面向物联网的SM4轻量级优化实现

蒲金伟[1],滕亚辉[1],高倾健[1],郑 欣[2],徐迎晖[2]
(1. 广东工业大学自动化学院,广东广州 510006;2. 广东工业大学集成电路学院,广东广州 510006)

摘 要: 针对物联网芯片中对加密算法低面积、高吞吐率需求的增加,提出了速度优先、面积优先、面积速度权衡的3种SM4轻量级硬件实现方案. 面积优先方案中,对线性函数L/L'进行优化实现,减少48位寄存器以及120比特的异或资源的使用;速度优先方案中,引入2个新的S盒,实现线性函数L/L'与查找表S盒的合并,从而避免线性函数L/L'的时延;面积速度权衡方案中,合并S盒线性映射、逆线性映射以及线性函数L/L'为一个函数,将加密计算均映射到复合域中进行,减少一个S盒线性映射时延,进一步提高速度. 与目前已有方案进行比较,面积优先方案面积减少约5.5%~44.8%以上,仅2 371 GE,功耗仅为0.88 mW,最大频率为324 MHz;速度优先方案面积为3 061 GE,最大频率提高约9.8%以上,可达549 MHz,吞吐率为439.2 Mbps.

关键词: 物联网;SM4;轻量级;S盒;优化设计
基金项目: 广东省基础与应用基础研究基金(No. 2021A1515110777);广东省重点领域研发计划(No.2022B0701180001)
中图分类号: TN918.4;TN918.1 **文献标识码:** A **文章编号:** 0372-2112(2024)06-1888-08
电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20230314

Internet of Things Oriented SM4 Lightweight Optimization Implementation

PU Jin-wei[1], TENG Ya-hui[1], GAO Qing-jian[1], ZHENG Xin[2], XU Ying-hui[2]
(1. *School of Automation, Guangdong University of Technology, Guangzhou, Guangdong* 510006, *China*;
2. *School of Integrated Circuits, Guangdong University of Technology, Guangzhou, Guangdong* 510006, *China*)

Abstract: Aiming at the increasing demand for low-area and high-throughput encryption algorithms in IoT chips, three SM4 lightweight optimization hardware implementation schemes are proposed, which are speed priority, area priority, and area-speed trade-off. In the area priority scheme, the linear function L/L' is optimized to reduce the use of 48-bit registers and 120-bit XOR resources. In the speed priority scheme, two new S-boxes are introduced to realize the combination of the linear function L/L' and the look-up table S-box, so as to avoid the delay of the linear function L/L'. In the area-speed trade-off scheme, the S-box linear mapping, the inverse linear mapping and the linear function L/L' are merged into a function, and the encryption calculation is mapped to the composite filed, the delay of the S-box linear mapping can be reduced and the speed can be further improved. Compared with the existing schemes, the area of the area priority scheme is reduced by 5.5%~44.8% (only 2 371 GE), the power consumption is only 0.88 mW, and the maximum frequency is 324 MHz; the area of the speed priority scheme is 3 061 GE, and the maximum frequency is increased by more than 9.8%, up to 549 MHz, with a throughput rate of 439.2 Mbps.

Key words: Internet of things; SM4; lightweight; S-box; optimization design
Foundation Item(s): Guangdong Basic and Applied Basic Research Foundation (No.2021A1515110777); Key-Area Research and Development Program of Guangdong Province (No.2022B0701180001)

1 引言

随着物联网(Internet of Things,IoT)的不断发展和信息交互量的增加,越来越多的人关注数据安全问题. 密码算法广泛应用在智能门锁、密码卡、智能卡等资源受限设备中,为确保新兴物联网平台的信息安全,以SM4算法[1]为代表的加密算法轻量级优化实现具有更

收稿日期:2023-04-07;修回日期:2023-11-05;责任编辑:李勇锋

为重要的实际意义,而这些平台需要在严格的面积和功耗预算内提供更高的吞吐率.

SM4密码算法是国内官方公布的第1个商用密码算法,为了优化SM4算法的电路实现,大量文献提出了一系列改进方法[2-5]. 然而,对于功耗、面积受到严格限制的物联网芯片来说,SM4仍存在减小面积和提高吞吐率两个挑战,不适合物联网应用. 在减小面积方面,2016年,文献[6]首次提出8位串行SM4电路,通过将数据路径由128位并行处理改为8位串行处理,大幅降低了功耗和面积,但同时也降低了吞吐率. 文献[7,8]在文献[6]的基础上,通过密钥扩展和轮函数共享1个S盒,并动态生成轮密钥以及8位轮常数,进一步减少了面积. 文献[9]引入循环移位4比特S盒替换原有8位查找表S盒,进一步减少了面积. 此外,S盒的实现方式对SM4轻量化实现也有一定的影响,文献[6~8]中S盒均采用了面积较大的查找表方式实现,文献[10]提出了基于多项式基的计算式S盒,相对于查找表,面积减少了约39%;文献[11]提出了基于正则基的计算式S盒,相对于查找表,面积减少了约27%. 在增大吞吐率方面,文献[7,8]均以面积最小为目标,采用单S盒进行实现,导致吞吐率较低;国内外有大量文献对与SM4相似的分组密码算法AES[12]进行轻量级优化实现,文献[13,14]设计并实现了单S盒和双S盒两种8位AES优化电路,并对两种结构在面积、吞吐率等方面进行了比较,相对于单S盒,双S盒AES的吞吐率增加约85.5%,面积仅增加10.3%左右,因此双S盒AES、SM4更适用于高吞吐率需求的轻量级物联网芯片. 此外,2011年,文献[15]首次提出将轮密钥加、列混合、行移位、字节替换操作映射到复合域中完成的Native-AES结构,能降低S盒27%的延时,文献[16]将Native-AES结构应用到AES 8位轻量级实现中,提出了一种超低成本、超低功耗的AES设计,但由于SM4结构的限制,无法直接采用相同的方法对SM4时延进行优化.

本文参考AES轻量级优化策略,对SM4在速度、面积、功耗上进行优化实现,主要工作如下:

(1) 采用线性函数L/L'优化实现以及双S盒设计方案,并优化存储系统参数 ,提出一种面积优先的SM4轻量级结构,具有更小的面积资源以及时延.

(2) 在(1)的基础上,引入2个新的查找表S盒,实现线性函数L与查找表S盒的合并,提出一种速度优先SM4轻量级结构,兼顾面积的情况下,具有更高的吞吐率.

(3) 在(2)的基础上,合并计算式S盒逆线性映射、线性映射以及线性函数L/L'为一个函数,进一步减少1个线性映射延时,提出一种速度面积权衡的SM4轻量级结构.

