[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 02806396.1

[45] 授权公告日 2008 年 11 月 26 日

[11] 授权公告号 CN 100437470C

[22] 申请日 2002.1.24 [21] 申请号 02806396.1

[30] 优先权

[32] 2001.3.13 [33] DE [31] 10111987.9

[86] 国际申请 PCT/EP2002/000734 2002.1.24

[87] 国际公布 WO2002/073394 德 2002.9.19

[85] 进入国家阶段日期 2003.9.12

[73] 专利权人 因芬尼昂技术股份公司

地址 德国慕尼黑

[72] 发明人 A·埃尔贝 H·塞德拉克

N·詹斯森 J·-P·塞弗特

[56] 参考文献

DE3631992A1 1987.11.5

US4870681 1989.9.26

CN1172390A 1998.2.4

WO99/14880A2 1997.3.25

IEEE TRANSACTIONS ON COMPUTERS. Colin D. Walter,, 139. 141, Space/Time Trade. Offs for Higher RadixModular Multiplication Using Repeated Addition. 1997

PROCEEDINGS OF THE CONFERENCE ON THEORY AND APPLICATIONS OF CRYPTOGRAPHIC TECHNIQUES (CRYPTO). Colin D. Walter,,313.323, Faster Modular Multiplication by Operand Scaling. 1991

审查员 唐楹琰

[74] 专利代理机构 中国专利代理(香港)有限公司

代理人 程天正 王忠忠

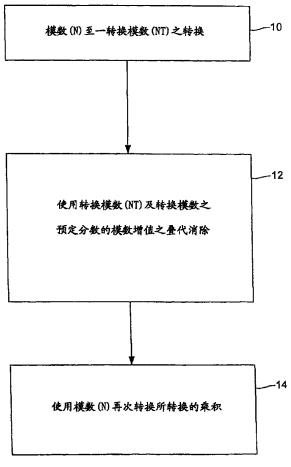
权利要求书 3 页 说明书 16 页 附图 7 页

[54] 发明名称

模数增值方法及装置

[57] 摘要

于一模数增值方法中，使用计算一乘法位移值之一增值预测处理以及计算一缩减位移值之一缩减预测处理，一模数首先被转换(10)为大于该模数之一转换模数。该转换被执行使得该转换模数之一预定分数具有被一包含一第二预定值之至少一低阶位数所跟随而包含一第一预定值之一高阶位数。于使用增值预测处理及缩减预测处理之叠代消除(12)期间，转换模数被使用以便在叠代终了时获得该模数增值之一转换乘积。最后，该转换乘积使用该原始模数藉由模数缩减而被再次转换(14)。藉由依据本发明之转换，模数增值之叠代消除被简化，所以模数增值可被较快的执行。



1.一种在一处理器中一乘数(M)乘上被乘数(C)的模数增值方法，其中一模数(N)被使用，该方法使用一增值预测处理以及一缩减预测处理，该处理器包括一模数缓存器(720)，一乘法中间乘积缓存器(730)，以及一算术单元(700)，该算术单元(700)的输入端与该模数缓存器(720)和该乘法中间乘积缓存器(730)相连接，且其中该算术单元(700)的输出端反馈至该乘法中间乘积缓存器(730)并且与一控制逻辑(760)相连接，该方法包括步骤：

转换(10)该模数(N)为大于该模数(N) 的一转换模数(N^T)，并储存该转换模数(N^T)于该模数缓存器(720)，该转换模数(N^T)的一预定分数具有包含一第一预定值的较高阶缓存器位数以及跟随其后的至少一包含一第二预定值的较低阶缓存器位数；

模数增值的迭代消除(12)，使用该增值预测处理以及该缩减预测处理并使用该转换模数(N^T)，以便在该迭代的终了获得该模数增值的一转换乘积；及

经由使用该模数(N)的该转换乘积的模数缩减，重新转换(14)该转换乘积，

其中，于该迭代消除步骤(12)中执行多个迭代步骤，于一迭代步骤中决定一乘法中间乘积并将其储存于该乘法中间乘积缓存器(730)，且于该迭代步骤中决定一缩减位移值(S_N)，该缩减位移值(S_N)的计算是使用一缓存器位数的数目(Si)的一决定，由该控制逻辑执行，该缓存器位数的数目(Si)是介于含该转换模数(N^T)的该第一预定值的该模数缓存器的该较高阶缓存器位数，与含该第一预定值的该乘法中间乘积缓存器(730)的最高阶缓存器位数之间。

2.如权利要求1所述的方法，

其中为决定该缓存器位数的数目(Si)，该模数缓存器和该乘法中间乘积缓存器(730)的一 XOR 位运算被执行，其中该数目(Si)等于该 XOR 运算输出一第一“1”与该 XOR 运算输出一第二“1”的位数价差。

3. 如权利要求2所述的方法，

其中于该增值预测处理中决定一乘法位移值(Sz)，且其中该缩减预测处理的缩减位移值(S_N)是藉由从该乘法位移值(Sz)减去该缓存器位

数的数目(Si)而被计算。

4.如前述权利要求任一项所述的方法，其中该迭代消除步骤包括以下步骤:

于第一迭代步骤:

(a) 执行一增值预测处理以获得一乘法位移值(Sz);

(b) 以一目前中间乘积(Z)乘上被提高至该乘法位移值的幂次的一基数以获得一位移中间乘积(Z');

(c) 执行一缩减预测处理以便藉由决定等于该缓存器位数数目(Si)的一辅助位移值而获得一缩减位移值(S_N)，并通过使用该辅助位移值及该乘法位移值(Sz)计算该缩减位移值;

(d) 以被提高至该缩减位移值的幂次的该基数乘上该转换模数(N^T)以获得一位移转换模数($N^{T'}$); 以及

(e) 加总该位移中间乘积(Z')与该被乘数(C)并减去该位移转换模数($N^{T'}$)以获得一更新的中间乘积(Z)。

5.如权利要求 1 所述的方法，其中该转换模数的预定分数为 2/3。

6.如权利要求 5 所述的方法，其中该被乘数(C)，该乘数(M)及该模数(N)为二进制数，具有基数 2，且其中该转换模数(N^T)的该预定分数的较高阶缓存器位数具有一第一预定值 1，以及该预定分数的至少一较低阶缓存器位数具有一第二预定值 0。

7. 如权利要求 6 所述的方法，

其中该转换模数的最大有效位是一符号位，而该转换模数的该预定分数的一较高阶区段内含如下:

01000 xx ... xx

其中被标示为 xx 的位具有任意值。

8. 如权利要求 7 所述的方法，

其中该转换模数(N^T)的较高阶区段内含如下:

01100 ... 00。

9.如权利要求1所述的方法，其中转换(10)该模数(N)为大于该模数(N)的一转换模数(N^T)的步骤包括模数的随机化，因此该转换模数是随机的。

10.一种处理器，用于以一乘数(M)乘上一被乘数(C)的模数增值，其中一模数(N)被使用，该处理器使用一增值预测处理以及一缩减预测处理，该处理器包括:

一装置，用来转换(10)该模数(N)为大于该模数(N)的一转换模数(N^T)，该转换模数的一预定分数具有包含一第一预定值的较高阶位数以及跟随其后的至少一包含一第二预定值的较低阶位数;

一装置，用以模数增值的迭代消除(12)，使用该增值预测处理以及该缩减预测处理并使用该转换模数(N^T)，以便在该迭代的终了获得该模数增值的一转换乘积;

一装置，经由使用该模数(N)的该转换乘积的模数缩减,以重新转换(14)该转换乘积。

11.如权利要求10所述的处理器，

包括一主控CPU以及一协处理器，转换(10)该模数(N)为大于该模数(N)的一转换模数(N^T)用的装置被设置于该主控CPU，而该用于模数增值的迭代消除的装置是设置于该协处理器中。

