基于有符号数字系统的 Montgomery 模逆算法及其硬件实现

陈 琳,孙万忠,陈性元,戴紫彬
(解放军信息工程大学,河南郑州 450004)

摘 要: 在椭圆曲线密码中,模逆运算是有限域运算中最复杂、最耗时且硬件实现难度最大的运算.本文在 Kaliski 算法的基础上,提出了基于有符号数字系统的 Montgomery 模逆算法,它支持素数域和二进制域上任意多精度参数的求模逆运算.据此算法,设计了相应的硬件结构方案,并给出了面积复杂度和时间复杂度分析.仿真结果表明,相比于其它模逆算法硬件设计方案,本文提出的基于有符号数字系统的 Montgomery 模逆算法在运算速度、电路面积、灵活性等方面具有显著的优越性.

关键词: 椭圆曲线密码;有符号数字系统;可伸缩;双有限域;模逆
中图分类号: TP309 **文献标识码:** A **文章编号:** 0372-2112 (2012) 03-0489-06
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2012.03.013

Montgomery Modular Inversion Algorithm Based on Signed Digit System and Hardware Implementation

CHEN Lin, SUN Wan-zhong, CHEN Xing-yuan, DAI Zi-bin
(*PLA Information Engineering University, Zhengzhou, Henan* 450004, *China*)

Abstract: Modular inversion is the most complex and time consuming finite fields arithmetic operation in elliptic curve cryptography (ECC). Further more, the hardware implementation of modular inversion is difficult. Based on signed digit system theory, this paper presents a scalable dual-field Montgomery modular inversion algorithm which supports inversion algorithm of any precision parameters in the prime fields and binary fields. According to the algorithm, the hardware architecture is designed and analyzed in area complexity and time complexity. Simulation results show that the hardware design is superior to other hardware designs in operation speed, circuit area and flexibility.

Key words: elliptic curve cryptography; signed digit system; scalable; dual-field; modular inversion

1 引言

随着社会信息化的发展,信息安全已经成为世人关注的重大问题,而密码技术的飞速发展为它提供了强有力的保证.但传统的对称密码系统也逐步暴露出一些弱点,即密钥管理和分配困难、数字签名和身份认证应用困难等.因此公钥密码系统,如 RSA 算法、ECC 算法以及 Diffie-Hallman 密钥交换算法,已成为信息安全领域中的研究热点.而在公钥密码系统中,由 Neal Koblitz 和 Victor Miller 提出的椭圆曲线密码(ECC),由于其在安全性、计算量、处理速度、存储空间、带宽要求、实现功耗等方面的许多优势,目前已逐渐取代 RSA 密码算法成为事实上的下一代公钥密码标准.但是现有的公钥密码处理设备已经无法满足现代通信网络对密码处理高效性和灵活性的要求.因此迫切需要从速度、灵活性与电路面积之间的矛盾出发,以提高密码处理的高效性与灵活性为目标,提出能够满足密码高速处理的算法需求和硬件结构.

在椭圆曲线密码中,模逆运算是有限域运算中最复杂、最耗时且硬件实现难度最大的运算[1].模逆运算的时间约相当于 10 个模乘的时间.因此模逆运算的运行速度会直接影响到整个椭圆曲线密码系统的工作效率.在实际工程实现中研究人员通常利用坐标转换将椭圆曲线转换到投影坐标系下,用多次模乘运算代替模逆运

收稿日期:2011-01-21;修回日期:2011-08-02
基金项目:国家 863 高技术研究发展计划(No.2008AA01Z0103)

算来达到减少模逆运算使用次数的目的[2].但是由于点乘运算结果由投影坐标转换回仿射坐标仍至少需要一次模逆运算,在实现中仍然无法完全避免模逆运算的使用.

目前国内外针对模逆算法的研究非常多,提出了很多适合素数域和二进制域的模逆算法.其中比较著名的算法是基于费马小定理(Fermat theorem)的费马算法、扩展的 Euclidean 算法以及 Montgomery 模逆算法[3,4].由于在素数域和二进制域下利用费马小定理求模逆时使用模乘次数过多,因此利用费马算法设计双有限域上的求模逆硬件非常不现实,本文不予讨论.目前大多数求模逆运算的硬件设计是基于扩展的 Euclidean 算法实现的.但是该算法仅面向二进制域,无法实现二进制域到素数域的扩展,而且面积复杂度非常高,硬件资源消耗巨大.Montgomery 模逆算法又称为 Kaliski 算法,它是在扩展的 Euclidean 算法基础上发展而来的,它使得 Montgomery 域数的运算速度和灵活性有所提高.但是它仍然不能满足密码处理的高效性和灵活性要求.

