一种高性能硬件加密引擎阵列架构

骆建军[①] 沈一凡[①] 周 迪[②] 冯春阳[①] 邓江峡*[①]

[①](杭州电子科技大学微电子研究中心 杭州 310018)

[②](浙江宇视科技有限公司 杭州 310051)

摘 要：该文提出一种高性能硬件加密引擎阵列架构，为大数据应用提供了先进的安全解决方案。该模块架构包括一个高速接口、一个中央管理和监视模块(CMMM)、一组多通道驱动加密引擎阵列，其中CMMM将任务分配给加密引擎，经由专用算法处理后再将数据传回主机。由于接口吞吐量和加密引擎阵列规模会限制模块性能，针对PCIe高速接口，采用MMC/eMMC总线连接构建阵列，发现更多加密引擎集成到系统后，模块性能将会得到提升。为验证该架构，使用55 nm制程工艺完成了一个PCIe Gen2×4接口的ASIC加密卡，测试结果显示其平均吞吐量高达419.23 MB。

关键词：专用集成电路；安全；加密；PCIe；eMMC

中图分类号：TN492; TN918.4 **文献标识码**：A **文章编号**：1009-5896(2021)12-3743-06

DOI: 10.11999/JEIT200855

High Performance Crypto Module with Array of Hardware Engines

LUO Jianjun[①] SHEN Yifan[①] ZHOU Di[②] FENG Chunyang[①] DENG Jiangxia[①]

[①](*Microelectronics Research Institute of Hangzhou Dianzi University*, *Hangzhou* 310018, *China*)

[②](*Uniview Research Institute*, *Hangzhou* 310051, *China*)

Abstract: A high-performance crypto module prescribed in this paper offers advanced security solutions in big data applications. A module architecture, which consists of a high throughput interface, Central Manage & Monitor Module (CMMM) and multiple channels driving a group of crypto engines, is discussed here. CMMM distributes the tasks to the crypto engines and guides the data back to the host after processing by the dedicated algorithm. Since the module's performance is limited by the interface throughput and the scale of the crypto engines, an array with MMC/eMMC bus connections is built for PCIe high-speed interfaces. The more crypto engines are integrated into a system, the higher performance of this system can reach. To verify this architecture, an ASIC encryption card with PCIe Gen2×4 interface is made under semiconductor manufacturing process technology of 55 nm, and tested. The average throughput of this card can achieve up to 419.23 MB.

Key words: ASIC; Security; Crypto; PCIe; eMMC

1 引言

随着越来越多的硬件设备连接到互联网中，物联网安全面临在扩展连接环境下安全威胁引发的复杂挑战[1]。执行具有最小延时的高计算力的安全认证是网络安全的保障[2,3]。为了增强服务器性能，可以使用更为强大的CPU/GPU[4]。然而，即使执行少量的安全负载，CPU的容量和效率也会因此而降低[5]。

本文介绍了一种替代解决方案，即一种具有高速、高吞吐量的可插拔式硬件加密卡，来应对安全需求，并将CPU从执行加解密算法(如RSA[6]、ECC[7]、AES[8]等)的任务中释放掉。这种类型的密码模块(设备)会帮助服务器在不降低CPU性能的情况下处理大量验证请求。同时，这个模块和CPU之间通过高速接口来交换数据。以PCIe接口[9]为例，目前流行的是具有16个Lane的PCIe Gen4的CPU/GPU，其传输速率可达到32 GB/s，假定物联网系统认证按4 kB大小的数据包进行处理，那么该模块可执行8×10^6次认证操作。

收稿日期：2020-10-04；改回日期：2021-09-30；网络出版：2021-10-25

*通信作者： 邓江峡 dengjiangxia@hdu.edu.cn

基金项目：国家基础科研项目(JCKY2018415C001)，浙江省固态硬盘和数据安全技术重点实验室(2015E10003)