2　预备知识

2.1　SM4加密算法

分组密码算法SM4的分组长度为128比特,由32次非线性迭代运算和1次反序变换组成. 设明文输入为 $(\boldsymbol{X}_0,\boldsymbol{X}_1,\boldsymbol{X}_2,\boldsymbol{X}_3)\in(Z_2^{32})^4$, 密文输出为 $(\boldsymbol{Y}_0,\boldsymbol{Y}_1,\boldsymbol{Y}_2,\boldsymbol{Y}_3)\in(Z_2^{32})^4$, 算法所需的轮密钥为 $\mathbf{rk}_i\in Z_2^{32}, i=0,1,2,\cdots,31$. 则SM4的迭代运算和反序变换如式(1)(2)所示.

$$\begin{aligned}\boldsymbol{X}_{i+4}&=F(\boldsymbol{X}_i,\boldsymbol{X}_{i+1},\boldsymbol{X}_{i+2},\boldsymbol{X}_{i+3},\mathbf{rk}_i)\\&=\boldsymbol{X}_i\oplus T(\boldsymbol{X}_{i+1}\oplus\boldsymbol{X}_{i+2}\oplus\boldsymbol{X}_{i+3}\oplus\mathbf{rk}_i)\end{aligned}\tag{1}$$

$$\begin{aligned}(\boldsymbol{Y}_0,\boldsymbol{Y}_1,\boldsymbol{Y}_2,\boldsymbol{Y}_3)&=R(\boldsymbol{X}_{32},\boldsymbol{X}_{33},\boldsymbol{X}_{34},\boldsymbol{X}_{35})\\&=(\boldsymbol{X}_{35},\boldsymbol{X}_{34},\boldsymbol{X}_{33},\boldsymbol{X}_{32})\end{aligned}\tag{2}$$

其中,$i=0,1,\cdots,31$,F函数为轮函数,T函数为合成置换,由非线性变换τ和线性变换L复合而成,即$T(\cdot)=L(\tau(\cdot))$.

非线性变换τ如式(3)所示,由4个并行的8比特S盒构成,其中$\boldsymbol{A}=(\boldsymbol{A}_0,\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{A}_3)\in(Z_2^8)^4$为输入,$\boldsymbol{B}=(\boldsymbol{B}_0,\boldsymbol{B}_1,\boldsymbol{B}_2,\boldsymbol{B}_3)\in(Z_2^8)^4$为输出.

$$\begin{aligned}(\boldsymbol{B}_0,\boldsymbol{B}_1,\boldsymbol{B}_2,\boldsymbol{B}_3)&=\tau(\boldsymbol{A})\\&=(\mathrm{Sbox}(\boldsymbol{A}_0),\mathrm{Sbox}(\boldsymbol{A}_1),\\&\quad\mathrm{Sbox}(\boldsymbol{A}_2),\mathrm{Sbox}(\boldsymbol{A}_3))\end{aligned}\tag{3}$$

线性变换L如式(4)所示,输入为非线性变换τ输出$\boldsymbol{B}$.

$$\begin{aligned}L(\boldsymbol{B})=\boldsymbol{B}&\oplus(\boldsymbol{B}\lll 2)\oplus(\boldsymbol{B}\lll 10)\oplus(\boldsymbol{B}\lll 18)\\&\oplus(\boldsymbol{B}\lll 24)\end{aligned}\tag{4}$$

2.2　SM4密钥扩展算法

密钥扩展算法由32次迭代运算组成. 设轮密钥为$\mathbf{rk}_i\in Z_2^{32}, i=0,1,\cdots,31$,$\mathbf{rk}_i$生成方法如式(5)所示.

$$\mathbf{rk}_i=\boldsymbol{K}_{i+4}=\boldsymbol{K}_i\oplus T'(\boldsymbol{K}_{i+1}\oplus\boldsymbol{K}_{i+2}\oplus\boldsymbol{K}_{i+3}\oplus\mathbf{CK}_i)\tag{5}$$

其中,$\mathbf{CK}_i$为固定参数;$T'(\cdot)$函数与加密算法中的$T(\cdot)$函数基本相同,线性函数如式(6)所示;$(\boldsymbol{K}_0,\boldsymbol{K}_1,\boldsymbol{K}_2,\boldsymbol{K}_3)$为初始轮密钥,计算过程如式(7)所示.

$$L'(\boldsymbol{x})=\boldsymbol{x}\oplus(\boldsymbol{x}\ll 13)\oplus(\boldsymbol{x}\ll 23)\tag{6}$$

$$\begin{aligned}(\boldsymbol{K}_0,\boldsymbol{K}_1,\boldsymbol{K}_2,\boldsymbol{K}_3)=(&\mathbf{MK}_0\oplus\mathbf{FK}_0,\mathbf{MK}_1\oplus\mathbf{FK}_1,\mathbf{MK}_2\\&\oplus\mathbf{FK}_2,\mathbf{MK}_3\oplus\mathbf{FK}_3)\end{aligned}\tag{7}$$

其中,$\mathbf{MK}=(\mathbf{MK}_0,\mathbf{MK}_1,\mathbf{MK}_2,\mathbf{MK}_3)\in(Z_2^{32})^4$为输入密钥,$\mathbf{FK}$为系统参数.

2.3　SM4 S盒替换

S盒是基于$\mathrm{GF}(2^8)$上求逆来定义的,是SM4算法硬件实现电路中面积最大的模块,其构造有查找表和计算式计算两种方式. 查找表的优势是速度快,但存在面积大、功耗大的缺点. 采用计算式实现的S盒具有面积小、功耗小的优点,但速度比查找表稍慢. 图1为文献[11]提出的基于正则基的SM4 S盒复合域实现结构.

上述模块中,LM由同构映射矩阵和仿射矩阵组

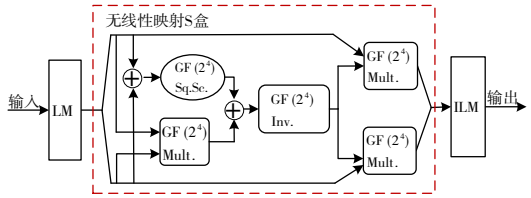


图1 复合域S盒结构

成，ILM由逆同构映射矩阵和仿射矩阵组成，如式(8)、(9)所示.

$$\mathrm{LM}(\boldsymbol{x})=\boldsymbol{M}(\boldsymbol{A}\boldsymbol{x}+\boldsymbol{c})=\boldsymbol{P}\boldsymbol{x}+\boldsymbol{m} \tag{8}$$

$$\mathrm{ILM}(\boldsymbol{x})=\boldsymbol{A}(\boldsymbol{M}^{-1}\boldsymbol{x})+\boldsymbol{c}=\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{n} \tag{9}$$

其中，$\boldsymbol{A}$为8×8仿射矩阵，$\boldsymbol{M}$为8×8同构映射矩阵，$\boldsymbol{M}^{-1}$为8×8逆同构映射矩阵，$\boldsymbol{c}$为行向量.

3 SM4优化设计

3.1 线性函数L/L'优化设计

SM4算法中采用的线性函数L/L'如式(4)、式(6)所示，在文献[6~8]中提出的轻量级电路结构中，均引入3个额外的8比特寄存器，对S盒输出的前3字节进行存储，并在第4个S盒输出字节产生后，一并输入到桶形移位器中进行线性函数L/L'运算.

