

一种改进的二进制左移模逆算法及其 ASIC 实现

马 帅, 李树国

(清华大学 微电子学研究所, 北京 100084)

摘 要: 针对已有的二进制左移模逆算法提出了一种改进算法. 这种改进的二进制左移算法相对于原算法, 在基本不增加运算步骤的情况下, 可以减少电路面积、降低功耗并简化比较逻辑. 实验结果表明, 改进后的算法与原算法相比, 电路面积减小了 16.4%, 功耗降低了 26.8%, 模逆运算速度增加了 7.6%.

关键词: 模逆; 左移; 低功耗; ASIC 实现

中图分类号: TP31

文献标识码: A

文章编号: 1000-7180(2013)12-0001-04

An Improved Left-Shift Binary Algorithm for Modular Inverse and ASIC Implementation

MA Shuai, LI Shu-guo

(Institute of Microelectronics, Tsinghua University, Beijing 100084, China)

Abstract: An improved algorithm for the left-shift binary algorithm is proposed. Compared with the original algorithm, the improved algorithm calculates the modular inverse in small circuit area, low power and simple logical comparison with almost the same number of operations. The experimental results show that the area and the power of circuit are reduced by 16.4% and 26.8% respectively and the computation speed is increased by 7.6%.

Key words: modular inverse; left-shift; low power design; ASIC implementation

1 引言

随着信息安全的快速发展, 基于数学难题的公钥密码体制得到了广泛应用. 椭圆曲线密码(elliptic curve cryptography, ECC)因其具有密钥短、速度快、安全性高、节省资源等一系列优点而广泛应用于移动通信、电子商务等领域^[1]. 大数模逆运算作为一种模运算, 是素数域椭圆曲线密码中不可或缺的运算. 在椭圆曲线密码体制中, 点加和倍点运算都需要用到求逆运算^[2]. 随着银行 IC 卡的产业升级, 非接触式金融 IC 卡将逐步增加^[3], 面向低端的密码系统应用也越来越迫切, 因此操作简便、面积小、功耗低的大数模逆算法具有重要的研究意义和应用价值.

本文在已有的二进制左移模逆算法的基础上提出了一种改进算法. 与原算法相比, 这种算法具有更

精简的操作步骤, 占用更少的存储单元, 从而减少了电路面积, 降低了电路功耗, 更好地满足了低端密码系统的低功耗要求.

2 几种求解模逆的算法

素数域 $GF(p)$ 上整数 a 的模逆定义为^[4]: 对于素数 p 和整数 $a \in [1, p-1]$, 存在唯一的整数 $r \in [1, p-1]$, 使得 $r \cdot a \equiv 1 \pmod{p}$, 则整数 r 为 a 的逆元, 记作: $r \equiv a^{-1} \pmod{p}$.

目前已经有的求解素数域上模逆的算法包括扩展欧几里得算法、二进制右移算法和二进制左移算法.

扩展欧几里得算法利用辗转相除法实现了模逆的求解, 但其缺点在于操作步骤中有复杂的除法运算, 不利于计算机处理. 二进制右移算法利用简单的

收稿日期: 2013-01-12; 修回日期: 2013-03-08

基金项目: 国家“八六三”计划(2012AA012402); 国家自然科学基金(61073173); 清华大学自主研发计划(2011Z05116)

移位操作和加法运算代替了扩展欧几里得算法中的除法运算. 而下面的二进制左移算法(算法一)则通过左移的方式去掉了加法运算, 节省了硬件资源.

3 二进制左移算法

文献[5]提出的二进制左移算法通过巧妙地使用左移的思想, 避免了右移算法中的加法操作, 见算法 1. 其中, c_u, c_v 分别表示 u, v 左移的次数, 在整个操作过程中, 等式 $ra + mp = u/(2^{\min(c_u, c_v)})$, $sa + tp = v/(2^{\min(c_u, c_v)})$ ^[6] 始终保持成立. p 的位数记为 $n = \lceil \log_2 p \rceil$.

