INTEGRATION, the VLSI journal 46 (2013) 323–332

Contents lists available at SciVerse ScienceDirect

INTEGRATION, the VLSI journal

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

Fast, compact and symmetric modular exponentiation architecture by common-multiplicand Montgomery modular multiplications

Tao Wu [a,*], Shuguo Li [b,**], Litian Liu [b]

[a] Department of Microelectronics and Nanoelectronics, Tsinghua University, Beijing 100084, PR China
[b] Institute of Microelectronics, Tsinghua University, Beijing 100084, PR China

ARTICLE INFO

Article history:
Received 1 April 2012
Received in revised form
23 September 2012
Accepted 24 September 2012
Available online 5 October 2012

Keywords:
Modular exponentiation architecture
Common-multiplicand Montgomery modular multiplication
Feedforwarding mechanism
Fault attack resistance
Simple power attack resistance

ABSTRACT

In this paper, the primitive common-multiplicand Montgomery modular multiplication is developed for modular exponentiation. Together with Montgomery powering ladder, a fast, compact and symmetric modular exponentiation architecture is proposed for hardware implementation. The architecture consists of one group of processing elements along the central line and two symmetric groups of accumulation units on two sides. The central elements perform modular reductions, while the symmetric units on both sides accumulate the modular multiplication results. A feedforwarding architecture is employed to decrease the latency between processing elements, in parallel with the word-based accumulation units, which are also pipelined. Meanwhile, due to the symmetric architecture and Montgomery powering ladder, the modular exponentiation is immune from fault and simple power attacks. Implemented in FPGA platform, the performance of our proposed design outperforms most results so far in the literature.

1. Introduction

Public-key cryptography leads to finite field arithmetic over long integers, such as RSA cryptography [1], ElGamal cryptography [2], and Diffie-Hellman key exchange protocol [3]. These cryptographic applications are interfered with modular exponentiations.

Efficient hardware architectures have been described for modular exponentiations in the literature, which are based on systolic arrays [4], residue number systems [5], high-radix scalable architectures [6], carry-save additions [7]. A review of general hardware architectures for modular exponentiation can be found in [8]. In fact, fast modular exponentiations can be implemented by two ways: (1) fast algorithms or architectures for modular multiplications; (2) efficient algorithms or architectures for modular exponentiation itself. Besides the performance with time and area of modular exponentiations, the resistance from intended attacks should also be considered since modular exponentiations themselves are used as the mathematical barrier for security. In this paper, we expect to improve modular exponentiations by merging the above requirements.

In this work, a fast, compact and symmetric modular exponentiation architecture is proposed, which outperforms most results so far in the literature both in speed and area overhead. There are several characteristics with this architecture:

(1) Implement a similar feedforwarding mechanism to a low-latency scalable Montgomery modular multiplier [9,10] in processing elements.
(2) Apply common-multiplicand Montgomery modular multiplication algorithm [11,12] for modular exponentiations.
(3) Employ Montgomery ladder in left-to-right binary method, and the resulted symmetry will keep the modular exponentiation from fault and simple power attacks [13].

Especially, although there are no two Montgomery modular multipliers in the architecture, the modular exponentiation architecture can be divided into three parts: two symmetric groups of accumulation units and one group of central processing elements. Together with the input pattern by Montgomery powering ladder, the centrosymmetry helps the architecture resistant from fault and simple power attacks. Nevertheless, the proposed architecture only deals with operands of fixed binary lengths, and therefore is not scalable [14].

The rest of this paper is organized as follows: Section 2 introduces the original common-multiplicand Montgomery modular multiplication; in Section 3, we develop a common-multiplicand algorithm for hardware implementation of modular exponentiations; then in

* Principal corresponding author.
** Corresponding author.
E-mail addresses: t-wu03@whu.edu.cn, twu03ster@gmail.com (T. Wu), lisg@tsinghua.edu.cn (S. Li), liulitian@tsinghua.edu.cn (L. Liu).

0167-9260/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.vlsi.2012.09.002

Section 4 the architecture to implement the common-multiplicand Montgomery modular exponentiation is proposed, where both radix-2 and radix-4 modular exponentiation architectures are presented; in Section 5 some experiments are given for comparison with the results in the literature; and Section 6 concludes this paper.

2. Common-multiplicand Montgomery modular multiplication

Common-multiplicand Montgomery modular multiplication is proposed for modular exponentiation in [11,12] and has been developed in [12,15,16]. In [12], together with signed-digit recoding technique about 66.7% computational cost is saved. However, on the one hand, this algorithm is aiming at software implementations and has only been applied in hardware architectures for modular multiplication/exponentiation in $GF(2^m)$ [17]. On the other hand, the more advanced common-multiplicand Montgomery modular multiplications [12,15,16] are really uneasy for hardware implementations due to complex operations. Therefore, in this paper we will focus on the primitive algorithm proposed in [11], from which a fast, compact and symmetric modular exponentiation architecture can be devised. In this section, we will firstly give an overview of the common-multiplicand Montgomery modular multiplication algorithm [11].