在上述背景下,本文提出了基于有符号数字系统的 Montgomery 模逆算法.通过引入有符号数字系统理论,将支持单域的 Montgomery 模逆算法改进为支持双有限域的 Montgomery 模逆算法,使得算法具备可伸缩性,达到降低关键数据路径延迟、节省电路面积、提高电路工作效率的目的.最后设计了相应的可伸缩双域模逆器硬件结构.该结构相比于其它模逆算法硬件设计方案,在运算速度、电路面积、复杂度等方面具有显著的优越性.

2 Kaliski 算法

1995 年 Kaliski 在文献[3]中采用 Montgomery P L 的思想,在扩展的 Euclidean 算法的基础上,首次引入了 Montgomery 模逆的概念,称之为原始的 Montgomery 模逆算法或 Kaliski 算法.假设 p 是一个 n 比特素数,a 是小于 p 的一个非负整数,文献中给出的 Montgomery 模逆定义为:

$$\mathrm{MontInv}(a) = a^{-1}R \bmod p \tag{1}$$

其中 $R = 2^m$, $m \geqslant n$, m 为 Montgomery 常数.

Kaliski 算法如图 1 所示,它分为两个阶段.阶段 1 称为近似 Montgomery 模逆,它输入 a 与 p,输出 r 与 k,并且 $r = a^{-1}2^k \bmod p$, $n < k < 2n$(n 为模数 p 的比特位宽,即 $n = \lceil \log_2 p \rceil$).阶段 2 称为 Montgomery 模逆修正,它将阶段 1 的输出作为输入,输出 Kaliski 算法的最终结果 $x = a^{-1}2^n \bmod p$.而通过修改算法阶段 2 的迭代次数,也可以直接获得 a 普通整数域下的逆元 a^{-1}.本文将该算法称为算法 1.

阶段 1:近似 Montgomery 模逆
Input: a 与 p,且 $a \in [1, p-1]$.
Output: r 与 k,且 $r = a^{-1}2^k \bmod p$, $n < k < 2n$.
Step 1: $u = p, v = a, r = 0, s = 1$; Step 2: $k = 0$; Step 3:当 $v > 0$ 时,重复执行 (1)IF u 是偶数,THEN $u = u/2, s = 2s$; (2)ELSE IF v 为偶数,THEN $v = v/2, r = 2r$; (3)ELSE IF $u > v$,THEN $u = (u - v)/2, r = r + s, s = 2s$; (4)ELSE $v = (v - u)/2, s = r + s, r = 2r$; (5)$k = k + 1$; Step 4: IF $r \geqslant p$,THEN $r = r - p$; Step 5: RETURN $r = p - r$ 与 k;
阶段 2:Montgomery 模逆修正
Input:阶段 1 输出结果 r, k 及 m, $m \geqslant n$.
Output: $x = a^{-1}2^n \bmod p$.
Step 6:对于 i 从 1 到 $k - m$,重复执行 (1)IF r 为偶数,THEN $r = r/2$; (2)ELSE $r = (r + p)/2$; Step 7: RETURN $x = r$.

图 1 Kaliski 算法

3 基于有符号数字系统的 Montgomery 模逆算法

3.1 GF(2^n)域上的 Montgomery 模逆算法

原始 Montgomery 模逆算法仅是针对素数域的求逆运算提出来的.由于素数域与二进制域具有不同的性质并且运算存在一定的差异,因而它不能直接应用于二进制域.但是二进制域 GF(2^n)的元素可以表示为多项式基的形式[5].在多项式基表示法中,素数域和二进制域上的元素都可以表示为二进制方式,因此理论上可以对两个有限域上的求逆算法采用统一的形式,此处证明省略.

令 $p(x) = x^n + p_{n-1}x^{n-1} + p_{n-2}x^{n-2} + \cdots + p_1 x + p_0$ 是二元域上的既约多项式.以 $p(x)$作为约减多项式构造二进制域 GF(2^n). GF(2^n)域上任一元素可以表示为多项式 $a(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1 x + a_0$,系数 $a_i \in \{0,1\}$.而域 GF(2^n)上的运算就可以转化为规则的多项式运算,即 GF(2^n)域元素加法转换为两个多项式系数的模 2 相加,域元素乘法转换为两个多项式相乘后取模 $p(x)$.扩展至二进制域 GF(2^n)上的 Montgomery 模逆算法如图 2 所示,它也是由近似 Montgomery 模逆与模逆修正两个阶段构成.本文将该算法称为算法 2.

