International Journal of Control and Automation
Vol.6, No.5 (2013), pp.393-400
http://dx.doi.org/10.14257/ijca.2013.6.5.34

Very Fast and Economical GF(2^8) Divider Circuit Based On GF(2^2) Field Operation

Hyeong-Keon An

Dept. of Information Engineering, Tongmyung University, 428 sinseon-ro, Nam-gu, Busan, 608-711, Republic of Korea

hkan@tu.ac.kr

Abstract

This paper describes how to design a Galois field GF(2^8)dividing circuit using double subfield transformation[5]. The circuit is much faster and more efficient than the classical GF(2^8) divider . Multiplier circuit is used mainly for ECC encoding and for the Encripting machine while the Divider is used for ECC decoding and the Decripting machine. The divider in the paper is designed based on the GF(2^2) inversing circuit and we applied double subfield transformation. We compared the new design with the classical design in both respects, speed and cost so find the new design is much better than the classical design.

Keywords: *Galois field, Encription, Decription, Reed-Solomon, Divider, Multiplier, GF(2^8)*

1. Introduction

This paper describes a new design for dividing circuit that are used for *Reed-Solomon* (RS) Error Correcting (ECC) *Codec* and RS Endecription circuit. RS codes has great power and utility, and are today found in many applications from mobile phones to deep-space applications [1]. Since most digital devices use GF(2^8) galois elements, we also design the Divider that performs the division in the GF(2^8) field. To develop the more economical and timely faster processor we apply the double subfield transformation theory that can enable us to transform GF(2^8) galois elements to GF(2^2) galois elements and vice versa. In GF(2^2) field, every arithmetic and logical operation is much simpler and faster than in GF(2^8) field and after we get the results in GF(2^2) field , the results in the subfield are doubly transformed to the elements in the GF(2^8) field not going through the GF(2^4) field [2, 3].

In Section 2, we describe the new divider design and its structure and in Section 3, the sub part operations of the new divider are part by part analyzed. Here we find the new divider is consists of the GF(2^2) multiplier and Inversing circuit. In Section 4 the new ECC processor structure which is using the new multiplier/divider is described to prevent the data corruption during transmission through the communication channel [4]. In Section 5, we simulated the new design and showed the result in GF(2^2), GF(2^4) and GF(2^8) fields. Finally in Section 6, we describe our future work toward the continuous improvements of the ECC processor and Endecription machine.

ISSN: 2005-4297 IJCA

2. The New Divider design

Suppose that the quotient of elements A and B is D, where A,B,D∈ GF(2^8) field, and let

$$D= A/B = (a0+a1\ \beta) / (b0 + b1\ \beta) = \ d0 + d1\ \beta\ . \qquad (1)$$

where a0,a1,b0,b1,d0 and d1∈ GF(2^4).

$$d0\ = \frac{a0(b0+b1)+a1b1\gamma}{b0(b0+b1)+\gamma(b1^2)}$$

$$d1\ = \frac{a1(b0+b1)+b1(a0+a1)}{b0(b0+b1)+\gamma(b1^2)} \qquad (2)$$

Figure 2 shows a block diagram for implementing the equation (2) by using 6 multipliers, 5 adders and an inversion circuit, γ multiplier, a square and γ multiplier over GF(2^4). Now each GF(2^4) arithmetic units in the GF(2^4) field are converted to the GF(2^2) field.

