

基于 FPGA 的 F_p 域模乘与模逆的设计与实现

杨 博, 孟李林, 陶 琼

(西安邮电大学 电子工程学院, 陕西 西安 710061)

摘 要: 在椭圆曲线体制中, 有限域的运算效率是至关重要的. 针对低端 FPGA 芯片, 从算法和硬件实现两方面优化有限域中的模乘、模逆运算. 改进后的模乘算法减少了一次 256 bit 的乘法; 在硬件实现时又节省了一次 128 bit 的乘法. 改进后的模逆算法在不增加硬件资源的情况下减少算法迭代次数; 并用减法运算和后续的修正算法代替了 256 bit 的比较器, 改进后的设计能够打断长的组合链路, 提高资源复用率, 较好地实现了面积和速度的平衡.

关键词: F_p 域; 模乘; 模逆; FPGA

中图分类号: TP309.7

文献标识码: A

文章编号: 1000-7180(2017)05-0054-05

Design of F_p Field Modular Multiplication and Modular Inversion Based on FPGA

YANG Bo, MENG Li-lin, TAO Qiong

(Electronic Engineering College, Xi'an University of Posts and Telecommunications, Xi'an 710061, China)

Abstract: In elliptic curve system, the operational efficiency of the finite field is essential. In this paper, for the low-end FPGA chip, we optimize finite field modular multiplication and modular inversion from both algorithms and hardware realization. The improved modular multiplication algorithm reduces a 256 bit multiplication and reduces a 128 bit multiplication in hardware implementation. Modular inversion algorithm improved without increasing the hardware resources available to reduce the number of iterations; with subtraction and subsequent correction algorithms instead of 256 bit comparator, improved design can interrupt a combination of long chain, improve resource complex with rates, and achieve a balance of size and speed.

Key words: F_p field; modular multiplication; modular inversion; FPGA

1 引言

椭圆曲线密码算法 (Elliptic Curve Cryptography, ECC) 最早于 1985 年由 Koblitz 和 Miller^[1-3] 提出, 其安全性依赖于椭圆曲线离散对数问题的难解性. 与 RSA 公钥密码体制相比, ECC 具有密钥更短、安全性更高、运算速度更快等优势^[4]. 我国于 2012 年公布了我国的 ECC 算法标准—《SM2 椭圆曲线公钥密码算法》. 在数字签名算法、秘钥交换协议及公钥加密算法中都不可避免地要使用模乘、模逆算法. 模乘运算的性能直接影响着点乘运算的性能, 而椭圆曲线相关运算的性能取决于点乘的运算速度. 因此, 本文对素数域 ($GF(p)$) 上模乘算法、模

逆算法进行分析研究, 提出算法改进及相应的硬件电路设计, 以提高有限域运算的效率.

2 素数域 F_p 的定义

设 p 是一个大于 2^{191} 的素数, F_p 由 $\{0, 1, 2, \dots, p-1\}$ 中 p 个元素构成, 称 F_p 为素数域, 其中整数 0 是加法运算单位元, 整数 1 是乘法运算单位元. F_p 上的元素满足下面运算规则.

(1) 加法运算: 设 $a, b \in F_p$, 则 $a+b=r$, 其中 $r=(a+b) \bmod p, r \in [0, p-1]$.

(2) 乘法运算: 设 $a, b \in F_p$, 则 $a \times b=s$, 其中 $s=(a \times b) \bmod p, s \in [0, p-1]$.

(3) 模逆运算: 设 $a \in [1, p-1]$, 求 $a^{-1} \in [1, p$

$-1]$,使得 $a \times a^{-1} \equiv 1 \pmod{p}$.

有限域上的基本运算模块包括模加/减模块、模乘模块和模逆模块. 由于有限域模加/减模块比较简单, 本文将重点介绍模乘模块和模逆模块.

3 模乘算法的分析优化及硬件设计

3.1 模乘算法比较

Barrett 和 Montgomery 方法都是用低成本的运算代替高成本的除法^[1]. 其中 Barrett 方法对模值有特殊要求, 需要高成本的预计算. Montgomery 方法虽然对模值没有要求, 但是需要对输入和输出数据进行变换, 所以这种方法不会提高单个模乘运算的效率, 但是对于包含多次模乘的运算, 这种方法是非常有效的.