基于算法 2 有 3 个定理成立,其证明过程省略.

定理 1 若 $p(x)$为既约多项式,多项式次数 $\deg(p(x)) > \deg(a(x)) > 0$,则 Montgomery 模逆算法中的中间二进制多项式 $r(x)$、$s(x)$、$u(x)$、$v(x)$的次数永远保持在$[0, \deg(p(x))]$范围内.

阶段 1:近似 Montgomery 模逆

Input: $a(x)$与 $p(x)$,且 $\deg(a(x)) < \deg(p(x))$.

Output: $r(x)$与 k,且 $r(x) = a(x)^{-1}x^k \pmod{p(x)}$,
$\deg(a(x)) \leqslant k \leqslant \deg(p(x)) + \deg(a(x)) + 1$.

Step 1: $u(x) = p(x), v(x) = a(x), r(x) = 0, s(x) = 1$;
Step 2: $k = 0$;
Step 3: 当$(v(x) \neq 0)$时,重复执行
(1) IF $u_0 = 0$, THEN $u(x) = u(x)/x, s(x) = xs(x)$;
(2) ELSE IF $v_0 = 0$, THEN $v(x) = v(x)/x, r(x) = xr(x)$;
(3) ELSE IF $\deg(u(x)) > \deg(v(x))$,
THEN $u(x) = (u(x) + v(x))/x, r(x) = r(x) + s(x), s(x) = xs(x)$;
(4) ELSE $v(x) = (v(x) + u(x))/x, s(x) = r(x) + s(x), r(x) = xr(x)$;
(5) $k = k + 1$;
Step 4: IF $\deg(r(x)) > \deg(p(x))$, THEN $r(x) = r(x) + xp(x)$;
Step 5: IF $\deg(r(x)) = \deg(p(x))$, THEN $r(x) = r(x) + p(x)$;
Step 6: RETURN $r(x)$与 k;

阶段 2:Montgomery 模逆修正

Input:阶段 1 输出结果 $r(x)$, k 及 $p(x)$, $m \geqslant n$.

Output: $b(x) = a(x)^{-1}x^{2m} \pmod{p(x)}$.

Step 7:对于 i 从 1 到 $2m - k$,重复执行 $r(x) = xr(x) + r_{n-1}p(x)$;
Step 8: RETURN $b(x) = r(x)$.

图 2　GF(2^n)域上的 Montgomery 模逆算法

定理 2　若 $p(x)$为既约多项式,多项式次数 $\deg(p(x)) > \deg(a(x)) > 0$,则 Montgomery 模逆算法第一阶段运算的迭代次数 k 的范围为 $n + 1 \leqslant k \leqslant \deg(a(x)) + n + 1$.

定理 3　若 $p(x)$为既约多项式,多项式次数 $\deg(p(x)) > \deg(a(x)) > 0$,则 Montgomery 模逆算法的第一阶段运算返回结果为 $a(x)^{-1}x^k \pmod{p(x)}$.

算法 1 和算法 2 在理论上已经将双有限域上的求逆算法实现了统一.但是由于运算过程需要连续的大数加法和减法,并不适合硬件实现.因此本文在这两种算法的基础上提出了基于字的双域 Montgomery 模逆算法,改进后的算法具备可伸缩性,能够基于字进行运算,达到节省电路面积、提高电路工作时钟频率的目的.

3.2　基于有符号数字系统的 Montgomery 模逆算法

为了进一步降低模逆算法关键数据路径延迟,提高模逆运算的整体速度,本文引入了有符号数字系统理论[6].利用有符号数字系统理论可以极大地提高电路面积、处理速度及功耗的综合性能,对提高椭圆曲线密码处理速度具有重要意义.本文采用基 2 冗余有符号数及相关运算单元对算法 2 进行了改进得到算法 3,如图 3 所示.

由于篇幅所限,这里就不再对算法 3 进行阐述.但是在改进时需要注意以下几点:

阶段 1:双域近似 Montgomery 模逆算法

Input: $a2^m \in [1, p-1]$, $m \geqslant n$(p 为既约多项式).

