Low Complexity Multiplication in a Finite Field Using Ring Representation

Rajendra Katti, *Member*, *IEEE*, and Joseph Brennan

Abstract—Elements of a finite field, $GF(2^m)$, are represented as elements in a ring in which multiplication is more time efficient. This leads to faster multipliers with a modest increase in the number of XOR and AND gates needed to construct the multiplier. Such multipliers are used in error control coding and cryptography. We consider rings modulo trinomials and 4-term polynomials. In each case, we show that our multiplier is faster than multipliers over elements in a finite field defined by irreducible pentanomials. These results are especially significant in the field of elliptic curve cryptography, where pentanomials are used to define finite fields. Finally, an efficient systolic implementation of a multiplier for elements in a ring defined by $x^n + x + 1$ is presented.

Index Terms—Finite field multiplication, ring representation, systolic arrays.

1 Introduction

Finite fields play an important role in coding theory and public-key cryptography [1]. Coding theory has applications in error-free communications and data storage. Public-key cryptography has applications in smart-card technology, e-commerce security, and internet security [2], [3], [4], [5]. Both coding theory and public-key applications use algorithms based on arithmetic of elements over an extension of a field GF(2) denoted by GF(2^k). Public-key cryptographic algorithms [6], [7] usually require words of very long length (greater than 160 bits). This leads to low performance cryptographic algorithms, making them impractical for commercial applications. To overcome this deficiency, fast hardware architectures for performing arithmetic in Galois fields GF(2^k) needs to be designed. The arithmetic operations normally performed are addition, multiplication, squaring, exponentiation, and inversion. This paper focuses on the multiplication of elements in a Galois field GF(2^k). Finite field multipliers can be expensive, both in terms of gate count and delay.

Multipliers can be classified into bit-parallel and bit-serial multipliers [8], [9], [10], [11], [12]. Bit-parallel multipliers compute the result in one clock cycle but generally have an area requirement proportional to k^2, where the elements that are multiplied belong to GF(2^k). Bit-serial multipliers compute the result in k clock cycles, but have an area requirement proportional to k. Multipliers can also be classified based on whether the finite field elements are defined by a normal basis or a canonical basis. One advantage of the normal basis representation of field elements is that the squaring of an element can be computed by the cyclic shift of the binary representation. However, canonical bases are widely used and lead to efficient implementations of multipliers. The time and space complexities of bit-parallel canonical basis multipliers are much better than that of multipliers based on the normal basis. In this paper, we present a new ring representation of field elements to design new bit-parallel, canonical basis multipliers that are better in time complexity and almost as good in space complexity compared to existing multipliers. The elements of a field are mapped to elements in a ring that results in extra bits being required in the representation of the elements of the field. This results in a modest increase in space complexity. However, this also results in faster multipliers as the ring elements are defined using simpler polynomials over the finite field GF(2).

The elements of a finite field are represented as binary words according to one of many traditionally available representations. We propose a new representation of the elements of a finite field inspired by [11]. In some cases, this representation leads to faster multipliers compared to other representations. A finite field, F_{2^m}, is defined by an irreducible polynomial, $f(x)$, of degree m and each element in the field can be represented as an m-bit binary number or a polynomial of degree $(m-1)$. Multiplication of two elements in a finite-field implies multiplication of the two polynomials that represent the finite-field elements and then taking the modulus of this product with respect to $f(x)$. The complexity of multiplication is dependent on the number of terms in the polynomial $f(x)$. The fewer the terms the lower is the complexity.

Let $(\gamma_0, \gamma_1, \gamma_2, \dots, \gamma_{m-1})$ be a basis of F_{2^m} over F_2. An element F_{2^m} is represented by the m-bit word $(\alpha_0, \alpha_1, \dots, \alpha_{m-1})$, $\alpha_i \in F_2$, corresponding to the expansion of α in the basis $(\gamma_0, \gamma_1, \gamma_2, \dots, \gamma_{m-1})$: $\alpha = \sum_{i=0}^{m-1} \alpha_i \gamma_i$. A canonical basis (also called the standard basis) is of the form $(1, \gamma, \gamma^2, \gamma^3, \dots, \gamma^{m-1})$, where γ is a root in F_{2^m} of a binary irreducible polynomial f(x) of degree m. Elements of F_{2^m} take the form of polynomials in γ of degree (m-1) or less. The multiplication of two elements of F_{2^m} is achieved by multiplication of the corresponding polynomials in γ

- *R. Katti is with the Department of Electrical and Computer Engineering, North Dakota State University, Fargo, ND 58105. E-mail: rajendra.katti@ndsu.nodak.edu.*
- *J.P. Brennan is with the Department of Mathematics, North Dakota State University, Fargo, ND 58105. E-mail: joseph.brennan@ndsu.nodak.edu.*

Manuscript received 15 may 2002; revised 25 Nov. 2002; accepted 25 Nov. 2002.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number 117861.

followed by a reduction modulo $f(\gamma)$. Canonical bases are widely used to implement multipliers. Another basis, called the normal basis, is of the form $(\gamma, \gamma^2, \gamma^4, \gamma^8, \ldots, \gamma^{2^{m-1}})$ for some element γ in F_{2^m}. Representing field elements with a normal basis leads to efficient implementation of squaring of an element (squaring is a circular shift of the binary representation of a field element). However, parallel multipliers are not as efficient as multipliers using the canonical basis.

