高性能Ed25519算法硬件架构设计与实现

于 斌 黄 海* 刘志伟 赵石磊 那 宁
(哈尔滨理工大学计算机科学与技术学院 哈尔滨 150080)

摘 要：针对签名验签速度难以满足特定应用领域需求的问题，该文设计了一种高性能Ed25519算法的硬件实现架构。采用宽度为2 bit的窗口法实现标量乘运算，减少了标量乘所需的总周期数；通过优化点加倍点操作步骤，提高了乘法器的硬件使用率；使用低计算复杂度的快速模约简实现模乘，提高了整体运算速度。为了使模L运算可复用标量乘中的快速模约简，该文提出一种基于Barrett约简的模L算法。通过优化解压过程中模幂操作过程，精简了步骤并使其可复用模乘。对所提架构做硬件实现，在TSMC的55 nm CMOS工艺下，面积为746×10^3等效门，最高频率360 MHz，每秒能够执行公钥生成9.06×10^4次、签名8.82×10^4次和验签3.99×10^4次。

关键词：椭圆曲线数字签名算法；爱德华兹曲线；硬件实现；标量乘；快速模约简
中图分类号：TN918; TP309 **文献标识码**：A **文章编号**：1009-5896(2021)07-1821-07
DOI: 10.11999/JEIT200876

High-performance Hardware Architecture Design and Implementation of Ed25519 Algorithm

YU Bin HUANG Hai LIU Zhiwei ZHAO Shilei NA Ning
(*School of Computer Science and Technology, Harbin University of Science and Technology, Harbin* 150080, *China*)

Abstract: The speed of existing signature and verification architecture is difficult to meet the requirement of the specific applications domain, to solve this problem a high-performance hardware architecture of Ed25519 algorithm is developed. The scalar multiplication algorithm is implemented by using the window method with 2 bit width to reduce the total cycle numbers of the algorithm significantly. By optimizing the order of operations of point addition and point doubling, the hardware utilization rate of multiplier is improved. The module multiplication is realized by using fast module reduction with low computational complexity, thus the overall operation speed is improved. The modular L algorithm based on Barrett reduction is proposed to reuse the fast modular reduction in scalar multiplications. By optimizing the modular power computation in the decompression process, the steps are simplified and the modular multiplication can be reused. Under the TSMC 55 nm CMOS process, the area of the proposed hardware architecture is 7.46×10^5 equivalent gate, and the maximum frequency is up to 360 MHz. It can perform 9.06×10^4 key generations, 8.82×10^4 signatures and 3.99×10^4 verifications per second.

Key words: Elliptic Curve Digital Signature Algorithm (ECDSA); Edwards-curve; Hardware implementation; Scalar multiplication; Fast modular reduction

1 引言

数字签名算法在信息安全领域中具有重要作用，常用的算法有RSA 和椭圆曲线数字签名算法(Elliptic Curve Digital Signature Algorithm, ECDSA)等。在最新的传输层安全性协议1.3版(the Transport Layer Security protocol version 1.3, TLS1.3)中，椭圆曲线算法被收录为基本算法[1]，其中爱德华兹曲线数字签名算法(Edwards-curve Digital Signature Algorithm, EdDSA)作为主要的数字签名算法之一，受到研究者的广泛关注。

Curve25519曲线最初由Bernstein提出，用于密钥交换时称为X25519[2]，其Edwards曲线形式用于数字签名算法，称为Ed25519，与另一同类曲线算法Ed448一起被收入到TLS1.3中，成为重要的签

收稿日期：2020-10-12；改回日期：2021-01-29；网络出版：2021-03-01
*通信作者： 黄海 ic@hrbust.edu.cn
基金项目：黑龙江省自然科学基金(YQ2019F010)，黑龙江省博士后科研启动基金(LBH-Q18065)，中央引导地方科技发展专项(ZY20B11)
Foundation Items: The Natural Science Foundation of Heilongjiang (YQ2019F010), Heilongjiang Postdoctoral Funds for Scientific Research Initiation (LBH-Q18065), The Science and Technology Development Special Project of Central Guide the Local Government of China (ZY20B11)

名算法。业界对相关算法做了大量研究，如Faz-Hernández等人[3]使用矢量指令的方式使计算机执行Ed25519的速度得到大幅提升；Islam等人[4]在P-256的曲线上兼容了Ed25519的计算，使其具有更好的通用性；戴紫彬等人[5]改善了架构，使密码处理器能够高效并行；Kim等人[6]讨论了Curve25519和Edwards25519曲线上加法运算的转换效率；Salarifard等人[7]通过快速约简实现模乘并完成了Curve25519曲线上标量乘的设计；Turan等人[8]在可编程门阵列(Field Programmable Gate Array, FPGA)上实现了完整的Ed25519功能；Mehrabi等人[9]做了低功耗低面积的设计，完成了X25519功能并支持ED25519的标量乘运算；魏伟等人[10]研究了椭圆曲线密钥交换协议的比特安全问题；在低延迟、侧信道保护方面的研究也有诸多进展[11–13]。

