

基于 HRCA 的可重构 SM4 密码算法研究与实现

张骁, 周清雷, 李斌

(郑州大学信息工程学院, 河南 郑州 450001)

摘要: 针对同时要求高吞吐率和高安全性的应用场景, 提出了基于 HRCA 的高性能可扩展的 SM4 实现方案。首先, 通过分析对 SM4 提取出的不同粒度的基础算核, 提出了一个通用的粗粒度可重构计算单元; 然后, 为满足不同加密模式的需要给出了多种映射策略, 根据不同策略将算法映射到重构计算单元; 最后, 通过分割控制平面和数据平面优化 SM4 整体架构。实验结果表明, 使用所提方法, SM4 算法在较低的资源消耗下吞吐量有了明显提高。

关键词: HRCA; SM4 密码算法; 可重构计算; 高吞吐率

中图分类号: TP393

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2020064

Research and implementation of reconfigurable SM4 cryptographic algorithm based on HRCA

ZHANG Xiao, ZHOU Qinglei, LI Bin

School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China

Abstract: For the application scenarios that require high throughput and high security at the same time, based on HRCA, a high-performance scalable SM4 implementation scheme was proposed. Firstly, SM4 was formed to different calculation units of different granularity, and a general coarse-grained reconfigurable calculation unit was designed. Then, various mapping strategies were given for the purpose of satisfying different encryption modes, and the algorithm was mapped to the reconstructed computing unit according to different strategies. Finally, the SM4 overall architecture was optimized by dividing the control plane and the data plane. The experimental results show that using the proposed method, the SM4 algorithm has a significant improvement in throughput under lower resource consumption.

Key words: HRCA, SM4 cryptographic algorithm, reconfigurable computing, high-throughput

收稿日期: 2019-12-06; 修回日期: 2020-03-31

通信作者: 周清雷, ieqlzhou@zzu.edu.cn

基金项目: 国家重点研发计划基金 (2016YFB0800100); 国家重点研发计划重点专项基金

Foundation Items: The National Key Research and Development Program of China (2016YFB0800100), Key Special Fund of National Key Research and Development Program of China

论文引用格式: 张骁, 周清雷, 李斌. 基于 HRCA 的可重构 SM4 密码算法研究与实现[J]. 网络与信息安全学报, 2020, 6(5): 101-109.

ZHANG X, ZHOU Q L, LI B. Research and implementation of reconfigurable SM4 cryptographic algorithm based on HRCA[J]. Chinese Journal of Network and Information Security, 2020, 6(5): 101-109.

1 引言

开放的互联网环境必然带来诸多安全问题, 为保证网络中通信与数据交换的安全性, 通常将传输的数据进行加密保护。SM4 分组密码算法比国外使用的同类算法具有较高的安全性^[1-2]; 同时它作为国家密码管理局发布的密码行业标准, 适用于无线局域网和可信计算系统中, 主要对具有敏感性的内部信息、行政事务信息、经济信息等进行加密保护。

目前对 SM4 分组加密算法方面的研究主要有: 文献[3]提出了“一次一密”的加密机制, 主要解决 SM4 算法加解密使用相同的密钥并缺乏对密钥保护的问题, 从而提高整个算法的安全性; 文献[4]基于 SM4 的通用结构, 利用混合线性整数规划设计出一种较优的认证加密算法 SAME; 文献[5]结合 AES-GCM 实现了 SM4-GCM 方案, 通过乒乓操作提高数据处理速度, 并验证了方案具有更高的吞吐量和较低的资源消耗; 文献[6]提出了一种逻辑化简方法, 通过减少算法关键路径上时延, 设计出多轮合一的组合逻辑电路减少时延, 从而提高 SM4 算法在 CBC 模式下的吞吐率; 文献[7]针对 SM4-CTR 模式进行了优化, 实现了高吞吐率的 ASIC 设计。

随着处理数据量的增大和多种加解密模式的

出现, 上述方案无法完全满足大规模文本、物联网、云计算以及视频流媒体等复杂多变的应用场景对多种加密模式的需求。为了满足高速的网络传输环境和密码行业标准的要求, 本文基于可重构计算提出了适用于超混合可重构计算阵列 (HRCA, hybrid reconfigurable computing array) 上映射实现的 SM4 加密算法方案。首先, 基于可重构计算思想对 SM4 算法计算、存储与通信特征的分析后, 实现了粗粒度的可重构计算单元。接着, 对 SM4 进行多种映射展开以适应不同的加密工作模式, 并对 SM4 整体架构进行了优化设计, 提高了算法的灵活性与可扩展性, 在保证高性能与安全性的前提下满足了不同应用场景的需要。

