[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 7/72 (2006.01)

[12] 发明专利说明书

专利号 ZL 200610020386.8

[45] 授权公告日 2008 年 11 月 19 日

[11] 授权公告号 CN 100435091C

[22] 申请日 2006.3.1

[21] 申请号 200610020386.8

[73] 专利权人 成都卫士通信息产业股份有限公司

地址 610041 四川省成都市高新区创业路 6 号

[72] 发明人 王金波

[74] 专利代理机构 成都九鼎天元知识产权代理有限公司

代理人 刘世权

[56] 参考文献

CN1547111A 2004.11.17

CN1172390A 1998.2.4

CN1492316A 2004.4.28

US5764554A 1998.6.9

快速模幂算法及其硬件实现. 周芬，高志强. 微电子学，第 30 卷第 6 期. 2000

审查员 李 琰

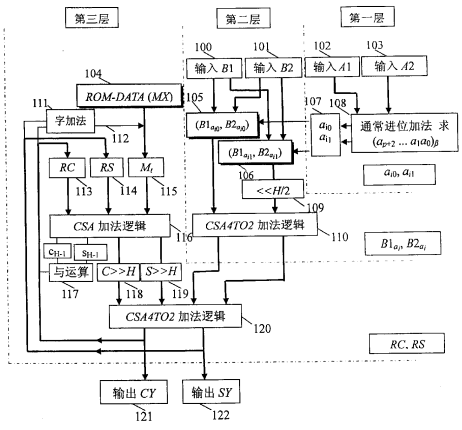
权利要求书 3 页 说明书 9 页 附图 6 页

[54] 发明名称

大数模幂系统的硬件高基实现方法

[57] 摘要

本发明公开一种大数模幂系统的硬件高基实现方法，涉及公钥密码系统中的模幂运算硬件实现方法，是为了解决现有技术中处理高基(2^H)数据的效率不高、实现频率较低、缺乏通用性的问题而提出，本发明将高基数据的模幂运算分为初始化处理单元、并行加法处理单元、模乘运算单元、模幂主体运算单元、数据输出恢复单元五个单元，通过构造动态并行加法并配用初始化存储数据表，采用简单逻辑实现模乘运算和模幂主体运算处理公钥密码系统中的高基数据，与现有技术相比，模幂主体运算仅使用或、异或、与等简单逻辑，实现频率高，相对二进制为基的数据处理方法，可提高硬件处理数据能力 H 倍，可应用于公钥密码系统的模幂运算硬件处理当中。



1、大数模幂系统的硬件高基实现方法，将数据输入可编程逻辑器件或ASIC 芯片进行模幂运算，其特征在于：所述的实现方法分为五个单元，分别是初始化处理单元、并行加法处理单元、模乘运算单元、模幂主体运算单元、数据输出恢复单元，其中，

a. 初始化处理单元：模幂运算的模数 N 按基 $\beta = 2^n$ 的展开式为 $N = (n_{p-1} \ldots n_1 n_0)_\beta$，p 为 N 的字长，各数字 $n_0, n_1, \ldots, n_{p-1}$ 依次按低位至高位排列，取

$$n' = \beta - n_0^{-1} \bmod \beta \,,\ R = 2^{p+2} \bmod N,\ R2 = R^2 \bmod N$$
$$M = n' \times N = (m_p \ldots m_1 m_0)_\beta,\ M_j = j \times M,\ j = 0, \ldots, \beta-1$$

在 ROM 中按二进制形式存储下面数据：

$n', N, R, R2, \{M_j, j = 0, \ldots, \beta-1\}$

b. 并行加法处理单元：对保存进位加法的进位输出进行 2 倍处理，满足 C + S = X + Y + Z + W，输入为(X, Y, Z, W)，输出为(C, S)的保存进位加法的公式为

$$(C, S) = \mathbf{CSA4TO2}(X, Y, Z, W) = \mathbf{CSA}(\mathbf{CSA}(X, Y, Z), W)$$

任给 $x = (x_{H/4-1} \ldots x_1 x_0)_{2^4} = 2^{4(H/4-1)} x_{H/4-1} + 2^{4(H/4-2)} x_{H/4-2} + \ldots + x_0$，$H>4$，$0 \le x_i < 2^4$，以及 X 和 Y，经$(\log_2 H-2)$层 **CSA4TO2** 运算以及简单移位处理，可得到$(X_x, Y_x) = x \times (X+Y)$；其中 CSA 表示对其后面括号内的参数进行保存进位加法处理，CSA4TO2 表示 4 个输入、2 个输出的保存进位加法处理；CSA（X, Y, Z）表示对 X、Y、Z 进行保存进位加法处理；CSA（CSA(X, Y, Z)，W）表示先对 X、Y、Z 进行保存进位加法处理，然后再将其结果与 W 进行保存进位加法处理。

c. 模乘运算单元：把 $MX = \{M_j, j = 0, \ldots, \beta-1\}$存储在 ROM 中，输入数据有(A1, A2), (B1, B2)以及循环次数 *l*，输出数据(CY, SY)，给定中间寄存器变量 RC 和 RS 按基 $\beta = 2^n$ 的展开式分别为 $RC = (rc_{p+1} \ldots rc_1 rc_0)_\beta$，$RS = (rs_{p+1} \ldots rs_1 rs_0)_\beta$，以及信号变量 C、S，模乘运算单元经历 *l* 次循环过程，循环运算被分成三个模块，

(1) 预计算模块，对 RC 和 RS 清零，同时根据输入数据(A1, A2)和(B1, B2)，计算 A1+ A2 的最低位数字 a_0 和 $(B1_{a0}, B2_{a0})$；

(2) 动态并行加法模块，在第 i 次循环时，$i = 0, 1, \ldots, l-1$，由 A1+ A2

按通常加法得到 $a_i = a_{i1}2^{H/2} + a_{i0}$，其中 $0 \le a_{i1}, a_{i0} < 2^{H/2}$，A1+ A2 = $(a_{p+2} \ldots a_1 a_0)_\beta$。利用 a_{i0} 和 a_{i1} 以及并行加法处理单元，经 $(B1_{a_{i0}}, B2_{a_{i0}})$ 和 $(B1_{a_{i1}}, B2_{a_{i1}})$，及下面计算得到$(B1_{a_i}, B2_{a_i})$：

$$(B1_{a_i}, B2_{a_i}) = \mathbf{CSA4TO2}(B1_{a_{i0}}, B2_{a_{i0}}, 2^{H/2}B1_{a_{i1}}, 2^{H/2}B2_{a_{i1}})$$