Foundation Items: The National Basic Research Program (JCKY2018415C001), Zhejiang Key Laboratory Foundation of Solid State Drive and Data Security (2015E10003)

要设计一种安全算法计算能力与8×10^6 IOPS总线能力相匹配的PCIe卡其实是一个棘手的工程问题。虽然已有一些用FPGA实现的密码卡产品[10–12]，但这样的产品里面每个引擎都需要进行数学运算，而这些运算受到硬件资源和物理逻辑门参数限制。要提供充足的逻辑门就需要用到多个FPGA芯片，即使这样其性能仍然低于期望值。

本文提出了一种新的加密引擎阵列架构，其多个加密引擎可以并行计算。这一架构用更少的引脚或连接来定义加密引擎接口，以降低封装成本。同时，考虑到eMMC总线与SPI总线类似，采用eMMC总线构建加密引擎阵列，使其在两个维度上都可以提供更高性能。此外，本文以PCIe接口为原型，还描述了一种高性能加密ASIC设计来验证该方法，并分享工程经验。

2 模块架构

2.1 架构描述

图1是经过实践检验达到预期性能的加密模块架构，由一个高速接口、一个中央管理监控模块(Central Manage & Monitor Module, CMMM)和一组多通道驱动的加密引擎构成。

高速接口被连接到使用PCIe, SAS, SATA的主机总线上。片上算法IP阵列会运行多个算法引擎，而每个引擎都有一个数据包接口，该接口可用于机械硬盘、固态硬盘、SD卡、eMMC卡[13]、SPI接口[14]或SATA接口[15]。外加的数据缓冲区用于接收从主机来的命令和数据包，或返回给主机已处理过的数据包和相应的状态信息，以提高模块的性能和稳定性。

加密引擎阵列有N个并行的加密通道(CC_i, $0\le i<N$)，每个加密通道驱动M个加密块(CB_i,j, $0\le i<N, 0\le j<M$)，每个加密块有K个加密引擎($\mathrm{CE}_p, 0\le p<K$)。任务通过条带化操作被分配到这些通道上。此处的数据条带化的基本思想是把通道当作驱动硬盘的总线，使其工作在独立磁盘冗余阵列(Redundant Arrays of Independent Disks, RAID)的0级配置(RAID0模式)下[16,17]。加密块随使用的接口变化而变化，例如，图2是使用MMC接口的加密块(CB)结构。图3是加密引擎(CE)的结构图。

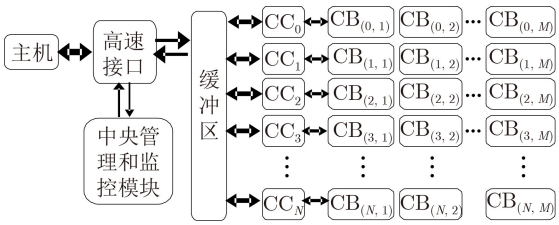

图 1 加密模块架构

加密引擎由通道接口、数据缓冲区和具有安全密钥寄存器的特定算法组成，如图3所示。

图2与图3的缓冲区非常重要，因为它实际上是一个由CMMM管理的队列。这就使得模块能连续地接收含有命令和数据包的任务而不管之前的任务是否完成。所有的任务都临时被压入队列，而CMMM将把任务分配给任何空闲着的加密引擎。根据应用程序要求，CMMM也可以按不同的优先级来分配这些任务。图4是CMMM管理任务流程示例，主机依次发送任务1、任务2、任务3，CMMM对这些任务动态进行调整，并分配给加密引擎。最终，主机先后收到任务2、任务3、任务1完成后的数据和状态信息。显然，缓冲越大，队列越长，模块可达到的性能也就越高。