2 研究背景

2.1 SM4 算法原理

SM4 迭代分组密码算法采用非平衡 Feistel 结构, 数据分组长度均为 128 bit, 加解密算法与密钥扩展算法均为 32 轮非线性迭代结构, 加密密钥和解密密钥相同^[8]。SM4 算法加密结构与轮密钥结构如图 1 所示。

SM4 算法包含轮函数 F 、合成置换 T 、非线性变换 τ 、线性变换 L 以及逆序转换 R 等函数过程。设明文输入为 $(X_0, X_1, X_2, X_3) \in (Z_2^{32})^4$, 密文

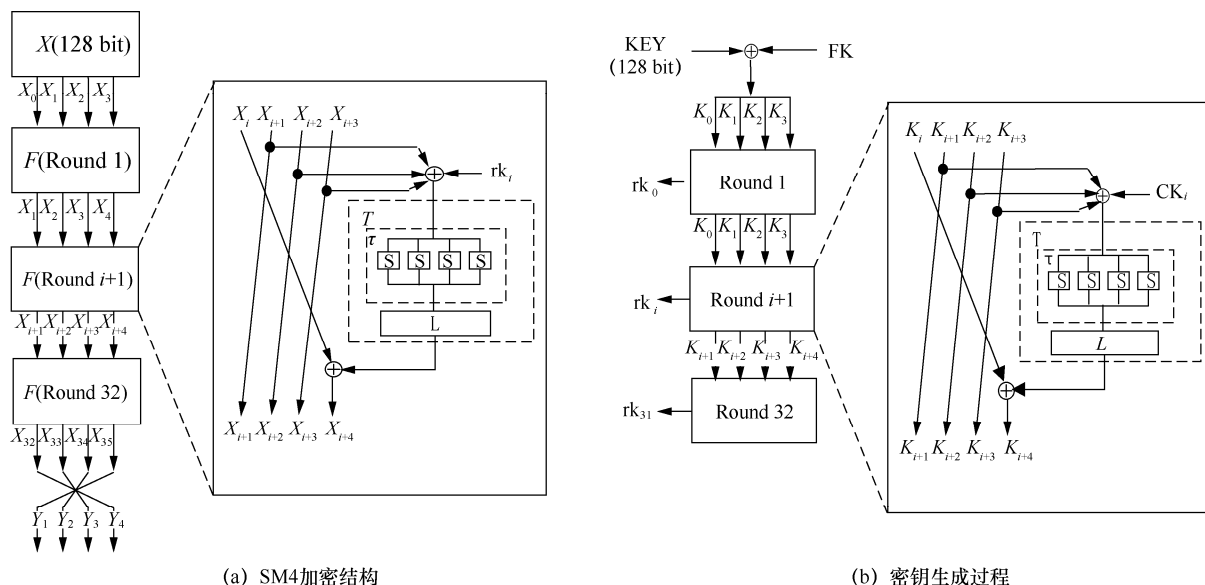


图 1 SM4 分组密码算法与密钥扩展算法
Figure 1 SM4 block cipher algorithm and key expansion algorithm

输出为 $(Y_0, Y_1, Y_2, Y_3) \in (Z_2^{32})^4$ ，加密密钥为 $(MK_0, MK_1, MK_2, MK_3), MK_i \in Z_2^{32}$ ，轮密钥为 $(rk_0, rk_1, rk_2, \dots, rk_{31}) \in (Z_2^{32})^4$ 。SM4 算法的解密结构与加密结构相似，但两者轮密钥的使用次序不同，加密时轮密钥使用次序为 $(rk_0, rk_1, rk_2, \dots, rk_{31})$ ；解密时逆序使用轮密钥。SM4 的加密原理描述如下。

$$\begin{aligned} X_{i+4} &= F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i) = \\ &X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus rk_i), \\ i &= 0, 1, \dots, 31 \\ (Y_0, Y_1, Y_2, Y_3) &= R(X_{32}, X_{33}, X_{34}, X_{35}) = \\ &(X_{35}, X_{34}, X_{33}, X_{32}) \end{aligned}$$