上述各类研究多使用在物联网(Internet of Things, IoT)设备中，偏重于轻量级设计，对签名和验签速度无特别要求。但在特定应用领域，如服务器端，每秒内需要进行大量的签名验签，运算量巨大，现有算法难以满足应用需求。针对这个问题，本文设计了一种高性能Ed25519算法的硬件实现架构，并在TSMC的55 nm互补金属氧化物半导体(Complementary Metal OxideSemiconducto, CMOS)工艺下完成了硬件实现。首先分析了完整的Ed25519算法，划分关键的运算单元；其次对其中的标量乘单元、模L单元和解压单元进行了设计，并充分考虑复用以减小硬件开销，采用速度较快的模约简方式实现模乘，配合此模乘结构调整了点加倍点操作步骤，并使用窗口法完成了标量乘功能；然后利用已有的模乘结构完成模L单元和解压单元；最后实现完整设计并进行性能分析。仿真结果表明，所设计的Ed25519算法硬件实现架构具有速度高、硬件使用率高、计算周期少等优点。

2 Ed25519算法介绍

Ed25519是椭圆曲线签名算法的一种，选用的曲线为Edwards25519曲线，方程为

$$-x^2+y^2=1-\mathrm{d}x^2y^2=1-\frac{121665}{121666}x^2y^2 \quad (1)$$

在EdDSA的标准中，共描述了公钥生成、签名和验签3个功能的算法[14]。按其工作过程，首先需进行公钥生成，如表1所示。

由公钥生成算法流程可以看出，运算部分主要是1次SHA-512和1次对椭圆曲线基点B的标量乘。最后的压缩过程实际是保留y的全部信息和x的末位信息，组合成为256 bit公钥。

表 1 Ed25519公钥生成算法流程

(1)选择256 bit私钥sk=$(\mathrm{sk}_{255}, \mathrm{sk}_{254},\cdots, \mathrm{sk}_1, \mathrm{sk}_0)_2$;
(2)对sk做SHA-512运算，即$H(\mathrm{sk})=(h_{511},h_{510},\cdots,h_1,h_0)_2$;
(3)取$H(\mathrm{sk})$的低256 bit，并整理为$s=(0,1,h_{253},h_{252},\cdots,h_3,0,0,0)_2$;
(4)做标量乘$A=sB=(x,y)$，其中$x=(x_{254},x_{253},\cdots,x_1,x_0)_2$，$y=(y_{254},y_{253},\cdots,y_1,y_0)_2$;
(5)压缩sB结果，得公钥pk=$(x_0,y_{254},y_{253},\cdots,y_1,y_0)_2$。

当有待签名消息M时，需进行的签名过程如表2所示。模操作中的参数L与参数p一样，均为椭圆曲线的固定参数。签名算法的运算部分主要是2次SHA-512、1次标量乘以及对L的模乘和约简运算。

当接收方获得消息M及签名结果$R||S$，结合发送方的公钥pk，可进行验签操作，具体过程如表3所示。验签过程的解压操作是表1和表2中压缩操作的逆运算，根据压缩结果来计算x的原值。验签算法的运算主要来自2次解压操作、1次SHA-512、2次标量乘和1次点加运算。

3 高速ED25519算法及硬件架构设计

根据Ed25519算法的工作流程，其硬件实现架构设计如图1所示。在整个架构中，SHA-512的设计目前已相对成熟，故不作重点讨论，本文主要针对运算量大的标量乘、模L和解压3部分进行研究和设计，包括标量乘中的点加倍点和模运算等具体操作。

标量乘单元由标量乘控制、乘法器、模约简单元、模逆和模加减构成。其中标量乘控制完成了标量乘算法及点加倍点算法的控制，但仅是对数据的选择控制和部分临时数据的暂存，最终的实现依托模运算，而乘法器和模约简单元配合完成模乘运

表 2 Ed25519签名算法流程

(1)取$H(\mathrm{sk})$的高256 bit $h=(h_{511},h_{510},\cdots,h_{257},h_{256})_2$;
(2)做SHA-512运算，$r=H(h||M) \bmod L$;
(3)做标量乘$R'=rB=(x,y)$，其中$x=(x_{254},x_{253},\cdots,x_1,x_0)_2$, $y=(y_{254},y_{253},\cdots,y_1,y_0)_2$;
(4)压缩rB结果，得$R=(x_0,y_{254},y_{253},\cdots,y_1,y_0)_2$;
(5)做SHA-512运算，$k=H(R||\mathrm{pk}||M) \bmod L$;
(6)计算$S=(r+ks) \bmod L$;
(7)返回签名结果$(R||S)$。