In [11], a new representation of elements of finite fields F_{2^m} was proposed. This representation was based on the isomorphism from F_{2^m} into the ring $F_2[x]/(x^n+1)$ for some n, with gcd(n, 2) = 1. The polynomial ring, $F_2[x]/(x^n+1)$, contains an isomorphic copy of F_{2^m} only if (x^n+1) has an irreducible factor h(x) such that deg(h(x)) = m. Let $(x^n+1) = \mathrm{g(x)h(x)}$. Then, n = deg(g(x)) + deg(h(x)) = deg(g(x)) + m. Let the degree of g(x) be r. Therefore, n = r + m. The number of bits in the new representation has now increased from m to n = m + r. However, since the multiplication in the ring is performed modulo (x^n+1), it turns out that multiplication in the ring is faster than multiplication modulo some degree m polynomial only if r is small. A polynomial of the form $f(x) = f_0 + f_1x + f_2x^2 + \cdots + f_nx^n$ over GF(2) ($f_i \in GF(2)$) is called an all-1 polynomial (AOP) of degree n if $f_i = 1$ for $i = 0, 1, 2, \cdots, n$. If r = 1, then h(x) is the all-1 polynomial and the ring representation is faster than any other representation with a modest increase in the number of XOR and AND gates. However, irreducible all-1 polynomials of degree m exist if and only if (m + 1) is prime and 2 is a generator of the field GF(m + 1). Whenever an irreducible all-1 polynomial cannot be found, then the smallest n such that (x^n+1) has an irreducible factor of degree m must be found in order for the ring $F_2[x]/(x^n+1)$ to have a copy of the field F_{2^m}. In such a case, n may turn out to be much larger than m. For example, for m = 9, the minimum value of n such that (x^n+1) has a factor of degree 9 is 73. This makes the field-elements of F_{2^9} have a 73-bit representation in the ring $F_2[x]/(x^{73}+1)$, thus making the multiplier that uses this representation very complex both in terms of time and circuit complexity. In this paper, we propose new rings in which we can find an isomorphic copy of F_{2^m}. The new rings are of the form $F_2[x]/(x^n+x^k+1)$ or $F_2[x]/(x^n+x^{k_1}+x^{k_2}+1)$ $(m < n)$. The multipliers that result are faster if k and n are as small as possible [16]. We use a well-known result (Wedderburn's Theorem) in algebra and the software Mathematica to help us find the required rings. These new rings are useful if the least number of terms in an irreducible polynomial is 5 and the ring $F_2[x]/(x^n+1)$ does not work due to reasons mentioned above. For example, consider m = 27. There is no irreducible trinomial of degree 27 and the least term irreducible polynomial of degree 27 is a pentanomial. One such pentanomial is $(x^{27}+x^5+x^2+x+1)$. However, the ring $F_2[x]/(x^{29}+x+1)$ contains an isomorphic copy of the field $F_{2^{27}}$. The ring $F_2[x]/(x^n+1)$, where n = 73, also contains an isomorphic copy of the field $F_{2^{27}}$. Multiplication in the ring $F_2[x]/(x^{29}+x+1)$ is more efficient than multiplication in the ring $F_2[x]/(x^{73}+1)$ because multiplication of 29-bit field elements modulo a trinomial is faster than multiplication of 73-bit elements modulo a binomial. Besides, the number of gates needed for 73-bit elements is too high.

The rest of the paper is organized as follows: Section 2 describes polynomial ring representations and Wedderburn's Theorem. Section 3 describes some new ring representations based on Wedderburn's Theorem that improve multiplication complexity. Section 4 describes and compares the complexity of the multiplier resulting from the new ring representations with other multipliers. Section 5 describes a systolic implementation of a multiplier for elements in the ring modulo x^n+x+1 and Section 6 concludes the paper. The advantage of a systolic implementation is the ease of implementation in VLSI. The complexity of the systolic multiplier presented is the same as any other implementation of the multiplier.

2 Polynomial Ring Representation

We begin this section by stating Wedderburn's Theorem and how it can be used to obtain a new representation for field elements. The Wedderburn Theorem in the form required for this paper states that if $f(x)$ is a polynomial over the field k with no repeated roots (that is $gcd(f(x), f'(x)) = 1$), then the ring $R = k[x]/f$ decomposes into a direct sum of ideals of R. Therefore, $R = I_1 \oplus I_2 \oplus \cdots \oplus I_r$. Each of these ideals is generated by a single element and is isomorphic to a finite field extension K of k. A more general version of Wedderburn's Theorem can be found in [13]. The decomposition of $R = k[x]/f$ into ideals is effectuated by the construction of the system of generators for these ideals. Such a system of generators $\{e_1, e_2, \ldots, e_r\}$ will consist of a complete orthogonal system of idempotents. In R, one has $e_i^2 = e_i, e_ie_j = 0$ for $i \neq j$, $e_1 + e_2 + \cdots + e_r = 1$ and the ideals $I_i = \mathrm{R}e_i$, are all isomorphic to fields. At this point, there is great advantage to working over fields of characteristic two. In characteristic two, the Frobenius map on R, the map

$$F : R = k[x]/f \rightarrow k[x]/f = R; \quad F : a \mapsto a^2,$$

is a linear transformation of k-vector spaces. As each of the ideals in the decomposition is stable under the Frobenius map, the complete orthogonal system of idempotents providing the decomposition is realized by the eigenvectors of the Frobenius. In general characteristic, the problem of computing the idempotents is more problematic [14].

Let f(x) be a polynomial of degree n over GF(2), that can be factorized into r, distinct, irreducible factors as follows:

$$f(x) = f_1(x)f_2(x)\ldots f_r(x).$$

Let the degrees of $f_i(x)$ be p_i, then Wedderburn's Theorem [13] states that the ring $F_2[x]/f(x)$ is a direct sum of fields $F_2[x]/f_i(x)$. Therefore:

$$\begin{aligned} F_2[x]/f(x) &= F_2[x]/f_1(x) \oplus F_2[x]/f_2(x) \oplus \ldots \oplus F_2[x]/f_r(x) \\ &= F_{2^{p_1}} \oplus F_{2^{p_2}} \oplus \ldots \oplus F_{2^{p_r}}. \end{aligned}$$

Here, $F_2[x]/f(x)$ is a polynomial ring modulo f(x) over GF(2) and $F_2[x]/f_i(x) = F_{2^{p_i}}$ is the finite extension field over GF(2) modulo $f_i(x)$. Wedderburn's Theorem also gives a procedure to map the ring elements to field elements. This

TABLE 1
Ring Elements and the Corresponding Field Elements

$F_2[x]/(x^3+1)$	$(x^2+x)\times F_2[x]/(x^3+1)$	$(x^2+x+1)\times F_2[x]/(x^3+1)$
0	0	0
1	x^2+x	x^2+x+1
x	x^2+1	x^2+x+1
x^2	$x+1$	x^2+x+1
$x+1$	$x+1$	0
x^2+1	x^2+1	0
x^2+x	x^2+x	0
x^2+x+1	0	x^2+x+1

is because each of the fields $F_2[x]/f_i(x) = F_{2^{p_i}}$ is generated by an idempodent, e_i, that belongs to the set of orthogonal idempotents. Therefore:

$$F_2[x]/f(x) = (F_2[x]/f(x) * e_1) \oplus (F_2[x]/f(x) * e_2) \oplus \dots \oplus (F_2[x]/f(x) * e_r).$$

Idempotents, e_i, can always be found by solving the linear system of equations $e_i^2 = e_i$. From these idempotents, we can choose a subset that are orthogonal. Therefore, each $F_2[x]/f(x) * e_i$ is isomorphic to $F_{2^{p_i}}$. Elements of the ring $F_2[x]/f(x)$ can be expressed as r-tuples $(a_1, a_2, \cdots, a_r)$, where each $a_i \in F_2[x]/f(x) * e_i$. To perform an add or multiply operation on two r-tuples, we simply perform the operation on their individual elements.