Output: result $= a^{-1}2^{k-m} \bmod p$, $n < k < 2n$. (result$\in [1, p-1]$ & k)

中间寄存器: u, v, r, s, x, y, z, p(位宽 $= 2n_{\max}$位)

Step1: $u = \text{convert}(p)$, $v = \text{convert}(a2^m)$, $r = 0$,
$s = 1, x = 0, y = 2^n, z = 0, k = 0$;
Step2: IF($u_{2.0} = 000$) THEN
{$u = \text{Shift}R(u,3)$; $s = \text{Shift}L(s,3)$; $k = k + 3$}, GOTO Step8;
(1) ELSE IF($u_{2.0} = 100$) THEN
{$u = \text{Shift}R(u,2)$, $s = \text{Shift}L(s,2)$, $k = k + 2$}, GOTO Step8;
(2) ELSE IF($u_{2.0} = 110$) THEN
{$u = \text{Shift}R(u,1)$, $s = \text{Shift}L(s,1)$}, GOTO Step7;
Step3: IF($v_{2.0} = 000$) THEN
{$v = \text{Shift}R(v,3)$, $r = \text{Shift}L(r,3)$, $k = k + 3$}, GOTO Step8;
(1) ELSE IF($v_{2.0} = 100$) THEN
{$v = \text{Shift}R(v,2)$, $r = \text{Shift}L(r,2)$, $k = k + 2$}, GOTO Step8;
(2) ELSE IF($v_{2.0} = 110$) THEN
{$v = \text{Shift}R(v,1)$, $r = \text{Shift}L(r,1)$}, GOTO Step8;
Step4: 令 $\varepsilon_1 = 0$,从 $i = 0$ 到 $e - 1$ 重复执行
$(s1[i], \varepsilon_1) \leftarrow \text{SADD}(u[i], v[i], \varepsilon_1)$.
$x[i] \leftarrow \text{SXOR}(v[i], u[i])$,
$z[i] \leftarrow \text{SXOR}(r[i], s[i])$.
Step5: IF $s1$ 为正数, THEN
{$u = \text{Shift}R(x,1)$, $r = z$, $s = \text{Shift}L(s,1)$}, GOTO Step7;
Step6: $s = z$, $v = \text{Shift}R(x,1)r = \text{Shift}L(r,1)$;
Step7: $k = k + 1$;
Step8: IF $v \neq 0$, THEN GOTO Step2;
Step9: IF $v = 0$, 令$\varepsilon_1 = 0, \varepsilon_2 = 0$,从 $i = 0$到 $e - 1$ 重复执行
$x[i] = \text{SXOR}(p[i], r[i])$,
$z[i] = \text{SXOR}((2p)[i], r[i])$,
$(S1[i], \varepsilon_1) \leftarrow \text{SADD}(y[i] - x[i], \varepsilon_1)$,
$(S2[i], \varepsilon_2) \leftarrow \text{SADD}(y[i] - z[i], \varepsilon_1)$;
Step10: IF $s1$ 为正数, THEN RETURN result $= x$,
ELSE IF $s2$ 为正数, THEN RETURN result $= z$,
ELSE RETURN result $= r$.

阶段 2:双域 Montgomery 模逆修正算法

Input: $r, p, m, 2^n, k$ 且 $r = a^{-1}2^{k-m} \bmod p$.

Output: result $= a^{-1}2^m \pmod{p}$.

中间变量: u, v, r, x, y, z, p(位宽 $= n_{\max}$bits)

Step11: $j = 2m - k - 1$, $x = 0$, $y = 0$, $z = 0$;
Step12: $v = \text{convert}(2p)$, $u = \text{convert}(3p)$; $s = 2^n$
Step13: 当 $j > 0$ 时,重复执行
(1) IF $j = 1$, THEN{$r = \text{Shift}L(r,1)$; $j = j - 1$}
ELSE{$r = \text{Shift}L(r,2)$; $j = j - 2$}.
(2) 令$\varepsilon_1 = 0, \varepsilon_2 = 0, \varepsilon_3 = 0$,
(3) 从 $i = 0$ 到 $e - 1$,重复执行
$x[i] = p[i] \oplus r[i]$; $y[i] = u[i] \oplus r[i]$; $z = u[i] \oplus r[i]$;
$(x[i], \varepsilon_1) \leftarrow r[i] - p[i] - \varepsilon_1$;
$(y[i], \varepsilon_2) \leftarrow r[i] - v[i] - \varepsilon_2$;
$(z[i], \varepsilon_3) \leftarrow r[i] - u[i] - \varepsilon_3$;
(4) IF $\varepsilon_3 = 0$, THEN $r = z$;
(5) IF $\varepsilon_2 = 0$, THEN $r = y$;
(6) IF $\varepsilon_1 = 0$, THEN $r = x$;
Step14: RETURN result $= r$.