为了避免引入额外的寄存器，减少面积资源，文献[17]提出一种面积更小、速度更快的SM4线性函数L/L'优化实现方案，如式(10)~式(13)所示. 在优化后的线性函数L/L'中，需要对每个输入字节进行$F(\boldsymbol{x})/G(\boldsymbol{x})$运算，通过异或运算和位拼接操作，将S盒的8比特输出扩展至32比特，再进行循环左移.

$$\begin{aligned}F(\boldsymbol{x})&=\{f_1(\boldsymbol{x}),f_2(\boldsymbol{x}),f_3(\boldsymbol{x}),f_3(\boldsymbol{x})\}\\&=\{\boldsymbol{x}[5:0]\oplus\boldsymbol{x}[7:2],\ \boldsymbol{x}[1:0],\ \boldsymbol{x}[7:2],\ \boldsymbol{x}[1:0]\\&\quad\oplus\boldsymbol{x}[7:6],\ \boldsymbol{x}[5:0],\ \boldsymbol{x}[7:6],\ \boldsymbol{x}[5:0],\ \boldsymbol{x}[7:6]\}\end{aligned} \tag{10}$$

$$\begin{aligned}L_1(\boldsymbol{x})&=F(\boldsymbol{a})\oplus(F(\boldsymbol{d})<<<8)\oplus(F(\boldsymbol{c})<<<16)\\&\quad\oplus(F(\boldsymbol{b})<<<24)\end{aligned} \tag{11}$$

$$\begin{aligned}G(\boldsymbol{x})&=\{g_1(\boldsymbol{x}),g_2(\boldsymbol{x}),g_3(\boldsymbol{x}),g_4(\boldsymbol{x})\}\\&=\{\boldsymbol{x}[7:0],\ 1'b0,\ \boldsymbol{x}[7:1],\ \boldsymbol{x}[0],\ 2'b0,\ \boldsymbol{x}[7:3],\\&\quad\boldsymbol{x}[2:0],\ 5'b0\}\end{aligned} \tag{12}$$

$$\begin{aligned}L_1'(\boldsymbol{x})&=G(\boldsymbol{a})\oplus(G(\boldsymbol{d})<<<8)\oplus(G(\boldsymbol{c})<<<16)\\&\quad\oplus(G(\boldsymbol{b})<<<24)\end{aligned} \tag{13}$$

其中，$\boldsymbol{a},\boldsymbol{b},\boldsymbol{c},\boldsymbol{d}$为$\boldsymbol{x}$的4个字节.

采用上述方法，每个S盒输出字节均能独立运算，无需提前存储S盒输出前3个字节，避免了额外的寄存器使用. 此外，相较于完成式(4)、式(6)运算共需要192比特异或运算，完成式(10)、式(13)仅需72比特的异或资源，可进一步减少面积资源并降低时延.

3.2 线性函数L/L'与S盒合并优化设计

观察式(10)和式(12)可知，$F(\boldsymbol{x})$函数中存在异或操作，而$G(\boldsymbol{x})$函数中没有异或操作，因此在加密过程中，轮函数为关键路径. $F(\boldsymbol{x})$由$\{\boldsymbol{x}[5:0],\boldsymbol{x}[7:6]\},\{\boldsymbol{x}[5:0]\oplus\boldsymbol{x}[7:2],\boldsymbol{x}[1:0]\},\{\boldsymbol{x}[7:2],\boldsymbol{x}[1:0]\oplus\boldsymbol{x}[7:6]\}$以及$\{\boldsymbol{x}[5:0],\boldsymbol{x}[7:6]\}$4个字节组成. 因此可以额外实现2个输入为$\boldsymbol{x}$，输出分别为如式(14)、(15)所示的查找表S盒$S_1(\boldsymbol{x})$，$S_2(\boldsymbol{x})$，其中$S(\boldsymbol{x})$为SM4原始S盒，从而避免异或运算. 虽然采用该方法会额外引入2个新的S盒，增大面积消耗，但可以消除一个异或操作的时延.

$$S_1(\boldsymbol{x})=\{S(\boldsymbol{x})[5:0]\oplus S(\boldsymbol{x})[7:2],\ S(\boldsymbol{x})[1:0]\} \tag{14}$$

$$S_2(\boldsymbol{x})=\{S(\boldsymbol{x})[7:2],\ S(\boldsymbol{x})[1:0]\oplus S(\boldsymbol{x})[7:6]\} \tag{15}$$

通过引入新的S盒，可以将S盒替换和$F(\boldsymbol{x})$进行合并得到优化后的线性函数L_2如式(16)、(17)所示. 由于$G(\boldsymbol{x})$中没有异或运算，可以避免新的查找表S盒实现.

$$\begin{aligned}F_1(\boldsymbol{x})=\{&S_1(\boldsymbol{x}),\ S_2(\boldsymbol{x}),\ S(\boldsymbol{x})[5:0],\ S(\boldsymbol{x})[7:6],\\&S(\boldsymbol{x})[5:0],\ S(\boldsymbol{x})[7:6]\}\end{aligned} \tag{16}$$

$$\begin{aligned}L_2(\boldsymbol{x})&=F_1(\boldsymbol{a})\oplus(F_1(\boldsymbol{d})<<<8)\oplus(F_1(\boldsymbol{c})<<<16)\\&\quad\oplus(F_1(\boldsymbol{b})<<<24)\end{aligned} \tag{17}$$

采用上述方法对轮函数中线性函数L进行优化实现，可以避免$F(\boldsymbol{x})$中异或操作，减少时延，且与密钥扩展具有相同的时延. 相对于3.1小节的优化，引入了4个查找表实现的S盒，因此面积会增加，但速度会更快，更适用于对速度有更高要求的物联网芯片中.

3.3 轮函数S盒线性映射合并优化设计

为了进一步减少时延，文献[15]首次提出Native-AES结构，由于线性函数L/L'的限制，并没有相关文献采用类似方法对SM4进行优化. 通过分析，本文3.1小节中优化后的线性函数L/L'，可将L/L'由单个周期扩展至4个周期完成，可独立计算L/L'4个输入字节中的每个字节. 基于线性函数L/L'的上述特性，结合Native-AES结构，本文提出一种将逆线性映射ILM、$F(\boldsymbol{x})$函数、线性映射LM进行合并的方法，映射所有计算过程到GF($(2^4)^2$)复合域中实现，进而减少一个线性映射的时延.

记矩阵$\boldsymbol{P}$前6列为矩阵$\boldsymbol{P}_1$，最后2列为矩阵$\boldsymbol{P}_2$，矩阵$\boldsymbol{Q}$前2行为矩阵$\boldsymbol{Q}_1$，后6行为矩阵$\boldsymbol{Q}_2$. 由于$F(\boldsymbol{x})$函数由3个子函数$f_1(\boldsymbol{x})$~$f_3(\boldsymbol{x})$组成，因此需要分3步进行合并.