Suppose F, S and M are all integers with n binary bits, and there are two Montgomery modular multiplications: $\mathrm{Mtul}(S,F)=S\cdot F\cdot 2^{-n} \bmod M$, and $\mathrm{Mtul}(S,S)=S\cdot S\cdot 2^{-n} \bmod M$. The common-multiplicand Montgomery modular multiplication takes the privilege of one common multiplicand S between two Montgomery modular multiplications and divides the two modular multiplications into two parallel processes:

- common modular reductions. $T_i \coloneqq S\cdot 2^{-i} \bmod M$, for $i=1,2,\ldots,n$;
- two separate accumulations. Suppose s_i and f_i are, respectively, the i-th digit of S and F, then there are

$$X=\sum_{i=1}^{n} f_i S\cdot 2^{-i} (\bmod M), \tag{1}$$

$$Y=\sum_{i=1}^{n} s_i S\cdot 2^{-i} (\bmod M). \tag{2}$$

In the above equations, Montgomery modular multiplication of $\mathrm{Mtul}(S,F)$ modulo M has been divided into a sum of partial products in the following [11]:

$$\begin{aligned}\mathrm{Mtul}(S,F) &= S\cdot F\cdot 2^{-n} (\bmod M)\\ &= S\cdot\left(\sum_{i=0}^{n-1} f_i\cdot 2^i\right)\cdot 2^{-n} (\bmod M)\\ &= f_0\cdot S\cdot 2^{-n}+f_1\cdot S\cdot 2^{-(n-1)}\\ &\quad +\cdots+f_{n-1}\cdot S\cdot 2^{-1} (\bmod M).\end{aligned} \tag{3}$$

A similar expression exists for $\mathrm{Mtul}(S,S)$.

The common multiplicand S within Eqs. (1) and (2) then enables parallel computation of two Montgomery modular multiplications. An algorithm that utilizing this parallelism is firstly proposed in [11] and is shown below:

Algorithm 1. Original common-multiplicand Montgomery modular multiplication.

Input: S and F are both n-bit numbers, with $S=\sum_{i=0}^{n-1} s_i\cdot 2^i$, $F=\sum_{i=0}^{n-1} f_i\cdot 2^i$. M is the modulus, with $2^{n-1}<M<2^n$, $\mathrm{GCD}(M,2)=1$. In addition, $\rho=-M_0^{-1} \bmod 2$.
Output: $X=\mathrm{Mtul}(S,S)$, $Y=\mathrm{Mtul}(S,F)$.
1: $X \coloneqq 0$, $Y \coloneqq 0$;
2: $T \coloneqq S$;
3: **for** $i=n-3$ downto 0 **do**
4: $\quad \mu \coloneqq T_0\cdot\rho \bmod 2$;
5: $\quad T \coloneqq (T+\mu M)/2$;
6: $\quad X \coloneqq X+f_i T$, $Y \coloneqq Y+s_i T$;
7: **end for**
8: **for** $i=0$ to 1 **do**
9: $\quad X \coloneqq X+f_{n-2-i}S$, $Y \coloneqq Y+s_{n-2-i}S$;
10: $\mu_X \coloneqq X_0\rho \bmod 2$, $\mu_Y \coloneqq Y_0\rho \bmod 2$;
11: $X \coloneqq (X+\mu_X M)/2$, $Y \coloneqq (Y+\mu_Y M)/2$;
12: **end for**
13: **while** $X\geq M$ **do**
14: $\quad X \coloneqq X-M$;
15: **end while**
16: **while** $Y\geq M$ **do**
17: $\quad Y \coloneqq Y-M$;
18: **end while**
19: **return** X, Y.

Algorithm 1 effectively cuts down the computational efforts with Montgomery modular multiplication owing to common computation. While it is suitable for software implementation, there is no report about hardware implementations in prime fields so far.

We will present an exponentiation architecture based on such common-multiplicand Montgomery modular multiplications in the following sections. The advanced common-multiplicand algorithms in [15,16] just find out more common computation in modular exponentiations, which requires complex operations and is inefficient for hardware implementation.

3. Montgomery modular exponentiation based on revised common-multiplicand Montgomery modular multiplications

Montgomery modular exponentiation is named after modular exponentiation by Montgomery modular multiplications, which is shown in Algorithm 2.

Algorithm 2. Montgomery modular multiplication.

Input: $M<R=2^n$, $\mathrm{GCD}(M,2)=1$. $0\leq A\leq 2^n$, $0\leq B\leq 2^n$, and $0\leq A\cdot B<M\cdot R$. Precompute $\mu=-M^{-1} \bmod R$.
Output: $S=A\cdot B\cdot R^{-1} \bmod M$, $0\leq S<M$.
1: $T=A\cdot B$;
2: $Q=T\cdot\mu \bmod R$;
3: $S=(T+Q\cdot M)/R$;
4: if $S\geq M$ **then**
5: $\quad S \coloneqq S-M$;
6: **end if**
7: **return** S.

Montgomery modular multiplication has many variant forms [18], and is efficient for hardware implementation. Combined with the binary method, it can be used to perform modular exponentiation.

3.1. Common-multiplicand Montgomery modular multiplication for hardware implementation

For the sake of hardware implementation, the original common-multiplicand Montgomery modular multiplication is rewritten in