计算机研究与发展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202331007
61(6): 1450−1457, 2024

# 国密 SM4 算法 CBC 模式的高效设计与实现

郝泽钰[1,2] 代天傲[1,2] 黄亦成[1,2] 段岑林[1,2] 董 进[2,3] 吴世勇[2,3] 张 博[2,3] 王雪岩[1,2] 贾小涛[1,2] 杨建磊[1,2]

[1](北京航空航天大学计算机学院 北京 100191)

[2](北京市未来区块链与隐私计算高精尖创新中心 北京 100191)

[3](北京微芯区块链与边缘计算研究院 北京 100080)

# Efficient Design and Implementation of SM4 Algorithm with CBC Mode

Hao Zeyu[1,2], Dai Tianao[1,2], Huang Yicheng[1,2], Duan Cenlin[1,2], Dong Jin[2,3], Wu Shiyong[2,3], Zhang Bo[2,3], Wang Xueyan[1,2], Jia Xiaotao[1,2], and Yang Jianlei[1,2]

[1](*School of Computer Science and Engineer*, *Beihang University*, *Beijing* 100191)

[2](*Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing*, *Beijing* 100191)

[3](*Beijing Academy of Blockchain and Edge Computing*, *Beijing* 100080)

**Abstract** Among various cryptographic algorithms, the SM4 block cipher stands out for its simplicity and efficiency, particularly when implemented on hardware. Consequently, it has found widespread applications in encrypted transmission, encrypted storage, and beyond. As the utilization of the SM4 algorithm continues to grow, the necessity for superior hardware encryption capabilities is also increased. Recently, the implementation of the SM4 algorithm on ASIC has demonstrated high throughput in electronic code book (ECB) mode, thanks to the utilization of pipelining technology. However, in cipher block chaining (CBC) mode, achieving similar throughput improvements through pipelining is challenging due to the dependency among adjacent data blocks. To tackle this issue, we introduce two innovative simplification techniques, applied to the round function iteration process and S-box substitution process respectively. ASIC synthesis results using TSMC 40 nm technology confirm that our design achieves a throughput rate of 4.2 Gb/s in CBC mode, with a remarkable throughput of 129.4 Gb·s$^{-1}$·mm$^{-2}$, outperforming previously published designs in this domain.

**Key words** SM4 algorithm; CBC mode; hardware acceleration; high-efficiency design; ASIC

**摘 要** 密码技术是现代信息安全技术产业发展的核心，其中，国密 SM4 分组密码算法因其硬件实现简单、效率高等优点，已广泛应用于加密传输、加密存储等领域. 随着应用领域的不断扩展，对硬件加密效率的需求也随之提高. 目前，借助流水线技术，基于 ASIC 实现的 SM4 算法在 ECB（electronic code book）工作模式下能够达到较高的吞吐量. 然而，在 CBC（cipher block chaining）模式下，由于相邻的数据存在依赖关系，流水线技术难以提高硬件设计的吞吐率. 为解决这一问题，提出了 2 种逻辑化简方法：一种作用于轮函数迭代过程，另一种作用于 S 盒置换过程. 这 2 种方法在每一轮迭代的关键路径中均减少了 2 个异或运算的延时. 在 TSMC 40 nm 工艺下的 ASIC 综合结果表明，该设计在 CBC 模式下的吞吐率达到 4.2 Gb/s，单位面积吞吐量达 129.4 Gb·s$^{-1}$·mm$^{-2}$，高于已发表的同类设计.

**关键词** 国密 SM4 算法；CBC 模式；硬件加速；高效设计；ASIC

**收稿日期**：2023-12-14；**修回日期**：2024-03-13

**基金项目**：国家自然科学基金项目(62072019)

This work was supported by the National Natural Science Foundation of China (62072019).

**通信作者**：杨建磊(jianlei@buaa.edu.cn)

**中图法分类号** TP302

国密 SM4 算法[1]是一种常用的分组密码算法，广泛应用于数据保护、加密通信等领域. SM4 算法常见工作模式有 ECB(electronic codebook)，CBC(cipher block chaining)等，对于相同的明文块，ECB 模式下会产生完全相同的密文，而在 CBC 模式下，当前的明文块会与前一块的密文异或后进行运算. 因此，即使是完全相同的明文输入也可能会有完全不同的密文输出. 相比于 ECB 模式，CBC 模式提供了更高的安全性和抵抗攻击的能力，有着更高的应用需求. 提高 SM4 算法在 CBC 模式下的性能，对于在边缘设备中使用 SM4 算法是至关重要的. 但是，在 CBC 模式下存在着难以提高吞吐率的问题：每组的输入必须等待前一组运算结束后才能获得，因而难以使用流水线方法提升吞吐率.

文献 [2] 中提到了一种改进方法，将电路中的 S 盒以外的其他逻辑结构进行预计算，并把预计算的结果与 S 盒进行融合构成新的查找表，从而提高 SM4 算法在 CBC 模式下吞吐率. 本文基于此工作进一步优化了 S 盒的表示，并针对轮函数的迭代过程进行了优化，最终减少了轮函数关键路径上的 2 次异或运算，有效提高了算法的性能.