(3) 循环反馈运算模块，进行第 i+1 次循环时，计算 $t = rc_0 + rs_0$，利用第 i 次循环时的计算值 $B1_{a_i}$ 和 $B2_{a_i}$，以及 ROM 中的 M_t 值，可以利用 t 或利用 t_0 和 t_1 寻址 M_t 值，其中 $t = t_1 2^{H/2} + t_0$，作如下运算更新 RC 和 RS，

$(C, S) = \mathbf{CSA}(RC, RS, M_t)$，或$(C, S) = \mathbf{CSA4TO2}(RC, RS, M_{t1}{<<}H/2, M_{t0})$

$$(RC, RS) = \mathbf{CSA4TO2}(C{>>}H, S{>>}H, B1_{a_i}, B2_{a_i}) .$$

$rc_0 = rc_0 + (c_{H-1} \wedge s_{H-1})$，$c_{H-1}$, s_{H-1} 表示 C 和 S 的第 H-1 比特，其中“∧”为与操作，“>>H”表示将数据右移 H 位，“<<H/2”表示将数据右移 H/2 位；CSA 的含义与并行加法处理单元的相同；

d. 模幂主体运算单元：模幂运算的模数为 N，计算 $Y = X^E \bmod N$，输入指数 E 的二进制展开式为 $E = (e_{h-1}, e_{h-2}, \ldots, e_1, e_0)_2$，最高位 $e_{h-1} = 1$, $0 \le h < p \times H$，输入明文 X 按基 $\beta = 2^H$ 的展开式为 $X = (x_{p-1}, x_{p-2}, \ldots, x_1, x_0)_\beta < N$，设模幂主体运算输出值为 W，其运行过程分成三个阶段，

(1) 初始阶段，模幂主体运算含两个并列的模乘运算单元，单元-1 和单元-2，分别对应两组输出变量 CZ 和 SZ，以及 CP 和 SP；

(2) 循环阶段，完成初始阶段后进入 h 次循环运算过程，每次循环时，并行同步运行单元-1 和单元-2，取 $l = p+3$，设单元-1 输出数据为 CZ 和 SZ，单元-2 的输出数据为 CP 和 SP，则单元-1 下次运算的输入数据为 CZ、SZ、CZ、SZ，单元-2 下次运算的输入数据为 CZ、SZ、CP、SP；

(3) 末尾阶段，完成循环阶段后取 CZ = 0、SZ = 1，取 $l = p+2$，运行单元-2，输出 CP 和 SP；

e. 数据输出恢复单元：计算 W = CP +SP，然后利用 $W = (w_{p-1}, w_{p-2}, \ldots, w_1, w_0)_\beta$ 以及 $q = n' \times w_0 \bmod \beta$，得到 $Y = (W + q \times N)/\beta = X^E \bmod N$。

2、如权利要求 1 所述的大数模幂系统的硬件高基实现方法，其特征在于：所述的模幂主体运算单元三个阶段中，

a. 初始阶段中，取 $l = p+3$，用 R2、0、X、0 作为输入，运行单元-1，其输出 CZ 和 SZ 反馈至单元-1 和单元-2 的输入单元，用 R 和 0 对 CP 和

SP 分别赋初值；

b. 循环阶段中，完成初始阶段后进入 h 次循环运算过程，每次循环时，并行同步运行单元-1 和单元-2，取 $l = p+3$，在第 i 次循环中，$i = 0, ..., h-1$，当 $e_i = 0$ 时，CP 和 SP 中的数据不变，不被单元-2 的输出数据更新。

3、如权利要求 1 或 2 所述的大数模幂系统的硬件高基实现方法，其特征在于：所述的数据输出恢复单元中，Y>N，输出 Y；否则若 Y≤N，输出 Y=Y-N。

大数模幂系统的硬件高基实现方法

技术领域

本发明涉及公开密钥密码系统中的模幂运算硬件实现方法，特别是涉及在大型模幂运算中为提高数据处理效率，通过构造动态并行加法并配用初始化存储数据表，采用简单逻辑实现模乘运算和模幂主体运算的高基(2^H进制)实现方法。

背景技术

为了提高公钥密码系统的运行效率，模乘运算以及模幂运算的效率是关键。传统的除余数法和累加法实现大数模的运算效率不理想，在各种模乘算法中，Montgomery 乘法是计算模乘最有效的算法之一，基本思想是由系列加法和移位来实现通常的除法操作，Montgomery 乘法已成为公钥密码系统中的基本运算单元。

用硬件实现两个或者两个以上的整数加法时，可按位并行方式进行，输出两个数据，一个含各位的进位信息 C，另一个含各位的异或信息 S。这种进位存储加法器(Carry Save Adders，以下简记为 CSA)可实现免链接保存进位加法，为公知技术。记'⊕'表示按位'异或'操作，'∧'表示按位'与'操作，'∨'表示按位'或'操作，':=' 表示把右边数据或运算值赋值给左边。对三个整数 X, Y, Z 进行 CSA 加法操作，输出为 C 和 S，满足 $2C+S=X+Y+Z$，则 CSA 计算公式为：

$$C := (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z), \quad S := X \oplus Y \oplus Z.$$

可见，CSA 对任意位整数的加法操作都可以在一个节拍内并行完成，但 CSA 并没有完成一次完整的加法操作。因此，对于通常的加法操作，CSA 并不适合，而对于需进行许多次循环的加法操作，CSA 却能高效地完成。

用 FPGA 或 CPLD 等可编程逻辑器件或者 ASIC 芯片来实现模幂，或者用专用硬件部件实现模幂运算并通过 IP 核(Intellectual Property)接口供上层调用来加速公钥密码运算，是一种非常流行的做法。目前，模幂运算中的大数模乘硬件实现方法基本上分为两种：一是以并行加法 CSA 和模 2 除法操作方式实现 Montgomery 模乘，二是以阵列结构(Systolic Arrays)处理高基数据实现 Montgomery 模乘。设 k 为模比特长度，d 为 RSA 系统中私钥指数比特长度。用第一种方法完成一次 Montgomery 模乘只需 k+2 个时针周期，RSA 签名运行

时间为$(k/2+2)(d/2+3)$个时钟周期。取 $k, d = 1024$，用这种方法实现 RSA 系统，最小时钟周期可达 9.5ns（器件 XC2V1500-8，模乘占用 8 万门）。第二种方法充分利用一些器件具有的快速进位加法链结构，构造 m 个联接运算单元以避免太长的进位链，按此作业方式处理高基($2^{k/m}$ 进制)数据，Montgomery 模乘需$(2m+3)$个时针周期，RSA 签名的运行时间为$(m+20)(d/2+2)$个时钟周期。取 $m = 128$，以 2^4 为基，用这种方法实现 RSA 系统，最小时钟周期达 20.7ns（器件 XC40150XV-8，模乘占用 3413 CLBs）。