12.如权利要求11所述的处理器，

其中该主控CPU是一位数数目小于或等于64的短数算术逻辑单元，而其中该协处理器是一位数数目大于或等于512的长数算术逻辑单元。

13.如权利要求10至12中任一项所述的处理器，

其中该用于模数增值的迭代消除的装置包括用以转换该模数的一缓存器以及该模数增值的中间乘积用的一缓存器。

模数增值方法及装置

本发明系关于密码算法及实施此种密码算法之装置，尤指一种使用增值预测(multiplication look-ahead)处理及缩减预测(reduction look-ahead)处理之模数(modular)增值之方法及装置。

密码翻译术系模数计算的必要应用。RSA系一种已知之密码翻译术之必要的算法。RSA算法系基于一模数乘方(exponentiation)，其可被表示如下:

$$C = M^{d} \bmod (N),$$

其中C代表一加密讯息，M系未加密讯息，d为秘密钥匙而N为模数。模数N通常由二主要数字p及q之增值所产生。模数乘方藉由已知的平方及乘数(square-and-multiply)算法而被展开为乘法运算。因此，指数d被展开为2的幂次，因此模数乘方可被展开为数个模数乘法运算。为了能有效率地在计算方面实施模数乘方，模数乘方因此被展开为模数乘法运算，模数乘法运算之后可被展开为模数加法运算。

DE 3631992 C2号文件揭露一种密码处理，其中的模数乘法可使用增值预测(look-ahead)处理及缩减预测处理之模数(modular)乘法而被加速。DE 3631992 C2所描述之处理也关于一种ZDN方法，并藉由图9做更详细的描述。在算法的启始步骤900之后，整体变量M,C及N被重置。目的在于计算以下的模数乘法:

$$Z = M * C \bmod N.$$

M是乘数，其中C是被乘数。Z是此模数乘法之乘积，其中N是模数。

然后，有许多暂时不需要被处理的被重置的区域变量(local variables)。之后使用二预测处理。在增值预测处理GEN_MULT_LA中，一乘法位移(shift)值Sz以及增值预测参数a使用不同的预测规则而被计算(910)。接着，Z缓存器目前的内容接受Sz次的左位移运算(920)。

实际上与之平行处理的是执行一缩减预测处理 GEN_Mod_LA(930)用以计算一缩减位移值 S_N 以及一缩减参数 b。于步骤 940，模数缓存器目前的内容，亦即 N，分别被向左或向右位移 S_N 次，以便产生一已位移的模数值 N'。ZDN 方法之中央三操作数(operand)操作发生于步骤 950。于此步骤中，步骤 920 之后的中间乘积 Z'被加到已被乘上增值预测参数 a 的被乘数 C 以及已被乘上缩减预测参数 b 之已位移的模数 N'。依据目前的情况而定，预测参数 a 及 b 的值可能为+1，0，或 -1。

一种典型的情况是增值预测参数 a 是+1，而缩减预测参数 b 是-1，因此被乘数 C 被加到已位移的中间乘积 Z'，而已位移之模数 N'则被减去。如果增值预测处理允许大于一预设数目之个别左移，亦即，如果 Sz 大于 Sz 的最大可接受值，其也被称为 k，则 a 的值将是 0。在 a 的值为 0 的情况下，而 Z'，由于已位移模数之前的缩减，将依然十分地小，尤其是小于已位移的模数 N'，b 为 0 的时候不产生缩减。

步骤 910 至 950 被执行直到所有被乘数的数字已被消除或处理，亦即，直到 m 为 0 且直到一参数 n 等于 0 为止；此参数指示位移的模数 N'是否依然大于原始的模数 N，或是是否尽管所有被乘数的数字已被消除，仍然有许多从 Z 减去之模数减法的进一步缩减步骤需要被执行。

最后，决定 Z 是否小于 0。如果是，则需要达成模数 N 被加到 Z 的最终缩减，所以在此获得模数乘法的正确乘积 Z。在步骤 960，藉由 ZDN 方法之模数乘法于此终止。

乘法位移值 Sz 以及在步骤 910 藉由增值预测算法所计算的乘法参数 a 系从乘法器拓朴(topology)以及 DE3631992 C2 所描述之预测规则所产生。

缩减位移值 S_N 以及在 DE3631992 C2 所描述之缩减参数 b 系由比较 Z 缓存器目前的内容与 2/3 乘以 N 的值所决定。此比较给予 ZDN 方法的名称(ZDN = Zwei Drittel N (= 3 分之 2 N))。

如图 9 所示之 ZDN 方法转变模数乘法为三个操作数加法(图 9 的方块 950)，其中增值预测处理以及同时发生的缩减预测处理被用以增加计算时间效率。因此，与蒙哥马利缩减(Montgomery reduction)相较之下可达成计算时间方面的益处。

在下文，于方块 930 中所执行的缩减预测处理将藉由图 10 而被详细描述。首先，在方块 1000，为区域变量，亦即缩减预测参数 b 及缩

减位移值 S_N，执行一个保留。在方块 1010，缩减位移值 S_N 被重置为 0。然后，数值 ZDN 在方块 1020 中被计算，其等于模数 N 的 2/3。在方块 1020 中所决定的这个值被储存于密码协处理器(crypto coprocessor)中其本身之缓存器，称为 ZDN 缓存器。

之后在方块 1030 中决定变量 n 是否为 1 或是位移值 S_N 是否为-k。k 是定义硬件所预设之最大位移值。于第一次运作中，方块 1030 被回答"否"，因此在方块 1040，参数 n 被减少且于方块 1060 中，缩减位移值被减少 1。在方块 1080 中，变量 ZDN 随后被重新配置，即其值的一半，其可轻易地藉由包含于 ZDN 缓存器中的一个右位移缓存器而达成。随后在方块 1100 中决定目前的中间乘积的绝对值是否大于 ZDN 缓存器中所包含的值。

方块 1100 中的比较运算是缩减预测处理的中央运算。如果此问题的答案为"是"，此叠代(iteration)停止，且缩减预测处理参数 b 将被配置如方块 1120 所示。如果，与之相反，在方块 1100 将被回答的问题的答案为"否"，此叠代往回跳以便检查方块 1030 中的目前述值 n 及 S_N。如果在叠代的任何时间中方块被回答"是"，流程跳回缩减参数被设定为 0 的方块 1140。在方块 950 所示之三操作数运算，这具有无模数被加或减的效用，其表示中间乘积 Z 很小以致于不需要模数缩减。在方块 1160，变量 n 随后被重新配置，且最后于方块 1180 此缩减位移值 S_N 被计算，其于图 9 的方块 940 中是需要的以便执行模数的左位移而获得一位移的模数。

在方块 1200,1220 及 1240，n 及 k 的目前的值最后为了检查 N 缓存器目前的配置而相对于下一步的变量 MAX 及 cur_k 被检查，为确定没有发生超出缓存器的情况。此结束的细节与本发明无关，但于 DE3631992 C2 中有详细描述。

图 9 及 10 所示之算法可以图 7 所示之硬件实施。对于方块 950 中所执行的三个操作数的运算，需要一个算术单元 700，图 7 中标示为 AU。之后被耦合至被乘数的缓存器 C 710，模数用的缓存器 N 720 以及模数乘法之目前中间乘积用之缓存器 Z 730。此外，图 7 揭露，三操作数的运算，经由回馈箭头 740，系被馈入 Z 缓存器 730。此外，图 7 表示这些缓存器的相互连接关系。在图 10 的方块 1020 中所计算的 ZDN 的值必须被储存于其本身的 ZDN 缓存器 750。此外，ZDN 比较，即图 10

所示之叠代循环(loop)，藉由其本身之 ZDN 比较用之一控制逻辑 760 而于其进度中被控制。

计算 Z: = M x C mod N 之 ZDN 算法的主要工作因此包含以下二运算:

1.计算缓存器 Z 及 N 之位移值 Sz 及 Si 以便满足以下的程序:

$$2/3\ N \times 2^{-si} < |z| \leqq 4/3\ N \times 2^{-si}$$ 以及

2.计算三操作数的总合:

$$Z: = 2^{sz}\ Z + Ac + b \times 2^{sz-si}\ N$$

此增值预测参数 a 及缩减预测参数 b 可如已知般被假设为数值-1, 0 及+1。

必须指出的是中间乘积 Z，被乘数 C 以及模数 N 是长的数字，亦即其位数或位的计数实际上可能大于 512，且其也可能达到大于 2048 位数。

然而，在方块 1100 中所执行的目前中间乘积 Z 与数值 ZDN 的比较为了计算时间的缘故，并不会对 Z 的所有位进行，而只对 Z 的最大有效位(most significant bits)进行; 在此情况下，32 位的数目被移除以便足以获得比较结果的高度精确度。

对于此必较所需之 2/3 N 之 32 个最大有效位，需要其本身的缓存器，其于图 7 中以参考标号 750 指示并被指定为一 ZDN 缓存器。

此外，其本身也需要一个计算 Z 缓存器中之目前数值以及 ZDN 缓存器中之目前数值的正确 Si 值以便满足以下方程式的硬件比较器:

$$2/3\ N \times 2^{-si}\ N < |z| \leqq 4/3 \times 2^{-si}\ N$$

因此，此方法的缺点在于一方面二个额外的 ZDN 缓存器及硬件比较器都需要额外的芯片区域。另一方面，2/3 N 的计算及图 10 所示之由叠代循环所执行的 ZDN 算法中的辅助的位移值 Si 的计算对整体算法而言是时间的关键，并且可能实际上对算法整体的运算时间有决定性。

本发明之一目的在于提供一种模数增值的概念，其一方面可以用

节省空间的方式被实施，且另一方面需要较少的计算时间。

此目的藉由申请专利范围第 1 项之模数增值方法或申请专利范围第 14 项之模数增值装置而达成。

本发明系基于更新的中间乘积与 ZDN 数值，即 2/3 乘以模数 N，的比较的发现，该比较包括计算时间中的高消耗时间可于模数 N 首次被转换(transform)为转换的模数 N^T 的时候被利用，且整个模数增值以转换的模数 N^T 取代特定的模数而被执行。依据本发明，此模数被转换，因此被转换模数之预定的分数(fraction)，亦即，于一较佳实施例中为 2/3 乘以转换的模数，变成一特定的数，此数被选择为使得 $2/3N^T$ 与中间成积 Z 的比较变为微不足道的。依据本发明，此转换被执行因此该转换模数的该预定分数具有第一预定值的较高阶数的位数，其后接续具有一第二预定值的至少一低阶数的位数。在数字的表示及二的补码规定中，最大有效位代表符号，模数至被转换模数的转换被执行因此 2/3 N^T 的第二最大有效位系一二进制值，然而第三最大有效位及次一小的有效位是 0。

于此种情况下，此比较是微不足道的，因此其仅需计算转换模数之预定部份的最大有效位数与模数代表之更新的中间乘积 Z 以便获得位移值 Si，随后从该值可轻易地藉由从平行发生的增值预测处理之乘法位移值减去从 ZDN 比较所得之所谓的辅助值 Si 而决定缩减位移值 S_N。

全部的 ZDN 运算如同习知情况般被消除。然而，取代模数 N，转换的模数 N^T 被使用，因此最后达成模数增值之"转换的结果"，其系于转换模数 N^T 之余项集(reminder class)中。最终的重新转换，因此模数增值以模数的方式被降低，使用原始的模数 N，将随后产生使用模数 N 以被乘数 C 乘以乘数 M 之模数乘法的适合乘积。

本发明之较佳实施例将于以下参照附图而被详细解释，其中：

图 1 系依据本发明之模数增值之概念的流程图；

图 2 系分离模数 N 为第一部份位 N^T 及第二部份位 N^R；

图 3 系分离模数 N 为具有长度 L(N^T)及剩余位之第一部份位 N^T

图 4 系 2/3 乘以转换模数 N^T 之位表示；

图 5 系随机化转换模数之位表示；

图 6 系依据本发明执行模数增值之算术逻辑单元图式；

图 7 系已知 ZDN 方法之算术逻辑单元图式；

图 8a-8c 系增值位移值 Sz 与辅助位移值 Si 及缩减位移值 S_N 之关系的表示；

图 9 系已知 ZDN 方法之流程图；

图 11 系已知缩减预测处理之流程图。

图 1 表示依据本发明使用模数 N 以乘数 M 乘以被乘数 C 之模数增值之方法的流程图。首先，模数 N 在步骤 10 依据以下方程序被转换为转换模数 N^T：

$$N^T = T \times N$$

在步骤 12，模数乘法随后使用转换模数 N 以及在较佳实施例中预定为 2/3 的转换模数的部份而被消除。相对于模数乘方，这表示以下形式的 RSA 方程式被计算：

$$C^T := M^d \bmod N^T$$

因此，模数乘方 C 的积不在由模数 N 所定义的余项集中被计算，而是在转换模数 N^T 所定义的余项集中被计算，因此，C^T，而非 C，位于上述方程式的左方。依据本发明概念之区别在于，由于转换模数 N^T 的使用，对应已之缩减预测处理之图 10 的叠代循环的辅助缩减位移值 Si 被高度简化。

在最后的步骤 14，N^T 至 N 的再转换再次被执行，藉由执行对应以下方程序的运算：

$$C := C^T \bmod N$$

在此情况中，位于转换模数 N^T 之余项集中的转换乘积 C^T 藉由简单的位移/减法缩减而回到模数 N 的余项集，因此 C 为模数乘方的乘积

使用步骤 10 的转换子 T 转换模数 N 为转换的模数 N^T 的转换被执行，因此该转换模数的该预定分数，亦即于较佳实施例中为 2/3 乘以转换模数，具有含较一第一预定值的较高阶位数，其后跟随至少一含

一第二预定值之低阶位数。具有 2/3 乘转换模数的中间乘积 Z 的比较可因此被高度简化，亦即，Z 的最高位，其具有预定值，寻找，且具有该转换模数的该预定分数的第一预定值的较高阶位数与具有第一预定值的中间乘积 Z 的最高位数的差异等于差异 Si。

综言之，这可如以下所示。N 在 32 位 CPU 而非在密码协处理中被转换为转换模数 N^T，因此适用以下：

$$N^T = T \times N$$

其中 T 为自然数。

以下由 N^T 所产生，如果所有的使用的数字为二进制数：

$$N^T = 1100...0\ XX...XX$$

对 2/3 乘以转换模数而言，则产生以下的结果：

$$2/3 = 100...\ 0\ X'X'...X'X'$$

可以从 N^T 及 2/3 N^T 看出二者皆具有例如 16 位的第一部份以及分别为位 X 及 X'的 L(N)部份。对于所谓的 ZDN 比较而言，只有 2/3 乘以转换模数 N^T 的最高 16 个位被使用，因为其已产生优于大约 2^{-10} 的误差可能性。因此，对 ZDN 比较不需要使用 2/3 乘以转换模数的所有 512，1024 或 2048 位，但其已足以执行与转换模数之最高 16 位的比较。当然，也可能使用 2/3 乘以转换模数的较少位来做比较，但误差可能性逐渐增加。然而，因为此误差是不重要的并且只产生在缩减预测处理的次最优行为中，此方法实际上很容易执行。

2/3 乘以转换模数 N^T 因此具有含数值 1 的较高阶位数，其后跟随具有一数值 0 的至少一低阶位数及因此一第二预定值。于之前所述之实施例中低阶位数的数目为 15。此处也有可能使用较高或较少的数，依据中间乘积 Z 与 2/3 乘以转换模数 N^T 之间所期望或处理的维度(dimensional)差异。对于模数乘积之中间乘积 Z，亦即图 9 方块 950 中的三操作数的相加，产生以下的形式：

$$| Z | = 00...01YY...Y$$

辅助位移值 Si 依据以下方程序被计算:

$$2/3\ N^T \times 2^{-si} < | z | \leqq 4/3\ N^T \times 2^{-si}$$

在 2/3 乘以转换模数 N^T 的拓朴下，数值 Si 永远是具有 1 的 2/3 乘以转换模数 N^T 最大有效位与中间乘积之值的最大有效 1 之间的距离。

依据本发明,位数或数值 Si 之间的差异可以在一般的方式中决定。因此不再需要叠代。

除了以上,储存 2/3 乘以转换模数 N^T 的 ZDN 缓存器也不再被需要，因为，就定义而言，2/3 乘以转换模数 N^T 的至少较高的，例如 16 位，总是具有相同的形式。也不再需要位比较器。具有一个"1"的 2/3 乘以转换模数 N^T 的最高阶位数与具有一个"1"的 Z 的最高阶位数之间的有效差异可轻易地被建立，例如藉由转换模数用的 XOR 运算缓存器以及中间乘积 Z 的缓存器。Si 随后等于位数的有效差异，其中 XOR 运算输出一第一"1"且其中 XOR 运算输出一第二个"1"。