Jacobian 射影坐标系下的点乘算法虽然包含大量的模乘运算, 但是这种点乘算法只需要在输入数据和输出数据时进行数据变换, 运算过程中不需要进行数据变换. 因此点乘算法的这种运算特性和模幂运算特性类似, 那么 Montgomery 方法对点乘算法的模乘将会非常有效. 算法 1 给出了 Montgomery 模乘算法^[5], 其中 x_0 和 y_0 是需要进行模乘运算的两个整数, $R = 2^{256}$, P 是模值, P' 是 P 模 R 的逆元.

算法 1 Montgomery 模乘算法

Input: 整数 X, Y, P' , 其中 $X \equiv x_0 \times R \pmod{P}, Y \equiv y_0 \times R \pmod{P}, P' \equiv (R - P)^{-1} \pmod{R}$
Output: $X \times Y \times R^{-1} \pmod{P}$
(1) $T = X \times Y$;
(2) $Q = T \times P' \pmod{R}$;
(3) $S = Q \times P$;
(4) $U = (T + S) / R$;
(5) if ($U \geq P$) $U = U - P$;
else $U = U$;

3.2 模乘算法改进

算法 1 的步骤(2)中 $T \times P'$ 是一个 768 bit 的数, 将该数对 R 取模后赋值给 Q , 则 Q 的值就等于 $T \times P'$ 运算结果的低 256 bit. 那么当算法运算到步骤(2)时, 只需执行 P' 与 T 的低 256 bit 的乘法运算, 且只取乘法运算结果的低 256 bit 赋值给 Q 即可. 步骤(4)中 $T + S$ 是一个 512 bit 的数, 将该数对 R 取商赋值给 U , 则 U 的值就等于 $T + S$ 运算结果的高 256 bit. 那么当算法运算到步骤(4)时, 只需在执行完加法运算后, 取其结果的高 256 bit 即可.

算法 2 是改进后 Montgomery 算法, 与改进前算法相比, 节省了一次 256 bit 乘法运算, 并简化了

步骤(2)和步骤(4)的运算复杂度. 改进后的算法不仅减少了运算时钟周期数, 还降低了电路设计复杂度.

算法 2 改进后的 Montgomery 模乘算法

Input: 整数 X, Y, N' , 其中 $X \equiv x_0 \times R \pmod{P}, Y \equiv y_0 \times R \pmod{P}, P' \equiv (R - P)^{-1} \pmod{R}$
Output: $X \times Y \times R^{-1} \pmod{P}$
(1) $T = X \times Y$;
(2) $Q = (T[255:0] \times P')[255:0]$;
(3) $S = Q \times P$;
(4) $U = (T + S)[511:256]$;
(5) if ($U \geq P$) $U = U - P$;
else $U = U$;

3.3 模乘模块设计

模乘模块是点乘模块的核心, 其性能决定了点乘运算的速率, 所以模乘模块的设计追求更高的运算速率. 本设计基于 Altera 公司 Cyclone IV EP4CGX75CF23C6 芯片^[6] 实现改进后的 Montgomery 模乘算法.

由于 FPGA 芯片资源有限, 直接所选用 256 bit 的乘法器 IP, 虽然会降低设计复杂度, 但是 256 bit 的乘法器 IP 会占用 FPGA 大量的总线资源, 会导致综合后电路面积不合理的增大. 若选用位宽比较小的乘法器 IP, 虽然会降低 FPGA 总线的占用率, 但是使用小位宽的乘法器构造 256 bit 位宽的乘法运算单元, 会大幅降低 256 bit 乘法运算单元的性能. 根据分析和测试发现: 选用该 FPGA 芯片中 128 bit 乘法器 IP 核设计 256 bit 乘法运算单元, 并且只使用一个 128 bit 乘法器 IP 核, 能够较好地实现面积和速度的平衡.