首先对S盒GF(2^4)求逆输出$\boldsymbol{x}$进行逆线性映射ILM($\boldsymbol{x}$)、$f_1(\boldsymbol{x})$函数的合并，计算过程如式(18)所示，其中$f_1(\boldsymbol{x})$、ILM($\boldsymbol{x}$)均为线性函数.

$$\begin{aligned}\boldsymbol{y}_1&=f_1(\mathrm{ILM}(\boldsymbol{x}))\\&=\mathrm{ILM}(f_1(\boldsymbol{x}))\\&=\mathrm{ILM}(\{\boldsymbol{x}[7:2]\oplus\boldsymbol{x}[5:0],\boldsymbol{x}[1:0]\})\\&=\mathrm{ILM}(\boldsymbol{x})+\mathrm{ILM}(\{\boldsymbol{x}[5:0],2'd0\})\\&=\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{n}+\begin{bmatrix}\boldsymbol{Q}_2\\0\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}\boldsymbol{n}_2\\0\end{bmatrix}\end{aligned} \tag{18}$$

接下来进行LM($\boldsymbol{x}$)运算,将上述结果映射到复合域中,完成线性映射LM($\boldsymbol{x}$)的合并,如式(19)所示.

$$\begin{aligned}z_1&=\mathrm{LM}(\boldsymbol{y}_1)=\boldsymbol{P}\boldsymbol{y}_1+\boldsymbol{m}\\&=\boldsymbol{P}\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{P}\boldsymbol{n}+[\boldsymbol{P}_1|\boldsymbol{P}_2]\begin{bmatrix}\boldsymbol{Q}_2\\0\end{bmatrix}\boldsymbol{x}+[\boldsymbol{P}_1|\boldsymbol{P}_2]\begin{bmatrix}\boldsymbol{n}_2\\0\end{bmatrix}+\boldsymbol{m}\\&=(\boldsymbol{P}\boldsymbol{Q}+\boldsymbol{P}_1\boldsymbol{Q}_2)\boldsymbol{x}+(\boldsymbol{P}\boldsymbol{n}+\boldsymbol{P}_1\boldsymbol{n}_2+\boldsymbol{m})\\&=\boldsymbol{T}_1\boldsymbol{x}+\boldsymbol{\mu}_1\end{aligned}\tag{19}$$

同理,对S盒$GF(2^4)$求逆输出$\boldsymbol{x}$进行逆线性映射ILM($\boldsymbol{x}$)、$f_2(\boldsymbol{x})$函数的合并,如式(20)所示,再进行LM($\boldsymbol{x}$)运算将结果映射回复合域,如式(21)所示.

$$\begin{aligned}\boldsymbol{y}_2&=\mathrm{ILM}(\{\boldsymbol{x}[7:2],\boldsymbol{x}[1:0]\oplus\boldsymbol{x}[7:6]\})\\&=\mathrm{ILM}(\boldsymbol{x})+\mathrm{ILM}(\{5'\mathrm{d}0,\boldsymbol{x}[7:6]\})\\&=\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{n}+\begin{bmatrix}0\\\boldsymbol{Q}_1\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}0\\\boldsymbol{n}_1\end{bmatrix}\end{aligned}\tag{20}$$

$$\begin{aligned}z_2&=\mathrm{LM}(\boldsymbol{y}_2)=\boldsymbol{P}\boldsymbol{y}_2+\boldsymbol{m}\\&=(\boldsymbol{P}\boldsymbol{Q}+\boldsymbol{P}_2\boldsymbol{Q}_1)\boldsymbol{x}+(\boldsymbol{P}\boldsymbol{n}+\boldsymbol{P}_2\boldsymbol{n}_1+\boldsymbol{m})\\&=\boldsymbol{T}_2\boldsymbol{x}+\boldsymbol{\mu}_2\end{aligned}\tag{21}$$

同理,对S盒$\mathrm{GF}(2^4)$求逆输出$\boldsymbol{x}$进行逆线性映射ILM($\boldsymbol{x}$)、$f_3(\boldsymbol{x})$函数的合并,如式(22)所示,再进行LM($\boldsymbol{x}$)运算将结果映射回复合域,如式(23)所示.

$$\begin{aligned}\boldsymbol{y}_3&=\mathrm{ILM}(\{\boldsymbol{x}[5:0],\boldsymbol{x}[7:6]\})\\&=\mathrm{ILM}(\{\boldsymbol{x}[5:0],2'\mathrm{d}0\})+\mathrm{ILM}(\{5'\mathrm{d}0,\boldsymbol{x}[7:6]\})\\&=\begin{bmatrix}\boldsymbol{Q}_2\\\boldsymbol{Q}_1\end{bmatrix}\boldsymbol{x}+\begin{bmatrix}\boldsymbol{n}_2\\\boldsymbol{n}_1\end{bmatrix}\end{aligned}\tag{22}$$

$$\begin{aligned}z_3&=\mathrm{LM}(\boldsymbol{y}_3)=\boldsymbol{P}\boldsymbol{y}_3+\boldsymbol{m}\\&=(\boldsymbol{P}_1\boldsymbol{Q}_2+\boldsymbol{P}_2\boldsymbol{Q}_1)\boldsymbol{x}+(\boldsymbol{P}_1\boldsymbol{n}_2+\boldsymbol{P}_2\boldsymbol{n}_1+\boldsymbol{m})\\&=\boldsymbol{T}_3\boldsymbol{x}+\boldsymbol{\mu}_3\end{aligned}\tag{23}$$

综上,通过合并S盒的逆线性映射ILM($\boldsymbol{x}$)、线性函数L_1变换、S盒的线性映射LM($\boldsymbol{x}$)为1个操作,推导出了具有更小时延的线性函数L_2,如式(24)、(25)所示,可进一步改善关键路径时延.

$$F_2(\boldsymbol{x})=\{\boldsymbol{T}_1\boldsymbol{x}+\boldsymbol{\mu}_1,\boldsymbol{T}_2\boldsymbol{x}+\boldsymbol{\mu}_2,\boldsymbol{T}_3\boldsymbol{x}+\boldsymbol{\mu}_3,\boldsymbol{T}_3\boldsymbol{x}+\boldsymbol{\mu}_3\}\tag{24}$$

$$\begin{aligned}L_2(\boldsymbol{x})=&F_2(\boldsymbol{a})\oplus(F_2(\boldsymbol{d})<<<8)\oplus(F_2(\boldsymbol{c})<<<16)\\&\oplus(F_2(\boldsymbol{b})<<<24)\end{aligned}\tag{25}$$

观察上式,线性函数L_2由4个S盒复合域求逆输出字节进行$F_2(\boldsymbol{x})$运算后,再进行循环左移并异或后得到.由此可知,L_2为线性函数,可以对$F_2(\boldsymbol{x})$进行如式(26)所示的分解.

$$F_2(\boldsymbol{x})=\{\boldsymbol{T}_1\boldsymbol{x},\boldsymbol{T}_2\boldsymbol{x},\boldsymbol{T}_3\boldsymbol{x},\boldsymbol{T}_3\boldsymbol{x}\}+\{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\boldsymbol{\mu}_3,\boldsymbol{\mu}_3\}\tag{26}$$

令函数$F_3(\boldsymbol{x})=\{\boldsymbol{T}_1\boldsymbol{x},\boldsymbol{T}_2\boldsymbol{x},\boldsymbol{T}_3\boldsymbol{x},\boldsymbol{T}_3\boldsymbol{x}\}$,那么函数$L_2$可以进一步表示为如式(27)所示.

$$\begin{aligned}L_3(\boldsymbol{x})=&F_3(\boldsymbol{a})\oplus(F_3(\boldsymbol{d})<<<8)\oplus(F_3(\boldsymbol{c})<<<16)\oplus\\&(F_3(\boldsymbol{b})<<<24)+\{\boldsymbol{\mu},\boldsymbol{\mu},\boldsymbol{\mu},\boldsymbol{\mu}\}\end{aligned}\tag{27}$$

其中,$\boldsymbol{\mu}=\boldsymbol{\mu}_1+\boldsymbol{\mu}_2$.

