■联合项目　　□重点项目

□一般项目

2024 年湖南省研究生科研创新项目

申　　请　　书

项目名称：	轻量级分组密码的软硬件优化研究与实现
主 持 人：	向嘉豪
所在单位：	衡阳师范学院
联合企业：	湖南省网安基地科技有限公司
所属学科/专业类别：	电子信息
联系电话：	13087286239
电子信箱：	simple.xjh@qq.com
申请日期：	2024 年 6 月 8 日

湖　南　省　教　育　厅

2024 年

一、简表

项目名称	轻量级分组密码的软硬件优化研究与实现			研究类别	基础研究	
					应用研究	√
					试验发展	
研究年限	2024 年至 2026 年			申请经费（万元）	1	
项目主持人	姓名	向嘉豪	性别	男	身份证号	430525200011025117
	在读层次	硕士	学科专业及研究方向	电子信息及嵌入式计算与信息安全研究方向	在读学年	一年级
指导教师	姓名	李浪	性别	男	学历学位	研究生、博士
	技术职称	教授	学科专业及研究方向	计算机专业		嵌入式计算与信息安全
	目前指导学生数	博士：　　名，硕士：23 名		联系电话	15873438955	
主要研究人员	姓　名	身份证号码	技术职务	专业	所在单位	本人签名
	邓联瑞	430503200101071514	无	电子信息	衡阳师范学院	
	夏生成	43042219980524379X	无	电子信息	衡阳师范学院	
	岳兴起	430502200202175519	无	网络空间安全	衡阳师范学院	
项目负责人主要学习和工作经历（从上大学开始）	2017.9~2021.6 项目负责人在长沙学院学习本科专业。 2023.9~2026.6 项目负责人在衡阳师范学院学习硕士专业。 2023.9~至今 加入嵌入式计算与信息安全实验室，从事轻量级密码算法软硬件优化实现研究。					

二、立项依据

（项目的研究目的、意义；国内外研究现状分析和发展趋势；项目应用前景和学术价值；能为行业企业解决的实际问题；现有研究基础、条件、手段以及指导教师情况等）

1. 项目的研究目的、意义

1. 1. 项目的研究目的

随着基础通信设施的不断完善，如5G网络的普及，人们对通信的需求逐渐增加。这种需求的增长催生了大量基于通信的应用，如工业互联网、车联网、物联网等。然而，随着这些网络中传输的数据量的增加，安全问题变得越来越严重。这主要是由网络数据中数据包传输机制引起的。幸运的是，对称加密算法提供了一种解决方案，只需双方协商好密钥，就能确保数据在公共信道下的安全传输，而无需调整传输机制。

对称加密算法中，最广泛使用的是2001年的AES国际算法。它在WEB、WIFI等领域，以及服务器与个人计算机上都得到了广泛的应用。然而，为了避免AES算法存在的未知门陷，我国在2006年提出了SM4对称加密算法来替代AES算法。值得一提的是，SM4在2021年正式成为ISO标准并得到了国际认可。这些传统的对称算法在计算资源充足的场景下，能够提供较高的安全性。但在计算资源受限的场景下，由于这些算法的计算复杂度较高，因此需要一种更加轻量级的对称加密算法来满足这种场景的需求。

美国国家标准局NIST于2019年启动了轻量级密码算法LWC的征召。在56个轻量级密码算法的竞争中，ASCON[1] 算法经过三轮筛选，最终脱颖而出。到了2023年，ASCON算法已经成为NIST的轻量级加密标准。由于ASCON算法在资源受限的场景下表现出较高的性能，因此在物联网、车联网等场景中具有广泛的应用前景。

相较而言，我国在轻量级加密算法上的起步较晚，暂时还没有自己的轻量级加密算法标准。然而，在2019年由中国密码学会组织的全国密码算法设计竞赛中，一等奖获得者是一种名为uBlock[2]的轻量级加密算法。这种算法在计算资源充足的环境下表现优秀，同时在资源受限的场景下，也展现出了高性能。尽管如此，uBlock算法并未像ASCON算法那样得到广泛的学者关注，这使得我国在轻量级加密算法上与国际顶尖水平存在较大的差距。

这种差距不仅体现在轻量级加密算法的设计上，也同样存在于其实现上。这包括硬件和软件实现。具体来说，需要考虑如何高效设计硬件加密的IP核，并将其集成入芯片，以实现硬件级别的安全。同时，也需要考虑如何将加密算法高效实现于8-bit或32-bit的微控制器中，以确保应用的软件级别安全。我国在这个领域的研究相对较少。因此，本项目的目标是研究轻量级加密算法的实现，以推动我国在轻量级加密算法领域

的研究进展。

1.2. 项目的研究意义

在上世纪的算法设计中，对称加密算法的设计主要考虑了安全性。然而，这些设计往往较少考虑其在资源受限的场景下的性能。例如，DES[3]的实现部分提及其在软件与硬件的实现，但并未给出具体的参考实现方案与实现性能。

在 NIST 的 LWC 竞赛中，轻量级加密算法的设计不仅考虑了安全性，还考虑了其在资源受限的场景下的实现性能。这无疑对算法的设计提出了更高的要求。实现算法设计与实现性能之间的桥梁，是本项目研究的出发点。在考虑最新的软硬件平台技术下，将加密算法的组件转化为相应的电路或程序，是本项目研究的手段。更具体来说，本项目的研究意义体现在以下几个方面：

（1）研究轻量级加密算法在专用集成电路（ASCI）和现场可编程门阵列（FPGA）上的硬件实现。这将提高算法实现的性能，同时确保其硬件级别的安全。

（2）研究轻量级加密算法在 8-bit 或 32-bit 的微控制器上的软件实现。这将提高算法实现的性能，同时确保其软件级别的安全。

（3）研究轻量级加密算法的软硬件协同实现。在确保算法实现的灵活性的前提下，最大程度地提高算法实现的性能。

2. 国内外研究现状分析和发展趋势

网络数据传输量的增加使得网络安全问题日益严重。对称加密算法是一种能够确保数据在公共信道下安全传输的解决方案。然而，传统的对称加密算法在计算资源受限的场景下计算复杂度较高。因此，需要一种轻量级的对称加密算法来满足这种场景的需求。

在 2011 年之前，轻量级加密算法的研究主要集中在硬件实现上。这是因为硬件实现能够提供更高的性能，如 ISO 的轻量级加密标准 PRESENT[4]。同时，硬件实现也能确保算法的安全性。然而，随着应用场景的增加，性能需求也在变化。现在，不仅关注硬件的电路面积，也开始关注软件加密的执行时间，如美国国家安全局（NSA）2013 年提出的 SIMON[5]。此外，还关注硬件加密的功耗，如 2015 年亚密会提出的 Midori[6]，以及硬件加密的时延，如 2016 年美密会提出的 SKINNY[7]。因此，轻量级加密算法的内涵正在不断扩展。

2.1. 轻量级加密算法硬件实现

轻量级加密算法的设计和实现都在不断发展和完善。早期的轻量级加密算法主要集中在硬件实现上。例如，Arich 等人对 DES 算法的硬件实现[8]，他们将 DES 的加密

速率提高到了 1 Gbits/s。随着硬件实现的进步，性能指标也从电路面积和加密吞吐量扩展到了功耗和时延等方面。为了实现这些性能指标，提出了一些全新的硬件实现技术。其中，串行、展开、迭代、流水线等技术得到了广泛应用。

串行实现的核心思想是降低加密时的数据带宽。这种方法可以实现对加密组件的重复使用，并减少电路中的寄存器使用数量，以实现更低的电路面积。Priya 等人提出了 AES 算法的串行实现 [9]。他们在 Xilinx Spartan-II FPGA （XC2S15）上实现了面积资源 174 slices 和两个内存块。在这种实现中，最小的数据带宽为一个字节。为了进一步降低数据带宽，Leong 等人提供了 IDEA 算法的比特级数据带宽的串行实现 [10] 。然而，对于以 S 盒作为基本加密组件的加密算法，实现一比特级的数据带宽是困难的。为了解决这个问题，Jean 等人为类 SPN 结构的加密算法提出了比特滑动技术 [11] 。这种技术可以实现比特级的数据带宽。

展开实现的核心思想是在同一周期内计算加密算法的多轮函数。这种方法可以降低加密所需的延迟，并在引入更多面积的情况下获取更高的吞吐量。Zodpe 等人在 AES 算法竞选的最终轮中使用了展开实现 [12]。他们的 2 轮展开实现 Rijndael 算法实现了最高的吞吐量，这在对比基于轮的迭代实现时尤为明显。Gupta 等人将展开实现与流水线结合 [13] ，这极大地提高了 RC4 [14] 算法的吞吐量。

迭代实现的核心思想是在同一周期内运算加密算法的一轮函数。通过反馈机制，可以实现多轮函数的计算。这种方法可以减少加密所需的面积，但会增加加密所需的延迟。值得注意的是，迭代实现在面积与吞吐量之间实现了较好的平衡。在 AES 算法竞选的最终轮中, Zodpe 等人的迭代实现的 Rijndael 获得了最高的吞吐量与面积比 。类似的情况也出现在 LWC 竞赛中。ASCON 算法的迭代实现在面积与吞吐量上取得了较好的平衡。

流水线实现的核心思想是将加密算法的一轮函数分解为多个阶段。这样，每个阶段的计算可以并行进行，实现多个分组同时加密。这种方法可以显著提高加密所需的吞吐量，但会增加实现的面积。Kryjak 等人运用了这种流水线技术 [15]。他们对 CLEFIA 算法[16]进行了实现，极大地提高了加密的吞吐量。然而，考虑到所需的资源，这种实现主要适用于超性能计算场景。

总的来说，对于传统的性能指标，如面积和吞吐量，轻量级加密算法的硬件实现已经有了较好的解决方案。然而，对于新的性能指标，如功耗和时延，还需要进一步的研究。未来的研究重点是如何提出新的硬件实现技术，以适应这些新的性能指标。

2.2.轻量级加密算法软件实现

轻量级加密算法的软件实现主要集中在微控制器上。这与硬件实现有所不同。微控制器提供了更高的灵活性。例如，可以在不同的应用场景下调整加密算法的参数，或者使用不同的加密算法。同时，微控制器也提供了更高的可移植性。例如，可以在不同的硬件平台上运行加密算法。然而，相比硬件实现，软件实现的性能指标通常较低。

早期的轻量级加密算法的软件实现主要集中在8-bit微控制器上。这些微控制器常用于RFID和传感器等设备。在1991年，Merkle尝试将Khufu算法进行软件实现。这为算法在其他不同场景下的应用打开了新的可能性[17]。后来，Osvik将AES算法实现在8-bit的AVR微控制器上，刷新了当时的最快加密记录[18]。

随着微控制器的发展，32-bit微控制器成为主流。软件实现的侧重也发生了改变。Rogaway开始在32-bit的现代微控制器上，对SEAL算法进行了软件实现[19]。Bertoni首次将AES算法实现在32-bit的微控制器上，并通过多种架构的仿真器对优化实现进行验证[20]。Schwabe在结合ARM平台在Cortex-M3和M4上实现了AES算法，扩展了AES算法的用途[21]。

在软件实现中，存在一些特殊的情况需要注意。例如，抵抗侧信道攻击通常需要添加额外的操作。如何更高效地实现这些抗攻击的操作，成为了一个重要的研究方向。例如，Rivain提出了一种高阶掩码技术[22]。这种技术可以有效地抵抗设备在侧信道攻击中的相关功耗攻击。另一方面，轻量级加密算法在某些场景下具有优势。例如，在区块链等对加密算法吞吐量要求较高的场景下，轻量级加密算法可以发挥重要作用。利用SIMD指令集可以提高加密算法的吞吐量[23]。此外，也可以利用GPU来实现加密。这些都是软件实现的重要方向。

