基于 FPGA 的 DES 加密算法的实现

赵莲清，王亚美
（华北电力大学 电气与电子工程学院，北京 102206）

摘 要：介绍了 DES 算法的基本原理与加密过程，给出了基于 FPGA 的 DES 算法的硬件实现，并且用 Quartus II 实现了模块仿真，验证了 DES 加密算法。

关键词：DES 算法；FPGA；仿真

Implementation of DES encryption based on FPGA

ZHAO Lian Qing, WANG Ya Mei
(Department of Electric and Electronic Engineering , North China Electricity Power University, Beijing 102206, China)

Abstract: The paper introduced the basic principle of DES algorithm and its encryption process in detail. It provided the implementation of DES based on FPGA and gave simulation results with Quartus II, which confirmed DES algorithm.

Key words: DES algorithm; FPGA; simulation

随着网络通信技术的发展，如何保护数据传输过程的安全已成为一项急迫的要求。政府部门、金融行业、通信行业、情报等系统都非常重视信息的安全。密码安全技术则是所有安全服务的基础。数据加密是信息安全的重要手段，DES 密码算法是最有代表性的分组加密算法，1976 年被美国政府采用，随后又被美国国家标准局和美国国家标准学会承认，发展至今已成为工业界的标准密码算法，目前广泛应用于保密通信中。DES 算法用 64 位的密钥对 64 位的明文加密，64 位密钥中每 8 位就有一奇偶校验位，不参与运算，因此有效密钥只有 56 位。此外，由于该算法的对称性，其加密和解密运算过程完全相同，只是在迭代运算时子密钥的使用顺序不同[1]。DES 加密算法有各种实现方案，本文提出一种 DES 加密算法的硬件实现方法。

DES 算法主要包括：初始置换、16 轮迭代的乘积变换、初始逆置换以及 16 个子密钥产生器[2]。DES 加密过程如图 1 所示。

初始置换主要用于对明文中的各位进行换位，目的在于打乱明文中各位的排列次序，输出结果按奇偶分为左右两路送入乘积变换部分。在这个过程中不使用密钥，仅仅对 64 位码进行移位操作。而进行逆初始置换是为了使加解密统一使用同一种算法。初始置换与初始逆置换只是将 64 位输入的数据按位进行重新排列，只需要一些输入输出端口，按照置换规则把输入与输出对应关联即可[3]。下面只介绍密钥的产生与 16 轮迭代乘积变换模块的设计。

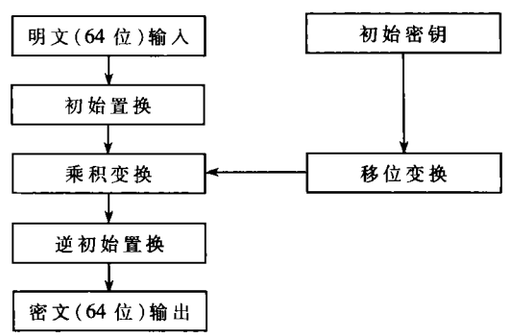

图 1 DES 加密过程

1 子密钥生成模块

DES 算法每一轮次迭代都需要一个子密钥，要实现 DES 算法就需要提前生成子密钥，并且按照时序准确地传递给迭代过程。

子密钥产生模块由选择控制、循环移位控制等部分构成。该模块的输入是 64 位初始密钥，输出为参加每次迭代运算的子密钥，另外还有一个控制信号(time)，用来控制在不同迭代过程中移位的次数。本设计采用硬件描述语言(VHDL)按照子密钥产生的过程，通过置换选择 1、循环移位、置换选择 2，一步一步得到子密钥的。循环移位仿真结果如图 2 所示。图中，c0、d0 是经过置换选择 1 后得到的前后 28 位，Time 为控制移位次数的控制信号，k1 为移位后的结果，再经过置换选择 2 即可得子密钥。

