一种基于 FPGA 的 ECC 点乘优化设计

王国强[1] 张明明[1] 任凯琦[1] 赵博文[1] 陈宗权[1] 张 丽[2]
WANG Guoqiang ZHANG Mingming REN Kaiqi ZHAO Bowen CHEN Zongquan ZHANG Li

摘 要 现场可编程门阵列（field programmable gate array，FPGA）具有丰富的计算、存储等资源，广泛应用于密码算法、实时通信等高并行、高数据吞吐量、计算复杂的领域。现有的基于 FPGA 实现 ECC 点乘的方案，存在计算架构设计并行程度低，计算所需的逻辑资源过大的问题。针对以上问题，提出了一种基于 FPGA 实现 ECC 点乘的优化设计。通过采用 RAM 寻址方式设计出计算架构，并构造了一个流水线状态机，利用 FPGA 并行计算的优势，多次并行使能调用点加、模加、模乘、模逆算法模块，实现了高效的 ECC 点乘操作。实验结果表明，所提出的优化设计有效地缩减了 FPGA 的 LUT 逻辑资源，并提高了 ECC 点乘运算的计算速度。

关键词 FPGA；ECC；点乘；RAM 寻址；仿真

doi：10.3969/j.issn.1672-9528.2024.02.008

0 引言

椭圆曲线密码算法（ECC）属于公钥加密体制，存在成对的公钥和私钥，其中私钥用于消息数据的解密，公钥用于消息数据的加密。最早于 20 世纪 80 年代，Koblitz 和 Miller 两人先后利用椭圆曲线上的有理点构成的 Abel 群 [1] 构造了椭圆曲线密码方案，其安全性依赖于椭圆曲线上离散对数问题的计算困难性。相比于熟知的基于有限域上离散对数问题的公钥加密算法，椭圆曲线加密算法可利用更短的密钥量提供更强的安全性，而且产生算法的椭圆曲线构造灵活 [2-3]，具有丰富的群结构和多选择性，因此基于椭圆曲线离散对数问题的椭圆曲线密码算法在密码应用领域中有着更广阔的前景。从密码方案实际应用角度来看，整个方案运算步骤中的点乘运算是最核心，也是最耗时的运算步骤，其效率直接决定着加解密操作或者签名验签操作的运算速度 [4]，决定着新的方案是否能够大范围推广和使用。

近年来，随着超算能力的不断提升，传统的、利用资源有限的 CPU 实现支持短位数的椭圆曲线密码方案，被成功破解的概率越来越高，信息的安全性方面受到了很大的威胁。而为了保障信息的安全，支持更长位数的椭圆曲线密码方案势必导致加解密运算速度的减慢，无法满足当今网络实时性要求高，数据吞吐量大的需求。现场可编程门阵列（field programmable gate array，FPGA）具有可并发和流水处理多任务的优势 [5-6]，通过高并行的计算架构设计能够获得比 CPU 高数倍的计算速度。许多研究人员围绕 FPGA 的优势进行了大量的研究，设计并实现了许多安全性高、运算速度快、低功耗的密码算法 [7-11]。然而，现有的基于 FPGA 的椭圆曲线密码方案在实现过程中采取串行的运算方式需要很长的时钟周期，导致完成一次 ECC 点乘需要大量时间。对于逻辑资源较小但对算法安全性要求高的应用场景，这种设计实现方式无法进行很好的适配使用。因此，如何在 FPGA 中实现高效的 ECC 加密算法且消耗更低的逻辑资源成为一个亟待解决的问题。

桂金瑶等人提出了一个基于 FPGA 的多项式基下二进制域 ECC 点乘设计方案，利用基于经典蒙哥马利点乘改进算法，在射影坐标系下设计实现模加、模乘、模逆的模块，并通过一系列的状态机调用各个模块组合，完成了点乘运算的操作。整个系统结构进行了优化处理，最终在 Cyclone 系列的 EP2C35F484C5 上，利用 Quartus Ⅱ平台分析得出时钟频率为 50.3 MHz，逻辑单元个数为 25 044 个。2016 年，陈俊杰等人提出了一种基于改进 NAF 的点乘并行调度算法 , 在深入分析 Jacobian 射影坐标系下点加算法和倍点算法的基础上，分别设计了点加并行运算算法和倍点并行运算算法。基于 Cyclone IV 系列的 FPGA 开发平台实现了改进后高效的 ECC 算法的硬件设计。硬件测试结果表明，完成一次点乘运算需要 111 860 个时钟周期。与改进前算法相比，运算速度提高了 40.3%。