其中， T 为合成置换，由非线性变换 τ 和线性变换 L 复合而成，即 $T(\cdot) = L(\tau(\cdot))$ 。线性变换 L 由移位操作完成，即 $L(B) = B \oplus (B \lll 2) \oplus (B \lll 10) \oplus (B \lll 18) \oplus (B \lll 24)$ ；非线性变化 τ 由 S 盒扰乱构成，输入 $A \in (Z_2^{32})^4$ ，输出 $B \in (Z_2^{32})^4$ ，非线性变换可以描述为

$$\begin{aligned} B &= (b_0, b_1, b_2, b_3) = \\ \tau(A) &= (Sbox(a_0), Sbox(a_1), Sbox(a_2), Sbox(a_3)) \end{aligned}$$

轮密钥由加密密钥通过密钥扩展算法得到，FK 与 CK 均为固定参数组，用于密钥扩展算法。SM4 密钥扩展算法公式如下。

$$\begin{aligned} (K_0, K_1, K_2, K_3) &= (MK_0 \oplus FK_0, MK_1 \oplus \\ &FK_1, MK_2 \oplus FK_2, MK_3 \oplus FK_3) \\ rk_i &= K_{i+4} = K_i \oplus T'(K_{i+1} \oplus K_{i+2} \oplus K_{i+3} \oplus CK_i), \\ i &= 0, 1, \dots, 31 \end{aligned}$$

T' 与加密算法轮函数 T 基本相同，只是线性变换移位长度不同，为

$$L'(B) = B \oplus (B \lll 13) \oplus (B \lll 23)$$

2.2 超混合可重构计算阵列原理

由于研究角度不同，人们对于可重构计算 (RC, reconfigurable computing) 的定义不尽相同。在 1999 年 ACM 国际会议上，由美国加州大学伯克利分校可重构技术研究中心的 Dehon 等^[9]提出的定义是目前比较公认的。可重构计算结构综合软件的灵活性和硬件的高效性，利用生产加工后结构上的空间特性优化算法，通过对可重构处理

单元硬件资源进行编程配置可以实现算法到计算引擎的空间映射，进而可以从时间和空间多维度上对计算任务进行展开。重构粒度是指可重构系统中映射工具所能寻址最小的处理单元数据通路的位宽^[9]。依据可重构单元的不同粒度，可重构计算系统可依次分为细粒度、粗粒度和混合粒度可重构计算系统。

HRCA 是一种新提出的可重构阵列，能够以可重构方式支持不同精度和多种聚合单位的变化^[10]。HRCA 具有应用驱动、非指令执行和高度可扩展等特点。其中，应用驱动是指通过对具体应用的分析从算法中提取可共用的基本操作，并形成在硬件上易于实现的基础算核集，在此基础上构建各种粒度的可重构单核；非指令执行与通用处理器结构对算法的执行过程有所区别，将算法直接映射到硬件结构中实现，即粗细粒度单元在单核内以硬件可重构方式实现算法中的基础算核集，减少了通用处理器取指译指的开销，进而大幅提高执行效率；高度可扩展是指核间采用全局异步、核内采用局部同步的互联通信方式，可实现以单核为基础的高度可扩展。HRCA 结构如图 2 所示，HRCA 中可重构单元构成的更大规模硬件结构 Re_Eng 形成了可以支持应用算法各子模块的高阶运算栈。对比通用处理器结构，在 HRCA 中具有一定相似度的基本操作由不同粗粒度的高效可重构计算单元完成，各计算单元间的互联结构支持具有并行特点的数据流，使高阶运算栈在计算过程执行上效率更高，因此通过这些结构可以支持各种应用的核心算法在计算过程中高效实现。

3 基于 HRCA 的可重构 SM4 实现

3.1 基础算核的提取

为满足高性能可重构计算阵列设计需求，需要对复杂的算法和计算部件进行分析，在一定条件下将复杂计算问题分解成若干简单的计算，并刻画出适合的粗粒度计算可复用算核，减少冗余的资源开销，提高算法的映射效率。统计目前公开文献中分组密码算法的基本操作，可以发现分组算法仅由有限的几种操作构成^[11]。这给算核提取与可重构计算单元的构建提供了便利。

