

RISC 微处理器 S 盒替换指令扩展

郭怡惠, 李森森, 戴紫彬, 张立朝

(解放军信息工程大学, 河南 郑州 450000)

摘 要: 为解决 RISC 微处理器实现 S 盒替换操作效率低下这一问题, 提出一种通过设计 S 盒替换专用模块和指令进行指令扩展的方法, 使得微处理器在执行 S 盒替换操作时具有较高性能. 根据分组密码算法 S 盒替换的操作特征, 设计支持 8-32、8-8、6-4、4-4 四种常用模式的 S 盒替换模块和操作指令, 最后采用布尔函数的方法完成 S 盒替换模块的电路设计. 用 EDA 软件进行验证, 实验结果表明处理器能够正确执行 S 盒替换操作且能够显著提升性能.

关键词: RISC; 微处理器; S 盒; 指令扩展; 布尔函数

中图分类号: TP302.1

文献标识码: A

文章编号: 1000-7180(2014)05-0053-05

S-box Substitution Instruction Extension of RISC Microprocessor

GUO Yi-hui, LI Sen-sen, DAI Zi-bin, ZHANG Li-chao

(PLA Information Engineering University, Zhengzhou 450000, China)

Abstract: Low efficiency is a main problem when the RISC microprocessors implement S-box substitution operation. In order to attack it, an approach is proposed in this paper, which is done by extending instructions through designing special module and instruction for S-box substitution, thus raising the performance of it. A new kind of S-box substitution module and operation instruction is designed based on the characteristics of S-box substitution operation of block cipher algorithm, which can support the four common modes of S-box substitution module, i. e., 8-32, 8-8, 6-4, 4-4 mode. The circuit implementing S-box substitution module is then designed adopting the Boolean function method. The concept is finally verified in EDA software, which shows the validity of the method and reveals the significant improved performance.

Key words: RISC; microprocessor; S-box; extend instruction; boolean function

1 引言

S 盒在分组密码算法中占有至关重要的地位, 在电路实现过程中, S 盒的实现直接决定了整个分组密码算法的性能和代价^[1].

在 RISC(精简指令集处理器)微处理器上编写软件可以实现 S 盒替换, 这种方式的优点是灵活性强, 可以实现各种模式的 S 盒替换操作, 但由于没有针对 S 盒替换的特定运算指令, 所以运算速度很慢, 降低了处理器的效率.

通过设计专用指令集, 使程序设计者能够使用专用指令实现 S 盒替换, 这一设计思路使 S 盒替换

操作具有高性能的同时又具有很高的灵活性, 能够广泛地适应不同密码算法的需求. 在不改变 RISC 微处理器总体结构的基础上, 设计支持常用模式的 S 盒替换模块和运算指令. S 盒替换模块要能够适合大部分 RISC 微处理器结构特点, 指令符合 RISC 指令格式, 并且不能破坏系统流水线结构.

2 S 盒硬件设计

2.1 S 盒特征分析

对大量的分组密码算法进行分析, S 盒替换常见的模式有 8-8、8-32、4-4、6-4 四种, 这四种模式占到了整个 S 盒模式的 83.33%. 因此, 本文在设计 S

盒替换电路时考虑支持 8-8、8-32、4-4、6-4 四种模式. 表 1 列出了不同 S 盒替换模式的使用频率^[2].

表 1 S 盒替换模块使用频率

S 盒模式	使用频率/%
8-8S 盒	33.33
8-32S 盒	30.00
4-4S 盒	13.33
6-4S 盒	6.66
其他模式	16.66

S 盒替换模块要支持 4 种模式, 理论上以 4-4 模式的 S 盒为基本单元通过级联可以实现其他三种模式的 S 盒, 但这种方式会使 8-8 和 8-32 模式的 S 盒级联网络变得相当复杂, 增加电路资源和延时, 降低了整个替换模块的性能. 考虑到 8-8 模式和 8-32 模式的 S 盒的使用频率最高, 为了提高电路的性能和降低结构复杂性, 本文采用 8-8 的 S 盒为基本单元.

