Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Concurrent error detection architectures for Gaussian normal basis multiplication over GF(2^m)

Chiou-Yng Lee *

Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology, No.300, Sec.1, Wanshou Rd., Taoyuan County 333, Taiwan, ROC

ARTICLE INFO

Article history:
Received 23 August 2007
Received in revised form
7 May 2009
Accepted 18 July 2009

Keywords:
Gaussian normal basis
Concurrent error detection
Parity prediction scheme
Digit-serial multiplier

ABSTRACT

This paper presents a method of using a parity prediction scheme for detecting erroneous outputs in bit-parallel, sequential, and digit-serial Gaussian normal basis (GNB) multipliers over GF(2^m). Although all-type NB multipliers have different time and space complexities, our analytical results indicate that all-type GNB multipliers have the same structure if they use parity prediction function. For example, in the field GF(2^{233}), we have estimated that the error detection rate for a sequential multiplier is nearly 100% if a comparison is made as per clock cycle. Our analytical results also show that the area overhead of the proposed digit-serial multiplier with concurrent error detection does not exceed 5%. Several efficient parity prediction techniques will be shown in this work to provide a low overhead solution to concurrent error detection particularly when the cryptography implementations using GF(2^m) multiplier require higher reliability and the protection against adversarial attacks.

1. Introduction

Arithmetic operations in finite (Galois) fields have received attention because of their important and practical applications in the areas of error correcting code, digital signal processing and public key cryptosystems [1–3]. In the basic arithmetic operations over GF(2^m), multiplication is the most important, complex, and time-consuming operation. In the binary field GF(2^m), there are three major representations, including polynomial basis, dual basis, and normal basis. Various GF(2^m) multipliers have received the most attention in the literature [4–9]. For cryptographic applications [10,11], the field size can be in the range of 160–2048 bits. Considering the structure of bit-parallel multipliers applied in cryptographic devices, the circuit space requires more than a million transistors, and it may lead to some kinds of computation faults in the field. Therefore, an efficient multiplier with concurrent error detection (CED) capability is required to improve the reliability of cryptographic operations.

The side-channel attacks using the techniques of power analysis and differential fault analysis will typically deliberate fault injection into cryptographic devices. These devices require a small amount of side-channel information to break common ciphers. Boneh et al. [12] firstly announced a fault-based side-channel attack against asymmetric cryptosystems. Biham and Shamir [13] developed a differential fault analysis that exploits faulty computing to find cryptographic keys for breaking Data Encryption Standard (DES) later. Anderson and Kuhn [14] showed that the fault-based side-channel attacks are also available for software implementations of encryption algorithms. Messerges et al. [15] proposed effective power analysis attacks for modular exponentiation algorithms. Furthermore, Coron [16] extended differential power analysis attacks to Elliptic Curve Cryptosystems (ECC) and even any scalar multiplication algorithms. The faulty outputs of cryptographic devices can lead to an active attack. Hence, effective and simple methods for protecting the encryption/decryption circuitry from an attacker are required to ensure that cryptographic devices can output accurate signatures. Many error detection schemes have been developed for symmetrical [17,18] and asymmetrical cryptosystems [19,20] to output confirmed results.

The parity prediction scheme is basically popularized to monitor the behavior of a circuit during its normal operation and to indicate the deviations from correct functioning. Quality assessment of concurrent test methods relies on several factors, including the model of detectable errors, the worst-case detection latency, and the incurred area overhead. In the progress of fault-tolerant cryptographic computations over finite fields, several CED methods have been proposed for both bit-parallel and bit-serial circuits. Fenn et al. [21] proposed bit-serial multipliers in GF(2^m) using a parity prediction scheme in which the fields are defined using irreducible all-one polynomials. Unfortunately, irreducible all-one polynomials are very rare. To overcome this problem, Reyhani-Masoleh and Hasan [22] provided error detection methods for bit-parallel and bit-serial polynomial basis multipliers in the generic field of GF(2^m). A bit-parallel systolic dual basis multiplier with CED was then presented by Lee et al. [23].