总的来说，软件实现的技术比硬件实现更为丰富。因此，许多研究都集中在如何将加密算法高效地实现在更先进的平台上。未来的研究重点将是开发新的软件实现技术，以适应这些新的平台。同时，也需要考虑如何预防新的攻击方式。

2.3 轻量级加密算法软硬件协同实现

相比硬件实现和软件实现，软硬件协同实现的研究相对较少。这主要是因为软硬件协同实现需要同时考虑硬件和软件的特性。然而，软硬件协同实现在保证算法性能的同时，也能提高算法的灵活性。例如，当加密算法的标准发生变化时，软硬件协同实现能够更快地适应这些变化。值得注意的是，与硬件实现和软件实现相比，软硬件协同实现起步较晚。

在早期的实现中，加密算法的硬件实现和软件实现通常是分开的。例如，加密算法

作为 IP 核心，以片上外设的形式挂载在数据总线上，如 Usselmann 实现的 AES 算法。由于外设与 CPU 之间的通信开销较大，这种实现方式的性能较低。为了提高性能，一些研究开始尝试将加密算法的硬件实现与软件实现集成在一起。这种实现方式可以减少外设与 CPU 之间的通信开销，从而提高性能。然而，由于各家厂商的 CPU 架构不同且闭源，这种实现方式的研究较少。

开源硬件的发展为软硬件协同实现的研究带来了新的机遇。例如，RISC-V 是一种开源的 CPU 架构，可以自由使用。这种架构的出现，为软硬件协同实现提供了新的研究方向。Marshall 设计了一套轻量级扩展指令集，将 ChaCha 算法实现在 RISC-V 上[24]。基于扩展指令集，Chen 在 2023 年提出了一组通用扩展指令集在 RISC-V 之上，并将其运用到了 LWC 最终轮的 10 个算法上[25]。这种方式将加密的硬件电路集成入了 CPU 的执行流水线中，保证了加密操作的速度。同时，加密算法中的通用操作被设计为对应的指令，这样即使加密标准更新，也能保持对加密算法的加速效果。

总的来说，软硬件协同实现为轻量级加密算法的实现开辟了新的研究方向。这种实现方式将不利于软件实现的部分转化为硬件实现，并以此作为基础操作。然后，利用软件指令来构建整体算法，从而在加密算法的性能与灵活性之间取得平衡。这也为算法设计提供了一条全新的思路。未来的研究重点是将轻量级加密算法中的通用操作设计为扩展指令集或通用加密 IP 核。

参考文献

[1] Dobraunig C, Eichlseder M, Mendel F, et al. Ascon v1. 2: Lightweight authenticated encryption and hashing[J]. Journal of Cryptology, 2021, 34: 1-42.

[2] Wu W L, Zhang L, Zheng Y F, et al. The block cipher uBlock[J]. Journal of Cryptologic Research, 2019, 6(6): 690-703.

[3] Alahdal A, Deshmukh N K. A systematic technical survey of lightweight cryptography on IoT environment[J]. International Journal of Scientific & Technology Research, 2020, 9(3): 1-30.

[4] Sravya G, Kumar M O V P, Sheeba G M, et al. Hardware lightweight design of PRESENT block cipher[J]. Materials Today: Proceedings, 2020, 33: 4880-4886.

[5] Rashidi B. High‐throughput and flexible ASIC implementations of SIMON and SPECK lightweight block ciphers[J]. International journal of circuit theory and applications, 2019, 47(8): 1254-1268.

[6] Li Y, Wang M, Ou H, et al. Improved integral analysis on lightweight block cipher Midori[C]. 2019 IEEE 5th International Conference on Computer and Communications (ICCC). IEEE, 2019: 1494-1498.

[7] Sevin A, Mohammed A A O. A survey on software implementation of lightweight block ciphers for IoT devices[J]. Journal of Ambient Intelligence and Humanized Computing, 2023, 14(3): 1801-1815.

[8] Zeebaree S R. DES encryption and decryption algorithm implementation based on FPGA[J]. Indones. J. Electr. Eng. Comput, 2020, 18(2): 774-781.

[9] Priya S S S, Karthigaikumar P, Teja N R. FPGA implementation of AES algorithm for high speed applications[J]. Analog Integrated Circuits and Signal Processing, 2022: 1-11.

[10] Ledda M K C, Gerardo B D, Hernandez A A. Enhancing IDEA algorithm using circular shift and middle square method[C]. 2019 17th International Conference on ICT and Knowledge Engineering (ICT&KE). IEEE, 2019: 1-6.

[11] Roldán Lombardía S, Balli F, Banik S. Six shades lighter: a bit-serial implementation of the AES family[J]. Journal of Cryptographic Engineering, 2021, 11(4): 417-439.

[12] Zodpe H, Sapkal A. An efficient AES implementation using FPGA with enhanced security features[J]. Journal of King Saud University-Engineering Sciences, 2020, 32(2): 115-122.

[13] Jiao L, Hao Y, Feng D. Stream cipher designs: a review[J]. Science China Information Sciences, 2020, 63(3): 131101.

[14] Noura H, Chehab A. An efficient and secure variant of RC4 stream cipher scheme for emerging networks[C]. 2019 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2019: 1-8.

[15] Rashidi B. Efficient and flexible hardware structures of the 128 bit CLEFIA block cipher[J]. IET Computers & Digital Techniques, 2020, 14(2): 69-79.

[16] Jangra M, Singh B. Performance analysis of CLEFIA and PRESENT lightweight block ciphers[J]. Journal of Discrete Mathematical Sciences and Cryptography, 2019, 22(8): 1489-1499.

[17] Hendi A Y, Dwairi M O, Al-Qadi Z A, et al. A novel simple and highly secure method for data encryption-decryption[J]. International Journal of Communication Networks and Information Security, 2019, 11(1): 232-238.

[18] Hajihassani O, Monfared S K, Khasteh S H, et al. Fast AES implementation: A high-throughput bitsliced approach[J]. IEEE Transactions on parallel and distributed systems, 2019, 30(10): 2211-2222.

[19] van der Hagen M K, Lucia B. Client-optimized algorithms and acceleration for encrypted compute offloading[C]. Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, 2022: 683-696.

[20] Dhanda S S, Jindal P, Singh B, et al. A compact and efficient AES-32GF for encryption in small IoT devices[J]. MethodsX, 2023, 11: 102491.

[21] Kim H, Jang K, Song G, et al. SPEEDY on Cortex–M3: efficient software implementation of SPEEDY on ARM Cortex–M3[C]. International Conference on Information Security and Cryptology. Cham: Springer International Publishing, 2021: 434-444.

[22] Ming J, Zhou Y, Li H, et al. A secure and highly efficient first-order masking scheme for AES linear operations[J]. Cybersecurity, 2021, 4: 1-15.

[23] Xu R, Xiang Z, Lin D, et al. High-throughput block cipher implementations with SIMD[J]. Journal of Information Security and Applications, 2022, 70: 103333.

[24] Marshall B, Page D, Pham T H. A lightweight ise for chacha on risc-v[C]. 2021 IEEE 32nd International Conference on Application-specific Systems, Architectures and Processors

(ASAP). IEEE, 2021: 25-32.

[25] Cheng H, Großschädl J, Marshall B, et al. RISC-V instruction set extensions for lightweight symmetric cryptography[J]. IACR Transactions on Cryptographic Hardware and Embedded Systems, 2023: 193-237.

3. 项目应用前景和学术价值

项目研究的三个方面，硬件实现、软件实现和软硬件协同实现，都具有广泛的应用前景。硬件实现主要适用于专用集成电路（ASIC）和现场可编程门阵列 （FPGA）。在对加密性能和成本有较高要求的场景下，硬件实现具有不可替代的地位。例如，可以应用于IC无线射频卡、云加密机等。软件实现主要适用于8-bit或32-bit的微控制器。在对加密算法部署灵活性有较高要求的场景下，软件实现具有广泛的应用前景。例如，可以应用于智能家居、车联网等。软硬件协同实现主要适用于同时对加密性能、成本和灵活性有较高要求的场景。例如，可以应用于区块链等。

在轻量级加密算法领域，我国相对于国际顶尖水平存在一定的差距。这种差距也体现在轻量级加密算法的实现上。本项目深入研究轻量级加密算法实现的三个方面，包括硬件实现、软件实现和软硬件协同实现。这将有助于推动我国在轻量级加密算法领域的研究进展，弥补与国际水平的差距。同时，这也为我国在轻量级加密算法的标准化过程中提供了实现性能上的评估，推进轻量级加密算法的标准化。

总的来说，本项目的研究成果具有广泛的应用前景和学术价值。对于不同的应用场景，可以提供各种实现方案，以实现更低的成本、更高的性能和更高的灵活性。此外，本项目的研究成果将推动我国在轻量级加密算法领域的研究进展，促进我国在轻量级加密算法实现上的独立自主。

4. 现有研究基础、条件、手段

4.1. 现有研究基础

本项目组主要成员中，有研究生4人（包括申请人向嘉豪）。本项目组一直从事轻量级分组密码算法的实现、分析与设计研究，在该领域取得一定的研究成果，目前取得的成果有中科院3区论文1篇。因此，本项目组对轻量级分组密码方向形成了一定的研究基础，对研究轻量级分组密码算法的实现有着重要意义。

4.2. 现有研究条件

项目组所依托的“嵌入式计算与信息安全研究所”湖南省重点实验室，目前占地面积80平米，相应重要的实验研发设备均已购置，能提供项目良好的研究开发和工作环境，学校对科研工作很重视，也能提供足够的科研工作时间。该研究所有一支稳定的队伍（在职在岗的教师 2 人，博士、硕士研究生30多人）长期从事轻量级密码算法构造及应用、轻量级密码算法优化、轻量级密码算法安全性分析、密码算法攻击的研究，目前发表多篇SCI论文、申请了多项专利，特别在SoC芯片、FPGA实现研究与设计方面积累了宝贵的经验与教训，有着良好的软硬件开发团队作风和项目经验,有

专业资深的老师指导，有产学研项目研究经历，在面向产业化应用研发方面具有较好成果。

4. 3. 现有研究手段

本项目组针对目前国内外轻量级分组加密算法优化和安全实现，利用三大数据库、欧洲密码会议、亚洲密码会议、美国密码会议、FSE、CHES 等渠道持续关注与学习。针对轻量级分组密码算法的实现优化，结合目前主流硬件平台 FPGA 和软件平台 ARM-Cortex，本项目组目前采用软件与硬件技术优化密码算法实现；针对轻量级密码算法的硬件实现，本项目组有使用迭代、展开、串行、流水线等硬件架构，优化算法的硬件实现，同时结合 S 盒约束条件下小空间搜索和布尔可满足性问题中启发式搜索，对加密算法的组件进行硬件实现优化；在算法的软件实现上，利用 ARMv7-M 中 thumb 指令集，对加密算法进行汇编实现优化；在侧信道分析上，结合深度学习，对优化实现后的算法进行相关功耗分析；在算法实现性能评估上，本项目组在 Xilinx Artix-7 FPGA 以及 ASIC 上进行硬件评估，评估的指标主要为面积、延迟、功耗、吞吐量等。在 8/32 位微处理器平台上进行软件评估，评估指标主要为速度、ROM、RAM 等。

5. 指导教师情况

（1）主持与参与的主要项目列表如下：

[1] 国家自然科学基金面上项目：抗功耗攻击的新型轻量级分组密码及其并行验证 (No.61572174), 2016.1-2019.12.

[2] 湖南省自然科学基金省市联合基金：自主知识产权的轻量级分组密码技术及产业化 (No.2019JJ60004), 2019.1-2021.12.