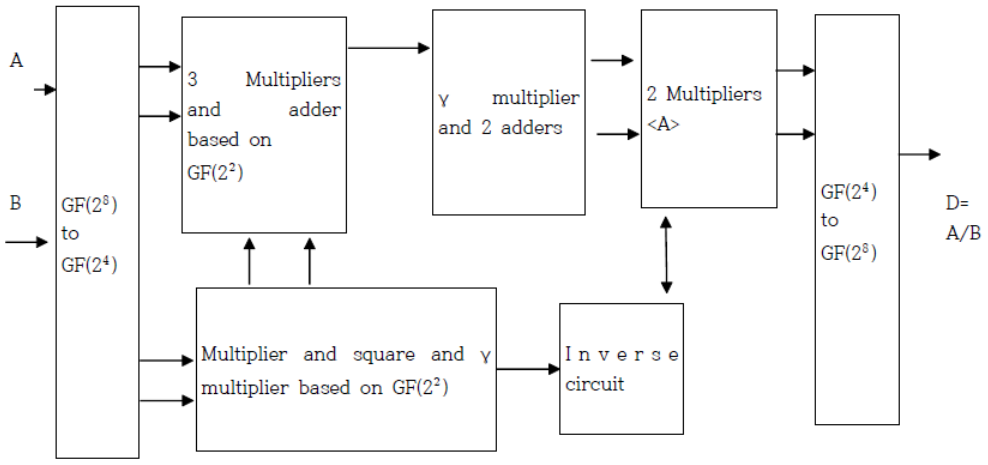

Figure 1. New GF(2^8) Divider block diagram using GF(2^4) Arithmetic units based on GF(2^2) field Logic. (<A> means GF(2^4) arithmetic machine based on GF(2^2) logic.)

In Figure 1, 3 Multipliers implement a1b1, a0(b0+b1), b0(a0+a1), using GF(2^2) multipliers, and square then γmultiplier implements $\gamma(b1^2)$ for equation (2) for equations (1) and (2), so on. All logic and arithmetic operations are performed in GF(2^2) field, therefore GF(2^4) elements are transformed to GF(2^2) elements and after processing ALU operations, GF(2^2) elements are reversely transformed to GF(2^4) elements again[6].

3. Analyzing the Subparts of the New GF(28) Divider based on the GF(22) Arithmetic and Logic Operations

In this section, we present first the transformation of the $GF(2^4)$ to and from $GF(2^2)$, secondly and thirdly $GF(2^4)$ multiplier and inverse circuit implemented using $GF(2^2)$ multiplier and inverse circuit.

i) A conversion between the $GF(2^2)$ and $GF(2^4)$ elements

This conversion is described in [ ]. So we show here just the result. A conversion from elements represented by the basis of $GF(2^2)$ into elements represented by the basis of $GF(2^4)$ is as follows,

$$b0 = z0+z1+z2+z3$$
$$b1=z1+z2+z3$$
$$b2=z2$$
$$b3=z1 \quad \ldots (3)$$

From equation (3), 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.

$$Z0= b0+b1$$
$$Z1= b3$$
$$Z2= b2$$
$$Z3= b1+b2+b3 \quad \ldots (4)$$

ii) $GF(2^4)$ multiplier using $GF(2^2)$ multiplier

The $GF(2^4)$ Multiplier using $GF(2^2)$ multiplier and Ψ multiplier is described in the paper [ ]. So we just here show the block diagram of $GF(2^4)$ multiplier based on the $GF(2^2)$ multiplier in Figure 2[5].

iii) $GF(2^4)$ inverse circuit using $GF(2^2)$ multiplier and inverse circuit.

The $GF(2^4)$ inverse circuit can be implemented using $GF(2^2)$ multiplier, inverse logic, adder, and Ψ multiplication and square logic as shown in Figure 3. We show the $GF(2^2)$ inverse table in the Table 1 and its logic circuit in the Figure 4. Also we derive $GF(2^2)$ square and Ψmultiplication logic by equation (6). Figure 5 shows the square and Ψmultiplication logic circuit.

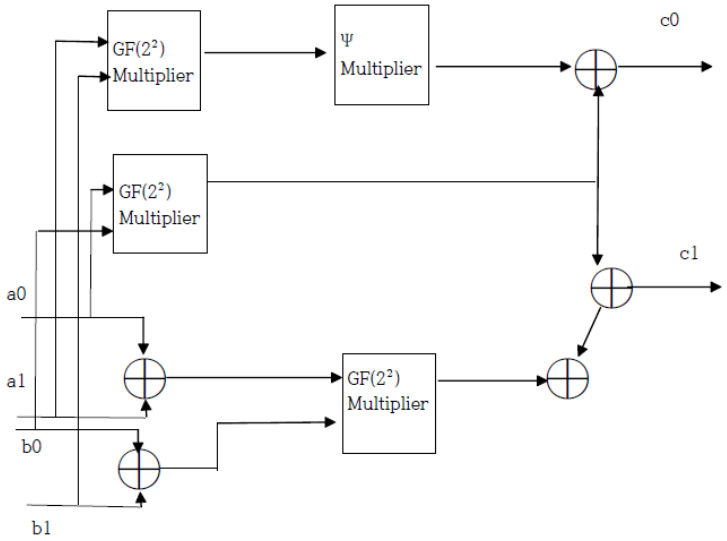

Figure 2. The GF(2^4) Multiplier block diagram using GF(2^2) multiplier, adder and Ψ multiplier