表 3 Ed25519验签算法流程

(1)解压R为点坐标R';
(2)解压pk为点坐标A;
(3)做SHA-512运算，$k=H(R||\mathrm{pk}||M) \bmod L$;
(4)验证$SB=R'+kA$是否成立，若成立则验签成功。

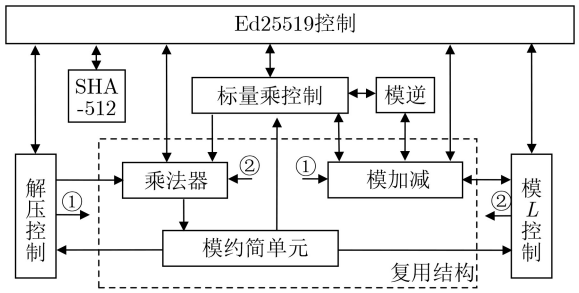

图 1 Ed25519完整结构图

算。模L单元由模L控制、模加减、乘法器和模约简单元构成，解压单元由解压控制、模加减、乘法器和模约简单元构成，由于标量乘、模L和解压是串行运算关系，故复用了模加减、乘法器和模约简。图1中的SHA-512使用通用结构[15]，乘法器使用单元库中现有结构以保证性能，此两部分不做单独设计。最终，Ed25519控制调用各部分实现公钥生成、签名和验签功能。

3.1 标量乘单元设计

标量乘单元性能的优劣是整个Ed25519的直接体现，完整的标量乘运算分为标量乘算法、点加倍点算法和模运算算法3个层次，标量乘调用点加倍点来实现，而点加倍点调用模运算完成。为了提高签名验签速度，在最底层的模运算中采用快速模约简的方式来设计模乘，点加倍点也需随之调整操作流程，最顶层的标量乘算法采用窗口法来实现。

3.1.1 标量乘算法

本设计中标量乘算法采用窗口法[16]，窗口宽度为2 bit，并针对Ed25519验签进行了设计，增加了1次点加操作，如表4所示。

并未采用较热门的非相邻形式(Non-Adjacent Form, NAF)算法，是因为整个系统中有统一时钟，采用NAF算法需对k进行预处理，会消耗额外周期而导致标量乘的总周期数增加，宽度为2 bit的窗口法比NAF算法额外计算11次点加和1次倍点，需要108个周期数完成(详见3.1.2节)，小于NAF算法预计算k所需的平均周期数128个。窗口大小若继续增加，虽会带来总周期数的减少，但会导致存储单元数量的指数增加，预计算所需周期数也会同样增多，故本文最终选定窗口大小为2 bit。

表 4　宽度为2的窗口法

输入：255 bit二进制数$k=\{k_{254}k_{253}\cdots k_1k_0\}$，点加标志位$n$，$n\in\{0,1\}$，椭圆曲线上任意点$P_1$, P_2。
输出：标量乘$Q=kP_1+nP_2$
(1)预计算$2P_1$,$3P_1$，共2个点坐标；
(2)若$k_{254}=0$，则令$Q=0$，为无穷远点，若$k_{254}=1$，则令$Q=P_1$；
(3)对于i从126～0，循环计算：
$Q=4Q$；
$Q=Q+\{k_{2i+1},k_{2i}\}P_1$；
(4)若$n=1$，则计算：$Q=Q+P_2$ //为适应验签算法；
(5)返回Q值。

3.1.2 点加倍点算法

点加和倍点运算在标准文档中给出了计算步骤，为得到更好的性能，将2元射影坐标(x, y)转换为4元扩展齐次坐标(X, Y, Z, T)，转换公式为

$$x=X/Z, y=Y/Z, x\times y=T/Z \tag{2}$$

点加和倍点计算按表5的步骤完成，左侧完成点加，右侧完成倍点，其中d与式(1)中相同，等式左侧均为计算中涉及的临时变量或输出结果。

由于底层的模乘采用模约简来实现，乘法结果需等待一个周期约简后才能再次相乘，考虑到乘法器的使用效率，把表5的操作流程做重新排布，表6

表 5　点加和倍点操作流程[14]

输入：$P(X_1,Y_1,Z_1,T_1)$, $Q(X_2,Y_2,Z_2,T_2)$		
输出：$Q(X_3,Y_3,Z_3,T_3)=P+Q$, $Q(X_3,Y_3,Z_3,T_3)=2P$		
(1)	$A=(Y_1-X_1)\cdot(Y_2-X_2)$	$A=(X_1)^2$
(2)	$B=(Y_1+X_1)\cdot(Y_2+X_2)$	$B=(Y_1)^2$
(3)	$C=T_1\cdot2\cdot d\cdot T_2$	$C=2\cdot(Z_1)^2$
(4)	$D=Z_1\cdot2\cdot Z_2$	$H=A+B$
(5)	$E=B-A$	$E=H-(X_1+Y_1)^2$
(6)	$F=D-C$	$G=A-B$
(7)	$G=D+C$	$F=C+G$
(8)	$H=B+A$	$X_3=E\cdot F$
(9)	$X_3=E\cdot F$	$Y_3=G\cdot H$
(10)	$Y_3=G\cdot H$	$T_3=E\cdot H$
(11)	$T_3=E\cdot H$	$Z_3=F\cdot G$
(12)	$Z_3=F\cdot G$	