2.2 总线分析

由于所有高速接口(如SATA, PCIe)的物理层

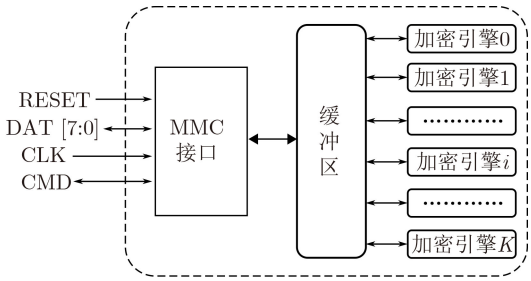

图 2 使用MMC接口的加密块

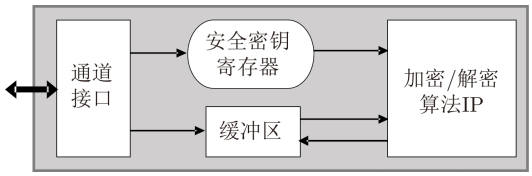

图 3 加密引擎结构

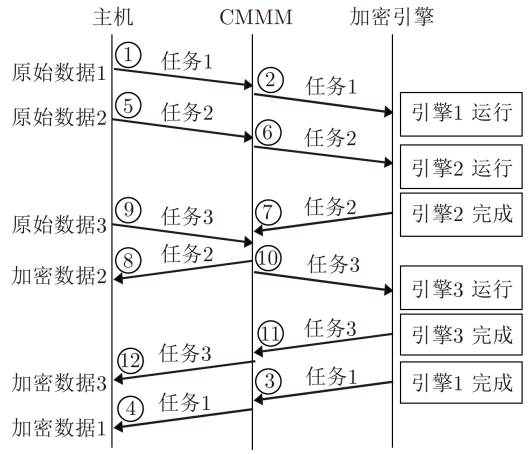

图 4 加密任务流程

使用差分信号进行传输，不能级联到阵列中，这也就意味着M只能为1，限制了该架构灵活地扩展到更大规模。因此，可以采用SPI总线或MMC/eMMC总线进行数据传输。

SPI总线可以通过添加片选信号(CS)实现级联来构建一个2维加密阵列，但是通常的SPI总线的最大数据传输速率为1.2 MB/s，假设用4 kB大小的IO测试，其远小于300 IOPS，不能为认证过程提供足够的吞吐量。

MMC/eMMC具有与SPI总线类似的总线结构，其总线宽度还可以扩展到4位，甚至8位，其定义的一种机制可以通过某个部件号来选择或激活模块(卡)，并且基本的MMC/eMMC模块可提供至少为25 MB/s的数据传输速率，最新的eMMC5.1可提供高达400 MB/s的数据传输速率，接近SATA III和USB3.0的性能。

鉴于以上分析，本文采用MMC/eMMC总线进行数据传输。MMC/eMMC设备通过可以配置数量的数据信号总线来传输数据。这些通信信号描述如下：

CLK：从主机输出的时钟信号，每个信号周期下命令总线上传输1 bit，所有数据总线上传输1 bit或2 bit。时钟频率可以在零到最大时钟频率之间变化。

CMD：双向命令通道，用于设备初始化和命令传输。

DAT0-DAT7：双向数据传输信号，在推挽模式下工作，同一时刻进行单向传输(主机或设备驱动该信号)。在上电或复位后，默认只有DAT0用于传输数据。

MMC/eMMC总线的一个特性是在CMD线上采用“线与”机制时，可使某个设备在总线仲裁程序设置的环境中作为一个多终端设备运行。主机在开漏模式下开始设备识别过程，CMD线上的开漏驱动级允许在设备识别过程中进行并行的设备操作。驱动多台设备的MMC/eMMC架构如图5和图6所示。

总线被激活后，主机将给设备发送有效运行条件 (CMD1)，对于CMD1的响应就是“已连线”和对系统中所有设备进行环境限制操作。将不兼容的设备设置成“Inactive State”。然后主机发送广播命令ALL_SEND_CID(CMD2)，要求所有设备发送其唯一的设备标识码(Device Identification Number, CID)。所有处于“Ready State”的设备在CMD线上同时相继发送其CID，只要这些设备发出CID的位与在任何一个“位”周期内相应的位

