Fast and Low cost GF\((2^8)\) Multiplier design based on Double Subfield Transformation

Hyeong-Keon An

Tong Myung University, 428 Sinseon-ro, Nam-gu, Busan, 608-711, Republic of Korea Tel: +82-51-629-1147
hkan@tu.ac.kr

Abstract

We describe the design method of the efficient GF\((2^8)\) multiplier using double subfield transformation. Here we first transform the GF\((2^8)\) field elements to the GF\((2^4)\) field elements. And then we again transform the GF\((2^4)\) field elements to the GF\((2^2)\) field. This double transform generates the very fast and cost effective GF\((2^8)\) multiplier. The Multiplier can be used for the Reed Solomon Error correction and Encryption/Decryption on the digital communication channel.

Keywords: Reed Solomon, Galois Field, GF\((2^8)\), Error correction, Encryption, Decryption, Multiplier, Subfield, Transformation

1. Introduction

The paper is about the new efficient design for the Galois GF\((2^8)\) multiplier based on the subfield transform algorithm [1]. The multiplier is used for the circuit of Error correcting Reed Solomon Codec and Endecrationg system.

Here Error control codec is used to protect digital information from errors during transmission or storage. Reed Solomon codes are a class of codes that have been widely used in applications such as satellite transmission and solid state Diskonchip, Modems of mobile phone. The purpose of encryption is to prevent unauthorized person from viewing or modifying the data. The unencrypted data is referred to as the plain text and the encrypted data as the cipher text [6].

In this paper, Chapter 2 describes the transformation between Galois GF\((2^4)\) and GF\((2^2)\) fields and GF\((2^8)\) multiplier design. Chapter 3 describes the transformation between GF\((2^5)\) and GF\((2^8)\) fields. Using these transformation, we design GF\((2^8)\) multiplier based on GF\((2^2)\) field logic. Chapter 4 describes the simulation result of the new design and we find much faster characteristics of the new method. In chapter 5, we compare the new multiplier and the classical direct GF\((2^8)\) multiplier in gate count respect. Here we find that the new method gives rise to the very cost effective multiplier, so gate count is greatly reduced [7]. In chapter 6, we make a concluding remarks and our future plan. Especially using the same double subfield transformation, we hope that we can design also the very efficient GF\((2^8)\) divider based on the GF\((2^2)\) field logic. The divider can be used for the endecryption machine.
2. Transformation between Galois GF($2^2$) and GF($2^4$) Fields

2.1. Transformation between 2 Fields

We first change GF($2^4$) galois elements to GF($2^2$) elements. The operational method and circuit over GF($2^4$) using a subfield GF($2^2$) are as follows.

Suppose that $\alpha^k$ over GF($2^4$) is represented as $\alpha^k = a + b \beta$, where $a, b \in$ GF($2^2$) and $\beta \in$ GF($2^4$). If so, it can be represented as

$$\alpha^k = z0 + z1\gamma + (z2 + z3\gamma)\beta$$

(1)

Here, $\{z_i\} = \{0, 1\}$ and $\gamma \in$ GF($2^2$).

According to [ ], $\beta = \alpha^7 \in \text{GF}(2^4)$, $\gamma = \alpha^5 \in \text{GF}(2^2)$. So

$$\alpha^k = (z0, z1, z2, z3$$

(2)

Now equation (2) is expanded as follows.

$$\alpha^k = (z0 + z1 + z2 + z3, z1 + z2 + z3, z2, z1)$$

(3)

From equation (3), we get following conversion formula. A conversion from elements represented by the basis of GF($2^4$) into elements represented by the basis of GF($2^4$) is as follows,

$$b0 = z0 + z1 + z2 + z3$$

(4)

From equation (4), a conversion from elements represented by the basis of GF($2^4$) into elements represented by the basis of GF($2^2$) is as follows.

$$Z0 = b0 + b1$$

$$Z1 = b3$$

$$Z2 = b2$$

$$Z3 = b1 + b2 + b3$$

(5)

We showed the equation (5) in Figure 1(c).
2.2. GF(2^2) Multiplier Design

We show in Table 1 the Multiplication results of GF(2^2) Galois elements.

