A Kind of Design for CCSDS Standard GF(2^8) Multiplier

Wei Zhang[✉], Aihua Dong, Hao Zhang, and Dacheng Cao

Shandong Institute of Space Electronic Technology, Yantai 264003, China
wzzwl219@126.com

Abstract. Through theoretical analysis, the calculation method of dual basis multiplication in GF (2^8) field based on CCSDS Berlekamp is given. Based on this calculation method, a VLSI architecture for parallel multiplication and serial operation in circuits is proposed. At the same time, the hardware resource occupation and the timing performance of each VLSI architecture are analyzed in detail.

Keywords: CCSDS reed-solomon Code · Dual basis · Multiplication

1 Preface

In the Reed Solomon code specified in CCSDS standard, the codeword is located in GF (2^8) Galois domain and is represented by Berlekamp [1]. Because the basis components used in Berlekamp representation are not directly related to common polynomial bases, polynomial dual bases and normal bases, the multiplication of two elements in Galois field under the Berlekamp representation can not be applied to the more mature design methods under the representation of other bases, such as polynomial base multiplication, dual base multiplication, normal base multiplication and so on [2]. In general, we can transform the Berlekamp base representation to other common base representations, and the results are then converted to the Berlekamp basis after calculating. This calculation process is relatively complicated. In this paper, a method for computing dual basis is proposed, which can be multiplied directly on the Berlekamp basis. The product of the two elements is still expressed by the Berlekamp basis, which greatly simplifies the calculation process and is convenient to realize in practical circuits.

2 Theoretical Analysis

The generating polynomial of GF (2^8) field in CCSDS standard is shown in Formula 1:

$$F(x) = x^8 + x^7 + x^2 + x + 1 \tag{1}$$

The Berlekamp representation of any element Z in the GF (2^8) field is shown in Formula 2:

$$z = z_0 l_0 + z_1 l_1 + z_2 l_2 + z_3 l_3 + z_4 l_4 + z_5 l_5 + z_6 l_6 + z_7 l_7 \tag{2}$$

The basic component series $\{l_i\}$ and a group of base component series $\{u_i\}$ are dual bases, and their corresponding relationship is shown in Table 1 (a is the primitive element).

Table 1. Dual relation between $\{l_i\}$ and $\{u_i\}$

$\{l_i\}$	$\{u_i\}$
$l_0 = a^{125}$	$u_0 = a^{117\times 0} = a^0$
$l_1 = a^{88}$	$u_1 = a^{117\times 1} = a^{117}$
$l_2 = a^{226}$	$u_2 = a^{117\times 2} = a^{234}$
$l_3 = a^{163}$	$u_3 = a^{117\times 3} = a^{96}$
$l_4 = a^{46}$	$u_4 = a^{117\times 4} = a^{213}$
$l_5 = a^{184}$	$u_5 = a^{117\times 5} = a^{75}$
$l_6 = a^{67}$	$u_6 = a^{117\times 6} = a^{192}$
$l_7 = a^{242}$	$u_7 = a^{117\times 7} = a^{54}$

We know that the key to the implementation of polynomial dual base multiplier is to generate polynomials in Galois domain, which defines the iterative relationship between polynomial basis components [3]. In GF (2^8) domain, the iterative formula is shown in Formula 3:

$$a^8 = a^7 + a^2 + a + 1 \tag{3}$$

In the dual basis system $\{u_i\}$ epresented by Berlekamp, there is no iterative relationship determined by the generating polynomial, but according to the relationship between dual basis and trace [4], the following Formula 4, holds:

$$u_8 = a^{117\times 8} = a^{171} = Tr(u_{8*} l_0)u_0 + \ldots + Tr(u_{8*} l_7)u_7 \tag{4}$$

Where $Tr(.)$ is the trace function. Through calculation, the iterative relationship shown in Formula 1 can be obtained:

$$u_8 = u_7 + u_3 + u_1 + 1 \tag{5}$$

The following is the multiplication operation. Suppose:

$$a = bc \tag{6}$$

In the above formula,

$$\begin{aligned} a &= \sum a_i l_i \\ c &= \sum c_i u_i \\ b &= \sum b_i l_i \end{aligned} \tag{7}$$

There is:

$$a_i = Tr(a * u_i) = Tr(bc * u_i) = \sum_{j=0}^{7} c_j Tr(bu_i u_j) = \sum_{j=0}^{7} c_j Tr(bu_{i+j}) \tag{8}$$

When i + j $\leq$ 7,

$$Tr(bu_{i+j}) = b_{i+j} \tag{9}$$

When i + j > 7, let i + j = 8 + n, according to Formula 5 we can get the following results:

$$Tr(bu_{i+j}) = Tr(bu_{8+n}) = Tr(bu_{7+n} + bu_{3+n} + bu_{1+n} + b_n) \tag{10}$$