[3] 湖南省教育厅资助科研重点项目：轻量级分组密码系统设计关键技术研究 (No.19A072)，2020.1-2022.12.

（2）近年发表的相关论文列表如下：

[1] Yu Ou, Lang Li Di Li, Jian Zhang.ESRM: An effiffifficient regression model based on random kernels for side channel analysis.International Journal of Machine Learning and Cybernetics,2022,13 (2022):3199-3209.

[2] Di Li, Lang Li, Yu Ou. CKGS: A way of compressed key guessing space to reduce ghost peaks. KSII Transactions on Internet and Information Systems, 2022, 16(3):1047-1062.

[3] Jinling Yang, Lang Li, Ying Guo, Xiantong Huang. DULBC: A dynamic ultra-lightweight block cipher with high throughput[J]. Integration, 2022, 87: 221-230.

[4] Lang Li, Jinggen Liu, Ying Guo, Botao Liu. A new S-box construction method meeting

strict avalanche criterion[J]. Journal of Information Security and Applications, 2022, 66:103135.

[5] Yu Ou, Lang Li. Side-channel analysis attacks based on deep learning network. Frontiers of Computer Science, 2022, 16(2):162303.

[6] Jingya Feng, Lang Li. SCENERY: A lightweight block cipher based on Feistel structure. Frontiers of Computer Science, 2022, 16(3):163813.

[7] Wen Chen, Lang Li, Ying Guo. DABC: A dynamic ARX-based lightweight block cipher with high diffusion[J]. KSII Transactions on Internet and Information Systems, 2023,17(1):165-184.

[8] Wen Chen, Lang Li,Ying Guo,Ying Huang. SAND-2: An optimized implementation of Lightweight block cipher[J].Integration,2023,91(2023):23-34.

[9] Lang Li, Yu Ou.A deep learning-based side channel attack model for different block ciphers[J]. Journal of Computational Science,72(2023):102078.

[10] Juanli Kuang, Ying Guo, Lang Li. IIoTBC: A lightweight block cipher for industrial IoT security[J]. KSII Transactions on Internet and Information Systems, 2023,17(1): 97-119.

[11] Xiantong Huang, Lang Li, Jinling Yang. IVLBC: An involutive lightweight block cipher for Internet of Things[J]. IEEE Systems Journal, 2023, 17(2):3192-3203.

[12] Liuyan Yan, Lang Li, Ying Guo. DBST: a lightweight block cipher based on dynamic S-box[J]. Frontiers of Computer Science, 2023, 17(3):173805.

[13] Ying Huang, Lang Li, Ying Guo, Yu Ou, Xiantong Huang. An efficient differential analysis method based on deep learning[J]. Computer Networks, 224 (2023) :109622.

[14] Di Li, Lang Li, Yu Ou. Side-hannel analysis based on Siamese neural network[J]. The Journal of Supercomputing,2024,80(2024):4423-4450.

[15] Qingling Song, Lang Li, Xiantong Huang. LELBC: A low energy lightweight block cipher for smart agriculture[J]. Internet of Things, 25(2024): 101022.

[16] Juanli Kuang, Xiawei Cao, Songxiao Li, Lang Li. DRcipher: A pseudo random dynamic round lightweight block cipher[J]. Journal of King Saud University-Computer and Information Sciences, 36 (2024):101928.

[17] Jiahao Xiang, Lang Li. Efficient implementations of CRAFT cipher for Internet of Things[J]. Computers and Electrical Engineering,116(2024):109168.

（3）近年的主要获奖：

[1] 李浪, 焦铬, 邹祎, 刘波涛. 新型轻量级分组密码关键技术及其应用, 湖南省技术发明三等奖, 2020.4.

[2] 李浪、焦铬、李秋萍、欧雨、刘波涛、郭影. 面向物联网环境的自主知识产权密码技术及应用.湖南省计算机学会科学技术二等奖, 2022.7.

[3] 李浪.地方院校计算机专业应用创新型人才培养实践与探索, 湖南省教学成果三等奖, 2016.5.

[4] 李浪,2021 年 7 月获湖南省第二届“优秀研究生导师”.

[5] 李浪、郑光勇、邓红卫、焦铬、王承龙、邹祎.“一生一系统”引领的计算机类专业应用型人才培养研究与实践.湖南省计算机学会优秀高等教育教学成果二等奖，2022.7.

（4）近年来已授权的发明专利：

[1] 李浪,黄现彤.一种轻量级 AEROGEL 分组密码的实现方法. ZL202010244240.1.

[2] 李浪,龙荣桀. 一种新型分组密码 CORL 的实现方法. ZL202010247023.8.

[3] 李浪,欧雨. 基于深度学习的侧信道分析方法. ZL202010368459.2.

[4] 李浪,刘嘉辉.一种基于遗传算法的功耗攻击高效筛选方法. ZL202110548000.5.

[5] 李秋萍,李浪,张剑,焦铬. 一种轻量级分组密码 CREF 实现方法及系统. ZL202210489183.2.

[6] 李浪,陈文. 基于广义二维猫映射的轻量级分组密码算法 GCM 实现. ZL202110746280.0.

[7] 李浪,宋庆玲,杨金玲,李永超.一种轻量级可调分组密码实现方法、系统、终端以及可读存储介质.ZL202011301394.6.

[8] 李浪,杨金玲,闫柳焰.轻量级分组密码加密及解密方法.ZL202110922748.7.

三、研究方案

1．研究目标、研究内容和拟解决的关键问题

1.1.研究目标

在轻量级分组密码算法设计上，设计者会考虑算法的安全、性能、成本等因素。但是设计者对算法实现存在局限性，多数算法实现未能达到最优。对于将算法应用到具体场景下，研究轻量级分组密码算法的优化实现有着重要意义。因此，本项目的研究目标是研究轻量级分组密码算法的多场景优化实现，具体来说有以下三方面的目标：

（1）研究轻量级分组密码算法在专用集成电路（ASIC）和现场可编程门阵列（FPGA）上的硬件实现。这将提高算法实现的性能，同时确保其硬件级别的安全。

（2）研究轻量级分组密码算法在 8-bit 或 32-bit 的微控制器上的软件实现。这将提高算法实现的性能，同时确保其软件级别的安全。

（3）研究轻量级分组密码算法的软硬件协同实现。在确保算法实现的灵活性的前提下，最大程度地提高算法实现的性能。

1.2.研究内容

本研究项目的内容是研究轻量级分组密码算法的多场景优化实现。研究轻量级分组密码在硬件、软件和软硬件平台下优化实现的关键技术，具体来说，包括以下三个方面的内容：

（1） 研究轻量级分组密码算法在专用集成电路(ASIC)和现场可编程门阵列(FPGA)上的硬件实现。

在需要高性能的场景下，硬件实现是不可替代的。其中就加密算法部署的规模，ASIC 是一种专用集成电路，可以提供更高的性能，在大批量部署下具有优势。同时 FPGA 是一种现场可编程门阵列，在少批量部署上，可以提供更高的灵活性。本项目将研究轻量级分组密码算法在 ASIC 和 FPGA 上的硬件实现，以提高算法实现的性能，同时考虑硬件实现过程中泄露的侧信道信息，防止功耗、故障攻击等，为算法提供硬件实现安全。

（2）研究轻量级分组密码算法在微控制器上的软件实现。

在物联网、智能家居等场景下，终端设备采用资源受限的微控制器，将硬件实现的加密算法部署到微控制器上，需要片外加密芯片或片内加密模块，这样会增加成本。因此，软件实现是一种更为经济的解决方案。本项目将研究轻量级分组密码算法在 8-bit 或 32-bit 的微控制器上的软件实现，以提高算法实现的性能，同时考虑软件实现过

程中泄露的侧信道信息，防止时间、缓存攻击等，为算法提供软件实现安全。

（3）研究轻量级分组密码算法的软硬件协同实现。

在一些场景下，需要兼顾硬件实现的性能和软件实现的灵活性。例如，区块链等场景下，需要高性能的加密算法，同时也需要灵活的加密算法。本项目将研究轻量级分组密码算法的软硬件协同实现，将加密算法的通用操作设计为扩展指令集或通用加密 IP 核，以提高算法的性能和灵活性。

1.3. 拟解决的关键问题

（1）研究轻量级分组密码组件与结构对于硬件实现的影响。在硬件实现原语之上，优化实现轻量级分组密码，同时设计硬件实现的侧信道防护方案。

（2）研究轻量级分组密码算法微控制器上的软件实现。在微控器的基础指令集与扩展指令集上，优化实现轻量级分组密码，同时设计软件实现的侧信道防护方案。

（3）研究轻量级分组密码算法的软硬件协同实现。利用硬件优化技术设计高效扩展指令集或通用加密 IP 核，结合软件优化技术提高算法部署的性能和灵活性，同时给出实现的侧信道防护方案。

2. 拟采取的研究方法及可行性分析

2.1. 拟采取的研究方法

鉴于本项目的研究内容以及项目组在相关领域的工作积累，拟采取如下详细研究方案和技术路线:

（1） 研究轻量级分组密码算法的硬件实现。

在硬件实现方面，本项目将研究轻量级分组密码算法的硬件实现技术，从算法架构、组件优化和安全实现出发，确保算法实现的性能与硬件安全。在算法架构层，结合轻量级分组密码的特点，优化实现轻量级分组密码。举例来说，将 CRAFT 算法进行串行架构实现，分析调整加密组件的执行次序，同时对加密组件进行时序状态分析，确保算法高效运行，最终减少数据带宽，降低硬件实现面积。在组件优化层，对轻量级分组密码的 S 盒、P 盒等组件进行优化，提高组件的性能。举例来说，将 CRAFT 算法的 S 盒在布尔满足性（SAT）上进行建模，约束 S 盒中的电路门数，求解出低面积 S 盒硬件实现。在硬件安全上，对于故障攻击，通过冗余检测电路和预计算编码表确保安全，在功耗攻击上，通过随机运算电路和掩码技术确保安全。

(2) 研究轻量级分组密码算法在微控制器上的软件实现。

在软件实现方面，从软件运行的平台、加密算法组件优化和安全实现出发，确保算法实现的性能与软件安全。在软件运行的平台层，本项目将研究轻量级分组密码算

法在主流 32-bit 的微控制器上的实现，包括 ARM-Cortex M3、M4、RISC-V 等。结合精简指令集平台所具备的基本指令，对算法进行汇编优化，同时探究特殊平台下的指令对算法的影响，如 ARM-Cortex M4 中循环移位指令对算法的加速效果。在加密算法组件优化层，对轻量级分组密码的 S 盒、P 盒和扩散矩阵等组件进行优化，提高组件的性能。举例来说，将 CRAFT 算法的 S 盒在可满足性（SMT）上比特切片建模，通过限制模型的指令数，提高 S 盒的执行效率。在软件安全上，对于时间攻击，通过随机延迟确保安全，在缓存攻击上，通过比特切片减少算法内存使用，保证组件相同运行时间，减少侧信道信息泄露。

（3）研究轻量级分组密码算法的软硬件协同实现。

在软硬件协同实现上，本项目将研究轻量级分组密码算法的软硬件协同实现技术，在开源的 RISC-V 平台上设计高效扩展指令集及通用加密外设。在扩展指令集上，结合寄存器的数量限制以及指令的设计准则，将加密算法中的通用操作设计为扩展指令，例如，加密算法中起到扩散功能的线性层，设计比特级置换指令，对起到混淆功能的非线性层，设计可配置的替换指令。在通用加密外设上，可以在不更改芯片核心流水线的情况下，将通用加密硬件电路集成到芯片中，将加密算法的通用操作设计为外设访问，通过配置外设的配置寄存器，实现不同加密算法的加速。同时，设计软件接口，将扩展指令集和通用加密外设的操作封装为软件指令，提供给软件开发者使用。

