基于RISC-V的AES密码加速引擎设计与验证

张晓磊，戴紫彬，郭朋飞，李　杨
（信息工程大学，河南 郑州 450001）

摘　要： 随着物联网技术的快速发展和物联网设备的广泛部署，信息安全问题日益凸显。密码是保障信息安全的关键核心技术，但传统的密码算法适配方案存在性能和灵活性难以兼顾的问题，提出了一种密码指令扩展方案在两者之间取得了很好的平衡。首先分析了AES算法的运算环节，结合蜂鸟E203处理器架构，提出了密码指令扩展和加速引擎设计方案；接着进行了软硬件实现，构建了RTL级仿真环境和FPGA板级验证环境；最后进行了实验验证和对比分析。实验结果表明，提出的方案在只增加接近2%的硬件资源的情况下可以取得约700%的加速比，具有较高的能效，可适用于在物联网等资源受限的场合。

关键词： RISC-V；密码指令扩展；加速引擎；信息安全
中图分类号： TN47　　**文献标志码：** A　　**DOI:** 10.16157/j.issn.0258-7998.223046

中文引用格式： 张晓磊，戴紫彬，郭朋飞，等. 基于RISC-V的AES密码加速引擎设计与验证[J]. 电子技术应用，2023，49(2)：39-44.
英文引用格式： Zhang Xiaolei，Dai Zibin，Guo Pengfei，et al. Design and verification of AES cryptographic acceleration engine based on RISC-V[J]. Application of Electronic Technique，2023，49(2)：39-44.

Design and verification of AES cryptographic acceleration engine based on RISC-V

Zhang Xiaolei，Dai Zibin，Guo Pengfei，Li Yang
(Information Engineering University，Zhengzhou 450001，China)

Abstract： With the rapid development of IoT technology and the widespread deployment of IoT devices, the issue of information security has become increasingly prominent. Cryptography is the key core technology to ensure information security, but the traditional cryptographic algorithm adaptation scheme is difficult to balance performance and flexibility, this paper proposes a cryptographic instruction extension scheme to achieve a good balance between the two scheme. Firstly, we analyze the computational aspects of the AES algorithm, and propose a cryptographic instruction extension and acceleration engine design scheme by combining the Hummingbird E203 processor architecture; then we complete hardware and software implementation, build an RTL-level simulation environment and an FPGA board-level verification environment; finally, we perform experimental verification and comparative analysis. The experimental results show that the proposed scheme can achieve about 700% acceleration ratio with only nearly 2% increase in hardware resources, which has high energy efficiency and can be applied in resource-constrained situations such as IoT.

Key words： RISC-V；cryptographic instruction extension；acceleration engine；information security

0 引言

随着物联网技术的快速发展，智能穿戴、自动驾驶、面部识别等应用场景成为现实，极大改变了人们的生活方式。物联网设备大量使用，人体生理指标、车辆行驶轨迹等用户隐私数据[1]也随之产生。由于大量用户数据需要传输到算力更强的计算终端，传输过程中的信息安全隐患[2]逐渐浮现并引起了人们的重视。因为受限于紧张的硬件资源，很多物联网设备并未运行必要的安全机制[3]。

密码技术[4]作为保障信息安全的核心技术，可在物联网设备中进行部署，传统的部署方案主要有两种[5]，一种是通过运行软件实现密码算法，这种方法利用了处理器的通用指令来支持不同的密码算法，虽然较为灵活，但该方法存在计算速度慢、代码密度低的问题；另一

种是专用的密码处理芯片，专用芯片虽然运算速度快但存在灵活性低、成本高的问题。由于智能手环等物联网终端存在计算资源紧张、存储空间有限和电池容量较低等问题[6]，传统的部署方案不能很好地解决上述问题。扩展专用密码指令方案的出现则克服了上述两种方案的缺点，通过向通用处理器中添加密码运算单元，使处理器在不失通用性的同时，还获取了较高的密码运算性能。RISC-V因为其短小精悍的架构和模块化的的设计理念已成为专用领域架构的首选[7]。

