(19)中華民國智慧財產局
(12)發明說明書公開本 (11)公開編號：TW 201320700 A1
(43)公開日： 中華民國 102 (2013) 年 05 月 16 日

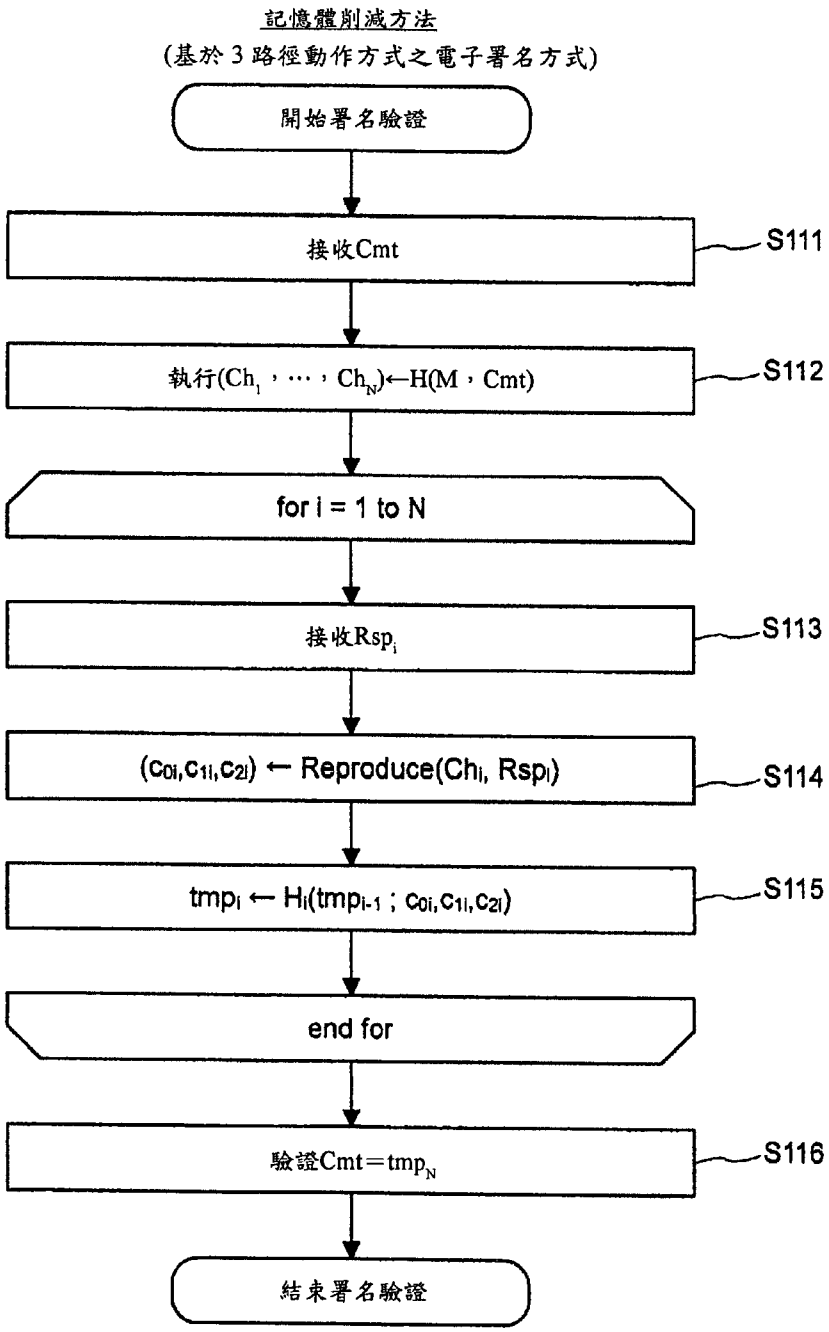
(21)申請案號：101128129 (22)申請日： 中華民國 101 (2012) 年 08 月 03 日
(51)Int. Cl.： *H04L9/14 (2006.01)* *H04L9/28 (2006.01)*
(30)優先權：2011/08/29 日本 2011-185946
(71)申請人：新力股份有限公司 (日本) SONY CORPORATION (JP)
日本
(72)發明人：作本紘一 SAKUMOTO, KOICHI (JP) ；白井太三 SHIRAI, TAIZO (JP) ；桶渡玄良 HIWATARI, HARUNAGA (JP)
(74)代理人：陳長文
申請實體審查：無 申請專利範圍項數：7 項 圖式數：22 共 91 頁

(54)名稱

署名驗證裝置、署名驗證方法、程式及記錄媒體

(57)摘要

本發明提供一種署名驗證裝置，其包括：署名取得部，其取得電子署名，該電子署名包含：第 1 資訊，其係根據多次多元多項式之組 $F=(f_1, ..., f_m)$、署名金鑰 s、及文件 M 而生成者；及複數個第 2 資訊，其係根據上述文件 M、上述多次多元多項式之組 F、及向量 y 而用以驗證上述第 1 資訊乃使用上述署名金鑰 s 生成所必需者；以及署名驗證部，其藉由確認使用上述電子署名中所含之複數個第 2 資訊是否可還原上述第 1 資訊而驗證上述文件 M 之正當性。上述署名驗證部使用每次取得特定數之上述第 2 資訊逐次地還原上述第 1 資訊，並將還原處理中無需之上述第 2 資訊於不再需要之階段刪除。


記憶體削減方法
(基於3路徑動作方式之電子署名方式)
開始署名驗證
接收Cmt
S111
執行(Ch1，…，ChN)←H(M，Cmt)
S112
for i = 1 to N
接收Rspi
S113
(C0i,C1i,C2i) ← Reproduce(Chi, Rspi)
S114
tmpi ← Hi(tmpi-1 ; C0i,C1i,C2i)
S115
end for
驗證Cmt=tmpN
S116
結束署名驗證

(19)中華民國智慧財產局
(12)發明說明書公開本 (11)公開編號：TW 201320700 A1
(43)公開日： 中華民國 102 (2013) 年 05 月 16 日

(21)申請案號：101128129 (22)申請日： 中華民國 101 (2012) 年 08 月 03 日
(51)Int. Cl.： *H04L9/14 (2006.01)* *H04L9/28 (2006.01)*
(30)優先權：2011/08/29 日本 2011-185946
(71)申請人：新力股份有限公司 (日本) SONY CORPORATION (JP)
日本
(72)發明人：作本紘一 SAKUMOTO, KOICHI (JP) ；白井太三 SHIRAI, TAIZO (JP) ；桶渡玄良 HIWATARI, HARUNAGA (JP)
(74)代理人：陳長文
申請實體審查：無 申請專利範圍項數：7 項 圖式數：22 共 91 頁

(54)名稱

署名驗證裝置、署名驗證方法、程式及記錄媒體

(57)摘要

本發明提供一種署名驗證裝置，其包括：署名取得部，其取得電子署名，該電子署名包含：第 1 資訊，其係根據多次多元多項式之組 $F=(f_1, ..., f_m)$、署名金鑰 s、及文件 M 而生成者；及複數個第 2 資訊，其係根據上述文件 M、上述多次多元多項式之組 F、及向量 y 而用以驗證上述第 1 資訊乃使用上述署名金鑰 s 生成所必需者；以及署名驗證部，其藉由確認使用上述電子署名中所含之複數個第 2 資訊是否可還原上述第 1 資訊而驗證上述文件 M 之正當性。上述署名驗證部使用每次取得特定數之上述第 2 資訊逐次地還原上述第 1 資訊，並將還原處理中無需之上述第 2 資訊於不再需要之階段刪除。

201320700

發明專利說明書

（本說明書格式、順序及粗體字，請勿任意更動，※記號部分請勿填寫）

※申請案號：101128129

※申請日：101.8.3　　※IPC分類：H04L 9/14 (2006.01)
H04L 9/28 (2006.01)

一、發明名稱：（中文/英文）

署名驗證裝置、署名驗證方法、程式及記錄媒體

二、中文發明摘要：

本發明提供一種署名驗證裝置，其包括：署名取得部，其取得電子署名，該電子署名包含：第1資訊，其係根據多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰s、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量y而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及署名驗證部，其藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性。上述署名驗證部使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

三、英文發明摘要：

四、指定代表圖：

(一)本案指定代表圖為：第（ 12 ）圖。

(二)本代表圖之元件符號簡單說明：

（無元件符號說明）

五、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

（無）

六、發明說明：

【發明所屬之技術領域】

本技術係關於一種署名驗證裝置、署名驗證方法、程式及記錄媒體。

【先前技術】

隨著資訊處理技術及通訊技術之急速發展，無論是公務文件還是私人文件，文件之電子化均在急速地發展。隨之，大量個人及企業開始對電子文件之安全管理寄予極大之關心。受到此種關心之高漲，人們開始於各方面積極地研究針對電子文件之監聽或偽造等篡改行為的對策。對於電子文件之竊聽，例如，藉由將電子文件暗號化而確保安全性。又，對於電子文件之偽造，例如，藉由利用電子署名而確保安全性。但是，若所利用之暗號或電子署名不具較高之篡改耐性，則無法保證充分之安全性。

電子署名用以特定電子文件之製作者。因此，電子署名應僅可由電子文件之製作者生成。假設，若帶有惡意之第三者可生成相同之電子署名，則該第三者便可完全成為電子文件之製作者。即，會由帶有惡意之第三者偽造電子文件。為防止此種偽造，關於電子署名之安全性始終交織著各種議論。作為目前廣泛利用之電子署名方式，已知有例如RSA署名方式及DSA署名方式等。

RSA署名方式係將「對於較大合成數之素因數分解之困難性(以下，稱為素因數分解問題)」作為安全性之依據。又，DSA署名方式係將「對於離散對數問題之解的導出之

困難性」作為安全性之依據。該等依據起因於不存在利用古典電腦而高效地解決素因數分解問題或離散對數問題之演算法。即，上述困難性係指古典電腦中之計算量上之困難性。然而，人們認為：若使用量子電腦，則會高效地算出對於素因數分解問題或離散對數問題之解答。

目前正在利用之電子署名方式或公鑰認證方式之多數係與RSA署名方式或DSA署名方式同樣地，將安全性之依據置於素因數分解問題或離散對數問題之困難性中。故而，此種電子署名方式或公鑰認證方式於量子電腦實用化之情形時，無法確保其安全性。因此，始終追求一種將安全性之依據置於與素因數分解問題或離散對數問題等可藉由量子電腦容易地解決之問題不同之問題中之新的電子署名方式及公鑰認證方式之實現。作為難以藉由量子電腦容易地解決之問題，例如有多元多項式問題。

作為將安全性之依據置於多元多項式問題中之電子署名方式，已知有基於例如MI(Matsumoto-Imai cryptography，松本-今井密碼法)、HFE(Hidden Field Equation cryptography，隱藏欄位方程式密碼法)、OV(Oil-Vinegar signature scheme，Oil-Vinegar簽章機制)、TTM(Tamed Transformation Method cryptography，馴變換密碼法)之方式。例如，於下述非專利文獻1、2中，揭示有基於HFE之電子署名方式。

[先行技術文獻]

[非專利文獻]

[非專利文獻1] Jacques Patarin Asymmetric Cryptography

with a Hidden Monomial. CRYPTO 1996, pp. 45-60.

[非專利文獻2] Patarin, J., Courtois, N., and Goubin, L. QUARTZ, 128-Bit Long Digital Signatures. In Naccache, D., Ed. Topics in Cryptology-CT-RSA 2001(San Francisco, CA, USA, April 2001), vol. 2020 of Lecture Notes in Computer Science, Springer-Verlag., pp. 282-297.

【發明內容】

[發明所欲解決之問題]

如上所述，多元多項式問題係即便使用量子電腦亦難以解決之被稱為NP困難問題之問題之一例。通常，HFE等所代表之利用多元多項式問題之公鑰認證方式利用編入有特殊之暗門之多次多元聯立方程式。例如，準備關於x_1，...，x_n之多次多元聯立方程式$F(x_1, ..., x_n)=y$與線性轉換A及B，並秘密地管理線性轉換A及B。於該情形時，多次多元聯立方程式F、線性轉換A及B成為暗門。

知曉暗門F、A、B之實體可解開關於x_1，...，x_n之方程式$B(F(A(x_1, ..., x_n)))=y'$。另一方面，暗門F、A、B之實體未知無法解開關於$x_1$，...，$x_n$之方程式$B(F(A(x_1, ..., x_n)))=y'$。藉由利用該結構，而實現以多次多元聯立方程式之解答困難性作為安全性之依據之公鑰認證方式或電子署名方式。

如上所述，為實現此種公鑰認證方式或電子署名方式，必需準備諸如滿足$B(F(A(x_1, ..., x_n)))=y$之類的特殊之多次多元聯立方程式。又，於署名生成時必需解開多次多元

聯立方程式F。因此，可利用之多次多元聯立方程式F僅限於可比較容易地解開者。即，於此種程度之方式中，僅可使用將比較容易解開之3個函數(暗門)B、F、A合成而成之形式的多次多元聯立方程式$B(F(A(x_1, \ldots, x_n)))=y$，因而難以確保充分之安全性。

本技術係鑒於上述情況，意圖提供一種可使用高效地解開之方法(暗門)未知之多次多元聯立方程式以較少之記憶體來實現具有較高安全性之高效之電子署名方式之署名驗證的新型且經過改良之署名驗證裝置、署名驗證方法、程式及記錄媒體而發明。

[解決問題之技術手段]

根據本技術之某觀點，提供一種署名驗證裝置，其包括：署名取得部，其取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及署名驗證部，其藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且上述多元多項式之組F及上述向量y為公鑰，上述署名取得部每次取得特定數之上述第2資訊，上述署名驗證部使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階

段刪除。

又，根據本技術之另一觀點，提供一種署名驗證方法，其包括：取得電子署名之步驟，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性之步驟；且上述多元多項式之組F及上述向量y為公鑰，於上述取得步驟中，每次取得特定數之上述第2資訊，於上述驗證步驟中，使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

又，根據本技術之另一觀點，提供一種程式，其係用以使電腦實現如下功能者：署名取得功能，其係取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係為根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及署名驗證功能，其係藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而

驗證上述文件M之正當性；

且上述多元多項式之組F及上述向量y為公鑰，上述署名取得功能每次取得特定數之上述第2資訊，上述署名驗證功能使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

又，根據本技術之另一觀點，提供一種記錄媒體，其係記錄有用以使電腦實現如下功能之程式且可藉由電腦讀取者：署名取得功能，其係取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及署名驗證功能，其係藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且上述多元多項式之組F及上述向量y為公鑰，上述署名取得功能每次取得特定數之上述第2資訊，上述署名驗證功能使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

[發明之效果]

如以上所說明般，根據本技術，可使用高效地解開之方法(暗門)未知之多次多元聯立方程式以較少之記憶體來實

現具有較高安全性之高效之電子署名方式之署名驗證。

【實施方式】

以下一面參照附圖，一面對本技術之較佳實施形態詳細地進行說明。再者，於本說明書及圖式中，對於具有實質上相同之功能構成之構成要素標註相同之符號，藉此省略重複說明。

[關於說明之順序]

