蒙哥马利模乘算法改进及硬件实现

任仕伟，王华阳，郝越，薛丞博
（北京理工大学 集成电路与电子学院，北京　100081）

摘　要：在嵌入式和物联网等领域的加密应用场景中，需要在加密实现的性能和资源消耗之间找到综合效率最佳的平衡点. 模乘法器是 Rivest-Shamir-Adleman 算法（RSA）和椭圆曲线密码（ECC）等公钥密码算法的核心运算模块，其资源占用和运算速度直接影响上层密码算法的整体性能. 本文提出高效低延迟的蒙哥马利模乘算法可以有效降低运算量，减少硬件设计的复杂度，结合使用提出的 5-2 低延迟加法器进一步降低模乘法器的关键路径长度，从而提高算法的运行效率. 在 Xilinx-K7 系列平台上实现的 1 024 位模乘运算模块系统主频可达 278 MHz，同时面积时间积（ATP）比已有同类算法提高了 15% 以上，综合效率表现最优. 结果表明，改进后的蒙哥马利模乘算法硬件资源消耗低，适用于物联网等轻量级密码系统.

关键词：加密算法；模乘；蒙哥马利；保留进位加法器
中图分类号：TN402　　**文献标志码：**A　　**文章编号：**1001-0645(2024)03-0306-06
DOI: 10.15918/j.tbit1001-0645.2023.082

An Improved Montgomery Modular Multiplication Algorithm and Its Hardware Implementation

REN Shiwei,　WANG Huayang,　HAO Yue,　XUE Chengbo
（School of Integrated Circuits and Electronics, Beijing Institute of Technology, Beijing 100081, China）

Abstract: In cryptographic application scenarios such as embedded and IoT, it is necessary to balance the performance and resource consumption of cryptographic implementation to find the best balance of comprehensive efficiency. As the core computing module of public key cryptographic algorithms such as Rivest-Shamir-Adleman algorithm (RSA) and elliptic curve cryptography (ECC), the resource consumption and computing speed of the modulo multiplier directly determine the overall performance of the upper layer cryptographic algorithms. The proposed efficient low-latency Montgomery modulo multiplication was designed to effectively reduce the amount of operations and the complexity of hardware design. On this basis, the length of the critical path in the modulo multiplier was arranged to be further reduced by using the proposed 5-2 low-latency adder in combination to improve the algorithm operation efficiency. The system main frequency of the 1024-bit modulo module implemented on the Xilinx-K7 series platform can reach 278 MHz, while the area-time-product（ATP）is improved by more than 15% compared with the existing similar algorithms, and the overall efficiency is optimal. The results show that the improved Montgomery modulo multiplication algorithm can give a low hardware resource consumption, being suitable for lightweight cryptosystems such as IoT.

Key words: encryption algorithm; modulo multiplier; Montgomery; carry save adder

数字通信技术的发展在提高信息传输的便利性和效率的同时，也加剧了信息泄漏和窃听的风险. 为了加强信息安全，许多加密算法被提出[1]. 目前，作为现代公钥密码学两个主要标准的 Rivest-Shamir-Adleman 算法（RSA）和椭圆曲线密码（elliptic curve cryptography, ECC）是密码系统研究的焦点之一[2－4].

收稿日期：2023－04－06
基金项目：国家自然科学基金资助项目（62201039）；重庆自然科学基金资助项目（cstc2021jcyj-msxmX1096）
作者简介：任仕伟（1985—），女，博士，副教授，E-mail: renshiwei@bit.edu.cn.
通信作者：薛丞博（1988—），男，博士，实验师，E-mail: silenuszhi@bit.edu.cn.

以 RSA 为例, 该算法使用一个模数(modulus) M 和两个密钥 d 和 e, 其中至少有一个密钥是私钥. 要传递的信息 A 通过模幂运算被加密为 $C=A^e \bmod M$, 解密阶段同样通过模幂运算 $A=C^d \bmod M$ 解密. $A^e \bmod M$ 的计算过程主要由模乘和幂运算组成.

