RSA 模乘器硬件优化设计*)

吴东锋 李 华 周 瑛
(重庆大学计算机学院 重庆 400044)

摘 要 在分析RSA算法的基础上,着重对核心的模乘运算进行了优化,并在FPGA上对改进后的模乘算法以及1024位的RSA密码算法进行了仿真。实验结果表明,优化效果较为理想。本文涉及RSA模乘器能够较好地满足现代电子政(商)务,变电站远程通讯等应用系统的实时性要求,具有良好的应用前景。

关键词 RSA,模乘运算,Montgomery

Optimization Design of RSA Modular Multiplication Hardware

WU Dong-Feng LI Hua ZHOU Ying
(Chongqing University College of Computer Science, Chongqing 400044)

Abstract Based on analysis about RSA algorithm, the paper focuses on the optimization of core modular multiplication operation and emulates improved modular multiplication algorithm and 1024-bit RSA cipher code algorithm in the FPGA platform. The experiment result indicates that optimization effect is relatively perfect. In the paper modular multiplication can satisfy modern electronic governmental affairs (electronic commerce) and remote communication of substation these application system's solid hour's need. It contains good application foreground.

Keywords RSA, Modular multiplication operation, Montgomery

1 引言

PKI(Public Key Infrastructure)是实现现代电子政(商)务、变电站远程通信等系统中对使用者的身份认证和信息的完整性、不可否认性认证的关键技术,而RSA是PKI体系中公钥加密和数字签名最常采用的加密算法。以RSA算法[1]为基础,来设计具有独立功能的硬件芯片的RSA密码计算构件,以其高速的性能和硬件本身特有的安全优势得到了广泛的关注与巨大的投入,国内外对此都做了大量的研究[2,3]。同时,国内外众多学者对RSA及其相关算法的实现研究也取得了不错的进展。

2 RSA公钥密码算法

由Rivest,Shamir和Adleman在1978年设计并公布的RSA[1]算法是最具有代表性的公开密钥密码体制,它是一个能同时用于加解密和数字签名的算法。也是研究得最深入的公钥算法。它从提出到现在的二十多年中,经历了各种攻击的考验,逐渐被人们所接受,普遍认为是目前最优秀的公钥方案之一。

RSA加/解密算法公式表述如下:

加密过程:$c=m^d \bmod n$

解密过程:$m=c^e \bmod n$

其中,m是明文,d是加密密钥(或公钥),e是解密密钥(或私钥),n是模数,c是密文,mod是取模函数。

RSA算法主要难点在于大数的模幂乘运算效率较低。所以,提高大数模幂乘的效率一直都是提高RSA算法速度的一个非常重要的课题。

3 模幂乘运算研究

3.1 长整数模幂乘运算

模幂乘运算$m^e \bmod n$不能先计算m^e然后再求模,这样m^e的结果会占用巨大的存储空间而无法实现必须对m^e的中间结果进行求模运算,使结果二进制位数始终保持在一定范围内。计算$m^e \bmod n$最简单的方法是重复进行$c=c*m \bmod n$模乘运算,直到求出$m^e \bmod n$为止,这种方法需要$e-1$次乘法运算。例如计算$m^{15} \bmod n$需要计算

$$m \to m^2 \to m^3 \to m^4 \to \cdots \to m^{15}$$

其需要14次乘法,而采用

$$m \to m^2 \to m^3 \to m^6 \to m^7 \to m^{14} \to m^{15}$$

其需则仅需六次乘法运算。这种平方——乘幂方法也称之为二进制法。在硬件实现上较为常用的方法是二进制位扫描法,分为从左到右和从右到左两种。在从左向右扫描法中,主要是面积可以得到优化,而在从右向左扫描法中,主要是速度得到了优化。

在从左到右扫描法中,指数e的位被从最高有效位扫描到最低位,对于每一位要作一次平方求模,如果e的位值是1,还需要做一次模乘。

从右到左扫描法则是最低有效位扫描到最高位,需要用中间变量保存中间值。

3.2 Montgomery模乘算法加速实现RSA

Montgomery算法是将部分积对任意的n取模转化为对基数r取模,简化了计算过程,提高了加解密的速度。这一有效的模约化方法是P. Montgomery1985年在文[4]中发表的,并在此后找到了广泛的实际应用。

原始的Montgomery算法如下:

吴东锋 硕士研究生,讲师,主要研究方向:网络安全;李 华 副教授,硕士生导师;周 瑛 博士生。

选择与(模数)n互素的基数r,为计算方便,它通常是机器字长的整数倍;并且选择r^{-1}及n',满足$0<r^{-1}<n$,$0<n'<r$,使得,$r\,r^{-1}-n\,n'=1$。

Montgomery乘积的计算按以下算法产生。

在$R(r,n)$中Montgomery乘积$\mathrm{mont}(a,b)$的计算

S1. 置$t=a*b$;

S2. 置$m=t*n' \bmod r$;

S3. 置$u=(t+m*n)/r$(商取整数);

S4. 若$u\geqslant n$,输出$u-n$,否则输出u。

4 Montgomery模乘算法在FPGA上的优化实现

4.1 改进的Montgomery算法

在对Montgomery算法的基本形式进行分析时[5],可以看出它有以下几个缺点:(1)所需存储空间较大(2s+1个存储单元);(2)带进位传输延迟,影响运算速度;(3)带有除法运算,需要另外有专门的运算部件。

我们可以采用下面描述的改进的Montgomery算法[6],把模乘转化为加法和移位的反复迭代过程,并都可以用相应的硬件结构来实现。

MonPro操作执行的伪代码在算法如下文所示,在这里:

$$A=\sum_{i=0}^{k-1}a_i 2^i \quad B=\sum_{i=0}^{k-1}b_i 2^i \quad M=\sum_{i=0}^{k-1}m_i 2^i$$

$a_i,b_i,m_i\in\{0,1\}$

模M是一个长度至多k-bit的整数$(0<M<2^k)$,并且$A,B<M$,所以乘法器的比特长度n被定义为等于$k+2$。使乘法器的比特长度等于$k+2$来确保中间结果S保持在范围内$(0=S=2^k)$是有必要的。这个条件允许中间结果被作为下个循环的输入来使用。

改进算法:MonPro (A,B,M)

```
MonPro (A,B,M)
{
  S-1=0;
  A=2×A;
  for i=0 to n do
  qi=(Si-1) Mod 2;        (Si-1的最低有效位(LSB))
  Si=(Si-1+qiM+biA)/2;
  end for
  Return Sn;
}
```

上面MonPro函数通过下面的形式计算Montgomery乘积:

$$\mathrm{MonPro}(A,B,M)=AB_r{}^{-1} \bmod M$$

因数r^{-1}必须在计算中得到,并且预计算和快速运算被要求产生正确的结果。在这里,r^{-1}是$r(\bmod M)$的倒数,也就是说$r^{-1}r=1(\bmod M)$,在这里r被给出如下:$r=2^n$。

4.2 Montgomery模乘器的优化

4.2.1 优化前的Montgomery模乘器

根据改进的Montgomery算法,下面我们来讨论相应的硬件结构。

采用从右至左的进位传送,将用到两个进位传送加法器,这两个加法器被直接映射到一个n位和一个$n+1$位的全加器。在实现b_iA和S_{i-1}的加法时,为了使运算结果并不由q_i决定,A能被往高位移了1比特,这样强制$S_{i-1}+b_iA$的最低有效位(LSB)始终保持为零,该结构完成一次计算需要$n+1$个时钟周期。

4.2.2 多路复用/加法Montgomery模乘单元

在图1中所示模乘器的基础上,我们如果把加法器重新排列,可得到如图2所示的结构。同样,A必须向高位移一位,所需时间为$n+1$个时钟周期。然而,这个结构与图1中所示结构相比,操作速度有明显提高。其原因在于,一旦完成两个最低有效位的加法,马上能执行b_i2A+q_iM的运算。这样,两个加法器能并行的工作。与图1中的模乘器相比,这种结构稍微要求更多的面积,因为在第二级加法器的输入需要$2n$位的寄存器。