此處，對有關於以下所記載之本技術之實施形態之說明的順序簡單地進行敘述。首先，一面參照圖1，一面對公鑰認證方式之演算法構成進行說明。接著，一面參照圖2，一面對電子署名方式之演算法構成進行說明。繼而，一面參照圖3，一面對n路徑動作之公鑰認證方式進行說明。

其次，一面參照圖4及圖5，一面對3路徑動作之公鑰認證方式之演算法之構成例進行說明。然後，一面參照圖6及圖7，一面對5路徑動作之公鑰認證方式之演算法之構成例進行說明。接著，一面參照圖8及圖9，一面對將3路徑動作及5路徑動作之公鑰認證方式之高效之演算法變形成電子署名方式之演算法之方法進行說明。

繼而，一面參照圖10~圖14，一面對於執行本實施形態之電子署名方式之演算法時削減署名驗證所需之記憶體量之方法進行說明。其次，一面參照圖15~圖19，一面對自2進制隨機數高效地抽選3進制隨機數之方法進行說明。然後，一面參照圖20及圖21，一面對高效地代入多元多項式

之係數之方法進行說明。接著，一面參照圖22，一面對可實現本技術之實施形態之各演算法的資訊處理裝置之硬體構成例進行說明。最後，對本實施形態之技術性思想進行總結，並對自該技術性思想中獲得之作用效果簡單地進行說明。

(說明項目)

1：序言

1-1：公鑰認證方式之演算法

1-2：電子署名方式之演算法

1-3：n路徑動作之公鑰認證方式

2：3路徑動作之公鑰認證方式之演算法之構成

2-1：具體之演算法之構成例

2-2：並行化演算法之構成例

3：5路徑動作之公鑰認證方式之演算法之構成

3-1：具體之演算法之構成例

3-2：並行化演算法之構成例

4：向電子署名方式之變形

4-1：自3路徑動作之公鑰認證方式向電子署名方式之變形

4-2：自5路徑動作之公鑰認證方式向電子署名方式之變形

5：關於署名驗證所需之記憶體量之削減方法

5-1：關於雜湊函數之構造

5-2：面向基於3路徑動作方式之電子署名方式之應

用例

5-3：面向基於5路徑動作方式之電子署名方式之應用例

6：關於自2進制數之隨機數列中抽選3進制數之隨機數列之方法

6-1：抽選方法#1(劃分成2位元)

6-2：抽選方法#2(不劃分)

6-3：抽選方法#3(劃分成k位元)

6-3-1：基本構成

6-3-2：追加抽選方法

7：關於高效地代入多元多項式之係數之方法

7-1：基本約定

7-2：資料之構造化

7-2-1：構造化方法#1

7-2-2：構造化方法#2

7-2-3：構造化方法#3

8：硬體構成例

9：總結

<1：序言>

本實施形態係關於一種將安全性之依據置於對於多次多元聯立方程式之求解問題之困難性中的公鑰認證方式及電子署名方式。其中，本實施形態與HFE電子署名方式等先前方法不同，係關於一種利用並不具備可高效地解開之方法(暗門)之多次多元聯立方程式的公鑰認證方式及電子署

名方式。首先，對公鑰認證方式之演算法、電子署名方式之演算法、及n路徑動作之公鑰認證方式之概要簡單地進行說明。

[1-1：公鑰認證方式之演算法]

首先，一面參照圖1，一面對公鑰認證方式之演算法之概要進行說明。圖1係用以說明公鑰認證方式之演算法之概要之說明圖。

公鑰認證係用以使某人(證明者)可利用公鑰pk及密鑰sk而讓他人(驗證者)認同其為本人。例如，證明者A之公鑰pk_A向驗證者B公開。另一方面，證明者A之密鑰sk_A由證明者A秘密地管理。於公鑰認證之結構中，知曉與公鑰pk_A對應之密鑰sk_A者被認為係證明者A本人。

為利用公鑰認證之結構向驗證者B證明證明者A係證明者A本人，只要經由對話協定，向驗證者B出示證明者A知曉與公鑰pk_A對應之密鑰sk_A之證據即可。而且，於向驗證者B出示證明者A知曉密鑰sk_A之證據，且驗證者B確認完該證據之情形時，證明者A之正當性(其係本人)得到證明。

其中，對於公鑰認證之結構，為保證安全性而要求以下條件。

第1個條件為「令於執行對話協定時由不持有密鑰sk之偽證者使偽證成立之概率無限小」。將該第1個條件成立稱為「健全性」。即，所謂健全性，換而言之係「不會由不持有密鑰sk之偽證者，於對話協定之執行中使偽證以無法

忽視之概率成立之情況」。第2個條件為「即便已執行對話協定，證明者A所擁有之密鑰sk_A之資訊亦絲毫不會向驗證者B洩露」。將該第2個條件成立稱為「零知識性」。

為安全地進行公鑰認證，必需利用具有健全性及零知識性之對話協定。假設，當使用不具健全性及零知識之對話協定進行認證處理之情形時，無可否認存在被偽證之可能性及密鑰之資訊被洩露之可能性，故而即便處理本身成功地完成亦無法證明證明者之正當性。因此，如何保證對話協定之健全性及零知識性變得尤為重要。

(模型)

如圖1所示，於公鑰認證方式之模型中，存在證明者與驗證者2個實體。證明者使用金鑰生成演算法Gen，生成證明者固有之密鑰sk與公鑰pk之組。接著，證明者利用使用金鑰生成演算法Gen而生成之密鑰sk與公鑰pk之組與驗證者執行對話協定。此時，證明者利用證明者演算法P執行對話協定。如上所述，證明者利用證明者演算法P，向驗證者出示於對話協定之中正保有密鑰sk之證據。

另一方面，驗證者利用驗證者演算法V執行對話協定，驗證該證明者是否正保有與證明者始終公開之公鑰對應之密鑰。即，驗證者係驗證證明者是否正保有與公鑰對應之密鑰之實體。如此，公鑰認證方式之模型係由證明者與驗證者2個實體、及金鑰生成演算法Gen、證明者演算法P、驗證者演算法V之3個演算法所構成。

再者，於以下說明中，使用「證明者」「驗證者」之表

現，但該等表現歸根到底係指實體。因而，執行金鑰生成演算法Gen、證明者演算法P之主體係與「證明者」之實體對應之資訊處理裝置。同樣地，執行驗證者演算法V之主體係資訊處理裝置。該等資訊處理裝置之硬體構成例如如圖10所示。即，金鑰生成演算法Gen、證明者演算法P、驗證者演算法V係根據記錄於ROM(Read Only Memory，唯讀記憶體)904、RAM(Random Access Memory，隨機存取記憶體)906、記憶部920、可移除式記錄媒體928等中之程式藉由CPU(Central Processing Unit，中央處理單元)902等而執行。

(金鑰生成演算法Gen)

金鑰生成演算法Gen由證明者利用。金鑰生成演算法Gen係生成證明者所固有之密鑰sk與公鑰pk之組之演算法。藉由金鑰生成演算法Gen而生成之公鑰pk公開。而且，被公開之公鑰pk由驗證者利用。另一方面，藉由金鑰生成演算法Gen而生成之密鑰sk由證明者秘密地管理。而且，由證明者秘密地管理之密鑰sk係用以向驗證者證明證明者正保有與公鑰pk對應之密鑰sk。形式上，金鑰生成演算法Gen作為輸入安全參數1^{λ}(λ為0以上之整數)且輸出密鑰sk與公鑰pk之演算法如下式(1)所表現。

[數1]

$$(sk, pk) \leftarrow Gen(1^{\lambda}) \quad \cdots(1)$$

(證明者演算法P)

證明者演算法P由證明者利用。證明者演算法P係用以向驗證者證明證明者正保有與公鑰pk對應之密鑰sk之演算法。即，證明者演算法P係輸入密鑰sk與公鑰pk，執行對話協定之演算法。

(驗證者演算法V)

驗證者演算法V由驗證者利用。驗證者演算法V係於對話協定之中，驗證證明者是否正保有與公鑰pk對應之密鑰sk之演算法。驗證者演算法V係輸入公鑰pk，根據對話協定之執行結果輸出0或1(1位元)之演算法。再者，驗證者於驗證者演算法V輸出0之情形時判斷證明者為不當者，於輸出1之情形時判斷證明者為正當者。形式上，驗證者演算法V如下式(2)所表現。

[數2]

$$0/1 \leftarrow V(pk) \qquad \cdots(2)$$

如上所述，為實現有意義之公鑰認證，必需使對話協定滿足健全性及零知識性2個條件。然而，為證明證明者正保有密鑰sk，必需使證明者執行取決於密鑰sk之次序，並將其結果通知驗證者，之後使驗證者執行基於該通知內容之驗證。執行取決於密鑰sk之次序係為保證健全性而必需。另一方面，必需使密鑰sk之資訊絲毫不會向驗證者洩漏。因此，為滿足該等必要條件，必需巧妙地設計上述金鑰生成演算法Gen、證明者演算法P、驗證者演算法V。

以上，對公鑰認證方式之演算法之概要進行了說明。

[1-2：電子署名方式之演算法]

其次，一面參照圖2，一面對電子署名方式之演算法之概要進行說明。圖2係用以說明電子署名方式之演算法之概要之說明圖。

與紙質文件不同，無法對某被電子化之資料進行蓋章或記載署名。因此，為證明被電子化之資料之製作者，必需可獲得與於紙質文件上蓋章或記載署名同等之效果之電子性之結構。該結構係電子署名。所謂電子署名係指將僅有資料之製作者知曉之署名資料與資料建立起關聯而提供給收置者，由收置者側驗證該署名資料之結構。

(模型)

如圖2所示，於電子署名方式之模型中，存在署名者及驗證者2個實體。而且，電子署名方式之模型係由金鑰生成演算法Gen、署名生成演算法Sig、署名驗證演算法Ver之3個演算法所構成。

署名者利用金鑰生成演算法Gen生成署名者固有之署名金鑰sk與驗證金鑰pk之組。又，署名者利用署名生成演算法Sig生成賦予於文件M之電子署名σ。即，署名者係對文件M賦予電子署名之實體。另一方面，驗證者利用署名驗證演算法Ver驗證賦予於文件M之電子署名σ。即，驗證者係為確認文件M之製作者是否為署名者而驗證電子署名σ之實體。

再者，於以下說明中，使用「署名者」「驗證者」之表現，但該等表現歸根到底係指實體。因而，執行金鑰生成

演算法Gen、署名生成演算法Sig之主體係與「署名者」之實體對應之資訊處理裝置。同樣地，執行署名驗證演算法Ver之主體係資訊處理裝置。該等資訊處理裝置之硬體構成例如如圖10所示。即，金鑰生成演算法Gen、署名生成演算法Sig、署名驗證演算法Ver係根據記錄於ROM904、RAM906、記憶部920、可移除式記錄媒體928等中之程式藉由CPU902等而執行。

(金鑰生成演算法Gen)

金鑰生成演算法Gen由署名者利用。金鑰生成演算法Gen係生成署名者固有之署名金鑰sk與驗證金鑰pk之組之演算法。藉由金鑰生成演算法Gen而生成之驗證金鑰pk公開。另一方面，藉由金鑰生成演算法Gen而生成之署名金鑰sk由署名者秘密地管理。而且，署名金鑰sk係用於賦予於文件M之電子署名σ之生成。例如，金鑰生成演算法Gen輸入安全參數1^{λ}(λ為0以上之整數)，輸出署名金鑰sk及公鑰pk。於該情形時，金鑰生成演算法Gen形式上可如下式(3)所表現。

[數3]

$$(sk, pk) \leftarrow Gen(1^{\lambda}) \quad \cdots(3)$$

(署名生成演算法Sig)

署名生成演算法Sig由署名者利用。署名生成演算法Sig係生成賦予於文件M之電子署名σ之演算法。署名生成演算法Sig係輸入署名金鑰sk與文件M且輸出電子署名σ之演

算法。該署名生成演算法Sig形式上可如下式(4)所表現。

[數4]

$$\sigma \leftarrow Sig(sk, M) \quad \cdots(4)$$

(署名驗證演算法Ver)

署名驗證演算法Ver由驗證者利用。署名驗證演算法Ver係驗證電子署名σ是否為相對於文件M之正當之電子署名之演算法。署名驗證演算法Ver係輸入署名者之驗證金鑰pk、文件M、電子署名σ且輸出0或1(1位元)之演算法。該署名驗證演算法Ver形式上可如下式(5)所表現。再者，驗證者於署名驗證演算法Ver輸出0之情形時(公鑰pk拒絕文件M與電子署名σ之情形時)判斷電子署名σ不當，於輸出1之情形時(公鑰pk受理文件M與電子署名σ之情形時)判斷電子署名σ正當。

[數5]

$$0/1 \leftarrow Ver(pk, M, \sigma) \quad \cdots(5)$$

以上，對電子署名方式之演算法之概要進行了說明。

[1-3：n路徑動作之公鑰認證方式]

其次，一面參照圖3，一面對n路徑動作之公鑰認證方式進行說明。圖3係用以說明n路徑動作之公鑰認證方式之說明圖。

如上所述，公鑰認證方式係於對話協定之中，向驗證者證明證明者正保有與公鑰pk對應之密鑰sk之認證方式。

又，對話協定必需滿足健全性及零知識性2個條件。因此，如圖3所示，於對話協定之中，證明者及驗證者雙方一面分別執行處理一面進行n次資訊交換。

於n路徑動作之公鑰認證方式之情形時，由證明者使用證明者演算法P執行處理(步驟#1)，且資訊T_1發送至驗證者。接著，由驗證者使用驗證者演算法V執行處理(步驟#2)，且資訊T_2發送至證明者。進而，相對於k=3~n依序進行處理之執行及資訊T_k之發送，最後執行處理(步驟#n+1)。從而，將發送接收N次資訊之方式稱為「n路徑動作」之公鑰認證方式。

以上，對n路徑動作之公鑰認證方式進行了說明。

<2：3路徑動作之公鑰認證方式之演算法之構成>

以下，對3路徑動作之公鑰認證方式之演算法進行說明。再者，於以下說明中，有時將3路徑動作之公鑰認證方式稱為「3路徑動作方式」。

[2-1：具體之演算法之構成例(圖4)]

首先，一面參照圖4，一面對3路徑動作方式之具體之演算法之構成例進行介紹。圖4係用以說明3路徑動作方式之具體之演算法之構成的說明圖。此處，對利用2次多項式之組$(f_1(x), \ldots, f_m(x))$作為公鑰pk之一部分之情形進行研究。其中，2次多項式$f_i(x)$設為如下式(6)所表現者。又，將向量$(x_1, \ldots, x_n)$記作x，將2次多項式之組$(f_1(x), \ldots, f_m(x))$記作多元多項式F(x)。