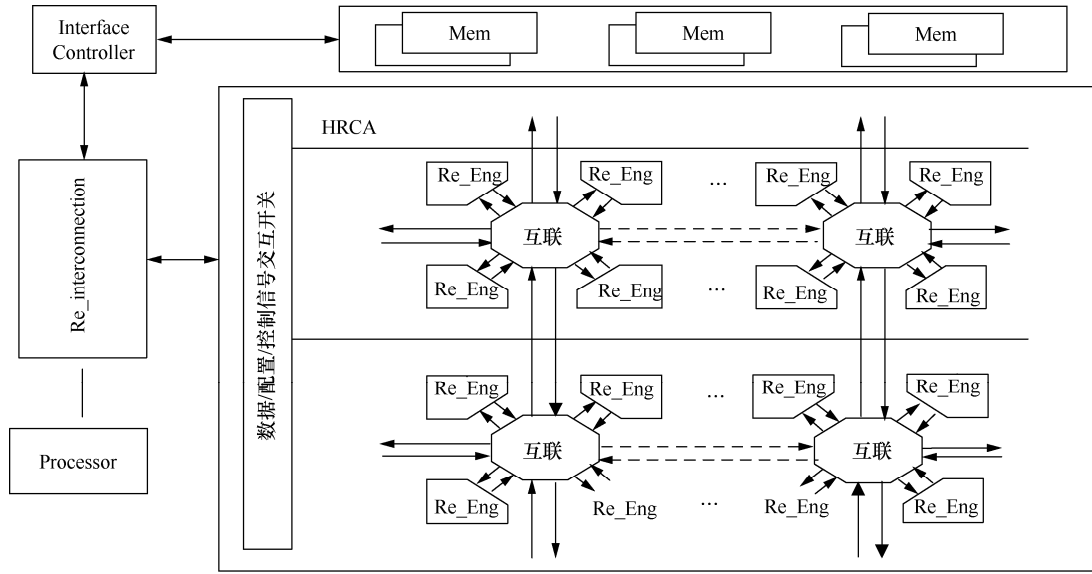


图 2 HRCA 结构
Figure 2 HRCA structure

作为构建 HRCA 中关键的算核提取过程, 在所选算法满足可重构的前提下, 保证提取出的算核在重构过程中被使用概率基本一致, 基础算核集提取过程如下。

令 HRCA 芯片上总资源为 R , 基础算核集表示 $CK = \{b_1 \cdot CK_1, b_2 \cdot CK_2, \dots, b_n \cdot CK_n\}$, 其中, b_n 为算核个数。 CK_i 所需要资源为 r_i , 则重构算核所需要资源即为 $r = \sum_{i=1}^k b_i r_i$ 。算法重构时所需要连接资源记为 ra 。算核 CK_i 被算法调用时表示为

$$u_{i,l} = \begin{cases} l=1, & \text{调用} \\ l=0, & \text{空闲} \end{cases}$$

则算核 CK_i 的利用率为

$$p_i = \frac{\sum_{l=1}^n u_{i,l}}{\sum_{i=1}^k b_i}$$

合理的基础算核集在满足重构约束 $(\sum_{i=1}^k b_i r_i + ra) \leq R$ 条件下, 算核被使用的概率应保持

基本一致, 即算核利用率方差 $\min \sum_i \left(p_i - \frac{\sum_i p_i}{\sum_{i=1}^n b_i} \right)^2$ 。

求解基础算核集的算法是一个迭代过程, 从算法各步骤所需算核集的简单并集出发, 调整使用算核集合, 使算核利用率的方差最小。

根据基础算核提取过程, 在 SM4 算法中提取出以下基础算核: S 盒的查找表 (LUT), 以 S 盒内存放的数值替换相对应的输入, 查找表通过互联方式使其他运算单元共享使用; 移位操作包含对多种移位位数的处理; 逻辑计算主要用于完成对多组相同位宽的输入数据进行异或运算; 比特位操作用于对数据进行拼接和拆分, 使数据分组的位宽符合算法要求。SM4 算法主要基础算核如表 1 所示。