由于不需要 ZDN 缓存器及 ZDN 比较器的事实，整体的算术逻辑单元可被容纳于较少的芯片区域。

除了以上，密码控制部，即 ZDN 比较的控制逻辑(图 7 的 760)较不复杂，因为不需要执行图 10 的复杂叠代。最后，此比较是较快的，因此辅助位移值 Si 的计算不再导致整个算法的时间问题。

在下文，依据本发明的转换将藉由图 2 至 5 被详细描述。

如同已被指出之部份，ZDN 算法的实质部份包括满足以下的方程式:

$$2/3\ \ 2^{-si}\ N < | z | \leqq 4/3\ \ 2^{-si}\ N$$

Si 系关于辅助位移值且为位移 Z，在位数方面，至与 N 相同的位置所需之位移值。在习知技术中，Si 的计算需要| Z |与 2/3 N 的比较

运算。

依据本发明，在任何以 N 的模数运算被执行之前以大于 N 的转换模数 N^T，与 2/3 N 的比较简单地藉由转换模数 N 至转换模数 N^T。所有的计算模数 N^T 于其后被执行。然而，因为计算的结果必须在余项集 N 之中，依据本发明执行以 N 的最终缩减。

如图 2 所示，N 被表示为长度为 N 位的整数。由于模数 N 总是为正整数的事实，亦即在二的补码表示中 MSB = 0，符号位等于 0 且模数 N 的最大有效位(MSB-1)总是为 1。ZDN 比较并不需要比较模数的所有位与中间乘积，相反地，ZDN 的比较使用 m 位的数字即已足够。模数 N 的最大 m 位定义模数的第一部份 NT，其中模数的剩余 N-m 位定义模数的第二部份 NR。在较佳实施例中，m 等于 16。较高或较低的 m 值，当然也是可能的。

如于图 3 所示，此转换被执行以便转换模数 N^T 比图 2 的原始模数长 16 位。

ZDN 比较使用 N^T 的前 16 位是足够的，以本发明较佳实施例仅使用 12 位以为比较之用，而 4 个最小有效位构成可能承载的缓冲，其可来自更少的有效位。

在此情况下，产生错误结果的比较可能性小于 2^{-12}。如果此比较产生错误的结果，其仅产生一次最佳位移值 S_N，然而，所产生的模数 N 依然正确。

如果此模数如图 2 以 2 的补码表示，模数 N 可被展开如下：

$$N = 2^{n-m} \quad NT \quad + \quad NR$$

现在 N 使用转换子 T 被转换为 N^T，为全等(congruence)的理由，T 为适合的被选择整数。N^T 应具有图 3 所示之形式，亦即，N^T 的最大有效位(MSB)必须为 0，因为 N^T 应为一正整数。如下文所解释，N^T 的第二最大有效位及第三最大有效位必须为 1，因此，转换模数 N^T 最高部份的有效位的其它部份，此部份为图 3 的参考标号 33，的数值应该为"0"。仅于此例中，2/3 乘以 N^T 的乘积，2/3 乘以 N^T 的最高部份，如图 4 所示，仅有一位为"1"，因此于此最高部份 44 的所有位皆为"0"，因此已经描述之决定 Si 之一般的比较可被执行。

然而，使用转换子 T 的转换模数 N^T 的计算将首先参照图 3 而讨论。假设以下的定义：

$$N^T = T\ N$$
$$= T(2^{n-m}\ NT\ +\ NR)$$

以下维持转换子 T：

$$T = \left| \frac{2^{p-2} + 2^{p-3}}{N^T} \right|$$

使用程序 17，以下产生转换模数 N^T：

$$N^T = \left| \frac{2^{p-2} + 2^{p-3}}{N^T} \right| \quad (2^{n-m}\ NT\ +\ NR)$$

$$N^T = (2^{n+p-m-2} + 2^{n+p-m-3}) \frac{NT}{NT} + (2^{p-2} + 2^{p-3}) \frac{NR}{NT}$$

如果，以 p 及 m 的典型值为例，亦即当 p 等于 32 位而 m 等于 16 位时，N^T 有如下的结果：

$$N^T = 2^{n+14} + 2^{n+13} + \left| \frac{2^{p-2} + 2^{p-3}}{N^T} \right|$$

将指出的是 N^T 的计算最好在主控 CPU 中执行而不是在密码协处理器中执行。此主控 CPU 包含一短数演算逻辑单元，其已足以计算 N^T。由于 T 必须为整数的事实且此计算必须于密码协处理器模数 N^T，而不是模数 N 中进行，以 N^T 大于 N，只有 N^T 的第一个 p-m 等于 16 位与一般的 ZDN 比较相关，以便计算辅助位移值 Si。N^T 其它的 n 位可以是任

何数字，它们与辅助位移值 Si 的计算无关，亦即，对于与 Z 的比较。然而，转换模数 N^T 的所有位，当然，对于三操作数的加法是需要的，其现在，取代位移模数的使用，使用位移转换模数而被执行。

如图 17 所示，转换子 T 系为 m 与 p 所选择之数值中之一 16 位整数。计算 T 及计算 N^T 的除法，因此必须分别仅对最大有效 32 位执行，且因此可以快速轻易地在主控 CPU 上被程序化。

图 4 表示 2/3 乘以转换模数 N^T。当 N^T 的 MSB-1 及 MSB-2 为"1"时，如图 3 所示，且因为以下适用

$$(11)_2 = (3)_{10} \text{ 以及 } (2/3 \times 3)_2 = (2)_{10} = (10)_2$$

2/3 乘以转换模数 N^T 之一简单位形式，具有 2/3 乘以转换模数 N^T 之长度为 n-m+p。

由于 2/3 N^T 的特定形式，与 | Z | 的比较现在变得比较简单。已知 2/3 N^T 的最高阶 1 为一模数运算初始的位置 n+p-m-2。在较佳实施例中之缓存器 Z 的指针(pointer)随后在 Z 的 MSM 开始，并寻找 Z 的第一个"1"。如果 Z 的 MSB 为 1，Z 将是一个负数，且相反地，将寻找 Z 的第一个 0。

缓存器 N 及缓存器 Z 中的第一个 1 的位位置的差异将决定辅助位移值 Si。

由于模数运算的结果必须在余项集 N 之中，最后缩减模数 N 依据本发明而被执行，其表示再次的转换必须被执行(图 1 的步骤 14)。

N 至 N^T 的转换与已知的 ZDN 相较之下有以下的优点：

取代密码协处理器中的 2/3N 的计算，简单的 N 至 N^T 的转换可于主控 CPU 中被执行。

在芯片上不需要 ZDN 缓存器也不需要比较器逻辑，因此芯片尺寸被降低且协处理器的复杂度被降低。

最后，N 至 N^T 的转换可与模数 N 的随机化结合，如图 5 所示。

当 R 为一长度 s 为 位的随机数(random number)时，随机的转换模数 N^T 具有图 5 所示的形式。由于随机数 N，此被随机化的转换模数，当与未执行随机化的情况(图 3)相较之下，较其长 s 位，亦即，藉由 R 的数字的数目。