由于在轮函数中,输入4个字节经过L函数后还需要异或上$\boldsymbol{X}_1$,因此可以在进行字节替换的同时,并行对$\boldsymbol{X}_1$中的每个字节加上$\boldsymbol{\mu}$,可以进一步减少1个异或运算的时延.

3.4　密钥扩展S盒线性映射合并优化设计

SM4密钥扩展与轮函数具有相似的结构,因此可采用3.3小节中相同的方法对式(15)所示的L_1'进行进一步优化.记矩阵$\boldsymbol{Q}$的前7行为$\boldsymbol{Q}_3$,第8行为$\boldsymbol{Q}_4$,前5行记为$\boldsymbol{Q}_5$,最后3行记为$\boldsymbol{Q}_6$.记矩阵$\boldsymbol{P}$的前1列为$\boldsymbol{P}_3$,最后7列为$\boldsymbol{P}_4$,前3列记为$\boldsymbol{P}_5$,后5列记为$\boldsymbol{P}_6$.

$G(\boldsymbol{x})$函数由4个子函数$g_1(\boldsymbol{x})$~$g_4(\boldsymbol{x})$组成,与3.3小节一样,分4步对逆线性映射ILM、$G(\boldsymbol{x})$函数、线性映射LM进行合并,如式(28)~(31)所示.

$$z_1=\boldsymbol{P}_4\boldsymbol{Q}_3\boldsymbol{x}+(\boldsymbol{P}_4\boldsymbol{n}_3+\boldsymbol{m})=\boldsymbol{R}_1\boldsymbol{x}+\boldsymbol{v}_1\tag{28}$$

$$z_2=(\boldsymbol{P}_3\boldsymbol{Q}_4+\boldsymbol{P}_6\boldsymbol{Q}_5)\boldsymbol{x}+(\boldsymbol{P}_3\boldsymbol{n}_4+\boldsymbol{P}_6\boldsymbol{n}_5+\boldsymbol{m})=\boldsymbol{R}_2\boldsymbol{x}+\boldsymbol{v}_2\tag{29}$$

$$z_3=\boldsymbol{P}_5\boldsymbol{Q}_6\boldsymbol{x}+(\boldsymbol{P}_5\boldsymbol{n}_6+\boldsymbol{m})=\boldsymbol{R}_3\boldsymbol{x}+\boldsymbol{v}_3\tag{30}$$

$$z_4=\boldsymbol{P}\boldsymbol{Q}\boldsymbol{x}+\boldsymbol{P}\boldsymbol{n}+\boldsymbol{m}=\boldsymbol{R}_4\boldsymbol{x}+\boldsymbol{v}_4\tag{31}$$

通过上述推导,可以推导出时延更小的线性函数$G_1(\boldsymbol{x})$如式(32)所示,大幅改善关键路径时延.线性函数L'可进一步优化为L_2',如式(33)所示.

$$G_1(\boldsymbol{x})=\{\boldsymbol{R}_1\boldsymbol{x}+\boldsymbol{v}_1,\boldsymbol{R}_2\boldsymbol{x}+\boldsymbol{v}_2,\boldsymbol{R}_3\boldsymbol{x}+\boldsymbol{v}_3,\boldsymbol{R}_4\boldsymbol{x}+\boldsymbol{v}_4\}\tag{32}$$

$$\begin{aligned}L_2'(\boldsymbol{x})=&G_1(\boldsymbol{a})\oplus(G_1(\boldsymbol{d})<<<8)\oplus(G_1(\boldsymbol{c})<<<16)\\&\oplus(G_1(\boldsymbol{b})<<<24)\end{aligned}\tag{33}$$

在式(32)(33)的基础上,采用式(26)(27)相同的方法,对$G_1(\boldsymbol{x})$进一步分解,可得到进一步优化后的线性函数L_3',如式(34)、(35)所示.

$$G_2(\boldsymbol{x})=\{\boldsymbol{R}_1\boldsymbol{x},\boldsymbol{R}_2\boldsymbol{x},\boldsymbol{R}_3\boldsymbol{x},\boldsymbol{R}_4\boldsymbol{x}\}\tag{34}$$

$$\begin{aligned}L_3'(\boldsymbol{x})=&G_2(\boldsymbol{a})\oplus(G_2(\boldsymbol{d})<<<8)\oplus(G_2(\boldsymbol{c})<<<16)\\&\oplus(G_2(\boldsymbol{b})<<<24)+\{\boldsymbol{v},\boldsymbol{v},\boldsymbol{v},\boldsymbol{v}\}\end{aligned}\tag{35}$$

其中,$\boldsymbol{v}=\boldsymbol{v}_1+\boldsymbol{v}_2+\boldsymbol{v}_3+\boldsymbol{v}_4$.

由于在密钥扩展函数中,4个字节经过L'函数后还需要加上$\boldsymbol{K}_1$,因此可以再进行字节替换的同时,并对$\boldsymbol{K}_1$对中的每个字节加上$\boldsymbol{v}$,进一步减少延时.

3.5　FK参数存储优化

在SM4算法中,输入初始密钥需要先与系统参数**FK**进行异或,参数**FK**无法通过计算得到,需要采用128位寄存器进行存储,且该参数仅在密钥扩展初始轮才会使用到.在进行逻辑综合时分析得到,参数**FK**存储所使用的面积资源占整体面积资源的约24.3%.本文提出一种适用于SM4系统参数**FK**的存储优化方法,复用128位状态寄存器,在加载128位明文前,对系统参数**FK**进行存储,可减少SM4轻量级设计的面积资源.

4　SM4轻量级整体电路结构

4.1　面积优先SM4轻量级电路结构

本文结合文献[6~8]提出的优化方法以及第3.1小

节中优化后的线性函数L_1/L_1'、3.5小节中系统参数**FK**存储优化方案，设计了面积优先的SM4（Area_SM4）轻量级电路结构，如图2所示．总体结构可以分为状态寄存器、轮函数电路、密钥寄存器、密钥扩展电路4个部分，采用8比特数据路径位宽，以串行方式实现轮操作和密钥调度．状态寄存器由16个8位寄存器$S_{0,0},S_{1,0},\cdots,S_{3,3}$组成，完成加密中间数据的同步与流动；密钥寄存器由16个8位寄存器$K_{0,0},K_{0,1},\cdots,K_{3,3}$组成，完成密钥扩展中间数据的同步与流动．