2.2. 可行性分析

首先，针对本项目的主要研究内容和研究目标，项目组做了充分的准备并制定了详细的研究方案和细致的研究计划。同时，针对本项目的各项主要研究内容，本项目组在理论和技术方面均已具备大量的研究基础和工作积累，这将会积极促进本项目的顺利的启动和研究深入。

其次，在研究内容的描述中，我们尽可能细化，明确研究对象，具体研究方法。尤其是在研究方案的阐述中，我们明确了需解决的问题，初步拟定了采取的技术方法等，研究计划清楚且现实可行。最后，项目的研究目标和预期成果制定合理适度，能够保证研究的广度和深度。

综上所述，我们认为完成本项目的研究是切实可行的。

3．本项目的创新之处

本项目的创新之处主要体现在以下三个方面：

（1）本项目将研究轻量级分组密码算法的侧信道攻击防护。在硬件实现和软件实现中，设计防护方案，防止功耗、时间、缓存等侧信道攻击，保证算法实现的安全性。

（2）本项目将研究轻量级分组密码算法的软硬件协同实现。将加密算法的通用操作设计为扩展指令集或通用加密 IP 核，以提高算法的性能和灵活性，面向未来多加密算法场景。

（3）目前对于轻量级分组密码算法的实现，主要集中在单一实现平台上。本项目将研究轻量级分组密码算法的多环境平台下的实现，为设计多平台下高效密码算法设计提供研究基础。

4．预期研究进展

（1）2024 年 6 月—2024 年 11 月

确定总体方案，对项目实施做出具体安排：文献、资料的跟踪及收集，研究和学习轻量级密码算法硬件、软件以及软硬件协同优化实现，为本项目算法设计打下基础。

（2）2024 年 12 月—2025 年 6 月

研究轻量级分组密码的硬件优化实现，并应用到对具体场景。组件上，对轻量级 S 盒在不同工艺库下的最优实现进行研究，架构上，分析已有架构对轻量级分组密码实现的影响。

（3）2025 年 7 月—2025 年 11 月

研究轻量级分组密码的软件优化实现，并在嵌入设备中对其验证。组件上，对轻量级 S 盒最优比特切片实现进行研究，平台上，研究轻量级分组密码在嵌入式平台实现的难点。

（4）2025 年 12 月—2026 年 6 月

通过硬件技术设计出通用加密组件，结合软件实现优化技术，提出一种软硬件协同，优化实现轻量级分组密码的方案。

对项目研究工作进行总结，准备项目结题，撰写总结报告，参加结题答辩。

5．预期成果

（1）发表 SCI 论文一篇以上。（2）申请软著或专利一项以上。

四、研究基础

与本项目有关的研究工作积累和已取得的研究工作成绩及目前承担项目的情况（项目负责人和其它成员情况分开填写并且项目、成果及奖励等须注明承担或完成人姓名等相关信息）

1. 与本项目有关的研究工作积累和已取得的成绩

本项目组一直从事分组密码的分析、设计与实现研究，对最近几年的一些轻量级分组密码算法的结构以及组件特点进行了研究与分析，阅读了相关的最新轻量级密码算法论文并对论文中提出的算法进行了实现，对轻量级密码算法已取得一定的成果和经验，并且本项目组对于面向轻量级分组密码的优化实现也已经有了初步的研究。在硬件优化实现轻量级分组密码算法上，已发表一篇中科院三区论文，与轻量级密码算法优化实现有关的论文如下。

项目负责人：

[1] **Jiahao Xiang,** Lang Li. Efficient implementations of CRAFT cipher for Internet of Things[J]. Computers and Electrical Engineering,116(2024):109168. (中科院三区)

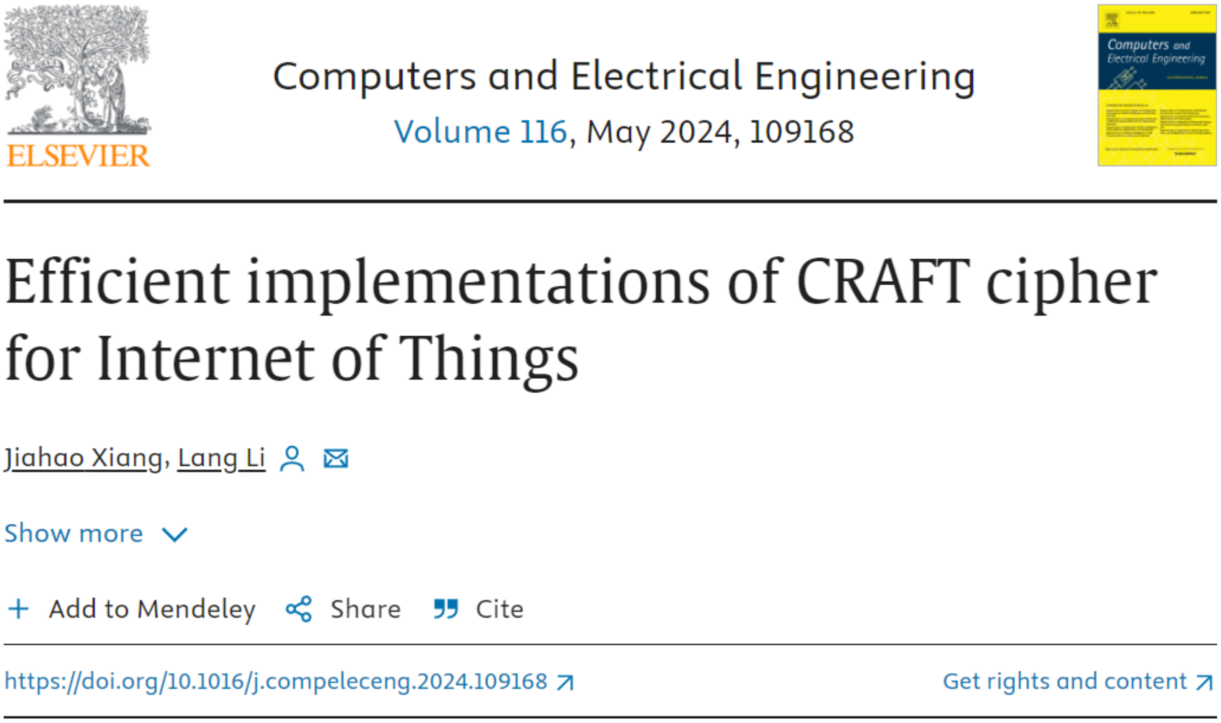
五、经费预算

支出科目	金　额 （万元）	计算根据及理由
合　　计	1	
科研业务费	0.5	参加相关会议、调研学习交流等差旅费用
实验材料费	0.5	所需易耗芯片、测试板等
仪器设备费		
相关经费		

指导教师意见	（就项目研究目标、内容、创新性、研究方案的可行性、预计创新成果等写出具体意见） 项目组成员均为实验室成员，有非常严格的管理制度，有一定的前期研究基础，均有较好的科研创新意识与自学能力。项目内容主要是面向轻量级分组密码，研究密码实现中的软硬件实现优化。项目方案切实可行，易操作，可执行，预期成果有很高的应用价值，同意推荐申报。 签名：　　年　月　日
指导教师承诺	我承诺：如果项目获得专项，我将依照《湖南省研究生科研创新项目管理办法》的有关章则和学校的有关规定对项目进行切实指导和监管。 承诺人：　　年　月　日
研究生院处审查以及经费保障意见	负责人：　　年　月　日 （公章）

论文佐证材料：

Xiang, Jiahao, and Lang Li. "Efficient implementations of CRAFT cipher for Internet of Things." Computers and Electrical Engineering 116 (2024): 109168.



ELSEVIER

Computers and Electrical Engineering

Volume 116, May 2024, 109168

Efficient implementations of CRAFT cipher for Internet of Things

Jiahao Xiang, Lang Li

https://doi.org/10.1016/j.compeleceng.2024.109168 ↗

Computers and Electrical Engineering 116 (2024) 109168

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Efficient implementations of CRAFT cipher for Internet of Things

Jiahao Xiang, Lang Li *

College of Computer Science and Technology, Hengyang Normal University, Hengyang, 421002, China
Hunan Provincial Key Laboratory of Intelligent Information Processing and Application, Hengyang Normal University, Hengyang, 421002, China

ARTICLE INFO

Keywords:
Internet of Things
Lightweight block cipher
Field Programmable Gate Arrays(FPGA)
Low-area
High-throughput

ABSTRACT

The rapid growth of the Internet of Things (IoT) highlights the importance of lightweight cryptography in maintaining security. However, enhancing performance while ensuring the same level of security remains a significant challenge. This paper presents two innovative architectures for the CRAFT lightweight block cipher, aiming to enhance performance without compromising security. The novel Serial and Unrolled architectures are introduced to achieve low area usage and high throughput, respectively. Specifically, the Serial architecture reduces the datapath from 64-bit to 4-bit, significantly decreasing the area. The Unrolled architecture, on the other hand, minimizes latency from 32 to 16. Additionally, Boolean satisfiability (SAT) solvers are employed to identify a lower-cost area implementation of the S-Box. The proposed designs underwent evaluation on three distinct FPGA platforms: Artix-7, Kintex-7, and Spartan-7. The results show that the low area design reduces area usage by 10.16% compared to the previous design. Additionally, the S-Box implementation achieves a significant area reduction of 28.9%. On the other hand, the unrolled design enhances the maximum throughput by 40.53% compared to the previous design. Therefore, the proposed designs could offer enhanced performance while maintaining security for IoT devices.

1. Introduction

The Internet of Things (IoT) is rapidly integrating into various aspects of everyday life. With this advancement, a growing number of security issues are emerging. These security concerns are extensively discussed in [1]. To ensure data protection, the cryptography techniques outlined in [2] are recommended.

However, the resource constraints of many IoT devices pose challenges for the implementation of robust security measures. These devices often have limited memory, processing power, and energy. Therefore, the security measures need to be lightweight to ensure they do not overburden the resources. Lightweight cryptography, a subset of cryptography, provides solutions specifically designed for these resource-limited devices, as discussed in [3].

The field of lightweight cryptography has received considerable attention in recent years. Examples of this include PRESENT [4], LED [5], Midori [6], QTL [7], GIFT [8], CRAFT [9], Shadow [10], DULBC [11], IVLBC [12], BipBip [13], and LELBC [14]. More ciphers can be found in [15]. Concurrently, there has been significant research into side channel attacks on lightweight ciphers. This research can provide valuable insights into the physical security of IoT devices.

Differential fault analysis, which is a type of side channel attack, was first introduced by [16]. This concept was later elaborated in more detail by [17]. In response to these fault attacks, several countermeasures have been suggested. For example, [18] introduced a scheme for error detection. In the realm of Post-Quantum Cryptography, [19] proposed specific error detection methods. It is

* Correspondence to: Hengyang Normal University, Hengyang, 421002, China.
E-mail address: lilang@hynu.edu.cn (L. Li).

https://doi.org/10.1016/j.compeleceng.2024.109168
Received 9 January 2024; Received in revised form 8 March 2024; Accepted 8 March 2024
Available online 15 March 2024

Table 1
Main notations.

Notation	Description
TK_i	tweakeys used in the ith round
RC_i	round constant for the ith round
R_i	Function for the ith round
SB	Sub-Box
MC	Mix-Columns
PN	PermuteNibbles
PK	Permutation used in key schedule
$\oplus$	XOR operation
$\|\|$	Concatenation operation
$\sim$	Inverse operation
$\wedge$	And operation
$\vee$	Or operation

important to note that these methods do increase the hardware consumption of the cipher system. The CRAFT cipher was designed with resistance to fault attacks in mind. However, it currently lacks efficient implementations. To make it suitable for use in more constrained environments, development of more efficient implementations is necessary.