We now consider an example that illustrates the above facts. Let $f(x) = x^3 + 1$. Since

$$x^3 + 1 = (x+1)(x^2+x+1) = f_1(x)f_2(x),$$

this implies that the following is true from Wedderburn's Theorem.

$$F_2[x]/(x^3+1) = F_2[x]/(x+1) \oplus F_2[x]/(x^2+x+1) = F_2 \oplus F_{2^2}.$$

Therefore, an isomorphic copy of $F_2[x]/(x+1)$ and $F_2[x]/(x^2+x+1)$ can be found in the ring $F_2[x]/(x^3+1)$. The two idempotents we are looking for, e_1 and e_2, are (x^2+x) and (x^2+x+1), respectively. They satisfy the following properties: $e_1 + e_2 = 1, e_1e_2 = 0, e_1^2 = e_1, e_2^2 = e_2$. Table 1 lists the elements of $F_2[x]/(x^3+1)$, $(x^2+x) \times F_2[x]/(x^3+1)$, and $(x^2+x+1) \times F_2[x]/(x^3+1)$. Note that $F_2[x]/(x^3+1)$ contains eight elements, $(x^2+x) \times F_2[x]/(x^3+1)$ contains four distinct elements, and $(x^2+x+1) \times F_2[x]/(x^3+1)$ contains two distinct elements. Columns two and three of the table are obtained by multiplying (modulo (x^3+1)) the elements of column 1 by $e_1 = (x^2+x)$ and $e_2 = (x^2+x+1)$, respectively. The distinct elements of $(x^2+x) \times F_2[x]/(x^3+1)$ are isomorphic to F_{2^2}. The multiplicative identity is the idempotent $e_1 = (x^2+x)$. The distinct elements of $(x^2+x+1) \times F_2[x]/(x^3+1)$ are isomorphic to F_2. The multiplicative identity is $e_2 = (x^2+x+1)$. An element $a \in F_2[x]/(x^3+1)$ can be written as a 2-tuple (a_1, a_2), where $a_1 = a \times e_1, a_1 \in e_1 \times F_2[x]/(x^3+1)$ and $a_2 = a \times e_2, a_2 \in e_2 \times F_2[x]/(x^3+1)$. The sum of $a, b \in F_2[x]/(x^3+1)$ is $a + b = (a_1 + b_1, a_2 + b_2)$. The product of two elements can be defined in a similar manner.

For example, from Table 1, we see that $x^2 \in F_2[x]/(x^3+1)$ can be written as $((x+1), (x^2+x+1))$ and $1 \in F_2[x]/(x^3+1)$ can be written as $((x^2+x), (x^2+x+1))$ and the sum of 1 and x^2 can be written as

$$((x+1) + (x^2+x), (x^2+x+1) + (x^2+x+1)) = (x^2+1, 0).$$

In [11], multiplication in a field F_{2^m} was performed by finding a ring $F_2[x]/(x^n+1)$ which contains an isomorphic copy of F_{2^m}. In other words, $f(x)$ is chosen to be (x^n+1) and one of its irreducible factors must be of degree m. Therefore, if (x^n+1) can be factorized as $(x^n+1) = f_1(x)f_2(x)\cdots f_r(x)$ and degree $(f_r(x)) = m$, then

$$F_2[x]/(x^n+1) = F_2[x]/f_1(x) \oplus F_2[x]/f_2(x) \oplus \dots \oplus F_2[x]/f_r(x) = F_{2^{p_1}} \oplus F_{2^{p_2}} \oplus \dots \oplus F_{2^{p_r}}.$$

Note that degree $(f_i(x)) = p_i$ and $p_r = m$. This implies that multiplication in $F_2[x]/f_r(x)$ can be performed in the ring $F_2[x]/(x^n+1)$. This method in [11] relies on the fact that finding the modulus with respect to (x^n+1) is computationally more efficient compared to finding the modulus with respect to $f_r(x)$. However, $n = p_1 + p_2 + \cdots + p_r$. Therefore, the number of bits in a ring element has $p_1 + p_2 + \cdots + p_{r-1}$ more bits than a field element of F_{2^m} which has $p_r = m$ bits. If $p_1 + p_2 + \cdots + p_{r-1}$ is large, then multiplication in the ring is expensive (too many gates) and slow, thus nullifying the advantage of computing the modulus with respect to a simple polynomial (x^n+1). For example, if m = 19, then $p_1 + p_2 + \cdots + p_{r-1} = 524,287$ and, thus, this technique in [11] fails to be effective. The method in [11] is effective only when $f(x) = f_1(x)f_2(x)$ and $f_1(x) = (x+1)$ and degree $(f_2(x)) = m$. This implies that $f_2(x)$ is an all-one polynomial and such polynomials are few in number. It is hard to find AOPs because a polynomial of degree n is AOP if and only if n + 1 is prime and 2 is a generator of GF(n + 1). In the next section, we suggest new rings that overcome this problem.

3 New Ring Representation

In cryptographic applications, fields based on low weight irreducible polynomials are desired. The weight of a polynomial is the number of terms in it. For example, the weight of $(x^7 + x^3 + x + 1)$ is 4. About half the irreducible, minimum weight polynomials of degree less than 10,000 have weight 3 (these are called trinomials) and the rest have weight 5 (these are called pentanomials). The new ring representations proposed by us are useful in cases where multiplication of field elements has to be performed in a field defined by a pentanomial. The new rings are $F_2[x]/(x^n + x^k + 1)$ and $F_2[x]/(x^n + x^{k_1} + x^{k_2} + 1)$. In each of these rings, we find an isomorphic copy of the field defined by pentanomials. Multiplication in the rings is faster than multiplication in a field defined by pentanomials because the new rings are based on 3 and 4-term polynomials. Let $f_2(x)$ be an irreducible polynomial of degree m. In what follows, we find polynomials, $f_2(x)$ such that either $x^n + x^k + 1 = f_1(x) \times f_2(x)$ or $x^n + x^{k_1} + x^{k_2} + 1 = f_1(x) \times f_2(x)$ is satisfied. In both cases, the degree of $f_1(x)$ must be as low as possible in order to make multiplication in rings based on 3-term and 4-term polynomials faster than multiplication in fields based on pentanomials. The two new rings satisfy the following (Wedderburn's Theorem):

$$F_2[x]/(x^n + x^k + 1) = F_2[x]/f_1(x) \oplus F_2[x]/f_2(x)$$

or

$$F_2[x]/(x^n + x^{k_1} + x^{k_2} + 1) = F_2[x]/f_1(x) \oplus F_2[x]/f_2(x).$$

Note that, in each of the above equations, $f_1(x)$ is different. Therefore, the procedure to find a ring which contains an isomorphic copy of a field, F_{2^m}, in which multiplication must be done, is as follows: Assume that no irreducible trinomial exists of degree m.