从前述可以看出, RSA 算法在加解密的过程中都会进行重复的模乘运算. 类似地, ECC 算法中的点乘操作基于模乘操作实现[5]. 对于模乘运算, 以 RSA 常用的 1 024 位大数为例, 设有乘数 A, 被乘数 B, 模数 N, 需要先计算两个 1 024 位大数的积 $S=A\times B$, 再将结果积对模数 N 取余数, 在此计算过程中将会产生 2 048 位的中间结果. 对中间结果 S 的模约减步骤通常需要将 S 与模 N 进行比较, 在最坏的情况下, 意味着需要依次检查两个长数的每一位. 因此模乘运算是 RSA 和 ECC 算法中主要的耗时操作, 提高模乘操作的效率对提高公钥密码系统的性能起着关键作用.

在现有模乘算法中, 蒙哥马利算法[6]是最有效的方案. 该算法提出基于加法及除以 $2^n(n\in Z)$ 的替代方法, 以避免除法操作[7], 而除以 $2^n(n\in Z)$ 可通过简单的移位操作在硬件上实现, 从而达到节省硬件资源的效果. 该算法在很多密码学应用中都得到了广泛的应用, 如用于抗侧信道攻击、密钥管理和其他依赖安全性的应用[8−10], 其高效性和安全性得到了广泛认可, 成为现代密码学领域中不可或缺的重要工具. 然而在物联网等轻量级的应用场景下, 由于要兼顾设备资源敏感、低功耗和速度, 该算法的性能表现较为有限[11−12].

为解决前述问题, 本文提出使用高效低延迟蒙哥马利模乘及 5-2 低延迟加法器相结合的方法优化基-2 的蒙哥马利模乘的结构, 以减少资源消耗并提高计算速度. 在 Xilinx-K7 平台上的实现结果表明, 与已有同类算法相比, 所提出的方法可降低 15% 以上的面积时间积(area-time-product, ATP), 在资源和速度上达到了很好的平衡, 更加适用于物联网安全等轻量型应用场景.

1　算法理论基础

1.1　蒙哥马利模乘算法

给定模数 N、两个整数 A 和 B 以及大整数 R, 满足 $2^k-1\leqslant A,B\leqslant 2^k$, $R=2^k$, 且 $gcd(N,R)=1$[13]. 此时存在自然数 R^{-1} 满足:

$$R\times R^{-1} \bmod N=1 \tag{1}$$

蒙哥马利算法首先将操作数转换为蒙哥马利表示形式, 然后进行蒙哥马利乘运算. 对于 $A<N$, 其蒙哥马利表示定义为

$$A':=A\times R \bmod N \tag{2}$$

A' 和 B' 的蒙哥马利乘积定义为

$$C':=A'\times B'\times R^{-1}(\bmod N) \tag{3}$$

故而 C 可以通过蒙哥马利约减算法计算得到

$$C:=\frac{[C'+(C'\times N' \bmod R)\times N]}{R} \tag{4}$$

其中 R^{-1} 和 N' 满足

$$R\times R^{-1}-N\times N'=1(\bmod N) \tag{5}$$

R^{-1} 和 N' 均通过扩展的欧几里得算法预先计算[6]. 综合上述可知, 蒙哥马利模乘算法(Monpro(A,B))主要有以下 3 个步骤:

(1) 求出大数 A 和 B 的蒙哥马利乘积为

$$T:=A'\times B'\times R^{-1}(\bmod N),$$

(2) 将结果进行蒙哥马利约减

$$U:=(T+(T\times N' \bmod R)\times N)/R,$$

(3) 判断 $U\geqslant N$ 是否成立, 若判断结果为成立, 则返回 $U-N$, 否则返回 N.

1.2　基-2 的蒙哥马利算法

基-2 的蒙哥马利模乘算法按位进行交错循环计算, 算法伪代码如算法 1 所示.

算法 1: 基-2 的蒙哥马利算法

输入: 操作数 A, B; 模数 N

输出: $A\times B \bmod N$ 结果的蒙哥马利表示 S

1 function Radix-2's *Monpro*(A,B)

2 $S[0]=0$;

3 for $i=0$ upto $k-1$ do

4 $q_i=(S[i]_0+A_i\times B_0) \bmod 2$;

5 $S[i+1]=(S[i]+A_i\times B+q_i\times N)/2$;

6 end for

7 return $S[k]$

8 end function

基-2 的蒙哥马利模乘算法具备实现简单、资源消耗少、系统主频高的特点, 更适用于硬件实现[14−15]. 观察算法 1 可知, 基-2 的蒙哥马利算法的延迟主要来自于循环迭代的大数加和的运算. 本文致力于对该部分运算的优化从而实现对基-2 的蒙哥马利算法的改进.