在专用指令扩展领域，学术界也有很多研究成果，李爱国等人[8]提出了利用MIPS处理器中的乘法结果寄存器实现了对64比特数据的操作能力，同时利用处理器的空闲流水周期缩短密码运算的关键路径，从而提升了密码运算速度，但该方案对处理器运算路径的修改增加了扩展密码指令的难度和硬件的复杂性。复旦大学的Wang Weizhen等人[9]设计了基于Rocket处理器的四级流水密码协处理器并通过ROCC接口扩展了密码指令。该协处理器采用同时支持128位和256位数据路径的统一流水线结构，支持AES、ECC和SHA等加密算法。但该方案所集成的Rocket处理器是通过Chisel语言构造的，目前该语言还未在业界得到广泛应用，对只熟悉Verilog的芯片工程师进一步研究和开发造成了困难。

本文通过分析基于RISC-V的蜂鸟E203处理器NICE扩展接口及AES密码算法，设计了基于蜂鸟E203处理器的密码加速引擎，并进行了仿真和FPGA验证，实验结果表明，本文提出并实现的密码加速方案获得了较好的效能。

1 AES算法分析及方案设计

本节主要针对AES算法轮函数运算特点，结合蜂鸟E203处理器结构，提出密码加速引擎设计及指令扩展方案。

1.1 AES算法

高级加密标准（AES）[10]是由比利时密码专家Daemen J和Rijmen V提交的Rijndael分组密码算法经过近3年的激烈角逐最后胜出的对称密码加密标准。该算法具有高效能、易实现和灵活性高等特点。

AES算法运算过程在一个4×4（4行4列）的状态矩阵上进行，状态矩阵如式（1）所示：

$$\boldsymbol{a}=\begin{bmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{10} & a_{11} & a_{12} & a_{13} \\ a_{20} & a_{21} & a_{22} & a_{23} \\ a_{30} & a_{31} & a_{32} & a_{33} \end{bmatrix} \tag{1}$$

轮函数迭代l_k轮完成运算，算法的分组长度为128比特，密钥长度N_r可以根据加密强度的需要设置为128、192或256比特，轮数l_k取决于密钥长度，两者满足关系式$N_r=6+l_k/32$。AES算法加密过程如图1所示，轮函数包括字节替代变换、行移位变换、列混合变换和圈密钥加法四个运算，其中字节替代变换和行移位变换两步运算顺序变化不会影响运算结果，最后一轮轮函数没有列混合变换。

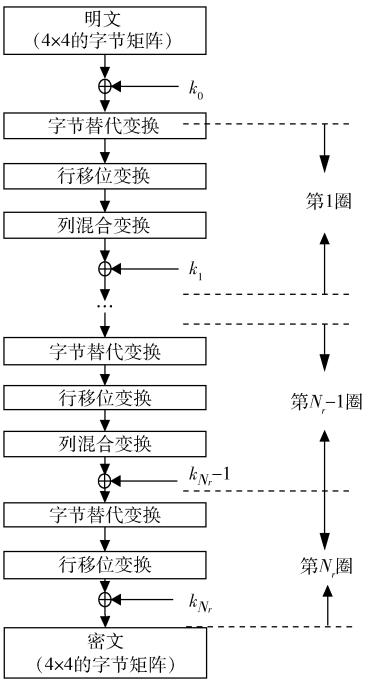

图1 AES加密算法

1.2 高效的密码加速实现方案

Ben Marshall等人[11]针对不同架构处理器AES算法加速方案进行了分析和对比，本文针对物联网场景下硬件资源紧凑的特点，在文献[11]的基础上，采用字节替代、列混合和行移位融合计算的方法实现AES-128加速。

轮函数首先对式（2）中状态矩阵进行字节替代（S盒替代变换用sbox表示）：

$$b_{i,j}=\mathrm{sbox}(a_{i,j}) \tag{2}$$

随后进行行移位，如式（3）所示：

$$\begin{bmatrix} c_{0,j} \\ c_{1,j} \\ c_{2,j} \\ c_{3,j} \end{bmatrix}=\begin{bmatrix} b_{0,\mathrm{j}} \\ b_{1,j+1} \\ b_{2,j+2} \\ b_{3,j+3} \end{bmatrix} \tag{3}$$