本文的设计针对 CBC 模式下的 SM4 算法，在 TSMC 40 nm，SMIC 55 nm 工艺下，使用 Synopsys Design Compiler 分别进行了 ASIC 综合. 综合结果显示，本文所提出的设计在 CBC 模式下的吞吐率达到了 4.2 Gb/s，同时单位面积吞吐量达到了 129.4 $\mathrm{Gb\cdot s^{-1}\cdot mm^{-2}}$，明显优于已发表的类似设计. 这些结果表明本文所提出的化简方法在改进 SM4 算法性能方面具有很大的潜力.

本文的结构为：首先介绍了 SM4 算法及其在 CBC 模式下存在的性能瓶颈问题. 然后，详细描述了本文提出的 2 个化简方法，并解释了它们在轮函数迭代和 S 盒置换过程中的作用. 接下来，介绍了实验设计并给出了实验结果分析和对比. 最后，对进一步改进和应用的方向进行了展望.

## 1 SM4 算法介绍

SM4 算法是一种对称密钥密码算法，被广泛应用于数据加密和保护领域，它是中国密码算法的标准之一，具有较高的安全性和良好的性能.

SM4 采用了分组密码的设计思想，将明文数据划分为 128 b 的数据块，并通过密钥对每个数据块进行加密和解密操作. 对单组数据进行加解密的流程如图 1 所示，分为密钥扩展算法和加解密算法 2 部分. 图 1 中的 $FK$ 是系统预设的参数，与用户密钥进行异或运算后作为密钥扩展算法的输入. 加解密算法接受密钥扩展算法产生的 32 轮轮密钥 $rk_i$ 对明文进行加解密，最后经反序变换输出. 加解密使用的是同一套计算流程，唯一的区别是解密时使用轮密钥的顺序与加密过程相反.

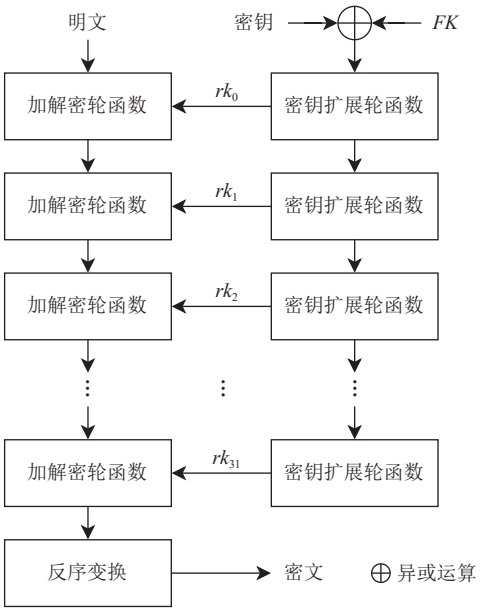


Fig. 1 Workflow of SM4 algorithm

图 1 SM4 算法工作流程

密钥扩展算法和加解密算法 2 部分均由 32 次轮函数迭代构成，整体结构均采用 4 路并行的 Feistel 结构，在计算过程中，以 128 b 数据为输入、128 b 数据为输出，其内部的运算逻辑如图 2 所示. 输出中的前 96 b 数据等于输入中的后 96 b 数据，输出后的 32 b 数据通过轮函数运算产生.

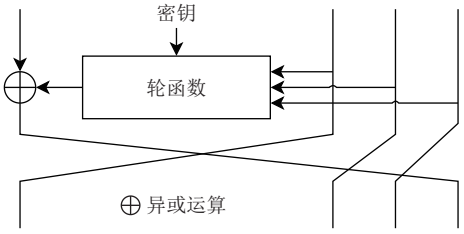


Fig. 2 Four parallel Feistel structure

图 2 4 路并行的 Feistel 结构

在密钥扩展算法中使用的密钥是算法给定的固定密钥，记作 $ck_i$. 在加解密算法中使用的密钥是由密

钥扩展算法通过用户给的密钥扩展出来的轮密钥，记作$rk_i$.

### 1.1 SM4 密钥扩展算法

SM4 密钥扩展算法结构如图 3 所示，密钥扩展的主要过程包括 32 轮密钥扩展的轮函数，其中，密钥为 128 b，$FK$为 SM4 标准中规定的一个 128 b 常数. 二者异或后的值将会作为密钥扩展轮函数的首轮输入，并通过一个选择器进行循环迭代，总计迭代 32 轮产生 32 个轮密钥.

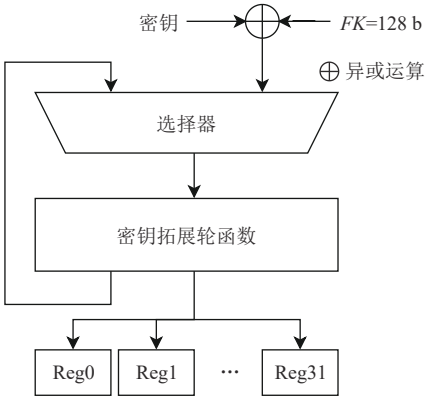


Fig. 3 Key expansion algorithm structure of SM4

图 3 SM4 的密钥扩展算法结构

设用户输入的密钥为$MK$，则该密钥对应的 32 轮轮密钥可以按照式(1)求出：

$$\begin{cases}(k_0,k_1,k_2,k_3)=MK\oplus FK,\\ k_{i+4}=k_i\oplus F(k_{i+1}\oplus k_{i+2}\oplus k_{i+3}\oplus ck_i),\\ rk_i=k_{i+4},\end{cases}\tag{1}$$

其中，$ck_i$是系统预设的 32 b 参数，$rk_i$代表第$i$轮的轮密钥，$F$代表密钥扩展轮函数，其由 S 盒置换算法$\tau:Z_2^{32}\to Z_2^{32}$和线性变换算法$L(x)=x\oplus(x<<<13)\oplus(x<<<23)$组成，其中<<<表示循环左移运算.