In Table 1, we use the primitive polynomial \( p(x) = x^2 + x + 1 = 0 \). So \( \alpha^3 = 1 \), \( \alpha^4 = \alpha \) so on. Therefore if \( c_0c_1 = (b_0b_1) (a_0a_1) \), the logic equations for GF(2^2) multiplier are as in equations (6), where \( c_0, c_1, b_0, b_1, a_0, a_1 \in GF(2) \):

\[
\begin{align*}
    c_0 + c_1 \alpha &= (b_0 + b_1 \alpha)(a_0 + a_1 \alpha) \\
    &= a_0b_0 + \alpha(a_1 + b_1) + \alpha^2a_1b_1 \\
    &= a_0b_0 + a_1b_1 + \alpha(a_1 + b_1 + a_1b_1)
\end{align*}
\]

So

\[
\begin{align*}
    c_0 &= a_0b_0 + a_1b_1 \\
    c_1 &= a_1 + b_1 + a_1b_1 \\
\end{align*}
\]

... (6)

The Logical circuit for the equation (6) are depicted in the Figure 1(b).

Table 1. Multiplication Table for GF(2^2) Elements Multiplication

<table>
<thead>
<tr>
<th></th>
<th>(0)</th>
<th>(1)</th>
<th>(( \alpha^2 ))</th>
<th>(( \alpha ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>b0b1</td>
<td>00</td>
<td>10</td>
<td>11</td>
<td>01</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>( \alpha^2 )</td>
<td>( \alpha )</td>
</tr>
<tr>
<td>00</td>
<td>0</td>
<td>0</td>
<td>( \alpha )</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>( \alpha^2 )</td>
<td>( \alpha )</td>
<td>1</td>
</tr>
<tr>
<td>01</td>
<td>0</td>
<td>( \alpha )</td>
<td>1</td>
<td>( \alpha^2 )</td>
</tr>
</tbody>
</table>

2.3. GF(2^4) Multiplier based on GF(2^2) Subfield Logic

Suppose \( M = A \cdot B \) where \( A, B, M \in GF(2^4) \). Then

\[
\begin{align*}
    M &= M_0 + \beta M_1 \\
    A &= A_0 + \beta A_1 \\
    B &= B_0 + \beta B_1, \quad A_0, B_0, C_0, A_1, B_1, C_1 \in GF(2^2) \quad \text{So,} \\
    M_0 &= A_0B_0 + A_1B_1 \Psi \\
    M_1 &= (A_0 + A_1)(B_0 + B_1) + A_0B_0 \\
\end{align*}
\]

... (7)

where \( M_0, M_1 \in GF(2^2) \)
Equation (7) can be implemented by using 3 GF($2^2$) multipliers and $\Psi$ multiplier over GF($2^2$). In Figure 1(a) we show the $\Psi$ multiplier.

(a) $\Psi$ multiplier: (A0,A1) goes to (A1,A0+A1), A0,A1 $\in$ GF(2)

(b) GF($2^2$) Multiplier Logic Circuit

(c) Logic Circuit of Equation (5)

**Figure 1.** $\Psi$ Multiplier, GF($2^2$) Multiplier and GF($2^4$) to GF($2^2$) Conversion logic Circuit
3. GF(2^8) Multiplier based on GF(2^2) Subfield Multiplier

3.1. Derivation of GF(2^8) Multiplier Logic based on GF(2^2) subfield Elements

If \( C_8 = A_8 B_8 \), \( C_s, A_s, B_s \), \( \beta_s \in \text{GF}(2^8) \) and let

\[
C_s = C_0 A_4 + \beta_4 C_1 A_4 = C_0 A_2 A_0 + \beta_4 C_0 A_1 A_1 + \beta_4 (C_1 A_0 + \beta_4 C_1 A_1)
\]

\[
B_s = B_0 A_4 + \beta_4 B_1 A_4 = B_0 A_2 A_0 + \beta_4 B_0 A_1 A_1 + \beta_4 (B_1 A_0 + \beta_4 B_1 A_1)
\]

\[
A_s = A_0 A_4 + \beta_4 A_1 A_4 = A_0 A_2 A_0 + \beta_4 A_0 A_1 A_1 + \beta_4 (A_1 A_0 + \beta_4 A_1 A_1)
\]

Where \( A_0, B_0, C_0, A_1, B_1, C_1, \beta_4 \in \text{GF}(2^4) \) and

\[
A_00, B_00, C_00, A_01, B_01, C_01, A_10, B_10, C_10, A_11, B_11, C_11 \in \text{GF}(2^2)
\]

\[
\cdots (8)
\]