2.2 基本单元设计

S 盒替换实际上就是一组布尔函数, S 盒的输入就是布尔函数的自变量, S 盒的输出就是函数值. 布尔函数实现的原理如下, 设输入为 $x = \sum_{i=0}^{n-1} x_i 2^i = (x_{n-1}, x_{n-2}, \dots, x_1, x_0)_2$ 的 $n-m$ 模式 S 盒布尔函数的通式为

$$f(x) = \sum_{i=0}^{2^n-1} c_i x_{n-1}^{i_{n-1}} x_{n-2}^{i_{n-2}} \cdots x_1^{i_1} x_0^{i_0} \quad (1)$$

式中, 系数 $c_i \in \{0, 1\}$, $i = \sum_{k=0}^{n-1} i_k 2^k = (i_{n-1}, i_{n-2}, \dots, i_1, i_0)_2$. 通项记为 $P_i(x) = \prod_{i=0}^{n-1} x_i^{i_i} = x_{n-1}^{i_{n-1}} x_{n-2}^{i_{n-2}} \cdots x_1^{i_1} x_0^{i_0}$, 则

$$f(x) = \sum_{i=0}^{2^n-1} c_i P_i(x) \quad (2)$$

为了使 S 盒能够支持不同的加 / 解密算法, 必须能够通过编程改变 S 盒模块实现的函数关系. 从式(2)中可以看出对输出产生影响的除了函数的输入还有最小项的系数, 在输入不变的情况下, 通过改变最小项系数就可以使 S 盒实现任意布尔函数的变换^[3].

从上面的分析知道, 任意布尔函数的输出是输入最小项的和, 因此在设计电路时应该把 n 项之积和 2^n 项之和作为固定的电路结构而把控制编码所对应的电路结构作为可变结构. 使用与或阵列可以实现任意布尔函数输出, 则 8 输入 8 输出的可编程 S

盒布尔函数实现模块如图 1 所示.

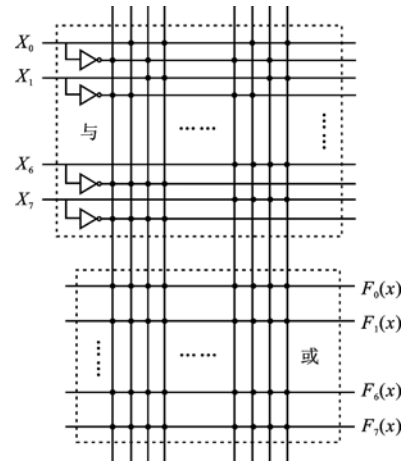


图 1 可编程 S 盒布尔函数实现模块

在图 1 中, 与阵列中的黑点是普通连接点, 而或阵列中的黑点是配置点. 每一个配置点可以用一个二输入的与门实现, 与门输入的一端是 $p_i(x)$, 另一端输入是 c_i , 而与门的输出就是 $c_i p_i(x)$. 或阵列配置点的微结构如图 2 所示.

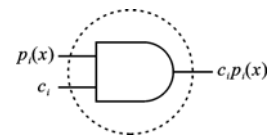


图 2 或阵列配置点的微结构

图 2 中配置数据 c_i 来源于模块内部的配置寄存器, 通过简单的分析一个 8 输入 8 输出的可编程 S 盒替换模块需要 256 bytes 的配置信息.

2.3 电路优化

按照上文内容, 取 $n = 8, m = 8$, 则图 1 中的与项可以表示为: $P_0(X) = \sim X_7 \& \sim X_6 \& \sim X_5 \& \sim X_4 \& \sim X_3 \& \sim X_2 \& \sim X_1 \& \sim X_0$, $P_{255}(X) = X_7 \& X_6 \& X_5 \& X_4 \& X_3 \& X_2 \& X_1 \& X_0$, 其他项与此相似, 不再一一列出; 或项表示为: $f[0](X) = C[0]_{255} P_{255}(X) \mid C[0]_{254} P_{254}(X) \mid \cdots \mid C[0]_1 P_1(X) \mid C[0]_0 P_0(X)$, $f[7](X) = C[7]_{255} P_{255}(X) \mid C[7]_{254} P_{254}(X) \mid \cdots \mid C[7]_1 P_1(X) \mid C[7]_0 P_0(X)$ 其他或项与此相似. 每个与项有 8 个输入, 每个或项有 256 个输入, 电路输出表达式复杂, 用电路实现以后延迟太大, 不能应用到实际中, 但是经过优化以后, 逻辑电路产生的路径延迟在允许范围之内^[4]. 对于一个 8 输入的与项, 经过优化路径延迟为 3 级与门的延迟时间, 考虑输入取反, 最大路径延迟为 3 级与门加 1 级非门, 图 3 所示为 8 输入的与项经过优化以后得到的电路形式.