Efficient implementation allows lightweight ciphers to be used in various settings. A hardware implementation can enhance the performance of these ciphers in resource-limited environments. Several researchers have proposed optimized architectures for various ciphers. Lara-Nino et al. [20] introduced a 16-bit datapath architecture for the PRESENT cipher, which resulted in reduced area and power consumption. Similarly, Pandey et al. [21] suggested an optimized key schedule for the same cipher, leading to a smaller area. Shahbazi et al. [22] put forth an 8-bit serial architecture for AES, which also reduces area and power consumption. Li et al. [23] presented unrolled architectures and a low-cost architecture for PRINCE, optimizing both throughput and area separately. Further enhancements have been made by Bharathi et al. [24], who improved the performance of the PRESENT cipher by expanding the key length. Lastly, Yang et al. [25] shared components in the cipher process for LILLIPUT, resulting in a smaller area.

This work presents the first implementation of CRAFT on FPGA platforms. Two architectures for CRAFT, Serial and Unrolled, are proposed. The Serial architecture reduces the datapath from 64-bit to 4-bit, meaning it only uses one S-Box, which significantly reduces the area usage. The Unrolled architecture reduces the latency of the encryption process, thereby improving the throughput rate. The optimal implementation of the S-Box, aimed at further area reduction, is determined using a SAT solver in conjunction with the GEC encoding scheme. The experiments are conducted on three different FPGA platforms: Artix-7, Kintex-7, and Spartan-7. The source code for the proposed designs is available online.[1] The main contributions of this article are as follows.

- Two architectures for CRAFT, Serial and Unrolled, are proposed. These are optimized for area and throughput, respectively. The Serial architecture reduces the area usage by 10.16% compared to the work of Beierle et al. [9]. The Unrolled architecture doubles the throughput rate compared to the same work.
- The optimal implementation of the S-Box, which results in further area reduction, is identified using a SAT solver. The proposed S-Box implementation achieves a 28.9% area reduction compared to the work of Bao et al. [26].
- The architectures are implemented across three different FPGA platforms: Artix-7, Kintex-7, and Spartan-7. This variety allows engineers to select the platform that best suits their application needs.

The remainder of this article unfolds as follows: Section 2 delves into the specifics of CRAFT. The duo of proposed architectures for CRAFT are explored in Section 3. Section 4 details the metrics and environment used for the experimental evaluation and comprehensive performance analysis. A thorough discussion of all the architectures is provided in Section 5. Lastly, Section 6 encapsulates the work done and points towards potential avenues for future research.

2. Specification of CRAFT

CRAFT is a lightweight tweakable block cipher that operates on a 64-bit plaintext size, a 128-bit key size, and a 64-bit tweak size. It outputs a 64-bit ciphertext. Although quantum computing has enhanced capabilities to attack ciphers, as highlighted by Darzi et al. [27] and Canto et al. [28], it is noteworthy that the probabilistic algorithm based on quantum computing, proposed by Grover et al. [29], could reduce the key space from 128-bit to 64-bit. However, this reduction is not sufficient to brute force attack the CRAFT cipher with current computational capabilities. The confusion and diffusion properties of CRAFT ensure that the distribution of probabilities between the plaintext and ciphertext is independent. For more details on CRAFT, refer to Fig. 1 which depicts its architecture. The encryption process of CRAFT is outlined in Algorithm 1. The decryption process is similar to the encryption process, with the only difference being that the round keys are applied in reverse order. The main notations used throughout this paper are outlined in Table 1.

[1] https://github.com/xjh2000/craft_implementation

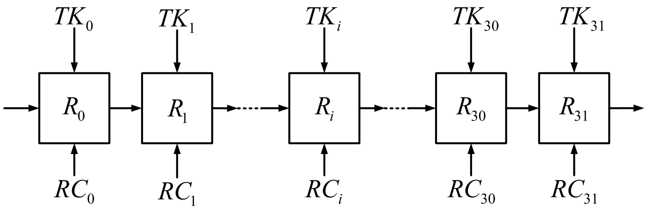


Fig. 1. Architecture of CRAFT.

Algorithm 1 CRAFT Encryption Process

```
Input: Plaintext $P$, Key $K_0||K_1$, Tweak $T$
Output: Ciphertext $C$
$TK_0 \leftarrow K_0 \oplus T$
$TK_1 \leftarrow K_1 \oplus T$
$TK_2 \leftarrow K_0 \oplus PK(T)$
$TK_3 \leftarrow K_1 \oplus PK(T)$
$C \leftarrow P$
for $i \leftarrow 0$ to 31 do
    $C \leftarrow MC(C)$
    $C_{4,5} \leftarrow C_{4,5} \oplus RC_i$
    $C \leftarrow C \oplus TK_{i \bmod 4}$
    if $i \neq 31$ then
        $C \leftarrow PN(C)$
        $C \leftarrow SB(C)$
    end if
end for
```

Table 2
S-Box of CRAFT.

Input	Output	Input	Output
0	c	8	8
1	a	9	9
2	d	a	1
3	3	b	5
4	e	c	0
5	b	d	2
6	f	e	4
7	7	f	6

The round function is composed of three distinct operations: Mix-Columns, PermuteNibbles, and Sub-Box. The Mix-Columns operation is a linear transformation that multiplies the input column by a constant matrix, M, to generate the output column. Notably, M is an involutory matrix, as shown in Eq. (1).

$$M = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \tag{1}$$

The PermuteNibbles operation, an involutory permutation, operates on 4-bit nibbles. This operation triggers additional S-Boxes, thereby bolstering the cipher's security. The illustration of the PermuteNibbles operation is provided in Eq. (2). The permutation PK is utilized in the key schedule, as depicted in Eq. (3).

$$PN = [15, 12, 13, 14, 10, 9, 8, 11, 6, 5, 4, 7, 1, 2, 3, 0] \tag{2}$$

$$PK = [12, 10, 15, 5, 14, 8, 9, 2, 11, 3, 7, 4, 6, 0, 1, 13] \tag{3}$$

The Sub-Box operation, a nonlinear transformation, introduces confusion into the cipher. This operation is performed using a 4-bit S-Box. The values are represented in hexadecimal notation, as shown in Table 2.

Two Linear Shift Feedback Registers (LSFRs), a and b, are used to concatenate the round constants. The round constants is defined as $RC = (a_3, a_2, a_1, a_0, 0, b_2, b_1, b_0)$. The initial round constant, RC_0, is set to 0×11.

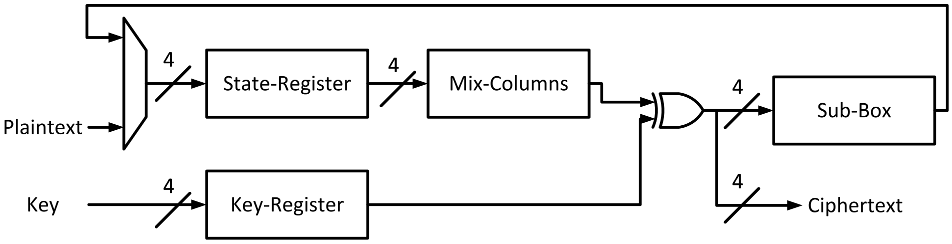


Fig. 2. Serial architecture of CRAFT.

The iterative architecture, often referred to as the round-based architecture, was first introduced in the original CRAFT paper [9]. This architecture works by executing one round function per cycle. It employs a single round function to encrypt a block. This function includes operations such as Mix-Columns, PermuteNibbles, Sub-Box, AddConstant, and AddTweakey. To encrypt a single block, this round function is run over 32 cycles. In the meantime, the Key Schedule runs concurrently with the round function.

3. Implementations

For the first time, the components of CRAFT have been optimized to achieve efficient area and throughput, resulting in two proposed implementation architectures: Serial and Unrolled.

3.1. Serial architecture (SA)

The purpose of the serial architecture is to reduce the datapath, which represents the number of bits dealt with in one cycle. For instance, the CRAFT cipher has a 64-bit block size, meaning it can process 64-bit data in one cycle when using the iterative architecture. In contrast, the serial architecture reduces the datapath from 64-bit to 4-bit, meaning it can only process 4-bit data in one cycle. Despite this limitation, serial architectures can significantly reduce area usage by reusing components, making them a viable alternative to iterative architectures. For example, the quantity of S-Boxes is diminished from 16 to 1. The clock gating technique is also employed to enable each component and minimize the energy consumption of encryption.

The architecture, depicted in Fig. 2, comprises a single Sub-Box, a 4-bit Mix-columns, and two register banks. The Key-Register is used to store keys. The State-Register, on the other hand, is used to store plaintext. They also hold intermediate results temporarily. The design incorporates a feedback path to store intermediate results in the State-Register bank. Additionally, the PermuteNibbles function is integrated into the State-Register bank.

3.1.1. S-Box optimization

The S-Box is a crucial component of the encryption algorithm, adding to its complexity. However, it also demands a significant amount of area. There are several ways to implement the S-Box. One prevalent approach is to use a lookup table (LUT), a technique described by Lara-Nino et al. [20]. This approach, while effective, requires many flip-flops, which can lead to a substantial increase in area consumption. An alternative method is to use the logical equivalent expression of the S-Box. This method, suggested by Bao et al. [26] and Feng et al. [30], can help reduce area consumption.

Boolean satisfiability (SAT) solvers can be used to find S-Boxes that meet specific implementation requirements, such as meet to certain software or hardware implementation requirements. To elaborate, the S-Box implementation can be encoded as Boolean constraints. This is done by representing the S-Box as a truth table and using Boolean variables to denote the input and output bits of the S-Box. The constraints are then formulated based on the desired properties of the S-Box. Once the S-Box properties are encoded as Boolean constraints, these constraints can be satisfied by a SAT solver. This assignment corresponds to an S-Box that fulfills the desired properties.

A measure of the number of logical gates required to implement the Boolean formula that represents a SAT instance is the Gate Equivalent Complexity (GEC). To calculate the GEC, the Boolean formula is converted into a circuit of logical gates, such as AND, OR, and NOT gates. The total count of these gates in the circuit gives the GEC of the instance. In this design, the GEC encoding scheme from Feng et al. [30] is optimized and used to implement the S-Box. The encoding scheme is detailed in Eqs. (4):

$$
\begin{aligned}
&\forall i \in \{0, 1, \dots, K-1\} : \\
&T_i = F_{if}(GT_i[0], \sim (X_{4i} \cdot X_{4i+1}) \cdot \sim X_{4i+2} \cdot X_{4i+3}) \\
&\quad + F_{if}(GT_i[1], X_{4i+2} \cdot (X_{4i} + X_{4i+1})) \\
&\quad + F_{if}(GT_i[2], X_{4i} \cdot X_{4i+1} \cdot X_{4i+2}) \\
&\quad + F_{if}(GT_i[3], X_{4i+2}) + F_{if}(GT_i[4], X_{4i}) \\
&\quad + F_{if}(GT_i[5], X_{4i} \cdot X_{4i+1}) \\
&\quad + F_{if}(GT_i[6], X_{4i} + X_{4i+1}) + F_{if}(GT_i[7], max)
\end{aligned}
\tag{4}
$$

Table 3
Encoding of different types of logical gates.