3.3.1 乘法器构建原理

使用 128 bit 乘法器 IP 核构造 256 bit 乘法运算单元, 理论上需要 4 次 128 bit 乘法运算和 3 次 256 bit 加法运算. 图 1 给出了使用 128 bit 乘法器 IP 核构建 256 bit 乘法运算单元的原理, 其中 A 和 B 是 256 bit 的整数, A_H 与 A_L 分别是 A 的高 128 bit 和低 128 bit, B_H 与 B_L 分别是 B 的高 128 bit 和低 128 bit. 由于只有一个 128 bit 的乘法器, 所以该运算的 4 次 128 bit 的乘法只能是串行运算. Altera 的 128 bit 乘法器 IP 的运算速度比 256 bit 加法器 IP 运算速度慢^[6], 那么乘法和加法可以并行运算.

3.3.2 模乘模块的硬件设计

算法 2 中的步骤(2)只取乘法运算结果的低 256 bit 赋值给 Q , 所以步骤(2)不需要执行第 4 次

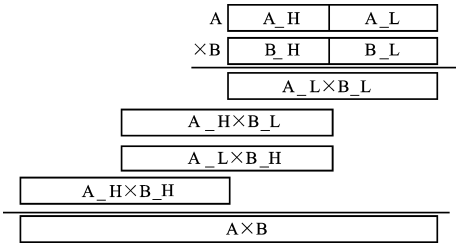


图1 乘法器构建原理

的 $A_H \times B_H$ 运算,可再节省一次 128 bit 乘法运算耗时.那么基于算法 2 实现模乘运算时,共可节省 5 次 128 bit 的乘法运算.图 2 给出了模乘模块的硬件设计,为了得到性能更优的模乘设计,256 bit 乘法运算单元采用 2 级流水线设计实现,支持乘法和加法同时运算,使 256 bit 乘法在 4 个时钟节拍内完成.图中 L_L 、 L_H 、 H_L 和 H_H 为 4 个 128 bit 的寄存器,通过这四个寄存器实现乘法运算和加法运算的数据交互.模乘控制器是一个有限状态机,通过控制数据流来调度乘法 IP 核和加法 IP 核,从而使模乘模块形成一个全局串行局部并行的硬件结构.

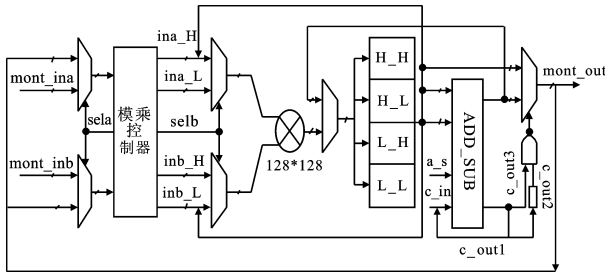


图2 模乘模块硬件设计

4 模逆算法的分析优化及硬件设计

有限域 F_{256} 上的模逆运算设计复杂度最高,运算耗时最多^[7].由于 Jacobian 射影坐标系下的点乘算法只包含两次模逆运算,则该电路的运算速度对加解密运算性能几乎没有影响,那么该电路设计时追求面积指标.

4.1 模逆算法比较与改进

4.1.1 模逆算法比较

算法 3 给出了有限域 F_{256} 上模逆的基本算法^[8],该算法对于任意的输入 $a \in F_{256}$,寻找一个元素 $a^{-1} \in F_{256}$,使得满足条件 $a \times a^{-1} \equiv 1 \pmod{p}$.

算法 3 有限域 F_{256} 上模逆运算的基本算法

Input: 输入素数 $a \in [1, p-1]$, p ;

Output: $a^{-1} \in [1, p-1]$, 使得 $a \times a^{-1} \equiv 1 \pmod{p}$;

目前有三种有效的模逆算法,分别是费马算法、

扩展的 Euclidean 算法和 Montgomery 模逆算法^[7].其中费马算法适合于二元扩域的模逆运算,不适合素数域.扩展的 Euclidean 算法通过求两个非零整数的最大公约数 gcd 来完成模逆运算,该算法把模逆运算转换成加/减法运算,占用硬件资源少. Montgomery 模逆算法和扩展的 Euclidean 算法原理相似,但需要将其运算结果从 Montgomery 域转换到实数域.基于以上分析,本文采用扩展的 Euclidean 算法.

