Fault-Tolerant Bit-Parallel Multiplier for Polynomial Basis of GF(2^m)

Chiou-Yng Lee, Pramod Kumar Meher, and Chia-Chen Fan

Abstract—Novel fault-tolerant architectures for bit-parallel polynomial basis multiplier over GF(2^m), which can correct the erroneous outputs using linear code, are presented. A parity prediction circuit based on the code generator polynomial that leads lower space overhead has been designed. For bit-parallel architectures, the space overhead is about 11%. Moreover, there is only marginal time overhead due to incorporation of error-correction capability that amounts to 3.5% in case of the bit-parallel multiplier. Unlike the existing concurrent error correction (CEC) multipliers or triple modular redundancy (TMR) techniques for single error correction, the proposed architectures have multiple error-correcting capabilities.

Index Terms—Fault tolerant system, finite field, parity prediction.

1. Introduction

Arithmetic operations in finite (Galois) fields have received much of attention because of their important and practical applications in areas of error correcting code and cryptography[1][2]. Hardware implementations in such applications require significant amount of circuits to realize the basic arithmetic operations, especially for multiplication. For cryptographic applications, the field size is from 160 bits to 2048 bits long. It is likely that one or more of so many transistors may become faulty during the operation of the system and that could lead to incorrect output in the computation of field multiplication. The design of efficient multipliers with fault tolerance feature is, however, highly desirable to have a reliable operation of cryptographic hardware.

In the finite GF(2^m), the field element is generally represented by one of three bases, i.e. polynomial basis (PB), normal basis (NB) and dual basis (DB). Various GF(2^m) multipliers, including bit-serial, bit-parallel and digit-serial architectures, have been suggested in the literatures [3]-[7]. In practical implementations, PB multipliers do not need basis conversion, while DB and NB multipliers require extra hardware for the basis conversion. Most of the existing architectures for GF(2^m) multipliers are focused on minimizing the time- and space-complexity. But, the major drawback of these circuits is that they cannot detect/correct the errors in the results.

Typically, to achieve fault tolerant architectures, redundancy is often used in the form of hardware, software, information, or time. The most common form of fault tolerant designs is based on hardware redundancy. For example, duplication of hardware can be used to detect an error while triple modular redundancy can correct an error[8][9]. Hardware redundancy on the other hand adversely impacts the physical size, power consumption, and cost of the system. The methods in [8] and [9] use at least 100% space overheads. Time redundancy technique involves re-computation with shifted operands (RESO) to achieve the concurrent error detection (CED) capability in arithmetic components. Finite field multipliers using this method are proposed in [10] and [11]. These architectures, however, also require extra hardware overhead to perform the basis conversion.

In this article, an error decoding scheme is proposed for correcting errors in bit-parallel multiplications over GF(2^m). The proposed scheme is based on linear code and can be applied in any finite field GF(2^m). The major contributions for this work are as follows. First, the proposed concurrent error correction (CEC) architecture combines single error-correcting linear code to achieve concurrent error-correcting capabilities. The proposed scheme could be used effectively to have a fault-tolerant cryptographic architecture even when the field size gets large. Secondly, the proposed parity predication circuit is based on the code generator polynomial to achieve efficient CEC architectures. This leads to reduce the space overhead of our proposed architectures. The space overhead is obtained by 11%, while traditional CED multiplier[12] is about 45% space overhead.
2. Mathematical Background

2.1 Review Stage

The finite fields $\operatorname{GF}(2^n)$ have $2^n$ elements such that $\operatorname{GF}(2^n) = \{0, 1, x, x^2, \ldots, x^{2^n-1}\}$, where $x$ is a primitive element and is the root of an irreducible (primitive) polynomial $F(x) = f_0 + f_1 x + \cdots + f_m x^{m-1} + x^m$. Since $F(x) = 0$, e.g., $x^m = f_0 x^{m-1} + \cdots + f_m x^{m-1}$, every element $A$ in $\operatorname{GF}(2^n)$ can then be represented by $A = a_0 + a_1 x + \cdots + a_m x^{m-1}$ over $\operatorname{GF}(2^n)$. The set $N = \{1, x, x^2, \ldots, x^{2^n-1}\}$ is called the PB of $\operatorname{GF}(2^n)$.

