中国集成电路
China Integrated Circuit
设计

RSA密码算法的可重构设计与实现

伍彬山，王云峰，刘智超，刘天翔
（厦门大学 电子工程系，福建 厦门 361005）

摘要：本文对RSA密码算法的实现和可重构性进行了分析，在对模幂模块和模乘模块进行了可重构设计的基础上，提出一种可重构RSA硬件架构，使其能够适配256bit、512bit、1024bit、2048bit四种不同密钥长度的应用。RSA可重构设计在FPGA上进行了实现与测试，结果表明，工作在200MHz时钟时，2048bit密钥长度RSA在最坏情况下数据吞吐量可达46 kb/s，能够满足高性能的信息安全系统对RSA算法的加密速度要求。

关键词：RSA；模乘运算；模幂运算；可重构设计
中图分类号：TP339　　**文献标识码**：A

Reconfigurable Design and Implementation of RSA Algorithm

WU Bin-shan,WANG Yun-feng,LIU Zhi-chao,LIU Tian-xiang
(Department of Electronic Engineering,Xiamen University,Xiamen 361005,China)

Abstract: In this paper, the implementation and reconfigurable feature of RSA cryptographic algorithm are analyzed. On the basis of the Reconfigurable design of the Modular Multiplication and Modular Exponentiation, we propose the reconfigurable RSA hardware architecture, which is able to fit 256bit, 512bit, 1024bit, 2048bit four applications of different key length. The RSA reconfigurable design and testing were carried out to achieve results, which show that in the worst case, 2048bit RSA get the data throughput achieved 46 kb/s when work in the 200MHz clock. It is able to meet the high- performance information security systems RSA encryption algorithm on the speed requirement.

Keywords: RSA; Modular Multiplication; Modular Exponentiation; Reconfigurable Design

1 引言

随着计算机网络的普及与发展，信息安全问题显得格外重要。以RSA密码算法[1]为代表的公钥密码体制[2]在保证数据的机密性、完整性以及签名和认可等方面的突出优点已经使其成为当今网络安全中最重要的解决方法，相应的密码芯片在网络中得到了广泛的应用。

目前，大多数密码芯片是实现一种固定密码算法的专用芯片，不能满足用户们的不同层次的安全性能和预留密码算法升级空间的要求。因此，近年来

国内外许多机构和个人都致力于可重构密码芯片设计的研究[3-6]。可重构密码芯片是采用可重构体系结构设计而成的用于对数据进行加/解密处理的集成电路芯片。其内部逻辑电路可以根据不同密码算法的需求重新组织,构成不同的电路结构,实现不同的功能,从而能够灵活、快速地实现多种不同密码算法[3]。

本文在设计 RSA 算法实现时,综合考虑密钥长度、安全性、性能、面积等因素,在对模幂和模乘运算模块进行了可重构设计的基础上,提出了一种可重构 RSA 算法结构,在增加很少逻辑单元的情况下,使其能够适配 256bit,512bit,1024bit 和 2048bit 四种密钥长度的 RSA 算法应用,满足不同层次安全性的信息系统的需要。FPGA 的原型实现和验证结果表明,该设计能够满足高性能信息安全系统对于公钥密码加密速度的要求,可以作为可重用 IP,用于信息安全 SoC 设计。

2 RSA算法

RSA 密码算法的明文空间 M 与密文空间 C 相等,为 Z_n(表示 mod n 所组成的整数空间,取值范围为 0~n-1)。

RSA 算法描述如下[1]:

(1) 选择两个互异的大素数 p 和 q(保密),计算 $n=p\cdot q$(公开),$\varphi(n)=(p-1)\cdot(q-1)$(保密),选择一个随机数 e($0<e<\varphi(n)$)(公开),满足 $\gcd(e,\varphi(n))=1$。计算 $d=e^{-1}\bmod\varphi(n)$(保密)。确定:公钥 $KU=\{e,n\}$,私钥 $KR=\{d,n\}$。

(2) 已知:明文 $M<n$ 和公钥 $KU=\{e,n\}$。

计算密文:$C=M^e \bmod n$

(3) 已知:密文 C 和私钥 $KR=\{d,n\}$。

计算明文:$M=C^d \bmod n$

3 RSA算法实现与可重构分析

大数模幂运算是 RSA 公钥密码算法的核心运算,实现时可以利用模运算的基本性质:$[(a \bmod N)\times(b \bmod N)] \bmod N=(a\times b)\bmod N$,把模幂运算的中间结果对 n 取模,从而限制了中间结果的大小,实现更加容易。