1. If an irreducible AOP of degree m exists, then perform the multiplication in $F_2[x]/(x^n + 1)$, where n = m + 1.
2. If no ring $F_2[x]/(x^n + 1)$ can be found such that $n = (m + 1)$, then find a ring $F_2[x]/(x^n + x^k + 1)$ such that $x^n + x^k + 1 = f_1(x) \times f_2(x)$ for some irreducible $f_2(x)$ of degree m. Note that n should be as small as possible.
3. Find a ring $F_2[x]/(x^n + x^{k_1} + x^{k_2} + 1)$ such that $x^n + x^{k_1} + x^{k_2} + 1 = f_1(x) \times f_2(x)$ for some irreducible $f_2(x)$ of degree m.
4. Perform multiplication of elements in a field, F_{2^m}, defined by a pentanomial. Note that, up to degree $m = 10,000$, there is either a trinomial or a pentanomial that is irreducible.
5. Choose one of the representations from Steps 2, 3, or 4 above for which the multiplier complexity is minimal.

We now consider the structure of irreducible polynomials that will help us find the rings mentioned above.

Definition. *A polynomial of the form* $f_2(x) = c_0 + c_1x + c_2x^2 + \cdots + c_mx^m$ *is written as* $(c_m, c_{m-1}, \cdots, c_2, c_1, c_0)$. *The coefficients* c_i *are in GF(2).*

Proposition 1. *To satisfy* $x^n + x^k + 1 = f_1(x) \times f_2(x)$, *where* $f_1(x) = x^2 + x + 1$, $f_2(x)$ *must be one of the following two forms:* $(1, 101, 101, \ldots, 101, 1, 101, 101, \ldots, 101, 1)$ *(call this type A) or* $(1, 101, 101, \ldots, 101, 10, 101, 101, \ldots, 101, 1)$ *(call this type B).*

Notes on Proposition 1. These forms have the pattern (1, 0 or more 101s, 1 or 10, 0 or more 101s, 1). If $f_2(x)$ is of the form (type A) (1, x repetitions of 101, 1, y repetitions of 101, 1), then it follows that $k = n - 3x - 2$ or $3y + 2$. If $f_2(x)$ is of the form (type B) (1, p repetitions of 101, 10, q repetitions of 101, 1), then it follows that $k = n - 3p - 4$ or $3q + 1$.

Proposition 2. *To satisfy* $x^n + x^{k_1} + x^{k_2} + 1 = f_1(x) \times f_2(x)$, *where* $f_1(x) = x + 1$, $f_2(x)$ *must be of the form* $(1, 1, 1, 1, \ldots 1, 0, 0, 0, \ldots 0, 1, 1, \ldots, 1)$.

Notes on Proposition 2. If $f_2(x)$ is of the form (p repetitions of 1, q repetitions of 0, r repetitions of 1), then it follows that $k_1 = n - p$ and $k_2 = r$.

The proof of the above propositions follows from the structure of the polynomials being considered. Note that it is much easier to find polynomials that satisfy Proposition 2 than those that satisfy Proposition 1. Tables 2 and 3 give some polynomials that satisfy the two observations above. These tables also contain the percentage increase in the AND and XOR gates for a multiplier implemented using each polynomial as compared to Mastrovito multipliers implemented using the original polynomial of degree m. In the tables, we have considered only those polynomials with $k = 1$, $k_1 = 2$, and $k_2 = 1$. For most of the entries in the tables, there does not exist a degree m all-1 polynomial.

We first consider the polynomial $x^n + x + 1$ and its factors. This polynomial is of special interest due to the fact that the time taken to multiply two polynomials modulo $x^n + x + 1$ is the same as the time taken to multiply two polynomials modulo $x^n + 1$ [16]. Table 2 gives the degrees of the irreducible factors of $x^n + x + 1$. We consider only those n which have an irreducible factor of degree m and there exists no irreducible trinomial of degree m. Therefore, multiplication of elements in F_{2^m} can be performed more efficiently in the ring $F_2[x]/(x^n + x + 1)$. Note that the table is not an exhaustive list of such polynomials. In the tables below, T_A is the delay of a 2-input AND gate and T_X is the delay of a 2-input XOR gate. The delays have been computed using Table 4.

Next, we consider the 4-term polynomial $x^n + x^2 + x + 1$ and its factors. This polynomial is the simplest 4-term polynomial for a given n. Table 3 gives the degrees of the irreducible factors of $x^n + x^2 + x + 1$. We are interested in finding an isomorphic copy of F_{2^m} in the ring $F_2[x]/(x^n + x^2 + x + 1)$ in order to reduce the multiplication complexity. This can be done if $x^n + x^2 + x + 1$ has an irreducible factor of degree m that is as close to n as possible. Table 3 gives degrees of factors of $x^n + x^2 + x + 1$, which has a factor of degree m and there exists no irreducible trinomial of degree m. Again, the times have been computed using Table 4.

4 Complexity

Table 4 gives the complexity of multiplying two field/ring elements when the field/ring is defined by a degree m binomial, trinomial, 4-term polynomial, and a pentanomial [16].

TABLE 2
Degrees of Irreducible Factors of $x^n + x + 1$

n	m	Percentage increase in AND gates	Percentage increase in XOR gates	Degrees of irreducible factors of x^n+x+1	Multiplier Time using x^n+x+1	Multiplier Time using degree m pentanomial
99	91	18.4	15.8	8,91	T_A+7T_X	T_A+13T_X
144	139	7.3	5.8	5,139	T_A+8T_X	T_A+14T_X
147	143	5.7	4.2	4,143	T_A+8T_X	T_A+14T_X
169	160	11.6	10.2	4,5,160	T_A+8T_X	T_A+14T_X
171	168	3.6	2.4	3,168	T_A+8T_X	T_A+14T_X
173	163	12.6	11.3	2,3,5,163	T_A+8T_X	T_A+14T_X
202	163	53.6	51.7	39,163	T_A+8T_X	T_A+14T_X
221	219	1.8	0.9	2,219	T_A+8T_X	T_A+14T_X
307	291	11.3	10.5	16,291	T_A+9T_X	T_A+15T_X
329	285	33.3	32.3	2,5,11,26,285	T_A+9T_X	T_A+15T_X
342	312	20.2	19.4	4,26,312	T_A+9T_X	T_A+15T_X
358	326	20.6	19.9	13,19,326	T_A+9T_X	T_A+15T_X
398	387	5.8	5.2	2,9,387	T_A+9T_X	T_A+15T_X
401	389	6.3	5.7	2,10,389	T_A+9T_X	T_A+15T_X
415	395	10.4	9.8	5,15,395	T_A+9T_X	T_A+15T_X
461	427	16.6	16.0	2,9,23,427	T_A+9T_X	T_A+15T_X
478	466	5.2	4.8	5,7,466	T_A+9T_X	T_A+15T_X
486	452	15.6	15.1	3,8,9,14,452	T_A+9T_X	T_A+15T_X
514	502	4.8	4.4	3,4,5,502	T_A+10T_X	T_A+16T_X