之后进行列混合变换，它将一个状态的每一列视为有限域GF(2^8)上的一个多项式且与一个固定多项式$a(x)=\{03\}x^3+\{01\}x^2+\{01\}x+\{02\}$模$x^4+1$相乘：

$$\begin{bmatrix} d_{0j} \\ d_{1j} \\ d_{2j} \\ d_{3j} \end{bmatrix}=\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}\otimes\begin{bmatrix} c_{0j} \\ c_{1j} \\ c_{2j} \\ c_{3j} \end{bmatrix} \tag{4}$$

最后与轮密钥key相加：

$$\begin{bmatrix} e_{0,j} \\ e_{1,j} \\ e_{2,j} \\ e_{3,j} \end{bmatrix}=\begin{bmatrix} d_{0,j} \\ d_{1,j} \\ d_{2,j} \\ d_{3,j} \end{bmatrix}\oplus\mathrm{key}_j \tag{5}$$

将上述字节替代、行移位、列混合与轮密钥相加合并后可得：

$$\begin{bmatrix} e_{0j} \\ e_{1j} \\ e_{2j} \\ e_{3j} \end{bmatrix}=\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix}\otimes\begin{bmatrix} \mathrm{sbox}(a_{0j}) \\ \mathrm{sbox}(a_{1,j+1}) \\ \mathrm{sbox}(a_{2,j+2}) \\ \mathrm{sbox}(a_{3,j+3}) \end{bmatrix} \tag{6}$$

针对AES-128，进一步换算后可得：

$$\begin{aligned}&[e_{0j}\quad e_{1j}\quad e_{2j}\quad e_{3j}]^{\mathrm{T}}=\\&(\mathrm{sbox}(a_{0,j})\cdot[02\quad 01\quad 01\quad 03]^{\mathrm{T}})\\&\oplus(\mathrm{sbox}(a_{1,j+1})\cdot[03\quad 02\quad 01\quad 01]^{\mathrm{T}})\\&\oplus(\mathrm{sbox}(a_{2,j+2})\cdot[01\quad 03\quad 02\quad 01]^{\mathrm{T}})\\&\oplus(\mathrm{sbox}(a_{3,j+3})\cdot[01\quad 01\quad 03\quad 02]^{\mathrm{T}})\oplus \mathrm{key}_j\end{aligned}\tag{7}$$

将式（7）分解为5部分，即$[\mathrm{sbox}(a)\cdot 02\quad \mathrm{sbox}(a)\quad \mathrm{sbox}(a)\quad \mathrm{sbox}(a)\cdot 03]^{\mathrm{T}}$、$[\mathrm{sbox}(a)\cdot 03\quad \mathrm{sbox}(a)\cdot 02\quad \mathrm{sbox}(a)\quad \mathrm{sbox}(a)]^{\mathrm{T}}$、$[\mathrm{sbox}(a)\quad \mathrm{sbox}(a)\cdot 03\quad \mathrm{sbox}(a)\cdot 02\quad \mathrm{sbox}(a)]^{\mathrm{T}}$、$[\mathrm{sbox}(a)\quad \mathrm{sbox}(a)\quad \mathrm{sbox}(a)\cdot 03\quad \mathrm{sbox}(a)\cdot 02]^{\mathrm{T}}$和$\mathrm{key}_j$。

分解之后发现，除了与密钥的异或运算外，其余四个部分的计算可以用4条自定义指令完成运算，每条指令参与运算的系数分别为{2,1,1,3}、{3,2,1,1}、{1,3,2,1}和{1,1,3,2}，通过4条自定义指令完成一次轮函数中一个字的运算并存储于32比特位宽的寄存器中，这样大大提高了运算效率，并且不会消耗太多的硬件资源，需要注意的是解密运算时参与有限域运算的系数不同，其系数分别为{b,d,9,e}、{d,9,e,b}、{9,e,b,d}和{e,b,9,d}。