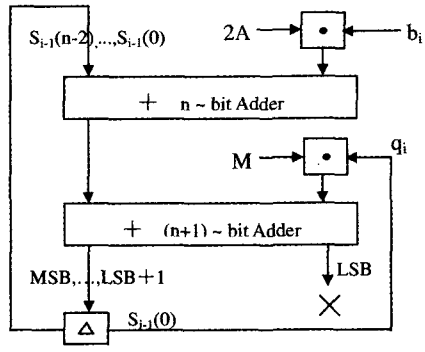

图1 优化前的Montgomery模乘器

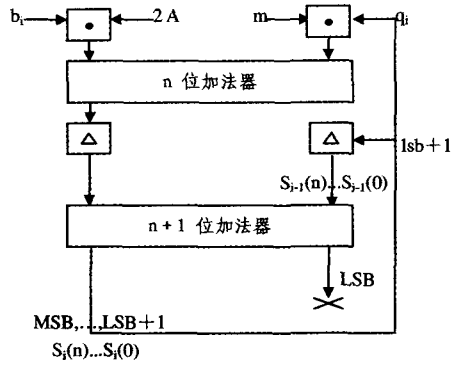

图2 改进后的Montgomery模乘单元

实际上,在图2中,在一个给定乘法每次反复的执行中,第一级加法器只有4种可能的输出。分别是:0,M,$2A$或$M+2A$,到底取何值由b_i和q_i决定。因此,2个加法器来执行太浪费了。可以改为,使用一个多路复用器来选择4种加法器可能的输入,如图3所示。

4.3 多路复用/加法模乘器的流水线体系结构

通常,RSA算法的安全性依赖于密钥的长度,长度越大,安全性越好。但是,对模乘器而言,如果其位数超过FPGA的1column时,就失去了逻辑进位的优势。这时,可以通过把计算划分成j个p-bit的字,用流水线来操作,这样就可以提高时钟速度(在这里p是进位串的最大长度,$j=n/p$)。

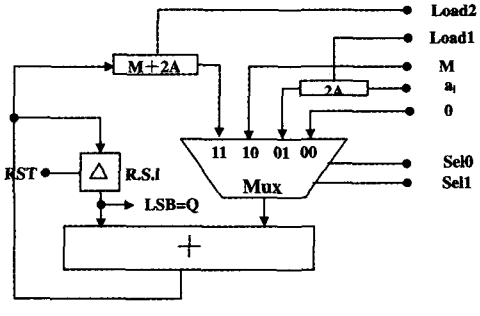

图3 多路复用/加法Montgomery模乘器

模乘器的输入必须是分成p-bit长的字,并且加法器的进

位必须在输入下一级的加法器之前延迟一个时钟。因为执行右移操作，和的最低有效位(LSB)必须直接的反馈到前一级加法器的最高有效位(MSB)。但是，这种操作并不会降低运算的速度，因为在需要计算下级加法器的 MSB 时，上级加法器的 LSB 已被预计算。图 4 所示是当 $j=4$ 的时候，通过一个完整流水线乘法器的数据流，图中每个矩形表示一个如图 3 所示单一的流水线模乘器单元。在装载第一个 p-bit 输入字 A_0 到产生第一个 p-bit 输出字 S_0，将花去 $n+3$ 个时钟周期。每次乘法运算将会花去 $n+3$ 个时钟周期，其中 2 个时钟周期用来初始化，另外 $n+1$ 个时钟周期用来执行模乘算法。在接着的 $j-1$ 个时钟周期里，输出字 S_1 至 S_{j-1} 是有效的，并且被装载时向右移，从而使得在 k 个时钟周期里连续的输出结果。

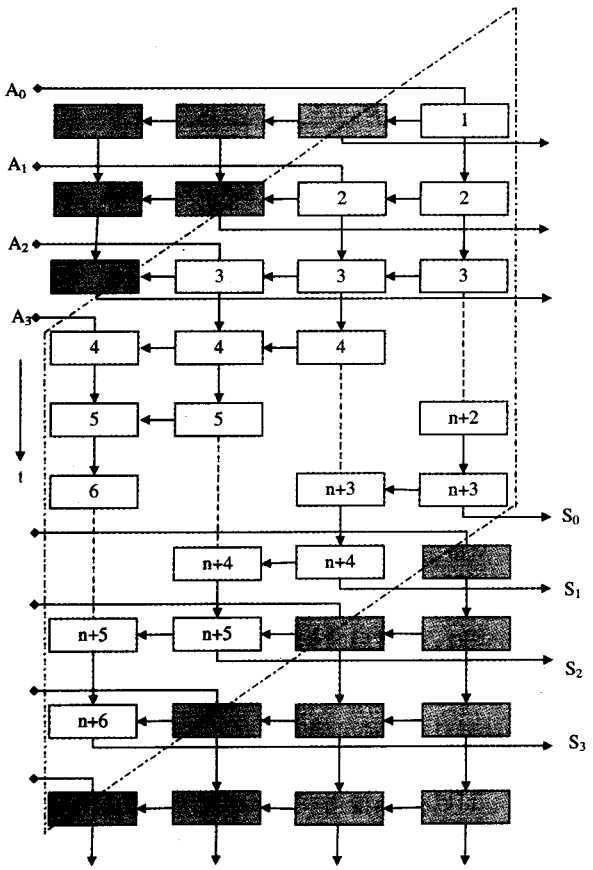

图 4　多路复用/加法模乘器的流水线体系结构

5　实验结果及分析

5.1　多路复用/加法模乘器仿真结果

本设计采用 Altera 公司的 EP20K300EQC240-1x 器件作为算法载体，采用 Synplify Pro 7.1 对系统的 VHDL 编程代码进行了编译综合，再利用 QuartusII2.1 进行了仿真。