针对现有的基于 FPGA 实现 ECC 点乘的方案，存在计算架构设计并行程度低，计算所需的逻辑资源过大的问题。本文通过采用 RAM 寻址计算和流水线技术对 ECC 算法的

1. 北京计算机技术及应用研究所 北京 100059
2. 63921 部队 北京 100028

点乘运算操作做出优化，并通过仿真测试结果表明，本文提出的一种基于 FPGA 的 ECC 算法点乘优化方案有效减少了 FPGA 逻辑资源，同时提高了运算速度。

本文的章节安排如下。第一章节简单介绍了 ECC 密码体制涉及关键理论；第二章节介绍了 ECC 点乘模块的构造；第三章节介绍了如何在 FPGA 上实现 ECC 点乘运算；第四章节将本文方案的实验数据与同类型实验数据进行了对比；第五章节根据仿真实验的对比结果，给出了最终结论。

1 ECC 相关理论

1.1 有限域 $GF(2^m)$

有限域 $GF(2^m)$ 只包含 2^m 个元素，这里 2 是 2^m 的特征，而 m 是 $GF(2^m)$ 在它的素域 GF_2 上的次数。构造元素个数为素数方幂的有限域的方法有很多，其中最直观的方法是利用多项式的加、乘、除和剩余来构造有限域 $GF(2^m)$。

令 $f(x)=x^m+f_{m-1}x^{m-1}+\ldots+f_2x^2+f_1x+f_0\ (f_i\in F_2, 0\le i\le m-1)$ 是 F_2 上次数为 m 的不可约多项式，即 $f(x)$ 不能分解为 F_2 上两个次数小于 m 的多项式的积。有限域 $GF(2^m)$ 由 F_2 上所有次数小于 m 的多项式组成，即

$$GF(2^m)=a_{m-1}x^{m-1}+a_{m-2}x^{m-2}+\ldots+a_1x+a_0$$

$$(a_i\in\{0,1\})$$

域元素（$a_{m-1}x^{m-1}+a_{m-2}x^{m-2}+\ldots+a_1x+\mathrm{a}_0$）通常用长度为 m 的二进制串$a_{m-1}a_{m-2}\ldots a_1a_0$表示，使得

$GF(2^m)=\{(a_{m-1}a_{m-2}\ldots a_1a_0): a_i\in\{0,1\}\}$。

因此，$GF(2^m)$ 中的元素可以用所有长度为 m 的二进制串的集合来表示。

1.2 $GF(2^m)$ 上的椭圆曲线

$GF(2^m)$ 上由参数a，$b\in F_{2^m}$，$b\neq 0$定义的一个非超奇异椭圆曲线 $E(F_{2^m})$ 是方程 $y^2+xy=x^3+ax^2+b$ 的解集合 (x,y)，$x,y\in F_{2^m}$，连同一个称为无穷远点（记为 0）的元素组成的点集合。其中两条重要的加法规则如下。

（1）若 P 和 Q 是满足 $x_1\neq x_2$ 的点，则直线 $L=PQ$ 与椭圆曲线有且仅有第三个交点 R，定义 $P+Q=-R$。令 $-R=(x_3,y_3)$，则有：

$$\begin{cases}x_3=\gamma^2+\gamma+x_1+x_2+a\\ y_3=\gamma(x_1+x_3)+x_3+y_1\end{cases}\quad(1)$$

式中：$\gamma=\frac{y_1+y_2}{x_1+x_2}$。

（2）若 $P=Q$，$x_1\neq 0$，令 L 为椭圆曲线上点 P 的切线，L 与椭圆曲线有且仅有第二个交点 R，则 $P+Q=2P=-R$。令 $-R=(x_3,y_3)$，则：

$$\begin{cases}x_3=\gamma^2+\gamma+a\\ y_3={x_1}^2+(\gamma+1)x_3\end{cases}\quad(2)$$

式中：$\gamma=x_1+\frac{y_1}{x_1}$。

1.3 椭圆曲线密码系统

椭圆曲线密码系统的安全性是基于椭圆曲线离散对数问题的难解性，给定有限域 $GF(2^m)$ 和定义在有限域上的椭圆曲线 $E(F_{2^m})$，对于已知椭圆上的点 P，求 $Q=kP$（kP 称为椭圆曲线上的点乘，即 k 个 P 在椭圆曲线上的点乘运算）很容易，但是反过来在已知 P 和 Q 的情况下求 k 却非常困难。