1.3 密码指令扩展方案

RISC-V指令集由基础指令集和扩展指令集构成，为了便于用户扩展，RISC-V指令集设计之初便为用户提供了custom-0、custom-1、custom-2和custom-3四个用户自定义编码空间，蜂鸟E203处理器识别到上述四个类型的指令编码后，就会把自定义扩展指令通过扩展接口派发给协处理器，由协处理器进行译码、执行和写回等操作。蜂鸟E203处理器支持的扩展指令编码格式[12]如图2所示，其中xd、xs1和xs2用来指示是否需要读取rd、rs1和rs2。

通过分析轮函数计算过程，四次计算中参与有限域乘法的系数不同，因此指令编码中包含的信息应当能够区分乘法系数的顺序；因为最后一轮轮函数仅需要完成字节替代函数和行移位运算，所以指令编码中也应当包括轮数信息以指示当前计算的轮函数是否为最后一轮。

综上所述，密码扩展指令编码空间使用custom-0，通过funct7字段区分系数顺序和是否为最后一圈轮函数，所有指令均为R型指令，源操作数rs1为上一次运算结果，源操作数rs2存储状态矩阵，目的寄存器存储本次运算结果。自定义扩展指令根据运算圈数和加解密状态分为四类，具体编码情况如表1所示。

2 基于蜂鸟E203的AES加速引擎设计

2.1 状态矩阵的表示与存储

AES状态矩阵为一个4×4的矩阵，每一个元素为一个字节，共16字节。蜂鸟E203处理器为RV32IMAC架构，寄存器位宽为32 bit，因此，四个寄存器可以表示一个状态矩阵。

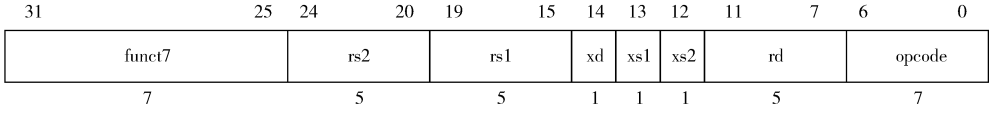

图2　扩展指令编码格式

表1　自定义指令编码

序号	指令助记符	指令功能	funct7	xd	xs1	xs2
1	Enc1FFM	第一顺序系数加密运算	0001011	1	1	1
2	Enc2FFM	第二顺序系数加密运算	0001100	1	1	1
3	Enc3FFM	第三顺序系数加密运算	0001101	1	1	1
4	Enc4FFM	第四顺序系数加密运算	0001110	1	1	1
5	EncL1FFM	最后一轮第一顺序系数加密运算	0001111	1	1	1
6	EncL2FFM	最后一轮第二顺序系数加密运算	0010000	1	1	1
7	EncL3FFM	最后一轮第三顺序系数加密运算	0010001	1	1	1
8	EncL4FFM	最后一轮第四顺序系数加密运算	0010010	1	1	1
9	Dec1FFM	第一顺序系数解密运算	0010011	1	1	1
10	Dec2FFM	第二顺序系数解密运算	0010100	1	1	1
11	Dec3FFM	第三顺序系数解密运算	0010101	1	1	1
12	Dec4FFM	第四顺序系数解密运算	0010110	1	1	1
13	EncL1FFM	最后一轮第一顺序系数解密运算	0010111	1	1	1
14	EncL2FFM	最后一轮第二顺序系数解密运算	0011000	1	1	1
15	EncL3FFM	最后一轮第三顺序系数解密运算	0011001	1	1	1
16	EncL4FFM	最后一轮第四顺序系数解密运算	0011010	1	1	1