于一方程式的形式中，这可被表示如下：

$$N^T = T\ N$$
$$= T\ (2^{n-m}\ NT\ +\ NR)$$

随后，此随机化的 T 如下：

$$T = \left| \frac{2^{p-2} - 2^{p-3} + R}{N^T} \right|$$

因此，产生以下的随机转换模数：

$$N^T = \left| \frac{2^{p-2} - 2^{p-3} + R}{N^T} \right| (2^{n-m}\ NT\ +\ NR)$$

$$N^T = (2^{n+p-m-2} + 2^{n+p-m-3} + R\ 2^{n-m}) \frac{NT}{NT} + (2^{p-2} + 2^{p-3} + R) \frac{NT}{NR}$$

当选择 p 具有 144 位，m 具有 16 位且 s 具有 112 位，包含随机化的转换模数 N^T 之数值结果如下：

$$N^T = 2^{n+126} + 2^{n+125} + R\ 2^{n-16} + NR \frac{2^{144} + 2^{143} + R}{N^T}$$

N^T 的位长度为：

$L(N) = n+p-m = n+m+s = n+16+112 = n+128$ 位

图 6 表示依据本发明之一算术逻辑单元，其与图 7 相较之下，不再需要一 ZDN 缓存器，而仅为一算术单元 700，一 C 缓存器 710，一 N 缓存器 720 以及一 Z 缓存器 730，以 N 缓存器 720 不再储存模数或一位移的模数，而是转换的模数或一位移转换模数，或一随机转换模数或一位移随机转换模数。

在下文，图 8a 至 8c 将被描述以说明使用辅助缩减位移值 Si 计算缩减位移值 Sz 的计算。图 8a 表示中间乘积 Z 以及模数 N。仅由范例，此中间乘积具有 4 位，而模数具有 9 位。现在假设图 2 的方块 214 计算一位移的中间乘积 Z'，其可藉由乘上 Sz 而达成。假设乘数具有 8 个 0，其乘法位移值 Sz 为 8。为获得模数缩减，模数 N 必须被位移到位移的中间乘积多项式 Z'的最高位等于位移的模数 N 的最高位的程度。如由图 8b 可见，在此情况中需要一个为 3 的缩减位移值 S_N。

如从图 8b 所示，S_N的决定实际上可仅于 Sz 已被计算后被执行，亦即，图 2 之方块 210 与方块 212 的平行实施，如本发明之较佳实施，是不可能的。为此理由辅助位移参数 Si 被导入。Si 的益处在于此数值可被计算而不需要已知之目前步骤的 Sz。

从图 8b 中可见，在所有时间 Sz 等于 Si 与 S_N的总合。因此 S_N总是与 Si 相关，因此适用下方程式：

$$S_N = Sz - Si$$

因此，决定 S_N用之消耗时间的叠代处理可被展开为决定 Si 用之一耗时的叠代步骤（循环 416）以及一快速差额运算（图 4 的循环 422）。因此，二预测处理的平行实施是可能的，以单一串行组件，存在于，在计算方块 422（图 4）之前 Sz 的实际数值已经被增值预测算法所计算及传递（图 2 的箭号 230）。

综上所述，与已知的 ZDN 相比较之下，本发明简化 2/3N 与 Z 值之间的比较。对照目前所知的方法，其中 2/3N 的最高 32 位在密马协处理器中被计算并被放置于一分离的 32 位缓存器，此 ZDN 缓存器，经由构成密码协处理器之控制部之组成部份的比较器，以 2/3N 与具有已经

依据习知 ZDN 方法在硬件中被执行的 Z 值的比较，此方法的进行如下。此模数 N 被主控 CPU 转换为大于 N 的转换 N^T，具有 N^T 的第一位为一常数，此常数被选择以便让 2/3 N^T 与 Z 值的比较是微小的。为改进信息渗漏侵害方面的安全性，如 SPA，DPA，时间侵害，...，N 至 N^T 的转换可与模数的随机化结合，如前所述。

因此在密码协处理器中的 2/3N 的计算可以被免除。ZDN 缓存器及比较逻辑也可被省略，因此提供较小的芯片区域并藉由省略比较器逻辑而降低密码协处理器中之控制部的复杂度。

参考标号表

10 模数转换
12 模数增值之叠代消除
14 转换乘积之再转换
33 转换模数之上区段
44 2/3 乘以转换模数之上区段
700 算术单元
710 C 缓存器
720 N 缓存器
730 Z 缓存器
740 叠代循环
750 ZDN 缓存器
760 ZDN 比较的控制逻辑
900 ZDN 方法的开始
910 ZDN 算法的增值预测处理
920 Z 的向左或向右位移
930 ZDN 算法的缩减预测处理
940 模数的向左位移
950 ZDN 算法的三个操作数运算
960 ZDN 算法的终止
1000 整体变量
1010 缩减位移值的重置
1020 ZDN 的计算
1030 n 及 S_N 的检查
1040 n 的缩减
1060 缩减位移值的缩减
1080 ZDN/2 的计算
1100 中间乘积与 ZDN 的比较
1120 决定缩减预测参数
1140 决定缩减预测参数
1160 n 的计算