同样对或项进行优化, 优化以后的电路结构如

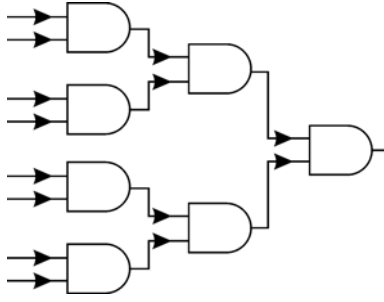


图3 与项优化后的电路结构

图4所示,一个256输入的或项经过优化以后的电路有8级二输入或门的延迟,加上与项和配置点延迟,8-8S盒替换单元输入到输出最大路径延迟为1级非门加4级与门加8级或门,这样一个延时在电路设计中是允许的.上面分析了8输入1输出的布尔函数电路结构,8-8S盒替换单元相当于8个8-1布尔函数电路的并列,输出的项数不会影响整体电路的延时.

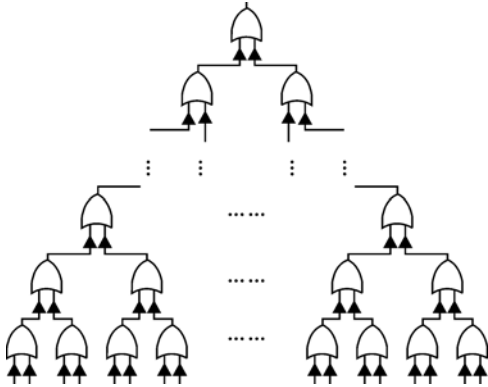


图4 或项优化以后的电路结构

上文设计了8输入布尔函数的电路结构并进行了优化,下面分析在此电路结构基础上,如何实现64输入的布尔函数电路.电路输入为 $X = (X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0)$,取 $X_7 X_6$ 固定为00,输入变成 $X = (0, 0, X_5, X_4, X_3, X_2, X_1, X_0)$,电路的与项 $P_{64}(X)$ 到 $P_{255}(X)$ 全部变成0,则 $f[0](X) = C[0]_{255} P_{255}(X) \mid C[0]_{254} P_{254}(X) \mid \cdots \mid C[0]_1 P_1(X) \mid C[0]_0 P_0(X)$ 变成了 $f[0](X) = C[0]_{63} P_{63}(X) \mid C[0]_{62} P_{62}(X) \mid \cdots \mid C[0]_1 P_1(X) \mid C[0]_0 P_0(X)$.电路的输出是 $X = (X_5, X_4, X_3, X_2, X_1, X_0)$ 输入的布尔函数.对于4输入原理也是一样的,高四位固定为0即可,由此可知,使用8-8为基本单元,能够支持其他三种模式的S盒替换.

3 S盒扩展结构设计

本文选用目前应用广泛的MIPS处理器为基础,在其硬件架构上设计S盒替换模块.图5为嵌入

S盒替换模块的仿MIPS处理器的整体结构图.

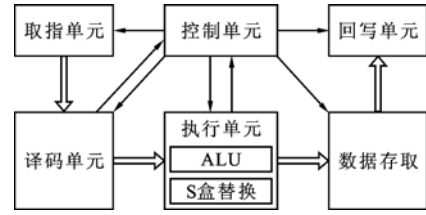


图5 仿MIPS处理器的整体结构

图5中,仿MIPS处理器流水线分为5级,分别为:取指、译码、执行、数据存取、回写.S盒替换模块在流水线中的位置有三种情况:位于执行级、位于数据存取级、位于执行级和数据存取级.前两种情况S盒只占用流水线的一级,这种情况S盒替换模块设计要求简单,但是如果设计的S盒替换模块关键路径太长,就会限制系统的时钟频率;第三种情况占用两级流水线,设计要求复杂,需要考虑数据冲突等问题,但是时钟频率可以达到前两种情况的两倍,适合应用于处理器要求性能高的场合.对于其他非MIPS结构的微处理器,有些流水线不包含数据存取级,为了本文的研究能够应用于其他结构的处理器,降低设计难度,S盒替换模块在流水线中的位置按第一种情况放置.

由于在分组密码算法中需要并行查找S盒,为了能够满足并行替换的要求,替换模块由8个8-8S盒替换单元组成,S盒替换模块总体结构如图6所示.

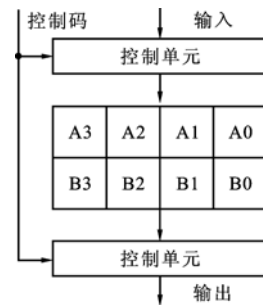


图6 S盒替换模块结构图

图6中A0-A3、B0-B3是8个基本的8-8S盒替换单元.S盒替换模块根据控制码进行不同模式的操作.