第一种方法以简单逻辑和并行加法设计，具有时针周期小，便于移植等优点，但没有突破以 2 为基的二进制数据处理模式，限制了数据处理的效率。第二种方法能处理高基数据并采用流水线作业方式，考虑到器件固有的延迟性，使得矩阵列设置不能太大，其实现频率与具体器件的特性密切相关，设计缺乏移植性。概括起来，第一种方法设计简单，可达到较高的实现频率，第二种方法设计较复杂，其实现频率也较低。但前者仅能处理二进制基数据，后者能处理高基数据，它们的实现速度相差并不大。

上述分析表明，利用上面两类方法进行模幂实现硬件设计，都不是最优的设计方法。并行加法和模 2 除法设计方法，不能处理高基(2^H 进制)数据的情形，限制了数据处理的效率；高基阵列设计方法的实现频率较低，实现效率与具体器件的特性密切相关，缺乏移植性和通用性。

发明内容

本发明的目的是为了解决现有模幂运算存在的效率不够理想、实现频率较低、缺乏通用性的问题，提供一种采用简单逻辑、有效提高数据处理效率，方便各类硬件芯片高速实现模幂系统的大数模幂系统的硬件高基实现方法。

本发明的目的是通过下述技术方案来实现的：

大数模幂系统的硬件高基实现方法，将数据输入可编程逻辑器件或 ASIC 芯片进行模幂运算，其特征在于：所述的实现方法分为五个单元，分别是初始化处理单元、并行加法处理单元、模乘运算单元、模幂主体运算单元、数据输出恢复单元，其中，

a. 初始化处理单元：模幂运算的模数 N 按基 $\beta = 2^H$ 的展开式为 $N = (n_{p-1} \ldots n_1 n_0)_\beta$，p 为 N 的字长，其中各数字 $n_i(i= 0,..., p-1)$按低位至高位排列，取

$$n' = \beta - n_0^{-1} \bmod \beta\ ,\ R = 2^{p+2} \bmod N,\ R2 = R^2 \bmod N$$

$$M = n' \times N = (m_p \ldots m_1 m_0)_\beta,\ M_j = j \times M\ (j = 0, ..., \beta - 1).$$

在 ROM 中按二进制形式存储下面数据：

n', N, R, R2, $\{M_j, j=0, ..., \beta-1\}$

b. 模乘运算单元：把 $MX = \{M_j, j = 0, ..., \beta-1\}$存储在 ROM 中，输入数据有(A1, A2), (B1, B2)以及循环次数 l，输出数据(CY, SY)，模乘运算单元经历 l ($\leq p+3$) 次循环过程，循环运算过程被分成三个模块，

(1) 预计算模块

(2) 动态并行加法模块

(3) 循环反馈运算模块

c. 模幂主体运算单元：模幂运算的模数为 N，计算 $Y = X^E \bmod N$，输入指数 $E = (e_{h-1}, e_{h-2}, ..., e_1, e_0)_2$，最高位 $e_{h-1} = 1 (0 \leq h < p \times H)$，输入明文 $X = (x_{p-1}, x_{p-2}, ..., x_1, x_0)_\beta < N$，设模幂主体运算输出值为 W，其运行过程分成三个阶段，

(1) 初始阶段，模幂主体运算含两个并列的模乘运算单元，单元-1 和单元-2，分别对应两组输出变量 CZ 和 SZ, 以及 CP 和 SP；

(2) 循环阶段，完成初始阶段后进入 h 次循环运算过程，每次循环时，并行同步运行单元-1 和单元-2，取 $l = p+3$，设单元-1 输出数据为 CZ 和 SZ，单元-2 的输出数据为 CP 和 SP，则单元-1 下次运算的输入数据为 CZ、SZ、CZ、SZ，单元-2 下次运算的输入数据为 CZ、SZ、CP、SP；

(3) 末尾阶段，完成循环阶段后取 CZ =0、SZ = 1，取 $l = p+2$，运行单元-2，输出 CP 和 SP；

d. 数据输出恢复单元：计算 W = CP +SP，然后利用 $W = (w_{p-1}, w_{p-2}, ..., w_1, w_0)_\beta$ 以及 $q = n' \times w_0 \bmod \beta$，计算 $Y=(W+q \times N)/\beta$，得到 $Y = X^E \bmod N$。

所述的并行加法处理单元按如下方法处理：

对并行加法的进位输出数据进行 2 倍处理，满足 C + S = X + Y + Z + W，输入为(X, Y, Z, W)，输出为(C, S)的并行加法公式为

(C, S) = **CSA4TO2**(X, Y, Z, W) = **CSA**(**CSA**(X, Y, Z), W)

给定数据向量(X, Y)，设$(X_j, Y_j) = j \times (X+Y)$，$j(j<2^4)$为奇数时，有

$(X_3, Y_3) = \boldsymbol{CSA4TO2}(X, Y, 2X, 2Y)$,

$(X_5, Y_5) = \boldsymbol{CSA4TO2}(X, Y, 4X, 4Y)$,

$(X_7, Y_7) = \boldsymbol{CSA4TO2}(X_3, Y_3, 4X, 4Y)$,

$(X_9, Y_9) = \boldsymbol{CSA4TO2}(X_5, Y_5, 4X, 4Y)$,

$(X_{11}, Y_{11}) = \boldsymbol{CSA4TO2}(X_3, Y_3, 8X, 8Y)$,

$(X_{13}, Y_{13}) = \boldsymbol{CSA4TO2}(X_5, Y_5, 8X, 8Y)$,

$(X_{15}, Y_{15}) = \boldsymbol{CSA4TO2}(X_5, Y_5, 8X, 8Y)$,

可见，任给 $j(0\leq j<2^4)$和数据向量(X, Y)，至多经 2 层 ***CSA4TO2*** 计算以及简单移位处理就得到(X_j, Y_j)。于是，任意给出 $x = (x_{H/4-1} \ldots x_1 x_0)_{2^4} = 2^{4(H/4-1)}x_{H/4-1} + 2^{4(H/4-2)}x_{H/4-2} + \ldots + x_0$（$H>4$，$0 \leq x_i < 2^4$）以及 X 和 Y，再经$(\log_2 H-2)$层（共 $2^0 + 2^1 + \ldots + H/8$ 个）**CSA4TO2** 运算以及简单移位处理，可得到$(X_x, Y_x) = x\times(X+Y)$。其中 CSA 表示对其后面括号内的参数进行保存进位加法处理，CSA4TO2 表示 4 个输入、2 个输出的保存进位加法处理；CSA（X, Y, Z）表示对 X、Y、Z 进行保存进位加法处理；CSA（CSA(X, Y, Z)，W）表示先对 X、Y、Z 进行保存进位加法处理，然后再将其结果与 W 进行保存进位加法处理。