椭圆曲线密码方案一般包含四个步骤，描述如下。

（1）初始化：选定协商好一条椭圆加解密曲线 $y^2+xy=x^3+ax^2+b$，此椭圆加密曲线定义为 $E_p(a,b)$。ECC 对应的加密椭圆曲线的参数 p 为位宽为 Nbit 的素数，一般选取时，p 值越大，实现的算法加解密安全性越高。

（2）密钥产生：选定一对在椭圆曲线上的 (x,y) 坐标值作为基点 G_{xy}，选择生成数值 K 作为私钥，私钥 K 和基点 G_{xy} 通过公式 $G=KG_{xy}$ 得到公钥 G。

（3）加密过程：取随机数 r，要加密的明文 M 和公钥 G 下发给 FPGA 进行加密计算，先将输入的明文 M 编码到椭圆曲线上，通过公式 $C_1=M+rG$，$C_2=rG_{xy}$ 计算得到加密后的密文 (C_1,C_2)，加密流程结束。

（4）解密过程：已知私钥 K，通过公式 $C_1-KC_2=M+rG-rKG_{xy}=M$ 计算得到明文 M，解密流程结束。

2 ECC 点乘模块的构造

2.1 椭圆曲线密码点乘结构图

在 FPGA 中实现的点乘由三层结构组成，如图 1 所示，点乘运算调用点加和倍点模块、点加和倍点运算调用模加、模乘和模逆模块。

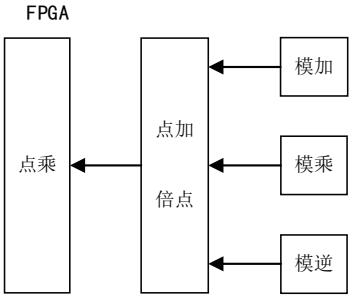

图 1 FPGA 实现点乘结构图

2.2 模加模块

有限域 $GF(2^m)$ 中的加法运算就是将表示元素的二进制序列进行按位异或运算，因而可以使用简单的组合逻辑来实现，利用 verilog 语句 $a=b\hat{}c$ 来完成。

2.3 模乘模块

模乘模块是实现 ECC 点乘的基础，在点乘、点加、模逆模块中会频繁调用，因此，模乘运算的速率决定了 ECC 点乘运算的效率。在 FPGA 中一般实现模乘模块使用串行方式。

模乘算法公式：$C(x)=A(x)B(x) \bmod F(x)$。

第一步：若 $A_0=1$，则 $C=B$。

第二步：对 i 从 1 到 n-1，$B=(B<<1 \bmod F(x))$，若 $A_i=1$，则 $C=Cx$ or B。

第三步：循环结束，返回 C。

本文将模乘模块优化为串并结合的实现方式，综合考虑资源使用和时序满足问题后，将每次循环移位位数修改为 4，在一个时钟周期内完成 4 次移位计算操作。若 $A(x)$ 的位宽为 n，则正常一次模乘运算所需时间为 n 个时钟周期，而优化后一次模乘运算所需的时间为 $n/4$ 个时钟周期，以牺牲逻辑资源的代价换取了更快的计算速度。

2.4 模逆模块

模逆算法由费尔马定理得出，模逆模块实现如图 2 所示，若 $A(x)$ 的位宽为 n，则一次模逆计算需要按串行方式进行 n-1 次模平方和 n-1 次模乘。完成一次模平方所需时间一个时钟周期，完成一次模乘所需时间 n 个周期。优化后的模逆算法将串行方式改为串并结合的方式，在进行模乘运算期间，并行完成 n 次模平方操作，并将得到的 n 个结果存储在 blockRAM 中，在下次模乘运算时直接进行调用，缩减了完成一次模乘运算所需的时间。