表 6　倍点操作流程

输入：点坐标$P(X_1,Y_1,Z_1,T_1)$, $Q(X_2,Y_2,Z_2,T_2)$		
输出：$Q(X_3,Y_3,Z_3,T_3)=2P$		
步骤	模乘	模加减
(1)	$A=X_1\cdot X_1$	
(2)	$t_1=Z_1\cdot Z_1$	
(3)	$B=Y_1\cdot Y_1$	$t_2=X_1+Y_1$
(4)	$t_2=t_2\cdot t_2$	$C=t_1+t_1$, $H=A+B$, $G=A-B$
(5)	$Y_2=G\cdot H$	$E=H-t_2$, $F=C+G$
(6)	$X_2=E\cdot F$	
(7)	$Z_2=F\cdot G$	
(8)	$T_2=E\cdot H$	
(9)	等待T_2约简	

是排布后倍点的操作流程，点加也按同样考虑，排布为表7所示操作流程。

表4窗口法的预计算中$2P_1$按倍点操作，4倍点则执行两次倍点，所以循环过程按“倍点→倍点→点加”的顺序来进行，表6中的步骤(7)，(8)能与表7中的步骤(1)，(2)同时完成，还可以和表6中的步骤(1)，(2)同时完成，两种运算最后的等待约简步骤也可以与下一运算的初始步骤同时运行，即倍点和点加无论如何搭配工作，乘法器均可连续工作，整个标量乘循环只有在结束时等待约简1次即可，乘法器的使用率近似达到百分之百。按此计算，1次“倍点→倍点→点加”实际只需25个周期，无点加时1次“倍点→倍点”实际只需16个周期。

3.1.3 模运算算法

模运算包含模加减、模乘和模逆。为保证表6和表7中的操作可以执行，需正常的模加和模减单元各1个。由于模乘设计采用快速模约简的方式实现，考虑到乘法器延迟较大，在模约简单元输出端连接1个模加和1个模减单元，可以在1个周期内对模约简的结果多做1次模加减操作。此外模逆也需要复用模加和模减单元来简单完成，整个模运算部分的结构如图2所示。

表 7 点加操作流程

输入：点坐标$P(X_1,Y_1,Z_1,T_1)$，$Q(X_2,Y_2,Z_2,T_2)$
输出：$Q(X_3,Y_3,Z_3,T_3)=P+Q$

步骤	模乘	模加减
(1)		$A_1\leftarrow Y_1-X_1$, $B_1\leftarrow Y_1+X_1$
(2)		$A_2\leftarrow Y_2-X_2$, $B_2\leftarrow Y_2+X_2$
(3)	$A\leftarrow A_1\cdot A_2$	
(4)	$B\leftarrow B_1\cdot B_2$	$t_1=T_1+T_1$
(5)	$t_1\leftarrow t_1\cdot d$	$t_2=Z_1+Z_1$
(6)	$D\leftarrow t_2\cdot Z_2$	
(7)	$C\leftarrow t_1\cdot T_2$	$E\leftarrow B-A$, $H\leftarrow B+A$
(8)	$X_3\leftarrow E\cdot H$	$F\leftarrow D-C$, $G\leftarrow D+C$
(9)	$Y_3\leftarrow G\cdot H$	
(10)	$Z_3\leftarrow F\cdot G$	
(11)	$T_3\leftarrow E\cdot H$	
(12)	等待Z_3约简	

计算标量乘只需255 bit乘法，但模L运算需要计算257 bit乘法，故选用单元库中的257 bit乘法器，可同时满足标量乘和模L运算需求。模约简公式通常可以根据模数特点利用多项式来推导[17]。在Ed25519中，参数$p=2^{255}-19=2^{255}-2^4-2^2+1$，位宽为255 bit，乘法结果$A$的位宽为510 bit，根据$p$值的形式特点，设$A=A_{254}\cdot 2^{508}+A_{253}\cdot 2^{506}+\cdots+A_1\cdot 2^2+A_0$，$A_i$的宽度为2 bit。其中低256 bit即$A_{127}\cdot 2^{254}+\cdots+A_0$，设为$T$；高254 bit即$A_{254}\cdot 2^{508}+\cdots+A_{128}\cdot 2^{256}$做两轮多项式除法，第1轮后得到剩余项$A_{254}\cdot 2^{257}+(A_{253}+A_{254})2^{255}+(A_{252}+A_{253}-A_{254})2^{253}+(A_{251}+A_{252}-A_{253})2^{251}+\cdots+(A_{128}+A_{129}-A_{130})2^5+(A_{128}-A_{129})2^3+(-A_{128})2^1$，第2轮消去$A_{254}\cdot 2^{257}+(A_{253}+A_{254})2^{255}$，得到新增剩余项$A_{254}\cdot 2^6+(A_{253}+2A_{254})2^4+A_{253}2^2-(A_{253}+A_{254})$，把各部分重新整理后，可得表8所示的快速模约简算法。其中T值是256 bit，S_i和D_i是255 bit，最后一步待约简的9个和差结果的范围在$-2p\sim 5p$内，然后接入一个加减法阵列计算出所有可能，并根据标志位来选择正确的输出，整个模约简结构如图3所示。