所述的模乘运算单元给定中间寄存器变量 $RC=(rc_{p+1} \ldots rc_1 rc_0)_{\beta}$, $RS= (rs_{p+1} \ldots rs_1 rs_0)_{\beta}$，以及信号变量 C、S，循环运算过程三个模块具体为，

a. 预计算模块，对 RC 和 RS 清零，同时计算 a_0 和 $(B1_{a0}, B2_{a0})$；

b. 动态并行加法模块，在第 i 次循环时$(i = 0, 1, \ldots, l-1)$，由 A1+ A2 按通常加法得到 $a_i = a_{i1}2^{H/2} + a_{i0}$（$0 \leq a_{i0}, a_{i1} < 2^{H/2}$），其中 $A1+ A2 = (a_{p+2} \ldots a_1 a_0)_{\beta}$。若设置的 $H \leq 4$，则根据上面(X_x, Y_x)处理直接计算出$(B1_{a_i}, B2_{a_i})$。若 $H > 4$，根据 a_{i0} 和 a_{i1}，利用并行加法处理模块，经 $(B1_{a_{i0}}, B2_{a_{i0}})$ 和 $(B1_{a_{i1}}, B2_{a_{i1}})$，及下面计算得到$(B1_{a_i}, B2_{a_i})$：

$(B1_{a_i}, B2_{a_i}) = \mathbf{CSA4TO2}(B1_{a_{i0}}, B2_{a_{i0}}, 2^{H/2}B1_{a_{i1}}, 2^{H/2}B2_{a_{i1}})$

c. 循环反馈运算模块，进行第 i+1 次循环时$(i = 0, 1, \ldots, l-1)$，计算 $t= rc_0 + rs_0$，利用第 i 次循环时的计算值 $B1_{a_i}$ 和 $B2_{a_i}$，以及 ROM 中的 M_t 值(利用 t 寻址，或 t_0, t_1 寻址，$t = t_1 2^{H/2} + t_0$) 作如下运算更新 RC 和 RS，

$(C, S) = \mathbf{CSA}(RC, RS, M_t)$（或 $\mathbf{CSA4TO2}(RC, RS, M_{t_1}{<<}H/2, M_{t_0})$）

$(RC, RS) = \mathbf{CSA4TO2}(C{>>}H, S{>>}H, B1_{a_i}, B2_{a_i})$

$rc_0 = rc_0 + (c_{H-1} \wedge s_{H-1})$

其中，c_{H-1}, s_{H-1} 表示 C 和 S 的第 H-1 比特，>>H 表示数据向右移 H 位，<<H/2 表示数据向左移 H/2 位，其中的 CSA 的含义与并行加法处理单元的含义相同。

所述的模幂主体运算单元三个阶段中，

a. 初始阶段中，取 $l = p+3$，用 R2、0、X、0 作为输入，运行单元-1，其

输出 CZ 和 SZ 反馈至单元-1 和单元-2 的输入单元，用 R 和 0 对 CP 和 SP 分别赋初值；

b. 循环阶段中，完成初始阶段后进入 h 次循环运算过程，每次循环时，并行同步运行单元-1 和单元-2，取 $l = p+3$，在第 i(i = 0, ..., h-1)次循环中，当 $e_i = 0$ 时，CP 和 SP 中的数据不变（不被单元-2 的输出数据更新）

所述的数据输出恢复单元中，Y>N，输出 Y；否则若 Y≤N，输出 Y=Y-N。

本发明的有益效果是，采用简单逻辑实现高基(2^H进制)大数模乘运算和模幂主体运算，数据运算以高基(2^H进制)形式进行，模幂主体运算仅使用或、异或、与等简单逻辑，实现频率高，实现方法与具体器件的特性无关，可移植性强。在整个模幂运算过程中，由于数据输出恢复单元的运算量极小，可以用软件形式来完成。

利用本发明实现的模幂系统，能够获得更高的数据处理能力和更快的系统响应速度，具体而言，本发明的优点主要有：

(1) 以 2^H(H>1)进制为基进行 Montgomery 乘法，相对于以二进制为基进行 Montgomery 乘法，使得硬件数据处理能力具有成倍(近 H 倍)的提高。

(2) 本发明中，设计的模乘运算单元仅用到或、异或、与等简单逻辑，避免了乘法和减法等复杂计算，便于各种硬件实现，并有利于改善时钟频率。

(3) 本发明中，设计了有效的数据输出恢复单元，使得模幂主体运算单元成为计算主体，由此主体外运算可结合软件实现，进一步降低硬件规模和实施难度。

(4) 本发明中，设计方案与具体器件特性无关，体现出良好的移植性，适合在 ASIC、CPLD、FPGA 等各种硬件平台上实现。

(5) 本发明模乘运算单元部分，一个普通 FPGA 芯片实现 512 比特位长模乘（共花费 70 个时钟周期）很容易获得了 120MHZ 以上的时钟频率。与公开文献上其它方法给出的 FPGA 实现比较，其实现速率具有显著优势。

本发明适用于对速度具有严格要求的 RSA、DSA 等公钥密码系统(如签名速度要求每秒上千次以上)，以及其它应用系统中的大数模幂(或模乘)运算部件的硬件开发。

附图说明

图 1 为本发明模乘运算单元框图；

图 2 为本发明并行加法处理模块示例框图；

图 3 为本发明模幂主体运算单元框图；

图 4 为本发明数据输出恢复单元框图；

图 5 为本发明模幂运算整体结构框图；

图 6 为本发明模乘运算流水线操作示例图。

图 1 中标记：100～103 为模乘运算的 4 个输入数据；121～122 为模乘运算的 2 个输出数据；104 为初始化处理后的 ROM 数据；105 和 106 为并行加法处理模块，得到标识结果；108 表示通常 2^H 进制加法器，107 为其第 i 次输出结果（两个半字）；109 为数据左移 H/2 位；110 和 120 为 4 个输入数据的并行 CSA 加法；111 为 H 位的字加法；112 表示利用 111 输出数据寻址 ROM，得到 115 数据；113～115 为积存器单元；116 为 3 个输入数据的并行 CSA 加法；117 为单比特与运算；118～119 为左移 H 位运算。

图 2 中 $x<2^4$，各标记含义：200～202 为 3 个输入数据；203～204 为数据左移 n 位，205～206 为数据左移 s 位，211 为数据左移 $2k$ 位，212 为数据左移 $2g$ 位，由 202 确定 n, s, k, g 等数据；209～210 为可能的输出数据，207～208 为 4 个输入数据的并行 CSA 加法；213 为输出数据向量。