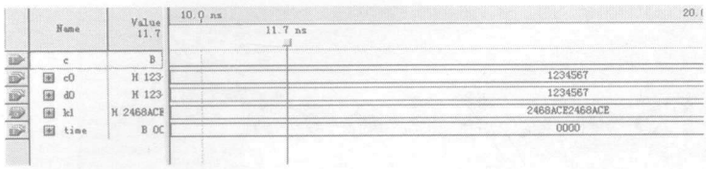

图 2 循环移位过程仿真结果

因为每轮移位的次数不同，所以每轮子密钥产生的时间也不相同，所以会给迭代运算带来延迟，这也是采用 VHDL 设计存在的不足。从 VHDL 设计结果可以看出，原密钥与每轮子密钥之间有固定的关系，因此，可以通过软件分析，建立子密钥相对于原始密钥的关系表，在硬件实现时直接使用此关系表即可，这样也可以有效地节省硬件资源。

2 迭代变换

DES 算法是典型的迭代分组密码算法，实现过程的核心是 16 轮次相同的迭代运算。输入的 64 位明文先执行初始置换对明文进行换位处理；然后通过子密钥 k1~k16 对明文进行 16 轮乘积变换，即进行 16 次迭代处理；最后经过逆初始置换的处理，得到 64 位的密文输出。16 次迭代的目的是使明文增加混乱性和扩散性，避免输出密文残留统计规律，使破译者无法反向推算出密钥。

轮迭代运算中的 f 函数是非线性的，它是每轮实现混乱和扩散的最关键的模块，也是整个加密算法的核心，它包括 E 盒扩展置换、S 盒置换和 P 盒置换，其基本过程如图 3[4]所示。其中，E 盒扩展置换、P 盒置换的原理和实现方法与初始置换以及逆初始置换类似，运算过程都是线性的，而 S 盒是一个复杂的非线性函数，正是经过了 S 盒的非线性变换，才使算法达到很好的“混乱”效果，从而具有较强的安全性。因此，S 盒的设计是 DES 算法的关键部分，S 盒设计的优劣将影响整个算法的性能。

基于 S 盒的表现形式是矩阵形式，因此在使用 VHDL 语言建立 S 盒模型时，一般都会想到直接用多重选择 CASE 语句[5]。在这里即采用

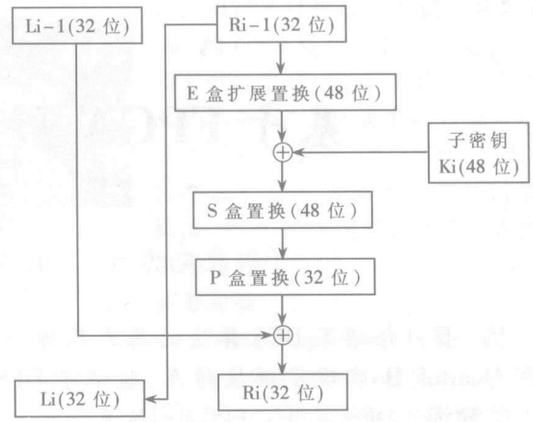

图 3 迭代过程

CASE 语句实现。用 CASE 选择语句实现 S 盒有两种方式：一种是直接使用 S 盒的 6 位输入为 6 个变量的 CASE 语句方式，另一种方式是使用双重 CASE 嵌套语句，即外层使用 2 个输入控制 S 盒的横向选择；内层使用 4 个输入控制 S 盒的纵向选择。采用双重 CASE 语句可以直接定位输出结果。这两种方式下占用资源的情况如表 1 所示。

根据表 1 比较的结果可以看出，选择双重 CASE 语句建立 S 盒模型可以大大节省资源，其仿真结果如图 4 所示。s0 为 S 盒的输入，s2 为 S 盒的输出，这样就完成了非线性的变换。图 5 为通过 Quartus II 实现的基于

表 1 两种 CASE 语句建立 S 盒模型占用资源对比表[3]