[數6]

$$f_i(x_1,\cdots,x_n)=\sum_{j,k}a_{ijk}x_jx_k+\sum_j b_{ij}x_j \quad \cdots(6)$$

又，2次多項式之組($f_1(x)$，...，$f_m(x)$)可如下式(7)所表現。又，A_1，...，A_m為n×n矩陣。進而，b_1，...，b_m分別為n×1向量。

[數7]

$$F(x)=\begin{pmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{pmatrix}=\begin{pmatrix} x^TA_1x+b_1^Tx \\ \vdots \\ x^TA_mx+b_m^Tx \end{pmatrix} \quad \cdots(7)$$

若使用該表現，則多元多項式F可如下式(8)及式(9)所表現。該表現成立可自下式(10)容易地得到確認。

[數8]

$$F(x+y)=F(x)+F(y)+G(x,y) \quad \cdots(8)$$

$$G(x,y)=\begin{pmatrix} y^T\left(A_1^T+A_1\right)x \\ \vdots \\ y^T\left(A_m^T+A_m\right)x \end{pmatrix} \quad \cdots(9)$$

$$\begin{aligned} f_l(x,y)&=(x+y)^T A_l(x+y)+b_l^T(x+y) \\ &=x^TA_lx+x^TA_ly+y^TA_lx+y^TA_ly+b_l^Tx+b_l^Ty \\ &=f_l(x)+f_l(y)+x^TA_ly+y^TA_lx \\ &=f_l(x)+f_l(y)+x^T\left(A_l^T\right)^Ty+y^TA_lx \\ &=f_l(x)+f_l(y)+\left(A_l^Tx\right)^Ty+y^TA_lx \\ &=f_l(x)+f_l(y)+y^T\left(A_l^Tx\right)+y^TA_lx \\ &=f_l(x)+f_l(y)+y^T\left(A_l^T+A_l\right)x \end{aligned} \quad \cdots(10)$$

當以此方式將F(x+y)分成取決於x之第1部分、取決於y之第2部分、取決於x及y兩者之第3部分時，與第3部分對應之項G(x，y)相對於x及y成雙線性。以下，有時將項

G(x，y)稱為雙線性項。若利用該性質，則可構建高效之演算法。

例如，使用向量$t_0 \in K^n$、$e_0 \in K^m$，將用於多元多項式F(x+r)之遮蔽之多元多項式$F_1(x)$表現為$F_1(x)=G(x，t_0)+e_0$。於該情形時，多元多項式$F(x+r_0)$與G(x)之和如下式(11)所表現。此處，若設定$t_1=r_0+t_0$、$e_1=F(r_0)+e_0$，則多元多項式$F_2(x)=F(x+r_0)+F_1(x)$可藉由向量$t_1 \in K^n$、$e_1 \in K^m$表現。因此，若設定為$F_1(x)=G(x，t_0)+e_0$，則使用K^n上之向量及K^m上之向量可表現F_1及F_2，從而可實現通訊所需之資料大小較小之高效之演算法。

[數9]

$$\begin{aligned} &F(x+r_0)+F_1(x) \\ &=F(x)+F(r_0)+G(x,r_0)+G(x,t_0)+e_0 \\ &=F(x)+G(x,r_0+t_0)+F(r_0)+e_0 \qquad \cdots(11) \end{aligned}$$

再者，不會自F_2(或F_1)洩漏絲毫關於r_0之資訊。例如，即便提供e_1及t_1(或e_0及t_0)，只要e_0及t_0(或e_1及t_1)未知，便絲毫無法知曉r_0之資訊。從而，保證零知識性。以下，對根據上述邏輯而構建之3路徑動作方式之演算法進行說明。此處所說明之3路徑動作方式之演算法係由如下之金鑰生成演算法Gen、證明者演算法P、驗證者演算法V所構成。

(金鑰生成演算法Gen)

金鑰生成演算法Gen生成定義於環K上之m個多元多項式$f_1(x_1，\ldots，x_n)，\ldots，f_m(x_1，\ldots，x_n)$、及向量$s=(s_1，\ldots，$

$s_n) \in K^n$。其次，金鑰生成演算法Gen計算$y=(y_1, \ldots, y_m) \leftarrow (f_1(s), \ldots, f_m(s))$。然後，金鑰生成演算法Gen將$(f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n), y)$設定為公鑰pk，將s設定為密鑰。

(證明者演算法P、驗證者演算法V)

以下，一面參照圖4，一面對於對話協定之中證明者演算法P所執行之處理及驗證者演算法V所執行之處理進行說明。於該對話協定之中，證明者絲毫未將密鑰s之資訊向驗證者洩漏，而將「自身知曉滿足$y=F(s)$之s之情況」向驗證者出示。另一方面，驗證者驗證證明者是否知曉滿足$y=F(s)$之s。再者，公鑰pk設定為向驗證者公開者。又，密鑰s設定為由證明者秘密地管理者。以下，沿著圖4所示之流程圖展開說明。

步驟#1：

如圖4所示，首先，證明者演算法P隨機地生成向量r_0，$t_0 \in K^n$及$e_0 \in K^m$。接著，證明者演算法P計算$r_1 \leftarrow s-r_0$。該計算相當於藉由向量r_0遮蔽密鑰s之操作。進而，證明者演算法P計算$t_1 \leftarrow r_0-t_0$。繼而，證明者演算法P計算$e_1 \leftarrow F(r_0)-e_0$。

步驟#1(繼續)：

其次，證明者演算法P計算$c_0 \leftarrow H(r_1, G(t_0, r_1)+e_0)$。接著，證明者演算法P計算$c_1 \leftarrow H(t_0, e_0)$。繼而，證明者演算法P計算$c_2 \leftarrow H(t_1, e_1)$。於步驟#1中生成之訊息$(c_0, c_1, c_2)$傳送至驗證者演算法V。

步驟#2：

收到訊息(c_0，c_1，c_2)之驗證者演算法V對利用3個驗證圖案中之哪個驗證圖案進行選擇。例如，驗證者演算法V自表示驗證圖案之種類之3個數值{0，1，2}之中選擇1個數值，並將所選擇之數值設定為要求Ch。該要求Ch傳送至證明者演算法P。

步驟#3：

收到要求Ch之證明者演算法P根據所收到之要求Ch生成向驗證者演算法V發送之回答Rsp。於Ch=0之情形時，證明者演算法P生成回答Rsp=(r_0，t_1，e_1)。於Ch=1之情形時，證明者演算法P生成回答Rsp=(r_1，t_0，e_0)。於Ch=2之情形時，證明者演算法P生成回答Rsp=(r_1，t_1，e_1)。於步驟#3中生成之回答Rsp傳送至驗證者演算法V。

步驟#4：

收到回答Rsp之驗證者演算法V利用所收到之回答Rsp執行以下驗證處理。

於Ch=0之情形時，驗證者演算法V驗證$c_1=H(r_0-t_1$，$F(r_0)-e_1)$之等號是否成立。進而，驗證者演算法V驗證$c_2=H(t_1$，$e_1)$之等號是否成立。驗證者演算法V於該等驗證全部成功之情形時輸出表示認證成功之值1，於驗證中發生失敗之情形時輸出表示認證失敗之值0。

於Ch=1之情形時，驗證者演算法V驗證$c_0=H(r_1$，$G(t_0$，$r_1)+e_0)$之等號是否成立。進而，驗證者演算法V驗證$c_1=H(t_0$，$e_0)$之等號是否成立。驗證者演算法V於該等驗證

全部成功之情形時輸出表示認證成功之值1，於驗證中發生失敗之情形時輸出表示認證失敗之值0。

於Ch=2之情形時，驗證者演算法V驗證$c_0=H(r_1, y-F(r_1)-G(t_1, r_1)-e_1)$之等號是否成立。進而，驗證者演算法V驗證$c_2=H(t_1, e_1)$之等號是否成立。驗證者演算法V於該等驗證全部成功之情形時輸出表示認證成功之值1，於驗證中存在失敗之情形時輸出表示認證失敗之值0。

以上，對3路徑動作方式之高效之演算法之構成例進行了說明。

[2-2：並行化演算法之構成例(圖5)]

其次，一面參照圖5，一面對將圖4所示之3路徑動作方式之演算法並行化之方法進行說明。再者，關於金鑰生成演算法Gen之構成省略說明。

另外，若應用上述對話協定，則可將偽證成功之概率抑制為2/3以下。從而，若執行2次該對話協定，則可將偽證成功之概率抑制為$(2/3)^2$以下。進而，若執行N次該對話協定，則偽證成功之概率成為$(2/3)^N$，只要將N設為充分大之數(例如，N=140)，則偽證成功之概率將縮小至可忽視之程度。

作為執行複數次對話協定之方法，例如可考慮逐次地重複複數次訊息、要求、回答之交換之串列方法、及以1次交換進行複數次之訊息、要求、回答之交換之並行方法。進而，亦可考慮將串列方法與並行方法組合而成之複合型方法。此處，一面參照圖5，一面對並行執行3路徑動作方

式之上述對話協定之演算法(以下稱為並行化演算法)進行說明。

步驟#1：

如圖5所示，首先，證明者演算法P相對於i=1~N執行以下處理(1)~處理(6)。

處理(1)：證明者演算法P隨機地生成向量r_{0i}，$t_{0i}\in K^n$及$e_{0i}\in K^m$。

處理(2)：證明者演算法P計算$r_{1i}\leftarrow s-r_{0i}$。該計算相當於藉由向量$r_{0i}$遮蔽密鑰s之操作。進而，證明者演算法P計算$t_{1i}\leftarrow r_{0i}+t_{0i}$。

處理(3)：證明者演算法P計算$e_{1i}\leftarrow F(r_{0i})-e_{0i}$。

處理(4)：證明者演算法P計算$c_{0i}\leftarrow H(r_{1i}, G(r_{1i}, t_{0i})+e_{0i})$。

處理(5)：證明者演算法P計算$c_{1i}\leftarrow H(t_{0i}, e_{0i})$。

處理(6)：證明者演算法P計算$c_{2i}\leftarrow H(t_{1i}, e_{1i})$。

步驟#1(繼續)

於針對i=1~N執行上述處理(1)~處理(6)之後，證明者演算法P計算$Cmt\leftarrow H(c_{01}, c_{11}, c_{21}, \ldots, c_{0N}, c_{1N}, c_{2N})$。將於步驟#1中生成之雜湊值Cmt傳送至驗證者演算法V。從而，於將訊息$(c_{01}, c_{11}, c_{21}, \ldots, c_{0N}, c_{1N}, c_{2N})$轉換成雜湊值之後傳送至驗證者演算法V，藉此可削減通訊量。

步驟#2：

收到雜湊值Cmt之驗證者演算法V針對i=1~N各者，自3個驗證圖案中選擇利用哪一驗證圖案。例如，驗證者演算

法V針對i=1~N各者，自表示驗證圖案之種類之3個數值{0，1，2}中選擇1個數值，並將所選擇之數值設定為要求Ch_i。要求Ch_1，...，Ch_N傳送至證明者演算法P。

步驟#3：

收到要求Ch_1，...，Ch_N之證明者演算法P根據所收到之要求Ch_1，...，Ch_N各者而生成送向驗證者演算法V之回答Rsp_1，...，Rsp_N。於$Ch_i=0$之情形時，證明者演算法P生成$Rsp_i=(r_{0i}, t_{1i}, e_{1i}, c_{0i})$。於$Ch_i=1$之情形時，證明者演算法P生成$Rsp_i=(r_{1i}, t_{0i}, e_{0i}, c_{2i})$。於$Ch_i=2$之情形時，證明者演算法P生成$Rsp_i=(r_{1i}, t_{1i}, e_{1i}, c_{1i})$。

於步驟#3中生成之回答Rsp_1，...，Rsp_N傳送至驗證者演算法V。

步驟#4：

收到回答Rsp_1，...，Rsp_N之驗證者演算法V利用所收到之回答Rsp_1，...，Rsp_N相對於i=1~N而執行以下處理(1)~處理(3)。其中，驗證者演算法V於$Ch_i=0$之情形時執行處理(1)，於$Ch_i=1$之情形時執行處理(2)，於$Ch_i=2$之情形時執行處理(3)。

處理(1)：於$Ch_i=0$之情形時，驗證者演算法V自Rsp_i中取出$(r_{0i}, t_{1i}, e_{1i}, c_{0i})$。接著，驗證者演算法V計算$c_{1i}=H(r_{0i}-t_{1i}, F(r_{0i})-e_{1i})$。進而，驗證者演算法V計算$c_{2i}=H(t_{1i}, e_{1i})$。然後，驗證者演算法V保持$(c_{0i}, c_{1i}, c_{2i})$。

處理(2)：於$Ch_i=1$之情形時，驗證者演算法V自Rsp_i中取

出 $(r_{1i}, t_{0i}, e_{0i}, c_{2i})$。接著，驗證者演算法V計算 $c_{0i}=H(r_{1i}, G(t_{0i}, r_{1i})+e_{0i})$。進而，驗證者演算法V計算 $c_{1i}=H(t_{0i}, e_{0i})$。然後，驗證者演算法V保持 (c_{0i}, c_{1i}, c_{2i})。

處理(3)：於 $Ch_i=2$ 之情形時，驗證者演算法V自 Rsp_i 中取出 $(r_{1i}, t_{1i}, e_{1i}, c_{1i})$。接著，驗證者演算法V計算 $c_{0i}=H(r_{1i}, y-F(r_{1i})-G(t_{1i}, r_{1i})-e_{1i})$。進而，驗證者演算法V計算 $c_{2i}=H(t_{1i}, e_{1i})$。然後，驗證者演算法V保持 (c_{0i}, c_{1i}, c_{2i})。

於相對於 $i=1\sim N$ 而執行處理(1)~處理(3)之後，驗證者演算法V驗證 $Cmt=H(c_{01}, c_{11}, c_{21}, \ldots, c_{0N}, c_{1N}, c_{2N})$ 之等號是否成立。驗證者演算法V於該驗證成功之情形時輸出表示認證成功之值1，於驗證失敗之情形時輸出表示認證失敗之值0。

以上，對3路徑動作方式之高效之並行化演算法之構成例進行了說明。

<3：5路徑動作之公鑰認證方式之演算法之構成>

其次，對5路徑動作之公鑰認證方式之演算法進行說明。再者，於以下說明中，有時將5路徑動作之公鑰認證方式稱為「5路徑動作方式」。

於3路徑動作方式之情形時每1次對話協定之偽證概率為2/3，但於5路徑動作方式之情形時每1次對話協定之偽證概率為1/2+1/q。其中，q係所要利用之環之位數。從而，

於環之位數充分大之情形時，5路徑動作方式可降低每1次之偽證概率，由此可以較少之對話協定之執行次數，使偽證概率充分小。

例如，當欲使偽證概率為$1/2^n$以下之情形時，於3路徑動作方式中，必需執行$n/(\log 3-1)=1.701n$次以上對話協定。另一方面，當欲使偽證概率為$1/2^n$以下之情形時，於5路徑動作方式中，必需執行$n/(1-\log(1+1/q))$次以上對話協定。從而，若設$q=24$，則就實現相同安全級別所需之通訊量而言，與3路徑動作方式相比，5路徑動作方式較少。