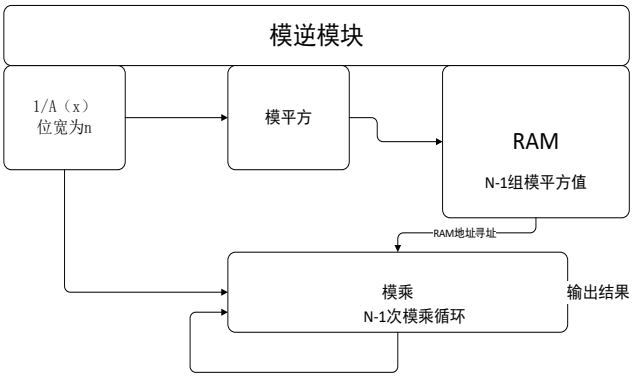

图 2 模逆模块结构图

2.5 点加和倍点模块

点加和倍点运算模块，在 FPGA 中一般用状态机控制流水线串行实现。如图 3 所示，对点加模块做了并行优化。ECC 椭圆曲线的点加法规则为 $X_3=\lambda^2+\lambda+X_1+X_2+a$，$Y_3=\lambda(X_1+X_3)+X_3+Y_1$，其中 $\lambda=(Y_1+Y_2)/(X_1+X_2)$。FPGA 实现该运算时将规则的顺序计算优化为并行计算，在计算模逆 $1/(X_1+X_2)$ 期间，同时计算模加 (Y_1+Y_2)，为将要进行的模乘 $\lambda=(Y_1+Y_2)/(X_1+X_2)$ 做好准备，减少计算延时。在计算模乘 $\lambda=(Y_1+Y_2)/(X_1+X_2)$ 期间，同时计算模加 X_1+X_2+a，为将要进行的模加 $X_3=\lambda^2+\lambda+X_1+X_2+a$ 做好准备，减少计算延时。Y_3 的计算同样做此操作。

对倍点模块做了并行优化类似在点加中的操作。ECC 椭圆曲线的倍点法规则为 $X_3=\lambda^2+\lambda+a$，$Y_3=X_1^2+(1+\lambda)X_3$，其中 $\lambda=(Y_1/X_1)+X_1$。在计算模乘 λ^2 期间，同时计算模加 $\lambda+a$，为将要进行的模乘 $X_3=\lambda^2+\lambda+a$ 做好准备，减少计算延时。在计算模乘 $\lambda=X_1^2$ 期间，同时计算模乘 $(1+\lambda)X_3$，为将要进行的模加 $Y_3=X_1^2+(1+\lambda)X_3$ 做好准备，减少计算延时。

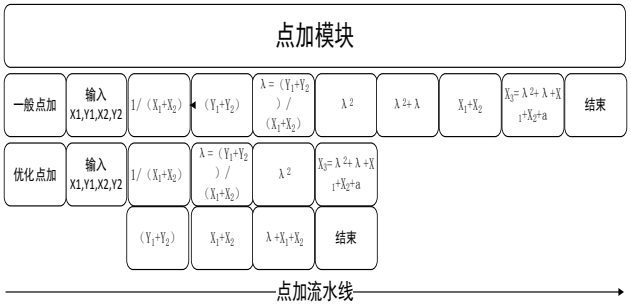

图 3 点加模块优化图

2.6 点乘模块

实现 ECC 算法的加解密实际上就是通过不断地调用点乘来实现椭圆曲线密码体制中的密钥对生成、加密、解密操作。点乘是对于给定椭圆曲线上的点 P 和正整数 k，求 k 个 P 相加的结果，记作 $Q=kP$。椭圆曲线上任意两点相加的结果还是椭圆曲线上的点，所以点乘的结果也是椭圆曲线上的点。

本文实现点乘的方法是重复进行倍点和点加运算的操作，算法流程如下。

输入：一个正整数 n 和一个椭圆曲线上的点 P。

输出：一个椭圆曲线上的点 nP。

（1）令 $n=n_rn_{r-1}...n_1n_0$ 是 n 的二进制表示，其中若 n 的最高位比特 n_r 为 1。

（2）则 $Q \leftarrow P$。

（3）对 i 从 r-1 循环到 0 执行：

① $Q=2Q$。

② 若 $n_i=1$，则 $Q=Q+P$。

（4）输出 Q。

3 点乘模块在 FPGA 上的实现

3.1 点乘模块在 FPGA 上的实现

本文 ECC 算法点乘的开发环境为 ISE14.7，逻辑完全为逻辑代码实现，用到的 FPGA 芯片为 xilinx 官方的 XCSX5V95T，有充足的逻辑资源和 RAM 存储资源。最终使用的 ECC 算法模块主频时钟是 50 MHz，有较大的时钟裕量。

在 ECC 点乘算法模块和各个子模块中普遍使用了状态

机控制流水线串并结合方式，优化了完成一次 ECC 算法点乘所需要的时间。通过利用 RAM 寻址的计算方式有效减少了逻辑资源，最终实现的 ECC 算法模块消耗的 LUT 逻辑资源为 9003 个，RAM 资源为 396 kB，完成点乘运算所需的存储资源，缩减了 FPGA 的 LUT 逻辑资源，提高了计算速度。此 FPGA 实现的 ECC 点乘算法可以灵活地被多种主控调用来实现 ECC 的加解密，有很强的移植性。时间周期为 4006 μs。消耗资源相关参数如表 1 所示。

表 1 点乘运算消耗资源相关参数表

编译仿真开发环境为 ISE14.7、modelsim，FPGA 芯片 XCSX5V95T		
资源类型	资源用量	单位
LUT	9003	个
RAM	396	kB
一次点乘时间	4006	μs