蜂鸟E203处理器核的存储器子系统结构如图3所示，通用访问指令首先通过地址产生单元（AGU）产生读和写指令的存储器访问地址，随后通过存储器访问控制模块（LSU）访问与自定义总线（ICB）相连的存储器。主处理器的LSU为协处理器提供了专用访存通道，与通用访存指令类似，专用访存指令也是将地址信息送到LSU并通过ICB访问存储器。两种方法不同之处是专用指令需要将主处理器分发的指令送到协处理器，协处理器需要进行译码后再进行访存操作，针对执行周期大于四个以上的指令，主处理器流水线会出现空泡。通过对访存过程对比后发现，一次读写多个数据相比一次读写一个数据而言，一条指令可以完成尽可能多的操作，但相应地增加了协处理器硬件复杂度和指令的执行周期，影响了主处理器的主频。因此，本文采用通用访存指令加载存储状态矩阵。

2.2 加速引擎设计

蜂鸟E203处理器将扩展指令通过NICE接口派发到加速引擎，加速引擎通过对扩展指令译码获取必要的配置信息，AES加速单元根据配置信息计算并将结果送至目的寄存器，加速引擎整体硬件结构图如图4所示。

蜂鸟E203处理器的NICE接口由请求通道、响应通道、存储器请求通道和存储器响应通道四个通道组成，主处理器通过NICE接口向加速引擎发起调用请求。NICE接口采用valid-ready机制实现同步握手，完成握手后，协处理器将会获得完整的扩展指令编码、源操作数1和源操作数2的值，当协处理器完成运算后，将会通过接口发送运算完成信号，握手时序如图5所示。

通过对扩展指令编码的func7字段分析，运算系数顺序编码、是否为最后一轮等编码信息和原操作数共同送到配置寄存器。配置寄存器的值只有在完成译码后才会更新，AES运算加速部件根据配置寄存器的值运算并将结果送到目的寄存器。

AES加速部件首先根据加密工作状态判断进行S盒或反S盒变换，传统的S盒查表方法[5]是通过计算索引地址来加载替换信息，该操作一般需要四条通用指令完成，而本文通过边计算边查表的方式极大地节省了指令数目，加快了运算速度。完成式（1）即字节替代操作后将得到的结果进行列混合运算，运算时将式（1）得到的字节与2、3、9、11、13和14相乘，根据加解密工作状态选择相应的乘数，随后依据系数顺序信息选择相应的运算结果，最后将运算的结果与密钥异或得到状态矩阵一列的运算结果，密码运算部件结构如图6所示。

3 FPGA实现及性能评估

3.1 仿真环境和FPGA验证环境搭建

本文设计的密码加速引擎需要编写软件代码，以自定义指令的形式调用密码加速引擎完成密码运算加速。首先使用IC前端仿真软件完成对密码扩展指令的实现和验证，通过仿真工具仿真处理器密码运算过程，完成协处理器的功能验证。最后在FPGA平台上进行实现并对协处理器资源占用、功耗、性能等做出评估，图7为FPGA验证环境。

为AES密码算法设计加密子密钥生成，解密子密钥生成、加密运算和解密运算四个函数，使用在C语言中内联汇编的方式进行编程，需要按照GCC编译器的规定将参数存储到约定的寄存器中。