Formula 10 produces the following iterative relationship:

$$\begin{aligned} Tr(bu_8) = Tr(bu_{8+0}) &= b_7 + b_3 + b_1 + b_0 \\ Tr(bu_9) = Tr(bu_{8+1}) &= b_8 + b_4 + b_2 + b_1 \\ &= b_7 + b_4 + b_3 + b_2 + b_0 \\ \ldots\ldots & \\ Tr(bu_{14}) = Tr(bu_{8+6}) &= b_{13} + b_9 + b_7 + b_6 \\ &= b_7 + b_6 + b_5 + b_3 + b_2 + b_1 + b_0 \end{aligned} \tag{11}$$

It can be seen that in CCSDS Berlekamp representation, there is also an iterative relationship similar to the common polynomial base representation, which suggests that we can use a similar implementation method to polynomial basis multiplication to design multipliers.

3 Hardware Structure

In GF (2^8), "addition" corresponds to XOR operation in logic circuit, and "multiplication" corresponds to "and" operation. The multipliers used in Reed Solomon codes usually have serial structure and parallel structure.

3.1 Serial Structure

The hardware structure of the serial multiplier represented by Berlekamp is described in Fig. 1. The structure adopts the form of parallel input and serial output. Each clock cycle outputs one bit of product data. It takes 8 cycles to calculate a multiplication. The working principle of the circuit in Fig. 1 is analyzed below.

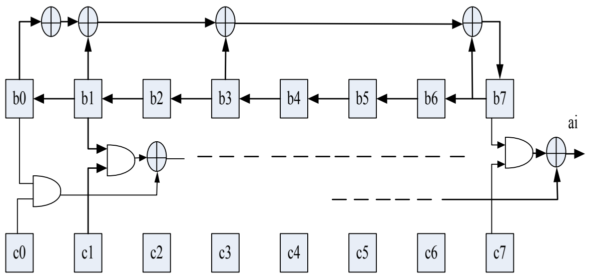

Fig. 1. Serial multiplier.

The first clock cycle: Multipliers b8 $\sim$ b0 and multipliers c8 $\sim$ c0 are input into corresponding registers in parallel. The output is:

$$a_i = b_0c_0 + b_1c_1 + \cdots + b_7c_7 = a_0 \tag{12}$$

The second clock cycle:

$$b_0^{'} = b_1, b_1^{'} = b_2, \cdots, b_6^{'} = b_7 \tag{13}$$

$$b_7^{'} = b_0 + b_1 + b_3 + b_7 \tag{14}$$

According to Formula 10, it can be calculated that:

$$b_7^{'} = b_8 \tag{15}$$

The output is:

$$a_i^{'} = c_0b_1 + c_1b_2 + \cdots + c_6b_7 + c_7b_8 = a_i \tag{16}$$

In this way, a2, a4,…, a7 can be calculated in turn. It can be seen that the circuit shown in Fig. 1 is essentially a pulsating structure.

3.2 Parallel Architecture

The structure of the parallel multiplier represented by Berlekamp is described in Fig. 2. The function of module A is to calculate b8–b14 from B0 to B7 according to the calculation method described in Eq. 8, and the function of module B is to calculate a bit of product according to Formula 6.

Taking B8 as an example, the circuit structure for calculating the bit in module A is shown in Fig. 3.

All B modules have the same circuit structure, as shown in Fig. 4.