1.3　保留进位加法器

传统的进位传播加法器在进行加法计算时, 需要等待每一位的进位信号从低位到高位逐个传递,

最终才能得到整个加法结果. 这种逐位传递的方式, 虽然可以确保计算的正确性, 但在计算速度上存在瓶颈. 相比之下, 保留进位加法器(carry save adder, CSA)会同时计算出所有位的结果, 并将进位信号暂时保留, 以备后续的加法计算使用. 待所有位计算完成后, 保留的进位信号会一次性地进行处理, 最终得到整个加法结果. 这种方式可以大大提升加法器的计算速度, 特别是在对长数字进行计算时, 其优势更为显著. 所以, 相对于传统的进位传播加法器, 保留进位加法器在计算速度和效率上具有明显的优势.

保留进位加法器将操作数表示为保留进位格式(carry save representation, CSR)进行计算, 在加法器链中, 每个操作数的和(SUM)与进位(CARRY)被计算并存储在两个独立的链路中, 即: SUM = X_1 xor X_2 xor X_3, CARRY = (X_1 and X_2) or (X_1 and X_3) or (X_2 and X_3), 在加法器链路末端, 和与进位相加得出最终结果.

2 算法改进

2.1 高效低延迟蒙哥马利模乘

如前所述, 在基-2 的蒙哥马利模乘法器中, 计算求和的过程是其中关键的一环. 该过程需要计算q_i值并将其乘以N, 然后将结果与另外两个参数的求和结果相加. 然而, 在计算q_i的过程中, 需要将B的最低位(B_0)和A进行逻辑与运算, 然后将其与上一轮次结果的最低位($S[i]_0$)相加. 为了避免这种情况的发生, 本文提出了一种改进方案, 即将 B 乘以 2(左移 1 位)后作为算法的输入, 以使得B_0=0. 通过这种方式, 可以省略掉$A_i \times B_0$的计算, 从而减小计算的复杂度. 使用改进后的蒙哥马利模乘无需额外计算q_i, 即

$$q_i = (S[i]_0 + A_i \times B_0) \bmod 2 = (S[i]_0 + A_i \times 0) \bmod 2 = S[i]_0 \bmod 2 = S[i]_0 \quad (6)$$

由于将 B 左移后, 新的操作数比原操作数多了一个二进制位, 算法的计算时间也相应地增加了一个时钟周期.

硬件实现的计算速度取决于计算所需周期数和系统主频. 公钥密码系统中的模幂运算实际上是模乘算法进行一定次数的迭代, 所以单次迭代延时直接影响了算法整体延时. 改进后的蒙哥马利模乘相比原算法减少一个乘加计算, 并且由于q_i与$S[i]_0$位数均为 1, 所以改进后的算法并不增加电路面积. 且由于原算法中q_i与$S[i+1]$的计算存在固定的前后串行关系, 因此改进后的算法单次迭代用时减半, 从而在减小计算复杂度的情况下直接缩减了整体计算用时.

为了验证改进算法的效率提升, 将原算法与高效低延迟蒙哥马利模乘分别使用 MATLAB 实现并计时进行对比. 实验采用 64 位无符号整数操作数循环计算 16 次来模拟 1 024 位大数的计算, 并在此基础上计算 100 次以得到足够的样本并减小误差, 结果表明, 本算法的计算时间为 39.575 ms, 而原算法的计算时间为 80.487 ms, 本文提出的改进算法具有更高的效率和更低的时间延迟, 同时由前述分析可知, 改进后的算法硬件资源占用方面也具有优势.