Let $B = b_0 + b_1 x + \cdots + b_m x^{m-1}$ be another element in $\operatorname{GF}(2^n)$. The multiplication of $A$ and $B$ over $\operatorname{GF}(2^n)$ is given by

$$C = AB \mod F(x)$$

where

$$C = (a_0 + a_1 x + \cdots + a_m x^{m-1}) B \mod F(x).$$

(1)

The above equation leads to least significant bit (LSB) first multiplication scheme. The recursive operation in the step $i$, $1 \leq i \leq m$, does the following computation in parallel.

$$C_i = C_{i-1} \cdot a_i B_i$$

(2)

$$B_i = (B_{i-1} \cdot x) \mod F(x).$$

(3)

Fig. 1 shows the bit-parallel multiplier architecture based on the operations of (2) and (3). It consists of one $\alpha$ module, one product-sum (PS) module and two registers. The $\alpha$ module performs the operation of (3). The PS module computes the product-sum operation of (2).

2.2 $(n, m)$ Linear Code

Linear code is an error-correcting code, which can detect and correct a single-bit error. In contrast, the simple parity code cannot correct errors, nor can it be used to detect more than one error. In the $(n, m)$ code, the binary information sequence is segmented into message blocks of fixed length $m$, denoted by $U = [u_0, u_1, \ldots, u_{m-1}]$. Let the $(n, m)$ code be constructed from an irreducible $P(x)$ of degree $n-m$, and let $P_U$ be the parity prediction of the message $U$ defined by

$$P_U = x^{n-m} U \mod P(x)$$

(4)

Thus, the systematic structure of the codeword $V$ is given as

$$V = P_U + x^{n-m} U$$

(5)

According to (4) and (5), each message $U$ can be translated into the binary $n$-tuple $V = [v_0, v_1, \ldots, v_{n-1}]$ with $n > m$. There should be a one-to-one correspondence between a message $U$ and its codeword $V$, the $2^n$ codewords must be distinct. By using $P(x)$ to establish the $(n, m)$ code, we have

$$P_0 = x^{m} \mod P(x) = p_0 + p_1 x + \cdots + p_{n-m-1} x^{n-m-1}$$

$$P_1 = x^{m-1} \mod P(x) = p_0 + p_1 x + \cdots + p_{n-m-1} x^{n-m-1}$$

$$P_{n-m-1} = x^{m-1} \mod P(x) = p_0 + p_1 x + \cdots + p_{n-m-1} x^{n-m-1}.$$  

Every message $U = u_0 + u_1 x + \cdots + u_{m-1} x^{m-1} = [u_0, u_1, \ldots, u_{m-1}]$ is based on above equations to encode the codeword $V$, represented by

$$V = U \cdot G$$

(6)

where

$$G = \begin{bmatrix}
\text{matrix } P & \text{identity matrix}
\end{bmatrix}$$

(7)

In (7), $p_{ij} = 0$ or 1. The matrix $G = [P|I_{n-m}]$ is called the generator matrix of the systematic codeword in (5). The components of the corresponding codeword $V = [v_0, v_1, \ldots, v_{n-1}]$ are

$$v_{0 \leq i \leq n} = u_i$$

(8)

and

$$v_{j} = u_0 p_{0j} + u_1 p_{1j} + \cdots + u_{m-1} p_{(m-1)j}$$

(9)

for $0 \leq j \leq n - m$.