如图8所示，软硬件协同工作的流程如下：以加密运算为例，密钥固定在程序中，待加密的明文通过串口输入。按照约定的格式从存储器中加载状态矩阵，加载操

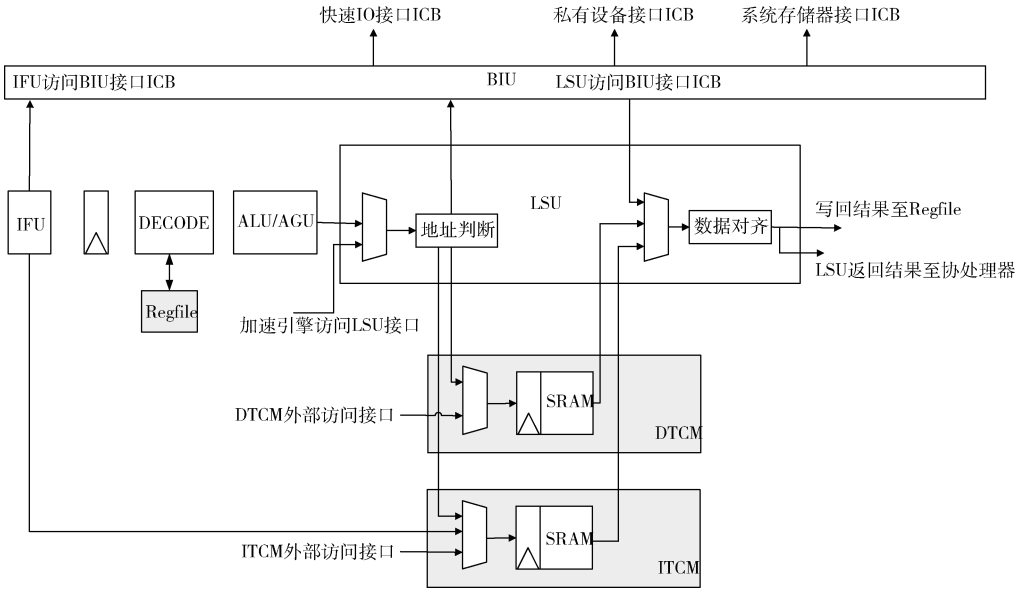

图3　蜂鸟E203处理器核存储器子系统结构示意

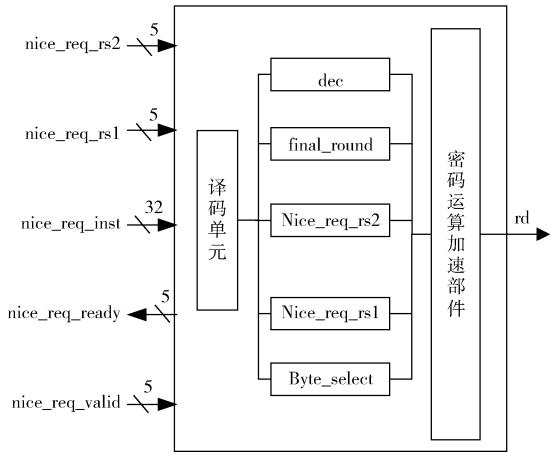

图 4　协处理器整体结构图

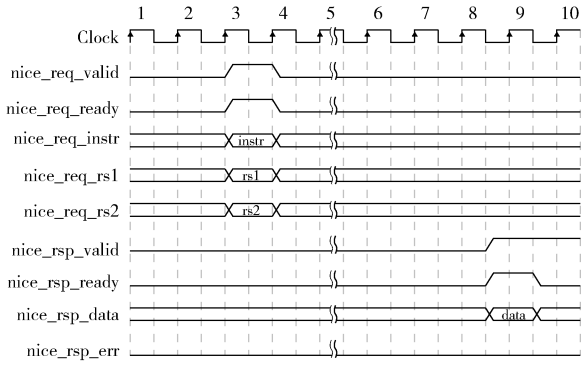

图 5　NICE 接口处理时序图

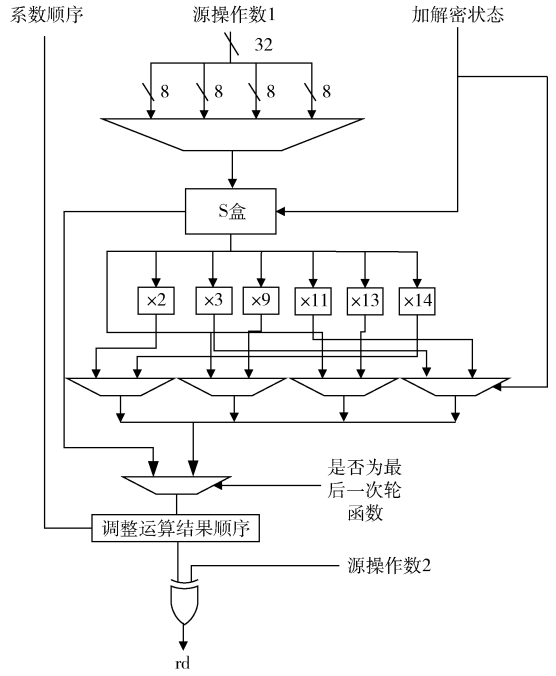

图 6　AES 加速部件结构图

作需 32 条指令完成。调用自定义密码运算指令完成 16 轮轮函数，一轮轮函数需要调用自定义运算指令 16 次，完成一次状态矩阵更新需要 160 条指令完成，因此，加