### 1.2 SM4 加解密算法

SM4 算法的加解密算法的整体结构与密钥扩展算法类似，均包含 32 轮的轮函数迭代，区别在于加解密算法中额外包含 1 次反序变换.

SM4 算法的轮函数迭代流程如图 4 所示，$X_1$~$X_4$为第 1 轮的输入，$X_2$~$X_5$为第 1 轮的输出，同时也是第 2 轮的输入. $rk_1$为第 1 轮的轮密钥，$T$函数代表加解密模块的轮函数. 与密钥扩展部分的轮函数$F$类似，由 S 盒置换算法$\tau$和一个线性变换算法$L'(x)=x\oplus(x<<<2)\oplus(x<<<10)\oplus(x<<<18)\oplus(x<<<24)$组成.

## 2 对 SM4 加解密算法关键路径的化简

通过多轮的迭代过程，SM4 算法能够实现高强度的数据加密和解密. 然而，在 CBC 模式下，由于相邻数据之间的依赖关系，传统的流水线技术难以提高算法的吞吐率. 因此，针对这一问题，本文提出了 2 种化简方法，以减少关键路径上的运算，从而提高 SM4 算法在 CBC 模式下的性能.

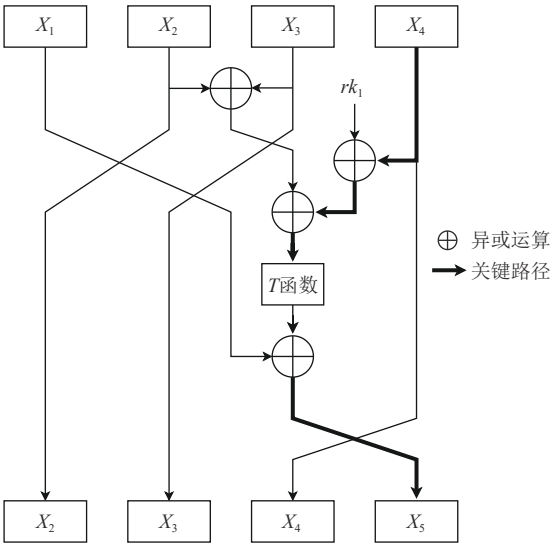


Fig. 4 Round function structure of SM4 encryption and decryption modules

图 4 SM4 加解密模块轮函数结构

### 2.1 轮函数优化

加解密模块的轮函数的结构如图 4 所示，若不考虑$T$函数带来的时序延迟，单次轮函数迭代的关键路径上共包含 3 次异或运算. 以公式的形式描述 SM4 算法加解密轮函数的迭代关系可得到式(2)：

$$X_{i+4}=X_i\oplus(X_{i+1}\oplus X_{i+2}\oplus X_{i+3}\oplus rk_i).\tag{2}$$

若考虑相邻的 2 次轮函数迭代，则有：

$$\begin{cases}X_{i+4}=X_i\oplus T(X_{i+1}\oplus X_{i+2}\oplus X_{i+3}\oplus rk_i),\\ X_{i+5}=X_i\oplus T(X_{i+2}\oplus X_{i+3}\oplus X_{i+4}\oplus rk_{i+1}).\end{cases}\tag{3}$$

观察式(1)~(3)不难发现，由于 SM4 采用了 4 条数据线路的 Feistel 结构进行设计，在相邻的 2 次轮函数迭代过程中，均有 96 b 的输入是完全一致的，在式(3)的计算过程中，相邻 2 轮的轮函数将$X_{i+2}\oplus X_{i+3}$计算了 2 次.

因此，一个简单的优化思路便是，我们在轮函数之间传递数据时，额外传递$X_{i+2}\oplus X_{i+3}\oplus rk_{i+1}$的运算结果，并作用于下一次计算，得到的流程图如图 5 所示.

相比于图 4 的运算流程，在计算当前轮次的输出时，二次优化过后的轮函数通过提前获取下一轮次使用的密钥，并利用 2 轮之间相同的数据提前计算，可以使得在加解密的流程中总计节省 32 次异或运算的时间.