首先介绍轮函数的时序．在加密开始前，需要16个时钟周期加载128位明文到状态寄存器中，且**FK**已存储在$S_{0,0},S_{1,0},\cdots,S_{3,3}$共128位寄中．在每个时钟周期，$S_{0,0}$将作为系统参数**FK**输出，8位明文**PT**同步到状态寄存器$S_{3,3}$中，并按$S_{3,3}\rightarrow S_{2,3}\rightarrow\cdots\rightarrow S_{0,0}$方向移位进行移动，即图2中红色箭头方向．接下来的第1轮到第32轮，每轮加密需要4个时钟周期．每个时钟周期内，$S_{0,1}\oplus S_{0,2}\oplus S_{0,3}$作为状态寄存器输出与轮密钥$\mathbf{rk}_{i,j}$进行异或后作为计算式S盒的输入，并按$S_{3,3}\rightarrow S_{2,3}\rightarrow\cdots\rightarrow S_{0,0}$方向进行按字节移位．由式(11)可知，优化后的线性函数L_1由每个时钟输入$\boldsymbol{x}$进行$F(\boldsymbol{x})$扩展后再进行循环左移组成．在图2所示结构中，循环左移通过选择器选择不同的状态寄存器进行实现，详细过程如下．第1个时钟周期，S盒输出$\boldsymbol{x}$进行$F(\boldsymbol{x})$扩展至32位，再与$\{S_{0,0},S_{1,0},S_{2,0},S_{3,0}\}$进行异或，并同步至$\{S_{3,3},S_{0,0},S_{1,0},S_{2,0}\}$；第2个时钟周期，$F(\boldsymbol{x})$与$\{S_{0,0},S_{1,0},S_{2,0},S_{3,3}\}$进行异或，并同步至$\{S_{3,3},S_{0,0},S_{1,0},S_{2,3}\}$；第3个时钟周期，$F(\boldsymbol{x})$与$\{S_{0,0},S_{1,0},S_{2,3},S_{3,3}\}$进行异或，并同步至$\{S_{3,3},S_{0,0},S_{1,3},S_{2,3}\}$；第4个时钟周期，$F(\boldsymbol{x})$与$\{S_{0,0},S_{1,3},S_{2,3},S_{3,3}\}$进行异或，并同步至$\{S_{3,3},S_{0,3},S_{1,3},S_{2,3}\}$．第33轮，首先进行反序变换，在16个时钟周期内以$S_{0,3}$作为输出，状态矩阵按红色箭头反方向进行移动，将状态寄存器中全部数据输出，即密文．由上述可知，完成整个Area_SM4需要$16\times2+32\times4=160$个时钟周期．

接下来介绍密钥扩展的时序．在加密开始前，需要完成128位初始密钥的加载，并完成与系统参数**FK**的异或运算，本设计选择在16个时钟周期内，按$K_{3,3}\rightarrow K_{2,3}\rightarrow\cdots\rightarrow K_{0,0}$方向移位同步16个8位初始密钥**KT**与**FK**的异或，并在13~16时钟周期之间并行完成第1轮密钥扩展．由式(13)可知，$G(\boldsymbol{x})$输出需要进行循环左移，采取与轮函数电路相同的方法，使用选择器选择不同的输入以及异或结果写回不同的寄存器实现循环左移操作．在第13~16时钟周期，密钥寄存器输出$K_{0,2}\oplus K_{0,3}$与**FK**⊕**KT**以及固定参数$\mathbf{CK}_{i,j}$进行异或，$\mathbf{CK}_{i,j}$计算如式(36)所示，异或结果作为计算式S盒的输入．在第13时钟周期，S盒输出$\boldsymbol{x}$通过$G(\boldsymbol{x})$扩展至32位，再执行异或操作$G(\boldsymbol{x})\oplus\{K_{0,1},K_{1,1},K_{2,1},K_{3,1}\}$，结果同步至$\{K_{3,0},K_{0,1},K_{1,1},K_{2,1}\}$；第14时钟周期，$G(\boldsymbol{x})$输出结果执行异或操作$G(\boldsymbol{x})\oplus\{K_{0,1},K_{1,1},K_{2,1},K_{3,0}\}$，结果同步至$\{K_{3,0},K_{0,1},K_{1,1},K_{2,0}\}$；第15时钟周期，$G(\boldsymbol{x})$输出结果执行异或操作$G(\boldsymbol{x})\oplus\{K_{0,1},K_{1,1},K_{2,0},K_{3,0}\}$，结果同步至$\{K_{3,0},K_{0,1},K_{1,0},K_{2,0}\}$；第16时钟周期，$G(\boldsymbol{x})$输出结果执行异或操作$G(\boldsymbol{x})\oplus\{K_{0,1},K_{1,0},K_{2,0},K_{3,0}\}$，结果同步至$\{K_{3,0},K_{0,0},K_{1,0},K_{2,0}\}$．在接下来的1到32轮密钥扩展中，每轮需要4个时钟周期．在第1~4时钟周期，$K_{0,0}$作为轮密钥$\mathbf{rk}_{i,j}$输出，并执行$K_{0,0}\oplus K_{0,2}\oplus K_{0,3}$与固定参数$\mathbf{CK}_{i,j}$异或运算，异或结果作为计算式S盒的输入，S盒输出完成与第0轮第13~16时钟周期相同的操作．

$$\mathbf{CK}_{i,j}=(4i+j)\times7(\mathrm{mod}\,256)\tag{36}$$

4.2 速度优先SM4轻量级电路结构

速度优先SM4（Speed_SM4）轻量级串行电路结构与图2基本类似，包括时序．由3.2小节可知，$G(\boldsymbol{x})$函数并

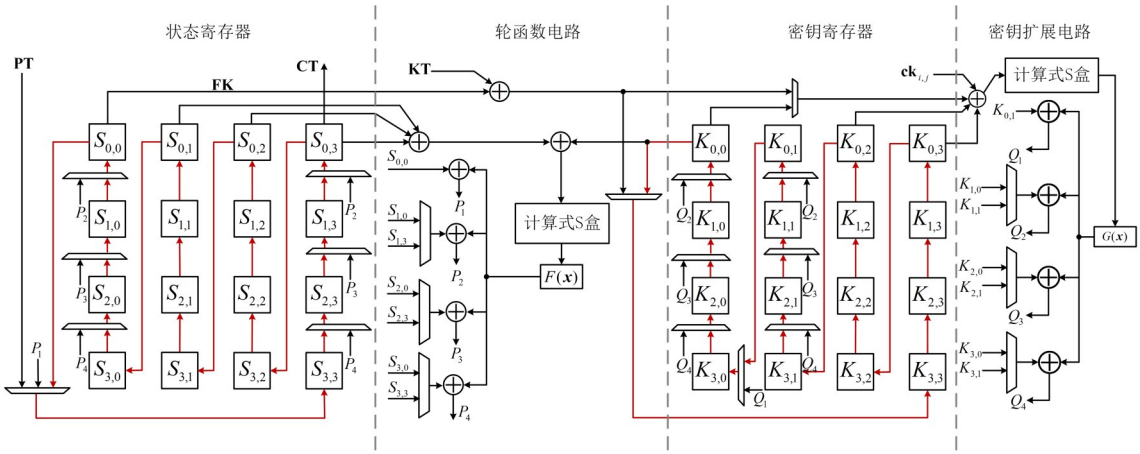


图2 Area_SM4串行电路结构

没有引入新的S盒,因此密钥扩展电路结构与图2结构基本相同,只需将计算式S盒替换为查找表S盒即可. 轮函数电路则采用了3.2小节中优化后的线性函数L_2,需引入2个新的查找表S盒,因此对图2中的轮函数电路按图3所示结构进行替换,即为Speed_SM4串行电路结构.

4.3　面积速度权衡SM4轻量级电路结构

在本小节中,提出一种面积和速度权衡的SM4(Native_SM4)轻量级电路结构,具备比4.1小节更高的速度,比4.2小节更小的面积. Native_SM4电路结构如图4所示,与图2结构基本类似,包括时序,除了对线性函数L/L'优化的电路不同外,还需要替换计算式S盒为无线性映射S盒,对初始明文输入**PT**、初始密钥输入**KT**、密文输出**CT**均进行LM/LM^{-1}线性映射.