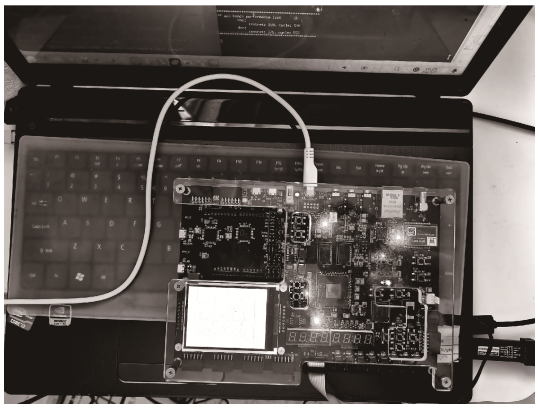

图 7　FPGA 验证环境图

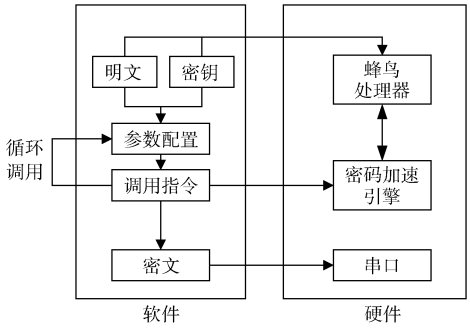

图 8　软硬件协同工作数据流图

密一次长度为 128 比特的明文需要不足 200 条指令，可见采用该方法效率较高。

3.2 测试结果分析

AES 算法程序的密钥为 0x0f1571c947d9e8590cb7add6af7f6798，单分组明文输入为 0x0123456789abcdeffedcba9876543210，通过仿真得到加密结果为 0xff0b844a0853bf7c6934ab4364148fb9，与标准向量一致。如图 9 所示，区域 A 为加密密钥扩展过程波形，区域 B 为加密过程波形，区域 C 为解密密钥扩展过程和解密过程波形。

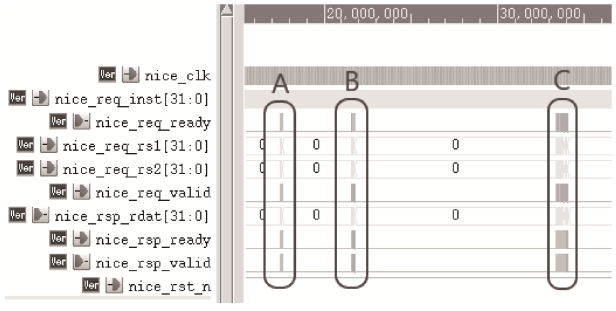

图 9　加速引擎波形图

如图 10 所示，通过执行 4 条自定义指令计算完成了状态矩阵其中一列运算结果。

根据表 2 所示运行结果可以看出，通过使用自定义指令，程序所需指令和周期数大幅减少。文献[7]中，同

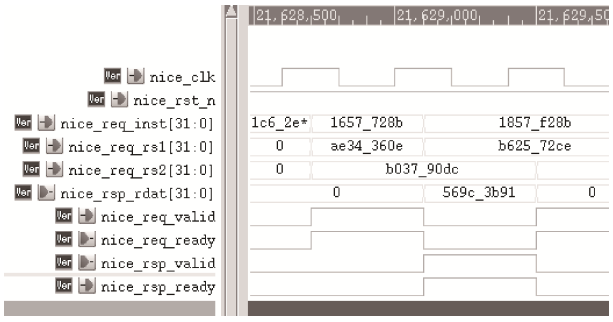

图10　完整加密过程

表2　软件运行情况

程序	不含扩展指令		含扩展指令	
	加密	解密	加密	解密
指令数	189 275	30 311	359	375
周期数	263 666	41 434	375	557

