椭圆曲线密码算法的硬件设计与实现

Elliptic Curve Cryptography Research Base on Hardware

周轶男　李 曦　(中国科学技术大学计算机科学技术系 合肥 230047)
朱兆国　(江南计算技术研究所 江苏无锡 214083)

摘要:本文重点介绍了椭圆曲线密码体制(ECC)的实施过程以及用到的相关算法和优化方案。文中利用大数模运算硬件实现中常用的蒙哥马利(Montgomery)算法和心缩(Systolic)算法实现了模乘运算,并在此基础上用 Verilog 硬件描述语言实现了 ECC 最核心的点乘运算。

关键词:椭圆曲线密码体制　蒙哥马利算法　心缩算法　点乘

1　引言

椭圆曲线密码(ECC)是公钥密码中的后起之秀,经过二十年的发展,已经成为一类正在被全球推广使用的公钥密码。它是基于椭圆曲线离散对数问题(ECDLP)的一类密码,该密码最大的特点是,在已知的所有公钥密码中具有最高的单比特安全性。这一特性使它能够在相同的安全强度下,具有最小的密钥长度,因而它的实现速度非常快,生成密钥对的负担较轻,它的这些特点能够使它较方便地应用于一些资源和环境受到较大限制的场合,如无线设备,移动电话,智能卡等。除此之外,椭圆曲线密码所具有的参数共享特性也能使它在未来的一些安全技术和安全解决方案中得到广泛应用。由于具有这些优点,椭圆曲线密码越来越受到人们的关注。

2　ECC 的硬件设计

通过研究发现,160bits 的 ECC 与 1024bitsRSA 具有相同的安全性,在当前条件下,只需选取 192bits 以上的密钥量便可以达到安全要求,这也是 ECC 与 RSA 相比有优势的地方。因此本文研究对象确定为素域 Fp (p 为 192bits 的素数)上的 ECC。从理论分析及工程实现两方面需要解决如下两个问题:

(1) 要保证从 P,Q 求 k 的困难性,必需要求 E(Fq) 上不存在小阶数的点,避免 k 落入较窄的范围内,所以在 EC 的生成中要求# E(Fq)为大素数,即安全的 EC 的生成问题。

(2) 求 Q=k * P 的容易性要体现在实用算法上,将 k 做二进制分解,通过连续倍点和点加运算可以有效完成 k * P 的运算。所以提出 EC 的倍点和点加运算的有效实现问题。

其中模乘运算是最主要最基本的运算。所谓模乘运算,就是计算 $a*b \pmod m$,在公钥体制中,a,b,m 一般为几百 bits 的大整数,其中 m 一般为素数。在过去的公钥体制中,模乘运算也一直是其中最关键,也是最耗时的运算,它的实现速度的快慢直接影响到整个密码体制的实现速度。因而关于它的运算一直是密码界研究的热点,也产生了很多有效的算法,其中最著名、应用最广泛的是蒙哥马利(Montgomery)算法和心缩(Systolic)算法。

众所周知,在计算机中,除法指令的执行非常耗时,而移位运算的速度则非常快,Montgomery 算法的思想是将模乘运算中的取模运算(即除法运算)转化为移位运算,很显然这能极大的提高运算速度。但是在硬件实现大数的模乘运算时,蒙哥马利算法不能直接用,因为直接计算大整数的乘法延迟很大,需要将大整数分开来计算,于是便产生了模乘运算硬件实现的心缩(Systolic)算法,具体算法描述如下:

算法 1: 输入:$A=\sum_{i=0}^{n-1}A[i]r^i$,r 为 2 的方幂,$M=\sum_{i=0}^{n-1}M[i]r^i$,整数 B(B<2M)

输出:$A*B*r^{-n} \pmod M$

流程: 1 令 P=0;