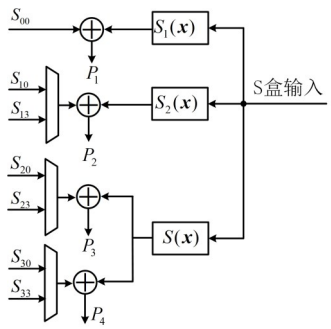


图3　Speed_SM4轮函数电路结构

由式(27)可知,线性函数$F_3(\boldsymbol{x})$引入了3个不同的线性映射矩阵$\boldsymbol{T}_1,\boldsymbol{T}_2,\boldsymbol{T}_3$,记为3个不同的线性映射操作$T_1(\boldsymbol{x}),T_2(\boldsymbol{x}),T_3(\boldsymbol{x})$,对图2轮函数电路中$F(\boldsymbol{x})$进行替换,便可得到Native_SM4轮函数电路结构,如图4所示. 此外,式(27)还需要完成$\{\boldsymbol{\mu},\boldsymbol{\mu},\boldsymbol{\mu},\boldsymbol{\mu}\}$与$\{S_{0,3},S_{1,3},S_{2,3},S_{3,3}\}$之间的异或. 为了减少时延,本文选择在在第2~32轮的每个时钟周期内,$S_{0,3}$先与$\boldsymbol{\mu}$进行异或,再与$S_{0,1}\oplus S_{0,2}$以及轮密钥$\mathbf{rk}_{i,j}$进行异或后作为无线性映射S盒的输入,同时$S_{0,3}\oplus\boldsymbol{\mu}$同步至$S_{3,2}$. 由式(35)可知,改进后的线性函数$L_3'$由4个字节输入经过$G_2(\boldsymbol{x})$后,再进行循环左移组成. 其中,$G_2(\boldsymbol{x})$线性函数引入了4个不同的线性映射矩阵$\boldsymbol{R_1},\boldsymbol{R_2},\boldsymbol{R_3},\boldsymbol{R_4}$,记4个不同的线性映射操作为$R_1(\boldsymbol{x}),R_2(\boldsymbol{x}),R_3(\boldsymbol{x}),R_4(\boldsymbol{x})$,同理对图2中$G(\boldsymbol{x})$进行替换便可得到Native_SM4密钥扩展电路结构. 在第1~32轮每个时钟周期,先进行异或运算$K_{0,0}\oplus\boldsymbol{v}$作为轮密钥输出,再与$K_{0,2}\oplus K_{0,3}$以及固定参数进行异或. 此外,图4中固定参数$\mathbf{CK}_{i,j}$需要替换为$\mathbf{CK}'_{i,j}$,如式(37)所示,保证固定参数也在复合域中.

$$\mathbf{CK}'_{i,j}=\mathrm{LM}((4i+j)\times 7(\mathrm{mod}\,256)) \tag{37}$$

为了减少固定参数进行LM线性变换带来的时延,需要采用时序逻辑,与字节替换并行执行,提前1个时钟计算好下一次需要的固定参数$\mathbf{CK}'_{i,j}$.

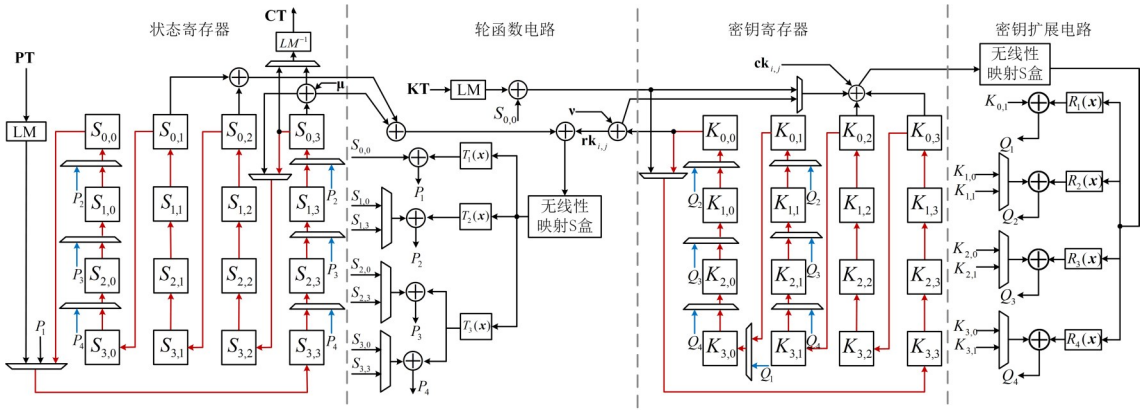


图4　Native_SM4串行电路结构

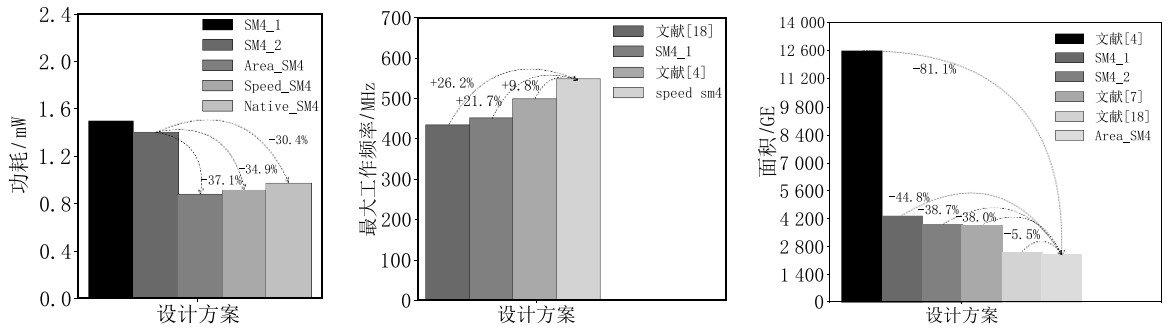
5　性能对比

本文采用Verilog硬件语言对提出的3种SM4轻量级实现方案进行实现,并复现了文献[6]提出的SM4超轻量级电路结构,并在此基础上采用双S盒设计结构,其中S盒分别选择查找表和计算式两种方式实现,分别为SM4_1,SM4_2. 此外,采用Synopsys Design Compiler工具以及SMIC 180 nm工艺库对上述5种SM4轻量级方案进行进行逻辑综合,并于现有文献进行性能对比,性能结果如表1和图5所示. 由图5(a)可知,本文提出的Area_SM4、Speed_SM4、Native_SM4相对于SM4_1、SM4_2方案功耗降低约30.4%~37.1%,仅0.880 mW、0.912 mW、0.975 mW,具有更好的功耗指标. 由图5(b)可知,本文提出的速度优先方案Speed_SM4,最大工作频率达549 MHz,相对同样采用查找表S盒实现的SM4_1、文献[4,6,18],最大工作频率提升9.8%以上. 由图5(c)可知,本文提出的面积优先方案Area_SM4仅2 371 GE,相对于文献[4]提出的128位SM4面积减少约81.1%,相对于同样采用8位数据位宽的SM4_1、SM_

2以及文献[7,18]面积减少约5.5%~44.8%. 此外,本文提出的面积速度权衡方案Native_SM4,相对于Area_SM4方案,最大工作频率提升约8.0%,相对于Speed_SM4方案,面积减少约9.3%.