样以RISC-V指令集为基础，通过扩展多种密码指令实现了多种密码算法。与本文AES算法单分组明文加密性能作对比，如表3所示，本文提出的加速方案需要更少的周期数，仍然获得了较好的加密性能。

表3　单分组明文加密性能

实现方案	文献[7]	本文
指令数	385	359
周期数	1 973	375

本文使用的FPGA验证平台为采用Xilinx XC7A200T-2 FPGA芯片的FPGA开发板。对FPGA工作频率约束在100 MHz，经Vivado软件综合、布局布线后，内部资源利用情况如表4所示，实验结果证明本文提出的设计方案消耗资源较少，适用于硬件资源紧张的场景。

表4　FPGA资源使用情况

硬件资源	LUT	FF	RAMB36	LUTRAM
蜂鸟处理器及外设	10 936	11 011	32	16
协处理器	158	73	0	0
总使用量	11 094	11 084	32	16
协处理器占用率/%	1.4	0.66	0	0

4　结论

本文分析了AES轮函数运算过程，本着节省硬件资源和提高并行性的原则，设计了基于蜂鸟E203的AES加速引擎，运行对比了包含扩展指令的程序和不包含扩展指令的算法程序，通过对运行结果分析，本文实现的密码加速引擎取得了较好的加速效果。为了充分发挥蜂鸟处理器低功耗和可扩展的特性，下一步将进一步分析SM3密码算法，提高杂凑运算速度，进一步完善身份认证、安全启动等功能，以构建一个较为完善的安全SoC验证系统。

参考文献

[1] 孙涛. 人体健康数据实时监测系统的设计与实现[D]. 西安：西安电子科技大学, 2021.
[2] ALKHALIL A, RAMADAN R A. IoT data provenance implementation challenges[J]. Procedia Computer Science, 2017, 109：1134-1139.
[3] DAR K S, ALAM T M, HAMEED I A, et al. A review on security challenges in Internet of Things (IoT) [C]// 26th IEEE International Conference on Automation and Computing (ICAC'21), Portsmouth, UK, 2021.
[4] ANDERSON R, BOND M, CLULOW J, et al. Cryptographic processors-a survey[J]. Proceedings of the IEEE, 2006,94(2):357-369.
[5] 刘元锋. RISC架构微处理器扩展对称密码处理指令的研究[D]. 郑州：解放军信息工程大学, 2006.
[6] 季子豪. 战术物联网中计算卸载机制的研究[D]. 南京：南京邮电大学, 2021.
[7] 侯鹏飞. RISC-V处理器扩展专用密码指令研究与设计[D]. 郑州：战略支援部队信息工程大学, 2018.
[8] 李爱国, 冯国松. 基于MIPS处理器的AES算法指令集扩展方法与实现[J]. 微电子学与计算机, 2012,29(6):126-129.
[9] WANG W, HAN J, CHENG X, et al. An energy-efficient crypto-extension design for RISC-V[J]. Microelectronics Journal, 2021, 116：105165.
[10] 金晨辉, 郑浩然, 张少武, 等. 密码学[M]. 北京: 高等教育出版社, 2009.
[11] MARSHALL B, NEWELL G, PAGE D, et al. The design of scalar AES Instruction Set Extensions for RISC-V[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2020:109-136.
[12] 胡振波. 手把手教你设计CPU——RISC-V处理器篇[M]. 北京：人民邮电出版社, 2018.

（收稿日期：2022-05-31）

作者简介：

张晓磊（1992-），男，硕士研究生，主要研究方向：安全专用芯片设计。

戴紫彬（1966-），男，博士生导师，主要研究方向：信息安全、体系结构。

郭朋飞（1987-），男，博士，主要研究方向：安全专用芯片设计。

扫码下载电子文档

版权声明