图 3　基于有符号数字系统的双域 Montgomery 模逆算法

(1)中间变量及结果采用冗余有符号数表示后数据位宽均变为原来位宽的 2 倍.

(2)操作数转换成基 2 冗余有符号数后,其左移操作与右移操作发生了变化.原来对传统二进制数左移或右移一位,对应转换后左移或右移一个数位(2-位).

(3)由于传统二进制数与基 2 冗余有符号数的数字表示范围不同,在算法 3 的步骤 5 中,两数相减后,根据最高有效字的借位来判断两数大小的方法已经无效.因此需要修正为依照相减结果的正负来判断两数的大小.

4 算法硬件实现

4.1 系统结构

本文在算法 3 的基础上,设计了基于有符号数字系统的双域 Montgomery 模逆运算硬件结构,如图 4 所示.它主要由可伸缩双域运算单元、可伸缩多比特可控移位器、数据路由单元、存储单元和控制器单元五部分组成.各个单元模块在控制单元的协调下严格按照算法 3 的流程工作.本文为了有效的进行电路面积评估,对存储单元采用寄存器结构.但是在实际应用中,为了更好的满足操作数可伸缩的要求,存储单元可以采用双端口 RAM 存储器.对于数据路由单元,由于其仅消耗布线资源,因此在电路面积评估时可以将其忽略不计.

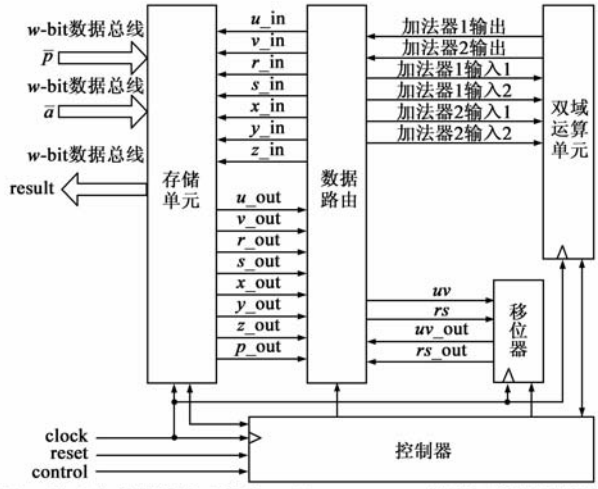

图4 基于有符号数字系统的双域Montgomery模逆运算硬件结构

4.2 双域运算单元设计

双域运算单元主要由两个 w 位传统有符号加法器、4 个 D 触发器和 1 个 1 位多路选择器构成,其结构如图 5 所示.控制器负责对该单元的运算进行启动或者停止,并对输入的 n 位有符号操作数进行迭代处理.采用三个触发器来保存有符号加法器中间的进位输出,从而实现任意 n bit 数据的多精度处理,达到节约资源的目的.

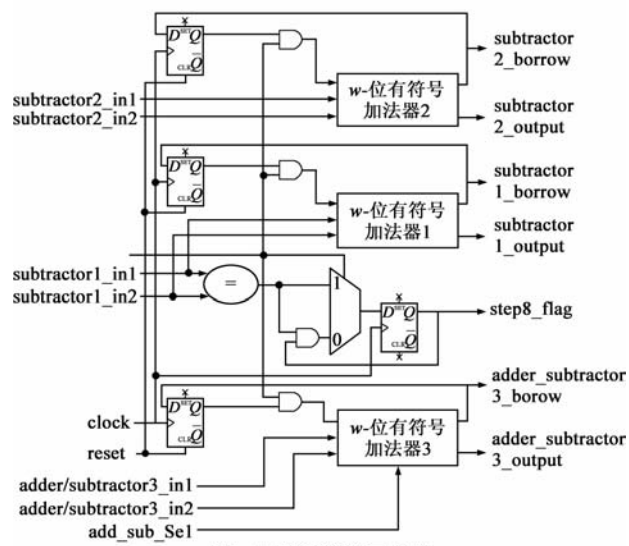

图5 双域运算单元结构

4.3 可控多比特移位单元设计

可控多比特移位单元主要由($w-1$)位寄存器、($w+3$)位寄存器、两个多路选择器构成,其结构如图 6 所示.该模块在控制器送来的控制信号作用下对外部送入的 n bit 数据进行左移 1 bit、2 bit、3 bit 处理,或者进行右移 1 bit、2 bit、3 bit 处理.移位周期数与输入数据的长度有关,移位操作时钟周期数(Number of Shift Clock)$N_OC_{\text{shift}}=\lceil n_{\max}/w\rceil$.

