x+1}} + T_{XOR}</td>
</tr>
<tr>
<td>Latency (unit=cycles)</td>
<td>3m</td>
<td>3m</td>
<td>3m</td>
</tr>
</tbody>
</table>

TABLE IV

The total number of transistors for various bit-parallel systolic multipliers over GF(2^m)

<table>
<thead>
<tr>
<th>multipliers</th>
<th>basis</th>
<th>The total number of transistors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wang-Lin [5]</td>
<td>polynomial</td>
<td>42m^2-4m</td>
</tr>
<tr>
<td>Fenn et al. [4]</td>
<td>dual</td>
<td>42m^2-4m</td>
</tr>
<tr>
<td>Lee-Chiu[10]</td>
<td>type-2 Normal</td>
<td>39m^2+6m</td>
</tr>
</tbody>
</table>

Fig. 6. the detailed circuit of the $U$ cell
Fig. 7. the detailed circuit of the $V$ cell