算法 4 给出了有限域 F_{256} 上扩展 Euclidean 模逆算法^[1],该算法运算过程中没有乘法和除法运算,只有加法、减法和除以 2 运算,由于除以 2 运算在硬件上等效于右移逻辑运算,因此扩展 Euclidean 模逆算法非常适合硬件实现.

算法 4 有限域 F_{256} 上扩展 Euclidean 模逆算法

Input: 输入素数 $a \in [1, p-1]$, p ;

Output: $a^{-1} \pmod{p}$

(1) $u \leftarrow a, v \leftarrow p, x_1 \leftarrow 1, x_2 \leftarrow 0$;

(2) 当 $u \neq 1$ 和 $v \neq 1$ 时,重复执行;

(2.1) 当 u 是偶数时,重复执行;

$u \leftarrow u/2$;

若 x_1 是偶数,则 $x_1 \leftarrow x_1/2$;

否则 $x_1 \leftarrow (x_1 + p)/2$.

(2.2) 当 v 是偶数时,重复执行;

$v \leftarrow v/2$;

若 x_2 是偶数,则 $x_2 \leftarrow x_2/2$;

否则 $x_2 \leftarrow (x_2 + p)/2$.

(2.3) 若 $u \geq v$,则 $u \leftarrow u - v, x_1 \leftarrow x_1 - x_2$;

否则 $v \leftarrow v - u, x_2 \leftarrow x_2 - x_1$.

(3) 若 $u = 1$,则返回 $a^{-1} \pmod{p} = x_1 \pmod{p}$;

否则返回 $a^{-1} \pmod{p} = x_2 \pmod{p}$.

4.1.2 模逆算法改进

扩展 Euclidean 模逆算法存在两点不足,第一点是该算法步骤(2.1)和步骤(2.2)中分别包含 u 和 v 的右移运算,能够减少算法迭代次数,但步骤(2.3)只包含减法运算,不一定能减少算法迭代次数,从而增加算法操作步数.由于步骤(2.3)中 u 和 v 都不是偶数,所以 u 和 v 之间减法运算的结果肯定是偶数,那么下一步必将执行步骤(2.1)和步骤(2.2)中的一步,即右移运算.则对步骤(2.3)的运算结果直接进行右移操作,使其在步骤(2.3)中完成,便能在不增加硬件资源的情况下减少算法迭代次数.

第二点是步骤(2.3)中判断 u 和 v 的大小,由于 u 和 v 是一个数据位宽为 256 bit 的数据,所以这个

比较运算会被综合工具综合成一个 256 bit 的比较器. 在运算过程中, 这个比较器的运算结果出来后, 才会在做后面的减法运算. 这样不仅会降低电路工作频率, 还会增大硬件资源的消耗. 本文设计在运算步骤(2.3)时, 不做判断操作, 直接做减法运算, 根据减法器的借位信息修正运算结果. 算法 5 给出了有限域 F_{256} 上改进的扩展 Euclidean 模逆算法.

算法 5 有限域 F_{256} 上改进的扩展 Euclidean 模逆算法

Input: 输入素数 $a \in [1, p-1], p$;

Output: $a^{-1} \bmod p$

(1) $u \leftarrow a, v \leftarrow p, x_1 \leftarrow 1, x_2 \leftarrow 0$;

(2) 当 $u \neq 1$ 和 $v \neq 1$ 时, 重复执行,

(2.1) 当 u 是偶数时, 重复执行,

$u \leftarrow u/2$;

若 x_1 是偶数, 则 $x_1 \leftarrow x_1/2$;

否则 $x_1 \leftarrow (x_1 + p)/2$.

(2.2) 当 v 是偶数时, 重复执行,

$v \leftarrow v/2$;

若 x_2 是偶数, 则 $x_2 \leftarrow x_2/2$;

否则 $x_2 \leftarrow (x_2 + p)/2$.