2.2 5-2 低延迟加法器

由 2.1 节可知, 使用了高效低延迟蒙哥马利模乘后, 基-2 的模乘在单次迭代中仅含一次乘加运算, 即$S[i+1] = (S[i] + A_i \times B + S[i]_0 \times N)/2$. 在实际应用中, 公钥密码方法通常选取位宽较大的操作数以满足安全性需求, 而在大位宽数的加法中, 常规的进位传播加法器会引起长进位链问题, 导致整体电路时序紧张、系统主频降低, 从而影响模乘运算的吞吐率, 降低电路的性能与稳定性, 保留进位加法器是在减少进位传播延迟的同时添加多个操作数的最常用方法之一, 可用于解决这个问题, 本节以此为出发点对该计算进行改进.

基于上述思想, 将式$S[i+1] = S[i] + A_i \times B + S[i]_0$转换为保留进位格式可得式$S_1[i+1], S_2[i+1] = CSR[S_1[i] + S_2[i] + A_i \times (B_1 + B_2) + S[i]_0 \times N]/2$. 观察可知, 新的算式操作数增加为 5 个, 即S_1、S_2、$A_i \times B_1$、$A_i \times B_2$和$S[i]_0 \times N$. 针对此情况, 本文提出了将 3 级 CSA 级联构建为 5-2 低延迟加法器的结构, 将 5 个操作数加和为 2 个输出, 最大限度地减小关键延迟, 更好地平衡资源占用与运算速度.

5-2 低延迟加法器的结构框图如图 1 所示.

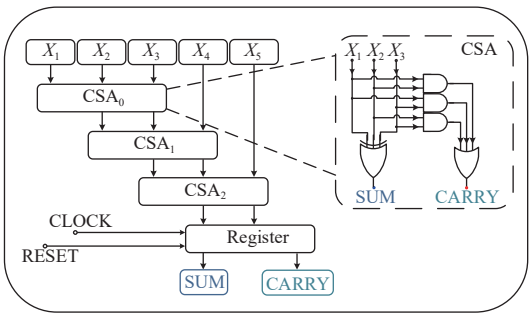

图 1 5-2 低延迟加法器结构框图

Fig. 1 Structure block diagram of 5-2 low latency adder

由图 1 可知, 5-2 低延迟加法器具有 5 个输入、2 个输出. 前 3 个操作数X_1、X_2和X_3作为第一级 CSA 的输入在第一个时钟周期参与运算, 运算结果以

CSR 表示与第 4 个操作数X_4共同作为第二级 CSA 的输入参与第二个时钟周期的计算, 依次类推输出最终计算结果. 其中 CSA 的电路组成表示在虚线框内, 3 个输入经过 3 输入异或门得到SUM, 两两相与后经过 3 输入或门得到CARRY.

结合 2.1 中提出的高效低延迟蒙哥马利模乘, 将新的操作数代入 5-2 低延迟加法器中可得.

由图 2 可知, 代入高效低延迟蒙哥马利模乘的 5-2 低延迟加法器输入和输出全部使用 CSR 表示, 以S_1、S_2及A_i即A第i位和B_1相与的结果作为第一级 CSA 的输入参与第一个时钟周期的运算, A_i和B_2相与作为第二级的输入之一参与第二个时钟周期的计算, 将S_{1_0}与S_{2_0}异或后的结果和模数N相与作为第 3 级 CSA 的输入之一参与第 3 个时钟周期的计算. 对应的算法伪代码如下.

算法 2: 代入高效低延迟蒙哥马利模乘的 5-2 低延迟加法器

输入: 操作数A_1, A_2, B_1, B_2; 模数N

输出: $A\times B \bmod N$蒙哥马利表示的 CSR

1 function new-5:2-compressor(A,B)

2 $S_1[0]=0$, $S_2[0]=0$;

3 for i=0 upto k-1 do

4 $S_1[i+1]$, $S_2[i+1]$=CSR[$S_1[i]$+ $S_2[i]$+$A_i\times(B_1+B_2)$+$S[i]_0\times N$]/2;