图3中额外增加了选择信号来选择输入参数是L或p，以及增加了$+3q$的输出，这两处是为了在模L约简过程中使用，由于模L时不会同时进行模p，可以最大限度复用硬件资源，模L约简算法及设计见3.2节。

模逆只在从4元坐标转换为2元坐标时使用1次，采用二进制右移算法[18]，至多512个周期即可完成运算，每次循环向右移位1次，并做除2操作，可用模加完成。

以上各部分按层次调用，即可完成标量乘运算。

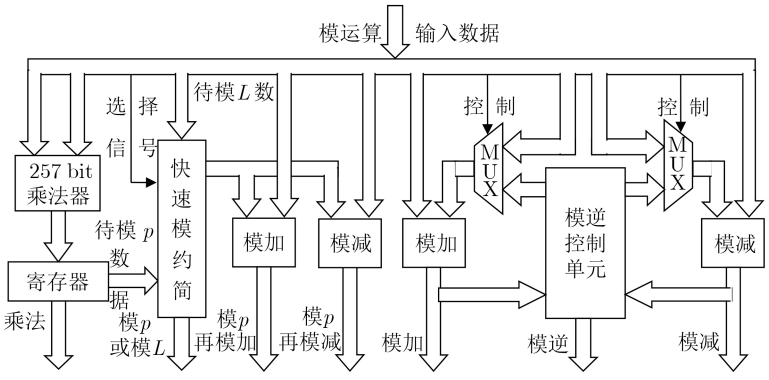

图 2 模运算单元

表 8　Ed25519快速约简算法

输入：$A=(A_{254}||A_{253}||A_{252}||\cdots||A_2||A_1||A_0)$

输出：$Q=A \bmod p$

(1) $T=(A_{127}||\ A_{126}||\ A_{125}||\ \cdots||\ A_2||\ A_1||\ A_0)$

(2) $S_1=(A_{252}||\ A_{251}||\ A_{250}||\ \cdots||\ A_{129}||\ A_{128}||5'\ \mathrm{h0})$

(3) $S_2=(A_{253}||\ A_{252}||\ A_{251}||\ \cdots||\ A_{129}||\ A_{128}||3'\ \mathrm{h0})$

(4) $S_3=(247'\ \mathrm{h0}||\ A_{254}||\ A_{254}||4'\ \mathrm{h0})$

(5) $S_4=(249'\ \mathrm{h0}||\ A_{253}||\ A_{253}||2'\ \mathrm{h0})$

(6) $S_5=(249'\ \mathrm{h0}||\ A_{254}||\ 4'\ \mathrm{h0})$

(7) $D_1=(A_{254}||\ A_{253}||\ A_{252}||\ \cdots||\ A_{129}||\ A_{128}||1'\ \mathrm{h0})$

(8) $D_2=(253'\ \mathrm{h0}||\ A_{254})$

(9) $D_3=(253'\ \mathrm{h0}||\ A_{253})$

(10) $Q=(T+S_1+S_2+S_3+S_4+S_5-D_1-D_2-D_3)\bmod p$

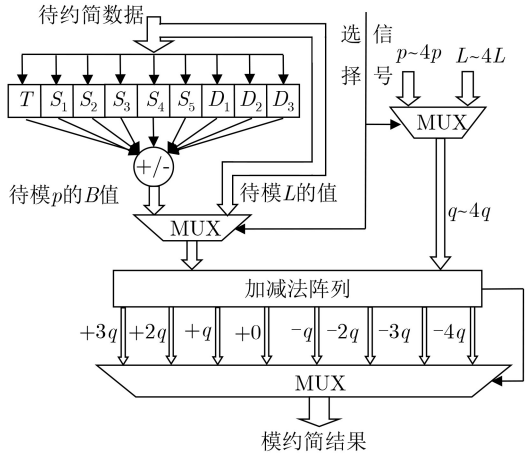

图 3 模约简整体结构

3.2　模L单元的硬件设计

模L运算中的L值形式复杂，不适合做快速约简；且L为253 bit，待约简数值最大为512 bit，异于常规约简的2倍位宽形式；同时考虑到模L运算使用次数较少，单独设计会造成资源闲置；综合以上几点，对比较常用的Barrett约简[19]做改进，复用已有的257 bit乘法器，得到基于Barrett约简的模L算法如表9所示，其中“[ ]”为取整操作。