Then

\[
C_0 A_4 = A_00 B_00 + A_01 B_01 \Psi + \beta_4 ((A_00 A_01) (B_01 + B_00 A_00)) + \gamma (A_10 B_10 + A_11 B_11 \Psi + \beta_4 ((A_10 A_11) (B_11 + B_10 A_10)) + A_00 B_10 A_10)
\]

and

\[
C_1 A_4 = (A_10 + A_00)(B_10 + B_00) + (A_11 + A_01)(B_11 + B_01) \Psi + \beta_4 ((A_10 A_01 + A_11 A_00)(B_00 + B_01) + (A_10 + A_00)(B_10 + B_00) + A_00 B_00 A_10 \Psi + \beta_4 ((A_00 + A_01)(B_00 + B_01) + A_00 B_00 A_10)
\]

\[
\cdots (9)
\]

From equation (9), we see that to implement \( C_0 A_4 \), we need 6 GF(2^2) multipliers and to implement \( C_1 A_4 \), we need 3 additional GF(2^2) multipliers, since the other 3 GF(2^2) multipliers are same as \( C_1 A_4 \) case[1].

3.2. Transformation between the GF(2^8) and GF(2^2) Elements

If any element in GF(2^8) field is (b0,b1, b2, b3, b4, b5, b6, b7) and it is expressed as in equation (10),

\[
A_8 = A_0 + \beta_8 A_1 = A_00 + \beta_4 A_01 + \beta_4 (A_10 + \beta_4 A_11)
\]

where \( A_0 = (y_0, y_1), A_1 = (y_2, y_3), A_00 = (y_4, y_5), A_11 = (y_6, y_7), A_01 = (z_0, z_1, z_2, z_3), A_14 = (z_4, z_5, z_6, z_7) \)

(10) then
1) From GF($2^8$) to GF($2^3$)

\[
\begin{align*}
    y_0 &= z_0 + z_1 = b_0 + b_3 \\
    y_1 &= z_3 = b_1 + b_3 + b_4 + b_6 \\
    y_2 &= z_2 = b_2 + b_3 + b_6 \\
    y_3 &= z_1 + z_2 + z_3 = b_2 + b_3 + b_4 + b_5 \\
    y_4 &= z_4 + z_5 = b_1 + b_3 + b_7 \\
    y_5 &= z_7 = b_1 + b_3 + b_4 + b_5 \\
    y_6 &= z_6 = b_1 + b_2 + b_3 + b_4 + b_5 + b_6 \\
    y_7 &= z_5 + z_6 + z_7 = b_5 
\end{align*}
\]

2) From GF($2^3$) to GF($2^8$)

From the equations (11) we get equations (12)

\[
\begin{align*}
    b_0 &= y_0 + y_5 + y_2 + y_6 \\
    b_1 &= y_1 + y_3 + y_5 + y_6 + y_7 \\
    b_2 &= y_1 + y_6 + y_7 \\
    b_3 &= y_2 + y_5 + y_6 \\
    b_4 &= y_1 + y_2 + y_3 + y_5 \\
    b_5 &= y_7 \\
    b_6 &= y_1 + y_5 + y_7 \\
    b_7 &= y_1 + y_2 + y_3 + y_4 + y_7 
\end{align*}
\]

(12)

3.3. The New GF($2^8$) Schematic Circuit

We show in Figure 2 the schematic circuit of GF($2^8$) multiplier based on GF($2^3$) field logic[5]. In the Figure, \(\Psi\) multiplier is used for GF($2^3$) field and \(\gamma\) multiplier is used for GF($2^8$) field so we need to GF($2^3$) field to GF($2^8$) field and GF($2^8$) to GF($2^3$) conversion continuously by the \(\Psi\ \gamma\) interface. As shown in the Figure 2 (b). Also in Figure 2, A, B are belong to GF($2^8$) and C is the product result of them.

(a) Top Level Block Diagram of the New GF($2^8$) Multiplier based on GF($2^3$) Logic
4. VHDL Simulation Result

Figure 3 shows the VHDL simulation result of the GF(2^8) multiplier operation using GF(2^2) multiplier and double subfield transformation. It is the result of multiplication of $\alpha_{195}$, $\alpha^2 \in GF(2^8)$ so the result is $\alpha_{197}$. Equation (13) is the corresponding GF(2^2) expansion of GF(2^8) multiplication[2].

\[
\alpha_{195} \cdot \alpha^2 = \alpha_{197}
\]

\[
= \{ ( \alpha + \beta_d(0)) + \beta_h(\alpha+ \beta_d(\alpha^2)) \} \bullet \\
\{ ( 0 + \beta_d(\alpha^2)) + \beta_h(0+ \beta_d(1)) \}
\]

\[
\ldots (13)
\]

Where all the elements RHS of the equations 13 are GF(2^2) elements.