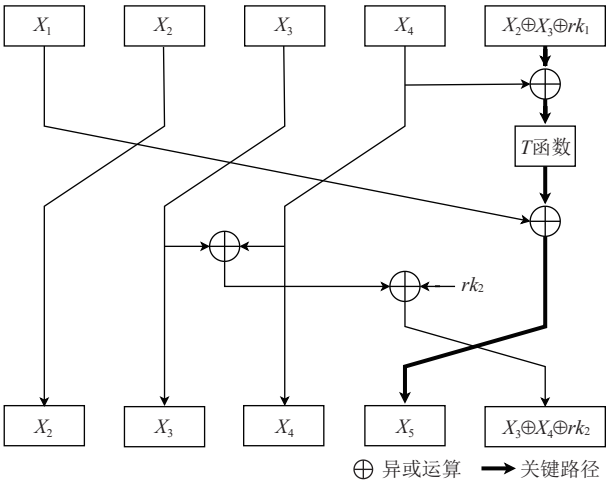


Fig. 5 Optimized round function structure

图 5 优化的轮函数结构

## 2.2 S 盒性能优化

S 盒是密码学领域的一个基本组件，其功能是实现数据的非线性变换，在 DES, AES, SM1, SM4 等算法中均有应用. 在 SM4 算法中，其提供了一个 8 b 到 8 b 的非线性变换.

在 SM4 算法中，S 盒模块通常与另一个线性变换函数 $L'$ 组合使用，即图 4 和图 5 中的 $T$ 函数，其位于加解密算法轮函数的关键路径上，因此，如果能找到优化 $T$ 函数关键路径的方法延时，也可以使得整个加解密模块的延时变小，进而提高运算效率. $T$ 函数的内部结构如图 6 所示，图中的<<<表示对 32 b 数据进行循环左移，关键路径包括 1 个 S 盒和 3 次异或运算. 在硬件实现中，循环移位可以通过硬件连线来实现，不会带来额外的路径延时.

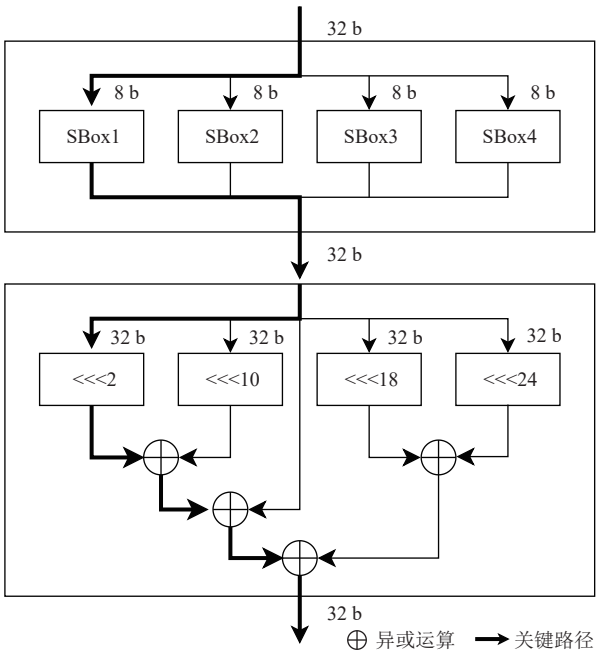


Fig. 6 $T$ function structure of SM4 encryption and decryption modules

图 6 SM4 加解密模块 $T$ 函数结构

$T$ 函数中包含 4 次异或运算，反映到电路设计中，其关键路径上至少存在 3 次异或运算. 因此，一个优化思路便是，将算法中的 S 盒的输入输出修改为 8 b 输入、32 b 输出[2-3]，并提前将 $L'$ 函数作用于图中的 4 个 S 盒，如图 7 所示. 图 7 中，通过编码的形式保存其运行结果，将图 6 中的 SBox 与后续的线性变换 $L'$ 组合形成 exSBox，之后仅需要将 4 个 exSBox 的输出异或即可，从而减少了 1 次异或运算.

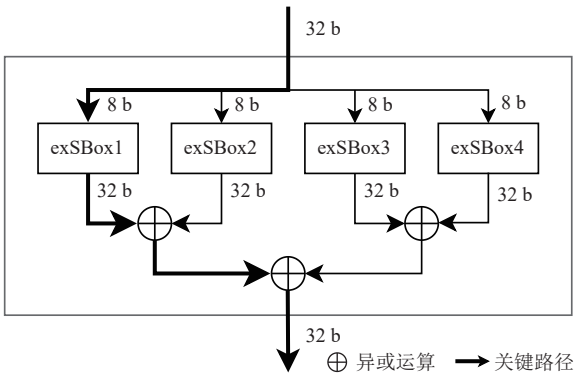


Fig. 7 Optimized $T$-function structure

图 7 优化的 $T$ 函数结构

虽然修改后的 S 盒比原先的 S 盒输出了更多的数据，但在硬件实现中，仍然是通过相同数量的多路选择器查表输出. 因此修改前后的 S 盒的路径延时及其安全性并未改变.

## 2.3 S 盒面积优化

以图 7 中的 exSBox1 为例，使用0xff作为输入展示 exSBox1 的构造方式，首先获得0xff作用于 S 盒后的运行结果0x48. 由于 exSBox1 的输入对应最高四位，因此，将其拓展为 32 b 数据为0x48000000. 在经过 $L'$ 函数后，得到的值是0x68492121. 如表 1 所示，表中前 5 行加粗部分表示传入的数据及其循环移位后所处位置，其余位置在任意输入下都恒等于 0.