In the following sections, we consider the $(n, m)$ code to develop CEC architectures in polynomial basis multiplications. Fig. 2 shows the functional block for CEC PB multiplier, which includes three major modules: 1) combinational module, 2) parity prediction generator (PPG) and 3) parallel error corrector (PEC) module. The PPG module is based on two signals $A$ and $B$ to obtain the predicted parity bits $P_C$ of $C = AB$. At this time, we combine both values $C$ and $P_C$ to form the codeword $V = [v_0, v_1, \ldots, v_{n-1}] = P_C + x^{n-m} C$, where $C$ is the output of the combinational circuit and $P_C$ is generated from the PPG module. The PEC module is based on two values $C$ and $P_C$ to correct errors in the codeword $V$.

![Fig. 1. Structure of bit-parallel multiplier over \( \operatorname{GF}(2^n) \).](image)

![Fig. 2. Functional block of CEC PB multiplier architecture.](image)
3. Single Error-Correcting Architecture in PB Multiplier over GF(2^m)

Since the structure of PB multiplier is composed of α and PS modules, this section considers the error-correcting decoding scheme of a linear code to develop a single error correction in PB multiplication.

3.1 Notation

Let \( P(x) \) of degree \( (n-m) \) be the generator polynomial of the \((n, m)\) code. From (5) we find that, every codeword \( V \) consists of \( m \)-bit message and \((n-m)\)-bit predicted parity digits. We can obtain the following properties.

Theorem 1. Let \( g \in \text{GF}(2) \) and \( A \) be an element of \( \text{GF}(2^m) \) in the polynomial basis representation. The scalar product \( gA \) is given by \( P_{gA} = gP_A \).

Proof. The message \( A \) uses (6) to encode the codeword \( V \), i.e., \( V = P_A + x^{n-m}A \) and \( P_A = x^{n-m}A \mod P(x) \). Since \( g \in \text{GF}(2) \), \( gV = g(P_A + x^{n-m}A) = gP_A + x^{n-m}gA \), we have \( P_{gA} = x^{n-m}gA \mod P(x) = gP_A \).

Theorem 2. Let \( A \) and \( B \) be any two elements of \( \text{GF}(2^m) \), and let \( P_A \) and \( P_B \) respectively be their parity predictions.

Proof. Based on (6), the codewords \( V_A \) and \( V_B \) is encoded from the element \( A \) and \( B \), respectively, e.g., \( V_A = P_A + x^{n-m}A \) and \( V_B = P_B + x^{n-m}B \). Thus, we can obtain

\[
V_A + V_B = P_A + x^{n-m}A + P_B + x^{n-m}B = (P_A + P_B) + x^{n-m}(A + B).
\]

That is,

\[
P_{A+B} = P_A + P_B.
\]

In the following section, assume that the multiplication of \( C = AB \) is served as the message block. By using the basic operations of the above theorems and (6), we will develop new error-correcting architectures in bit-parallel PB multiplication.

3.2 Compute Parity Prediction of \( xB \) in \( \alpha \) Module

Let \( B \) be any element in \( \text{GF}(2^m) \) defined by the irreducible polynomial \( F(x) = x^m + \bar{F} \) of degree \( m \), where \( \bar{F} = f_0 + f_1x + \cdots + f_{m-1}x^{m-1} \), the \( \alpha \) module obtains that

\[
xB \mod F(x) = \bar{B}x + b_{m-1}\bar{F}
\]

where

\[
\bar{B} = b_0 + b_1x + \cdots + b_{m-2}x^{m-2}.
\]

For computing the parity prediction of \( xB \), we have the following properties.

Theorem 3. Let \( B = (b_0, b_1, \ldots, b_{m-1}) \) be the element of \( \text{GF}(2^m) \), and let \( P(x) \) be used to construct the \((n, m)\) code and \( F(x) = x^n + \bar{F} \) be used to generate the field \( \text{GF}(2^m) \). Thus, we have that \( P_{xB} = xP_B + b_{m-1}(1 + P_T) \mod P(x) \).