[3-1：具體之演算法之構成例(圖6)]

首先，一面參照圖6，一面對5路徑動作方式之具體之演算法之構成例進行介紹。圖6係用以說明5路徑動作方式之具體之演算法之構成的說明圖。此處，對利用2次多項式之組$(f_1(x)，\ldots，f_m(x))$作為公鑰pk之一部分之情形進行研究。其中，2次多項式$f_i(x)$設為如上式(6)所表現者。又，將向量$(x_1，\ldots，x_n)$記作x，將2次多項式之組$(f_1(x)，\ldots，f_m(x))$記作多元多項式$F(x)$。

與3路徑動作方式之演算法同樣地，使用2個向量$t_0 \in K^n$、$e_0 \in K^m$，以$F_1(x)=G(x、t_0)+e_0$之方式表現用以遮蔽多元多項式$F(x+r_0)$之多元多項式$F_1(x)$。若使用該表現，則關於多元多項式$F(x+r_0)$，可獲得下式(12)中所表現之關係。

[數10]

$$Ch_A \cdot F(x+r_0)+F_1(x)$$
$$=Ch_A \cdot F(x)+Ch_A \cdot F(r_0)+Ch_A \cdot G(x,r_0)+G(x,t_0)+e_0$$
$$=Ch_A \cdot F(x)+G(x,Ch_A \cdot r_0+t_0)+Ch_A \cdot F(r_0)+e_0 \quad \cdots(12)$$

因此，若設$t_1=Ch_A \cdot r_0+t_0$、$e_1=Ch_A \cdot F(r_0)+e_0$，則遮蔽後之多元多項式$F_2(x)=Ch_A \cdot F(x+r_0)+F_1(x)$亦可藉由2個向量$t_1 \in K^n$、$e_1 \in K^m$表現。自該等理由，若設定為$F_1(x)=G(x, t_0)+e_0$，則使用$K^n$上之向量及$K^m$上之向量可表現$F_1$及$F_2$，從而可實現通訊所需之資料大小較小之高效之演算法。

再者，不會自F_2(或F_1)洩露絲毫關於r_0之資訊。例如，即便提供e_1及t_1(或e_0及t_0)，只要e_0及t_0(或e_1及t_1)未知，便絲毫無法知曉r_0之資訊。從而，保證零知識性。以下，對根據上述邏輯而構建之5路徑動作方式之演算法進行說明。此處所說明之5路徑動作方式之演算法係由如下之金鑰生成演算法Gen、證明者演算法P、驗證者演算法V所構成。

(金鑰生成演算法Gen)

金鑰生成演算法Gen生成定義於環K上之多元多項式$f_1(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n)$、及向量$s=(s_1, \ldots, s_n) \in K^n$。其次，金鑰生成演算法Gen計算$y=(y_1, \ldots, y_m) \leftarrow (f_1(s), \ldots, f_m(s))$。而且，金鑰生成演算法Gen將$(f_1, \ldots, f_m, y)$設定為公鑰pk，將s設定為密鑰。再者，以下，將向量$(x_1, \ldots, x_n)$記作x，將多元多項式之組$(f_1(x), \ldots, f_m(x))$記作F(x)。

(證明者演算法P、驗證者演算法V)

以下，一面參照圖6，一面對於對話協定之中由證明者演算法P及驗證者演算法V執行之處理進行說明。於該對話協定之中，證明者絲毫未將密鑰s之資訊向驗證者洩漏，而將「自身知曉滿足y=F(s)之s之情況」向驗證者出示。另一方面，驗證者驗證證明者是否知曉滿足y=F(s)之s。再者，公鑰pk設定為向驗證者公開者。又，密鑰sk設定為由證明者秘密地管理者。以下，沿著圖6所示之流程圖展開說明。

步驟#1：

如圖6所示，首先，證明者演算法P隨機地生成向量$r_0 \in K^n$、$t_0 \in K^n$、$e_0 \in K^m$。接著，證明者演算法P計算$r_1 \leftarrow s-r_0$。該計算相當於藉由向量r_0遮蔽密鑰s之操作。繼而，證明者演算法P生成向量r_0，t_0，e_0之雜湊值c_0。即，證明者演算法P計算$c_0 \leftarrow H(r_0, t_0, e_0)$。然後，證明者演算法P生成$G(t_0, r_1)+e_0$及$r_1$之雜湊值$c_1$。即，證明者演算法P計算$c_0 \leftarrow H(r_1, G(t_0, r_1)+e_0)$。於步驟#1中生成之訊息$(c_0, c_1)$傳送至驗證者演算法V。

步驟#2：

收到訊息(c_0, c_1)之驗證者演算法V自q位所存在之環K之源中隨機地選擇1個數Ch_A，並將所選擇之數Ch_A傳送至證明者演算法P。

步驟#3：

收到數Ch_A之證明者演算法P計算$t_1 \leftarrow Ch_A \cdot r_0 - t_0$。進而，

證明者演算法P計算$e_1 \leftarrow Ch_A \cdot F(r_0) - e_0$。然後，證明者演算法P將$t_1$及$e_1$傳送至驗證者演算法V。

步驟#4：

收到t_1及e_1之驗證者演算法V對利用2個驗證圖案之中之哪個驗證圖案進行選擇。例如，驗證者演算法V自表示驗證圖案之種類之2個數值{0，1}之中選擇1個數值，並將所選擇之數值設定為要求Ch_B。該要求Ch_B傳送至證明者演算法P。

步驟#5：

收到要求Ch_B之證明者演算法P根據所收到之要求Ch_B生成返送向驗證者演算法V之回答Rsp。於$Ch_B=0$之情形時，證明者演算法P生成回答$Rsp=r_0$。於$Ch_B=1$之情形時，證明者演算法P生成回答$Rsp=r_1$。於步驟#5中生成之回答Rsp傳送至驗證者演算法V。

步驟#6：

收到回答Rsp之驗證者演算法V利用所收到之回答Rsp執行以下驗證處理。

於$Ch_B=0$之情形時，驗證者演算法V執行$r_0 \leftarrow Rsp$。然後，驗證者演算法V驗證$c_0=H(r_0, Ch_A \cdot r_0 - t_1, Ch_A \cdot F(r_0) - e_1)$之等號是否成立。驗證者演算法V於該驗證成功之情形時輸出表示認證成功之值1，於驗證中存在失敗之情形時輸出表示認證失敗之值0。

於$Ch_B=1$之情形時，驗證者演算法V執行$r_1 \leftarrow Rsp$。然後，驗證者演算法V驗證$c_1=H_1(r_1, Ch_A \cdot (y - F(r_1)) - G(t_1, r_1) -$

e_1)之等號是否成立。驗證者演算法V於該驗證成功之情形時輸出表示認證成功之值1，於驗證中存在失敗之情形時輸出表示認證失敗之值0。

以上，對5路徑動作方式之高效之演算法之構成例進行了說明。

[3-2：並行化演算法之構成例(圖7)]

其次，一面參照圖7，一面對將圖6所示之5路徑動作方式之演算法並行化之方法進行說明。再者，關於金鑰生成演算法Gen之構成省略說明。

如上文所述，若應用5路徑動作方式之對話協定，則可將偽證成功之概率抑制為(1/2+1/q)以下。從而，若執行2次該對話協定，則可將偽證成功之概率抑制為$(1/2+1/q)^2$以下。進而，若執行N次該對話協定，則偽證成功之概率為$(1/2+1/q)^N$，只要使N為充分大之數(例如，N=80)，偽證成功之概率便會縮小為可忽視之程度。

作為執行複數次對話協定之方法，可考慮例如：逐次地重複複數次訊息、要求、回答之交換之串列方法、及於1次交換中進行複數次之訊息、要求、回答之交換之並行方法。進而，亦可考慮將串列方法與並行方法組合而成之複合型方法。此處，對並行執行5路徑動作方式之上述對話協定之演算法(以下，稱為並行化演算法)進行說明。

步驟#1：

如圖7所示，首先，證明者演算法P相對於i=1~N執行處理(1)~處理(4)。

處理(1)：證明者演算法P隨機地生成向量r_{0i}，$t_{0i}\in K^n$及$e_{0i}\in K^m$。

處理(2)：證明者演算法P計算$r_{1i}\leftarrow s-r_{0i}$。該計算相當於藉由向量$r_{0i}$遮蔽密鑰s之操作。

處理(3)：證明者演算法P計算$c_{0i}\leftarrow H(r_{0i}, t_{0i}, e_{0i})$。

處理(4)：證明者演算法P計算$c_{1i}\leftarrow H(r_{1i}, G(t_{0i}, r_{1i})+e_{0i})$。

於相對於i=1~N執行處理(1)~處理(4)之後，證明者演算法P執行雜湊值$Cmt\leftarrow H(c_{01}, c_{11}, \ldots, c_{0N}, c_{1N})$。然後，於步驟#1中生成之雜湊值Cmt傳送至驗證者演算法V。

步驟#2：

收到雜湊值Cmt之驗證者演算法V相對於i=1~N各者，自q位所存在之環K之源中隨機地選擇1個數Ch_{Ai}，並將所選擇之數Ch_{Ai}(i=1~N)傳送至證明者演算法P。

步驟#3：

收到數Ch_{Ai}(i=1~N)之證明者演算法P相對於i=1~N各者，計算$t_{1i}\leftarrow Ch_{Ai}\cdot r_{0i}-t_{0i}$。進而，證明者演算法P相對於i=1~N各者，計算$e_{1i}\leftarrow Ch_{Ai}\cdot F(r_{0i})-e_{0i}$。接著，證明者演算法P計算雜湊值$d\leftarrow H(t_{11}, e_{11}, \ldots, t_{1N}, e_{1N})$。然後，證明者演算法P將雜湊值d傳送至驗證者演算法V。

步驟#4：

收到雜湊值d之驗證者演算法V相對於i=1~N各者，對利用2個驗證圖案之中之哪個驗證圖案進行選擇。例如，驗證者演算法V自表示驗證圖案之種類之2個數值{0，1}之中

選擇1個數值，並將所選擇之數值設定為要求Ch_{Bi}。要求Ch_{Bi}(i=1~N)傳送至證明者演算法P。

步驟#5：

收到要求Ch_{Bi}(i=1~N)之證明者演算法P相對於i=1~N，根據所收到之要求Ch_{Bi}生成返送向驗證者演算法V之回答Rsp_i。於Ch_{Bi}=0之情形時，證明者演算法P生成回答$Rsp_i=(r_{0i}, t_{0i}, e_{0i}, c_{1i})$。於$Ch_{Bi}$=1之情形時，證明者演算法P生成回答$Rsp_i=(r_{1i}, t_{1i}, e_{1i}, c_{0i})$。於步驟#5中生成之回答$Rsp_i$(i=1~N)傳送至驗證者演算法V。

步驟#6：

收到回答Rsp_i(i=1~N)之驗證者演算法V利用所收到之回答Rsp_i(i=1~N)執行以下處理(1)及處理(2)。

處理(1)：於Ch_{Bi}=0之情形時，驗證者演算法V執行$(r_{0i}, t_{0i}, e_{0i}, c_{1i})\leftarrow Rsp_i$。然後，驗證者演算法V計算$c_{0i}=H(r_{0i}, t_{0i}, e_{0i})$。進而，驗證者演算法V計算$t_{1i}\leftarrow Ch_{Ai}\cdot r_{0i}+t_{0i}$、及$e_{1i}\leftarrow Ch_{Ai}\cdot F(r_{0i})-e_{0i}$。然後，驗證者演算法V保持$(c_{0i}, c_{1i}, t_{1i}, e_{1i})$。

處理(2)：於Ch_{Bi}=1之情形時，驗證者演算法V執行$(r_{1i}, t_{1i}, e_{1i}, c_{0i})\leftarrow Rsp_i$。然後，驗證者演算法V計算$c_{1i}=H(r_{1i}, Ch_{Ai}\cdot(y-F(r_{1i}))-G(t_{1i}, r_{1i})-e_{1i})$。然後，驗證者演算法V保持$(c_{0i}, c_{1i}, t_{1i}, e_{1i})$。

於相對於i=1~N執行處理(1)及處理(2)之後，驗證者演算法V驗證$Cmt=H(c_{01}, c_{11}, \ldots, c_{0N}, c_{1N})$之等號是否成立。進而，驗證者演算法V驗證$d=H(t_{11}, e_{11}, \ldots, t_{1N}, e_{1N})$之

等號是否成立。然後，驗證者演算法V於該等驗證全部成功之情形時輸出表示認證成功之值1，於驗證中存在失敗之情形時輸出表示認證失敗之值0。

以上，對5路徑動作方式之高效之並行化演算法之構成例進行了說明。

<4：向電子署名方式之變形>

其次，對使上述公鑰認證方式向電子署名方式變形之方法進行介紹。

容易理解：若使公鑰認證方式之模型中之證明者與電子署名方式中之署名者對應，則於僅證明者認同驗證者之方面與電子署名方式之模型近似。基於此種想法，對使上述公鑰認證方式向電子署名方式變形之方法進行說明。

[4-1：自3路徑動作之公鑰認證方式向電子署名方式之變形(圖8)]

首先，對自3路徑動作之公鑰認證方式向電子署名方式之變形進行說明。

如圖8所示，3路徑動作方式之高效之演算法(例如，參照圖5)係藉由3次對話及4個步驟#1~步驟#4而表現。

步驟#1包括：處理(1)，其相對於i=1~N，生成$a_i=(r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i}, c_{2i})$；及處理(2)，其計算$Cmt \leftarrow H(c_{01}, c_{11}, c_{21}, \ldots, c_{0N}, c_{1N}, c_{2N})$。於步驟#1中藉由證明者演算法P而生成之Cmt向驗證者演算法V傳送。

步驟#2包括選擇$Ch_1, \ldots, Ch_N$之處理。於步驟#2中藉由驗證者演算法V而選擇之$Ch_1, \ldots, Ch_N$向證明者演算法P傳

送。

步驟#3包括使用Ch_1，...，Ch_N及a_1，...，a_N生成Rsp_1，...，Rsp_N之處理。將該處理表現為$Rsp_i \leftarrow Select(Ch_i, a_i)$。於步驟#3中藉由證明者演算法P而生成之$Rsp_1$，...，$Rsp_N$向驗證者演算法V傳送。

步驟#4包括：處理(1)，其使用Ch_1，...，Ch_N及Rsp_1，...，Rsp_N，再生c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N}；及處理(2)，其使用再生之c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N}驗證$Cmt=H(c_{01}, c_{11}, c_{21}, ..., c_{0N}, c_{1N}, c_{2N})$。

藉由上述步驟#1~步驟#4而表現之公鑰認證方式之演算法變形成如圖8所示之署名生成演算法Sig及署名驗證演算法Ver。

(署名生成演算法Sig)

首先，對署名生成演算法Sig之構成進行敘述。署名生成演算法Sig包括以下處理(1)~處理(5)。

處理(1)：署名生成演算法Sig生成$a_i=(r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i}, c_{2i})$。