TABLE 3
Degrees of Irreducible Factors of $x^n + x^2 + x + 1$

n	m	Percentage increase in AND gates	Percentage increase in XOR gates	Degrees of irreducible factors of x^n+x^2+x+1	Multiplier Time using x^n+x^2+x+1	Multiplier Time using degree m pentanomial
86	85	2.4	1.2	1,85	T_A+11T_X	T_A+13T_X
90	78	33.1	31.3	1,11,78	T_A+11T_X	T_A+13T_X
98	91	16.0	14.7	1,6,91	T_A+11T_X	T_A+13T_X
104	70	120.7	116.8	1,5,28,70	T_A+11T_X	T_A+13T_X
112	88	62.0	59.8	1,4,19,88	T_A+11T_X	T_A+13T_X
114	99	32.6	31.1	1,14,99	T_A+11T_X	T_A+13T_X
124	120	6.8	5.9	1,3,120	T_A+11T_X	T_A+13T_X
164	152	16.4	15.6	1,11,152	T_A+12T_X	T_A+14T_X
180	176	4.6	4.0	1,3,176	T_A+12T_X	T_A+14T_X
186	164	28.6	27.8	1,3,7,11,164	T_A+12T_X	T_A+14T_X
366	365	0.5	0.3	1,365	T_A+13T_X	T_A+15T_X
422	421	0.5	0.2	1,421	T_A+13T_X	T_A+15T_X
468	454	6.3	6.0	1,13,454	T_A+13T_X	T_A+15T_X
556	520	14.3	14.1	1,35,520	T_A+14T_X	T_A+16T_X
602	595	2.4	2.2	1,6,595	T_A+14T_X	T_A+16T_X
676	672	1.2	1.0	1,3,672	T_A+14T_X	T_A+16T_X
712	704	2.3	2.1	1,2,4,704	T_A+14T_X	T_A+16T_X

In Table 4, T_A is the delay of a 2-input AND gate and T_X is the delay of a 2-input XOR gate. The second and third columns give the number of 2-input AND and 2-input XOR gates in an implementation of a multiplier and the fourth column gives the time required to perform a multiplication. Our technique performs multiplication in rings based on three and four term polynomials instead of a field based on a pentanomial. However, the rings defined by us are based on polynomials of higher degree than the pentanomial. This increases the gate count in the multiplier a little, but the speed of the multiplier increases. Wire length will also increase due to increase in the gate count, thereby increasing the delay by an additional amount. However, this is not a factor if the increase in the gate count is low.

TABLE 4
Complexity of Multipliers for Polynomials with Two, Three, Four, and Five Terms

	AND	XOR	Time
Binomial	m^2	$m(m-1)$	$T_A + \lceil \log_2 m \rceil T_X$
Trinomial $x^m + x^k + 1$	m^2	$m^2 - 1$	$T_A + (\lfloor (m-2)/(m-k) \rfloor + \lceil \log_2 m \rceil) T_X$
4-term polynomial	m^2	$(m+2)(m-1)$	$T_A + (4 + \lceil \log_2 m \rceil) T_X$
Pentanomial	m^2	$(m+3)(m-1)$	$T_A + (6 + \lceil \log_2 m \rceil) T_X$

For example, consider multiplication in the field $F_{2^{91}}$. From the first row of Table 2, we know that there exists a polynomial $x^{99} + x + 1$ which has two irreducible factors of degree 8 and 91. Therefore, from Wedderburn's Theorem, we know that there exists an isomorphic copy of the field $F_{2^{91}}$ in the ring $F_2[x]/(x^{99} + x + 1)$. Multiplication in the ring takes $T_A + 7T_X$ time units, whereas multiplication in the field $F_{2^{91}}$ defined by the pentanomial $x^{91} + x^8 + x^5 + x + 1$ takes $T_A + 13T_X$ time units. Thus, a savings of $7T_X$ time units is achieved using our approach. However, each ring element is 99 bits long and each field element is 91 bits long. Therefore, from Table 4, we can say that multiplication in the ring requires 9,801 2-input AND gates and 9,800 2-input XOR gates. Multiplication in the field requires 8,281 2-input AND gates and 8,460 2-input XOR gates. Thus, our technique requires extra hardware. Another advantage of our technique is that a systolic multiplier can be designed for multiplication modulo $x^n + x + 1$. This makes implementation in VLSI very easy.

5 Systolic Computation

In this section, we describe a systolic implementation of multipliers for elements in the ring defined by the polynomial $x^n + x + 1$. Systolic multipliers are modular and, hence, easy to implement in VLSI. Low complexity systolic multipliers were described in [15]. However, these multipliers were restricted to fields defined by the all-1 polynomial (AOP). AOPs of degree n can be found only if (n + 1) is a prime and 2 is a generator of the field GF(n + 1). When AOPs cannot be found, we can use Wedderburn's Theorem to define new rings defined by the polynomial $x^n + x + 1$ which contain the finite field of interest. A systolic implementation of multiplication in this new ring is presented next. We will show that the multiplier is similar to the multiplier of elements in a ring defined by the polynomial $x^n + 1$. We describe the Mastrovito multiplier [16] next and show how this can be used to design a systolic multiplier.