表 1 是本文实现的模乘器硬件结构与文[7](文[7]中的算法载体的器件与本文相同)中的结果的比较。

面积-时间积指电路的结合效率，等于面积乘时间，表 1 中的数据为相对面积-时间积[8]，即把文[7]中的硬件结构的面积-时间积看成单位 1，本文的实现方式相对于文[7]的硬件结构的面积 -时间积的百分率。

从表 1 中，可以看出，本文中的模乘器的工作频率比文[7]中的低，但是，对于 512 位的模乘，本文实现的模乘器的面积-时间积比文[7]中的小 1.00－0.34/1.00＝66%；对于 1024 位的模乘，本文实现的模乘器的面积-时间积比文[7]中的硬件结构小 1.00－0.68/1.00＝32%。因此，本文实现的模乘器具有较高的性价比。

表 1 本文实现的模乘器与文[7]中的结果的比较

比较内容		文[7]	本文算法
512 位	面积利用率	41%	8%
	一次循环时间	13.464ns	23.547ns
	工作频率	74.27MHz	42.47MHz
	相对面积-时间积	1.00	0.34
1024 位	面积利用率	83%	17%
	一次循环时间	13.524ns	44.645ns
	工作频率	73.94MHz	22.40MHz
	相对面积-时间积	1.00	0.68

5.2　基于多路复用/加法模乘器的 RSA 密码算法的实时性分析

根据文[7]中的用模乘器工作频率估算 RSA 算法的运行时间的方法，采用本文的模乘器来实现 1024 位 RSA 算法需要的时间约为 12.4ms。

目前大部分信息安全系统对 RSA 算法实现的实时性具有较高的要求，其中，电力系统的广域保护是要求远程通信实时性的典型代表。国际大电网会议(CIGRE)将广域保护的功能及控制手段和目标进行了定义，广域保护(稳控)的动作时间范围在 100ms 到 100s 之间[9]。因此，本文设计的模乘器能较好地满足变电站远程通信系统的实时性要求，同时，也能满足其它系统的实时性要求。

结束语　本文对的 Montgomery 模乘算法的硬件实现进行了优化设计，并且基于此模乘器，用流水线的方式在 FPGA 上对 1024 位的 RSA 算法进行了实现。实验结果表明，这种模乘器结构具有较高的性价比。基于该硬件结构模乘器的 RSA 密码计算构件，能较好地满足电子商(政)务、变电站自动化远程通信及其他系统信息安全的实时性要求，具有良好的应用前景。

参 考 文 献

1　Rivest R L, Shamir A, Adleman L. A Method of obtaining Digital Signatures and Public Key Cryptosystems. Comm. Of ACM, 1978,21(2):120～126

2　孙宏，杨义先. RSA 算法在 TMS320C62x 中的高速实现. 计算机工程与应用，2003.11

3　Krishnamurthy A, Tang Y, Xu C, Wang Y. An Efficient Implementation of Multi-prime RSA on DSP Processor. ICASSP 2003. http://www.icme2003.com/Papers. 2004

4　Montgomery P L. Modular multiplication without trial division. Math. Computation, 1985,44:519～521

5　K K C, Acar T, Kaliski B S Jr. Analyzing And Comparing Montgomery Multiplication Algorithms IEEE Micro, 1996,16(3):26～33

6　Daly A, Marnane W. Efficient Architectures for implementing Montgomery Modular Multiplication and RSA Modular Exponentiation on Reconfigurable Logic. Copyright 2002 ACM, 2002.24～26

7　李涛，张盛兵，李瑛. RSA 高速模乘单元的设计[J]. 计算机工程与应用，2003,39(26):48～50

8　罗耀国，娄淑琴. 基于 FPGA 的字串行 FIR 滤波器的实现[J]. 北方交通大学学报，2003,27(6):48～51

9　蔡运清，汪磊，Morison K，等. 广域保护(稳控)技术的现状与展望[J]. 电网技术，2004,28(8): 20～25