算法 1: 二进制左移算法

输入: $a \in [1, p-1]$ 和 p

输出: $r \in [1, p-1]$, 其中 $r = a^{-1} \pmod{p}$

(1) $u \leftarrow p, v \leftarrow a, r \leftarrow 0, s \leftarrow 1$

(2) $c_u \leftarrow 0, c_v \leftarrow 0$

(3) 当 $(u \neq \pm 2^{c_u} \text{ 且 } v \neq \pm 2^{c_v})$ 时, 重复执行
若 $(u_n, u_{n-1} = 0)$ 或 $(u_n, u_{n-1} = 1 \text{ 且 } \text{OR}(u_{n-2},$

$\dots, u_0) = 1)$, 则

若 $(c_u \geq c_v)$ 则

$u \leftarrow 2u, r \leftarrow 2r, c_u \leftarrow c_u + 1$

否则

$u \leftarrow 2u, s \leftarrow s/2, c_u \leftarrow c_u + 1$

否则若 $(v_n, v_{n-1} = 0)$ 或 $(v_n, v_{n-1} = 1 \text{ 且 } \text{OR}(v_{n-2},$

$\text{OR}(v_{n-2}, \dots, v_0) = 1)$, 则

若 $(c_v \geq c_u)$, 则

$v \leftarrow 2v, s \leftarrow 2s, c_v \leftarrow c_v + 1$

否则

$v \leftarrow 2v, r \leftarrow r/2, c_v \leftarrow c_v + 1$

否则

若 $(v_n = u_n)$, 则

oper = “-”

否则

oper = “+”

若 $(c_u \leq c_v)$, 则

$u \leftarrow u \text{ oper } v, r \leftarrow r \text{ oper } s$

否则

$v \leftarrow v \text{ oper } u, s \leftarrow s \text{ oper } r$

(4) 若 $(v = \pm 2^{c_v})$, 则

$r \leftarrow s, u_n \leftarrow v_n$

(5) 若 $(u_n = 1)$, 则

若 $(r < 0)$, 则

$r \leftarrow -r$

否则

$r \leftarrow p - r$

(6) 若 $(r < 0)$, 则

$r \leftarrow r + p$

(7) 返回 r

4 改进的二进制左移算法

在上述算法一中, 跳出循环的条件为 $u = \pm 2^{c_u}$ 或 $v = \pm 2^{c_v}$, 在硬件实现时, 需要 4 组 n 比特存储单元分别存放 $2^{c_u}, -2^{c_u}, 2^{c_v}, -2^{c_v}$, 并分别比较 u 和 $\pm 2^{c_u}$ 以及 v 和 $\pm 2^{c_v}$ 是否相等, 而且随着 c_u 和 c_v 的增加, 要不断通过左移更新 $\pm 2^{c_u}$ 和 $\pm 2^{c_v}$ 的值. 当 p 较大时, 会消耗较多的硬件资源. 为此, 本文重点提出对算法 1 的改进——算法 2. 在算法 2 中, 跳出循环的条件改为 $c_u = n-1$ 或 $c_v = n-1$, 也就是说将原来的 u, v 与 $\pm 2^{c_u}, \pm 2^{c_v}$ 的比较换成 c_u, c_v 与常数 $n-1$ 的比较. 这样, 算法 2 相对于算法 1 就有两方面的改进: 将存储单元的数量从 $4n$ 降到 $2\lceil \log_2 n \rceil$; 省掉了 $\pm 2^{c_u}$ 和 $\pm 2^{c_v}$ 的更新操作. 另外, 算法 2 需要增加 $\lceil \log_2 n \rceil$ 比特变量 d , 用于记录多左移的次数.

算法 2: 改进的二进制左移算法

输入: $a \in [1, p-1]$ 和 p

输出: $r \in [1, p-1]$, 其中 $r = a^{-1} \pmod{p}$

(1) $u \leftarrow p, v \leftarrow a, r \leftarrow 0, s \leftarrow 1$

(2) $c_u \leftarrow 0, c_v \leftarrow 0, d \leftarrow 0$

(3) 若 $(a = 1)$, 则

返回 $r = 1$

否则

执行第(4)步

(4) 当 $(c_u \neq n-1 \text{ 且 } c_v \neq n-1)$ 时, 重复执行

若 $(u_n, u_{n-1} = 0)$ 或 $(u_n, u_{n-1} = 1 \text{ 且 } \text{OR}(u_{n-2}, \dots, u_0) = 1)$, 则