5.1 Mastrovito Multiplier for $x^n + x + 1$

Finite field/ring multiplication using a canonical basis is carried out by polynomial multiplication and modulo operation. Let $c(x)$ be the product of $a(x)$ and $b(x)$, where $a(x), b(x), c(x) \in F_2(x)/(x^n + x + 1)$. The polynomial multiplication $d(x) = a(x).b(x)$ can be performed as $d = A.b$, where $b = [b_0, b_1, \ldots, b_{n-1}]^T$ and $d = [d_0, d_1, \ldots, d_{n-1}]^T$ are the coefficient vectors of $b(x)$ and $d(x)$ and the $(2n-1) \times n$ matrix A is given by:

$$A = \begin{bmatrix} a_0 & 0 & \ldots & 0 & 0 \\ a_1 & a_0 & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n-2} & a_{n-3} & \ldots & a_0 & 0 \\ a_{n-1} & a_{n-2} & \ldots & a_1 & a_0 \\ -- & -- & -- & -- & -- \\ 0 & a_{n-1} & \ldots & a_2 & a_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \ldots & a_{n-1} & a_{n-2} \\ 0 & 0 & \ldots & 0 & a_{n-1} \end{bmatrix} = \begin{bmatrix} A_s \\ A_t \end{bmatrix}.$$

Note that $a = [a_0, a_1, \ldots, a_{n-1}]^T$ is the coefficient vector of $a(x)$ and the matrix A_s is $n \times n$ and A_t is $(n-1) \times n$. The modulo operation is performed next, as follows:

$$c(x) = d(x) \bmod (x^n + x + 1) = \sum_{k=0}^{2n-2} d_k x^k \bmod (x^n + x + 1)$$
$$= \sum_{k=0}^{n-1} d_k x^k + \sum_{k=n}^{2n-2} d_k x^k \bmod (x^n + x + 1).$$

Let $x^k \bmod (x^n + x + 1)$ be denoted as $u_j(x)$, where $j = k - n + 1$ and $n \le k \le 2n - 2$. Therefore, $u_j(x)$ is defined as follows:

$$u_j(x) = \begin{cases} x^n \bmod (x^n + x + 1); & j = 1 \\ u_{j-1}(x) \bmod (x^n + x + 1); & j = 2, 3, \ldots, n-1. \end{cases}$$

Therefore, $c(x)$ can be written as follows:

$$c(x) = \sum_{k=0}^{n-1} d_k x^k + \sum_{k=1}^{n-1} u_k(x).$$

This can be written in matrix form as follows:

$$c = [\,I_n \quad U\,] A.b.$$

In the above equation, I_n is the $n \times n$ identity matrix and U is an $n \times (n-1)$ matrix defined as $U = [u_1, u_2, \ldots, u_{n-1}]$, where $u_i = [u_{i,0}, u_{i,1}, \ldots, u_{i,n}]^T$ is the coefficient vector of $u_i(x), i = 1, 2, \ldots, (n-1)$. The $u_i(x)$ can be computed as follows:

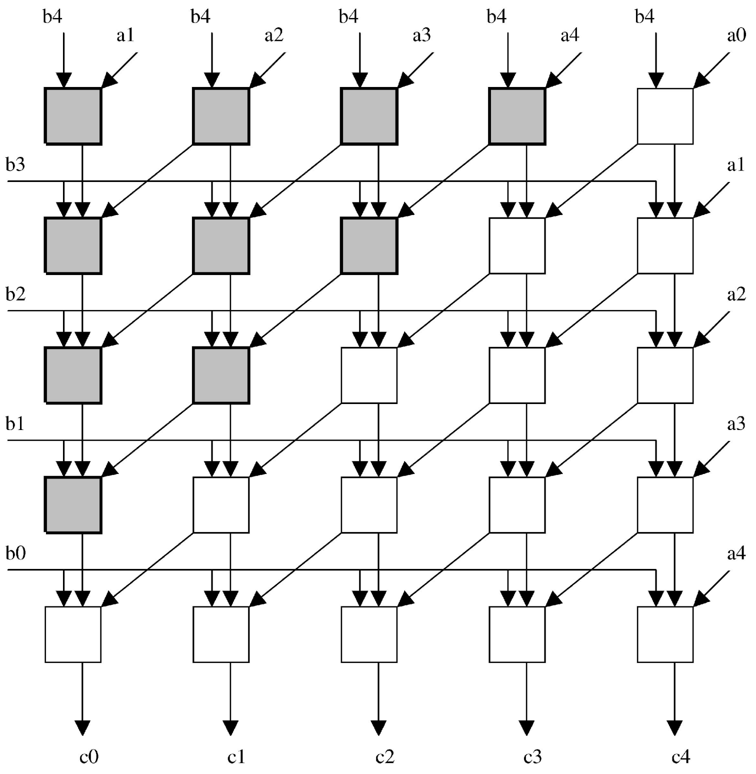


Fig. 1. Systolic array for computing $A_c.b$.

$$u_1(x) = x^n \bmod (x^n + x + 1) = (1 + x) = [1, 1, 0, \dots, 0]^T$$
$$u_2(x) = x(x + 1) = [0, 1, 1, 0, \dots, 0]^T$$
$$\vdots$$
$$u_i(x) = (x^{i-1} + x^i) = [0, \dots, 0, 1, 1, 0, \dots, 0]^T$$
$$\vdots$$
$$u_n(x) = (x^{n-1} + x^n) = [0, \dots, 0, 1, 1]^T.$$

Therefore, U can be written as,

$$U = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & 1 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 1 \\ 0 & 0 & \dots & 1 \end{bmatrix} = \begin{bmatrix} I_{n-1} \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ I_{n-1} \end{bmatrix}.$$

In the above expression, the size of the matrices

$$\begin{bmatrix} I_{n-1} \\ 0 \end{bmatrix}$$

and

$$\begin{bmatrix} 0 \\ I_{n-1} \end{bmatrix}$$

is $n \times (n-1)$.

The required product c can now be expressed as follows:

$$c = [\, I_n \quad U \,] \begin{bmatrix} A_s \\ A_t \end{bmatrix} .b = [A_s + UA_t].b$$
$$= A_s b + \begin{bmatrix} I_{n-1} \\ 0 \end{bmatrix} A_t b + \begin{bmatrix} 0 \\ I_{n-1} \end{bmatrix} A_t b = A_s.b + \begin{bmatrix} A_t b \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ A_t b \end{bmatrix}.$$

Note that the matrices A_s,

$$\begin{bmatrix} A_t \\ 0 \end{bmatrix}$$

and

$$\begin{bmatrix} 0 \\ A_t \end{bmatrix}$$

are of size $n \times n$.