4 S盒指令设计

4.1 操作研究

(1)当S盒类型为8-8时,对于32bit输入将分为四组位宽为8bit的数据分别进行替换,即4个字节.每个字节独立进行8-8的S盒替换操作,组合逻辑将4个独立输出的8bit数据拼接为32bit数据

并送入目的寄存器,操作过程如图 7 所示.

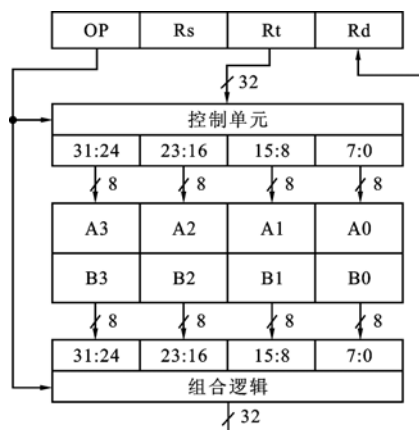


图 7 8-8 类型 S 盒操作

图 7 中, S 盒替换模块包含 8 个 8-8 替换单元, 8 个单元分为 A 组和 B 组, 每个分组可以独立完成一个 32 bit 输入 32 bit 输出的替换操作, 指令执行 8-8 替换时可以选择使用 A 组或 B 组. 设置两组替换单元是为了执行逆 S 盒替换时不需要重新配置参数, 第二个原因就是执行小规模替换操作时不需要拆分 S 盒.

(2) 当 S 盒类型为 8-32 时, 32 bit 的输入数据中只有低 8 bit 为有效数据, 最低字节同时送给 4 个替换单元, 替换单元输出的 4 个 8 bit 数据拼接为 32 bit 数据以后送入目的寄存器. 同 8-8 替换操作一样, 可以选择 A 组和 B 组.

(3) 当 S 盒类型为 4-4 时, 只要高 4 位固定为零, 8-8 替换单元就可以实现 4-4S 盒替换操作, 32 bit 的输入数据将分裂为 8 组 4 bit 数据, 作为 S 盒替换单元输入低 4 位, 高四位经过控制单元以后被 0 填充. 每 4 bit 数据独立进行 4-4 的替换操作, 组合逻辑将输出的 8 组 4 bit 数据拼接为 32 bit 数据, 送入目的寄存器, 操作过程如图 8 所示.

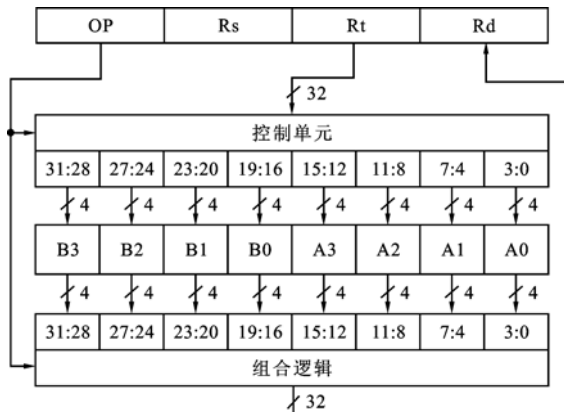


图 8 4-4 类型 S 盒操作

(4) 当 S 盒类型为 6-4 时, 输入数据需要 2 个操

作数即 2 个 32 bit 数据, 其中输入的第二个源操作数的低 16 bit 与第一个操作数拼装成为一个 48 bit 数据, 分为 8 个 6 bit 数据, 分别进行 6-4 的替换操作, 输出的 32 bit 数据送入目的寄存器.

4.2 指令设计

(1) 配置指令

为了实现 S 盒算法, 无论是四种模式的那一种, 都需要在进行 S 盒替换操作以前, 将提前计算好的配置信息注入到 S 盒替换模块^[5]. 设计的 S 盒配置指令表示如下:

指令:

PUSH_SBOX Rt, Rs, Imm

指令描述:

Imm 为 S 盒基本单元编号, 0-3 对应 A0-A3, 4-7 对应 B0-B3, 8 对应 A 组全体单元, 9 对应 B 组全体单元, 10 对应 A、B 组全体单元. Rs 为 S 盒基本单元中配置寄存器地址, Rt 为配置数据. 该指令的功能就是将 Rt 中的数据配置到 Imm 所指示的基本单元中地址为 Rs 的寄存器

配置指令主要在微处理器执行程序初始时刻对 S 盒替换模块进行配置, 根据不同的算法选择相应的 S 盒单元进行配置, 有些算法并行替换使用的 S 盒相同, 可以通过指令同时配置, 不需要每个单元单独配置, 提高配置效率. 配置数据需要根据具体 S 盒替换算法提前计算好.