The major advantage of the normal basis (NB) multiplication is that the squaring of the element in GF(2^m) could be implemented simply through the right cyclic shifting of its coordinates. Various

* Tel.: +886 2 82093211x5710; fax: +886 2 82093211x5707
E-mail address: PP010@mail.lhu.edu.tw

0167-9260/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.vlsi.2009.07.002

NB multipliers over GF(2^m) have been developed in [24–26] depending on the key function to implement efficient bit-parallel and bit-serial architectures. The other approach is suggested by Feisel et al. [27]. This scheme uses a special primitive element called a Gauss period to generate a normal basis of GF(2^m). This normal basis is called the Gaussian normal basis (GNB), which exists for every positive integer m not divisible by 8. Many standards, such as ANSI X9.62 [28], FIPS 186-2 [11] and IEEE Standard 1363–2000 [10], include the GNB concept. In order to achieve efficient hardware implementations, Trujillo et al. [29] present a hardware design for the GNB multiplier over GF(2^{163}). In [30], Reyhani-Masoleh proposed GF(2^m) multiplication algorithms and architectures using type-t GNBs. To the best of our knowledge, no previous article addressed the issue of error detection in a GNB multiplier. This article thus firstly tries to investigate the fault detection architectures in a GNB multiplication over GF(2^m). In this paper, the proposed sequential, bit-parallel, and digit-serial multipliers with CED capability adopt the parity prediction scheme. Analytical results show that the extra area overhead for the proposed digit-serial multiplier with concurrent error detection does not exceed 5%. Efficient parity prediction techniques are given to provide a low overhead solution to concurrent error detection in the situation of cryptography implementations using GF(2^m) multiplier requiring higher reliability and protection from adversarial attacks.

The organization of this article is as follows. In Section 2, Gaussian normal basis representation and its multiplication algorithm are discussed. Section 3 presents bit-parallel and sequential multipliers. Fault detection architectures in GNB multipliers are then considered in Section 4. Section 5 introduces the proposed digit-serial GNB multiplier using multiple parity prediction schemes. In Section 6, we will analyze the probability of error detection and the time/space overheads. Finally, Section 7 offers some concluding remarks.

2. Preliminaries

2.1. Gaussian normal basis representation

It is commonly known that the finite field GF(2^m) can be viewed as a vector space of dimension m over GF(2). If the trace function $tr(\alpha)$ is satisfied by $tr(\alpha)=\sum_{i=0}^{m-1}\alpha^{2^i}=1$, α is then called a normal element of GF(2^m). For $A\in$GF(2^m), we can represent as $A=a_0\alpha+a_1\alpha^2+\cdots+a_{m-1}\alpha^{2^{m-1}}=(a_0,a_1,\cdots,a_{m-1})$, where the coordinates $a_i\in$GF(2) for $0\le i\le m-1$, and $N_1=\{\alpha,\alpha^2,\ldots,\alpha^{2^{m-1}}\}$ is the normal basis (NB) of GF(2^m). In hardware implementation, the squaring of A can be easily performed by a right cyclic shift, i.e., $A^{2^i}=(a_{m-i},a_{m-i-1},\ldots,a_{m-i+1})$.

Definition 1 ((*Feisel et al. [27]*).). Let $p=mt+1$ be a prime number and $gcd(mt/k,m)=1$, where k denotes the multiplicative order of 2 module p. Let γ be a primitive $(mt+1)$th root of unity in some extension field of Fp. A Gauss period of type (m,t) over Fp is defined as $\alpha=\gamma+\gamma^{2^m}+\cdots+\gamma^{2^{m(t-1)}}$.

By employing α in Definition 1, we generate a normal basis $N_1=\{\alpha,\alpha^2,\ldots,\alpha^{2^{m-1}}\}$ for GF(2^m). Such a normal basis is called the Gaussian normal basis of type (m,t), denoted by GNB of type t. Significantly, GNBs exist for GF(2^m) whenever m is not divisible by 8. Each element $A=a_0\alpha+a_1\alpha^2+\cdots+a_{m-1}\alpha^{2^{m-1}}$ of GF(2^m) can also be given as

$$A=a_0(\gamma+\gamma^{2^m}+\cdots+\gamma^{2^{m(t-1)}})+a_1(\gamma^2+\gamma^{2^{m+1}}+\cdots+\gamma^{2^{m(t-1)+1}})+\cdots+a_{m-1}(\gamma^{2^{m-1}}+\gamma^{2^{2m-1}}+\cdots+\gamma^{2^{mt-1}}) \quad (1)$$