5.2 Systolic Array

The product c can be written as

$$c = \left\{ A_s + \begin{bmatrix} A_t \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ A_t \end{bmatrix} \right\} .b.$$

We construct a systolic array first for computing

$$A_c b = \left\{ A_s + \begin{bmatrix} A_t \\ 0 \end{bmatrix} \right\} .b.$$

This systolic array can then be modified easily to compute

$$c = A_c b + \begin{bmatrix} 0 \\ A_t \end{bmatrix} .b.$$

Computing

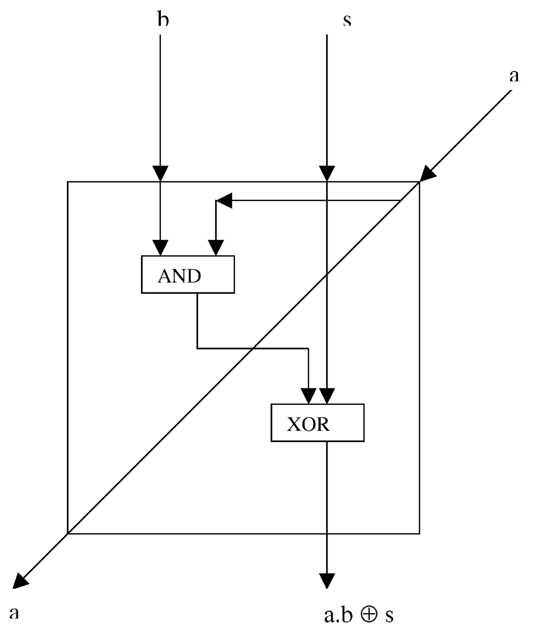


Fig. 2. A cell in the array of Fig. 1.

$$\begin{bmatrix} 0 \\ A_t \end{bmatrix} b$$

can be performed by reusing the values computed while performing the multiplication

$$\begin{bmatrix} A_t \\ 0 \end{bmatrix} .b.$$

Thus, the systolic array for computing c is simply a modification of the array for computing $A_c.b$. This is especially important because the array to compute $A_c.b$ is very simple. This is evident from the following expression for A_c:

$$A_c = \begin{bmatrix} a_0 & a_{n-1} & \cdots & a_1 \\ a_1 & a_0 & \cdots & a_2 \\ \vdots & \vdots & \vdots & \vdots \\ a_{n-1} & a_{n-2} & \cdots & a_0 \end{bmatrix}.$$

Note that every row of A_c is a circular shift of the row above it. Therefore, the rows of A_c consist of all circular shifts of vector a. Let us consider multiplication in the ring $R/(x^5 + x + 1)$. The systolic array for computing $A_c.b$ for this ring is shown in Fig. 1. Note that flip-flops are not shown in the figure. In Fig. 1, each cell consists of a 2-input AND gate and a 2-input XOR gate. The cells in the top row of Fig. 1 do not contain an XOR gate. The cells shaded in gray in Fig. 1 compute $A_t.b$. The output of these cells can be reused to compute c. An individual cell is shown in Fig. 2.

Fig. 3 shows the array to compute

$$c = A_c b + \begin{bmatrix} 0 \\ A_t \end{bmatrix} b.$$

The only difference between Fig. 3 and Fig. 1 is that (n-1) cells in Fig. 3 contain an extra 2-input XOR gate. These cells actually perform the addition of the term

$$\begin{bmatrix} 0 \\ A_t \end{bmatrix} b$$

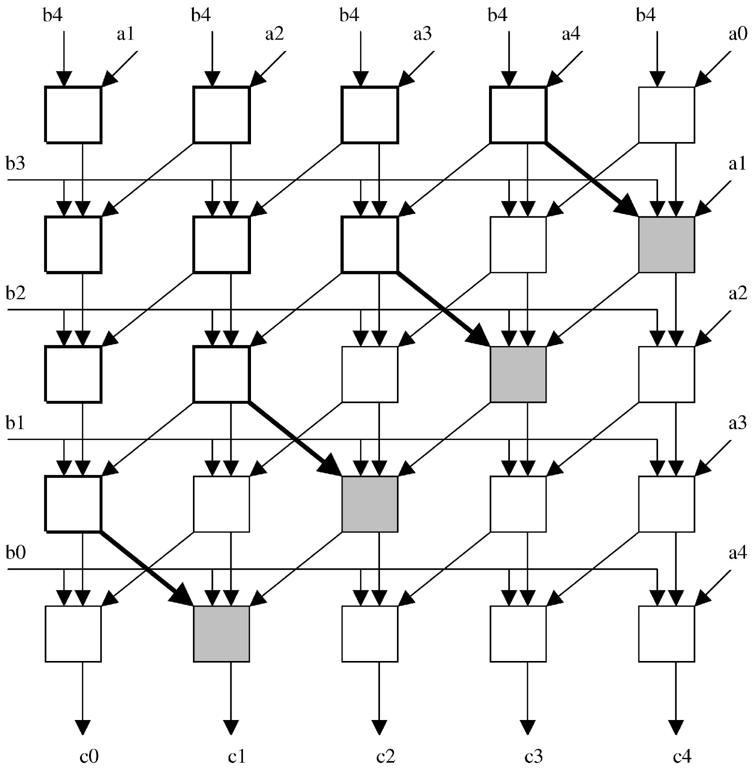


Fig. 3. Systolic array to compute the product c.

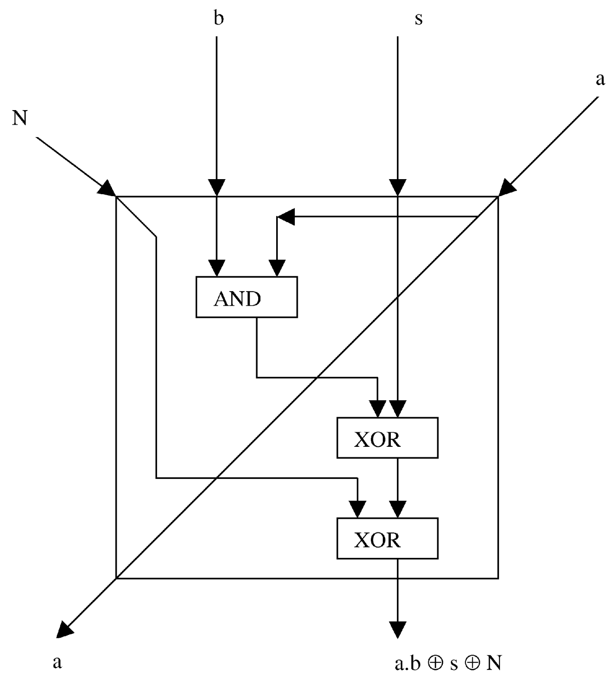


Fig. 4. The shaded cell of Fig. 3.

to $A_c.b$. These (n-1) cells are shaded gray in Fig. 3. Note that the inputs to the cells have been arranged in such a way that the upper left triangular portion computes $A_t.b$ (these cells have been shaded gray in Fig. 1). This is then reused to perform the addition

$$A_c b + \begin{bmatrix} 0 \\ A_t \end{bmatrix} b.$$

The resulting extra connections between cells are shown in Fig. 3 by bold lines. Fig. 4 shows the new cell of Fig. 3. Except for these (n-1) new cells, all the other cells are the same as in Fig. 2.