處理(2)：署名生成演算法Sig計算$Cmt \leftarrow H(c_{01}, c_{11}, c_{21}, ..., c_{0N}, c_{1N}, c_{2N})$。

處理(3)：署名生成演算法Sig計算$(Ch_1, ..., Ch_N) \leftarrow H(M, Cmt)$。該M係賦予署名之文件。

處理(4)：署名生成演算法Sig計算$Rsp_i \leftarrow Select(Ch_i, a_i)$。

處理(5)：署名生成演算法Sig將(Cmt，Rsp_1，...，Rsp_N)

設定為署名。

(署名驗證演算法Ver)

其次，對署名驗證演算法Ver之構成進行敘述。署名驗證演算法Ver包括以下處理(1)~處理(3)。

處理(1)：署名驗證演算法Ver計算(Ch_1，...，Ch_N)←H(M，Cmt)。

處理(2)：署名驗證演算法Ver使用Ch_1，...，Ch_N及Rsp_1，...，Rsp_N生成c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N}。

處理(3)：署名驗證演算法Ver使用再生之c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N}驗證Cmt=H(c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N})。

如以上所說明，使公鑰認證方式之模型中之證明者與電子署名方式中之署名者對應，藉此可使公鑰認證方式之演算法向電子署名方式之演算法變形。

[4-2：自5路徑動作之公鑰認證方式向電子署名方式之變形(圖9)]

其次，對自5路徑動作之公鑰認證方式向電子署名方式之變形進行說明。

如圖9所示，5路徑動作方式之高效之演算法(例如，參照圖7)係藉由5次對話及6個步驟#1~步驟#6而表現。

步驟#1包括：處理(1)，其相對於i=1~N，生成a_i=(r_{0i}，t_{0i}，e_{0i}，r_{1i}，t_{1i}，e_{1i}，c_{0i}，c_{1i})；及處理(2)，其計算Cmt←H(c_{01}，c_{11}，...，c_{0N}，c_{1N})。於步驟#1中藉由證明者演算法P而生成之Cmt向驗證者演算法V傳送。

步驟#2包括選擇Ch_{A1}，...，Ch_{AN}之處理。於步驟#2中藉由驗證者演算法V而選擇之Ch_{A1}，...，Ch_{AN}向證明者演算法P傳送。

步驟#3包括：相對於i=1~N生成$b_i=(t_{1i}, e_{1i})$之處理、及生成$d=H(t_{11}, e_{11}, ..., t_{1N}, e_{1N})$之處理。於步驟#3中藉由證明者演算法P而生成之d向驗證者演算法V傳送。

步驟#4包括選擇Ch_{B1}，...，Ch_{BN}之處理。於步驟#4中藉由驗證者演算法V而選擇之Ch_{B1}，...，Ch_{BN}向證明者演算法P傳送。

步驟#5包括使用Ch_{B1}，...，Ch_{BN}、a_1，...，a_N、b_1，...，b_N生成Rsp_1，...，Rsp_N之處理。將該處理表現為$Rsp_i \leftarrow Select(Ch_{Bi}, a_i, b_i)$。於步驟#5中藉由證明者演算法P而生成之$Rsp_1$，...，$Rsp_N$向驗證者演算法V傳送。

步驟#6包括：使用Ch_{A1}，...，Ch_{AN}，Ch_{B1}，...，Ch_{BN}，Rsp_1，...，Rsp_N再生c_{01}，c_{11}，...，c_{0N}，c_{1N}，t_{11}，e_{11}，...，t_{1N}，e_{1N}之處理、使用再生之c_{01}，c_{11}，...，c_{0N}，c_{1N}驗證$Cmt=H(c_{01}, c_{11}, ..., c_{0N}, c_{1N})$之處理、及驗證$d=H(t_{11}, e_{11}, ..., t_{1N}, e_{1N})$之處理。

藉由上述步驟#1~步驟#6而表現之公鑰認證方式之演算法變形成如圖9所示之署名生成演算法Sig及署名驗證演算法Ver。

(署名生成演算法Sig)

首先，對署名生成演算法Sig之構成進行敘述。署名生成演算法Sig包括以下處理(1)~處理(7)。

處理(1)：署名生成演算法Sig生成$a_i=(r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i})$。

處理(2)：署名生成演算法Sig計算$Cmt \leftarrow H(c_{01}, c_{11}, \ldots, c_{0N}, c_{1N})$。

處理(3)：署名生成演算法Sig計算$(Ch_{A1}, \ldots, Ch_{AN}) \leftarrow H(M, Cmt)$。該M係賦予署名之文件。

處理(4)：署名生成演算法Sig相對於i=1~N，生成$b_i=(t_{1i}, e_{1i})$。進而，署名生成演算法Sig算出$d=H(t_{11}, e_{11}, \ldots, t_{1N}, e_{1N})$。

處理(5)：署名生成演算法Sig計算$(Ch_{B1}, \ldots, Ch_{BN}) \leftarrow H(M, Cmt, Ch_{A1}, \ldots, Ch_{AN}, d)$。再者，亦可$(Ch_{B1}, \ldots, Ch_{BN}) \leftarrow H(Ch_{A1}, \ldots, Ch_{AN}, d)$地變形。

處理(6)：署名生成演算法Sig計算$Rsp_i \leftarrow Select(Ch_{Bi}, a_i, b_i)$。

處理(7)：署名生成演算法Sig將$(Cmt, d, Rsp_1, \ldots, Rsp_N)$設定為電子署名。

(署名驗證演算法Ver)

其次，對署名驗證演算法Ver之構成進行敘述。署名驗證演算法Ver包括以下處理(1)~處理(4)。

處理(1)：署名驗證演算法Ver計算$(Ch_{A1}, \ldots, Ch_{AN}) \leftarrow H(M, Cmt)$。

處理(2)：署名驗證演算法Ver計算$(Ch_{B1}, \ldots, Ch_{BN}) \leftarrow H(M, Cmt, Ch_{A1}, \ldots, Ch_{AN}, d)$。再者，於署名驗證演算法Ver所執行之處理(5)中，當$(Ch_{B1}, \ldots,$

Ch_{BN})←H(Ch_{A1}，...，Ch_{AN}，d)地變形之情形時，署名驗證演算法Ver計算(Ch_{B1}，...，Ch_{BN})←H(Ch_{A1}，...，Ch_{AN}，d)。

處理(3)：署名驗證演算法Ver使用Ch_{A1}，...，Ch_{AN}，Ch_{B1}，...，Ch_{BN}，Rsp_1，...，Rsp_N生成t_{11}，e_{11}，...，t_{1N}，e_{1N}，c_{01}，c_{11}，...，c_{0N}，c_{1N}。

處理(4)：署名驗證演算法Ver使用再生之c_{01}，c_{11}，...，c_{0N}，c_{1N}驗證Cmt=H(c_{01}，c_{11}，...，c_{0N}，c_{1N})及d=H(t_{11}，e_{11}，...，t_{1N}，e_{1N}，)。

如以上所說明，使公鑰認證方式之模型中之證明者與電子署名方式中之署名者對應，藉此可使公鑰認證方式之演算法向電子署名方式之演算法變形。

<5：關於署名驗證所需之記憶體量之削減方法>

但是，於上述電子署名方式之演算法中，在署名驗證演算法Ver接收到電子署名整體之後執行署名驗證之處理。然而，於上述電子署名方式之情形時，電子署名之資料大小比較大。因此，於使用RFID(Radio Frequency IDentification，射頻識別)等僅具有較小記憶體容量之器件進行認證之情形時，必需注意記憶體之空閒容量或認證處理中之記憶體使用率等。又，亦設想到於利用不具有充分之記憶體容量之器件之情形時無法進行認證之情形。因此，本案發明者研究出一種署名驗證所需之記憶體量之削減方法。

[5-1：關於雜湊函數之構造(圖10)]

首先，本案發明者關注於雜湊函數之構造。於多數情形時，雜湊函數具有將輸入劃分成區塊單位，以區塊為單位依序展開處理之構造。例如，於SHA-1之情形時，雜湊函數具有如圖10所示之構造。圖10所示之雜湊函數將填充結束之輸入M劃分成Z個區塊m_1、...、m_Z，一面使索引j增量一面逐次地使區塊m_j與初始值IV或中間值CV_j一併作用於特定之函數CF而生成雜湊值。從而，於獲得中間值CV_j之時間點，無需在此以前所利用之區塊。因此，研究出一種利用該特性有效地削減演算法之執行所需之記憶體量之結構(以下，稱為記憶體削減方法)。以下，對將該結構應用於上述電子署名方式中之方法進行說明。

[5-2：面向基於3路徑動作方式之電子署名方式之應用例(圖12)]

首先，對將上述記憶體削減方法應用在基於圖8所示之3路徑動作方式之電子署名方式之演算法中之方法進行說明。

(通常之安裝方法：圖11)

通常，上述電子署名方式之署名驗證演算法Ver如圖11所示，1次性接收構成電子署名之(Cmt，Rsp_1，...，Rsp_N)(S101)。接著，署名驗證演算法Ver執行(Ch_1，...，Ch_N)←H(M，Cmt)(S102)。繼而，署名驗證演算法Ver執行(c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N})←重新產生(Ch_1，...，Ch_N；Rsp_1，...，Rsp_N)(S103)。然後，署名驗證演算法Ver驗證Cmt=H(c_{01}，c_{11}，c_{21}，...，c_{0N}，c_{1N}，c_{2N})(S104)，而

結束署名驗證之一系列處理。

(記憶體削減方法：圖12)

於通常之安裝方法之情形時，如圖11之步驟S101般，若一次性接收電子署名，則必需用以保持(Rsp_1，...，Rsp_N)直至步驟S103之處理完成為止之記憶體。然而，如自圖5之演算法構成所知，於步驟S103中所執行之(c_{0i}，c_{1i}，c_{2i})之再生中不利用(Ch_i；Rsp_i)以外之資訊。又，可知：若考慮到圖10所示之雜湊函數之構造，則於步驟S104中所執行之雜湊函數之計算可以區塊為單位進行分割再執行。因此，將署名驗證之處理之構成改良成圖12所示之構成。

於圖12所示之構成之情形時，署名驗證演算法Ver首先只接收電子署名中所含之Cmt(S111)。接著，署名驗證演算法Ver執行(Ch_1，...，Ch_N)←H(M，Cmt)(S112)。繼而，署名驗證演算法Ver相對於i=1~N，一面使i增量一面逐次地執行步驟S113~S115之處理。

於步驟S113中，署名驗證演算法Ver接收Rsp_i(S113)。接著，署名驗證演算法Ver使用所接收到之Rsp_i，執行(c_{0i}，c_{1i}，c_{2i})←重新產生(Ch_i；Rsp_i)(S114)。於執行步驟S114之處理之後，便無需Ch_i及Rsp_i。因此，署名驗證演算法Ver於執行步驟S114之處理之後將Ch_i及Rsp_i自記憶體中刪除。

繼而，署名驗證演算法Ver執行tmp_i←H_i(tmp_{i-1}；c_{0i}，c_{1i}，c_{2i})(S115)。再者，函數H_i係將於雜湊函數H之內部計算至c_{0i}，c_{1i}，c_{2i}為止時所生成之中間值輸出之函數。由於函數H_i之輸入大小會根據實際所選擇之雜湊函數而有所不

同，所以根據需要，可進行附加位元等適當之輸入長度之修正。若使用函數H_i，則雜湊函數H係藉由包括以下所示之處理(1)~處理(3)之演算法而表現。而且，tmp_N成為雜湊函數H之最終之輸出(雜湊值)。實際上可根據雜湊函數之規格，而於最終處理中進行填充之追加處理。

處理(1)：$tmp_0 \leftarrow$空字串

處理(2)：

for i=1 to N

$tmp_i \leftarrow H_i(tmp_{i-1}；c_{0i}，c_{1i}，c_{2i})$

end for

處理(3)：output tmp_N

於相對於i=1~N，執行步驟S113~S115之處理之後，署名驗證演算法Ver驗證$Cmt=tmp_N$是否成立(S116)，而結束署名驗證之一系列處理。如上所述，署名驗證演算法Ver將於重複執行步驟S113~S115之處理之過程中無需之資訊自記憶體中刪除。因此，署名驗證所需之記憶體量可抑制為最小限度。其結果，即便僅具有較少之記憶體容量之器件，亦可驗證上述署名。

[5-3：面向基於5路徑動作方式之電子署名方式之應用例(圖14)]

其次，對將上述記憶體削減方法應用在基於圖9所示之5路徑動作方式之電子署名方式之演算法中之方法進行說明。

(通常之安裝方法：圖13)

通常，上述電子署名方式之署名驗證演算法Ver如圖13所示，1次性接收構成電子署名之(Cmt，d，Rsp_1，...，Rsp_N)(S121)。接著，署名驗證演算法Ver執行(Ch_{A1}，...，Ch_{AN})←H(M，Cmt)(S122)。繼而，署名驗證演算法Ver執行(Ch_{B1}，...，Ch_{BN})←H(M，Cmt，Ch_{A1}，...，Ch_{AN}，d)(S123)。然後，署名驗證演算法Ver執行(c_{01}，c_{11}，...，c_{0N}，c_{1N}，d_{11}，e_{11}，...，d_{1N}，e_{1N})←重新產生(Ch_{A1}，...，Ch_{AN}，Ch_{B1}，...，Ch_{BN}；Rsp_1，...，Rsp_N)(S124)。接著，署名驗證演算法Ver驗證Cmt=H(c_{01}，c_{11}，...，c_{0N}，c_{1N})與d=H(d_{11}，e_{11}，...，d_{1N}，e_{1N})(S125)，而結束署名驗證之一系列處理。

(記憶體削減方法：圖14)

如圖13之步驟S121般，若一次性接收電子署名，則必需用以保持(Rsp_1，...，Rsp_N)直至步驟S124之處理完成為止之記憶體。然而，如自圖7之演算法構成所知，於步驟S124中所執行之(c_{0i}，c_{1i}，d_{1i}，e_{1i})之再生中不利用(Ch_{Ai}，Ch_{Bi}；Rsp_i)以外之資訊。又，可知：若考慮到圖10所示之雜湊函數之構造，則步驟S125中所執行之雜湊函數之計算可以區塊為單位進行分割再執行。因此，將署名驗證之處理之構成改良成如圖14所示之構成。

於圖14所示之構成之情形時，署名驗證演算法Ver首先只接收電子署名中所含之Cmt(S131)。接著，署名驗證演算法Ver執行(Ch_{A1}，...，Ch_{AN})←H(M，Cmt)(S132)。

繼而，署名驗證演算法Ver接收d(S133)。接著，署名驗

證演算法Ver使用所接收到之d，執行(Ch_{B1}，...，Ch_{BN})←H(M，Cmt，Ch_{A1}，...，Ch_{AN}，d)(S134)。於執行步驟S134之處理之後，便無需d。因此，署名驗證演算法Ver於執行步驟S134之處理之後將d自記憶體中刪除。接著，署名驗證演算法Ver一面相對於i=1~N，使i增量一面逐次地執行步驟S135~S137之處理。