**Table 1 Search Space Reduction Rate and Hit Rate**

**表 1 搜索空间降低比率和命中率**

| | | | | |
|---|---|---|---|---|
| 原数据 | **01001000** | 00000000 | 00000000 | 00000000 |
| <<<2 | **00100**000 | 00000000 | 00000000 | 00000**01** |
| <<<10 | 00000000 | 00000000 | 00000**01** | **00100**000 |
| <<<18 | 00000000 | 00000**01** | **00100**000 | 00000000 |
| <<<24 | 00000000 | **01001000** | 00000000 | 00000000 |
| 异或和 | **01101**000 | 01001**01** | 00100001 | 00100001 |

注：加粗部分表示传入的数据及其循环移位后所处位置.

观察表 1 的运算结果不难发现，除最后一行加粗数字表示的第 0~5 位，第 14，15 位由异或运算产生，其余的 24 位均是输入的 8 位数据的排列组合，因此在硬件设计时，可以仅使用 8 b 输入、16 b 输出的 S 盒实现. 对于图 7 中剩余的 3 个 exSBox，在相同的输入下，可以通过对表 1 中的数据进行循环移位，得到对应的输出. 上述结论对 4 个位于不同部位的 S 盒均成立.

具体而言，令 $p$ 为输入的 8 b 数据，$\tau(p)$ 为标准 SM4 算法中 S 盒的输出. $X=(x_0,x_1,\cdots,x_{15})$ 为 exSBox1 中存储的 16 b 数据，$Y=(y_0,y_1,\cdots,y_{31})$ 为优化后的 $T$ 函数中需要的 32 b 输出. $\tau$ 为 SM4 算法标准中使用的 S 盒置换函数，其对于 8 b 输入，产生对应的 8 b 输出，则 $X$ 可以由式(4)产生：

$$\begin{cases}(x_0,x_1,\cdots,x_7)=\tau(p),\\(x_8,x_9,\cdots,x_{15})=\tau(p)\oplus(\tau(p)<<<2).\end{cases}\tag{4}$$

由表 1 可知，$Y$ 的取值实际上可以由 $X$ 经过排列组合得到，对于 exSBox2，exSBox3，exSBox4 的取值，可以通过 $Y$ 循环移位得到，且由于该过程中仅包含赋值运算，在电路设计中可以通过物理连线完成. 相比于文献 [2] 中的设计，节约了 1/3 的面积消耗. 具体的计算方式如式(5)所示.

$$\begin{cases}(y_0,y_1,\cdots,y_5)=(x_8,x_9,\cdots,x_{13}),\\(y_6,y_7)=(x_6,x_7),\\(y_8,y_9,\cdots,y_{13})=(x_0,x_1,\cdots,x_5),\\(y_{14},y_{15})=(x_{14},x_{15}),\\(y_{16},y_{17},\cdots,y_{21})=(x_2,x_3,\cdots,x_7),\\(y_{22},y_{23})=(x_0,x_1),\\(y_{24},y_{25},\cdots,y_{29})=(x_2,x_3,\cdots,x_7),\\(y_{30},y_{31})=(x_0,x_1).\end{cases}\tag{5}$$

## 3 硬件实现与实验对比

现场可编程逻辑门阵列(FPGA)和专用集成电路(ASIC)是目前主流使用硬件电路实现密码算法的 2 个方式. FPGA 虽然具有可编程性、灵活性和快速设计等优势，但 ASIC 相较于 FPGA 拥有更高的性能，与本文设计追求的高效率目标相符，所以选择在 ASIC 下实现.

### 3.1 硬件整体设计

SM4 硬件系统的整体结构设计如图 8 所示，包括密钥扩展模块、加解密模块和适配 CBC 工作模式的组合逻辑. 对于单个加解密任务，若明文被分为 $n$ 组，会执行 1 次密钥扩展和 $n$ 次加解密. 因此，优化

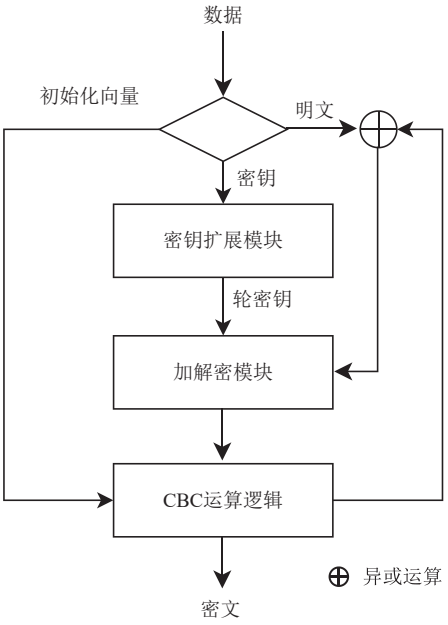


Fig. 8 Overall architecture of SM4 hardware

图 8 SM4 硬件整体架构

加解密算法的执行效率是优化 SM4 硬件设计的重点. 本文所提出的 2 种化简方法，对于每一组明文输入，可以减少 64 级异或门的延时，极大地提升了运算效率.

### 3.2 加解密模块设计

SM4 算法的硬件实现主要有 2 种方案：一种方案是流水线结构，即通过寄存器连接多个加解密模块同时工作以提高加解密的效率，如图 9(a)所示；另一种方案是使用循环迭代的方式. 即一次性提取 32 个轮函数中的 $n$ 轮组合成一个组合电路，称为 $n$ 合 1 电路，如图 9(b)所示. 流水线结构的优势是可以充分利用 $n$ 个加密核心的性能，在不影响整体工作频率的情况下加速运算. 对于 SM4 算法而言，在合理范围内堆叠流水线可以实现极高的吞吐量.