图 3 中标记：301～306 为 6 个输入数据；307 为 306 数据的第 i 比特值信号；316～317 为 2 个输出数据；308～309 为 2 个并列的模乘运算单元；310～313 分别为 CZ、SZ、CP、SP 积存器单元；314 为由 307 控制的数据选通装置；315 为控制信号产生装置，其中 l 为 2 个模乘运算单元的内循环计数器，h 为模幂主体运算外循环计数器，clk 为时钟控制信号。

图 4 中标记：400～403 为 4 个输入数据；413 为输出数据；404 为通常 2^H 进制加法器；405 为 404 输出数据的最低 H 位数据；406 为 H 位数据乘法；407 和 408 分别为 404 和 406 的输出；409 为通常 2^H 进制加乘和右移运算；410 为 409 的输出；411 对 Y 和 N 进行大小判断；412 为通常 2^H 进制减法操作。

图 5 中标记：500 为模幂的 2 个输入数据；505 为模幂输出数据；501 为模幂初始化处理单元，输出数据存储于 ROM；502 为模幂主体运算单元；503 为并行加法处理模块(采取预存储模式时)或其它单元需要的 RAM；504 为数据输出恢复单元。

具体实施方式

下面结合附图和具体实施例对本发明作进一步的说明。

大数模乘模块是目前使用的各类公开密钥密码系统的核心单元，是模幂运

算的循环体。各类公开密钥密码系统的模幂运算因模长度以及幂指数长度的不同，其大数模乘的规模和循环次数也不同。比如，系统的模幂模块(基本确定系统实现时间)共需 h 次大数模乘循环，每次大数模乘需花费 l 个时针周期，加上为调用模乘模块花费 d 个时针周期，则模幂模块共花费共 $h(l+d)$ 个时针周期，d 值一般在 0 到 2 之间，因此大数模乘运算消耗基本上确定了模幂运算效率。

下面以取基 2^8(H = 8)为例，阐述本发明方案中模乘运算单元所需时钟耗费与运算效率关系。

本发明利用硬件进行模幂高基运算的实现方法，如图 5 所示，包括初始化处理单元 501、并行加法处理单元、模乘运算单元、模幂主体运算单元 502、数据输出恢复单元 504 共五个单元，其中，模幂主体运算单元内含有两个并行模乘运算单元，模乘运算单元内包含并行加法处理单元。模幂运算输入 500 中将 X、E 输入，经过这五个单元的处理，最后输出 $Y=X^E \bmod N$。

首先进行初始化处理单元，在 ROM 中按二进制形式存储有数据{M_j, j=0, ..., β−1}。若 ROM 存储空间较小，仅计算并储存 $M_j = j \times M, j = 0, ..., 2^4-1$。因为，任给 $x = x_1 2^4 + x_0$（$0 \le x_0, x_1 < 2^4$）以及 M，得到 $x \times M = (2^4 M_{x_1}, M_{x_0})$。在模幂主体运算单元的模乘运算中，可用两个输入数据$(2^4 M_{t_1}, M_{t_0})$代替 M_t。

完成初始化处理单元后进入模幂主体运算单元，如图 3 所示，它由反复调用模乘运算单元来完成。下面依图 1 描述模乘运算单元。

本发明中模乘运算单元如图 1 所示，循环运算过程被分成预计算模块、动态并行加法模块、循环反馈运算模块共三个模块。

模乘运算单元中并行加法处理模块，如图 2 所示，任给 $j(0 \le j < 2^4)$和数据向量(X, Y)，至多经 2 层 ***CSA4TO2*** 计算以及简单移位处理就得到(X_j, Y_j)。于是，任给 $x = (x_1 x_0)_{24} = 2^{4(2-1)} x_1 + 2^0 x_0$ （$0 \le x_i < 2^4$）以及 X 和 Y，经 ***CSA4TO2*** 运算以及简单移位处理（即图 1 中 105-106 模块），可得到$(X_x, Y_x) = x \times (X+Y)$，对应图 1 中 110 的输出。

模乘运算单元中通常加法 A1+ A2 = $(a_0, a_1, ..., a_{p+2})$可在 p+2 个时钟内完成，在各时钟内依次输出各数字值。预计算模块对 *RC* 和 *RS* 清零，同时计算 a_0 和 $(B1_{a0}, B2_{a0})$。完成此预计算后进入后续两个模块的循环运算过程。如图 1 所示，$(B1_{a_{i0}}, B2_{a_{i0}})$和 $(B1_{a_{i1}}, B2_{a_{i1}})$的计算可依次在前后两个时钟内完成。如图

6 所示，在第一个时钟周期内计算 a_0 值，在第二个时钟周期内并行计算 a_1 值以及($B1_{a00}$, $B2_{a00}$)值，在第三个时钟周期内并行计算 a_2 值以及($B1_{a10}$, $B2_{a10}$)和($B1_{a01}$, $B2_{a01}$)值，在第四个时钟周期内并行计算 a_3 值、($B1_{a20}$, $B2_{a20}$)、($B1_{a11}$, $B2_{a11}$)以及(*RC*, *RS*)值，此后已建立起四级流水线作业。在第 *i* 个时钟周期内并行计算 a_i 值、($B1_{a_{i-1,0}}$, $B2_{a_{i-1,0}}$)、($B1_{a_{i-2,1}}$, $B2_{a_{i-2,1}}$)以及(*RC*, *RS*)值，其中，$a_i = a_{i1}2^{H/2} + a_{i0}$（$0 \le a_{i0}, a_{i1} < 2^{H/2}$）。在已取得 *t* 值、$B1_{a_i}$ 和 $B2_{a_i}$ 值后可在一个时钟内完成更新 *RC* 和 *RS* 的计算：

$(C, S) = \boldsymbol{CSA}(RC, RS, M_t)$（或 $\boldsymbol{CSA4TO2}(RC, RS, M_{t_1}\text{<<H/2}, M_{t_0})$），

$(RC, RS) = \boldsymbol{CSA4TO2}(C\text{>>H}, S\text{>>H}, B1_{a_i}, B2_{a_i})$,

$rc_0 = rc_0 + (c_{H-1} \wedge s_{H-1})$.