於步驟S135中，署名驗證演算法Ver接收Rsp_i(S135)。接著，署名驗證演算法Ver使用所接收到之Rsp_i，執行(c_{0i}，c_{1i}，t_{1i}，e_{1i})←重新產生(Ch_{Ai}，Ch_{Bi}，Rsp_i)(S136)。於執行步驟S136之處理之後，便無需Ch_{Ai}，Ch_{Bi}及Rsp_i。因此，署名驗證演算法Ver於執行步驟S136之處理之後將Ch_{Ai}，Ch_{Bi}及Rsp_i自記憶體中刪除。

其次，署名驗證演算法Ver執行tmp_i←H_i(tmp_{i-1}；c_{0i}，c_{1i})與tmp_i'←H_i(tmp_{i-1}'；t_{1i}，e_{1i})(S137)。於相對於i=1~N，執行步驟S135~S137之處理之後，署名驗證演算法Ver驗證Cmt=tmp_N與d=tmp_N'是否成立(S138)，而結束署名驗證之一系列處理。如上所述，署名驗證演算法Ver將於重複執行步驟S135~S137之處理之過程中無需之資訊自記憶體中刪除。因此，署名驗證所需之記憶體量抑制為最小限度。其結果，即便為僅具有較少之記憶體容量之器件，亦可驗證上述署名。

以上，對署名驗證所需之記憶體量之削減方法進行了說明。

<6：關於自2進制數之隨機數列抽選3進制數之隨機數列

之方法>

但是，於基於3路徑動作方式之公鑰認證方式之演算法中，存在生成N個以上3進制數之均勻隨機數之場景。然而，生成3進制數之均勻隨機數之優異之隨機數生成器並不普通。因此，必需考慮使用生成2進制數之均勻隨機數之優異之隨機數生成器生成3進制數之均勻隨機數之方法。因此，本案發明者研究出一種自2進制數之均勻隨機數高效地生成3進制數之均勻隨機數之方法。以下，對該方法詳細地進行說明。再者，於以下說明中，對以l進制表現(l為2或3)表達之1個數計數為1記號。

[6-1：抽選方法#1(2位元劃分)(圖15)]

首先，一面參照圖15，一面對每2位元地劃分M位元之2進制數而抽選3進制數之方法(以下，稱為抽選方法#1)進行介紹。如圖15所示，若每2位元地劃分2進制表現之隨機數列，則可獲得M/2個2位元之隨機數。例如，若將「00」關聯為3進制數之「0」、將「01」關聯為3進制數之「1」、將「10」關聯為3進制數之「2」，則可自以2位元為單位之2進制表現之隨機數列獲得3進制數之隨機數列。其中，2位元之值「11」除外。即，抽選方法#1係自由2進制數2記號表現之2^2組數之中抽選由3進制數1記號表現之3^1組數的方法。從而，無法抽選N個以上3進制數之概率P_1如下式(13)。

[數 11]

$$P_1 = \sum_{M/2-N<i\leq M/2} C_i (1/4)^i (3/4)^{M/2-i} \quad \cdots(13)$$

[6-2：抽選方法#2(無劃分)(圖 16)]

其次，一面參照圖 16，一面對未將 2 進制數 M 記號之隨機數劃分利用而抽選 3 進制數 L 記號之隨機數之方法(以下，稱為抽選方法#2)進行介紹。其中，L 係使 $3^L \leqq 2^M$ 之最大之整數。可由 2 進制數 M 記號表現之數存在 2^M 組。另一方面，可由 3 進制數 L 記號表現之數僅存在 3^L 組。因此，於由 2 進制數 M 記號表現之 2^M 組之數之中，2^M-3^L 組數未用作 3 進制表現之隨機數。從而，無法抽選 N 個以上 3 進制數之概率 P_2 如下式(14)。

[數 12]

$$P_2 = 1-3^L/2^M \quad \cdots(14)$$

[6-3：抽選方法#3(k 位元劃分)(圖 17)]

上述抽選方法#1 係以最小之劃分單位劃分 2 進制表現之隨機數列之方法。另一方面，上述抽選方法#2 係(因為考慮到 M 位元劃分)以最大之劃分單位劃分 2 進制表現之隨機數列之方法。如自上式(13)及式(14)所知，根據劃分之長度，無法抽選 N 個以上 3 進制數之概率不同。另外，如圖 17 所示，於以 k 位元為單位劃分 2 進制 M 記號之隨機數列之情形時，無法抽選 N 個以上 3 進制數之概率 P_3 如下式(15)。

[數 13]

$$P_3 = \sum_{M/k-N/L<i\leq M/k} {}_{M/k}C_i \left(1-3^L/2^k\right)^i \left(3^L/2^k\right)^{M/k-i} \quad \cdots(15)$$

只要可將無法抽選N個以上3進制數之概率P_3最小化，便可效率最佳地抽選3進制表現之隨機數列。例如，於M=512、N=140之情形時，當k=8之時概率P_3最小。

(6-3-1：基本構成(圖18))

此處，一面參照圖18，一面對自2進制M記號之隨機數列抽選3進制L記號之隨機數列之處理之流程進行說明。如圖18所示，首先，生成2進制M記號之隨機數列(S201)。接著，2進制M記號之隨機數列以k位元單位劃分(S202)。繼而，抽選以k位元為單位地劃分之位元行X_{2k}之中使$X_{2k} \leqq 3^L$之位元行(S203)。其次，以3進制數表現輸出所抽選之位元行(S204)，而一系列處理結束。

(6-3-2：追加抽選方法(圖19))

算出上式(15)所表現之概率P_3最小之劃分之長度k，執行圖18所示之演算法，藉此可高效地自2進制表現之隨機數列抽選3進制表現之隨機數列。然而，本案發明者關注於在圖18之步驟S204中利用使$X_{2k}>3^L$之位元行之點，從而研究出一種更高效地抽選3進制表現之隨機數列之方法。以下，一面參照圖19，一面對該方法進行說明。

該方法係利用於圖18之步驟S204中未被抽選之位元行，抽選3進制表現之記號行之方法。如圖19所示，首先，抽選於圖18之步驟S204中未被抽選之使$X_{2k}>3^L$之位元行之組(例如，若將位元行之組表現為$y_1y_2...y_{N'}$，則各個位元行y_i

為$3^L \leqq y_i < 2^k$)(S211)。接著，自所抽選之位元行y_i分別減算3^L，算出新的位元行之組(例如，若將新的位元行之組表現為$z_1 z_2 \ldots z_{N'}$，則各個位元行$z_i = y_i - 3^L$為$0 \leqq z_i < 2^k - 3^L$)(S212)。

其次，自新的位元行之組，抽選位元行X為$X < 3^{L'}$者(S213)。其中，L'為使$3^{L'} \leqq 2^k - 3^L$之最大之整數。接著，於步驟S213中所抽選之位元行以3進制表現輸出(S214)，從而一系列處理結束。藉由應用該演算法，可以概率$3^{L'}/(2^k - 3^L)$重新抽選L'個3進制數。再者，藉由回復性地使用該方法，可抽選更多3進制數。即，可於步驟S213中自位元行X為$X \geqq 3^{L'}$之位元行同樣地抽選3進制數。

以上，對自2進制數之均勻隨機數高效地生成3進制數之均勻隨機數之方法進行了說明。

<7：關於高效地代入多元多項式之係數之方法>

另外，至此為止關於證明者(或署名者)與驗證者之間共有多元多項式之方法並未具體地明示。作為共有多元多項式之方法，可考慮雙方間共有於生成多元多項式之係數(隨機數)時所使用之種子之方法。然而，只要雙方間不共有將利用所共有之種子而生成之隨機數適用於係數之順序，便無法共有多元多項式。

[7-1：基本之規定]

因此，於證明者(或署名者)與驗證者之間，對以哪種順序將使用所共有之種子而生成之隨機數列適用於多元多項式進行基本之規定。而且，於利用多元多項式時，遵循該

基本之規定將隨機數列適用於多元多項式。若使用此種方法，則證明者(或署名者)與驗證者之間共有多元多項式。

[7-2：資料之構造化]

其中，構成多元多項式之係數之數較為龐大。於以1位元單位表現1個係數之情形時，為表現多元多項式最低亦需數萬位元以上之資料。因此，將數代入多元多項式之係數中之處理之負荷非常高。因此，本案發明者研究出一種將多元多項式之係數以特定之單位構造化，而使向係數之代入處理效率化之方法(構造化方法#1及#2)。進而，本案發明者研究出一種於對相同之多元多項式之係數執行複數次代入處理之情形時使該處理效率提高之方法(構造化方法#3)。以下，對該等方法詳細地進行說明。

(7-2-1：構造化方法#1(圖20))

首先，對構造化方法#1進行說明。構造化方法#1係如圖20所示，將構成多元多項式之相同種類之項之係數整合為1個資料構造之方法。於圖20之例中，係數a_{11J}~a_{m1J}整合為資料構造A，係數b_{11}~b_{m1}整合為資料構造B。

於不應用構造化方法#1之情形時，相對於m個n元多項式之係數之代入係藉由如下演算法(例1)而進行。於例1之情形時，必需執行2×N×(N-1)×M/2次1位元之AND運算(&)。進而，必需執行N×(N-1)×M/2次1位元之XOR運算(^)。

(例1)

for L=1 to M

for I=1 to N

for J=I to N

[f之第L位元]^=[a_{IJL}]&[x之第I位元]&[x之第J位元]；

end for

end for

end for

output f；

另一方面，於如圖20所示將係數構造化，並將所生成之隨機數作為多元多項式之係數逐一部分地逐次應用之情形時，係數之代入演算法如(例2)。於例2之情形時，只要執行2×N×(N-1)/2次L位元之AND運算(&)，N×(N-1)/2次M位元之XOR運算(^)即可。再者，$a_{IJ(1\sim M)}$分別以循環之時序生成。又，亦可來回使用係數。例如，於循環執行N(N-1)/2次時，亦可並不每次都生成[$a_{IJ(1\sim M)}$]，而是每L次僅生成1次。又，於M次循環中，亦可一面使[$a_{IJ(1\sim M)}$]逐1位元地旋轉一面利用。

(例2)

for I=1 to N

for J=I to N

[f之第1~M位元]^=[$a_{IJ(1\sim M)}$]&[x之第I位元]&[x之第J位元]；

end for

end for

output f；

又，亦可如圖20所示將係數構造化，並將應用多元多項式之係數所得之中間之結果保持於表格中。於該情形時，係數之代入演算法如(例3)。再者，於排列$a_{IJ}[0][0]$~$a_{IJ}[2^k-1][2^k-1]$中，分別儲存有$a_{IJ}[x_1, \ldots, x_k][z_1, \ldots, z_k]=(a_{(k(I-1)+1)(k(J-1)+1)}\&x_1\&z_1)\wedge\ldots\wedge(a_{(k(I-1)+1)(k(J-1)+k)}\&x_1\&z_k)\wedge\ldots\wedge(a_{(k(I-1)+k)(k(J-1)+1)}\&x_k\&z_1)\wedge\ldots\wedge(a_{(k(I-1)+k)(k(J-1)+k)}\&x_k\&z_k)$。於例3之情形時，只要執行(N/k)(N/k-1)/2次L位元之XOR運算(^)即可。但是，與(例2)之演算法相比，所需記憶體量為$2^{2k}/k^2$倍。

例如，於k=1之時，L位元XOR運算為120＊119/2=7140次，所需之記憶體量為(例2)之2^2=4倍，循環次數無變化。又，於k=2之時，L位元XOR運算為60＊59/2=1770次，所需之記憶體量為$2^4/4$=4倍，循環次數為1/4。於k=4之時，L位元XOR運算為30＊29/2=435次，所需之記憶體量為$2^8/4^2$=16倍，循環次數為1/16.4。於k=6之時，L位元XOR運算為20＊19/2=190次，所需之記憶體量為$2^{12}/6^2$=114倍，循環次數為1/37.6。於k=8之時，L位元XOR運算為15＊14/2=135次，所需之記憶體為$2^{16}/8^2$=1024倍，循環次數為1/52.9。

(例3)

for I=1 to N/k

for J=I to N/k

[f之第1~M位元]^=$a_{IJ(1\sim M)}$[x之第k(I-1)+1~k位元][x之第k(J-1)+1~k位元]；

end for

end for

output f；

以上，對構造化方法#1之具體之係數之代入演算法進行了說明。藉由該構成，可期待執行演算法時處理之高速化。

(7-2-2：構造化方法#2(圖21))

其次，對構造化方法#2進制行說明。構造化方法#2係如圖21所示，以2次形式表現多元多項式，將2次形式之列或行整合成1個資料構造之方法。於圖21之例中，在列方向上整合有資料構造。

於如圖21所示將係數構造化，並將所生成之隨機數作為多元多項式之係數逐一部分地逐次應用之情形時，係數之代入演算法如(例4)。於例4之情形時，只要執行(N+1)×M次N位元之AND運算(&)、N×M次N位元之XOR運算(^)、M次函數Q之運算即可。

(例4)

for I=1 to N

$T^{\wedge}=A_I\&[x之第I位元]$；

end for

T&=x；

output Q(T)；

Q(z){

```
z=z^(z>>1);
z=z^(z>>2);
z=z^(z>>4);
z=z^(z>>8);
...
z=z^(z>>2^Log(N));
return z&1;
}
```

又，亦可如圖21所示將係數構造化，並將應用多元多項式之係數所得之中間之結果保持於表格中。於該情形時，係數之代入演算法如(例5)所示。再者，於排列$A_I[0]$~$A_I[2^k-1]$中，分別儲存有$A_I[x_1, ..., x_k]=(A_{(k(I-1)+1)}\&x_1)^...^(A_{(k(I-1)+k)}\&x_k)$。於例5之情形時，只要執行(N/k)×M次N位元之XOR運算(^)，執行M次N位元之函數Q之運算即可。但是，與(例4)之演算法相比，所需之記憶體量為2^k/k倍。

例如，於k=1之時，N位元XOR運算為120次，所需之記憶體量為(例4)之2倍，循環次數無變化。又，於k=4之時，N位元XOR運算為30次，所需之記憶體量為$2^4/4=4$倍，循環次數為1/4。於k=8之時，N位元XOR運算為15次，所需之記憶為$2^8/8=32$倍，循環次數為1/8。於k=16之時，N位元XOR運算為8次，所需之記憶體為$2^{16}/16=4096$倍，循環次數為1/15。

(例5)

```
for I=1 to N/k
```

T^=A_I[x之第k(I-1)+1~k(I-1)+k位元]；

end for

T&=x；

output Q(T)；

以上，對構造化方法#2之具體之係數之代入演算法進行了說明。藉由該構成，於執行演算法時可期待處理之高速化。

(7-2-3：構造化方法#3)

其次，對構造化方法#3進行說明。構造化方法#3係於對相同之多元多項式進行N次(N≧2)代入處理之情形時，並非自隨機數生成N次多項式並進行代入處理，而是將以「生成一部分係數，並進行關於該部分之N次之處理」部分為單位之逐次之處理並行進行N次之方法。若應用該方法，則於無法忽視隨機數生成之成本之情形時可改善N次之處理整體上之處理量。