若 $(c_u \geq c_v)$, 则

$u \leftarrow 2u, r \leftarrow 2r, c_u \leftarrow c_u + 1, d \leftarrow d + 1$

否则

$u \leftarrow 2u, s \leftarrow s/2, c_u \leftarrow c_u + 1$

否则若 $(v_n, v_{n-1} = 0)$ 或 $(v_n, v_{n-1} = 1 \text{ 且 } \text{OR}(v_{n-2},$

OR($v_{n-2}, \dots, v_0$) = 1) 则

若($c_v \geq c_u$), 则

$v \leftarrow 2v, s \leftarrow 2s, c_v \leftarrow c_v + 1, d \leftarrow d + 1$

否则

$v \leftarrow 2v, r \leftarrow r/2, c_v \leftarrow c_v + 1$

否则

$d \leftarrow 0$

若($v_n = u_n$), 则

oper = “-”

否则

oper = “+”

若($c_u \leq c_v$), 则

$u \leftarrow u \text{ oper } v, r \leftarrow r \text{ oper } s$

否则

$v \leftarrow v \text{ oper } u, s \leftarrow s \text{ oper } r$

(5) 若($c_v = n - 1$), 则

$r \leftarrow s, u_n \leftarrow v_n$

(6) 当($d > 0$) 时, 重复执行

$r \leftarrow r/2, d \leftarrow d - 1$

(7) 若($u_n = 1$), 则

若($r < 0$), 则

$r \leftarrow -r$

否则

$r \leftarrow p - r$

(8) 若($r < 0$), 则

$r \leftarrow r + p$

(9) 返回 r

5 改进算法的分析及电路结构图

算法 2 的思想在于改变算法 1 的循环结束条件, 即 $u = \pm 2^{c_u}$ 或 $v = \pm 2^{c_v}$. 具体表现为: 在算法 2 中, 当 $u = \pm 2^{c_u}$ (或 $v = \pm 2^{c_v}$) 时并不跳出循环, 而是将 u (或 v) 继续左移直至其变为 $\pm 2^{n-1}$, 这时 c_u (或 c_v) 恰好等于 $n - 1$, $c_u = n - 1$ 或 $c_v = n - 1$, 就是算法 2 跳出循环的条件. d 用来记录多移动的次数, 跳出循环后再相应地进行 d 次右移操作以得到正确的运算结果. 为了得到准确的 d , 需要对 d 进行如下操作: 将 d 初始化为 0; 当 u (或 v) 左移一位时, 若 $c_u \geq c_v$ ($c_v \geq c_u$), 则对 d 进行加 1 操作, 否则不对 d 做操作; 当 u 和 v 都是奇数时将 d 赋值为 0. 下面将说明 d 的平均值很小.

对算法 1 在 $n = 256$ 情况下的左移次数进行了统计. 将 p 固定在 256 比特 (最高位为 1), 利用 MIRACL 库^[7] 所提供的随机数产生函数, 对素数 p 以及 $a \in [1, p-1]$ 随机选取 1.2 亿次, 统计得循环结束即 $u = \pm 2^{c_u}$ 或 $v = \pm 2^{c_v}$ 时, c_u 或 c_v 的平均值为 253 ($u = \pm 2^{c_u}$ 时统计 c_u , $v = \pm 2^{c_v}$ 时统计 c_v), 也就是说平均再左移两次就满足算法 2 的循环结束条件 ($c_u = n - 1 = 255$ 或 $c_v = n - 1 = 255$). 所以算法 2 相对于算法 1 只增加了很少的操作 (平均两次左移和两次右移) 就左移到最高位. 算法 2 中的 d 即用于记录多左移的次数, 如上面所述其平均值为 $(n - 1) - 253 = 2$. 因此, 与算法 1 相比, 算法 2 在简化了操作的同时, 对操作步骤数的影响可以忽略不计.

改进后的算法必须左移固定的次数才能结束, 对于给定的 p , 设 p 的位数 n , 则循环结束条件用 $n - 1$ 来判断. 如果 p 的位数小于 n , 不妨设为 m 位 ($m < n$), 循环结束条件也可以用 $n - 1$ 来判断, 只不过会比使用 $m - 1$ 多做 $n - m$ 次左移和右移操作.

图 1 为算法 2 的电路结构简图, 图中给出了计算 u, v, r, s, c_u, c_v, d 等的通路. 其中加法器 1 和加法器 2 都是 n 位加法器, 而加法器 3 为 $\lceil \log_2 n \rceil$ 位特殊加法器 (其中一个输入固定为 1).