2 for i=0 to n-1 do

begin Q[i]=((P[0]+A[i] * B[0]) * (r-M

[0])$^{-1}$) mod r

P = (P + A[i] * B + Q[i] * M)/r

End

3 若 P≥M,输出 P − M,否则输出 P

算法 1 的描述非常适合硬件实现,根据它可以实现图 1 中的硬件结构,这个结构称为心缩组(systolic array)。图中每个小框中计算 P_i 的部分值,由图中可以看出,这个算法将大数的运算分割为小整数的运算,每个小模块内需要完成 r bits 数的运算,运算结果自左到右、自上到下进行传递,经过 3n +1 拍后得到最终的结果,实现该算法时,r 的大小根据情况选择,r 越大,得到结果所用的节拍数会减少,但主频会降低,反之,r 越小,得到结果所用节拍数会增加,但主频会提高。

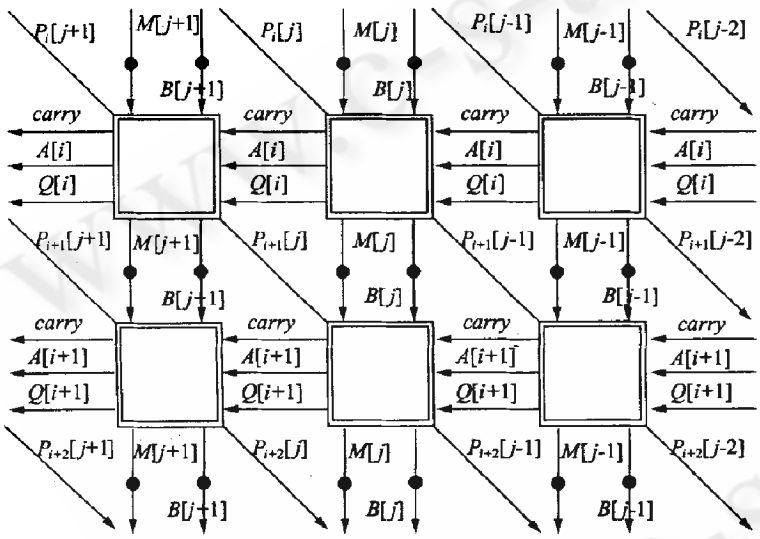

图 1　心缩算法硬件实现框图

由于模乘运算是 ECC 硬件实现中最基本最重要的运算,有效合理的实现算法 1 对提高整个系统的速度有重要的意义。因此还可以对算法 1 做一改进,令 $A=\sum_{i=0}^{n}A[i]r^i$, $A[n]=0$;这样可以使算法 1 第 3 步中的比较省去,虽然这样做使得到结果的延时增加到 3n +4,但由于在硬件实现中比较器是一个很耗时的运算部件,最终会使运算速度得到提高。

另外,本文中所选取的椭圆曲线为素域 F_p (p > 3) 上的椭圆曲线: $y^2=x^3+ax+b$ ($a,b\in F_p 4a^3+27b^2\neq 0$ (modp))。$P(x_1,y_1)$, $Q(x_2,y_2)$ 为曲线上任意两点,由于要进行点加运算和倍点运算,虽然两个公式中模乘运算的次数比较少,但却都需要做一次求逆运算,而求逆运算的开销非常大。通过研究发现,我们通过引进一种新的坐标,称为雅可比(Jacobian)坐标来表示椭圆曲线上的点,这样可以避免在点加和倍点运算中作求逆运算(原来的坐标称为仿射坐标(Affine coordinates),简记为 A)。

3　ECC 的硬件实现

前面说过,在椭圆曲线的实现中,多倍点运算 Q = kP(P 为椭圆曲线点,k 为一整数)是其中最重要的运算,不论是加解密算法还是签名算法,都是由几个多倍点运算组合而成,因而在 ECC 的硬件实现过程中,我们没必要将整个加密签名算法都用硬件实现,而只需要实现其中最主要的多倍点运算,而当实现整个加密签名算法时,可以使用软件实现,在软件实现过程中,调用多倍点运算模块。这样增加了灵活性,提高了整体的效率。