3.1 模幂运算算法

本文模幂采用是左到右二进制位扫描算法。首先应该把指数 e 或者 d 表示成如下二进制形式:

$$e=[e_0,e_1,\ldots e_{n-1}]_2=\sum_{i=0}^{n-1} e_i 2^i=e_0+e_1 2+\ldots+e_{n-1}2^{n-1}$$

然后从 e 或者 d 的最高位扫描到最低位。

输入:m,e,N

输出:$c=m^e \bmod N$

```
{c=1;
 for  i=n-1 to 0 do
   {c=(c·c) mod N;
     If (e_i=1) c=(m·c) mod N;
   }
 return C;
}
```

由于算法同一次循环运算中的两次模乘运算数据相关,所以必须顺序处理,不能并行运算;但硬件实现时只需一个模乘运算单元,实现面积小。

3.2 模乘运算算法

模乘模块是左到右二进制位扫描模幂算法的主要运算单元,本文选择基于 2 的 Montgomery 算法[7-8]实现模乘运算。

基于 2 的 Montgomery 算法如下:

记 $S=Monprod(A,B,N)$。其中:$A=\sum_{i=0}^{k+2} A_i\cdot 2^i$,$B=\sum_{i=0}^{k} B_i\cdot 2^i$,$N=\sum_{i=0}^{k-1} N_i\cdot 2^i$,$A_i, B_i, N_i\in\{1,0\}$,$A_{K+1}, A_{K+2}=0$

输入:A,B,N

输出:$S=Monprod(A,B,N)=(A\cdot B\cdot 2^{-(K+2)})\bmod N$

```
{ S=0;
  for j=0 to K+2 do
```

```
{  if（S0=1） then S=S+N;
   S=（S/2）+Aj·B;
  }
 return S;
}
```

该算法主要由 2 个大数的加法组成。为了保证计算结果 S 小于 N，该算法的循环要执行 K+3 次[9]，因此输出的结果是$(A\cdot B\cdot 2^{-(K+2)})\bmod N$。

3.3 由 Montgomery 算法构造的从左到右二进制扫描位扫描法

由于 Montgomery 算法的结果并不是 $(A\times B)\bmod N$，而是$(A\cdot B\cdot 2^{-(K+3)})\bmod N$。所以必须对从左到右二进制位扫描法进行修改，首先，必须先进行一次预处理步骤，分别计算 $C=Monprod(1,R,N)$ 和 $M2=Monprod(M,R,N)$，其中 R 为与 N 相关的常数，$R=2^{2(n+3)}\bmod N$。其次，算法结束后再进行一次后处理步骤，计算 $C=Monprod(1,C,N)$，这样就可以消除模乘运算结果中多余的参数。具体算法如下：

输入：M，e，N，R

输出：$C=M^e \bmod N$

```
{
 M2=Monprod（M,R,N）
 C=Monprod（1,R,N）
 for i=n-1 to 0 do
   {
    C=Monprod（C,C,N）；
    If（ei=1） C=Monprod（C,M2,N）；
   }
 C=Monprod（1,C,N）
 return C;
}
```

3.4 可重构性分析

可重构设计以软件编程的方式快速灵活的实现不同硬件电路，克服了软件和硬件实现各自的不足，可以比软件实现有着更好的性能，同时比硬件实现更具有灵活性。可重构模块包含许多可由外部编程控制的计算单元，这些单元由一些可配置连线资源连接着，可以通过对连线资源的改变来形成需要的不同电路。

通过对模幂和模乘运算算法分析可知，不同密钥长度的 RSA 算法的主要部件大数加法运算模块是可重用部件，而差别主要是在模幂运算的循环次数 n、e_i 的起始位 e_x 和模乘算法的循环次数（K+3）。因此可以以大数加法运算模块为重构元素进行可重构设计，设置 n、K+3 和 e_x 为可控节点，通过输入信号对可控节点进行可编程控制。例如：当密钥长度为 1024 的时候，选择 n 为 1024、e_x 为 e[1023]、K+3 为 1027；而当密钥长度为 2048 的时候，选择 n 为 2048、e_x 为 e[2047]、K+3 为 2051。