使用的方法	所占用的资源(LE 的个数)	所用的门
6 个变量的 CASE 语句	960	16 600
双重 CASE 语句	431	7 500

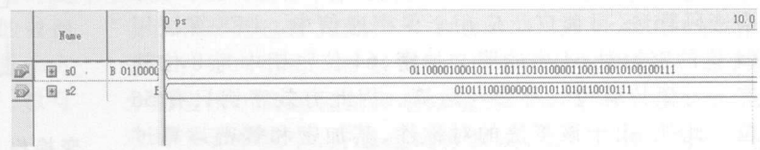

图 4 S 盒仿真结果

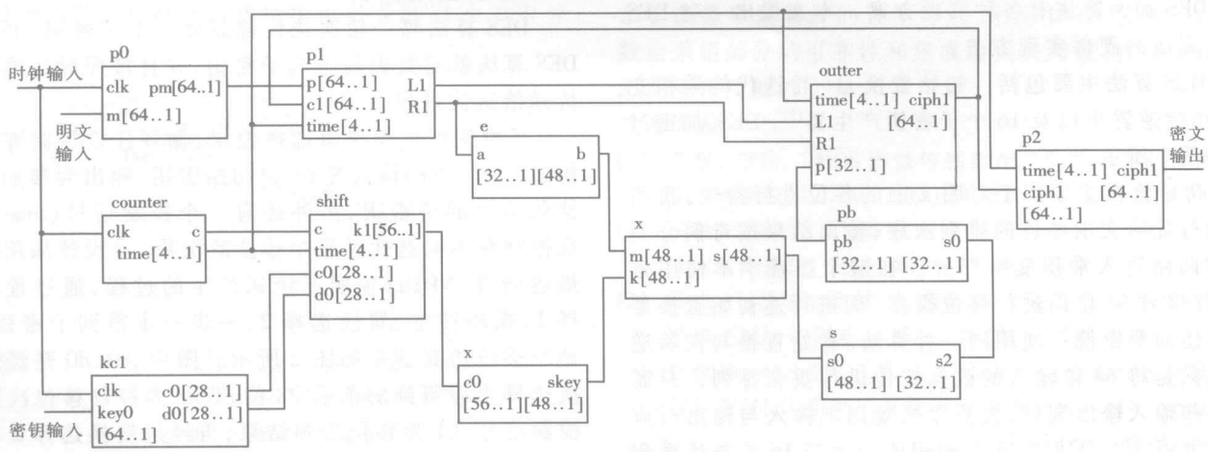

图 5 DES 加密过程仿真线路图

(下转第 139 页)

表 1 修改的双 Booth 2 编码表

MUX1|CIN1 &MUX2|CIN2		c_0 低 4bit							
		0000	0001	0010	0011	0100	0101	0110	0111
p_0 低 4bit $GF(p)$	0001	0|0&0|0	0|0&-p|1	0|0&-2p|1	-p|1&p|0	-p|1&0|0	-p|1&-p|1	-p|1&-2p|1	-2p|1&p|0
	0011	0|0&0|0	p|0&p|0	-p|1&-2p|1	0|0&-p|1	p|&0|0	-2p|1&p|0	0|0&-2p|1	p|0&-p|1
	0101	0|0&0|0	p|0&-p|1	-2p|1&-2p|1	-2p|1&p|0	-p|1&0|0	0|0&-p|1	p|0&-2p|1	p|0&p|0
	0111	0|0&0|0	-2p|1&p|0	p|0&-2p|1	-p|1&-p|1	p|&0|0	-p|1&p|0	-2p|1&-2p|1	0|0&-p|1
p_0 低 4bit $GF(2^m)$	0001	0|0&0|0	0|0&p|0	p|0&-2p|1	p|0&-p|1	p|&0|0	p|0&p|0	-2p|1&-2p|1	2p|0&-p|1
	0011	0|0&0|0	p|0&-p|1	-2p|1&-2p|1	-2p|1&p|0	-p|1&0|0	0|0&-p|1	p|0&-2p|1	p|0&p|0
	0101	0|0&0|0	p|0&p|0	-p|1&-2p|1	0|0&-p|1	p|&0|0	-2p|1&p|0	0|0&-2p|1	p|0&-p|1
	0111	0|0&0|0	2p|0&-p|1	0|0&-2p|1	p|0&p|0	-p|1&0|0	p|0&-p|1	-p|1&-2p|1	0|0&p|0