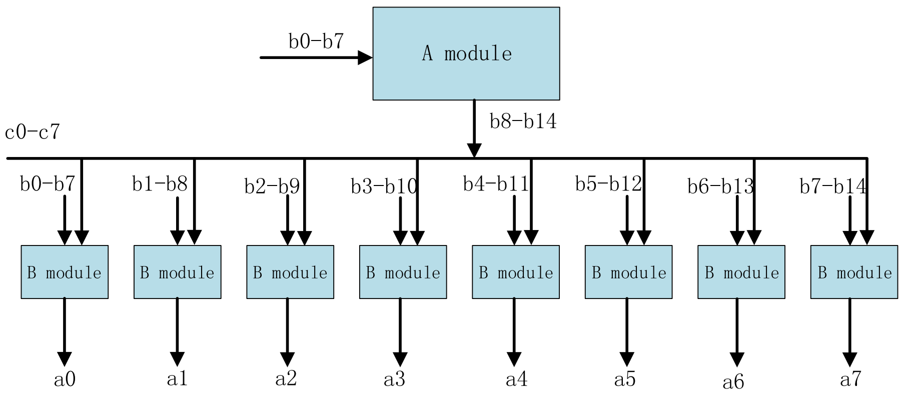

Fig. 2. Parallel multiplier

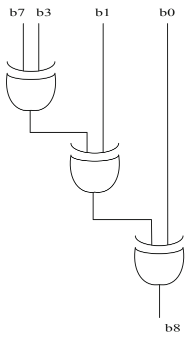

Fig. 3. Example of A module

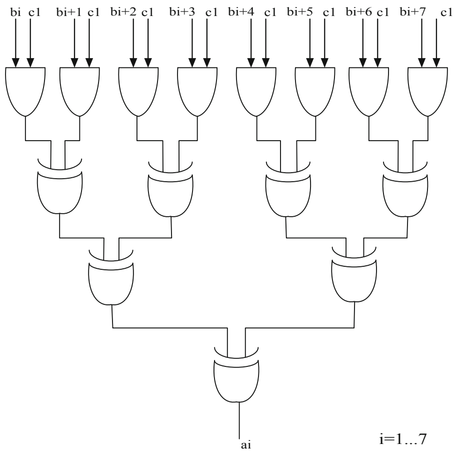

Fig. 4. Example of B module

4 Analysis of Resources and Time Sequence

Table 2 shows the resource usage and timing comparison of serial and parallel multipliers. In the table, Na is the number of two input AND gates. Da is AND gates's delay. NX is the number of two input XOR gates. DX is XOR gates's delay. Nd is the number of triggers.

Table 2. Performance analysis of serial and parallel architectures.

		Na	Nx	Nd	Delay
Serial		8	7	8	Da + 7Dx
Parallel	A module	0	26	0	6Dx
	B module	8	7	0	Da + 7Dx

The statistics of hardware resources in CCSDS dual base multiplier do not include the hardware circuits needed to convert $\{l_i\}$ basis to $\{u_i\}$ basis. This is because in CCSDS Reed Solomon code decoding scheme, the common Berlekamp Massey algorithm or Euclid algorithm uses multiple multipliers. These multipliers share the same multiplier [5] and only need one conversion.

We can see that the XOR gate resources occupied by parallel multipliers are more than 4 times that of serial multipliers. In FPGA, XOR gates are scarce resources (one XOR gate for each LE), and the overall resource consumption of parallel multipliers is relatively large. The multiplier (B0 $\sim$ B7) of serial multiplier needs to be saved for the next clock cycle, which increases the usage of 8 flip flops. In FPGA, flip flops are rich resources and will not become a limiting factor.

From the timing point of view, the parallel structure consumes more than six XOR gates' inherent delay produced by a module than the serial structure. Assuming that the delay of the parallel structure is approximately equal to that of the gate and XOR gate, it can be estimated that the signal path delay of the parallel structure is twice that of the serial structure.

5 Conclusion

The key to the design of CCSDS standard Reed Solomon encoder and decoder is the multiplier design on Galois GF (2^8). In order to reduce the resource consumption and improve the timing performance, the multiplier with simple structure and short path delay must be adopted.The design method of multiplier proposed in this paper uses the iterative relation of GF (2^8) field on the dual basis of Berlekamp basis, and adopts the structure similar to polynomial basis multiplication. Compared with the commonly used dual base multiplier, the multiplier does not need to transform from the Berlekamp base representation to the dual base representation before the multiplication. In the calculation process, the basis transformation is performed automatically, and the calculated product is directly expressed on the Berlekamp basis. The hardware structure of the base transformation is omitted and the hardware structure is greatly simplified. The serial multiplier needs 8 clock cycle processing delay, while the parallel multiplier only needs 1 clock cycle processing delay, but its resource usage is large and the timing path delay is long. In the actual using process, we can reasonably choose the serial or parallel structure according to the requirements of encoding and decoding delay clock cycle.

References

1. CCSDS 101.0-B-6: Telemetry Channel Coding, October 2002
2. Hsu, I.S., Troung, T.K., Deutsch, L.J., Reed, L.S.: A comparison of VLSI architecture of finite field multipliers using dual, normal, or standard bases. IEEE Trans. Comput. **37**(6), 735–739 (1988)
3. Fenn, S.T.J., Benaissa, M., Taylor, D.: Bit-serial Berlekamp-like multipliers for GF(2m). Electron. Lett. **31**(22), 1893–1894 (1995)
4. Wang, X., Zhen, X.: Error correcting codes - Principles and methods. Xidian University Press, Xi'an (2001)
5. Troung, T.K., Cheng, T.C.: A new decoding algorithm for correcting both erasures and errors of reed-solomon codes. IEEE Trans. Commun. **51**(3), 381–388 (2003)