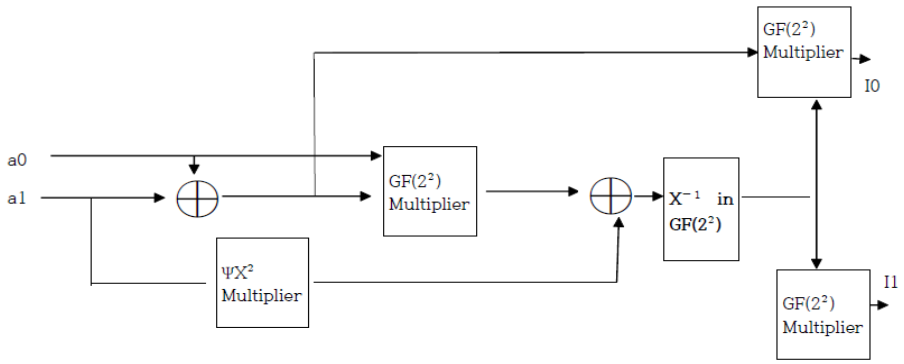

Figure 3. The GF(2^4) Inverse Logic block diagram using GF(2^2) multiplier, adder and Ψ and square circuit

In Figure 3 ,I =(I0,I1)= $\frac{1}{A}$, where A=(a0,a1), I ∈GF(2^4), and I0,I1,a0,a1 ∈GF(2^2).

GF(2^2) inverse logic is derived as follows. From the primitive function $p(\alpha)=0$, we know $\alpha^2=\alpha+1$, so we get the Table 1 for the GF(2^2) inverse logic.

Table 1. GF(2^2) Inverse elements table

α (b0,b1)	α^{-1}(I0,I1)
0 (00)	Don't care
1 (10)	1 (1,0)
α (01)	α^2(1,1)
α^2(11)	α (0,1)

From the Karnaugh mapping, we get the equation (5) and its logic circuit is depicted in Figure 4.

$$\mathbf{I_0} = (\ \overline{b_1}\ \ \mathbf{OR}\ \ \overline{b_0}\ )$$
$$\mathbf{I_1} = b_1 \qquad \ldots (5)$$

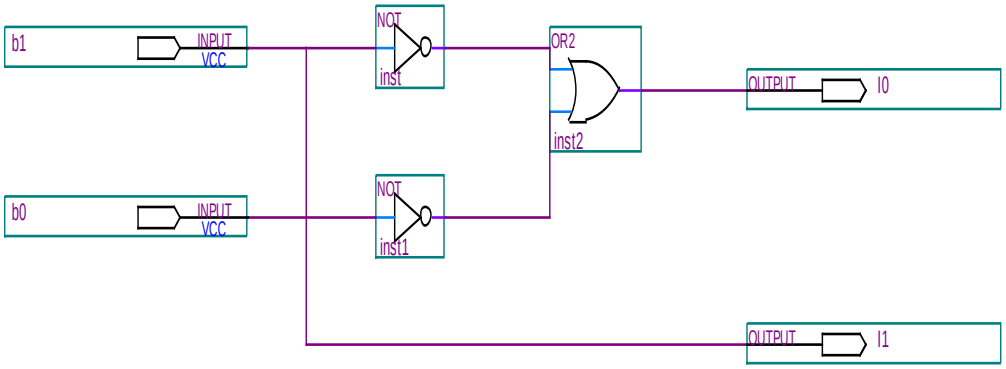

Figure 4. The GF(2^4) Inverse Logic block diagram using GF(2^2) multiplier, adder and Ψ and square circuit

Now we derive ΨX^2(square and Ψ multiplication) circuit as equation (6).