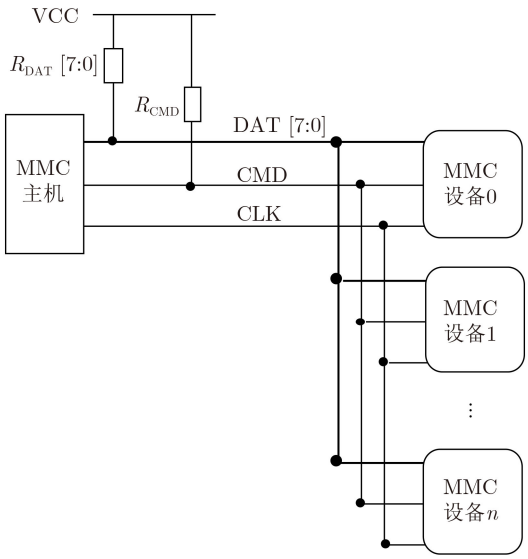

图 5 一个MMC主机驱动多个MMC设备

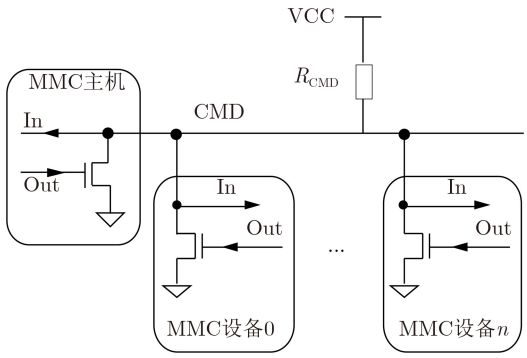

图 6 MMC设备在CMD信号上“线与”

有不匹配时，这些设备将不会被选上，还会立即停止发送其CID，保持“Ready State”，等待下一次识别周期。因为每个设备都有唯一的CID，所以最终只有一个设备给主机成功发送其完整的CID，这个设备之后进入“Identification State”状态。然后，主机发送SET_RELATIVE_ADDR(CMD3)，为这个设备分配一个关联设备地址(Relative Device Address, RCA)，用于数据传输。一旦RCA被接收，Device进入“Stand-by State”，由开漏模式变为推挽模式，不再对之后的身份识别周期做出反应。主机会重复设备识别过程(CMD2, CMD3循环)，直到所有设备被识别，不再有设备响应。识别过程超时的条件是发送CMD2后超过一段时间(eMMC规范中定义的NID时钟周期)没有收到响应。之后，主机可以使用CID激活任意设备，并强制其他设备在总线上保持安静。

3　性能分析

如果一个加密模块共有$M \times N \times K$个加密引擎，那么它能同时处理$M \times N \times K$个认证请求。

比如每个加密引擎每秒可以运行R次AES或SHA算法，那么这个模块可以每秒执行$M \times N \times K \times R$次AES算法，也就是说它的最大计算能力$\mathrm{CA}_{\max}$可以表示成：

$$\mathrm{CA}_{\max} = M \times N \times K \times R \tag{1}$$

但这不是最终测试到的性能，因为加密模块与主机间的接口吞吐量很可能是性能瓶颈。SATA接口的最大数据传输率$\mathrm{TS}_{\max}$=750 MB/s，PCIe Gen3×4的$\mathrm{TS}_{\max}$=4 GB/s。模块的最终性能由$\{\mathrm{CA}_{\max}, \mathrm{TS}_{\max}\}$的最小值决定。