3.2 椭圆曲线密码体制实现

FPGA 实现点乘模块后对外接口可灵活改变，匹配不同的主控，通过被调用点乘的方式来完成一次完整的 ECC 算法加解密操作。如图 4 所示，给出一种调用 FPGA 点乘实现 ECC 算法加解密流程图。

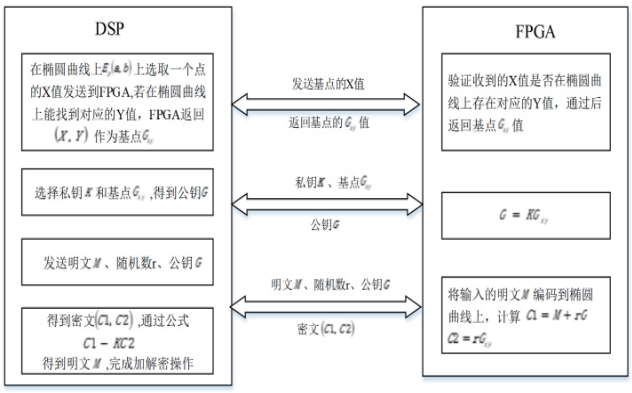

图 4 调用 FPGA 点乘实现 ECC 算法加解密流程图

4 实验结果

ECC 点乘算法优化前所占用的逻辑资源 LUT 数为 41 454，通过采用 RAM 寻址计算和流水线技术后占用逻辑资源 LUT 数为 9003，有效地减少了逻辑资源的使用。

5 结论

本文提出了一种基于 FPGA 的 ECC 算法的点乘模块优化设计。该设计通过一个流水线状态机，根据不同状态下多次并行使能调用点加、模加、模乘、模逆算法模块完成一个完整的点乘操作，同时对点乘操作过程中涉及的点加、倍点、模乘和模逆算法模块的实现均作出了优化，最终仿真实验结果证明，本文的设计不仅有效降低了 FPGA 的逻辑资源，还提高了 ECC 算法点乘模块的计算速度。

参考文献：

[1] KOBLITZ N. Elliptic curve cryptosystems[J]. Mathematics of computation, 1987,148:203-209.

[2] 杨波 . 现代密码学 [M]. 北京 : 清华大学出版社 , 2015.

[3] 全国信息安全标准化技术委员会 . 信息安全技术 SM2 椭圆曲线公钥密码算法 :GB/T 32918.1-2016[S]. 北京 : 中国标准出版社 ,2016.

[4] 秦志光 . 密码算法的现状和发展研究 [J]. 计算机应用 , 2004, 24(2):1-3.

[5] 李鑫 , 闫雪梅 , 高媛媛 , 等 . FPGA 发展现状和行业应用分析 [J]. 信息通信技术与政策 ,2022,48(7) :65-72.

[6] 侯伯亨 , 顾新 . VHDL 硬件描述语言与数字逻辑电路设计 [M]. 西安 : 西安电子科技大学出版社 , 2001.

[7] 任晓东 , 文博 . CPLD/FPGA 高级应用开发指南 [M]. 北京 : 电子工业出版社 , 2003.

[8] 桂金瑶 , 陈大军 . 基于 FPGA 的多项式基下二进制域 ECC 点乘设计 [J]. 微型电脑应用 ,2010,26(1):59-61+6..

[9] 陈俊杰 , 孟李林 , 袁阳 . 基于 FPGA 的 ECC 快速算法研究及设计 [J]. 微电子学与计算机 , 2016,33(8):139-143+148.

[10] 黄威 . 椭圆曲线密码（ECC）算法的 FPGA 实现及优化设计 [D]. 武汉 : 武汉理工大学 ,2006.

[11] 程鑫 . 基于 FPGA 的椭圆曲线点乘运算器设计研究 [D]. 西安 : 西安理工大学 ,2012.

【作者简介】

王国强（1996—），男，山西太原人，硕士研究生，工程师，研究方向：软件研发。

张明明（1986—），男，山东聊城人，硕士研究生，工程师，研究方向：信息安全。

任凯琦（1991—），男，内蒙古鄂尔多斯人，硕士研究生，高工，研究方向：信息安全。

赵博文（1994—），男，河北保定人，硕士研究生，工程师，研究方向：信息安全。

陈宗权（1973—），男，重庆铜梁人，博士研究生，工程师，研究方向：计算机科学与技术。

张丽（1987—），女，河北曲周人，硕士研究生，专业技术研究实习员，研究方向：计算机类、通信类。

（收稿日期：2023-10-09）