4 RSA算法可重构设计

基于模幂、模乘模块的可重构性，本文提出了一种可以根据密钥长度的不同进行可重构的 RSA 算法实现结构，结构框图如图 1 所示，分为模乘模块（包括模乘控制器和模乘运算单元）、模幂模块、存储模块三个部分。

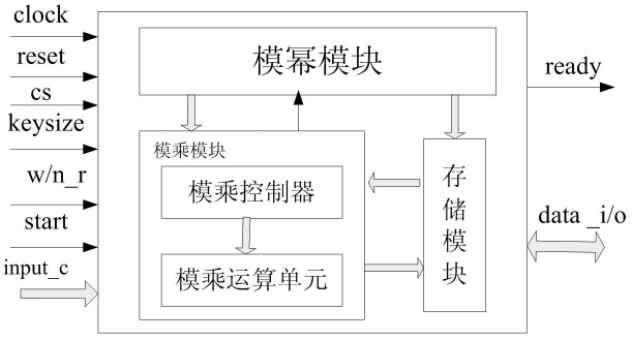

图 1 可重构 RSA实现总体结构

图 1 中各个信号的定义如表 1 所示。

4.1 模乘模块

模乘模块包括模乘运算单元和模乘控制单元。实现模乘的主要运算是大数的加法。可重构 RSA 算法的最长密钥长度是 2048bit，所以模乘运算模块为

表 1 外围接口定义

信号名称	位宽	方向	描述
clock	1	输入	时钟信号
reset	1	输入	复位信号
cs	1	输入	片选信号
keysize	2	输入	密钥长度选择信号
w/n_r	1	输入	读写控制信号
start	1	输入	运算启动信号
input_c	5	输入	输入数据类型选择信号
ready	1	输出	运算完成信号
data_i/o	8	双向	数据输入输出信号

2048位。由于 2048 位的加法是循环进行的，使用进位保留加法器 CSA（carry saved adder）是很好的解决方案[10]。CSA 将加法结果的和数和进位分别用 S 和 C 表示，即进位用 C 保留下来，作为下一次加法的进位输入。一个一位全加器中，输入 A、B 是被加数，输入 C 是上次加法保留的进位。一个 k 位的 CSA 是由 k 个 1 位全加器并行组成的，如图 2 所示。

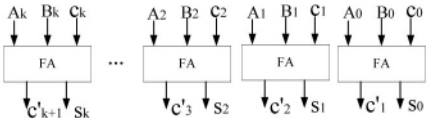

图 2 k 位 CSA结构

第 i 位的结果 S_i 和 C_{i+1} 同输入之间的关系为：

$S_i=A_i \oplus B_i \oplus C_i$ ；

$C_{i+1}=A_i B_i+A_i C_i+B_i C_i$ 。

CSA 加法器的特点是不会随着位数的增加而产生冗长的进位链，这样既能提高速度，又简化了硬件结构。

由 CSA 构造的模乘运算单元如图 3 所示。用两个 CSA 加法器来实现模乘运算中的两次加法。

在模乘算法循环结束后需要将两个 2048bit 的数相加得到结果。使用一个 32 位的 CPA（carry propagation adder）将 2048bit 位的加法分成 64 个时钟周期完成，CPA 的运算结果就是模乘运算的最终结果。CPA 加法器的结构如图 4 所示。