1180 缩减预测参数的计算

1200 n 的检查

1220 cur_k 的计算

1240 cur_k 的计算

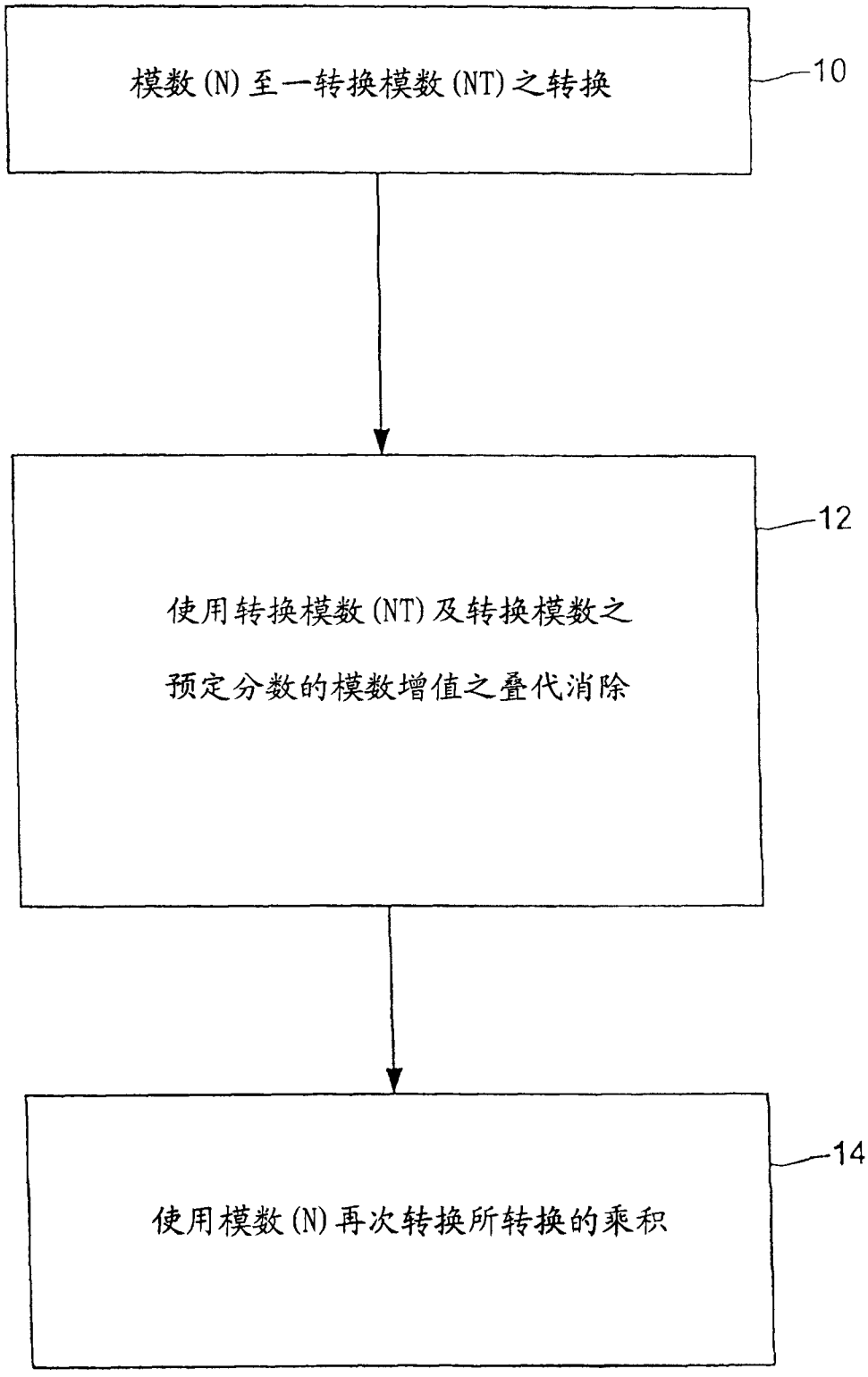

模数(N)至一转换模数(NT)之转换
10
使用转换模数(NT)及转换模数之
预定分数的模数增值之叠代消除
12
使用模数(N)再次转换所转换的乘积
14

图 1

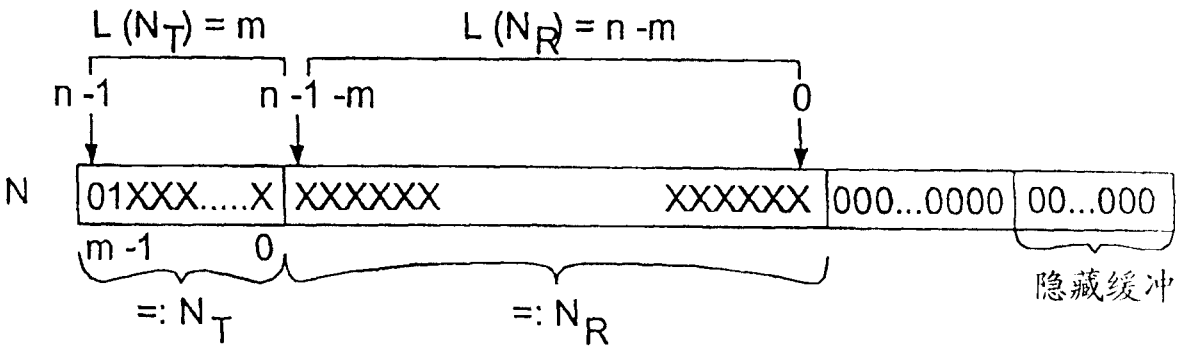


图 2
(模数)

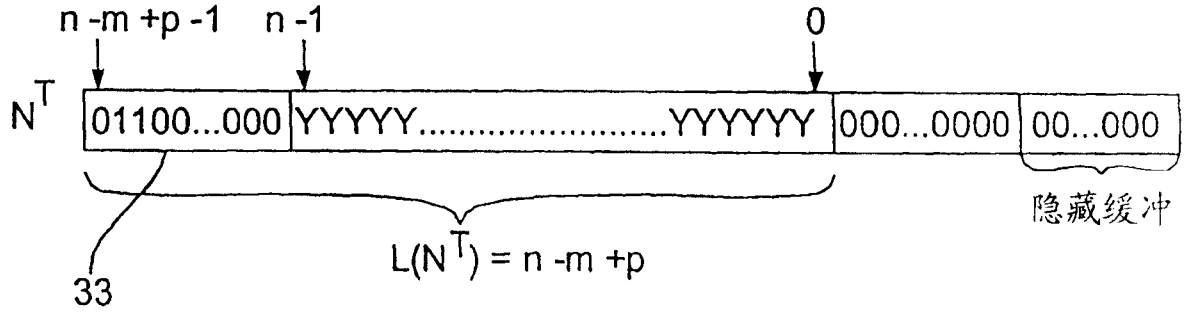


图 3
(转换模数)

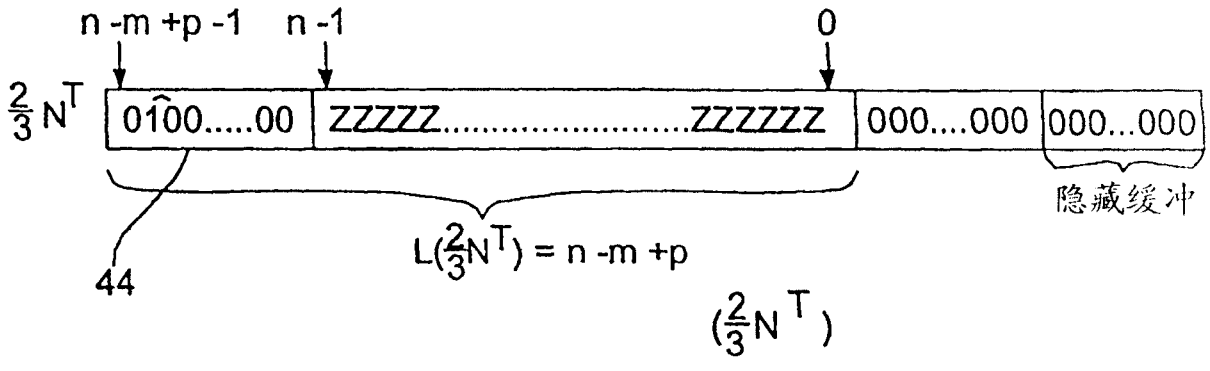


图 4

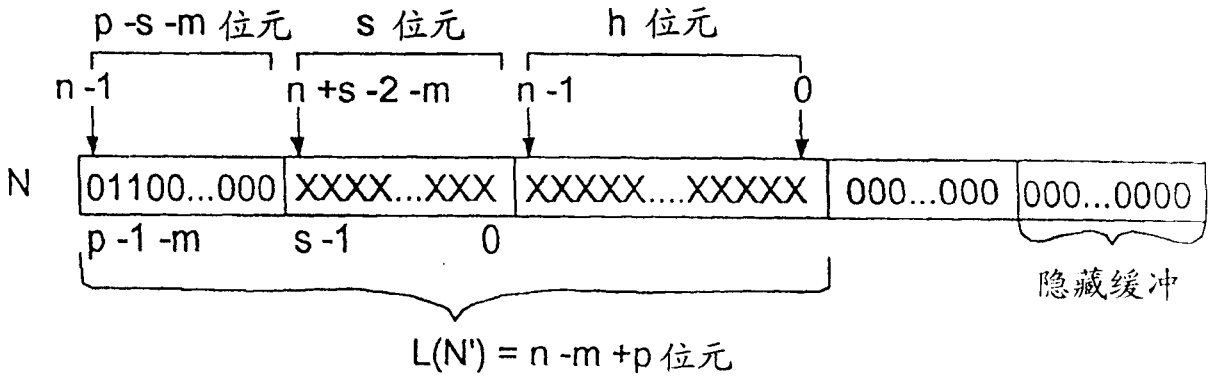


图 5

(随机化的转换模数)