(2.3) $u \leftarrow u - v, x_1 \leftarrow x_1 - x_2$

(2.3.1) 若 $u - v$ 有借位,

$v \leftarrow (\sim u + 1)/2$

若 $x_1 - x_2$ 有借位,

若 x_1 为偶数, 则 $x_2 = (\sim x_1 + 1)/2$

若 x_1 为奇数, 则

$x_2 = (\sim x_1 + 1 + p)/2$

若 $x_1 - x_2$ 无借位,

若 x_1 为偶数, 则 $x_2 = (p - x_1)/2$

若 x_1 为奇数, 则 $x_2 = (p - x_1 + p)/2$

(2.3.2) 若 $u - v$ 无借位,

$u \leftarrow u/2$

若 $x_1 - x_2$ 有借位,

若 x_1 为偶数, 则 $x_1 = (x_1 + p)/2$

若 x_1 为奇数, 则 $x_1 = (x_1 + p + p)/2$

若 $x_1 - x_2$ 无借位,

若 x_1 为偶数, 则 $x_1 = x_1/2$

若 x_1 为奇数, 则 $x_1 = (x_1 + p)/2$

(3) 若 $u = 1$, 则返回 $a^{-1} \bmod p = x_1 \bmod p$;

否则返回 $a^{-1} \bmod p = x_2 \bmod p$.

改进后的算法在步骤(2.3)上, 用减法运算和后续的修正算法代替了 256 bit 的比较器, 虽然将一步运算拆成两步来完成, 但是这样的设计能够打断长的组合链路, 提高资源复用率. 另外改进后的算法使每一步都包含右移操作, 能够减少算法的迭代次数. 虽然改进后算法在条件判断上比原算法复杂一些, 但是增加的条件判断操作对硬件资源的消耗和电路

的工作频率影响不大.

4.2 模逆模块设计

深入研究改进后的模逆算法, 可以得出 u, v 之间的减法操作和 x_1, x_2 之间的减法操作能够同时进行, 为了提高算法运算效率, 节省硬件资源的消耗, 本文选用两个 128bit 的加/减法 IP 实现这两组减法的并行运算. 图 3 给出了有限域 F_{256} 上模逆模块的硬件设计, 设计中只包含加/减法和移位寄存器两种运算单元. 通过复用 ADD_SUB1_128 加/减法器代替 256 bit 比较器功能, 通过 u, v, x_1 和 x_2 四个移位寄存器实现步骤(2.1)、步骤(2.2)和步骤(2.3) 每一步都包含右移操作. 图中模逆控制电路是一个有限状态机, 通过控制数据流来调度 ADD_SUB1_128 和 ADD_SUB2_128 这两个加减法器.

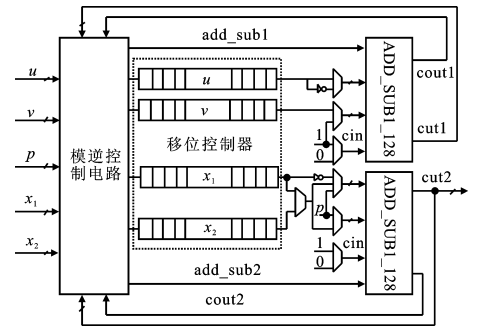


图 3 模逆模块硬件设计

5 模乘模块与模逆模块功能仿真

5.1 模乘模块功能仿真

图 4 给出了模乘模块的功能仿真, 仿真输入数据 $\text{ina} = 256'h421d_ebd6_1b62_eab6_7464_34eb_c3cc_315e_3222_0b3b_add5_0bdc_4c4e_6c14_7fed_d43d$; $\text{inb} = 256'h0680_512b_cbb4_2c07_d473_49d2_153b_70c4_e5d7_fdcf_bfa3_6ea1_a858_41b9_e46e_09a2$. 输入以上数据进行模乘运算, 期望 mont_out 端在 mont_start 信号有效后的第 27 和 28 个时钟节拍分别输出: $128'hddd1_566c_8156_6e1b_59f6_450a_f0bc_3daa$ 和 $128'h1f7c_901c_a824_d7b1_1ca4_fcfa_c64d_91d6$. 从仿真结果能够得出, 期望结果与仿真结果一致, 说明模乘模块功能正确, 性能满足设计要求.