Logical expression	GT_i	Gate type
$X_0 \oplus X_1$	2	XOR
$\sim (X_0 \oplus X_1)$	3	XNOR
$X_0 \wedge X_1$	4	AND
$\sim (X_0 \wedge X_1)$	5	NAND
$X_0 \vee X_1$	6	OR
$\sim (X_0 \vee X_1)$	7	NOR
$\sim X_0$	9	NOT
$\sim X_1$	11	NOT
$\sim X_2$	17	NOT
$X_0 \oplus X_1 \oplus X_2$	18	XOR3
$\sim (X_0 \oplus X_1 \oplus X_2)$	19	XNOR3
$X_0 \wedge X_1 \wedge X_2$	32	AND3
$\sim (X_0 \wedge X_1 \wedge X_2)$	33	NAND3
$X_0 \vee X_1 \vee X_2$	118	OR3
$\sim (X_0 \vee X_1 \vee X_2)$	119	NOR3
$\sim ((X_0 \wedge X_1) \vee (\sim (X_2 \vee X_3)))$	176	MAOI1
$\sim (\sim (X_0 \wedge X_1) \wedge ((X_2 \vee X_3)))$	177	MOAI1

where K is numbers of the logical gates, $X_{4i} - X_{4i+3}$ is the input of the ith logical gate, T_i is the output of the ith logical gate, and F_{if} is a function that returns the value of the second argument if the first argument is true and returns the value of zero otherwise. The value of max is all one's in the binary expression, which is represented logically as an inverse. GT_i denotes the type of the ith logical gate, represented as an 8-bit binary number. The least significant bit of GT_i is indexed at seven. Table 3 enumerates the various types of logical gates employed in this encoding scheme.

Eqs. (5) display the optimized scheme of the S-Box, where $X_3 - X_0$ represents the input and $Y_3 - Y_0$ represents the output. The proposed S-Box scheme is implemented using four MOAI1 gates, three MAOI1 gates, and one AND3 gate. This configuration of the S-Box module results in a 28.9% reduction in area compared to the method proposed by Bao et al. [26], based on gate equivalent (GE) estimation using the UMC 180 nm library.

$$\begin{aligned}
&T_0 = \mathrm{MAOI1}(X_0, X_1, X_0, X_1)\\
&T_1 = \mathrm{AND3}(X_3, X_2, X_3)\\
&T_2 = \mathrm{MAOI1}(X_1, X_2, X_0, X_3)\\
&T_3 = \mathrm{MOAI1}(X_1, X_0, X_2, X_2)\\
&T_4 = \mathrm{MOAI1}(X_3, T_0, T_3, T_3)\\
&T_5 = \mathrm{MOAI1}(T_3, T_0, X_0, T_1)\\
&T_6 = \mathrm{MAOI1}(X_0, T_0, X_3, T_0)\\
&T_7 = \mathrm{MOAI1}(X_0, T_1, T_2, T_2)\\
&Y_3 = T_4 \quad Y_2 = T_6 \quad Y_1 = T_7 \quad Y_0 = T_5
\end{aligned} \tag{5}$$

3.1.2. Mix-Columns optimization

The Mix-Columns component is a linear transformation of the input column. The output column is generated by multiplying the input column with a constant matrix M. M is a involutory matrix, which means $M^2 = E$, where E is the identity matrix. It is easy to decrypt the ciphertext by multiplying the ciphertext with M again. Eq. (6) illustrates the Mix-columns component. Here, $I_{3,j}$, $I_{2,j}$, $I_{1,j}$, and $I_{0,j}$ represent the input column, while $I'_{3,j}$, $I'_{2,j}$, $I'_{1,j}$, and $I'_{0,j}$ denote the output column. The column index is given by j, where j ranges from 0 to 3.

$$\begin{bmatrix} I'_{3,j} \\ I'_{2,j} \\ I'_{1,j} \\ I'_{0,j} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} I_{3,j} \\ I_{2,j} \\ I_{1,j} \\ I_{0,j} \end{bmatrix} \tag{6}$$

In order to reduce the area of this component, the serial architecture of Mix-Columns is utilized, as shown in Fig. 3. The serial architecture of Mix-Columns requires four 4-bit registers, two multiplexers, and three XOR gates. The operation of Mix-Columns involves three distinct stages: freeze, shift, and add. During the freeze stage, the register values are kept unchanged by setting both CM_0 and CM_1 to 0. In the shift stage, a shift in the register values from RM_0 to RM_4 is induced by setting both CM_0 and CM_1 to 1. Finally, in the add stage, an addition operation on the column values is executed according to Eq. (6). This is achieved by setting CM_0 and CM_1 to 0 and 1, respectively.

Fig. 4 presents the timing diagram for the serial architecture of the Mix-Columns operation. It requires five clock cycles to compute the next columns from the previous ones, and an additional four clock cycles to transfer data from the internal register of Mix-Columns to the State-Register. Therefore, a complete state round requires a total of 36 clock cycles.

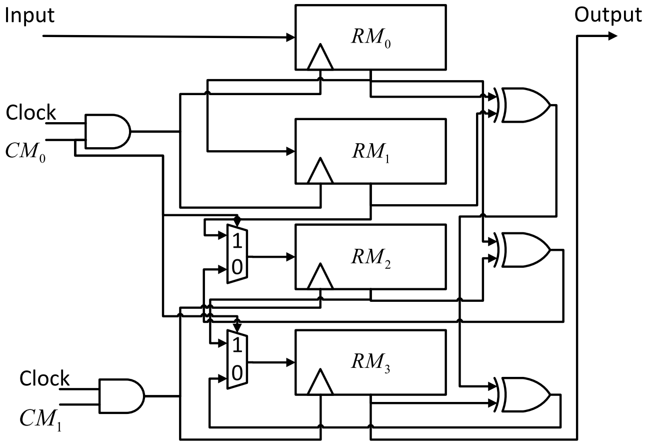


Fig. 3. Serial architecture of Mix-Columns with clock gating.

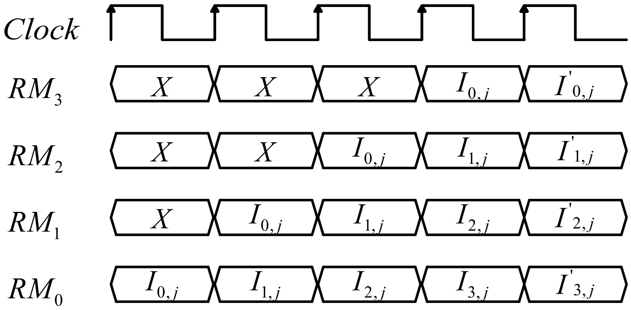


Fig. 4. Timing diagram for the Serial Architecture of Mix-Columns.

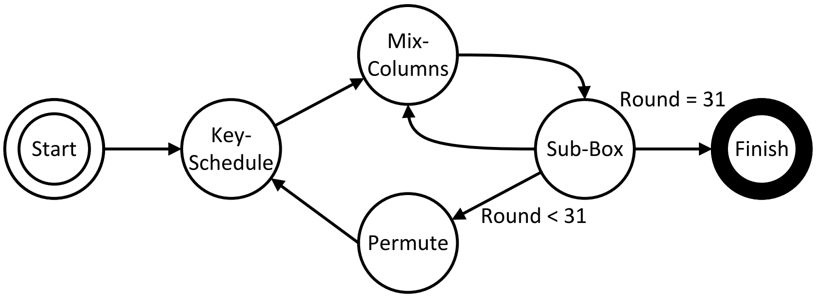


Fig. 5. Finite-state machine for Serial Architecture.

3.1.3. Control units

As depicted in Fig. 5, the finite-state machine (FSM) initiates the encryption process by loading the initial key into the Key-Register and the plaintext into the State-Register. During the Key-Schedule phase, the key is expanded while the gate clocks of the Mix-Columns and State-Register are turned off. Next, the Mix-Columns phase begins, where one column of the State-Register is stored in the Mix-Columns registers. The Mix-Columns operation on one column takes five clock cycles to execute in this phase. Once this phase is finished, the gate clocks for the State-Register and Key-Register are turned off. Following this, the Sub-Box phase commences. During this phase, the data from the Mix-Columns registers is transferred again to the State-Register and XORed with the keys. This process requires an additional four clock cycles. This cycle between the Mix-Columns and Sub-Box phases is repeated four times for the four columns of the State-Register. Subsequently, the Permute operation is carried out within the State-Register, requiring a single clock cycle. The encryption process concludes when the Round counter reaches 31, at which point the ciphertext is stored in the State-Register.

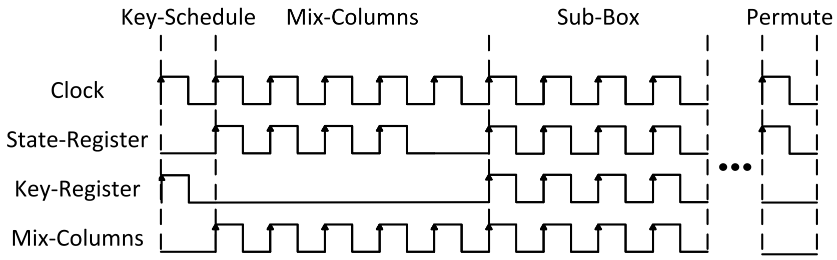


Fig. 6. Timing diagram for Serial Architecture.

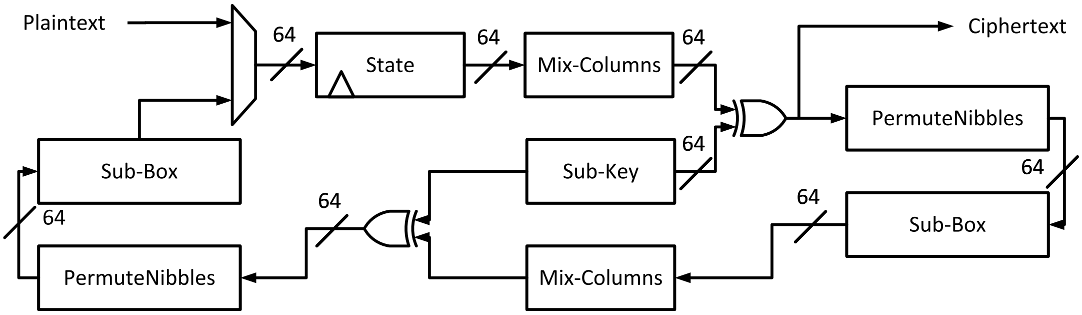


Fig. 7. Unrolled architecture of CRAFT.

In the work of Shahbazi et al. [22], it is discussed that the dynamic power consumption of the encryption process can be mitigated through the use of clock gating. The clock gating technique is independently applied to the State-Register, Key-Register, and Mix-Columns. For instance, in the Permute phase, the gate clocks of the Key-Register and Mix-Columns are disabled as these components are not in use. This helps save a significant amount of power. Fig. 6 shows the timing diagram of a design that uses the clock gating technique.

3.2. Unrolled architecture (UA)

The unrolled architecture allows for the execution of more than one round function in a single cycle. In contrast, the iterative architecture of the CRAFT cipher only runs one round function per cycle. This approach can significantly reduce the latency of the encryption process. It does this by reducing the number of cycles needed to encrypt one block size plaintext, thereby improving the throughput rate.

The unrolled architecture shown in Fig. 7 includes two Sub-Boxes, two Mix-Columns, a Key, a State-Register, two PermuteNibbles, and one feedback path. It is designed to perform a 30-round encryption process in just 15 cycles. Only the Mix-Columns and Add-Key operations are carried out in the final cycle, finishing the encryption process in a total of 16 cycles.

The unrolled architecture, which is based on the iterative architecture from the work of Beierle et al. [9], completes the encryption process in only 16 cycles, compared to the 32 cycles needed by the iterative architecture. Here, a cycle includes two round functions of CRAFT. While this approach might use more area, it provides higher throughput at the same frequency.