表1 ASIC性能对比

设计方案	工艺	面积/GE	无密钥扩展面积/GE	时钟数	最大工作频率/MHz	最大吞吐率/Mbps	功耗(20 MHz)/mW
SM4_1	SMIC 180 nm	4 297	2 489	160	451	360.8	1.494
SM4_2	SMIC 180 nm	3 871	2 274	160	284	227.2	1.400
Area_SM4	SMIC 180 nm	2 371	1 351	160	324	259.2	0.880
Speed_SM4	SMIC 180 nm	3 061	1 796	160	549	439.2	0.912
Native_SM4	SMIC 180 nm	2 775	1 470	160	350	280	0.975
文献[6]	SMIC 180 nm	—	3 060	128	185	185	—
文献[7]	SMIC 180 nm	3 824	2 493	256	—	—	—
文献[18]	SMIC 180 nm	2 510	—	256	435	217.5	—
文献[4]	SMIC 180 nm	12 560	—	64	500	1 000	—



(*a*) 功耗对比图 (*b*) 最大工作频率对比图 (*c*) 面积对比图

图5 ASIC性能对比图

6 结语

随着物联网芯片中对加密算法低面积、高吞吐率需求的增加,本文针对分组密码算法SM4轻量级硬件实现,提出了S盒与线性函数L/L'合并,S盒逆线性映射、线性映射与线性函数L/L'合并,系统参数**FK**优化存储方式共3种优化方法,并引入双S盒实现方案对SM4进行轻量级设计. 基于上述提出的优化方法,设计实现了速度优先、面积优先、面积速度权衡3种不同的SM4轻量级电路结构,适用于对面积、速度有不同要求的物联网芯片中.

参考文献

[1] Office of State Commercial Cipher Administration. Block cipher for WLAN products-SMS4[EB/OL]. (2012-03-21)[2023-11-05]. http://www.oscca.gov.cn/UpFile/20060210164231979 90.pdf.

[2] PENG P, MA C Q, GE J Q, et al. A hardware/software collaborative SM4 implementation resistant to side-channel attacks on ARM-FPGA embedded SoC[C]//2020 IEEE Symposium on Computers and Communications (ISCC). Rennes: IEEE, 2020: 1-7.

[3] WANG C, DING Y, HUANG C, et al. An Optimized Isomorphic Design for the SM4 Block Cipher Over the Tower Field[C]//2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom).Wuhan: IEEE, 2023: 422-428.

[4] CHIANG W, CHANG H C, LEE C Y. An area-efficient high-throughput SM4 accelerator with SCA-countermeasure for TV applications[C]//2020 IEEE International Symposium on Circuits and Systems (ISCAS). Seville: IEEE, 2020: 1-5.

[5] CHEN Y, SONG J, CHEN S, et al. Exploring the high-throughput and low-delay hardware design of SM4 on FPGA[C]//2022 19th International SoC Design Conference (ISOCC). Gangneungsi: IEEE, 2023: 211-212.

[6] MING S, ZHANG Q, LIU Z, et al. An ultra-compact hardware implementation of SMS4[C]//2014 IIAI 3rd International Conference on Advanced Applied Informatics. Kokura: IEEE, 2014. 86-90.

[7] 郑朝霞, 资义纯, 吴旭峰, 等. SMS4算法串行化设计及其轻量级电路实现[J]. 华中科技大学学报(自然科学版), 2016, 44(2): 61-64.
ZHENG Z X, ZI Y C, WU X F, et al. Serialized design of SMS4 and lightweight implement[J]. Journal of Huazhong University of Science and Technology (Natural Science Edition), 2016, 44(2): 61-64. (in Chinese)

[8] 朱坤崧, 戴紫彬, 张立朝, 等. 面向物联网的SM4算法轻量级实现[J]. 电子技术应用, 2016, 42(12): 27-30.
ZHU K S, DAI Z B, ZHANG L C, et al. Lightweight implementation of SM4 for Internet of Things[J]. Application of Electronic Technique, 2016, 42(12): 27-30. (in Chinese)

[9] CHEN B W, XIA X, LIANG Q M, et al. Lightweight design of SM4 algorithm and realization of threshold scheme[J]. Journal of Physics: Conference Series, 2021, 1871(1): 012124.

[10] 徐艳华, 白雪飞, 郭立. 适合SMS4算法硬件实现的S盒构造新方法[J]. 中国科学技术大学学报, 2009, 39(11): 1164-1170.
XU Y H, BAI X F, GUO L.A new algorithm of S-box for hardware implementation of SMS4[J]. Journal of University of Science and Technology of China, 2009, 39(11): 1164-1170. (in Chinese)

[11] 梁浩, 乌力吉, 张向民. 基于复合域的SM4算法的设计与实现[J]. 微电子学与计算机, 2015, 32(5): 16-20.
LIANG H, WU L J, ZHANG X M. Design and implementation of SM4 Block Cipher Based on composite field[J]. Microelectronic & Computer, 2015, 32(5):16-20. (in Chinese)

[12] HERON S. Advanced encryption standard (AES)[J]. Network Security, 2009, 2009(12): 8-12.

[13] LU M, FAN A, XU J, et al. A compact, lightweight and low-cost 8-bit datapath AES circuit for IoT applications in 28nm CMOS[C]//2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE). New York: IEEE, 2018: 1464-1469.

[14] 范傲. 面向IoT应用的高能效AES电路设计[D]. 南京: 东南大学, 2018.
FAN A. Energy-efficient AES Circuit Design for IoT Applications[D]. Nanjing: Southeast University, 2018. (in Chinese)

[15] MATHEW S K, SHEIKH F, KOUNAVIS M, et al. 53 Gbps native GF$(2^4)^2$ composite-field AES-encrypt/decrypt accelerator for content-protection in 45 nm high-performance microprocessors[J]. IEEE Journal of Solid-State Circuits, 2011, 46(4): 767-776.

[16] ZHAO W F, HA Y J, ALIOTO M. AES architectures for minimum-energy operation and silicon demonstration in 65nm with lowest energy per encryption[C]//2015 IEEE International Symposium on Circuits and Systems (ISCAS). Lisbon: IEEE, 2015: 2349-2352.

[17] 蒲金伟, 高倾健, 郑欣, 等. SM4抗差分功耗分析轻量级门限实现[J]. 计算机应用, 2023, 43(11): 3490-3496.
PU J W, GAO Q J, ZHENG X, et al. SM4 small threshold implementation against differential power analysis[J]. Journal of Computer Applications, 2023, 43(11): 3490-3496. (in Chinese)

[18] CHEN H F, JIANG Y B. An efficient hardware implementation of SM4[C]//Proceedings of the Fourth International Conference on Industrial Technology and Career Education (ICITCE). Wuhan: Asian Academic Press, 2017: 1-5.

作者简介

蒲金伟　男,1998年出生,重庆人. 2021年在广东工业大学获得工学学士学位. 现为广东工业大学硕士研究生. 主要研究方向为密码算法硬件加速及侧信道防护.
E-mail: jinweipu@126.com

滕亚辉　男,1998年出生,山东临沂人. 现为广东工业大学硕士研究生. 主要研究方向为对称密码算法侧信道防护.
E-mail: 18565099684@163.com