5.2 模逆模块功能仿真

图 5 给出了模逆模块的仿真, 仿真输入数据 $\text{invert_in} = 256'h421d_ebd6_1b62_eab6_7464_34eb_c3cc_315e_3222_0b3b_add5_0bdc_4c4e_6c14_7fed_d43d$. 输入 invert_in 进行模逆运算, 期望 invert_out

端分别在 invert_Done 信号有效后的第 1 和第 2 个时钟节拍输出:128'h2593_afbf_571c_dae7_baf8_a353_3cd6_677b 和 128'h4447_f05e_3e21_7156_8b14_1bc3_9e55_2a1e. 从仿真结果能够得出,期望结果与仿真结果一致,说明模逆块功能正确.

6 结束语

在椭圆曲线体制中,有限域的模加减、模乘及模逆是必不可少的. 由于有限域模加/减模块比较简单,本文将重点介绍了模乘模块和模逆模块. 本文从

算法和硬件实现两方面对模乘和模逆算法进行了优化. 改进后的模乘算法减少了一次 256 bit 的乘法;在硬件实现时又节省了一次 128 bit 的乘法. 改进后的模逆算法在不增加硬件资源的情况下减少算法迭代次数;并用减法运算和后续的修正算法代替了 256 bit 的比较器,虽然将一步运算拆成两步来完成,但是这样的设计能够打断长的组合链路,提高资源复用率. 本文在模乘、模逆算法方面的改进适用于所有硬件电路.



图 4 模乘模块的功能仿真

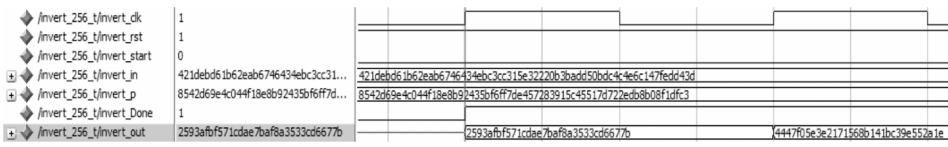


图 5 模逆模块的功能仿真

参考文献:

[1] Drrel Hankerson, Alfred Menezes. 椭圆曲线密码学导论[M]. 张焕国,译. 北京:电子工业出版社,2005: 22-235.

[2] 丁勇. 椭圆曲线密码快速算法理论[M]. 北京:人民邮电出版社,2012:1-60.

[3] 李海峰,马海云,徐燕文. 现代密码学原理及应用[M]. 北京:国防工业出版社,2013:104-132.

[4] 王立川. 椭圆曲线上标量乘快速算法研究[D]. 西安:西安电子科技大学,2011.

[5] 周晚,张靖,彭亚雄. Montgomery 形式椭圆曲线的标量乘法运算[J]. 硅谷,2010,9(15): 124-132.

[6] Altera Corporation. Cyclone IV device handbook [EB/OL]. [2015-02-30]. <https://www.altera.com/con->

[tent/dam/altera-www/global/en_US/pdfs/literature/hb/cyclone-iv/cyclone4-handbook.pdf](tent/dam/altera-www/global/en_US/pdfs/literature/hb/cyclone-iv/cyclone4-handbook.pdf).

[7] 杨同杰,戴紫杉,杨晓辉. 一种双域 Montgomery 求逆算法与硬件实现[J]. 计算机工程与应用,2010,46(13): 63-75.

[8] 胡皓. 基于椭圆曲线密码体系的 RFID 加密研究[D]. 北京:北京交通大学,2014.

作者简介:

杨 博 男,(1991-),硕士研究生. 研究方向为集成电路系统. E-mail:120188228@qq.com.

孟李林 男,(1962-),教授,硕士生导师. 研究方向为集成电路设计.