4. Experimental results

In ASIC implementations, the Gate Equivalent (GE) is often used to evaluate the area consumption of a design. One GE corresponds to the area of a two-input NAND gate. The area is computed in terms of GEs. This is done by dividing the total area (measured in μm^2) by the area of a two-input NAND gate (also measured in μm^2). However, the number of GEs can vary depending on the specific technology used, as Turan et al. discuss in their work [31]. For instance, the number of GEs for the same design will differ between UMC 180 nm technology and TSMC 180 nm technology. Therefore, GE is not suitable for comparing the area consumption of different designs on different technologies. For a fair comparison, the area consumption of the proposed designs is assessed using FPGA implementations. This is a technique similarly employed in the study by Mohajerani et al. [3].

Table 4
Description of different architectures.

Architecture	Description	Reference
SA	Serial Architecture	This work
UA	Unrolled Architecture	This work
IA	Iterative Architecture	[9]

4.1. Platform

The designs proposed in this study were implemented on a Xilinx FPGA board, utilizing the Vivado v2023.2 software for deployment. Benchmarking was performed across three distinct FPGA platforms to ensure a diverse testing environment: Artix-7(xc7a100tcsg324-1), Kintex-7(xc7k70tfbg484-1), and Spartan-7(xc7s100fgga484-1). Artix-7 offers high performance in resource-limited situations. Spartan-7 is designed for high-restriction environments. Kintex-7 is well-suited for use in applications such as 3G and 4G wireless technologies.

4.2. Area

The area consumed by the proposed designs is quantified using the Area metric, which encompasses components such as Flip-Flops, LUTs, and Slices. For a balanced comparison, the embedded memory blocks of the FPGA were not utilized. This was achieved by disabling the relevant settings in the VHDL, as recommended in the design guidelines. Also, all designs were synthesized and implemented using the same settings, specifically, the default settings of Vivado Synthesis and Implementation.

4.3. Throughput

The efficiency of the proposed designs is assessed using the Throughput metric. This metric uses three parameters: the maximum throughput rate, the throughput rate at 100 MHz, and the throughput rate per slice. The maximum throughput rate is the highest rate that our designs can achieve, calculated using Eq. (7). The throughput rate at 100 MHz shows the rate achievable when the clock frequency is set to 100 MHz, calculated using Eq. (8). The throughput rate per slice is a measure of efficiency, calculated by dividing the throughput rate by the number of Slices, as defined in Eq. (9). In these computations, the Plaintext Size is set to 64-bit. Latency denotes the count of clock cycles needed to encrypt a single block, and Slices represent the quantity of Slices consumed by the design.

$$MaxThroughput(Thr) = \frac{MaxFrequency \times BlockSize}{Latency} \tag{7}$$

$$Throughput_{@100\ \mathrm{MHz}}(Thr^{*}) = \frac{100\ \mathrm{MHz} \times BlockSize}{Latency} \tag{8}$$

$$ThroughputPerSlice = \frac{Thr}{Slices} \tag{9}$$

4.4. Power and energy

The Power metric, which includes both dynamic and static power consumption, is used to evaluate the power consumption of the proposed designs, as defined in Eq. (10). On the other hand, the Energy metric measures the energy consumption of the designs. It is calculated by multiplying the power consumption by the time needed to encrypt a single block. This time is determined by dividing the latency by the frequency, as explained in Eq. (11).

$$\begin{aligned}\text{Total Power}(TP) = {}& \text{Dynamic Power}(DP)\\ &+ \text{Static Power}(SP)\end{aligned} \tag{10}$$

$$Energy(E) = \frac{TP \times Latency}{Frequency} \tag{11}$$

4.5. Results

This section presents the results of the proposed designs, focusing on three key aspects: area consumption, throughput performance, and power and energy metrics. These results are demonstrated across three different FPGA platforms, namely Artix-7, Kintex-7, and Spartan-7. The various architectures used in this study are described in Table 4. For a comparison of area consumption across these designs, refer to Table 5. Throughput results are provided in Table 6, while Table 7 contains information on power and energy consumption.

Table 5
Area used in three architectures.

Platform	Design	$State(bit)$	$Key(bit)$	FF	LUT	$Slices$
Artix-7	SA	64	128	**144**	**177**	**59**
	UA	64	128	157	378	111
	IA	64	128	159	201	65
Kintex-7	SA	64	128	**144**	**178**	**58**
	UA	64	128	157	377	115
	IA	64	128	159	205	66
Spartan-7	SA	64	128	**144**	**177**	**57**
	UA	64	128	157	381	118
	IA	64	128	158	200	64

Table 6
Throughput results in three architectures.

Platform	Design	$Latency$	$MaxF$(MHz)	Thr(Mbps)	Thr^{*}(Mbps)[a]	$Thr/Slices$(Kbps/Slices)
Artix-7	SA	1215	**557.41**	29.36	5.27	497.65
	UA	**16**	142.38	**569.52**	**400.00**	5130.81
	IA	32	202.63	405.26	200.00	**6234.77**
Kintex-7	SA	1215	**853.97**	44.98	5.27	775.57
	UA	**16**	175.25	**701.00**	**400.00**	6095.65
	IA	32	301.38	602.76	200.00	**9132.73**
Spartan-7	SA	1215	**525.76**	27.69	5.27	485.87
	UA	**16**	138.86	**555.44**	**400.00**	4707.12
	IA	32	186.98	373.96	200.00	**5843.13**

[a] Throughput rate at 100 MHz.

Table 7
Power and energy consumption in three architectures.

Platform	Design	DP(mW)	SP(mW)	TP(mW)	E(uJ)	E/bit(nJ/bit)
Artix-7	SA	**2.00**	139.00	**141.00**	1.71	26.77
	UA	7.00	139.00	146.00	**0.02**	**0.37**
	IA	3.00	139.00	142.00	0.05	0.71
Kintex-7	SA	**2.00**	145.00	**147.00**	1.79	27.91
	UA	8.00	145.00	153.00	**0.02**	**0.38**
	IA	3.00	145.00	148.00	0.05	0.74
Spartan-7	SA	**2.00**	140.00	**142.00**	1.73	26.96
	UA	7.00	140.00	147.00	**0.02**	**0.37**
	IA	3.00	140.00	143.00	0.05	0.72

DP: Dynamic Power & SP: Static Power & TP: Total Power & E: Energy.

The area consumption of the proposed designs is evaluated based on three factors: Flip-Flops (FF), Look-Up Tables (LUT), and Slices. These designs are compared with the iterative architecture of CRAFT, as detailed in the work of Beierle et al. [9]. The results indicate that the proposed serial architecture consume 10.16% less area than the iterative architecture of CRAFT.

Regarding the FF, the key schedule of the CRAFT cipher is implemented using multiplexers. This eliminates the need for FF to store the sub-key, resulting in a lower FF count compared to other ciphers. This is a significant factor contributing to the CRAFT cipher's requirement of less than 1000 GE, which is the lowest known requirement on the IBM 130 nm ASIC library, as demonstrated in the study by Beierle et al. [9]. A comparison of FF counts is provided in Fig. 8.

As illustrated in Fig. 9, when it comes to LUT, the proposed serial architecture require fewer LUTs than the iterative architecture of CRAFT. This is attributed to the fact that the proposed designs utilize a single S-Box, in contrast to the 16 S-Boxes used by the iterative architecture of CRAFT. Furthermore, the proposed designs also require fewer LUTs than the unrolled architecture of CRAFT, which uses 32 S-Boxes, compared to just one in the proposed designs.

Thanks to the reduction in Flip-Flop (FF) and Look-Up Table (LUT) usage, the serial architecture has fewer slices compared to the iterative architecture of CRAFT. In terms of Slices efficiency, Spartan-7 outperforms both Artix-7 and Kintex-7 platforms. These results are illustrated in Fig. 10. However, the lower Max Frequency of Spartan-7 will be considered in the Throughput comparison.

Fig. 11 illustrates that the proposed serial architecture have a higher Max Frequency than the iterative architecture of CRAFT. This improvement is due to two key factors. First, the S-Box is optimized with the GEC encoding scheme, reducing its delay. Second, the serial architecture of the design further reduces the overall delay of the encryption process. However, among all platforms, Spartan-7 has the lowest Max Frequency, primarily because it has the fewest LUTs, as shown in Fig. 9. The unrolled architecture reduces the latency to 16. This effectively doubles the throughput rate at 100 MHz when compared to the iterative architecture of CRAFT. Additionally, the unrolled architecture improves the maximum throughput by 40.53% on the Artix-7 platform. This data is presented in Table 6.

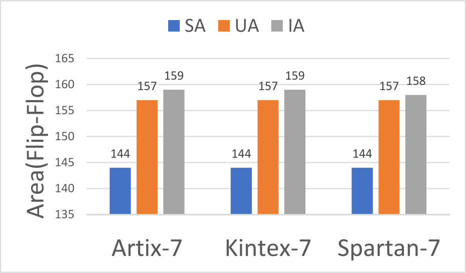


Fig. 8. Comparison of flip-flop in three architectures.

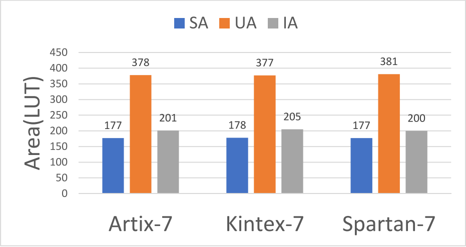


Fig. 9. Comparison of look-up tables in three architectures.

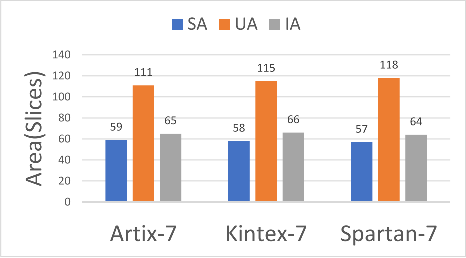


Fig. 10. Comparison of slices in three architectures.

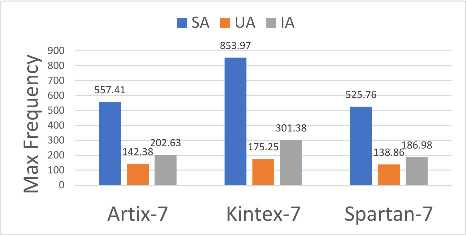


Fig. 11. Comparison of max frequency in three architectures.

The serial architecture has the highest energy per bit due to its higher latency, which results in the smallest area. Conversely, the unrolled architecture has the lowest energy per bit because it has the lowest latency. The energy per bit of the iterative architecture of CRAFT falls between that of the serial architecture and the unrolled architecture. Compared to the iterative architecture of CRAFT, the unrolled architecture reduces energy per bit by 47.89%. Fig. 12 illustrates the energy per bit for the three architectures.

5. Discussion

This section discusses the performance of three different architectures of CRAFT: the serial architecture, the unrolled architecture, and the iterative architecture. These architectures are compared and analyzed to determine which one is best suited for different environments.

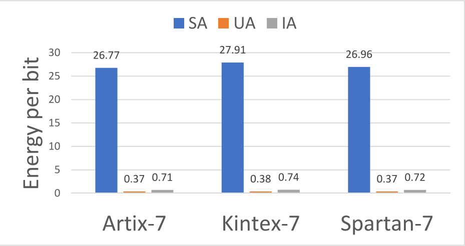


Fig. 12. Comparison of energy per bit in three architectures.

The serial architecture of CRAFT is designed to minimize area consumption. It has the lowest area consumption among the three architectures, which results in it having the lowest dynamic power. Additionally, it boasts the highest frequency among the three architectures. However, it has a high latency, leading to the lowest maximum throughput among the three architectures. The serial architecture is suitable for resource-limited environments where high throughput is not a requirement.