If

$\mathbf{A=a0+a1}\ \Psi$, where A, $\Psi \in GF(2^2)$ and a0,a1$\in$GF(2)

Then

$$\Psi\ \mathbf{A^2 = (a0+a1}\ \Psi^2)\ \Psi = a0\ \Psi + a1\ \Psi^3 = a0\ \Psi + a1(\Psi+1)\ \Psi$$
$$= (a0+a1)\ \Psi + a1(\Psi+1)$$
$$= a0\ \Psi + a1$$
$$= (a1,a0)$$

$\ldots$ (5)

Equation (5) is drawn in the Figure 5.

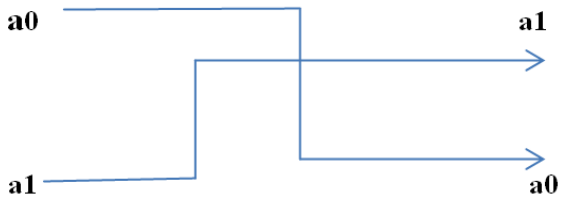

Figure 5. Ψ and square circuit (ΨX^2 circuit). (a0,a1) maps to (a1,a0)

4. New Error Correcting Code and Endecripting Processor structure based on GF(2^2) field ALU

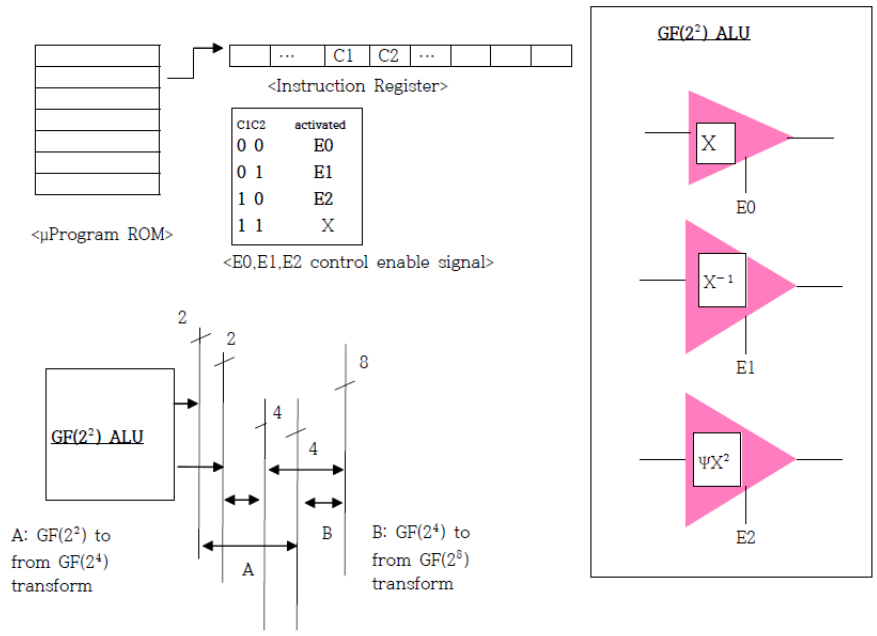

Figure 6. GF(2^8) ECC and Endecripting Processor structure based on the GF(2^2) ALU

In Figure 6 all the GF(2^8) ALU processings are performed in the GF(2^2) field and later final results are transformed to GF(2^8) elements by way of the GF(2^4) transformation. The speed and cost of the machine is much more efficient and better than the classical machine [7, 8].

5. Simulation of the New GF(2^8) Divider using the GF(2^2) ALU

5.1 Simulation result

Now let's calculate equation (6) using our new divider:

$$\frac{\alpha^2}{\alpha^{54}} = \alpha^{-52} = \alpha^{203} \quad \in GF(2^8)$$

and

$$\frac{\alpha^{251}}{\alpha^{54}} = \alpha^{197} \qquad \in GF(2^8) \qquad \ldots (6)$$

Here $\alpha^{203} = \alpha^6 + \beta\alpha^4 \quad \in GF(2^4)$

$= (\alpha + \Psi\alpha^2) + \beta(\alpha^2 + \Psi\alpha) \quad \in GF(2^2)$

and $\alpha^{197} = \alpha^5 + \beta\,\alpha^5 = (\alpha + 0\Psi) + \beta((\alpha + 0\Psi)) \qquad \ldots (7)$