Figure 4 also shows the VHDL simulation result of the GF(2^8) multiplier operation using GF(2^2) multiplier and double subfield transformation. It is the result of multiplication of $\alpha_{194}$, $\alpha^4 \in GF(2^8)$ so the result is $\alpha_{198}$. Equation (14) is the corresponding GF(2^2) expansion of GF(2^8) multiplication.

\[
\alpha_{194} \cdot \alpha^4 = \alpha_{198}
\]
\begin{equation}
= \{ (0 + \beta_d(0)) + \beta_e(\alpha^2 + \beta_d(\alpha^2)) \} \bullet \\
\{ (\alpha + \beta_d(\alpha)) + \beta_e(\alpha + \beta_d(1)) \}
\end{equation}

\ldots (14)

In Figure 3, we see that the speed of the GF(2^8) multiplier based on GF(2^2) field elements is faster than that of direct GF(2^8) multiplier without subfield transformation. This is because the new multiplier is operating 9 GF(2^2) multipliers parallelly so its critical path is coming from GF(2^2) multiplier not from direct GF(2^8) multiplier. The GF(2^2) expressions for \(\alpha^{195}\), \(\alpha^2 \in \text{GF}(2^8)\) are (C4) and (2E) and the GF(2^2) expression for \(\alpha^{197}\) is 22 in Radix Hexcode base.

\[\text{Figure 3. VHDL Simulation of the GF}(2^8)\text{ Multiplier Circuit based on GF}(2^2)\text{ Field Logic}\]

In Figure 4, we see also that the speed of the GF(2^8) multiplier based on GF(2^2) field elements is faster than that of direct GF(2^8) multiplier without subfield transformation from the same reason. The GF(2^2) expressions for \(\alpha^{194}\), \(\alpha^4 \in \text{GF}(2^8)\) are (0F) and (A6) and the GF(2^2) expression for \(\alpha^{198}\) is F3 in Radix Hexcode base[3].

\[\text{Figure 4. 2^{nd} VHDL Simulation of the GF}(2^8)\text{ Multiplier Circuit based on GF}(2^2)\text{ Field Logic}\]
5. Comparison between the New GF(2\(^8\)) Multiplier with Classical Multiplier without the Transformation

In Section 4 we see that the speed of the new multiplier is absolutely faster than the classical GF(2\(^8\)) multiplier. If the number of the multiplication terms for the Error correcting machines is increasing, the speed advantage becomes more profound. Here we summarizes and compares he gate counts for the 3 cases, i.e., GF(2\(^8\)) multiplier gate counts, 3 GF(2\(^4\)) multiplier gate counts and finally 9 GF(2\(^2\)) multiplier gate counts.

Here 3 GF(2\(^4\)) multipliers and 9 GF(2\(^2\)) multipliers are equivalent to 1 GF(2\(^8\)) multiplier in size aspects.

<table>
<thead>
<tr>
<th></th>
<th>9 GF(2(^2)) multipliers</th>
<th>3 GF(2(^4)) multipliers</th>
<th>1 GF(2(^8)) multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND gate</td>
<td>18</td>
<td>48</td>
<td>64</td>
</tr>
<tr>
<td>EXOR gate</td>
<td>27</td>
<td>45</td>
<td>73</td>
</tr>
<tr>
<td>Total gates</td>
<td>45</td>
<td>93</td>
<td>137</td>
</tr>
</tbody>
</table>

As we see in the Table 2, we also find that gate counts of 9 GF(2\(^2\)) multipliers case (Our new design case) is smallest comparing with the other 2 cases. So in speed aspect and cost aspect, the new design of double subfield transformation has the best characteristics of the 3 cases [4].

6. Conclusion

In this paper, we show that double transformation gives rise to the very cost effective and very high speed GF(2\(^8\)) multiplier based on GF(2\(^2\)) field logic. We show the speed advantage over the classical GF(2\(^8\)) multiplier without subfield transform in Section 4 and gatecount advantage in Section 5. The Galois GF(2\(^8\)) multiplier is widely used such as in the Error correction of digital communication channel and data encryption and decryption through the Internet data communication. In our future study, we will focus how to implement the very efficient Galois GF(2\(^8\)) divider for the same application area. Even we can imagine the triple and quadruple subfield transformation for implementing the GF(2\(^16\)) or GF(2\(^32\)) multiplier and divider.

References


Authors

Hyeong-Keon An is a Associate Professor of the Department of Information and Telecommunication engineering at the Tong Myung University in Pusan, Republic of KOREA. His research areas include Encryption/Decryption, Error control coding and OLED/LED display design. Before joining the University, he worked for SAMSUNG Electronics Co.ltd for more than 10 years.