例如，於圖5所示之演算法中，在步驟#1中一面更新引數一面重複執行N次多元多項式F及G之計算。對此，關於此種計算部分，係以利用相同之係數進行重複運算之方式構成。於使用上述(例2)之演算法計算多元多項式$F(r_{0i})(i=1\sim N)$之情形時，關於一次性生成之$[a_{IJL}]$，係以於將N個r_{0i}全部套用之後，執行下個關於$[a_{IJL}]$之處理之方式構成。若以此方式構成，則無需生成N次相同之係數$[a_{IJL}]$。

以上，對構造化方法#3之具體之係數之代入演算法進行

了說明。藉由該構成，可改善N次之處理中之合計之處理量。

<8：硬體構成例(圖22)>

上述各演算法例如可使用圖22所示之資訊處理裝置之硬體構成而執行。即，該各演算法之處理係藉由使用電腦程式控制圖22所示之硬體而實現。再者，該硬體之形態為任意，例如，個人電腦、行動電話、PHS、PDA等行動資訊終端、遊戲機、接觸式或非接觸式之IC晶片、接觸式或非接觸式之IC卡、或者各種資訊家電均包含在內。其中，上述PHS係Personal Handy-phone System(個人手持式電話系統)之縮寫。又，上述PDA係Personal Digital Assistant(個人數位助理)之縮寫。

如圖22所示，該硬體主要包含CPU902、ROM904、RAM906、主匯流排908、及橋接器910。進而，該硬體包含外部匯流排912、介面914、輸入部916、輸出部918、記憶部920、驅動器922、連接埠924、及通訊部926。其中，上述CPU係Central Processing Unit(中央處理單元)之縮寫。又，上述ROM係Read Only Memory(唯讀記憶體)之縮寫。而且，上述RAM係Random Access Memory(隨機存取記憶體)之縮寫。

CPU902發揮例如運算處理裝置或控制裝置之功能，根據記錄於ROM904、RAM906、記憶部920、或可移除式記錄媒體928中之各種程式，控制各構成要素之動作總體或其一部分。ROM904係儲存讀入CPU902中之程式或用於運

算中之資料等之機構。於RAM906中，臨時地或永久地儲存例如讀入CPU902中之程式、或於執行該程式時適當變化之各種參數等。

該等構成要素例如經由可高速地傳輸資料之主匯流排908而相互連接。另一方面，主匯流排908例如經由橋接器910而連接於資料傳輸速度相對較低速之外部匯流排912。又，作為輸入部916，使用例如：滑鼠、鍵盤、觸控面板、按鈕、開關、及搖桿等。進而，作為輸入部916，有時亦使用可利用紅外線或其他電波發送控制信號之遙控器。

作為輸出部918係例如CRT、LCD、PDP、或ELD等顯示裝置、揚聲器、頭戴式耳機等音訊輸出裝置、印表機、行動電話、或傳真機等可視覺性或聽覺性地向利用者通知所取得之資訊之裝置。其中，上述CRT係Cathode Ray Tube(陰極射線管)之縮寫。又，上述LCD係Liquid Crystal Display(液晶顯示器)之縮寫。而且，上述PDP係Plasma Display Panel(電漿顯示面板)之縮寫。進而，上述ELD係Electro-Luminescence Display(電致發光顯示器)之縮寫。

記憶部920係用以儲存各種資料之裝置。作為記憶部920，使用例如：硬碟驅動器(HDD)等磁記憶器件、半導體記憶器件、光記憶器件、或光磁記憶器件等。其中，上述HDD係Hard Disk Drive(硬碟驅動器)之縮寫。

驅動器922係將記錄於例如磁碟、光碟、光磁碟、或半導體記憶體等可移除式記錄媒體928中之資訊讀出或者將

資訊寫入可移除式記錄媒體928中之裝置。可移除式記錄媒體928例如為DVD(Digital Versatile Disc，數位多功能光碟)媒體、Blu-ray媒體、HD DVD(High Definition Digital Versatile Disc，高清晰度數位多功能光碟)媒體、各種半導體記憶媒體等。當然，可移除式記錄媒體928亦可為例如搭載有非接觸型IC晶片之IC卡、或電子機器等。其中，上述IC係Integrated Circuit(積體電路)之縮寫。

連接埠924例如為USB埠、IEEE1394埠、SCSI、RS-232C埠、或用以連接諸如光纖音訊端子等之類的外部連接機器930之埠。外部連接機器930例如為印表機、可攜式音樂播放器、數位相機、數位視訊攝影機、或IC記錄器等。其中，上述USB係Universal Serial Bus(通用串列匯流排)之縮寫。又，上述SCSI係Small Computer System Interface(小電腦系統介面)之縮寫。

通訊部926係用以連接於網路932之通訊器件，例如為有線或無線LAN、Bluetooth(註冊商標)、或WUSB用之通訊卡、光通訊用之路由器、ADSL用之路由器、或接觸或非接觸通訊用之器件等。又，連接於通訊部926之網路932係由利用有線或無線而連接之網路所構成，例如為網際網路、家庭內LAN、紅外線通訊、可見光通訊、廣播、或衛星通訊等。其中，上述LAN係Local Area Network(區域網路)之縮寫。又，上述WUSB係Wireless USB(Wireless Universal Serial Bus，無線通用串列匯流排)之縮寫。而且，上述ADSL係Asymmetric Digital Subscriber Line(非對

稱數位用戶線路)之縮寫。

<9：總結>

最後，對本技術之實施形態之技術內容簡單地進行總結。此處所述之技術內容可相對於例如PC(Personal Computer，個人電腦)、行動電話、遊戲機、資訊終端、資訊家電、汽車導航系統等各種資訊處理裝置而應用。再者，以下所述之資訊處理裝置之功能既可利用1台資訊處理裝置而實現，亦可利用複數台資訊處理裝置而實現。又，以下所述之資訊處理裝置執行處理時所使用之資料記憶機構及運算處理機構既可設置於該資訊處理裝置上，亦可設置於經由網路而連接之機器上。

上述資訊處理裝置之功能構成係以如下方式表現。例如，下述(1)所記載之資訊處理裝置具有執行將安全性之依據置於多次多元聯立方程式之求解困難性中之高效之公鑰認證方式或電子署名方式之演算法的功能。

(1)

一種署名驗證裝置，其包括：

署名取得部，其取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

署名驗證部，其藉由確認使用上述電子署名中所含之複

數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且

上述多元多項式之組F及上述向量y為公鑰，

上述署名取得部每次取得特定數之上述第2資訊，

上述署名驗證部使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

(2)

如上述(1)之署名驗證裝置，其中上述第1資訊係使用將所輸入之資訊劃分成區塊單位而進行處理之單向性函數而生成；

上述署名驗證部重複執行：使用上述每次取得特定數之上述第2資訊而生成在上述區塊單位之處理中上述單向性函數所輸出之中間值之處理、及使用該中間值及其次所取得之特定數之上述第2資訊而生成下一個中間值之處理，而還原上述第1資訊。

(3)

如上述(2)之署名驗證裝置，其中特定數之上述第2資訊於上述中間值之生成完成之階段被刪除。

(4)

如上述(2)或(3)之署名驗證裝置，其中上述特定數為1，且

包含於上述第2資訊中且於自上述單向性函數生成上述中間值時所使用之資訊之大小，為上述區塊單位之資料大

小以上。

(5)

一種署名驗證方法，其包括：

取得電子署名之步驟，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係為根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性之步驟；且

上述多元多項式之組F及上述向量y為公鑰，

於上述取得步驟中，每次取得特定數之上述第2資訊，

於上述驗證步驟中，使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

(6)

一種程式，其係用以使電腦實現如下功能者：

署名取得功能，其係取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第

1資訊乃使用上述署名金鑰s生成所必需者；以及

署名驗證功能，其係藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且

上述多元多項式之組F及上述向量y為公鑰，

上述署名取得功能每次取得特定數之上述第2資訊，

上述署名驗證功能使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

(7)

一種記錄媒體，其係記錄有用以使電腦實現如下功能之程式且可藉由電腦讀取者：

署名取得功能，其係取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

署名驗證功能，其係藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且

上述多元多項式之組F及上述向量y為公鑰，

上述署名取得功能每次取得特定數之上述第2資訊，

上述署名驗證功能使用每次取得特定數之上述第2資訊

逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

(備註)

上述署名驗證演算法Ver係署名取得部、署名驗證部之一例。

以上，一面參照附圖一面對本技術之較佳實施形態進行了說明，當然本技術並不限定於該例。若為業者，則瞭解：於專利申請範圍所記載之範疇內，毫無疑問可想出各種變更例或修正例，對於該等當然亦屬於本技術之技術性範圍內。

於上述說明中，對使用雜湊函數H之演算法進行了介紹，但亦可使用授權函數COM(commission)取代雜湊函數H。授權函數COM係採用字串S及隨機數ρ作為引數之函數。作為授權函數之例，存在由Shai Halevi與Silvio Micali於國際會議CRYPTO1996上發表之方式等。

【圖式簡單說明】

圖1係用以說明公鑰認證方式之演算法之構成之說明圖。

圖2係用以說明電子署名方式之演算法之構成之說明圖。

圖3係用以說明n路徑動作之公鑰認證方式之演算法之構成之說明圖。

圖4係用以說明3路徑動作之公鑰認證方式之高效之演算法之說明圖。

圖5係用以說明3路徑動作之公鑰認證方式之高效之演算法之並行化的說明圖。

圖6係用以說明5路徑動作之公鑰認證方式之高效之演算法之構成例的說明圖。

圖7係用以說明5路徑動作之公鑰認證方式之高效之演算法之並行化的說明圖。

圖8係用以說明將3路徑動作之公鑰認證方式之高效之演算法變形成電子署名方式之演算法之方法之說明圖。

圖9係用以說明將5路徑動作之公鑰認證方式之高效之演算法變形成電子署名方式之演算法之方法之說明圖。

圖10係用以說明雜湊函數之構造例之說明圖。

圖11係用以說明基於3路徑動作方式之電子署名方式之署名驗證方法(通常之安裝方法)之說明圖。

圖12係用以說明基於3路徑動作方式之電子署名方式之署名驗證方法(記憶體削減方法)之說明圖。

圖13係用以說明基於5路徑動作方式之電子署名方式之署名驗證方法(通常之安裝方法)之說明圖。

圖14係用以說明基於5路徑動作方式之電子署名方式之署名驗證方法(記憶體削減方法)之說明圖。

圖15係用以說明自2進制隨機數抽選3進制隨機數之方法(抽選方法#1)之說明圖。

圖16係用以說明自2進制隨機數抽選3進制隨機數之方法(抽選方法#2)之說明圖。

圖17係用以說明自2進制隨機數抽選3進制隨機數之方法

(抽選方法#3)之說明圖。

圖18係用以說明自2進制隨機數抽選3進制隨機數之方法(抽選方法#3)之說明圖。

圖19係用以說明自2進制隨機數抽選3進制隨機數之方法(抽選方法#3)之說明圖。

圖20係用以說明用以高效地代入多元多項式之係數之資料構造化方法(構造化方法#1)之說明圖。

圖21係用以說明用以高效地代入多元多項式之係數之資料構造化方法(構造化方法#2)之說明圖。

圖22係用以說明可執行本技術之各實施形態之演算法之資訊處理裝置之硬體構成例的說明圖。

【主要元件符號說明】

Gen	金鑰生成演算法
P	證明者演算法
Sig	署名生成演算法
V	驗證者演算法
Ver	署名驗證演算法

七、申請專利範圍：

1. 一種署名驗證裝置，其包括：

署名取得部，其取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

署名驗證部，其藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且

上述多元多項式之組F及上述向量y為公鑰，

上述署名取得部每次取得特定數之上述第2資訊，

上述署名驗證部使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

2. 如請求項1之署名驗證裝置，其中上述第1資訊係使用將所輸入之資訊劃分成區塊單位而進行處理之單向性函數而生成；

上述署名驗證部重複執行：使用上述每次取得特定數之上述第2資訊而生成在上述區塊單位之處理中上述單向性函數所輸出之中間值之處理、及使用該中間值及其次所取得之特定數之上述第2資訊而生成下一個中間值之處理，而還原上述第1資訊。

3. 如請求項2之署名驗證裝置，其中特定數之上述第2資訊於上述中間值之生成完成之階段被刪除。

4. 如請求項2之署名驗證裝置，其中上述特定數為1，且

包含於上述第2資訊中且於自上述單向性函數生成上述中間值時所使用之資訊之大小，為上述區塊單位之資料大小以上。

5. 一種署名驗證方法，其包括：

取得電子署名之步驟，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性之步驟；且

上述多元多項式之組F及上述向量y為公鑰，

於上述取得步驟中，每次取得特定數之上述第2資訊，

於上述驗證步驟中，使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

6. 一種程式，其係用以使電腦實現如下功能者：

署名取得功能，其係取得電子署名，該電子署名包

含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

署名驗證功能，其係藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且

上述多元多項式之組F及上述向量y為公鑰，

上述署名取得功能每次取得特定數之上述第2資訊，

上述署名驗證功能使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

7. 一種記錄媒體，其係記錄有用以使電腦實現如下功能之程式且可藉由電腦讀取者：

署名取得功能，其係取得電子署名，該電子署名包含：第1資訊，其係根據定義於環K上之多次多元多項式之組$F=(f_1, \ldots, f_m)$、署名金鑰$s \in K^n$、及文件M而生成者；及複數個第2資訊，其係根據上述文件M、上述多次多元多項式之組F、及向量$y=(f_1(s), \ldots, f_m(s))$而用以驗證上述第1資訊乃使用上述署名金鑰s生成所必需者；以及