模乘控制单元主要由计数器来实现，不同密钥

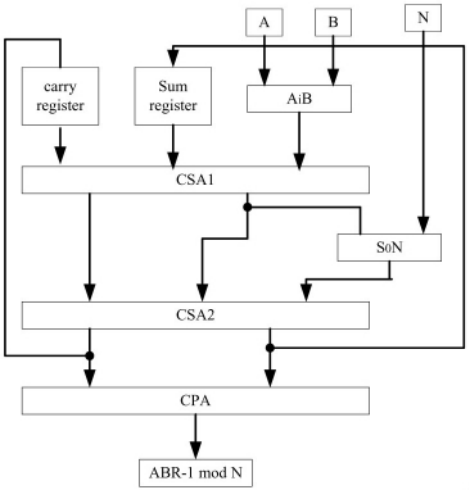

图 3 模乘运算单元结构

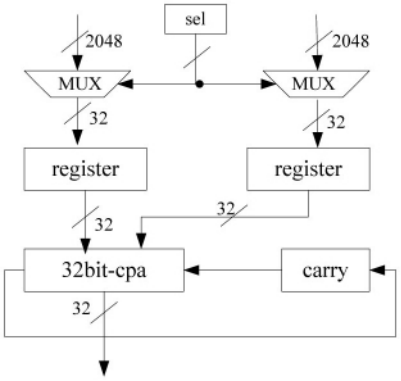

图 4 CPA结构

长度的 RSA 算法的模乘循环次数不一样，完成一次循环需要一个运算周期，所以计数器的初始载入值也不一样。可重构 RSA 算法实现由外部输入信号 keysize 对初始载入值的选择进行控制。初始载入值为密钥长度加上 67 （3 次增加的循环次数和 64 个 CPA 加法周期）。例如，当 keysize 为 “00” 时，初始载入值为 323 ；当 keysize 为 “01” 时，初始载入值为 579。每完成一次循环，计数器减一。当计数器为零的时候，表示模乘运算完成。

4.2 模幂模块

模幂模块主要功能是控制模乘模块的循环运算。对模幂模块的可重构设计主要包括对模乘运算循环次数和密钥的控制。由 Montgomery 构造的从左到右二进制扫描位扫描法硬件结构图如图 5 所示。

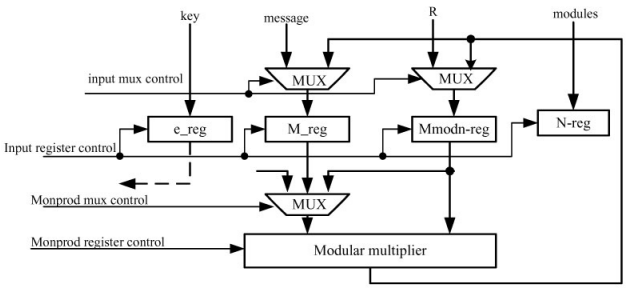

图 5 模幂模块结构

每个输入都有经过相应位数的寄存器。第一次模乘运算（预运算 $M2=Monprod(M,R,N)$）结束后，结果存在 M- reg 中，以备下次使用。第二次模乘运算（预运算 $C=Monprod(1,R,N)$））结束后，结果存在 rrmodn- reg 中。接下来每次循环模乘运算的结果都存入 rrmodn- reg 中。后处理结束后，rrmodn- reg 中就是模幂运算的结果。

e_reg 用来存储密钥。当 load 有效时，将外部密钥载入 e_reg 中。当 shift_en 有效时，e_reg 开始由低位向高位进行位移。具体结构图如图 6 所示。

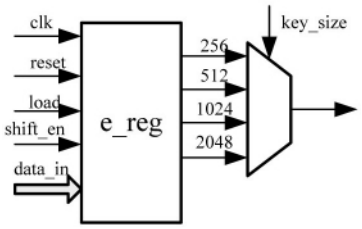

图 6 e_reg 结构图

e_reg 的输出有 4 位，分别为 e_reg 的第 256、512、1024、2048 位，这四位输入一个四选一选择器，由 key_seze 来选择哪位做为输出。

模幂运算控制模块主要是由计数器来实现。不同密钥长度的 RSA 算法的模幂循环次数不一样，所以计数器的初始载入值也不一样。初始值为密钥长度加上 3（2 次预处理和 1 次后处理模乘运算）。每执行完一次模乘运算，计数器就减一，e_reg 也进行一次移位，当计数器的值为零时，就表示所有模乘运算都完成，rrmodn- reg 中就是模幂运算的结果了。

对于 n 位的输入，这种设计要对输入密钥进行 n 次扫描，每次扫描需要做一次或者两次模乘，同时，在扫描之前要进行预处理，需要做两次模乘，扫描之后要进行后处理，需要做一次模乘，每做一次模乘需要 n+3+64 个时钟周期。所以，在最坏情况下，加密需要（2n+3）*（n+3+64）个时钟周期，当 n 为 2048 时，需要 8.67M 个时钟周期。

5 性能分析

两种设计方法可以实现采用固定密钥长度的 RSA 算法结构所设计的电路同时适配 256bit、512bit、1024bit、2048bit 四种不同密钥长度的应用。第一种方法是电路包含 256bit、512bit、1024bit、2048bit 四种不同密钥长度的固定密钥长度 RSA 密码算法结构；第二种方法是采用 2048bit 的固定密钥长度 RSA 算法结构。前一种方法浪费资源，后一种方法处理数据速度慢。