The systolic array consists of n^2, 2-input AND gates and $n^2 - 1$, 2-input XOR gates. Since the top row of cells requires no XOR gates, the array to compute $A_c.b$ contains $n^2 - n$ XOR gates and the extra (n-1) XOR gates are necessary for computing

$$A_c b + \begin{bmatrix} 0 \\ A_t \end{bmatrix} b.$$

6 Conclusion

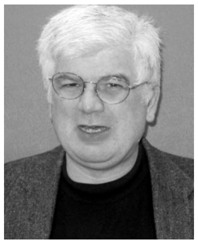
In this paper, we have presented a new representation of field elements using Wedderburn's Theorem. This enables us to find rings, which have an isomorphic copy of the field we are interested in. The decomposition of a ring into ideals isomorphic to extension fields is done by finding generators of these ideals. These generators are a set of orthogonal idempotents that can be found easily when dealing with fields of characteristic 2. The resulting multipliers are faster, but lead to a modest increase in hardware. In particular, we have considered rings of the form $F_2[x]/(x^n + x + 1)$ and $F_2[x]/(x^n + x^2 + x + 1)$. In both cases, we have provided examples for various values of n when multiplication is faster. It should be noted that this procedure can be used to find new rings that are based on polynomials other than the ones considered in this paper.

We have also shown that a multiplier for elements in $F_2[x]/(x^n + x + 1)$ can be easily implemented as a systolic array. This makes our procedure more attractive as it allows a simplified implementation in VLSI. A major advantage of the method proposed in this paper is that it can be used to improve encryption and decryption algorithms in elliptic curve cryptography.

References

[1] D.R. Hankerson, D.G. Hoffman, D.A. Leonard, C.C. Lindner, K.T. Phelps, C.A. Rodger, and J.R. Wall, *Coding Theory and Cryptography: The Essentials,* second ed. Marcel Dekker, 2000.
[2] E. Trichina, M. Bucci, D. De Seta, and R. Luzzi, "Supplemental Cryptographic Hardware for Smart Cards," *IEEE Micro,* pp. 26-35, Nov./Dec. 2001.
[3] J. Dhem and N. Feyt, "Hardware and Software Symbiosis Helps Smart Card Evolution," *IEEE Micro,* pp. 14-25, Nov./Dec. 2001.
[4] M. Smith, "Smart Cards: Integrating for Portable Complexity," *Computer,* pp. 110-115, Aug. 1998.
[5] P.W. Dowd and J.T. McHenry, "Network Security," *Computer,* pp. 24-28, Sept. 1998.
[6] B. Schneier, *Applied Cryptography.* John Wiley & Sons, 1996.
[7] N. Koblitz, *A Course in Number Theory and Cryptography.* Springer-Verlag, 1994.
[8] C.K. Koc and B. Sunar, "Low-Complexity Bit-Parallel Canonical and Normal Basis Multipliers for a Class of Finite Fields," *IEEE Trans. Computers,* vol. 47, no. 3, pp. 353-356, Mar. 1998.
[9] A. Halbutogullari and C.K. Koc, "Mastrovito Multiplier for General Irreducible Polynomials," *IEEE Trans. Computers,* vol. 49, no. 5, pp. 503-518, May 2000.
[10] M.A. Hasan and A.G. Wassal, "VLSI Algorithms, Architectures, and Implementation of a Versatile GF(2^m) Processor," *IEEE Trans. Computers,* vol. 49, no. 10, pp. 1064-1073, Oct. 2000.
[11] G. Drolet, "A New Representation of Elements of Finite Fields GF(2^m) Yielding Small Complexity Arithmetic Circuits," *IEEE Trans. Computers,* vol. 47, no. 9, pp. 938-946, Sept. 1998.
[12] C. Paar, P. Fleischmann, and P. Soria-Rodriguez, "Fast Arithmetic for Public-Key Algorithms in Galois Fields with Composite Exponents," *IEEE Trans. Computers,* vol. 48, no. 10, pp. 1025-1034, Oct. 1999.
[13] T.W. Hungerford, *Algebra,* eighth ed. Springer-Verlag, 1997.
[14] W.V. Vasconcelos, *Computational Methods in Commutative Algebra and Algebraic Geometry.* Springer-Verlag, 1997.
[15] C. Lee, E. Lu, and J. Lee, "Bit-Parallel Systolic Multipliers for $GF(2^m)$ Fields Defined by All-One and Equally Spaced Polynomials," *IEEE Trans. Computers,* vol. 50, no. 5, pp. 385-393, May 2001.
[16] T. Zhong and K.K. Parhi, "Systematic Design of Original and Modified Mastrovito Multipliers for General Irreducible Polynomials," *IEEE Trans. Computers,* vol. 50, no. 7, pp. 734-749, July 2001.

Rajendra S. Katti received the BTech degree from the Indian Institute of Technology (Bombay), India, in 1983. He received the MS degree in mechanical engineering from the University of Idaho in 1985 and the MS degree in electrical engineering from Washington State University in 1987, where he also earned the PhD degree in electrical engineering in 1991. Dr. Katti teaches courses related to digital systems and computer architecture. He has received funding from the US National Science Foundation in the area of performance modeling of computer architectures. His interests are in smart sensors, cryptographic hardware, finite field arithmetic, fault-tolerant computing, and computer architecture. He has published more than 20 journal and conference papers on these topics. He was a senior design engineer at Intel Corporation in 2000 and 2001, where he worked in the Design for Testability Group. He has also taught at Wichita State University in Kansas. He has collaborated with the IBM Almaden Research Center for the development of unidirectional error correcting codes. He is currently an associate professor in the Department of Electrical and Computer Engineering at North Dakota State University. He is a member of the IEEE.



Joseph P. Brennan received the BA degree in mathematics from the University of Chicago in 1977 and received the PhD degree in mathematics from the University of Illinois at Urbana-Champaign in 1984. He has held positions at Michigan State University, Seton Hall University, the University of Mississippi, and Rutgers University. In 1999-2001, he was a program director for the Algebra, Number Theory, and Combinatorics Program of the Division of Mathematical Sciences of the US National Science Foundation. Since 1989, he has been at North Dakota State University, where is is currently an associate professor of mathematics. His research has involved questions related to the homological theory of modules and the study of invariants.