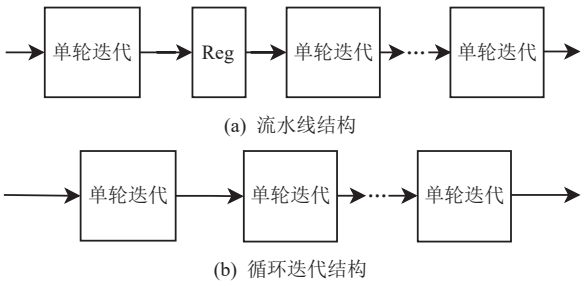


Fig. 9 Pipeline architecture and loop iteration architecture

图 9 流水线结构与循环迭代结构

然而，流水线结构仅适用于 ECB 等数据无前后依赖的工作模式. 在 CBC 工作模式下，由于需要将前

一轮的输出与本轮的输入进行异或运算，相邻的数据存在依赖，故而无法使用流水线加速运算. 因此，在本设计中没有选用流水线结构.

虽然循环迭代结构会降低整体模块的工作频率，对吞吐量的提升较为有限，但可以同时兼容 ECB, CBC 这 2 种工作模式. 本设计最终选择了循环迭代的设计方式.

### 3.3 密钥扩展模块设计

在 SM4 算法中，密钥扩展与加解密算法类似，均包含 32 轮迭代. 密钥扩展模块采用图 2 所示的单轮组合逻辑电路循环 32 次来实现 32 轮迭代.

在密钥扩展模块的输出端，使用寄存器存放每一轮电路的轮密钥，标号为 0~31，如图 10 所示. 标号从 0 开始的好处是：在解密时，使用到的密钥顺序相反的，加密的第 $k$ 轮使用的是第 $k-1$ 号密钥，解密的第 $k$ 轮使用的是第 $32-k$ 号密钥. 在二进制下，二者的标号可以通过取反操作相互转化.

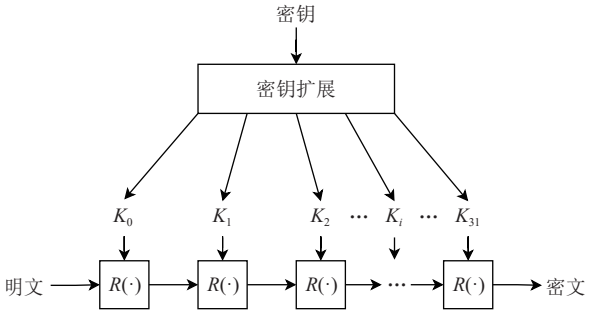


Fig. 10 Storage and usage of round keys

图 10 轮密钥的存储与使用

为了保证运算结果的准确性，密钥扩展模块还会向加解密模块发出控制信号表明自己的工作状态，以避免在轮密钥尚未完全更新时使用错误的轮密钥进行加解密.

### 3.4 综合验证方案

在国家标准文档[1]中，并没有针对 CBC 工作模式给出具体的测试用例. 因此，本文设计方案通过完整的 Verilog HDL 语言实现，通过在 FPGA 平台进行综合、仿真和上板验证，以确保功能正确并进行相关性能分析，如图 11 所示. 具体而言，通过 PCIE 上位机下发随机的明文数据到 FPGA 开发板，开发板完成加密后传回上位机，通过与软件对比实现功能验证. 若在循环验证多次后二者的输出均完全相同，则认为设计的 SM4 电路的功能正确.

最终，本文的设计在 Zynq 7020 FPGA 开发板上完成了上板验证，确保了功能的正确性，工作频率最高可达 95 MHz，吞吐量约为 1.5 Gb/s.

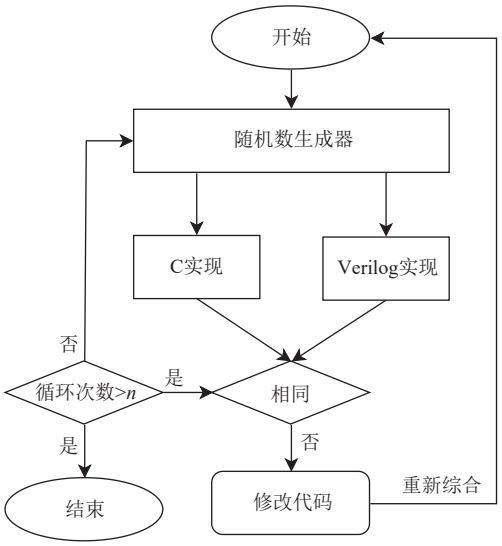


Fig. 11 Testing procedures

图 11 测试流程

### 3.5 ASIC 综合结果

ASIC 上主要针对 2 种工艺 SMIC 55 nm 与 TSMC 40 nm 进行了测试、通过 Synopsys 公司的 EDA 工具 DesignCompiler 进行时序等综合约束，我们选择了芯片面积和芯片使用的逻辑门数量(gates)作为评估指标，其结果如表 2 和表 3 所示，在 CBC 模式下，本文