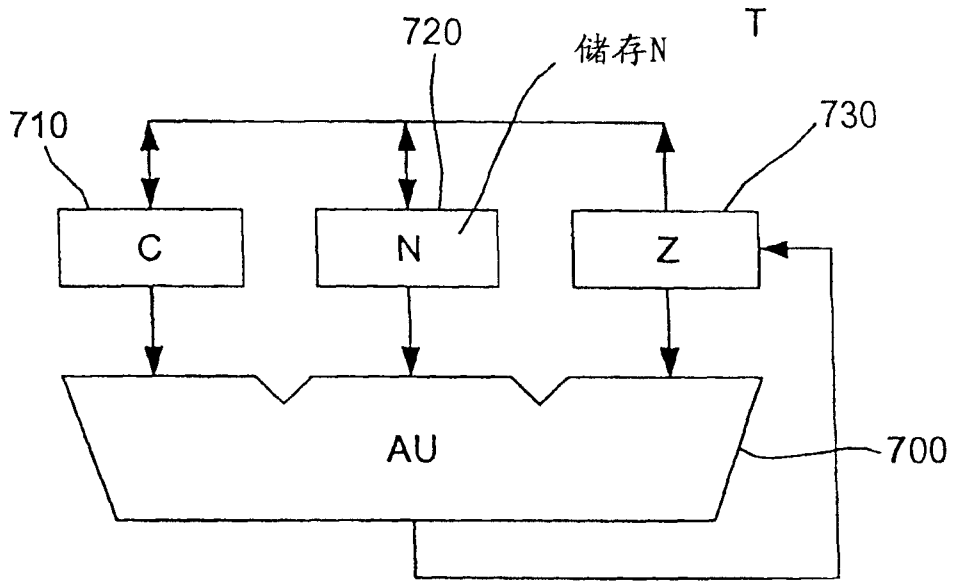


图 6

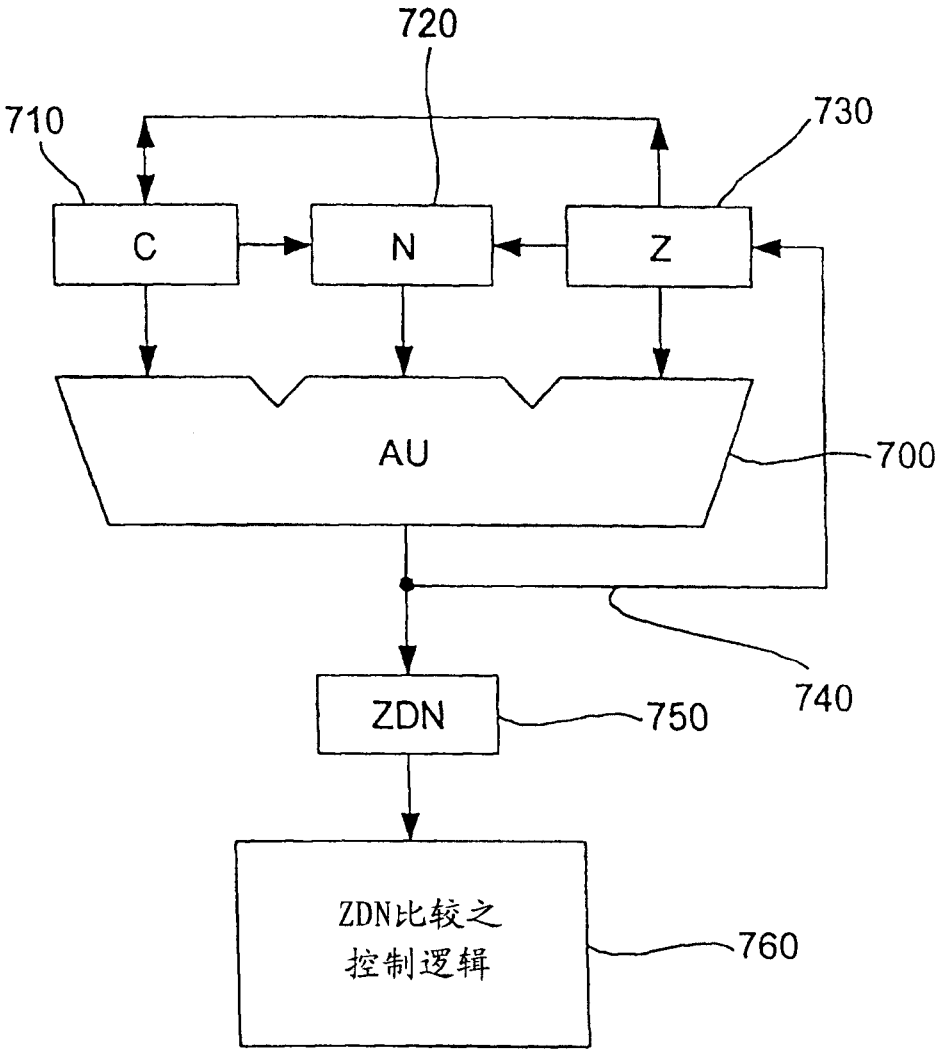


图 7

(现有技术)