上面 rc_0 值第 0 比特总为 0，于是计算 $rc_0 + (c_{H-1} \wedge s_{H-1})$只需将 rc_0 的第 0 比特用$(c_{H-1} \wedge s_{H-1})$值代替。

模乘运算单元须进行 $l = p+3$ (模幂主体末尾阶段的模乘运算中 $l = p+2$) 次循环计算。模乘运算循环过程按时间顺序设计成三个层次：第一层计算 a_i，第二层计算($B1_{a_i}$, $B2_{a_i}$)，第三层计算(*RC*, *RS*)，于是，一次模乘运算需 2~3 个时钟周期完成预计算模块，以及 *l* 个时钟周期完成其它计算。设模数 N 位长为 $k = 512$ 比特，若取 $H = 8$，则按基 $\beta = 2^8$ 的展开式 $N = (n_{p-1} \ldots n_1 n_0)_\beta$ 中，$p = 64$。于是，$l = 67$（或 66）。如图 6 所示，将模乘运算单元建立起 4 级流水线作业，由此完成一次模乘运算花费共$(l + 3) = 70$ 个时钟周期。若采用通常的二进制基方法，完成一次模乘运算花费共 $k + 2 = 514$ 个时钟周期（时钟周期 9.5ns, 占用 8 万门）；若采用通常的阵列结构方法，构造 *m* 个联接运算单元（快速进位加法链结构，避免太长的进位链），以 2^4 为基，取 $m = 128$，完成一次模乘运算花费共$(2m+3) = 259$ 个时钟周期（时钟周期 20.7ns, 占用 3413CLBs）。

完成模幂主体运算单元后进入数据输出恢复单元，如图 4 所示，输入 CP, SP, N, n'，其中 CP 和 SP 分别为图 3 中 316 和 317 数据。数据输出恢复单元完成后得到 $Y = X^E \bmod N$。

在实际硬件开发中，可进行多项时钟设置，即充分利用模乘运算单元可能的最大时钟频率，提高模乘运算单元处理速度。模幂循环体单元中两个并行模乘运算单元也可改变为含模幂指数窗口处理的单个模乘运算单元设置，可减少近一半的硬件开销，但增加了模乘运算的次数。

按本发明提供的模乘方案在 FPGA（Stratix-epls10f780c6 芯片）上实现

（H = 8），测试结果显示，硬件开销不到 10 万门，最高频率达 126Mhz（时钟周期 8ns），其速度约为通常实现速度的 8 倍。

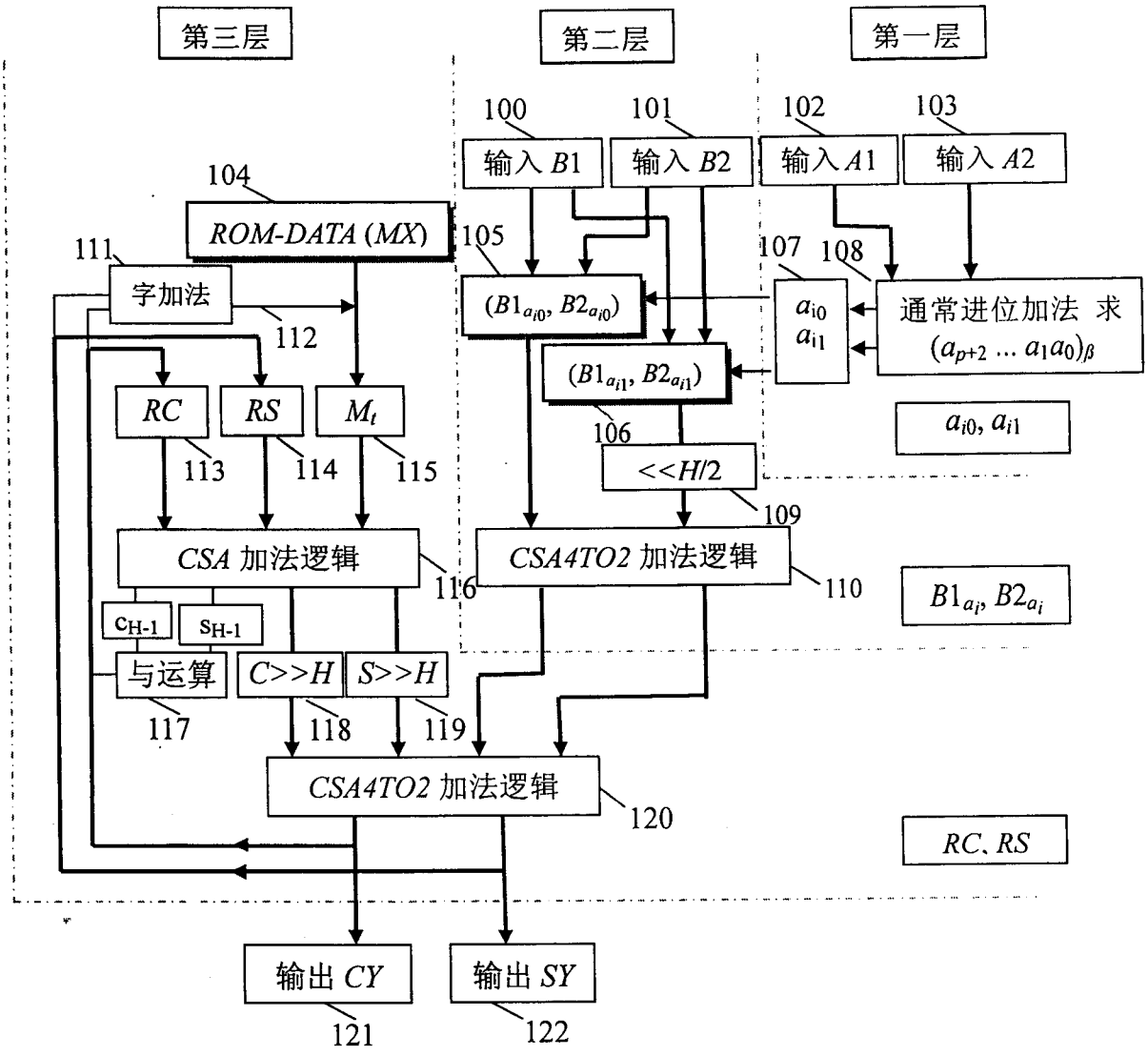

第三层
第二层
第一层
100
101
102
103
输入 B1
输入 B2
输入A1
输入 A2
104
ROM-DATA (MX)
105
111
字加法
107
108
$(B1_{a_{i0}}, B2_{a_{i0}})$
a_{i0}
a_{i1}
通常进位加法 求
$(a_{p+2} \ldots a_1 a_0)_\beta$
112
$(B1_{a_{i1}}, B2_{a_{i1}})$
RC
RS
M_t
106
a_{i0}, a_{i1}
113
114
115
<<H/2
109
CSA 加法逻辑
116
CSA4TO2 加法逻辑
110
$B1_{a_i}, B2_{a_i}$
c_{H-1}
s_{H-1}
与运算
C>>H
S>>H
117
118
119
CSA4TO2 加法逻辑
120
RC、RS
输出 CY
输出 SY
121
122