Proof. Let \( B \in \text{GF}(2^m) \), and let the intermediated parameter \( x \) be the root of \( F(x) \), i.e., \( x^n = \bar{F} \). Thus, we have

\[
xB = b_0x + b_1x^2 + \cdots + b_{m-1}x^n = \bar{B}x + b_{m-1}\bar{F}.
\]

Using (4), the parity prediction of \( xB \) can be obtained as

\[
P_{xB} = x^{n-m}(xB) \mod P(x) = x^{n-m}(\bar{B}x + b_{m-1}\bar{F}) \mod P(x) = xP_B + b_{m-1}\bar{F} \mod P(x).
\]

The \((n, m)\) code is a cyclic code. In fact, the well-known cycle code has the corresponding modulus as \( x^m + 1 \). We have \( P(x)x^m + 1 \), e.g., \( x^m = 1 \mod P(x) \). Thus, the parity prediction of \( xB \) can be obtained

\[
P_{xB} = x^{n-m}B \mod P(x)
\]

\[
= x^{n-m}(b_0 + b_1x + \cdots + b_{m-1}x^{m-1}) \mod P(x)
\]

\[
= x^{n-m}(\bar{B} + b_{m-1}\bar{F}) \mod P(x)
\]

\[
= P_B + b_{m-1}x^{n-m} \mod P(x)
\]

or

\[
P_{xB} = P_B + b_{m-1}x^{n-m} \mod P(x).
\]

Substituting (13) to (11), \( P_{xB} \) can be simplified to have

\[
P_{xB} = xP_B + b_{m-1}x^n + b_{m-1}P_T \mod P(x)
\]

\[
= xP_B + b_{m-1}(1 + P_T) \mod P(x).
\]

Fig. 3 shows the structure of the PPG circuit for evaluating the parity prediction of \( xB \) according to (14), when \( P_B \) and \( (1 + P_T) \) are previously determined. The \( \alpha' \) module is performed by computing \( xP_B + b_{m-1}(1 + P_T) \mod P(x) \).

When \( P(x) \) is a primitive trinomial of the degree \((n-m)\), the \( \alpha' \) module involves only one XOR gate to compute \( xP_B \mod P(x) \). In the structure of Fig. 3, the PPG module consists of \((n-m)\) AND gates and \((n-m+1)\) XOR gates.

Fig. 3. PPG circuit for computing the parity prediction of \( xB \).
Generally, at the $i$th computational loop the $\alpha$ module performs $x^iB = x(x^{i-1}B) \mod F(x)$, where $x^{i-1}B$ is the result of the previous computational loop. Thus, based on (14), the parity bits of $x^iB$ can be represented by

$$P_{x^iB} = xP_{x^{i-1}B} + b_{i-1,m-1}(1 + P_2) \mod P(x)$$

where $x^{i-1}B = b_{i-1,0} + b_{i-1,1}x + \cdots + b_{i-1,m-1}x^{m-1}$.

### 3.3 Computing Parity Prediction of Intermediate Result in PS Module

In the $i$th loop the intermediate result is obtained as $C_i = C_{i-1} + a_i x^i B$, where the parity prediction of $x^iB$ could be estimated as shown in the previous subsection. Using Theorem 2 and (15), the parity prediction of $C_i$ can thus be obtained as

$$P_{C_i} = P_{C_{i-1}} + a_iP_{x^iB}.$$  (16)

Fig. 4 shows the parity prediction of the PS module, denoted by PS module which performs according to (16). This module is like the structure of PS module, which requires the space complexity of $(n-m)$ XOR gates and $(n-m)$ AND gates.

### 3.4 Proposed CEC PB Multiplier Architecture

Let us consider the implementation of the CEC bit-parallel PB multiplier with CEC capability. According to (1), the parity prediction of the product

$$C = AB$$

is computed by

$$P_{C} = a_0B \mod F(x) + a_1xB \mod F(x) + \cdots + a_{m-1}x^{m-1}B \mod F(x)$$

where

$$P_{C} = a_0P_{B} + a_1P_{xB} + \cdots + a_{m-1}P_{x^{m-1}B}.$$  (17)

Fig. 5 shows the bit-parallel multiplier with CEC capability designed according to (17). The circuit has the space overhead of $m$ PPG modules, $m$ PS modules and one PEC module.