论文采用 FPGA 对可重构 RSA 算法结构进行了原型实现与验证，与采用类似结构实现的固定密钥 RSA 算法所占资源、最高时钟频率如表 2 所示。

表 2 各种 RSA 实现方法使用资源

RSA 算法类型	占用资源		F_{max}（MHz）
	逻辑单元	存储单元（bits）	
RSA（256bit）	3073	3710	362
RSA（512bit）	5890	7314	293
RSA（1024bit）	11597	12516	253
RSA（2048bit）	22748	24808	206
可重构 RSA	22897	24807	200

采用第一种设计方法所占资源为四种固定密钥长度 RSA 结构之和，即逻辑单元为 43308，存储单元为 48348；而可重构 RSA 使用了 22897 个逻辑单元和 24807 比特存储单元，节约了 47%的逻辑单元和 48%的存储单元。

采用可重构 RSA 与采用第二种设计方法（即使用 2048bit 固定密钥长度）分别应用于 256bit、

512bit、1024bit、2048bit 密钥长度系统时的数据吞吐量如表 3 所示，当应用于 256bit、512bit、1024bit 密钥长度应用时，采用可重构 RSA 结构性能更优。

表 3 可重构与固定密钥长度 RSA 数据吞吐率

算法类型	最高工作频率（MHz）	不同密钥长度最坏情况下数据吞吐率（kb/s）			
		256bit	512bit	1024bit	2048bit
2048bit 固定密钥长度 RSA	206	10.6	19.0	31.6	47.5
可重构 RSA	200	365.6	183.4	92.0	46.1

6 小结

本文提出并实现了一种使用较少硬件资源的能够适配 256bit，512bit，1024bit 和 2048bit 四种密钥长度的可重构 RSA 算法结构，适配不同密钥长度 RSA 算法应用，能够更好的满足不同层次安全性的信息系统的需要。

参考文献

[1] Rivest R , Shamir A , Adleman L. A method for obtaining digital signature and public- key cryptosystems [J].Communications of the ACM, 1978, 21（2）: 120- 126.

[2] Diffie W, Hellman M E. New directions in cryptograph [J]. IEEE Transactions on Information Theory , 1976 , 6（22）:644- 654.

[3] 曲英杰. 可重构密码协处理器的组成与结构.[J] 计算机工程与应用, 2003;23:32- 34

[4] R Reed Taylor. A high- performance flexible architecture for cryptography [A]. Seth Copen Goldstein Proceeding of the Workshop on Cryptographic Hardware and Embedded Systems [C]. London: Springer- Verlag Press, 1999. 231 - 245.

[5] Rainer Bcuchy. A programmable crypto processor architecture for high- bandwidth applications [D]. Germany: Technische University Mü nchen, 2002.

[6] T W Arnold, L P Van Doorn. The IBM PC IXCC: A new cryptographic coprocessor for the IBM Server [J]. I BM Journal of Research and Development, 2004, 48（3）: 475 - 487.

[7] A.Mazzeo, L.Romano, G. P. Saggese FPGA- based Implementation of a serial RSA processor. [C]. Design, Automation and Test in Europe Conference and Exhibition,2003 : 582- 587.

[8] Montgomery P L. Modular multiplication without trial division [J].Mathematics of computation, 1985, 44（170）: 519- 521

[9] 王超，沈海斌，孟庆.RSA 密码算法的硬件实现.[J] 计算机工程与应用，2004.14：127~128，147

[10] T- W Kwon , et al . Two implementation methods of a 1024 bit RSA cryptoprocesor based on modified Montgomery algorithm [A] . Proceed- ings of the 2001 IEEE International Symposium on Circuits and Systems（ISCAS 2001） [C].New York: IEEE press,2001,4 : 650- 653.

作者简介

伍彬山，厦门大学电子工程系在读硕士研究生，研究方向 集成电路设计与应用；

王云峰，讲师，厦门大学电子工程系硕士研究生导师；

刘智超，厦门大学电子工程系在读硕士研究生，研究方向 集成电路设计与应用；

刘天翔，厦门大学电子工程系在读硕士研究生，研究方向 集成电路设计与应用。