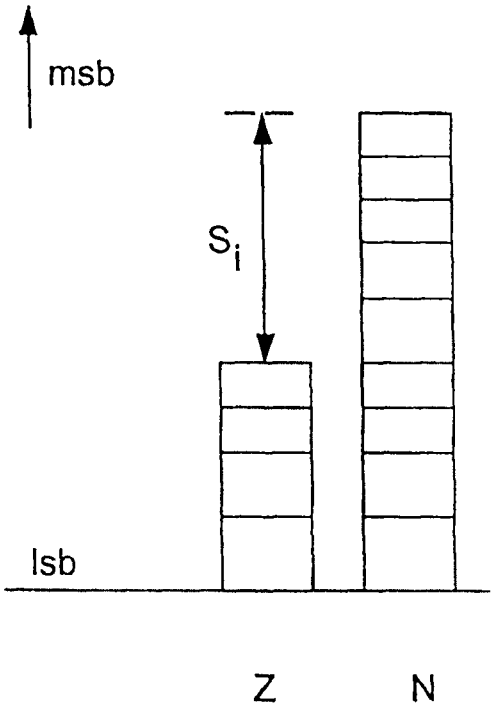


图 8A

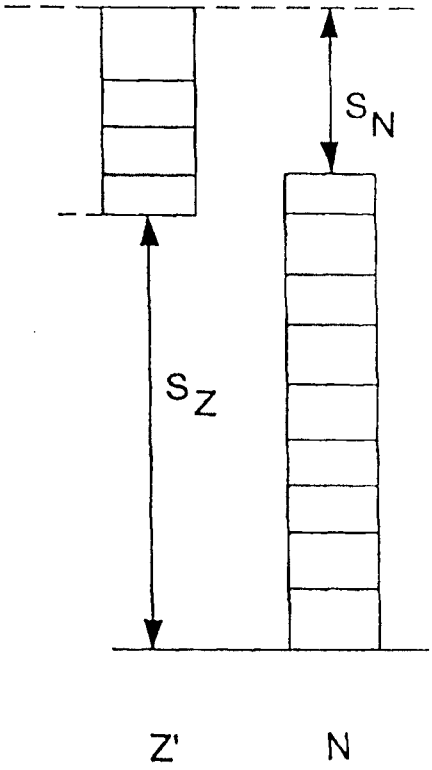


图 8B

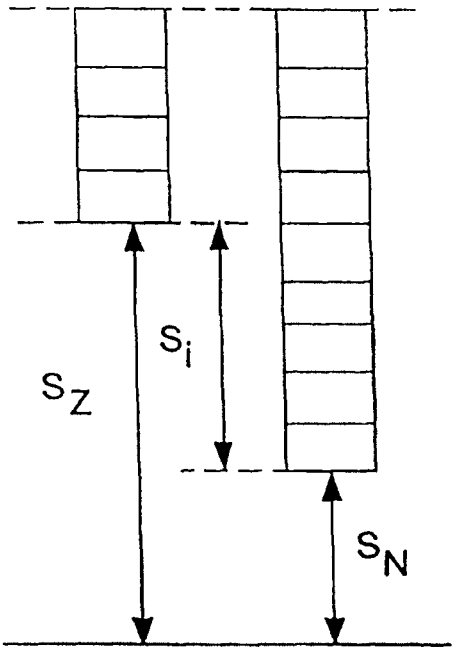


图 8C

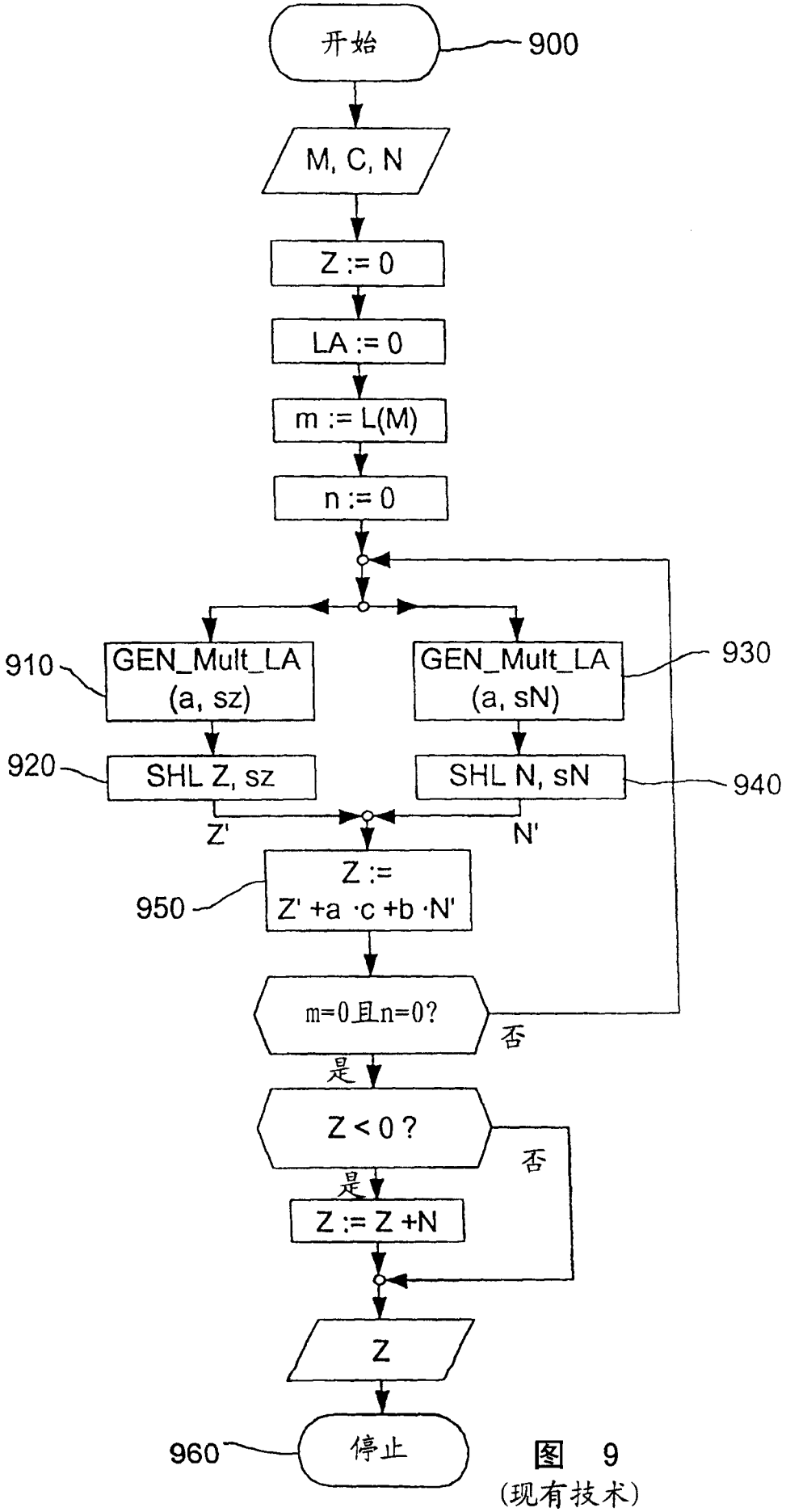

开始
900
M, C, N
Z := 0
LA := 0
m := L(M)
n := 0
910
GEN_Mult_LA
(a, sz)
930
GEN_Mult_LA
(a, sN)
920
SHL Z, sz
SHL N, sN
940
Z'
N'
950
Z :=
Z' +a ·c +b ·N'
m=0且n=0?
否
是
Z < 0 ?
否
是
Z := Z +N
Z
960
停止

图 9
(现有技术)

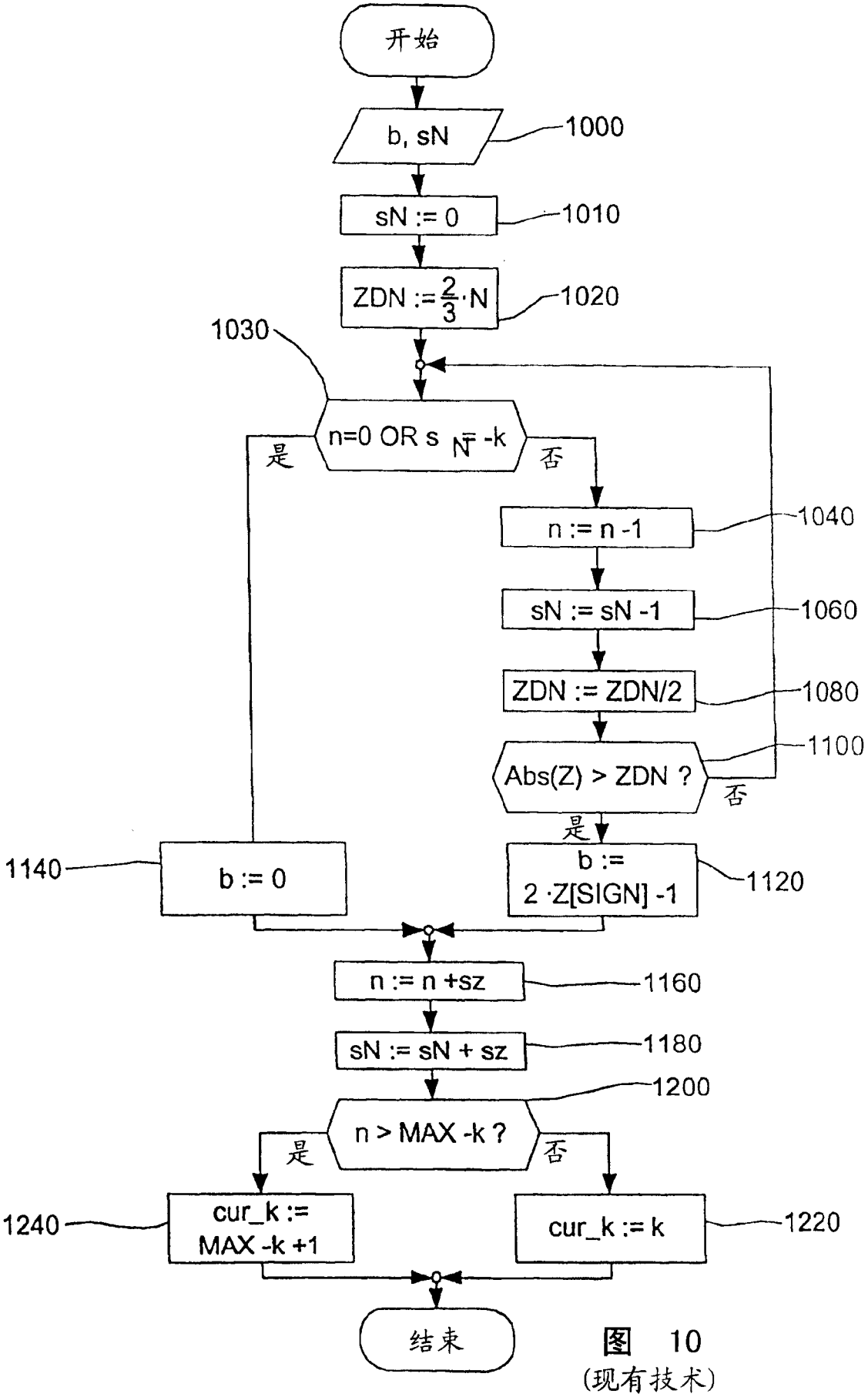

开始
b, sN
1000
sN := 0
1010
ZDN := 2/3 · N
1020
1030
n=0 OR s N -k
是
否
n := n -1
1040
sN := sN -1
1060
ZDN := ZDN/2
1080
Abs(Z) > ZDN ?
1100
否
是
1140
b := 0
b := 2 ·Z[SIGN] -1
1120
n := n +sz
1160
sN := sN + sz
1180
1200
n > MAX -k ?
是
否
1240
cur_k := MAX -k +1
cur_k := k
1220
结束

图 10
(现有技术)