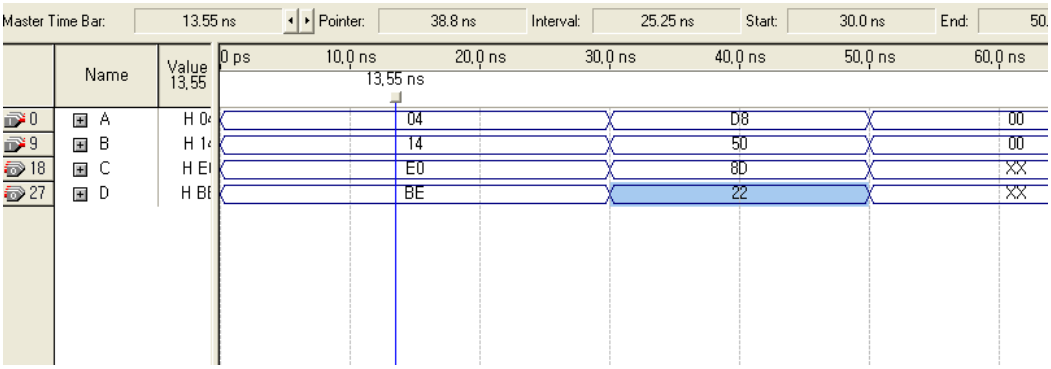

Figure 7. C=D=$\frac{A}{B}$ ∈ GF(2^8) VHDL simulation result, where D is GF(2^2) expression of C

5.2 Comparison of GF(2^4) Inverse logic gate counts between the logic based on GF(2^2) field logic and the direct logic

Table 2. Comparison of gate counts and number of TRs

	GF(2^4) inverse logic	GF(2^4) inverseUsingGF(2^2)logic
AND(MOS TRs)	**19(6)**	**6 (6)**
OR(PNMOS TRs)		**1 (6)**
Inverter(MOSTRs)		**2 (2)**
EXOR(MOS TRs)	**10(4)**	**9 (4)**
Total(MOS TRs)	**29(154)**	**18(82)**

In Table 2, we see when we use the subfield logic, gate counts and number of TRs used are greately reduced. For the speed becomes about 3 times faster when we use the double subfield transformation [7].

6. Conclusion

In the decades since their discovery, Reed Solomon codes have enjoyedcountless applications from compact discs and digital TV in livingroom to spacecraft and satellite in outer space [1]. Here we propose the new design of the GF(2^8) divider using double subfield transformation. We summarize the characteristics of the new design

1. New design using GF(2^2) logic consumes about the half of the gate and TRs used for the classical design.
2. Speed becomes about 3 times faster than the classical design.

In the near future we will design the new decrypting machine using the new divider/multiplier and want to show its wonderful efficiency.

References

[1] Payment Card Industry, “Encription Decription and Key Management within secure Cryptographic Devices”, **(2011)** September.

[2] J. Brauchle and R. Koetter, "A Systematic Reed Solomon Encoder with Arbitrary Parity positions", IEEE GLO BECOM 2009 Proceedings, **(2009)**.
[3] M. L. Cury, A. Skjellum and H. L. Ward, "Accelerating Reed-Solomon coding in RAID systems with GPUs", IEEE International Symposium in Parallel and Distributed Processing, IPDPS 2008, **(2008)**.
[4] P. Patel, "Parallel Multiplier design for Galois/Counter Mode of operation", MS Thesis, Univ. of Wateroo Canada, Dep. of EE, **(2008)**.
[5] H. K. An, "Fast and Low cost GF(2^8) Multiplier design based on Double Subfield Transformation", International Journal of Software Engineering and Its Applications, vol. 7, no. 4, **(2013)** July.
[6] D. Canright, "Avoid Mask Re-use in Galois Multipliers", Applied Math., Naval postgraduate School, Monterey CA 93943, USA, **(2008)** November 26.
[7] G. C. Cardarilli, "Analysis of Errors and Erasures in Parity Sharing RS Codecs", IEEE Transactions on Computers, vol. 56, Issue 12, (2007) December, pp. 1721-1726.
[8] Y. T. Hsu, USP7984366, "Efficient Chien search method in Reed Solomon ECC codec", **(2007)** August.

Author

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 Encription/Decription, Error control coding and OLED/LED display design. Before joining the University, he worked for AMSUNG Electronics Co. ltd for more than 10 years.