**Table 2 Comparison of SM4 Synthesis Results and Area Efficiency**

**表 2 SM4 综合结果与面积效率对比**

| 工艺节点 | 芯片面积/mm$^2$ | 吞吐率/(Gb·s$^{-1}$) | 单位面积吞吐率/(Gb·s$^{-1}$·mm$^{-2}$) | 功耗/mW |
|---|---|---|---|---|
| 40 nm* | 0.033 5 | 4.34 | 129.40 | 3.97 |
| 55 nm* | 0.087 7 | 4.41 | 50.30 | 10.88 |
| 65 nm[2] | 0.126 0 | 5.24 | 41.59 | |
| 180 nm[4] | 0.079 0 | 0.10 | 1.27 | 5.31 |
| 55 nm[5] | 0.087 0 | 0.40 | 4.59 | 4.35 |
| 350 nm[6] | 0.027 0 | 0.412 | 15.26 | |

注：*标注的表示本文的结果.

**Table 3 Comparison of SM4 Synthesis Results and Gates Efficiency**

**表 3 SM4 综合结果与门效率对比**

| 工艺节点 | gates | 吞吐率/(Gb·s$^{-1}$) | 单位面积吞吐率/(Gb·s$^{-1}$·gates$^{-1}$) |
|---|---|---|---|
| 40 nm* | $21.2\times10^3$ | 4.34 | $0.205\times10^{-3}$ |
| 55 nm* | $21.1\times10^3$ | 4.41 | $0.209\times10^{-3}$ |
| 180 nm[6] | $32.0\times10^3$ | 0.80 | $0.025\times10^{-3}$ |
| 65 nm[7] | $31.0\times10^3$ | 1.23 | $0.040\times10^{-3}$ |
| 55 nm[8] | $22.0\times10^3$ | 0.27 | $0.012\times10^{-3}$ |
| 130 nm[9] | $22.0\times10^3$ | 0.80 | $0.036\times10^{-3}$ |

注：*标注的表示本文的结果.

的设计在 3.97 mW 的功耗下, 单位面积吞吐率达 129.4 Gb·s$^{-1}$·mm$^{-2}$, 明显优于同类设计. 此外, 以使用逻辑门的数量为评估标准, 本文提出的设计在该指标上也明显优于同类设计, 单位面积吞吐率为 0.205×10$^{-3}$ Gb·s$^{-1}$·gates$^{-1}$.

在不同工艺、电压下对该设计进行综合, 可以得到本文设计在不同使用场景下的吞吐率. 在 TSMC 40 nm、SMIC 55 nm、SMIC 130 nm 下使用不同的工艺角分别对本文的设计进行综合, 结果如表 4 所示.

**Table 4 Comparison of SM4 Synthesis Results and Efficiency with Different Process Corners**

**表 4 不同工艺角下的 SM4 综合结果与效率对比**

| 工艺节点 | 工艺角 | 面积/gates | 吞吐率/(Gb·s$^{-1}$) | 功耗/mW |
|---|---|---|---|---|
| 40 nm | 0.99V/125°C/SS | 21.0×10$^3$ | 2.40 | 2.55 |
| | 1.1V/25°C/TT | 21.2×10$^3$ | 4.34 | 3.97 |
| | 1.21V/0°C/FF | 20.9×10$^3$ | 6.96 | 8.35 |
| 55 nm | 1V/25°C/TT | 20.0×10$^3$ | 2.78 | 4.10 |
| | 1.2V/25°C/TT | 21.1×10$^3$ | 4.41 | 10.88 |
| | 1.32V/0°C/FF | 17.8×10$^3$ | 6.84 | 33.59 |
| 130 nm | 1.08V/125°C/SS | 20.8×10$^3$ | 1.11 | 6.86 |
| | 1.2V/25°C/TT | 21.0×10$^3$ | 1.75 | 15.70 |
| | 1.32V/0°C/FF | 21.8×10$^3$ | 2.45 | 23.03 |

## 4 结 论

根据本文提出的 2 种对 SM4 加解密模块关键路径进行化简以及降低面积的方法, 实现了 4 合 1 的 SM4 电路, 并基于 Zynq7020 开发板进行了功能验证. 此外, ASIC 综合结果表明本文的 SM4 电路相比于其他方案有更高的单位面积吞吐率和更低的功耗. 因此, 这种对 SM4 算法进行的优化是有效的, 并且对其他分组算法提高 CBC 模式下的单位面积吞吐率具有参考价值.