为了分析模块性能和时延，定义以下几个变量：

T_{c}：加密引擎处理一个命令数据包的时间。

T_{t}：处理器的本地缓冲区和模块中央缓冲区之间的数据传输时间。

T_{i}：通过通信接口接收数据的时间。

T_{w}：数据包在IO队列缓冲的时间。

模块中的加密引擎越多，电路可以提供的性能越高。假设原始数据被分割为P个小数据包，每个数据包的大小为4 kB。加密或解密一个数据包需要T_{c}，共有N个并行运行的加密通道，每个通道可运行M个加密块。那么电路每秒可以执行$N \cdot M/(T_{\mathrm{c}} + T_{\mathrm{t}})$次加密/解密操作，在此忽略了给模块分配任务的时间。那么，IOPS可表示成

$$\mathrm{IOPS} = N \times M/(T_{\mathrm{c}} + T_{\mathrm{t}}) \tag{2}$$

加密/解密处理器的延迟由多个参数决定。比如，如果有一个或多个空闲的加密引擎，此时数据包以最小的延迟通过中央缓冲区被加密引擎处理[18]，T_{w}可以取最小值并忽略不计。如果所有的加密引擎繁忙且IO队列已满，接收到的数据包在中央缓冲区处于等待状态，直到有一个加密引擎被释放到空闲状态，T_{w}具有最大值。T_{w}的计算公式为

$$T_{\mathrm{w}} = (T_{\mathrm{c}} + T_{\mathrm{t}}) \left[\frac{C}{N \times M} + 1 \right] \tag{3}$$

其中，C是在IO队列里等待的命令量。

总时延T_{L}的计算公式为

$$\begin{aligned} T_{\mathrm{L}} &= T_{\mathrm{i}} \times N \times M + T_{\mathrm{w}} \\ &= \mathrm{T_i} \times N \times M + (T_{\mathrm{c}} + T_{\mathrm{t}}) \left[\frac{C}{N \times M} + 1 \right] \end{aligned} \tag{4}$$

假设缓冲区的最大容量$C_{\max}$=4000，T_{i}=1 ms，T_{c}=5 ms，T_{t}=1 ms，当N=1, 2, 4, 8, M=1, 8时，理论计算得到的最大总时延T_{L}如表1所示。

缓冲区的容量，即IO队列的最大深度，必须满足队列请求的数量大于加密引擎的数量，否则一些加密引擎将会被闲置，模块整体性能无法达到最高。另外，如果IO队列深度配置得过大，响应的时延就会增加。因此，IO队列的深度应该有一个极值。一般来说，队列深度可以等于密码引擎数量，并且每个队列请求在缓冲区可占用1～4 kB空间。也就是说在时延可以接受的情况下，队列深度越大越好。

4 测试结果

为验证该架构，本文完成了一个PCIe Gen2×4接口的ASIC加密模块。使用55 nm制程工艺，最终制备出尺寸为4.8 mm×4.7 mm的芯片，时钟频率高达500 MHz。

该模块共有8个通道(N=8)，每个通道驱动8个eMMC模块(M=8)，每个模块有16个加密引擎(K=16)，单个加密引擎每秒可以执行133次SHA-256加密操作(R=133)，并且IO大小为4 kB。测试结果见表2，表明其平均吞吐率等于419.23 MB/s。

目前已报道了具有高性能密码设计的FPGA解决方案。比如文献[19]中由3块Xilinx Virtex FPGA芯片实现的模块，其SHA-256吞吐率为47.75 MB/s。而本文ASIC架构其吞吐量是它的8～9倍。

PCIe Gen4的速度比Gen2的快4倍，如果使用PCIe Gen4×8接口，理论上，该模块的IOPS可达到容量为100k×4×2=800k，每个加密通道可提供100k的IOPS吞吐量，每个eMMC加密块的IOPS应大于12.5k。实际上，eMMC总线频率高达200 MHz，这可以支持运行最大为25k的IOPS。这意味着模块可以在1 s内用SHA-256算法处理800k次认证请求。即使面向更高性能需求，$(M, N) = (8, 8)$ 甚至 $(M, N) = (8, 4)$的配置仍然能满足这一需求。