5 end for

6 return $S_1[k]$, $S_2[k]$

7 end function

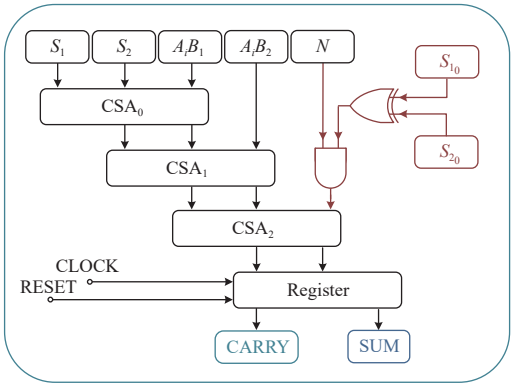

图 2　改进算法的 5-2 低延迟加法器结构框图

Fig. 2　Structure block diagram of improved algorithm using 5-2 low latency adder

3　硬件实现

基于上述改进后算法进行硬件实现, 将 FPAG 设计分为 2 个子功能模块: 基于 CSA 的 5-2 低延迟加法器和桶形移位全加器(barrel shifter adder, BSA). FPGA 系统的电路结构框图如图 3 所示.

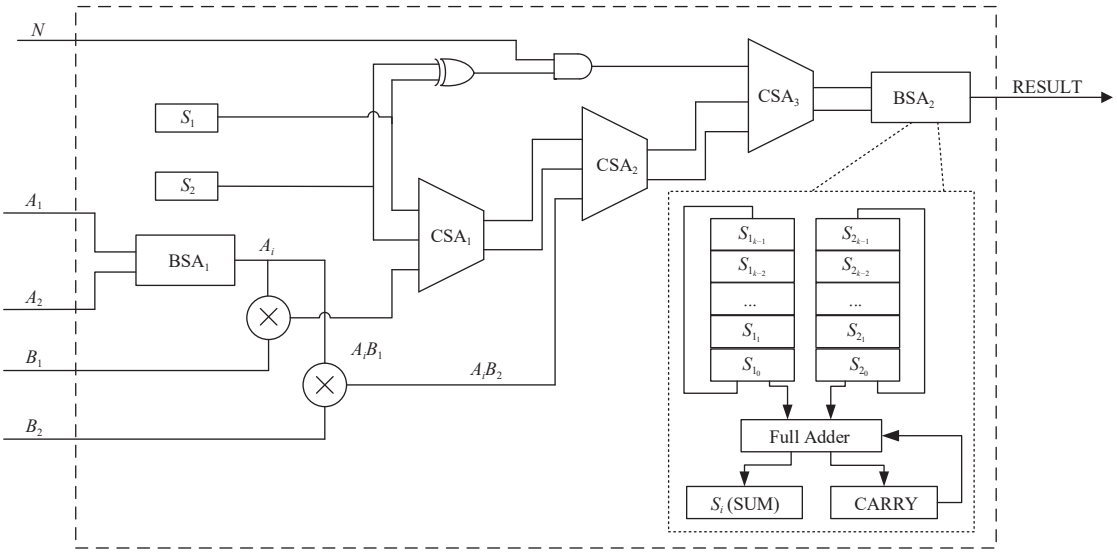

图 3　整体系统电路结构框图

Fig. 3　Circuit block diagram of the overall system

5-2 低延迟加法器模块是算法的核心模块, 也是改进后的算法速度得以提升的关键, 即图 3 中 3 个级联的 CSA.

对于运算$A\times B \bmod N$, 操作数A和B以进位保留格式作为输入, 即表示为A_1,A_2和B_1,B_2的形式. A_1与A_2作为第一个桶形移位全加器的输入, 在k轮循环

中生成第一级与第二级CSA在第i轮次中所需的操作数A_i，B_1与B_2分别与A_i相乘生成第一级与第二级在第i轮次中所需操作数.

本设计将$N\times(S_{1_0}\ \text{xor}\ S_{2_0})$置于第3级CSA输入端，在输入端计算$S_1+S_2+A_i\times(B_1+B_2)$时，可以同时得到的结果，从而最大限度地缩小了关键路径长度，系统最终关键路径长度包含2个BSA与3层CSA.