多倍点运算的整个程序流程可概括为:将输入的点和参数化为 Montgomery 表示,然后将点的坐标由仿射坐标化为投射坐标,然后开始点乘法运算,期间不停的调用点加和倍点运算模块,乘法运算结束后,将点的坐标重新化为仿射坐标,再接着由 Montgomery 表示化为正规表示,最终输出结果。文中将生成的结构分为前后相关五层,如图 2 所示。

其中:

第一层:主控模块(Main Controller,亦 MC):MC 包含一个具有 5 个状态的有限状态机,通过状态机的改变对第二层的模块进行调度,最终完成多倍点运算。

第二层:

(1) 仿射坐标向投射坐标转化(AtoP):在进行点的乘法运算之前,要用雅可比(Jacobian)坐标来表示椭圆曲线上的点,所以必须进行仿射坐标向投射坐标转化,即:(X,Y)——>(X,Y,Z,aZ4)

(2) 正规(Normal)表示向 Montgomery 表示转化:由于采用 Montgomery 算法,所以需要将正规表示首先转换为 Montgomey 表示。即:

x——> XRmodM、y——> YRmodM、1——> RmodM、a——>aRmodM

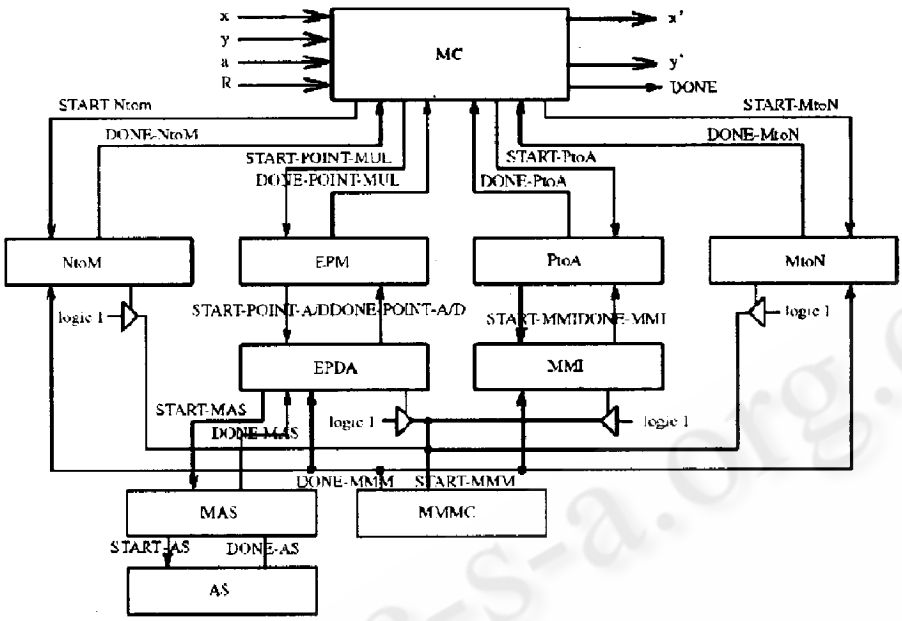

图 2　椭圆曲线点乘运算硬件实现框图

(3) 椭圆曲线点的乘法运算(EPM):通过调用椭圆曲线点的加法及倍点运算模块(EPDA)完成点乘法运算。

(4) 投射坐标向仿射坐标的转化(PtoA):在完成了点的乘法运算之后,所得结果 Q 必须要从投射坐标转化为仿射坐标,即:

XRmod2M——> XZ^{-2} Rmod2M 、YRmod2M——> YZ^{-3}Rmod2M

(5) Montgomery 表示向正规表示转化(MtoN):因为最终的结果应用正规表示,因此由 M toN 模块来完成 Montgomey 表示到正规表示的操作,即:

XRmod2M——>x、YRmod2M——>y

第三层:

(1) 椭圆曲线倍点、加法运算(EPDA)

(2) 逆运算(MMI)

第四层:

(1) Montgomery 模乘运算(MMMC):MMMC 是整个椭圆曲线硬件实现的核心所在,它的实现速度和效率直接影响到整个工程的优劣。MMMC 主要完成运算 MMM:$Mont(a,b) = abR^{-1} \bmod M$ 这一工程中最常用也是最耗时的运算。在我们的实现中,采用前面提到的心缩算法来完成这一运算,根据实现情况来选择 r 的大小,这里为实现方便,我们在实现时选用 r=2。

(2) 模加、减运算(MASC)

第五层:加、减运算(ASC)

以上每个模块被设计成一个独立的电路,有各自的状态转换和数据流向,每个模块可被独立编写并分别验证,但应注意与上下层电路的接口。

4　实现结果

上述模块运行时,可以计算其运行时间,我们在设计时,为了减少延迟拍数,令输入数据的基为 24,令输入的椭圆曲线点 P 的坐标长为 N bits,模乘运算 kP 中 k 的长度为 l bits,表 4.1 给出了各模块的延迟统计,其中计算运行时间时,令 N=192,l=192,时钟主频为 40MHz。

将上面设计的电路用 FPGA 进行仿真,我们选用 Xilinx 的 V1000E-BG-560-8(Virtex E),所选取的素数域规模为 192 比特,曲线固定,最终生成的下载文件需要占用芯片 12288 slices 中的 9055 个,相当于 155,500 个逻辑门。由于选用的基为 24,因而需要做几次 4X4 的乘法,造成延迟较大,最终的仿真结果主频最高为 41.4MHz。虽然基的选择使主频有所降低,但却使总的延迟拍数减少,如表统计所示,点乘运算的速度超过 200 次/秒。而 ECC 中的加解密和签名操作中最多使用两次点乘运算,因而采用本文中运算模块实现的 ECC 最终的加解密和签名认证操作速度超过 100 次/秒。

表 1　各模块的延迟统计

模块名称	MMM 运算次数	clk 数	运算时间(ms)
NtoM	4MMM	3N+12	0.009
EPM	l EPM(a)+l/2 EPM(d)	l(12N+48)	4.8
PtoA	MMI+4MMM	9N2/8+15N/2+12	0.94
MtoN	2MMM	3N/2+6	0.04
EPM(d)	8MMM	6N+24	0.027
EPM(a)	12MMM	9N+36	0.04
MMI	3N/2 MMM	9N2/8+9N/2	0.92
MMM		3(N/4+1)	0.002

(下转第 48 页)

(上接第 45 页)

5 总结和展望

本文中实现的 ECC 点乘运算模块,是一个很基本的结构,还有很多地方可以改进以提高运算的速度和灵活性,可以从以下几个方面作进一步研究:

· 文中直接使用 FPGA 提供的乘法器,使延迟较大。乘法器是一个比较基本的运算模块,只有编写高效的乘法器才能提高整个系统的效率。

· 在心缩算法中,选取不同的基会对系统的主频和资源产生很大的影响,在具体实现时,应多试验,寻求最佳解决方案。

· 存储等方法,都可以使最终的运算速度地得到提高。

· 本论文中实现的 ECC 算法模块,选取的曲线可变,但域是固定的,可以进一步对结构进行改进,使生成的算法模块可以支持可变的域以提高灵活性。

· 论文中实现的 ECC 算法模块是基于素域 Fp 的。

椭圆曲线密码体制提出已经快二十年了,最近几年它成为了研究的热点,主要原因是过去一直使用的 RSA 体制所需的密钥量越来越大,增加了实现的困难,而 ECC 却在这方面有很大的优势,相信在未来一段时间内,ECC 将会更多的为业界所关注,相关的安全产品也会在市场上占有一席之地。

参考文献

1 Qizhi Qiu and Qianxing Xiong, "Research on Elliptic Curve Cryptography"

2 Julio Lopez, Ricardo Dahab "Performance od Elliotic Curve Cryptosystems"

3 Thomas Blum, "Modular Exponentiation on Recongurable Hardware", IEEE Transactions on Computers, 47(7):766 - 76, July 1999.