The unrolled architecture of CRAFT aims to maximize throughput. It boasts the lowest latency among the three architectures, which contributes to its highest maximum throughput, despite having the lowest maximum frequency. This architecture is also energy-efficient, offering the lowest energy per bit among the three architectures. However, it does have the highest area consumption. The unrolled architecture is best suited for environments where high throughput and low energy are priorities, and low area is not a requirement.

The iterative architecture of CRAFT is designed to strike a balance between area consumption and throughput. While it does not have the lowest area consumption, the highest frequency, the lowest latency, the highest maximum throughput, or the lowest energy per bit among the three architectures, it does have the highest throughput per slice. The iterative architecture is suitable for environments where moderate throughput is required at the lowest possible area cost.

6. Conclusion

Given the diverse performance requirements arising from the use of IoT devices in various contexts, achieving optimal security without compromising performance presents a significant challenge. Implementing effective solutions is one way to attain this balance between security and performance.

This research presents two unique designs for the CRAFT Lightweight cipher – Serial and Unrolled – both aimed at boosting performance. The Serial architecture reduces the area consumption by 10.16% compared to the iterative architecture of CRAFT. The Unrolled architecture reduces the latency to 16. This effectively doubles the throughput rate at 100 MHz compared to the iterative architecture of CRAFT, resulting in a 40.53% improvement in maximum throughput. Furthermore, the Unrolled architecture also reduces the energy per bit by 47.89% when compared to the iterative architecture of CRAFT. To the best of our knowledge, the Serial architecture establishes a new record for area efficiency on an FPGA configured with a 64-bit block size and 128-bit key size. These proposed architectures are therefore highly suitable for environments with IoT devices.

Future work could extend the proposed architectures to other lightweight ciphers and examine their performance. Additionally, these lightweight ciphers could be implemented in a way that makes them resistant to side channel attacks.

CRediT authorship contribution statement

Jiahao Xiang: Conceptualization, Methodology, Software, Data curation, Writing – original draft, Visualization, Investigation. **Lang Li:** Software, Validation, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The source code related to this paper has been made publicly available online. The URL can be found within the paper itself.

Acknowledgments

This research is supported by the Hunan Provincial Natural Science Foundation of China (2022JJ30103), "the 14th Five Year Plan" Key Disciplines and Application-oriented Special Disciplines of Hunan Province, China (Xiangjiaotong [2022] 351), the Science and Technology Innovation Program of Hunan Province, China (2016TP1020).

References

[1] Meneghello F, Calore M, Zucchetto D, Polese M, Zanella A. IoT: Internet of threats? A survey of practical security vulnerabilities in real IoT devices. IEEE Internet Things J 2019;6(5):8182–201. http://dx.doi.org/10.1109/JIOT.2019.2935189.

[2] Swessi D, Idoudi H. A survey on Internet-of-Things security: Threats and emerging countermeasures. Wirel Pers Commun 2022;124(2):1557–92. http://dx.doi.org/10.1007/S11277-021-09420-0.

[3] Mohajerani K, Haeussler R, Nagpal R, Farahmand F, Abdulgadir A, Kaps J, et al. FPGA benchmarking of round 2 candidates in the NIST lightweight cryptography standardization process: Methodology, metrics, tools, and results. IACR Cryptol ePrint Arch 2020;1207.

[4] Bogdanov A, Knudsen LR, Leander G, Paar C, Poschmann A, Robshaw MJB, et al. PRESENT: An ultra-lightweight block Cipher. In: Paillier P, Verbauwhede I, editors. Cryptographic hardware and embedded systems - CHES 2007, 9th international workshop, vienna, Austria, September 10-13, 2007, proceedings. Lecture notes in computer science, vol. 4727, Springer; 2007, p. 450–66. http://dx.doi.org/10.1007/978-3-540-74735-2_{3}{1}.

[5] Guo J, Peyrin T, Poschmann A, Robshaw MJB. The LED block Cipher. In: Preneel B, Takagi T, editors. Cryptographic hardware and embedded systems - CHES 2011 - 13th international workshop, nara, Japan, September 28 - October 1, 2011. proceedings. Lecture notes in computer science, vol. 6917, Springer; 2011, p. 326–41. http://dx.doi.org/10.1007/978-3-642-23951-9_{2}{2}.

[6] Banik S, Bogdanov A, Isobe T, Shibutani K, Hiwatari H, Akishita T, et al. Midori: A block Cipher for low energy. In: Iwata T, Cheon JH, editors. Advances in cryptology - ASIACRyPT 2015 - 21st international conference on the theory and application of cryptology and information security, auckland, New zealand, November 29 - December 3, 2015, proceedings, part II. Lecture notes in computer science, vol. 9453, Springer; 2015, p. 411–36. http://dx.doi.org/10.1007/978-3-662-48800-3_{1}{7}.

[7] Li L, Liu B, Wang H. QTL: A new ultra-lightweight block Cipher. Microprocess Microsyst 2016;45:45–55. http://dx.doi.org/10.1016/j.micpro.2016.03.011.

[8] Banik S, Pandey SK, Peyrin T, Sasaki Y, Sim SM, Todo Y. GIFT: a small present - towards reaching the limit of lightweight encryption. In: Fischer W, Homma N, editors. Cryptographic hardware and embedded systems - CHES 2017 - 19th international conference, taipei, Taiwan, September 25-28, 2017, proceedings. Lecture notes in computer science, vol. 10529, Springer; 2017, p. 321–45. http://dx.doi.org/10.1007/978-3-319-66787-4_{1}{6}.

[9] Beierle C, Leander G, Moradi A, Rasoolzadeh S. CRAFT: Lightweight tweakable block Cipher with efficient protection against DFA attacks. IACR Trans Symmetric Cryptol 2019;2019(1):5–45. http://dx.doi.org/10.13154/TOSC.V2019.I1.5-45.

[10] Guo Y, Li L, Liu B. Shadow: A lightweight block Cipher for IoT nodes. IEEE Internet Things J 2021;8(16):13014–23. http://dx.doi.org/10.1109/JIOT.2021.3064203.

[11] Yang J, Li L, Guo Y, Huang X. DULBC: A dynamic ultra-lightweight block Cipher with high-throughput. Integr 2022;87:221–30. http://dx.doi.org/10.1016/J.VLSI.2022.07.011.

[12] Huang X, Li L, Yang J. IVLBC: An involutive lightweight block Cipher for Internet of Things. IEEE Syst J 2023;17(2):3192–203. http://dx.doi.org/10.1109/JSYST.2022.3227951.

[13] Belkheyar Y, Daemen J, Dobraunig C, Ghosh S, Rasoolzadeh S. BipBip: A low-latency tweakable block Cipher with small dimensions. IACR Trans Cryptogr Hardw Embed Syst 2023;2023(1):326–68. http://dx.doi.org/10.46586/TCHES.V2023.I1.326-368.

[14] Song Q, Li L, Huang X. LELBC: A low energy lightweight block Cipher for smart agriculture. Internet Things 2024;25:101022. http://dx.doi.org/10.1016/j.iot.2023.101022.

[15] Zakaria AA, Azni AH, Ridzuan F, Zakaria NH, Daud M. Systematic literature review: Trend analysis on the design of lightweight block Cipher. J King Saud Univ Comput Inf Sci 2023;35(5):101550. http://dx.doi.org/10.1016/J.JKSUCI.2023.04.003.

[16] Biham E, Shamir A. Differential fault analysis of secret key cryptosystems. In: Lecture notes in computer science, Springer Berlin Heidelberg; 1997, p. 513–25. http://dx.doi.org/10.1007/bfb0052259.

[17] Kermani MM, Azarderakhsh R, Mirakhorli M. Multidisciplinary approaches and challenges in integrating emerging medical devices security research and education. In: 2016 ASEE Annual Conference &amp; Exposition Proceedings. ASEE Conferences; 2016, http://dx.doi.org/10.18260/p.25761.

[18] Kaur J, Canto AC, Kermani MM, Azarderakhsh R. Hardware constructions for error detection in WG-29 stream Cipher benchmarked on FPGA. IEEE Trans Comput-Aided Des Integr Circuits Syst 2024;1. http://dx.doi.org/10.1109/tcad.2023.3338108.

[19] Canto AC, Sarker A, Kaur J, Kermani MM, Azarderakhsh R. Error detection schemes assessed on FPGA for multipliers in lattice-based key encapsulation mechanisms in post-quantum cryptography. IEEE Trans Emerg Top Comput 2023;11(3):791–7. http://dx.doi.org/10.1109/tetc.2022.3217006.

[20] Lara-Nino CA, Diaz-Perez A, Morales-Sandoval M. Lightweight hardware architectures for the present Cipher in FPGA. IEEE Trans Circuits Syst I Regul Pap 2017;64-I(9):2544–55. http://dx.doi.org/10.1109/TCSI.2017.2686783.

[21] Pandey JG, Goel T, Karmakar A. Hardware architectures for PRESENT block Cipher and their FPGA implementations. IET Circuits Devices Syst 2019;13(7):958–69. http://dx.doi.org/10.1049/IET-CDS.2018.5273.

[22] Shahbazi K, Ko S. Area-efficient nano-AES implementation for Internet-of-Things devices. IEEE Trans Very Large Scale Integr Syst 2021;29(1):136–48. http://dx.doi.org/10.1109/TVLSI.2020.3033928.

[23] Li L, Feng J, Liu B, Guo Y, Li Q. Implementation of PRINCE with resource-efficient structures based on FPGAs. Front Inf Technol Electron Eng 2021;22(11):1505–16. http://dx.doi.org/10.1631/FITEE.2000688.

[24] Bharathi R, Parvatham N. Light-weight present block Cipher model for IoT security on FPGA. Intell Autom Soft Comput 2022;33(1):35–49. http://dx.doi.org/10.32604/iasc.2022.020681.

[25] Yang J, Li L, Huang X. Low area and high throughput hardware implementations for the LILLIPUT Cipher. Int J Circuit Theory Appl 2023. http://dx.doi.org/10.1002/cta.3892.

[26] Bao Z, Guo J, Ling S, Sasaki Y. PEIGEN–a platform for evaluation, implementation, and generation of S-boxes. IACR Trans Symmetr Cryptol 2019;330–94. http://dx.doi.org/10.46586/tosc.v2019.i1.330-394.

[27] Darzi S, Ahmadi K, Aghapour S, Yavuz AA, Kermani MM. Envisioning the future of cyber security in post-quantum era: A survey on PQ standardization, applications, challenges and opportunities. 2023, http://dx.doi.org/10.48550/ARXIV.2310.12037, CoRR abs/2310.12037, arXiv:2310.12037.

[28] Canto AC, Kaur J, Kermani MM, Azarderakhsh R. Algorithmic security is insufficient: A comprehensive survey on implementation attacks haunting post-quantum security. 2023, http://dx.doi.org/10.48550/ARXIV.2305.13544, CoRR abs/2305.13544, arXiv:2305.13544.

[29] Grover LK. A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on theory of computing - STOC '96. ACM Press; 1996, http://dx.doi.org/10.1145/237814.237866.

[30] Feng J, Wei Y, Zhang F, Pasalic E, Zhou Y. Novel optimized implementations of lightweight cryptographic S-boxes via SAT solvers. IEEE Trans Circuits Syst I Regul Pap 2023;1–14. http://dx.doi.org/10.1109/tcsi.2023.3325559.

[31] Turan MS, McKay K, Chang D, Bassham LE, Kang J, Waller ND, et al. Status report on the final round of the NIST lightweight cryptography standardization process. US: National Institute of Standards and Technology; 2023, http://dx.doi.org/10.6028/nist.ir.8454.