由图3可看出，桶形移位全加器由2组移位寄存器和全加器组成，桶形移位寄存器随时钟完成数据的移位操作，全加器将移位寄存器与上一轮次的进位(CARRY)作为输入相加，输出本轮次的SUM与CARRY. 桶形移位全加器的作用：1)是将保留进位格式的输入通过移位相加得到A_i作为5-2低延迟加法器的输入之一，5-2低延迟加法器将接收到的A_i连同保留进位表示下的B_1,B_2,S_1,S_2作为输入进行计算，经过多轮循环得到CSR形式下的结果；2)是以5-2低延迟加法器的输出作为输入，将其从CSR形式转回直接表示得到最终结果. 图中所示输出最终结果(RESULT)的BSA输入位宽设计为2位，操作数为1 024位的情况下需512个时钟周期即可完成运算；在改进后的算法结构中，需要一个时钟周期将$S_1[0]$和$S_2[0]$复位，同时由于采用了以$B\times2$作为输入，会导致循环计算过程中增加一个时钟周期，因此改进后的算法共需$k+512+2$个时钟周期.

4 实验结果与分析

为了对改进后的算法予以验证，本文使用Vivado 2017.4软件对其功能进行了实验仿真. 设计在Xilinx Kintex7 FPGA上实现，芯片型号为XC7K325T. 在1 024位的模乘运算场景下，系统消耗8 219个查找表(look up table, LUT)，系统主频可达278 MHz，单次模乘运算耗时5.53 μs.

本设计与其他算法的对比如表1所示，表中所有算法指标均在1 024位计算场景下得出，其中面积时间积指标综合考量算法的资源占用与计算速度，计算方法为LUT数量×时间×10^{-3}，等效ATP计算方法为等效LUT数量(LUT数+DSP等效LUT数)×时间×10^{-3}.

由表1可知，本文算法与文献[16－17]所提算法均使用纯逻辑实现. 相比文献[16]所提算法，本文算法提升了164%的系统主频，节省了45%的运算时间. 虽然本文算法使用了更多的LUT资源，但是通过比较ATP可知，本文算法具备更小的面积时间积，在计算时间和资源消耗上达到了更好的平衡. 与文献[17]所提算法相比，本文算法节省73% LUT数，同样实现了更小的ATP.

表1 资源消耗对比

Tab. 1 Resource consumption comparison

资源	本文	文献[16]	文献[17]	文献[18]
LUT/个	8 219	5 310	30 467	2 317
DSP/个	0	0	0	131
频率/MHz	278	105	/	237
计算时间/μs	5.53	10.09	2.16	3.6
ATP	45.45	53.58	65.08	8.34
等效ATP	45.45	53.58	65.80	1 176.02
等效提升/%		15	30	96

与文献[18]所提算法相比，本文算法提升了17%的系统主频. 文献[18]所提算法消耗LUT数量低于本文算法，由表1可知原因是其使用了DSP资源，而本算法未使用. 采用文献[19]中的等效方法计算，即一个DSP可等效为619个slices(Xilinx FPGA中最小可编程逻辑单元之一)，每个slice内部包含4个LUT)，从而文献[18]提出算法所用LUT等效数量为131 619×4+2 317=326 673个，远超本文算法消耗的7 191个LUT资源，从而可知其等效ATP为1 176.02，远大于本文算法的45.45. 进一步地，由于本文提出的算法采用的是纯逻辑实现，从而更适合在嵌入式设备等资源敏感性的设备中实现，也更适合ASIC实现.

5 结 论

本文通过分析目前已有文献中对经典蒙哥马利模乘算法的硬件实现，在算法与电路设计两个方面提出改进方案，改进的方案采用针对性的硬件结构设计，减少计算量，进一步提高经典蒙哥马利模乘算法的执行效率. 改进后的设计在大数模乘计算的场景取得更好的ATP表现，相比已有同类算法等效ATP提升了15%以上，在保障性能的前提下，节省了硬件资源，在小型设备和嵌入式设备中更加具有优势.

参考文献：

[1] IQBAL W, ABBAS H, DANESHMAND M, et al. An in-Depth analysis of iot security requirements, challenges, and their countermeasures via software-defined security[J]. IEEE Internet of Things Journal, 2020, 7(10): 10250 – 10276.

[2] RIVEST R L, SHAMIR A, ADLEMAN L. A method for obtaining digital signatures and public-key cryptosystems[J].

Communications of the ACM, 1978, 21(2): 120–126.

[3] MENEZES A J, VANSTONE S A, OORSCHOT P. Handbook of applied cryptography[M]. Florida: CRC Press, 2018.