When the $(n, m)$ code is decided, the input data $1 + P_2$ is pre-computed. For example, let the $(7, 4)$ code be applied for correcting errors in PB multiplier over $GF(2^4)$, and the field element is constructed from the primitive polynomial $F(x) = 1 + x + x^3$. Thus, we can have $1 + P_2 = P_{111} = x^2$. Assume that the generator matrix $G$ is derived according to (7) from a primitive polynomial $P(x)$ and the parity-check matrix $H$ of the codeword is defined by $H = [I_{(n-m)\times(n-m)} \mid P^T]$. Since the codeword is generated by $G$, we obtain $GH^T = 0$. Let the syndrome $S = [s_0, s_1, \ldots, s_{n-1}]$ be computed by $S = VH^T$ and each component of $S$s can be calculated as

$$s_i = v_i + \sum_{j=0}^{n-1} v_{j,i}P_{ii}.$$  (18)

If $S \neq 0$, it implies that stuck-at fault has occurred in the multiplier circuit. Since the syndrome $S$ can be mapped into the error pattern $E = [e_0, e_1, \ldots, e_{n-1}]$, we can perform the addition of the codeword $V$ and the error pattern $E$, i.e., $V = V + E$, to correct the multiplication of $C = AB$. For example, let $P(x) = 1 + x + x^3$ be used to construct the $(7, 4)$ code, and let $V = P_C + x^2C$ be the received codeword, the component of the syndrome $S = VH^T = [s_0, s_1, s_2]$ can then be computed as $s_0 = v_0 + v_1 + v_2 + v_3$, $s_1 = v_1 + v_2 + v_3 + v_4$, $s_2 = v_2 + v_3 + v_4 + v_5$. If the error occurs at the fourth bit of $V$, i.e., $E = x^4$, then, the corresponding syndrome $S$ for this error $E$ is $S = [1, 1, 0]$. The predicted error bit $\overline{e}_3$ is calculated by $\overline{e}_3 = s_0\Lambda s_1\Lambda s_2$. After $\overline{e}_3$ is computed, we perform $\overline{e}_3 + v_3$ to correct the error in the fourth bit of $V$. As mentioned above, the computation in the PEC module includes the following three steps: 1) computing the syndrome $S$, 2) finding the error pattern $E$, and 3) correcting errors in the codeword $V$.

### 4. Time and Area Overheads

In the previous section, we have proposed the PB multiplier with CEC capability by employing the decoding of the linear code. For $GF(2^4)$ PB multiplier (Fig. 1), the space-complexity of $\alpha$ module depends on the Hamming weight $w$ of $F(x)$. It requires $(w-2)$ XOR gates to achieve the operation of $xB \mod F(x)$. From the structure of Fig. 1, we find that a bit-parallel multiplier over $GF(2^k)$ consists of $k^2$ AND gates and $k(k+w-2)$ XOR gates.

Recently, Bayat-Saramdi and Hasan[13] have suggested a CED multiplier for two error detection. In the field $GF(2^{163})$, we use the field defined by the irreducible polynomial $F(x) = x^{163} + x^{159} + x^{148} + x^{137} + x^{135} + x^{133} + x^{129} + x^{127} + x^{125} + x^{123} + x^{121} + x^{119} + x^{117} + x^{115} + x^{113} + x^{111} + x^{109} + x^{107} + x^{105} + x^{103} + x^{101} + x^{99} + x^{97} + x^{95} + x^{93} + x^{91} + x^{89} + x^{87} + x^{85} + x^{83} + x^{81} + x^{79} + x^{77} + x^{75} + x^{73} + x^{71} + x^{69} + x^{67} + x^{65} + x^{63} + x^{61} + x^{59} + x^{57} + x^{55} + x^{53} + x^{51} + x^{49} + x^{47} + x^{45} + x^{43} + x^{41} + x^{39} + x^{37} + x^{35} + x^{33} + x^{31} + x^{29} + x^{27} + x^{25} + x^{23} + x^{21} + x^{19} + x^{17} + x^{15} + x^{13} + x^{11} + x^{9} + x^{7} + x^{5} + x^{3} + x^{1} + 1$. Table 1 lists the complexities of the proposed architectures with these two architectures for comparison. Our proposed CEC architecture has single-error capabilities, while the existing architecture[13] can detect two errors.
Table 1. Comparison of related CED/CEC PB multipliers using a linear code