5 硬件性能分析

5.1 面积复杂度分析

由图 4 和图 5 可以看出,基于有符号数字系统的双域 Montgomery 模逆运算模块的面积消耗主要在处理单元 PU 上.由于外部控制逻辑根据算法流程可以采用状态机形式实现,其资源消耗相对于模逆运算模块数据路径资源消耗可以忽略不计.因此本文仅对数据路径消耗的电路面积进行分析.由图 4 可以看出,它的数据路径是由双域运算单元、存储单元、可控多比特移位单元、数据路由单元构成.下面分别对各单元的面积进行分析(均以标准门计).

(1)若双域运算单元处理字宽为 w 位,则它由 3 个 w 位的有符号加法器、4 个 1-位触发器、运算核心单元构成,因此双域运算单元消耗的电路面积为:

$$\begin{aligned}A_{\text{AU}}&=3\cdot w\cdot A_{\text{SADD}}+4\cdot A_{\text{DFF}}+A_{\text{MUX}}+A_{\text{other\_logic}}\\&=3w\cdot 19+4\cdot 2\cdot 3+3+4\\&=57w+31\end{aligned}\tag{2}$$

(2)消耗电路面积较大的是可控多比特移位单元,它主要由寄存器与多路选择器构成,其中每个多路选择器可看作由两个 w 位 2 选 1 选择器组成.该单元占用的电路面积为:

$$A_{shf\_unit} = A_{reg(w+3)} + A_{reg(w-1)} + 2\cdot A_{MUX(w)}$$
$$= 2\cdot(w+3)\cdot 3 + 2\cdot(w-1)\cdot 3 + 2\cdot 2\cdot w\cdot 3$$
$$= 24w + 12 \quad (3)$$

(3)存储单元实际上主要由 8 个 n-位寄存器构成,用于暂存算法用到的输入操作数或中间结果.该单元占用的电路面积为:

$$A_{mem} = 8\cdot 2\cdot n\cdot 3 = 48n \quad (4)$$

因此,模逆运算模块占用的总电路面积 A_{ModInv}为:

$$A_{ModInv} = A_{PU} + A_{shf\_unit} + A_{mem} \quad (5)$$

通过以上分析可知,模逆运算模块占用的面积由其最大支持操作数长度及运算单元的处理字宽决定,因此其面积复杂度为 $O(n)$.

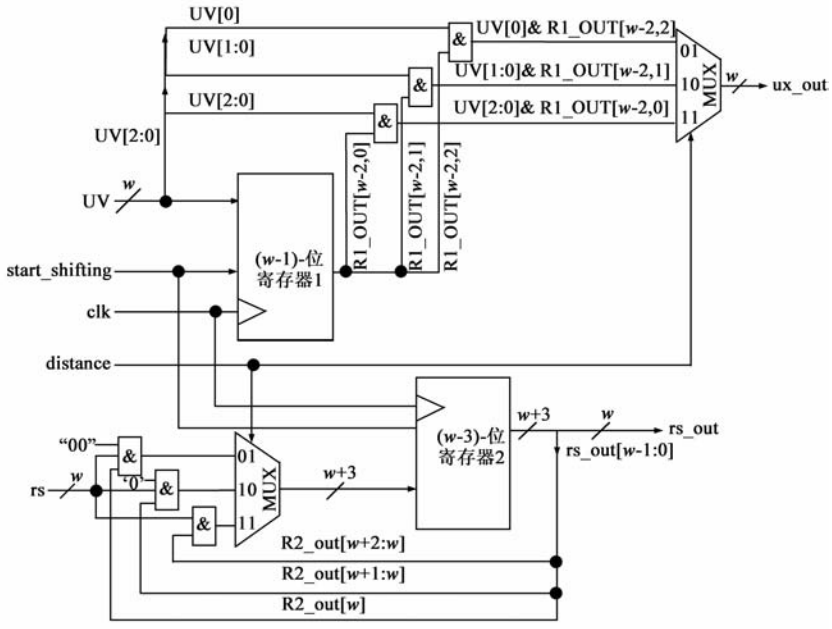

图6　可控多比特移位单元

5.2　时间复杂度分析

模逆运算总的计算时间为算法运算的时钟周期数与时钟周期的乘积,下面对这两项分别进行分析.