表 1 SM4 主要基础算核
Table 1 Basis computing kernel of SM4

算核	类型	数量	运算位宽/bit
比特位操作	拼接、拆分	3	128, 32, 8
逻辑操作	异或	21	32
S 盒		8	32
移位	循环左移	6	32

注: 空白表示不适用。

3.2 粗粒度可重构单元

粗粒度的可重构体系结构主要针对计算密集型应用, 一般要求配置的最小数据位宽大于 4 bit, 可以并行处理大量算数运算。同时, SM4 作为分

组加密算法具有如下特性: 运算类型集中, 加解密过程主要通过轮函数运算完成; 循环迭代为主, 需要经过 32 轮完全相同的轮函数运算; 密钥扩展过程与加解密过程结构基本相同且形式固定。因此, 在提取相关算核的基础上, 除部分移位操作外, 可以对算法中关键操作进行重构, 使用相似的结构与运算单元来实现算法。本文将 SM4 算法中每一次轮函数单元计算构建为粗粒度重构运算单元^[12]。

粗粒度可重构运算单元作为可重构的适用于 HRCA 的高阶运算栈, 包含逻辑功能单元、选择器、寄存器组、算术逻辑单元和移位异或选择单元, 其基本结构如图 3 所示, 主要功能如下。

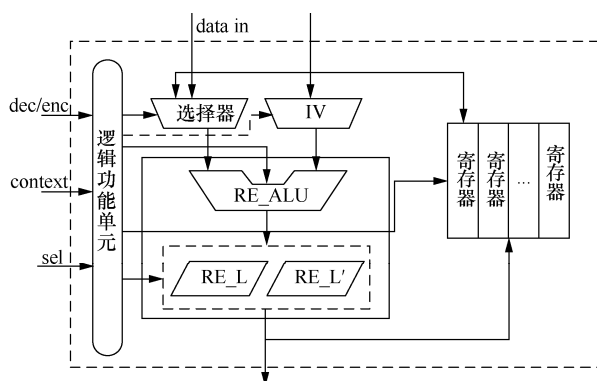


图3 粗粒度可重构运算单元
Figure 3 Coarse-grained reconfigurable computing unit

(1) 逻辑功能单元: 作为控制配置中心负责接收阵列中的配置信息, 主要执行以下几个任务。一是控制粗粒度可重构单元待执行的运算功能; 二是确定片上存储器和主存储器之间的运算类型以及数据的流向, 通过不同的配置条件确定任务特征及部件状态, 将任务分配到匹配的部件或子结构上计算; 三是允许配置 IV 寄存器使算法工作在不同的加密模式下。

(2) 选择器: 用来处理输入数据的拆分与拼接, 拆分成 32 位数据字长并输入算术逻辑单元进行计算。

(3) 寄存器组: 作为片上阵列中的存储单元为运算部件提供加密算法所需的系统参量和固定参量, 同时存储一些必要的中间结果。

(4) 重构的算术逻辑单元 RE_ALU 与异或移位单元 RE_L: 作为依据算法处理流程中提取出的基础算核所重构而成的逻辑部件, 是完成 SM4 算法运算的主要功能模块。RE_ALU 主要完成

32 位加减法、逻辑异或运算等基本算术和逻辑操作, 并通过查找表的方式对 S 盒访问, 将计算数据作为 SBOX 存储器的地址, 并将相应地址中的内容作为输出, 从而在算法中实现数据的替换功能。移位异或选择单元 RE_L 和 RE_L' 在完成基本的逻辑异或操作基础上分别完成不同位长的移位操作。

粗粒度可重构运算单元的运算周期为一个时钟, 运算单元按照运算步骤划分算法, 确保了多个重构单元可以串行或并行地映射到流水线上, 达到提高计算性能和效率的目的。可重构单元作为 HRCA 中的高阶运算栈 Re_Eng 采用交叉的可重构互联结构, 充分利用单元件的程序执行局部性, 提高系统的灵活性与可扩展性。

3.3 SM4 算法结构的映射策略

SM4 作为分组密码算法拥有多种工作模式, 其中, CBC 模式和 CTR 模式具有较高的安全性^[13]。CBC 模式中存在数据依赖性, 不适于使用流水线并行结构, 而 CTR 模式适合数据分组, 不存在依赖性, 可并行实现^[14-15]。为了同时满足 CBC 和 CTR 等加密模式的 SM4 分组算法的运行, 本文按照不同策略对算法分别进行循环串行展开映射以及流水线并行展开映射以提高算法在不同工作模式下的处理性能, 同时降低对硬件资源的占用。构建的粗粒度可重构计算单元作为算法中的关键循环块, 将运算分配到这些可重构计算单元上进行执行。