原Barrett约简是对256 bit做模约简，在步骤4中判断输出结果是否大于模数，并进行修正。由于L是253 bit，将L扩大8倍，变为256 bit $8L$，采用Barrett约简会得到一个范围在0～$8L$的值，减去$3L$后送入图3所示的模约简单元，复用该单元便可得到最终结果，所以直接在步骤(4)中把修正的数值改为减$11L$和减$3L$。此方法只需将乘法器扩展2 bit、模约简单元增加少量结构、模减单元扩展1 bit以及额外增加部分控制逻辑即可实现模L约简。

3.3　解压单元的硬件设计

解压的目的是根据坐标x值的最后一位和坐标y值，来计算x的完整坐标值，计算公式为

表 9　基于Barrett约简的模L算法

输入：素数L, 512 bit数值a, $T=[2^{512}/(8L)]$

输出：$r=a \bmod L$

(1)$q_1=[a/2^{255}]\cdot T$;

(2)$q_2=[\ q_1/2^{257}]\cdot(8L)$;

(3)$r=(a \bmod 2^{257})-(q_2 \bmod 2^{257})$;

(4)若果$r<0$，则$r=r+2^{257}$;

(5)如果$r\geq 8L$，则$r=r-11L$，否则$r=r-3L$；//可复用快速模约简

(6)计算$r=r \bmod L$并返回r值。

$$x^2=\frac{y^2-1}{d\cdot y^2+1}=\frac{u}{v} \bmod p \tag{3}$$

解压运算采用标准中推荐的算法来实现[14]，需计算

$$t=uv^3(uv^7)^{(p-5)/8} \tag{4}$$

再对t进行判断和修正即可，主要计算量集中在模幂操作，由于参数p固定，可得

$$(p-5)/8=2^{252}-3 \tag{5}$$

根据其形式特点，复用快速模约简的模乘结构，为达到乘法器的最高使用率，设计了如图4的模幂操作步骤图，其中的$b=uv^7$，是模幂的底数。按此操作步骤运行，完成b的$(2^{252}-3)$次模幂只需506个周期。

4　硬件实现与性能分析

最终设计采用55 nmCMOS工艺库进行综合，可得表10和表11所示的性能参数结果。整个设计能够工作在360 MHz主频下，约占用746×10^3个等效门，平均每秒可以运行11.12×10^4次的公钥生成、10.76×10^4次的签名和4.81×10^4次的验签，即使对于极端特殊的情况，理论上也能达到每秒9.06×10^4次的公钥生成、8.82×10^4次的签名和3.99×10^4次的验签。由于未设置额外存储器，1次待签名信息需小于512 bit。

由于Ed25519的实现多用于IoT设备，习惯基于短位宽乘法的迭代来设计，而FPGA中的DSP单元可以更灵活地提供乘法性能，故已知研究常在FPGA上进行。文献[8]在FPGA上完成了Ed25519的全过程，其优化算法每秒分别可以计算474次的公钥生成、621次的签名和272次验签，表12使用的是文献[8]中每个功能所需的平均时间。

完整的Ed25519硬件实现很少，为了进一步评估性能，用Xilinx的Zynq-7035对本设计的标量乘做简单约束实现，并给出表13的对比结果。本设计中并未投入精力对257 bit乘法器做进一步时序优化，用以在FPGA上达到较高工作频率，所以使用

了周期数来衡量运算速度，用消耗的Slices和Cycles的乘积再乘以10^{-6}得到的结果值做性能对比的参考，其中文献[9]未给出Slices参数，使用LUT/4来近似计算。