(1)时钟周期数分析

根据文献[7]提出的利用 Montgomery 模乘计算模逆的方法可知,在采用多比特移位方式时,Montgomery 模逆算法最少的循环迭代次数为 $2n$ 次.

根据算法 3 对多比特移位方式的分析,当最大移位比特数目为 3 时,阶段 1 外循环迭代次数降低为单比特移位的 0.88 倍,也即 $1.5n\times 0.88 = 1.32n$.阶段 2 运算消耗的时钟周期数均为 $0.5n$.算法 3 采用了基于字的处理方式,对每个 n-位操作数运算时,平均消耗$\lceil n/w\rceil$个时钟.因此,本文模逆运算消耗的总时钟数 NoC(number of clock)为:

$$NoC_{ModInv} = (1.32n + 0.5n)\cdot\lceil n/w\rceil = 1.82n\lceil n/w\rceil \quad (6)$$

(2)时钟周期分析

由图 4 和图 5 可知,模逆运算模块的关键数据路径延迟由双域运算单元的延迟决定,即模逆运算时钟周期等于双域运算单元的延迟.因此模逆运算模块的时钟周期 T_{ModInv}为:

$$T_{ModInv} = Delay_{SADD(w)} = 4(\text{标准门延迟}) \quad (7)$$

模逆运算总的计算时间由双域运算单元的处理字宽 w 与操作数长度 n 决定,因此其时间复杂度为 $O(n\lceil n/w\rceil)$.

5.3　性能比较

目前国内外针对模逆算法的硬件设计方案非常多,比较典型的有 Guo & Wang、Choudhury & Barus、Goodman & Chandrakasan 等人的设计方案.其中 Goodman 与 Chandrakasan 在文献[8]中提出了一种通用密码处理器,它采用微代码形式进行双有限域上的求逆运算,其数据路径对于比特宽度为 8bit 到 1024bit 范围内的数是可重构及参数可配置的.该方案具有一定的灵活性,但是其最大缺陷是消耗面积过大.王健等人在文献[9]中提出了将两个独立的基于扩展的 Euclidean 算法素数域与二进制域的模逆算法统一为一个双域模逆算法,但是其数据路径宽度固定为 256-位,采用向下兼容低位宽操作数的方式,即低位宽的操作数也表示为 256-位,因此该算法不具备可伸缩性.而本文提出的基于有符号数字系统的支持双有限域的快速可伸缩 Montgomery 模逆算法硬件设计方案,不仅能支持双域运算,具有可伸缩性,而且在电路面积、速度和强度之间是一种非常折衷的设计方案,如表 1 所示.

表 1　基于有符号数字系统的模逆运算结构与其它设计综合性能比较

硬件设计方案	面积复杂度	时间复杂度	支持双域运算	可伸缩性
Guo & Wang[5]	$O(n^2)$	$O(1)$	否	否
Choudhury & Barus[10]	$O(n)$	$O(n^2)$	否	否
Fenn & Benaissa[11]	$O(n^2)$	$O(n)$	否	否
Kovac & Ranganathan[12]	$O(n.n^2)$	$O(1)$	否	否
Goodman & Chandrakasan[8]	—	$O(\lceil n/w\rceil)$	是	是
王健,蒋安平[9]	—	$O(n)$	是	否
陈琳,孙万忠	$O(n)$	$O(n\lceil n/w\rceil)$	是	是

最后本文用硬件描述语言 VHDL 对各关键运算电路和该模逆算法的硬件结构进行了设计,采用 Xilinx 公司 Virtex5 系列的 EP2S180F1508C5 器件在 FPGA 开发工具 ISE9.02i 环境下编译综合和布局布线,并用 Modelsim SE6.1d 进行了功能仿真.为便于同其它模逆硬件结构

进行比较,本文选用相同的器件对固定位宽的硬件结构进行了实验.当数据长度在 128 位以内时,采用本文设计的时钟频率能够达到 248MHz,其它的模逆运算结构时钟频率为 117MHz.选择 NIST 推荐的素数进行实验,完成一次 256 比特素数域上的求逆运算需要 10.4μs,相对于文献[13]中 15.22μs 的求逆时间,速度提高接近 32%.完成二元域上的求逆运算的时间与其他设计大致相等.