3.3.1 循环迭代映射

SM4 完成一组数据的加解密需要 64 轮迭代, 其中 32 轮是对轮密钥 rk 进行计算。每一轮迭代所需运算时间可以描述为 $T_F + T_{reg}$, 其中 T_F 代表轮函数运算时间, T_{reg} 代表访存时间, 经过加解密运算总共的耗时为 $64T_F + 64T_{reg}$ 。本文将轮函数定义成粗粒度可计算单元, 在硬件平台上可以在一个时钟周期完成运算。因此完成一组数据加密共需要 128 个时钟周期。为加速运算, 通过合并粗粒度可重构计算单元, 形成如图 4 所示逻辑结构, 使两次轮计算可以在一个时钟周期内完成。

这种结构下完成一组数据的加解密只需要进行原来轮计算次数的一半, 即 32 次轮计算, 则总耗时为 $32T_F + 32T_{reg}$ 。这种映射方式中, 轮函数在

但中间结果需要大量 FIFO 进行存储。

具体分析两种流水线映射方式带来的性能提高, 主要包括以下两个方面。

(1) 流水线的使用可以减少系统运算部件的空闲时间。由于 SM4 需要经过密钥扩展流程得到轮密钥才能进行加密过程, 得到一组完整的加解密数据的时延至少是 64 个周期, 其中 32 个周期用来进行轮密钥计算, 另外 32 个周期进行加解密过程。这就造成一半的时间主要运算模块处于等待状态。用流水线方法映射 SM4 算法, 可减少运算单元的等待时间, 大幅度提高资源利用效率。

(2) 减少通信量。如果运算单元按照顺序执行时, 系统需要为这些模块的数据预留很多空间来存储中间运算结果, 同时在时序上增加了中间数据的存储与读取。采用流水线的方式并行处理各个运算单元可以减少中间数据存储和通信的开销。

3.4 整体算法优化

为了使 FPGA 芯片满负荷工作, 基于控制算法细化、模块化和最佳适用性原则, 本文按照所选择硬件架构对整体算法进行设计, 主要划分为控制平面的上层控制和数据平面的核心算子两部分完成。

SM4 整体结构如图 7 所示, 主要包括控制平面的轮密钥扩展控制与加解密控制以及数据平面的密钥扩展模块与加解密模块。加解密模块仅用控制命令确定算法进行加密操作还是解密操作。控制平面通过实例化多个核心算子以及放置多条

流水线以增加芯片的利用率, 并对各算子进行资源的分配回收与任务的分发调度, 同时可以控制加解密模式, 选择算法运行在串行或流水线结构中。数据平面的核心算子分为密钥扩展模块和加解密模块, 这些模块主要由粗粒度可重构计算单元组成, 模块间可以进行数据传输。SM4 算法作为计算密集型任务在数据平面通过可重构计算单元高速实现, 使管理控制与数据间存在不同的吞吐量和复杂度。因此, 通过控制平面与数据平面的分割避免速度差异导致算法整体性能降低。

4 实验与性能对比

为评估本文提出的粗粒度重构计算单元与多种展开方案并验证性能与资源开销, 本文使用 Verilog HDL 硬件描述语言对以上方案进行描述, 然后使用 XILINX 公司的 Vivado2018.2 进行功能和时序仿真验证算法功能的正确性, 并在 FPGA 集成加速卡进行实现, 芯片型号为 XILINX 公司的 XCKU060。

4.1 资源占用

表 2 给出了本文各种映射方案的资源开销情况。从表中可以看出, 循环迭代映射与流水线映射中, 流水线并行策略消耗资源多, 因为后者为达到可用的流水线级数需要的可重构计算单元较多。在全流水线并行和数据分配并行中, 后者每一级流水线都需要寄存器存放中间结果, 因此所需资源明显增多。