**作者贡献声明:** 郝泽钰提出研究方案并完成了论文的撰写; 代天傲、黄亦成、段岑林协助完成了 ASIC 平台上的验证实验; 董进、吴世勇、张博、王雪岩、贾小涛提出指导意见并修改论文; 杨建磊提出指导意见并讨论定稿.

## 参 考 文 献

[1] GNational Information Security Standardization Technical Committee. Information Security Technology—SM4 Block Cipher Algorithm, GB/T 32907—2016[S]. Beijing: Standards Press of China, 2016 (in Chinese)
(全国信息安全标准化技术委员会. 信息安全技术 SM4 分组密码算法: GB/T 32907—2016[S]. 北京: 中国标准出版社, 2016)

[2] Fu Tianshu, Li Shuguo. A high-throughput ASIC implementation of SM4 algorithm in CBC mode[J]. Microelectronics & Computer, 2016, 33((10): ): 13−18 (in Chinese)
(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)

[3] Shastry P V S, Somani N, Gadre A, et al. Rolled architecture based implementation of AES using T-Box[C]//Proc of 2012 IEEE 55th Int Midwest Symp on Circuits and Systems (MWSCAS). Piscataway, NJ: IEEE, 2012: 626−630

[4] Wang Chenguang, Qiao Shushan, Hei Yong. Design of low complexity SM4 block cipher IP core[J]. Science Technology and Engineering, 2013(2): 347−350 (in Chinese)
(王晨光, 乔树山, 黑勇. 低复杂度 SM4 加密算法 IP 核设计[J]. 科学技术与工程, 2013(2): 347−350)

[5] Zhang Hanyu. Reconfigurable design and hardware implementation of block cipher algorithm [D]. Guangdong: Guangdong University of Technology, 2021 (in Chinese)
(章涵宇. 分组密码算法的可重构设计与硬件实现[D]. 广东: 广东工业大学, 2021)

[6] Bai X, Guo L, Huang L, et al. A fast VLSI design of SMS4 cipher based on twisted BDD S-box architecture[C]//Proc of 2009 Int Conf on Networks Security, Wireless Communications and Trusted Computing. Piscataway, NJ: IEEE, 2009, 1: 345−348

[7] Fan Lingyan, Zhou Meng, Luo Jianjun, et al. IC design with multiple engines running CBC mode SM4 algorithm[J]. Journal of Computer Research and Development, 2018, 55(6): 1247−1253 (in Chinese)
(樊凌雁, 周盟, 骆建军, 等. 多引擎并行 CBC 模式的 SM4 算法的芯片级实现[J]. 计算机研究与发展, 2018, 55(6): 1247−1253)

[8] Rao Bo. Design and implementation of anti-attack based on SM4 cryptographic algorithm [D]. Guangdong: Guangdong University of Technology, 2021 (in Chinese)
(饶博. 基于 SM4 密码算法抗攻击的设计与实现[D]. 广东: 广东工业大学, 2021)

[9] Yan W, You K, Han J, et al. Low-cost reconfigurable VLSI implementation of the SMS4 and AES algorithms[C]//Proc of 2009 IEEE 8th Int Conf on ASIC. Piscataway, NJ: IEEE, 2009: 135−138

**Hao Zeyu**, born in 2001. Master candidate. His main research interest includes privacy computing processor.

**郝泽钰**, 2001 年生. 硕士研究生. 主要研究方向为隐私计算处理器.

**Dai Tianao**, born in 1999. Master candidate. His main research interests include cryptography hardware acceleration and end-to-end AI hardware acceleration.

**代天傲**, 1999 年生. 硕士研究生. 主要研究方向为密码算法硬件加速、端到端人工智能硬件加速.

**Huang Yicheng**, born in 2001. Master candidate. His main research interests include integrated circuit design and information security.

**黄亦成**, 2001 年生. 硕士研究生. 主要研究方向为集成电路设计、信息安全.

**Duan Cenlin**, born in 1992. PhD candidate. Her main research interests include processing-in-memory architectures and neural network accelerator.

**段岑林**, 1992 年生. 博士研究生. 主要研究方向为存内计算架构、神经网络加速器.

**Dong Jin**, born in 1971. PhD, professor. His main research interests include integrated circuit design, blockchain, privacy computing, and AI.

**董　进**, 1971 年生. 博士, 研究员. 主要研究方向为集成芯片设计、区块链、隐私计算、人工智能.

**Wu Shiyong**, born in 1988. Engineer. His main research interests include cryptographic algorithms, blockchain and privacy computing hardware acceleration, hardware security, and heterogeneous computing.

**吴世勇**, 1988 年生. 工程师. 主要研究方向是密码算法、区块链和隐私计算硬件加速、硬件安全、异构计算.

**Zhang Bo**, born in 1987. PhD, associate professor. His main research interests include blockchain and privacy computing domain specific integrated circuit, micro-sensor integrated circuit, and advanced computing system.

**张　博**, 1987 年生. 博士, 副研究员. 主要研究方向为区块链与隐私计算领域专用集成芯片、微型传感器集成芯片、先进计算系统.

**Wang Xueyan**, born in 1992. PhD, assistant professor. Her main research interests include processing-in-memory architectures, AI chip, and hardware security.

**王雪岩**, 1992 年生. 博士, 助理教授. 主要研究方向为内存处理架构、人工智能芯片、硬件安全.

**Jia Xiaotao**, born in 1990. PhD, associate professor. His main research interests include large-scale integrated circuit design and processing-in-memory architectures.

**贾小涛**, 1990 年生. 博士, 副教授. 主要研究方向为大规模集成电路设计、存内计算架构.

**Yang Jianlei**, born in 1987. PhD, associate professor. His main research interests include integrated circuit design, computer architecture, and deep learning chips and systems.

**杨建磊**, 1987 年生. 博士, 副教授. 主要研究方向为集成电路设计、计算机体系结构、智能芯片与系统.