6 结论

本文提出了一种新的基于有符号数字系统的可伸缩双域 Montgomery 模逆算法,设计了相应的硬件结构,并分析了其性能.文章的主要贡献包括:(1)基于 Kaliski 算法,将仅支持素数域的求模逆运算推广到二进制域上的求模逆运算,实现了双有限域上求模逆算法的统一;(2)提出基于字的双域 Montgomery 模逆算法,使算法具有伸缩性;(3)引入基 2 冗余有符号数字系统,有效避免模逆运算中超长操作数的串行进位传播问题,提高模逆运算的整体速度;(4)自主设计了基于有符号数字系统的可伸缩双域 Montgomery 模逆算法硬件方案;(5)结合仿真实验,研究了本硬件方案的工作性能以及与其它硬件方案的性能差异.本文提出的基于有符号数字系统的模逆算法与原始模逆算法相比具有可伸缩性、支持双有限域操作,并具有适合硬件实现的特点.其硬件设计与原始算法的硬件设计在运算时间、电路面积、灵活性等方面具有显著的优越性,提高了运算电路的数据吞吐率和密码处理器的性能.本文所做的工作具有重要的理论意义,并且在实际工程中具有重要的实用价值.

参考文献

[1] 侯整风,李岚.椭圆曲线密码系统(ECC)整体算法设计及优化研究[J].电子学报,2004,32(11):1904 – 1906.
Hou Zheng-feng, Li Lan. The research on designing and optimizing of the algorithm for elliptic curve cryptography (ECC) [J]. Acta Electronica Sinica, 2004, 32(11): 1904 – 1906. (in Chinese)

[2] Johann Groβschadl, Erkay Savas. Instruction set extensions for fast arithmetic infinite fields GF(p) and GF(2^m)[A]. Cryptographic Hardware and Embedded Systems-CHES2004 [C]. Cambrige, MA, USA: Computer Science, 2004. 133 – 147.

[3] Burton S Kaliski Jr. The montgomery inverse and its applications[J]. IEEE Transactions on Computers, 1995, 44(8): 1064 – 1065.

[4] Adnan Abdul-Aziz Gutub. New hardware algorithms and designs for montgomery modular inverse computation in galois fields GF(p) and GF(2^n)[D]. Oregon USA: Oregon State University, 2002.

[5] Guo J H, Wang C L. Hardware-efficient systolic architecture for inversion and division in GF(2^m)[J]. IEE Proceedings: Computers and Digital Techniques, 1998, 145(4): 272 – 278.

[6] Keshab K Parhi. VLSI 数字信号处理系统设计与实现[M]. 陈弘毅,白国强,译.北京:机械工业出版社,2004.

[7] C McIvor, M McLoone, J V McCanny. Improved Montgomery modular inverse algorithm [J]. Electronics Letters, 2004, 40 (18): 234 – 251.

[8] James Goodman, Anantha P Chandrakasan. An energy-efficient reconfigurable public-key cryptography processor [J]. IEEE Journal of Solid-State Circuits, 2001, 36(11): 1808 – 1820.

[9] 王健,蒋安平,等.同时支持两种有限域的模逆算法及其硬件实现[J].北京大学学报(自然科学版),2007,43(1):138 – 143.
Wang Jian, Jiang An-ping, et al. A dual field modular inversion algorithm and hardware implementation[J]. Acta Scientiarum Naturalium Universitatis Pekinensis, 2007, 43(1): 138 – 143. (in Chinese)

[10] Choudhury P Pal, Barua R. Cellular automata based VLSI architecture for computing multiplication and inverses in GF (2^m)[A]. Proc The IEEE 7th Conference on VLSI Design [C]. Calcutta, India: IEEE Press, 1994. 512 – 518.

[11] Sebastian T J Fenn, Mohammed Benaissa, David Taylor. GF (2^m) multiplication and division over the dual basis[J]. IEEE Transactions on Computers, 1996, 45(3): 319 – 327.

[12] Kovac, M Ranganathan, N Varanasi. A VLSI systolic array implementation of Galois Field GF(2^m) based multiplication and division algorithm[J]. IEEE Transactions on VLSI, 1993, 1(1): 22 – 30.

[13] E Savas, C K Koc. The Montgomery modular inverse-revisited [J]. IEEE Transactions on Computers, 2000, 49(7): 763 – 766.

作者简介

陈 琳 女,1975 年生于河南开封,解放军信息工程大学博士研究生,主要研究方向为安全专用芯片. E-mail: chenlin916@163.com

孙万忠 男,1978 年生于山东陵县,博士,讲师.主要研究方向为安全专用芯片.

陈性元 男,1963 年生于安徽无为,博士,教授,博士生导师,主要研究方向为密码与信息安全.

戴紫彬 男,1966 年生于河南西峡,博士,教授,博士生导师,主要研究方向为安全专用芯片.