本文实现的ASIC加密模块中，IO队列深度被设置为128，由于IO队列深度提供了足够的等待缓冲区，所以所有加密引擎都有足够的时间运行相应

表 1 不同情况下的最大总时延(ms)

	N=1	N=2	N=4	N=8
M=1	24007	12008	6010	3014
M=8	3014	1522	788	445

表 2 性能测试

	#1	#2	#3	#4
连续读(MB/s)	1105.00	1102.00	1103.00	1103.00
连续写(MB/s)	912.60	912.10	912.00	912.20
随机读(k-IOPS)	50.85	84.98	82.83	85.23
随机写(k-IOPS)	105.00	104.75	104.75	104.73
吞吐率(MB/s)	**420.00**	**419.00**	**419.00**	**428.92**

的算法。如果IO队列深度过小，IOPS将会下降到1.3k左右。PCIe Gen3×4的数据传输速率高达4 GB/s。其发送和接收数据的时延T_i=4 kB/(4 GB/s)=1 μs。

考虑到大多数情况下数据传输率较为稳定，在相同硬件配置的情况下，T_i不会有太大不同。因此，时延主要由在IO队列中的等待时间决定。测试结果如表3所示。

表 3　随机读写的时延(μs)

	平均时延	最大时延
随机读(4 kB)	53	1894
随机写(4 kB)	24	1039

5　结论

本文描述了一种高性能的支持并行运算加密引擎阵列的PCIe接口加密卡，其平均吞吐率高达419.23 MB/s。此外，模块集成的加密引擎越多，模块的性能就越好。下一步的工作目标是采用最新的PCIe接口、更多的PCIe物理通道以及更强大的加密算法引擎，为物联网安全系统提供更高性能的加密卡。

参 考 文 献

[1] SEZER S. T1C: IoT Security: -Threats, security challenges and IoT security research and technology trends[C]. Proceedings of 2018 31st IEEE International System-on-Chip Conference, Arlington, USA, 2018: 1–2. doi: 10.1109/SOCC.2018.8618571.

[2] WAZID M, DAS A K, ODELU V, *et al*. Secure remote user authenticated key establishment protocol for smart home environment[J]. *IEEE Transactions on Dependable and Secure Computing*, 2020, 17(2): 391–406. doi: 10.1109/TDSC.2017.2764083.

[3] 闫宏强, 王琳杰. 物联网中认证技术研究[J]. 通信学报, 2020, 41(7): 213–222. doi: 10.11959/j.issn.1000-436x.2020131.
YAN Hongqiang and WANG Linjie. Research of authentication techniques for the Internet of things[J]. *Journal on Communications*, 2020, 41(7): 213–222. doi: 10.11959/j.issn.1000-436x.2020131.

[4] 纪兆轩, 杨秩, 孙瑜, 等. 大数据环境下SHA1的GPU高速实现[J]. 信息网络安全, 2020, 20(2): 75–82. doi: 10.3969/j.issn.1671-1122.2020.02.010.
JI Zhaoxuan, YANG Zhi, SUN Yu, *et al*. GPU high speed implementation of SHA1 in big data environment[J]. *Netinfo Security*, 2020, 20(2): 75–82. doi: 10.3969/j.issn.1671-1122.2020.02.010.

[5] 孙婷婷, 黄皓, 王嘉伦, 等. 面向CPU-GPU异构系统的数据分析负载均衡策略[J]. 计算机工程与科学, 2019, 41(3): 417–423. doi: 10.3969/j.issn.1007-130X.2019.03.005.
SUN Tingting, HUANG Hao, WANG Jialun, *et al*. A load balancing strategy on heterogeneous CPU-GPU data analytic systems[J]. *Computer Engineering and Science*, 2019, 41(3): 417–423. doi: 10.3969/j.issn.1007-130X.2019.03.005.

[6] MENEZES A J, VAN OORSCHOT P C, and VANSTONE S A. Handbook of Applied Cryptography[M]. Boca Raton: CRC Press, 1996: 433–446.