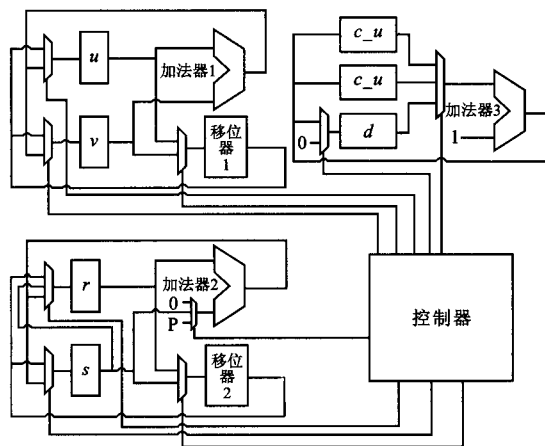


图 1 改进的左移二进制算法(算法 2)的电路结构简图

6 硬件实现与算法比较

我们用 Verilog 硬件描述语言实现两种左移二进制模逆算法 ($n = 256$) 并在 SMIC 0.18 μm 工艺以及 SMIC 65 nm 工艺下进行综合. 表 1 分别给出了两种工艺下各项指标的对比. 从表 1 和表 2 中可看出, 与原算法(算法 1)相比, 改进后的二进制左移算

法(算法 2)的面积、功耗都明显降低,延时也有所减少,模逆运算的速度也略有提高. 具体表现为:在 SMIC 0.18 μm 工艺下,面积减小了 16.4%,功耗降低了 26.8%,延时缩短了 7.6%,模逆运算速度增加了 7.6%;在 SMIC 65 nm 工艺下,面积减小了 17.6%,功耗降低了 29.4%,延时缩短了 8.1%,模逆运算速度增加了 7.6%.

表 1 左移模逆算法的各项指标(SMIC 0.18 μm)

算法	面积/ 万门	功耗/ mW	延时/ ns	时钟 周期数	运算速度/ (次/秒)
算法 1	6.7	72.0	2.89	702	492 907
算法 2	5.6	52.7	2.67	706	530 498

表 2 左移模逆算法的各项指标(SMIC 65 nm)

算法	面积/ 万门	功耗/ mW	延时/ ns	时钟 周期数	运算速度/ (次/秒)
算法 1	0.749	12.6	2.09	702	681 580
算法 2	0.617	8.9	1.92	706	737 724

7 结束语

文中针对二进制左移算法提出了一种改进的二进制左移算法,这种算法在操作步骤上与原算法基本相同,而在一定程度上降低了操作复杂度. 在用 verilog 硬件描述语言实现并在 SMIC 0.18 μm 和 SMIC 65 nm 工艺下综合时,改进后的算法相比原算法在面积、功耗方面都有明显的降低,在延时和运算速度方面也有一定程度的改善.

参考文献:

[1] 张金中,寇应展,王韬,等. 针对滑动窗口算法的椭圆曲线密码故障分析[J]. 通信学报,2012,33(1):71-78.
[2] 张霄鹏,李树国. 素数域参数可选高速椭圆曲线密码芯

片 ASIC 设计实现[J]. 微电子学,2008,38(3):330-337.
[3] 李煜平. 金融 IC 卡在轨道交通领域的应用研究[J]. 城市轨道交通研究,2012,15(5):28-31.
[4] Darrel Hankerson, Menezes Alfred, Vanstone Scott. 椭圆曲线密码学导论[M]. 张焕国,译. 北京:电子工业出版社,2005.
[5] Lorenz Robert. New algorithm for classical modular inverse[C]// Cryptographic Hardware And Embedded Systems, 2002, Lecture Notes in Computer Science. Redwood Shores:Springer,2003:57-70.
[6] Hars Laszlo. Modular inverse algorithms without multiplications for cryptographic applications[J]. EUR-ASIP Journal on Embedded System, 2006:1-13.
[7] Certivox. Multiprecision integer and rational arithmetic C/C++ library (MIRACL) [EB/OL]. [2012-12-29]. <https://certivox.com/solutions/miracle-crypto-sdk/>.

作者简介:



马 帅 男,(1990—),硕士. 研究方向为信息安全算法的数字大规模集成电路设计与实现.



李树国 男,(1963—),教授,博士生导师. 研究方向为信息安全算法的数字大规模集成电路设计与实现.