(2) 替换指令

由于设计的 S 盒替换模块支持四种类型, 所以设计的指令也应该支持四种类型的 S 盒替换操作. 则 SBOX 替换操作的指令设计如表 2 所示.

表 2 S 盒替换指令

S 盒类型	指令格式	指令描述
8-8	SBOX8T8a Rd, Rt	S 盒 A 组替换
	SBOX8T8b Rd, Rt	S 盒 B 组替换
8-32	SBOX8T32a Rd, Rt	S 盒 A 组替换
	SBOX8T32b Rd, Rt	S 盒 B 组替换
4-4	SBOX4T4 Rd, Rt	S 盒替换
6-4	SBOX6T4 Rd, Rs, Rt	S 盒替换

Rt, Rs 为源寄存器, Rd 为目的寄存器

表 2 中, Rt, Rs 为存放 S 盒替换模块输入数据的寄存器, Rd 为存放 S 盒替换模块输出数据的寄存器.

5 性能分析

以上设计采用 verilog 语言描述, 用 EDA(Elec-

tronic Design Automation)软件对多组数据进行测试仿真,均得到正确结果。

以 Rijndael 中 8-8 的 S 盒替换操作的查表动作为例^[6], 32 bit 数据要完成 8-8 的 S 盒查找表变换需要 $3(\text{SHIFT}) + 3(\text{AND}) + 4(\text{ADD}) + 4(\text{LOAD}) + 3(\text{SHIFT}) + 3(\text{OR}) = 20$ 条指令。依此类推,实现 8-32、4-4、6-4 的 S 盒替换^[7]也需要较多的指令才能完成。使用本文指令扩展以后的微处理器实现相同 S 盒替换只需要一条指令,性能提升至少 20 倍。

为了得出 S 盒替换模块用 ASIC 实现以后的电路性能,本文使用 $0.18\ \mu\text{m}$ 及 6 层金属工艺通过 EDA 软件综合,得出的电路最大延时为 $2.76\ \text{ns}$,工作频率可以达到 333 MHz,电路面积为 143 693 (门),而著名的 OpenRISC1200 开源处理器使用相同的工艺主频运行在 300 MHz,默认配置为 100 万门^[8],可见本文设计的 S 盒替换模块完全能够实际应用微处理器中。

6 结束语

本文在通用微处理器增加 S 盒替换模块、设计专用指令,实现了通用微处理器进行 S 盒替换的操作,具有良好的编程性而且具有较高的性能。在以后研究中,可以设计其他专用指令,使通用微处理器可以实现完整的密码运算。

参考文献:

- [1] 高娜娜,王沁,李占才.基于 AES 和 DES 算法的可重构 S 盒硬件实现[J].小型微型计算机系统,2006,27(3):446-449.
- [2] 张琰. RISC 结构专用指令密码处理器研究与设计[D]. 郑州:解放军信息工程大学,2008.
- [3] 杨晓辉.面向分组密码处理的可重构设计技术研究[D]. 郑州:解放军信息工程大学,2007.
- [4] 韩维维.高级加密标准 S 盒的 ASIC 实现[J].计算机工程与设计,2008,29(9):2222-2223.
- [5] 戴紫彬.面向分组密码处理的协处理器体系结构研究与设计实现[D]. 郑州:解放军信息工程大学,2007.
- [6] 杨宏志,韩文报,赵龙.适于硬件实现的 s 盒构造方法[J].计算机应用,2010,30(3):674-676.
- [7] 孟涛,戴紫彬.可重构 S 盒运算单元的设计与实现[J].电子技术应用,2007,33(5):139-141.
- [8] 倪继利,陈曦,李挥. CPU 源代码分析与芯片设计及 Linux 移植[M]. 北京:电子工业出版社出版社,2009.

作者简介:

郭怡惠 男,(1988—),硕士研究生.研究方向为密码 SoC 设计.

李森森 男,(1988—),硕士研究生.研究方向为密码 SoC 验证.

戴紫彬 男,(1966—),教授,博士生导师.研究方向为密码通信微系统设计.

张立朝 男,(1979—),讲师.研究方向为网络安全技术.

(上接第 52 页)

作者简介:

吴君钦 男,(1966—),副教授.研究方向为无线通信、嵌入式系统.

卢陶 男,(1989—),硕士研究生.研究方向为嵌入式系统.