由表13可以看出，文献[8]的标量乘使用了15个

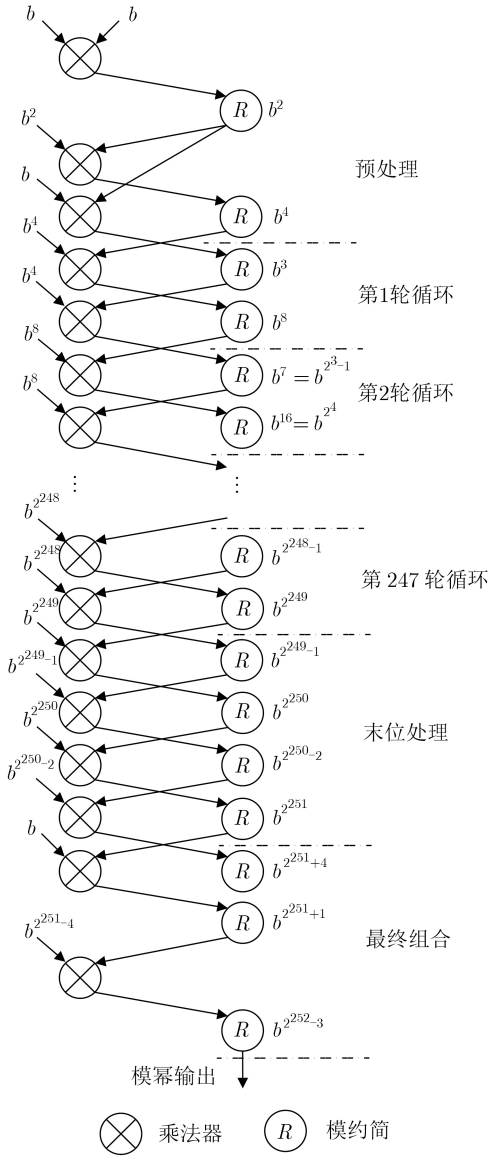

图 4 b的$2^{252}-3$次模幂操作步骤图

DSP做循环迭代来计算256 bit模乘，这样消耗的硬件资源很少，对于IoT设备是较好的选择，但需要消耗很多周期。最终使用该标量乘来完成的Ed25519，在其执行速度上也如表12所列一样，并未有太多竞争性。文献[9]采用基8的循环移位模乘，所以没有DSP单元的消耗，但整体性能稍差。文献[11]使用快速模约简，分为高128 bit和低128 bit来进行模乘，所以所需周期数是所有文献中最少的，同样，硬件资源的消耗情况也是所有文献中最大的。文献[12,13]也是使用FPGA中的DSP单元按17 bit分段进行模乘，只是迭代过程各有区别。与上述文献相比，本设计中标量乘的整体性能可提高约10%，具备一定的优势，而标量乘是整个Ed25519的核心单元，也能从一定层面上反映整体的性能。

5 结论

本文对椭圆曲线签名算法中的Ed25519算法进行了研究，使用快速模约简计算模乘，并据此完成了点加倍点操作步骤的重新排布，然后使用窗口法实现了标量乘运算。设计中改进Barrett约简用于

表 10 Ed25519性能参数

	时钟周期	主频	面积	等效门数
参数值	2.78 ns	360 MHz	1.433×10^6	746×10^3

注1：按工艺库提供的系数1.92从面积折算

表 11 周期及运算次数

	平均周期	理论最大周期	平均每秒运算次数	理论最慢每秒运算次数
公钥生成	3237	3975	111.2×10^3	90.6×10^3
签名	3345	4083	107.6×10^3	88.2×10^3
验签	7488	9020	48.1×10^3	39.9×10^3

表 12 Ed25519运算速度对比(μs)

	公钥生成	签名	验签
本文	9.0	9.3	20.8
文献[8]	2109.7	1610.3	3676.5

表 13 标量乘单元性能对比

	器件	硬件开销Slices/LUT/FF/DSP/BRAM	周期数Cycles	折算值Slices×Cycles×10^{-6}
本文	Zynq-7035	14759/52512/9342/225/0	3895	57.5
文献[8]	Zynq-7020	775/2707/962/15/0	120260	93.2
文献[9]1)	Zynq-7000	2170/8680/3472/0/0	86348	187.4
文献[9]2)	Zynq-7000	2193/8770/3729/0/0	74783	189.3
文献[11]	Zynq-7020	6161/17939/21077/175/0	10465	64.5
文献[12]	Zynq-7020	1006/0/0/20/2	114980	115.7
文献[13]	Zynq-7020	1029/2783/3592/20/2	79400	81.7

1)：采用D&A算法；2)：采用NAF算法

模L的实现，还为解压算法设计模幂的操作步骤，充分复用已有单元，在保证运算速度的前提下，尽可能地减小硬件开销。最后实现的硬件电路能够完成Ed25519的完整操作，相较于其他设计，在运算速度上具有显著优势。

参 考 文 献

[1] RESCORLA E. IETF RFC 8446 The Transport Layer Security (TLS) protocol version 1.3[S]. 2018.

[2] LANGLEY A, HAMBURG M, and TURNER S. IRTF RFC 7748 Elliptic curves for security[S]. 2016.

[3] FAZ-HERNÁNDEZ A, LÓPEZ J, and DAHAB R. High-performance implementation of elliptic curve cryptography using vector instructions[J]. *ACM Transactions on Mathematical Software*, 2019, 45(3): 25.1–25.35. doi: 10.1145/3309759.

[4] ISLAM M M, HOSSAIN M S, HASAN M K, *et al*. FPGA implementation of high-speed area-efficient processor for elliptic curve point multiplication over prime field[J]. *IEEE Access*, 2019, 7: 178811–178826. doi: 10.1109/ACCESS.2019.2958491.