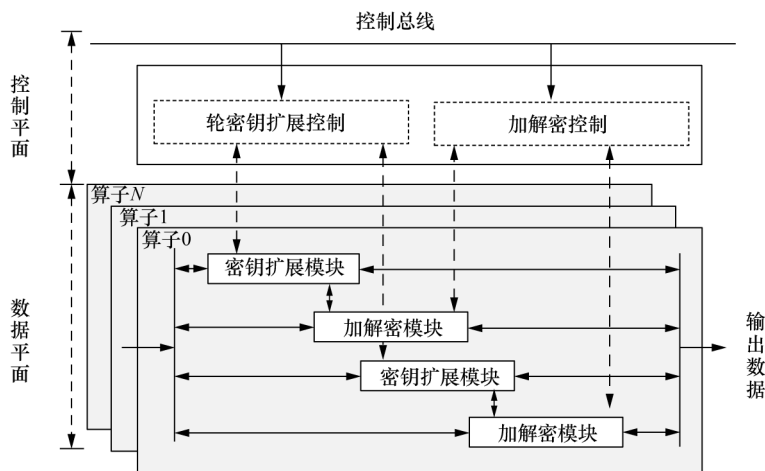


图 7 SM4 整体结构

Figure 7 Overall structure of SM4

表 2 不同方案资源开销情况
Table 2 Resources utilization of different schemes

映射方案	类型	lut	reg
循环迭代映射	操作合并串行	1 087	1 365
	全流水线并行	14 243	8 226
流水线映射	数据分配并行	15 768	12 649

4.2 性能对比

本文在对 SM4 进行重构的基础上,通过多种映射策略对算法进行实现。表 3 给出了本文所有映射策略的性能。从表中看出,在流水线展开策略有较高的吞吐率,可达到 25.6 Gbit/s。

表 3 不同方案性能分析
Table 3 Performance analysis of different schemes

类型	频率/MHz	吞吐率/(Gbit·s ⁻¹)	功耗/W
操作合并串行	300	1.2	0.852
全流水线并行	200	25.6	458.317
数据分配并行	200	25.6	479.406

为了尽可能利用芯片资源,在 200 MHz 频率下对循环迭代映射和流水线映射方案分别实例化了 41 个 SM4 算子和 20 个 SM4 算子,占用资源分别为 82%和 80%。为了说明本文基于可重构计算单元进行映射在性能方面的提高,给出了与其他文献提出的方法的对比情况,如表 4 所示。

表 4 不同文献性能与资源对比
Table 4 Comparison of performance and resources with other references

对比文献	算法模式	吞吐率/(Gbit·s ⁻¹)	频率/MHz	资源开销
文献[5]	GCM	32.1	250.6	10 609 lut
文献[6]	4in1-CBC	5.24		126 744 μm ²
文献[7]	CTR	14.647	114.43	7 423ALMs
文献[14]	SM4	10.083		
本文	循环迭代串行	49.2	300	1 087 lut
	全流水线加密	512	200	14 243 lut

注:空白表示不适用。

5 结束语

当前 SM4 算法的应用主要集中在专用 ASIC 和 CPU 这两种平台上实现,然而,随着应用需求的不断发展与极速变化,应用使用 SM4 算法进行

身份认证、系统自检以及数据加密等不同的场景阶段所需加密的数据量与工作模式有所不同,上述两种平台不能平衡性能与灵活性的需求。针对这些问题,本文基于超混合可重构计算阵列对 SM4 算法进行了算粒集提取和粗粒度可重构单元的设计,同时对算法按照不同的策略进行了展开,结合 FPGA 并行性和灵活可编程的特性,通过状态机实现减少访存时间,使算法处理速度进一步提高,兼顾了高性能与灵活性。通过实验分析了不同映射方案对性能和资源的影响,结果表明,对于 CBC 模式使用的串行运算和 CTR 模式使用的流水线运算,本文通过采用重构设计和映射方案,在单算子和多算子情况下对算法的运行效率有不同程度的提高。本文方案可以加快数据的加解密速度,并通过配置可在硬件平台下实现多模式转换,在要求算法具备多种加密工作模式的复杂应用场景中具有重要的意义。

参考文献:

- [1] KOCHER P, JAFFE J, JUN B, et al. Introduction to differential power analysis [J]. Journal of Cryptographic Engineering, 2010, 1(1): 5-27.
- [2] JI W, HU L, OU H. Algebraic attack to SMS4 and the comparison with AES[C]//International Conference on Information Assurance & Security. 2009: 662-665.
- [3] 艾心, 吴鸣旦, 武旭东, 等. 一次一密 SM4 算法的设计[J]. 网络安全, 2018, 9(2): 20-23.
- [4] AI X, WU M D, WU X D, et al. Design of one-time pad SM4 algorithm[J]. Cyberspace Security, 2018, 9(2): 20-23.
- [5] 张建, 吴文玲. 基于 SM4 轮函数设计的认证加密算法[J]. 电子学报, 2018, 46(6): 1294-1299.
- [6] ZHANG J, WU W L. Authenticated encryption based on SM4 round function[J]. Acta Electronica Sinica, 2018, 46(6): 1294-1299.
- [7] LI L, YANG F, PAN Y, et al. An implementation method for SM4-GCM on FPGA[C]//IEEE Advanced Information Technology, Electronic & Automation Control Conference. 2017: 1921-1925.
- [8] 符天枢, 李树国. SM4 算法 CBC 模式的高吞吐率 ASIC 实现[J]. 微电子学与计算机, 2016, 33(10): 13-18.
- [9] FU T S, LI S G. A high-throughput ASIC Implementation of SM4 algorithm in CBC mode[J]. Microelectronics & Computer, 2016, 33(10): 13-18.
- [10] 王泽芳, 唐中剑. SM4 算法 CTR 模式的高吞吐率 ASIC 实现[J]. 电子器件, 2019, 42(1): 177-181.
- [11] WANG Z F, TANG Z J. A high-throughput ASIC implementation of SM4 algorithm in CTR mode[J]. Chinese Journal of Electron

- Devices, 2019, 42(1): 177-181.
- [8] 吕述望, 苏波展, 王鹏, 等. SM4 分组密码算法综述[J]. 信息安全研究, 2016, 2(11): 995-1007.
- LYU S W, SU B Z, WANG P, et al. Overview on SM4 algorithm[J]. Journal of Information Security Research, 2016, 2(11): 995-1007.
- [9] DEHON A, WAWRZYNEK J. Reconfigurable computing: what, why, and implications for design automation[C]//Proceedings 1999 Design Automation Conference. 1999.
- [10] 陈伟男. 基于 FPGA 的可重构系统及 CAD 技术研究[D]. 上海: 复旦大学, 2009.
- CHEN W N. Research on reconfigurable system and CAD technology based on FPGA[D]. Shanghai: Fudan University, 2009.
- [11] ELBIRT A J. Reconfigurable computing for symmetric-key algorithms[J]. 2003.
- [12] 何琪辰, 沈文枫, 孙思齐, 等. 超混合深度可重构计算阵列调度策略的优化研究[J]. 计算机应用与软件, 2014(6): 284-287, 313.
- HE Q C, SHEN W F, SUN S Q, et al. Study on scheduling policy optimisation based on heterogeneous reconfigurable computing array[J]. Computer Applications and Software, 2014(6): 284-287, 313.
- [13] DWORKIN M. Recommendation for block cipher modes of operation: methods and techniques[J]. National Institute of Standards & Technology, 2001, 5(6):669-675.
- [14] 徐金甫, 杨宇航. SM4 算法在粗粒度阵列平台的并行化映射[J]. 电子技术应用, 2017, 43(4):39-42.
- XU J F, YANG Y H. The parallel mapping implementation of SM4 based on coarse-grained reconfigurable block encryption array[J]. Application of Electronic Technique, 2017, 43(4):39-42.
- [15] 杨子煜, 严明, 王大伟, 等. 面向 CGRA 循环流水映射的数据并行优化[J]. 计算机学报, 2013, 36(6): 1280-1289.

YANG Z Y, YAN M, WANG D W, et al. Data parallelism optimization for the CGRA loop pipelining mapping[J]. Chinese Journal of Computers, 2013, 36(6): 1280-1289.

[作者简介]



张骁(1993-), 男, 河南新乡人, 郑州大学硕士生, 主要研究方向为信息安全、高性能计算。



周清雷(1962-), 男, 河南新乡人, 博士, 郑州大学教授、博士生导师, 主要研究方向为信息安全、自动机理论及计算机复杂性理论。



李斌(1986-), 男, 河南郑州人, 博士, 郑州大学讲师, 主要研究方向为信息安全、高性能计算。