[7] HANKERSON D, MENEZES A J, and VANSTONE S. Guide to Elliptic Curve Cryptography[M]. New York: Springer Science & Business Media, 2004: 6–14.

[8] Federal Information Processing Standards Publication 197. Advanced encryption standard (AES)[S]. 2001.

[9] BUDRUK R, ANDERSON D, and SHANLEY T. PCI Express System Architecture[M]. Boston: Addison-Wesley Professional, 2004: 9–11.

[10] 刘金峒, 梁科, 王锦, 等. SM4加密算法可裁剪式结构设计与硬件实现[J]. 南开大学学报:自然科学版, 2019, 52(4): 41–45.
LIU Jintong, LIANG Ke, WANG Jin, *et al*. Cuttable structure design and hardware implementation of SM4 encryption algorithm[J]. *Acta Scientiarum Naturalium Universitatis Nankaiensis:Natural Science Edition*, 2019, 52(4): 41–45.

[11] SUHAILI S B and WATANABE T. Design of high-throughput SHA-256 hash function based on FPGA[C]. Proceedings of the 6th International Conference on Electrical Engineering and Informatics, Langkawi, Malaysia, 2017: 1–6. doi: 10.1109/ICEEI.2017.8312449.

[12] 赵军, 曾学文, 郭志川. 支持国产密码算法的高速PCIe密码卡的设计与实现[J]. 电子与信息学报, 2019, 41(10): 2402–2408. doi: 10.11999/JEIT190003.
ZHAO Jun, ZENG Xuewen, and GUO Zhichuan. Design and implementation of high speed PCIe cipher card supporting GM algorithms[J]. *Journal of Electronics & Information Technology*, 2019, 41(10): 2402–2408. doi: 10.11999/JEIT190003.

[13] JEDEC. JESD 84-B50 Embedded multi-media card (*e*·MMC) electrical standard (5.0)[S]. Arlington: JEDEC Solid State Technology Association, 2013.

[14] Motorola, Inc. SPI block guide V03.06[S]. Motorola Inc. , 2001.

[15] Serial ATA International Organization. Serial ATA revision 3.0[S]. Serial ATA International Organization, 2009.

[16] PATTERSON D A, GIBSON G, and KATZ R H. A case for redundant arrays of inexpensive disks (RAID)[C]. Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data, Chicago, USA, 1988: 109–116. doi: 10.1145/50202.50214.

[17] CHANG Lipin and KUO T W. An adaptive striping architecture for flash memory storage systems of embedded systems[C]. Proceedings of the Eighth IEEE Real-Time and Embedded Technology and Applications Symposium, San Jose, USA, 2002: 187–196. doi: 10.1109/RTTAS.2002.1137393.

[18] REDDY A K, PARAMASIVAM P, and VEMULA P B. Mobile secure data protection using eMMC RPMB partition[C]. Proceedings of 2015 International Conference on Computing and Network Communications, Trivandrum, India, 2015: 946–950. doi: 10.1109/CoCoNet.2015.7411305.

[19] GREMBOWSKI T, LIEN R, GAJ K, *et al*. Comparative analysis of the hardware implementations of hash functions SHA-1 and SHA-512[C]. Proceedings of the 5th International Conference on Information Security, Sao Paulo, Brazil, 2002: 75–89. doi: 10.1007/3-540-45811-5_6.

骆建军：男，1970年生，教授，博士生导师，研究方向为集成电路、数字存储和数据安全系统.

沈一凡：女，1996年生，硕士，研究方向为集成电路.

周 迪：男，1975年生，正高级工程师，研究方向为物联网、视频安全.

冯春阳：男，1966年生，高级工程师，研究方向为集成电路设计、功率半导体.

邓江峡：女，1983年生，副教授、硕士生导师，研究方向为自旋电子学、集成电路设计与验证.

责任编辑：陈 倩