图 1

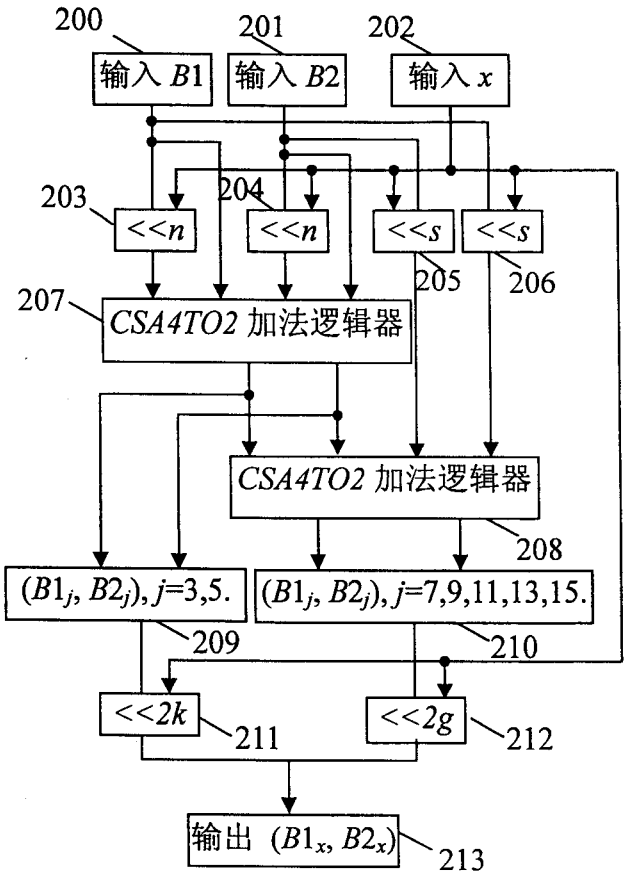

200
201
202
输入 B1
输入 B2
输入 x
203
204
<<n
<<n
<<s
<<s
205
206
207
CSA4TO2 加法逻辑器
CSA4TO2 加法逻辑器
208
(B1j, B2j), j=3,5.
(B1j, B2j), j=7,9,11,13,15.
209
210
<<2k
<<2g
211
212
输出 (B1x, B2x)
213

图 2

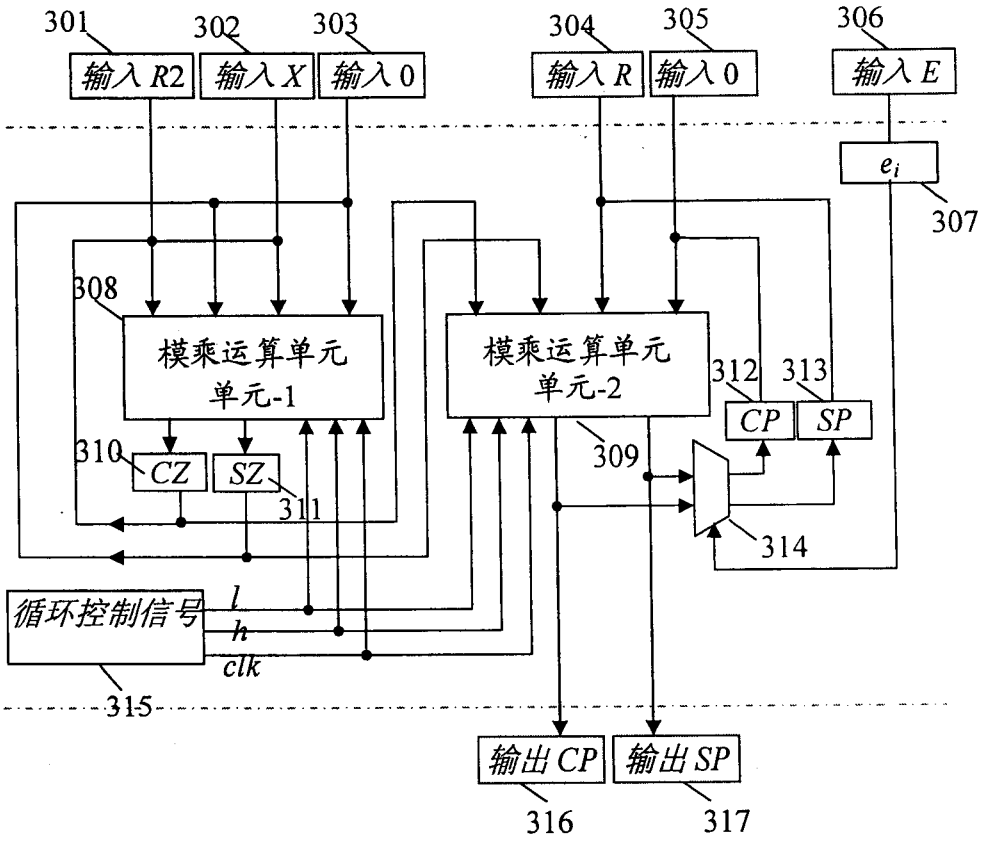

301
302
303
304
305
306
输入 R2
输入 X
输入 0
输入 R
输入 0
输入 E
e_i
307
308
模乘运算单元
单元-1
模乘运算单元
单元-2
312
313
CP
SP
310
CZ
SZ
311
309
314
循环控制信号
l
h
clk
315
输出 CP
输出 SP
316
317