[5] 戴紫彬, 易肃汶, 李伟, 等. 椭圆曲线密码处理器的高效并行处理架构研究与设计[J]. 电子与信息学报, 2017, 39(10): 2487–2494. doi: 10.11999/JEIT161380.
DAI Zibin, YI Suwen, LI Wei, *et al*. Research and design of efficient parallel processing architecture for elliptic curve cryptographic processor[J]. *Journal of Electronics & Information Technology*, 2017, 39(10): 2487–2494. doi: 10.11999/JEIT161380.

[6] KIM J, PARK J H, KIM D C, *et al*. Complete addition law for Montgomery curves[C]. The 22nd International Conference on Information Security and Cryptology– ICISC 2019, Seoul, South Korea, 2019: 260–277. doi: 10.1007/978-3-030-40921-0_16.

[7] SALARIFARD R and BAYAT-SARMADI S. An efficient low-latency point-multiplication over curve25519[J]. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 2019, 66(10): 3854–3862. doi: 10.1109/TCSI.2019.2914247.

[8] TURAN F and VERBAUWHEDE I. Compact and flexible FPGA implementation of Ed25519 and X25519[J]. *ACM Transactions on Embedded Computing Systems*, 2019, 18(3): 24. doi: 10.1145/3312742.

[9] MEHRABI M A and DOCHE C. Low-cost, low-power FPGA implementation of ED25519 and CURVE25519 point multiplication[J]. *Information*, 2019, 10(9): 285. doi: 10.3390/info10090285.

[10] 魏伟, 陈佳哲, 李丹, 等. 椭圆曲线Diffie-Hellman密钥交换协议的比特安全性研究[J]. 电子与信息学报, 2020, 42(8): 1820–1827. doi: 10.11999/JEIT190845.
WEI Wei, CHEN Jiazhe, LI Dan, *et al*. Research on the bit security of elliptic curve Diffie-Hellman[J]. *Journal of Electronics & Information Technology*, 2020, 42(8): 1820–1827. doi: 10.11999/JEIT190845.

[11] KOPPERMANN P, DE SANTIS F, HEYSZL J, *et al*. Low-latency X25519 hardware implementation: Breaking the 100 microseconds barrier[J]. *Microprocessors and Microsystems*, 2017, 52: 491–497. doi: 10.1016/j.micpro.2017.07.001.

[12] SASDRICH P and GÜNEYSU T. Exploring RFC 7748 for hardware implementation: Curve25519 and Curve448 with side-channel protection[J]. *Journal of Hardware and Systems Security*, 2018, 2(4): 297–313. doi: 10.1007/s41635-018-0048-z.

[13] SASDRICH P and GÜNEYSU T. Implementing Curve25519 for side-channel--protected elliptic curve cryptography[J]. *ACM Transactions on Reconfigurable Technology and Systems*, 2015, 9(1): 3. doi: 10.1145/2700834.

[14] JOSEFSSON S and LIUSVAARA I. IRTF RFC 8032 Edwards-curve digital signature algorithm (EdDSA)[S]. 2017.

[15] VENGALA D V K, KAVITHA D, and KUMAR A P S. Secure data transmission on a distributed cloud server with the help of HMCA and data encryption using optimized CP-ABE-ECC[J]. *Cluster Computing*, 2020, 23(3): 1683–1696. doi: 10.1007/s10586-020-03114-1.

[16] LI Hui. Pseudo-random scalar multiplication based on group isomorphism[J]. *Journal of Information Security and Applications*, 2020, 53: 102534. doi: 10.1016/j.jisa.2020.102534.

[17] ZHANG Neng, YANG Bohan, CHEN Chen, *et al*. Highly efficient architecture of NewHope-NIST on FPGA using low-complexity NTT/INTT[J]. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2020, 2020(2): 49–72. doi: 10.13154/tches.v2020.i2.49-72.

[18] HOSSAIN M S, KONG Yinan, SAEEDI E, *et al*. High-performance elliptic curve cryptography processor over NIST prime fields[J]. *IET Computers & Digital Techniques*, 2017, 11(1): 33–42. doi: 10.1049/iet-cdt.2016.0033.

[19] KNEZEVIC M, VERCAUTEREN F, and VERBAUWHEDE I. Faster interleaved modular multiplication based on Barrett and Montgomery reduction methods[J]. *IEEE Transactions on Computers*, 2010, 59(12): 1715–1721. doi: 10.1109/TC.2010.93.

于　斌：男，1984年生，讲师，研究方向为密码算法、密码芯片设计和数字集成电路设计等.

黄　海：男，1982年生，硕士生导师，研究方向为信息安全、可重构技术、集成电路设计等.

刘志伟：男，1987年生，讲师，研究方向为可重构计算、高速密码算法、并行加密技术、密码芯片的安全设计等.

赵石磊：男，1979年生，硕士生导师，研究方向为信息安全、高速密码算法、密码芯片的安全设计等.

那　宁：男，1995年生，博士生，研究方向为信息安全和集成电路设计等.

责任编辑：余　蓉