乘数 A 每次扫描 4 位，分成两组，每组 2 位，经 Booth 2 编码后得到 3 位的输入数据控制数据选择器的输出。函数 f 通过编码器实现，有限域 $GF(p)$ 与 $GF(2^m)$ 上的 q 值计算不一样，因此需要同时编码并通过域选择信号 field 选择输出。编码输出的四位 q 值再次经过双 Booth 2 编码控制数据选择器的输出。上述分两步的编码方式其中间值 q 是可以省略的，直接修改 Booth 2 编码的输入信号，一次编码完成，部分编码如表 1 所示。

3 实现结果

基于双 Booth 2 编码的双有限域模乘法器作为核心部件应用在双有限域 ECC 协处理器中，根据协处理器的实际需要，处理的最长操作数为 384 位，因此模乘法器采用 4 级流水；b_i、p_i 的字宽为 32 位，便于与常用总线连接。这种设计既考虑了处理速度，又兼顾了电路占用的面积。模乘法器用 Verilog 语言描述，采用 Synopsys 公司的 Design Complier 在 SIMC 0.18μm-typcal 工艺库下综合，等效“与非门”为 4.5 万门，最高工作频率可达 280MHz，完成一次 GF(p)上的 256bit 模乘运算只需要 0.51μs。表 2 是本文设计的模乘法器与已发表文献中同类设计的比较结果。表中结果显示，本文设计的速度比最好的已发表的设计[3]提高了 16%。

表 2 模乘计算时间比较

设计	操作数长度/bit	工艺库/μm	模乘时间/μs@256b	时钟频率/MHz	适应有限域
参考文献[2]	256	0.5	6.6	80	双域
参考文献[3]	0~1 024	0.18	0.7	200	$GF(p)$
参考文献[4]	0~512	0.18	0.81	384	双域
本文的设计	0~384	0.18	0.51	280	双域

双有限域模乘法器使用同一套硬件电路实现有限域 $GP(p)$ 与 $GF(2^m)$ 上的 ECC 模乘运算，节约了硬件成本，扩展了运用空间。本文基于双 Booth 2 编码对基为 16 的 Booth 4 编码的模乘法器进行了优化，使得电路更加规范、简单。从实现结果可以看出，本文的设计有效地减小了关键路径延迟，提高了模乘法器的运算速度。

参考文献

[1] MONTGOMERY P L. Mondular multiplication without trial division. Mathematics of Computation, 1985,44(7):519-521.

[2] TENCA A F, SAVAS E, KOC C K. A design framework for scalable and unified multipliers in $GF(p)$ and $GF(2^m)$. International Journal of Computer Research, 2004,13(1):68-83.

[3] FAN Yi Bo, ZENG Xiao Yang, GANG Yi Yu,et al. A modified high-radix scalable montgomery multiplier. IEEE International Symposium on Circuit and System(ISCAS), Island of Kos, Greece, May. 2006.

[4] 史焱,吴行军.高速双有限域加密协处理器设计.微电子学与计算机,2005,22(5):8-12.

(收稿日期:2007-10-30)

(上接第 136 页)

FPGA 的 DES 加密算法的硬件仿真线路图。

本文通过对各个模块特别是 S 盒与子密钥生成模块的详细分析，给出了 DES 加密的一种实现方法，并在此基础上对部分模块进行了验证仿真，给出了硬件仿真线路图。这种设计方法是根据功能模块分层进行的，因此可以节省设计时间，减少设计输入的错误，简化验证的过程。

参考文献

[1] STALLINGS W. Cryptography and network security principles and practices[M]. prentice Hall, 1996.

[2] 胡向东,魏琴芳.应用密码学[M].北京:电子工业出版社,2006.

[3] 高献伟,周玉坤,路而红,等.DES 算法硬件实现的研究[J]. 北京电子科技学院学报, 2001,(1):11-15.

[4] 张福泰.密码学教程[M].武汉:武汉大学出版社,2006.

[5] 李永彬,雷 菁.DES 加密算法的高速 FPGA 实现[J]. 电子工程师,2005,(7):39-40.

(收稿日期:2007-12-12)