图 3

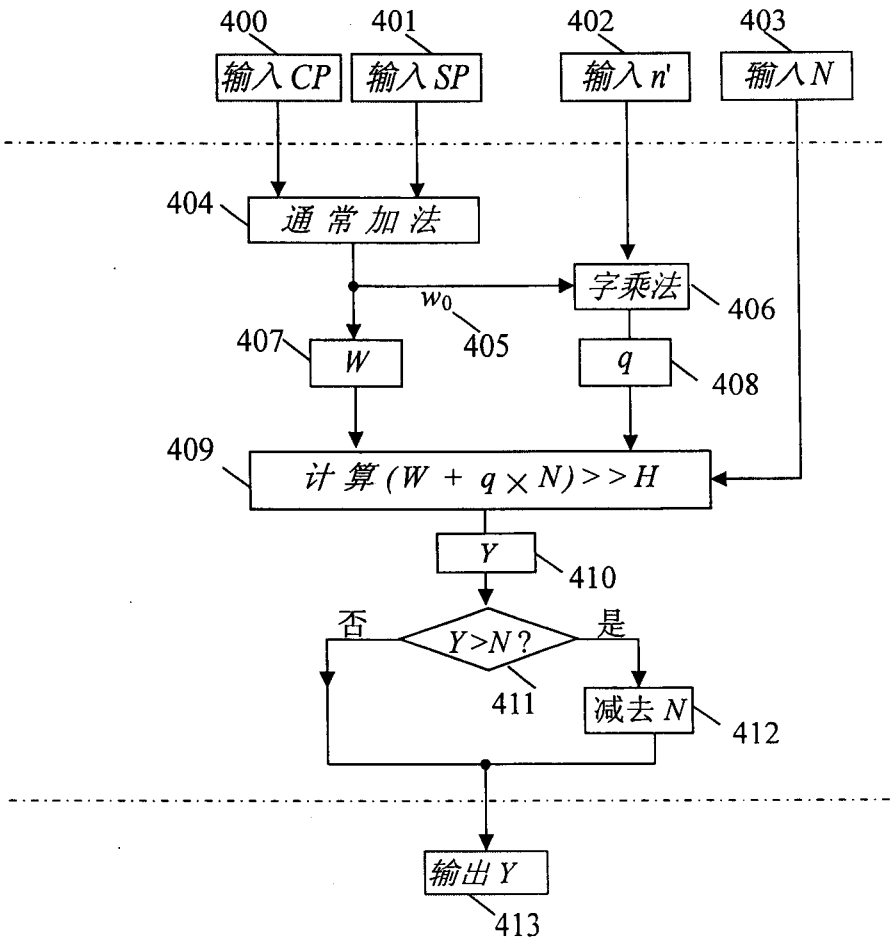

400
401
402
403
输入 CP
输入 SP
输入 n'
输入 N
404
通 常 加 法
字乘法
406
w_0
405
407
W
q
408
409
计 算 (W + q × N) >> H
Y
410
否
Y>N ?
是
411
减去 N
412
输出 Y
413

图 4

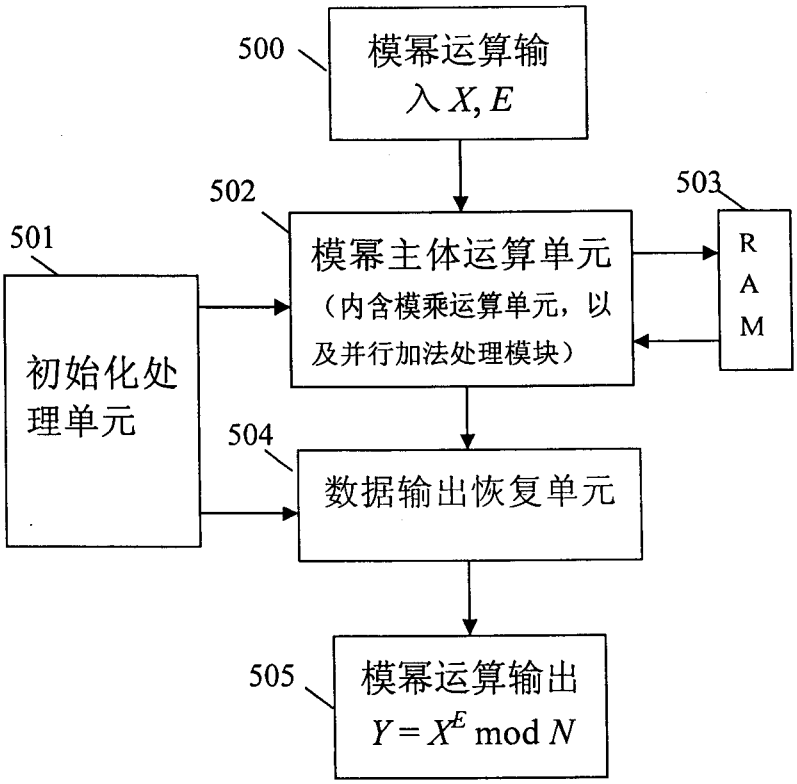

500
模幂运算输入 X, E
502
503
501
模幂主体运算单元
（内含模乘运算单元，以及并行加法处理模块）
R
A
M
初始化处理单元
504
数据输出恢复单元
505
模幂运算输出
$Y = X^E \bmod N$

图 5

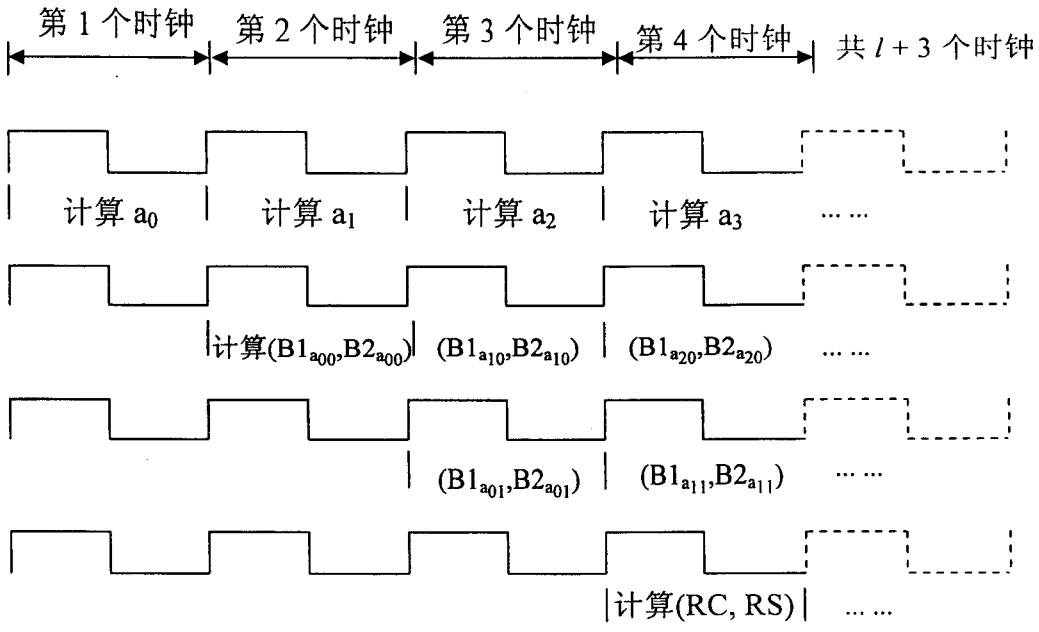

第 1 个时钟
第 2 个时钟
第 3 个时钟
第 4 个时钟
共 l + 3 个时钟
计算 a0
计算 a1
计算 a2
计算 a3
… …
计算(B1a00,B2a00)
(B1a10,B2a10)
(B1a20,B2a20)
… …
(B1a01,B2a01)
(B1a11,B2a11)
… …
计算(RC, RS)
… …

图 6