[4] 高巍, 骆宜萱, 李佳琨, 等. 素数域椭圆曲线密码点乘的高性能硬件实现 [J]. 北京理工大学学报, 2021, 41(9): 977 – 984.
GAO Wei, LUO Yixuan, LI Jiakun, et al. High performance hardware implementation of elliptic curve cryptography point multiplication over GF(p) [J]. Transactions of Beijing Institute of Technology, 2021, 41(9): 977 − 984. (in Chinese)

[5] 李佳琨, 李喆, 张靖奇, 等. 基于改进 $x \sim (2 \sim n)$ 次方器的二进制域快速模逆 [J]. 北京理工大学学报, 2020, 40(7): 765 – 770.
LI Jiakun, LI Zhe, ZHANG Jingqi, et al. A fast modular inversion architecture over GF(2^m) based modified $x \sim (2 \sim n)$ units [J]. Transactions of Beijing Institute of Technology, 2020, 40(7): 765 − 770. (in Chinese)

[6] MONTGOMERY P L. Modular multilication without trial division[J]. Math of Computation, 1985, 44(170): 519 – 521.

[7] ZHANG B, CHENG Z, PEDRAM M. High-radix design of a scalable montgomery modular multiplier with low latency[J]. IEEE Transactions on Computers, 2022, 71(2): 436 – 449.

[8] KOLAGATLA V R, DARBAR M J S, SELVAKUMAR D, et al. A randomized montgomery powering ladder exponentiation for side-channel attack resilient RSA and leakage assessment[C]//2021 25th International Symposium on VLSI Design and Test (VDAT). [S.l.]: IEEE, 2021: 1 – 5.

[9] JANANI V S, MANIKANDAN M S K. A secured key management scheme for mobile ad hoc networks with modified montgomery modular arithmetic[C]//2022 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES). [S.l.]: IEEE, 2022: 1 – 4.

[10] SRINITHA S, NIVEDA S, RANGEETHA S, et al. A high speed montgomery multiplier used in security applications [C]//2021 3rd International Conference on Signal Processing and Communication (ICPSC). Coimbatore, India: IEEE, 2021: 299 – 303.

[11] 赵云正. 基于 RSA/SHA-256 加解密协处理器的研究与设计 [D]. 成都: 电子科技大学, 2022.
ZHAO Yunzheng. Research and design of RSA/SHA-256 based encryption and decryption coprocessor[D]. Chengdu: University of Electronic Science and Technology, 2022.(in Chinese)

[12] 蒲誉文. 物联网安全与隐私保护关键技术研究 [D]. 重庆: 重庆大学, 2021.
PU Yuwen. Research on key technologies for security and privacy protection of internet of things[D]. Chongqing: Chongqing University, 2021. (in Chinese)

[13] KOC C, ACAR T, JR B. Analyzing and comparing montgomery multiplication algorithms[J]. Micro IEEE, 1996, 16(3): 26 – 33.

[14] IBRAHIM A, GEBALI F, EL-SIMARY H, et al. High-performance, low-power architecture for scalable radix 2 montgomery modular multiplication algorithm[J]. Canadian Journal of Electrical and Computer Engineering, 2009, 34(4): 152–157.

[15] WALTER C D. Still faster modular multiplication[J]. Electronics Letters, 2002, 31(4): 263–264.

[16] CHEN D D, YAO G X, CHEUNG R C C, et al. Parameter space for the architecture of FFT-based montgomery modular multiplication[J]. IEEE Transactions on Computers, 2016, 65(1): 147 – 160.

[17] DAI W, CHEN D D, CHEUNG R C C, et al. Area-time efficient architecture of FFT-based montgomery multiplication [J]. IEEE Transactions on Computers, 2017, 66(3): 375–388.

[18] 程碧倩, 刘光柱, 肖昊. 改进的蒙哥马利模乘算法及 FPGA 实现 [J]. 电子科技, 2022, 35(7): 58–63.
CHENG B Q, LIU G Z, XIAO H. Improved Montgomery modulo multiplication algorithm and FPGA implementation[J]. Electronic Science and Technology, 2022, 35(7): 58 − 63. (in Chinese)

[19] HAO Y, ZHONG S, MA M, et al. Lightweight architecture for elliptic curve scalar multiplication over prime field[J]. Electronics, 2022, 11(14): 2234.

（责任编辑：刘芳）