<table>
<thead>
<tr>
<th>CECC/CEC multipliers</th>
<th>Error capability</th>
<th>Space overhead (%)</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference [13]</td>
<td>Two error detection</td>
<td>39.7</td>
<td>Bit-serial</td>
</tr>
<tr>
<td>Fig. 5</td>
<td>One error correction</td>
<td>11</td>
<td>Bit-parallel</td>
</tr>
</tbody>
</table>

5. Conclusions

This article presents the use of a linear code for correcting errors in bit-parallel PB multiplication. The proposed architectures have capability for single error correction, while the existing CED multiplier detects two errors. Moreover, in $\text{GF}(2^{163})$, the space overhead of the proposed bit-parallel multipliers is about 11% while CED multiplier\textsuperscript{[13]} is about 39.6%. Moreover, there is only marginal time overhead due to incorporation of error-correction capability that amounts to 3.5% in case of the bit-parallel multiplier. The proposed architectures can therefore be used effectively in fault tolerant cryptosystems.

References


Chiu-Yung Lee received the Bachelor’s degree in medical engineering in 1986 and the M.S. degree in electronic engineering in 1992, both from the Chung Yuan University, Taiwan, China, and the Ph.D. degree in electrical engineering from Chang Gung University, Taiwan, China, in 2001. From 1988 to the present, he has been a research associate with Chunghwa Telecommunication Laboratory in Taiwan, China. He is currently an associate professor with Department of Computer Information and Network Engineering, Lung-Hwa University of Science and Technology. His research interests include computations in finite fields, error-control coding, signal processing, and digital transmission system. He is a senior member of the IEEE. He was also an honor member of Phi Tao Phi in 2001.

Pramod Kumar Meher received the B.S. (Honours) and M.S. degrees in physics and the Ph.D. in science from Sambalpur University, Sambalpur, India, in 1976, 1978, and 1996, respectively. Currently, he is a senior scientist with the Institute for Infocomm Research, Singapore. Prior to this assignment he was a visiting faculty with the School of Computer Engineering, Nanyang Technological University, Singapore. He was a professor of Computer Applications with Utkal University, Bhubaneswar, India from 1997 to 2002, a reader in electronics with Berhampur University, Berhampur, India from 1993 to 1997, and a lecturer in physics with various Government Colleges in India from 1981 to 1993. His research interest includes design of dedicated and reconfigurable architectures for computation-intensive algorithms pertaining to signal processing, image processing, communication, and intelligent computing. He has published more than 100 technical papers in various reputed journals and conference proceedings.

Dr. Meher is a fellow of the Institution of Electronics and Telecommunication Engineers (IETE), India and a fellow of the Institution of Engineering and Technology (IET), UK. He is currently serving as associate editor for the IEEE Transactions on Circuits and Systems-II: Express Briefs, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, and Journal of Circuits, Systems, and Signal Processing. He was the recipient of the Samanta Chandrasekhar Award for excellence in research in engineering and technology for the year 1999.

Chia-Chen Fan is currently the candidate of Bachelor’s degree in computer information and network engineering with Lungwa University of Science and Technology. His research interests include computations in finite fields, error-control coding, and signal processing.