Observing the above equation, the normal basis N_1 could be converted by the set $N_2=\{\gamma,\gamma^{2^m},\ldots,\gamma^{2^{m(t-1)}},\gamma^2,\ \gamma^{2^{m+1}},\ldots,\gamma^{2^{m(t-1)+1}},\ldots,\ \gamma^{2^{m-1}},\gamma^{2^{2m-1}},\ldots,\gamma^{2^{mt-1}}\}$. From $\gamma^p=1$, the set N_2 can then be translated into the redundant basis $N_3=\{\gamma,\gamma^2,\ldots,\gamma^{p-1}\}$. For example, type-1 GNB can be represented by the set $\{\gamma,\gamma^2,\ldots,\gamma^m\}$. Therefore, from the redundant basis representation, the normal basis element $A=(a_0,a_1,\ldots,a_{m-1})$ can be represented by

$$A=a_{F(1)}\gamma+a_{F(2)}\gamma^2+\cdots+a_{F(p-1)}\gamma^{p-1} \quad (2a)$$

where

$$F(2^i2^{mj}\ mod\ p)=i,0\le i\le m-1,0\le j\le t-1. \quad (2b)$$

The function $F(x)=i$ in (2b) is mapped from x into i if x is satisfied by the form 2^i2^{mj}. For example, let $m=5$ and $t=2$, then $p=11$ is prime. The value $x=10$ can be represented by the form $x=2^02^5$ mod 11; thus, we have $F(10)=0$. Analogically, $F(1)=0$, $F(2)=1$, $F(3)=3$, $F(4)=2$, $F(5)=4$, $F(6)=4$, $F(7)=2$, $F(8)=3$ and $F(9)=1$. Using (2b), the basis conversion from the GNB to the NB is described in the following two steps [39]:

Step 1. $(a_0,\ldots,a_0,\ldots,a_{m-1},\ldots,a_{m-1})\leftarrow(a_{F(1)},a_{F(2)},\ldots,a_{F(p-1)})$,
Step 2. $(a_0,a_1,\ldots,a_{m-1})\leftarrow(a_0,\ldots,a_0,\ldots,a_{m-1},\ldots,a_{m-1})$.

2.2. Conventional GNB multiplication

Let $A=(a_0,a_1,\ldots,a_{m-1})$ and $B=(b_0,b_1,\ldots,b_{m-1})$ indicate two normal basis elements in GF(2^m), and $C=(c_0,c_1,\ldots,c_{m-1})\in$GF($2^m$) represent their product, i.e., $C=AB$. Assume that both elements A and B are represented by the GNB if the sequence $F(1)$, $F(2)$, …, $F(p-1)$ is defined by (2b), and $p=mt+1$ is prime. The first coordinate of product C can be calculated using the following formula [10]:

$$c_0=\sum_{k=1}^{p-2}a_{F(k+1)}b_{F(p-k)} \quad (3)$$

According to $\gamma^p=1$, the squaring of an element A in (1) is carried out by a simple permutation. From the GNB representation in (1), we can use a simple permutation to obtain the redundant basis in (2). Thus, the squaring of an element A in (2) is given by

$$A^2=\sum_{i=1}^{p-1}a_{F(i^2\ mod\ p)}\gamma^i. \quad (4)$$

Using the function $F(x)$ in (2b), if $F(x)=k$, we will have

$$F(x^2\ mod\ p)=k+1\ mod\ m. \quad (5)$$

For example, let $m=5$ and $t=2$, we have $F(9)=1$ and $F(9^2\bmod 11)=F(7)=2$. In (3), c_0 is the first coordinate of product $C=AB$. Applying (4) and (5), c_1, the first coordinate of product $C^2=A^2B^2$, is obtained by cycling the subscripts modulo m in the formula for c_0. Therefore, applying (3), a type-t GNB multiplication for an even t is addressed as follows.

Algorithm 1. (type-t GNB multiplication for t even) [10]
Input: $A=(a_0,a_1,\ldots,a_{m-1})$ and $B=(b_0,b_1,\ldots,b_{m-1})\in$GF($2^m$), and $Q(X,Y)=\sum_{k=1}^{p-2}x_{F(k+1)}y_{F(p-k)}$
Output: $C=(c_0,c_1,\ldots,c_{m-1})=AB$

1. $X=A$ and $Y=B$
2. for $j=0$ to $m-1$ {
3. $\quad c_j=Q(X,Y)$
4. $\quad X=X\ll 1$ and $Y=Y\ll 1$
5. }
6. output $C=(c_0,c_1,\ldots,c_{m-1})$.