Low-Power Design for a Digit-Serial Polynomial Basis Finite Field Multiplier Using Factoring Technique

Abstract:

In CMOS-based application-specific integrated circuit (ASIC) designs, total power consumption is dominated by dynamic power, where dynamic power consists of two major components, namely, switching power and internal power. In this paper, we present a low-power design for a digit-serial finite field multiplier in $GF(2^m)$. In the proposed design, a factoring technique is used to minimize switching power. To the best of our knowledge, factoring method has not been reported in the literature being used in the design of a finite field multiplier at an architectural level. Logic gate substitution is also utilized to reduce internal power. Our proposed design along with several existing similar works have been realized for $GF(2^{233})$ on ASIC platform, and a comparison is made between them. The synthesis results show that the proposed multiplier design consumes at least 27.8% lower total power than any previous work in comparison. The proposed architecture of this paper is analysis the logic size, area and power consumption using tanner tool.

Existing System:

Binary extension field, denoted by $GF(2^m)$, is very attractive for hardware implementation, because it offers carry-free arithmetic. Multiplication operation has been paid most attention by researchers, because addition is simply bitwise XOR operation between two field elements, and the more complex operations, inversion, can be carried out with a few multiplications. In $GF(2^m)$, there are various methods to represent field elements, such as polynomial basis (PB), normal basis, and dual basis. PB is probably the most popularly used basis, because it is adopted as one of the basis choices by organizations that set standards for cryptography applications. Thus, a large number of architectures for efficient implementation of PB finite field multipliers have been proposed. In addition, new representations based on PB called shifted PB (SPB) and generalized PB have been proposed for efficient implementation of multipliers over $GF(2^m)$.

The choice of the irreducible polynomial $p(x)$ affects the complexity of a finite field multiplier. Various types of irreducible polynomials include trinomials, pentanomials, allone polynomials, and equally spaced polynomials. Standard organizations recommend irreducible polynomials
with less number of nonzero terms (irreducible trinomials and pentanomials) for practical use as these types of irreducible polynomials can provide multipliers with lower complexity. PB finite field multiplier architectures can be categorized into bit-serial, bit-parallel, and digit-serial architectures. Bit-parallel structure is fast, but it is expensive in terms of area. In EC cryptography, the binary extension field size, m, is required to be on the order of $10^2$, and thus a bit-parallel structure requires a very high I/O bandwidth, which is usually not available in the small portable and wireless devices. Bit-serial architecture is area efficient, but it is too slow for many applications. Power optimization were also considered in some of these works.

**Disadvantages:**

- High power consumption

**Proposed System:**

A factoring technique is adopted in design of a digit-serial PB multiplier in GF(2m). To the best of our knowledge, a factoring method has not been reported in the literature being used in the design of a finite field multiplier at an architectural level. A logic gate substitution technique is also used in our design to reduce the internal power consumption of the proposed digit-serial multiplier. The synthesis results show that our new design has both the lowest dynamic power consumption and the lowest total power consumption among several similar existing works.

**Binary Extension Field GF(2^m):**

A finite field is defined as a set of finite many elements, where addition and multiplication are the operations defined on the set. A binary extension field, GF(2m), is generated by a degree monic irreducible polynomial, $p(x) = x^m + p_{m-1}x^{m-1} + \cdots + p_2x^2 + p_1x + 1$, where $p_i$ is either 0 or 1. $p(x)$ also specifies a PB $\{1, x, x^2, \ldots, x^{m-1}\}$. Each element of GF(2m) can be represented as a polynomial of degree at most $m-1$ over GF(2m) with respect to the PB. For instance, an element $A \in GF(2m)$ can be expressed as

$$A(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \cdots + a_2x^2 + a_1x + a_0(1)$$

With $ai \in GF(2), 0 \leq i \leq m-1$. 
Multiplication of two field elements $A(x)$ and $B(x)$ of the binary extension field can be given by

$$C(x) = A(x)B(x) \mod p(x). \quad (2)$$

Digit-Serial PB Multiplication:

In digit-serial multiplication, the bits of one operand are divided into digits of size $k$ while the bits of the other input operand are processed in parallel. Only one digit of the first operand is accessible in each clock cycle.

Power Dissipation for CMOS-Based Circuits:

Power consumption in a CMOS-based design contains two major components: static power and dynamic power. For a CMOS-based design, dynamic power plays a dominant role in the total power consumption.

Proposed low-power design of a digit-serial multiplier in $\text{GF}(2^m)$:

Multiplier Architecture:

An architecture diagram for the proposed digit-serial PB multiplier in $\text{GF}(2^m)$ is shown in Fig. 2. There are three modules, as shown in Fig. 2, namely, $k \times m$ multiplier, constant multiplier, and field adder.

1. $k \times m$ multiplier takes one operand $B$ of $m$-bit and the other operand $A_j$ of $k$-bit. Note that $A_j$ changes for different clock cycles $j$. Thus, it has higher switching activity compared with operand $B$. A straightforward realization of this module was used. For the comparison purpose, it is given in Algorithm 1. Note that a modification to this algorithm using a factoring method is proposed in Section III-B. The three steps in Algorithm 1 are, respectively, realized with the circuit blocks from left to right, as shown in Fig. 3(a).

2. Constant multiplier module realizes multiplication between a field element and the constant $x^k$.

3. Field adder module implements finite field addition using $m$ two-input XOR gates formed as a one-layer network.

Advantages:
Do Your Projects With Technology Experts…

- Low power consumption

Software implementation:

- Tanner EDA