署名驗證功能，其係藉由確認使用上述電子署名中所含之複數個第2資訊是否可還原上述第1資訊而驗證上述文件M之正當性；且

上述多元多項式之組F及上述向量y為公鑰，

上述署名取得功能每次取得特定數之上述第2資訊，

上述署名驗證功能使用每次取得特定數之上述第2資訊逐次地還原上述第1資訊，並將還原處理中無需之上述第2資訊於不再需要之階段刪除。

八、圖式：

公鑰認證方式

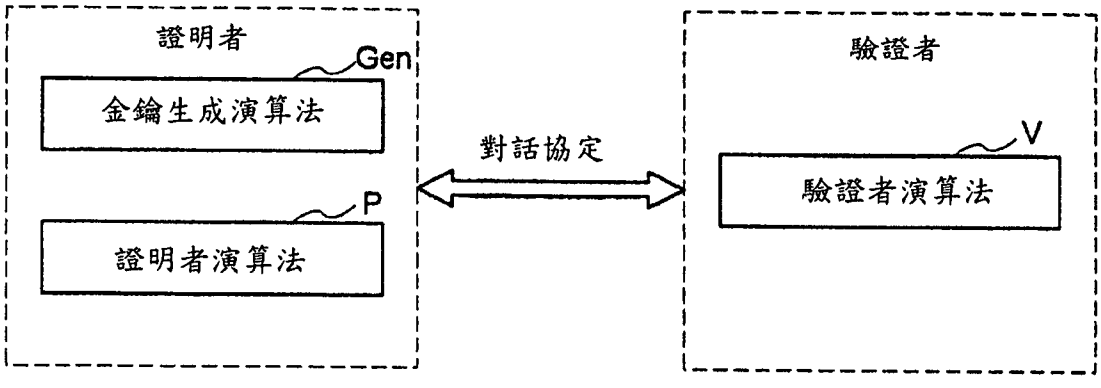


圖1

電子署名方式

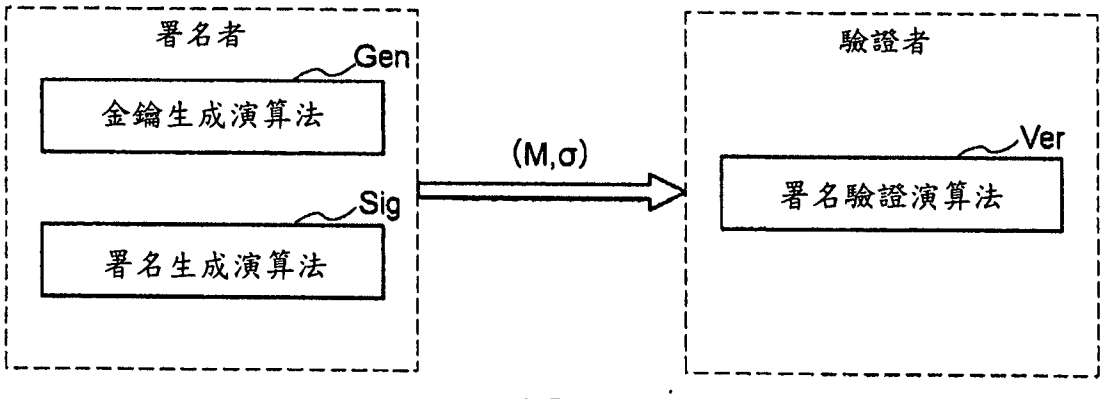


圖2

n 路徑動作之公鑰認證方式

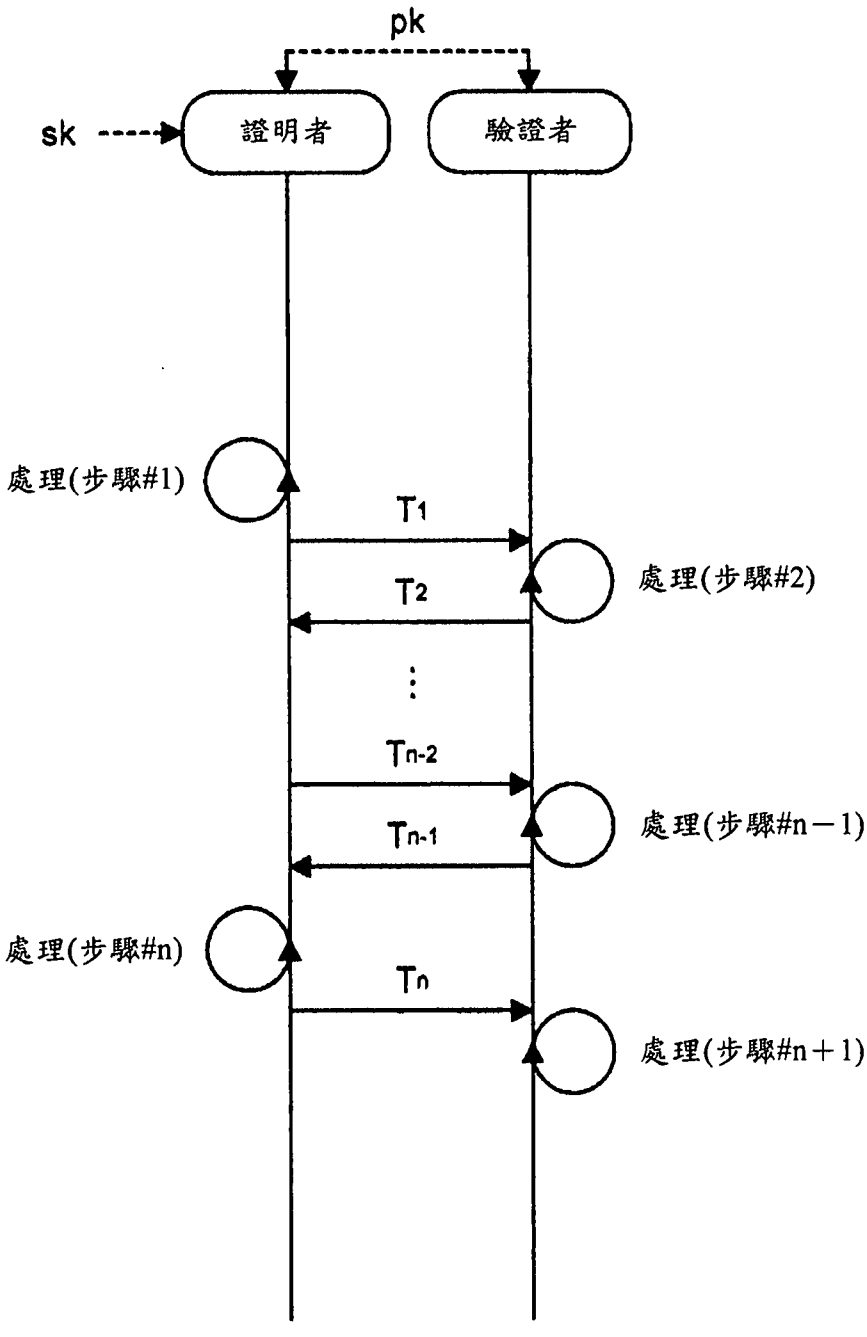


圖3

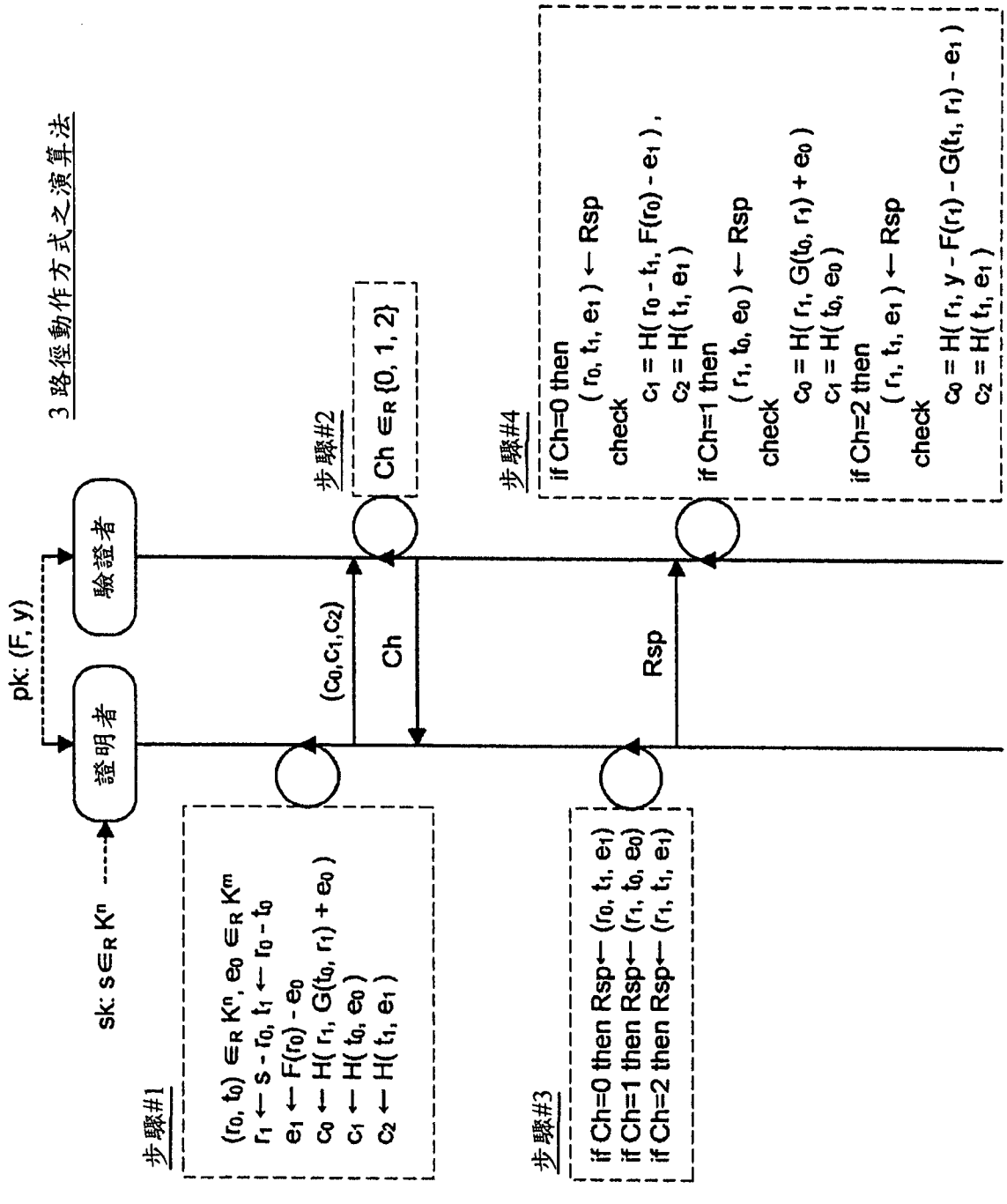

3 路徑動作方式之演算法
證明者
驗證者
pk: (F, y)
sk: s∈R Kn
步驟#1
(r0, t0) ∈R Kn, e0 ∈R Km
r1 ← s - r0, t1 ← r0 - t0
e1 ← F(r0) - e0
c0 ← H(r1, G(t0, r1) + e0)
c1 ← H(t0, e0)
c2 ← H(t1, e1)
(c0,c1,c2)
步驟#2
Ch ∈R {0, 1, 2}
Ch
步驟#3
if Ch=0 then Rsp← (r0, t1, e1)
if Ch=1 then Rsp← (r1, t0, e0)
if Ch=2 then Rsp← (r1, t1, e1)
Rsp
步驟#4
if Ch=0 then
(r0, t1, e1) ← Rsp
check
c1 = H(r0 - t1, F(r0) - e1) ,
c2 = H(t1, e1)
if Ch=1 then
(r1, t0, e0) ← Rsp
check
c0 = H(r1, G(t0, r1) + e0)
c1 = H(t0, e0)
if Ch=2 then
(r1, t1, e1) ← Rsp
check
c0 = H(r1, y - F(r1) - G(t1, r1) - e1)
c2 = H(t1, e1)

圖 4

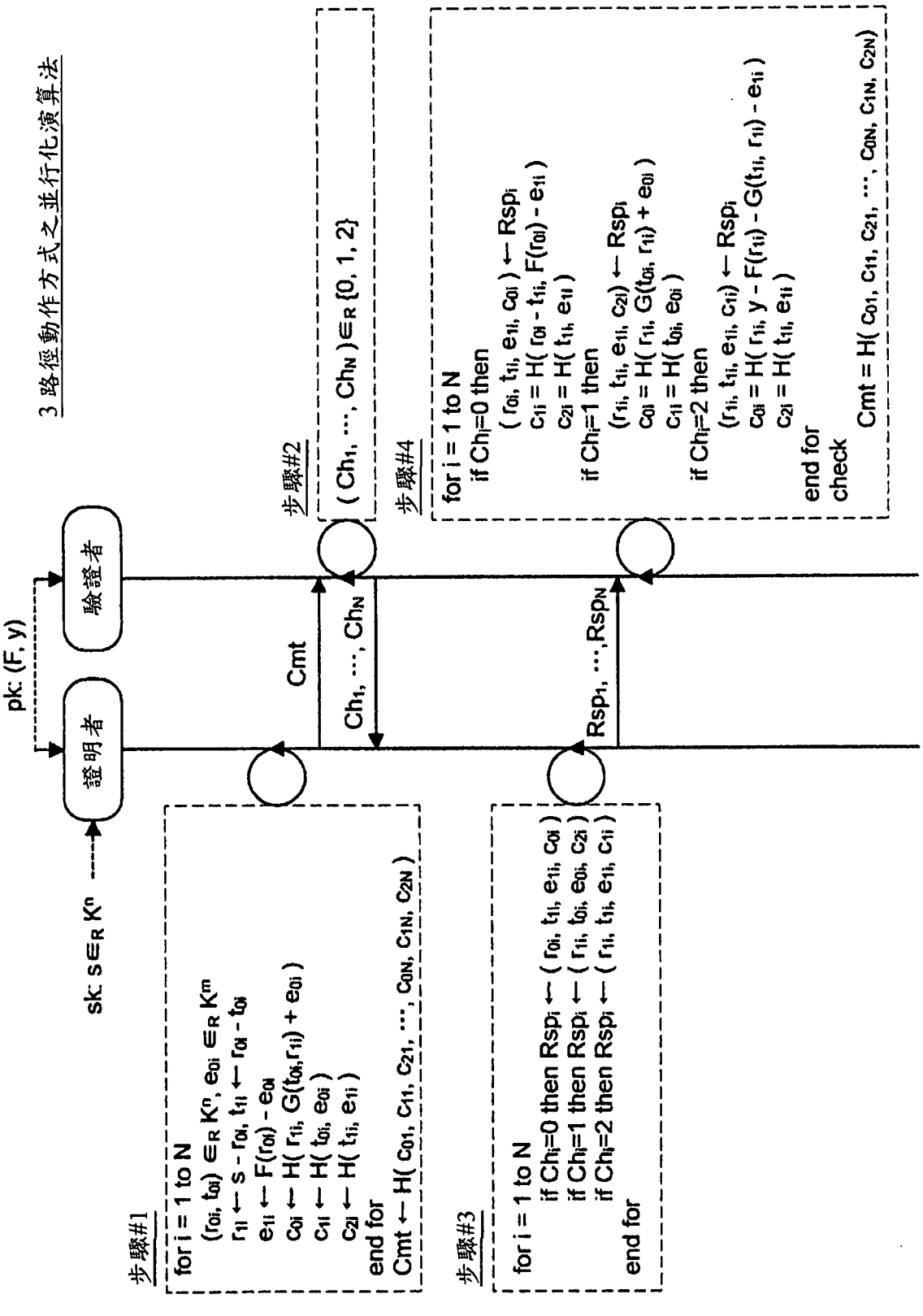

3 路徑動作方式之並行化演算法
pk: (F, y)
sk: s∈R Kn
證明者
驗證者
Cmt
Ch1, ···, ChN
Rsp1, ···, RspN
步驟#1
for i = 1 to N
(r0i, t0i) ∈R Kn, e0i ∈R Km
r1i ← s - r0i, t1i ← r0i - t0i
e1i ← F(r0i) - e0i
c0i ← H(r1i, G(t0i,r1i) + e0i)
c1i ← H(t0i, e0i)
c2i ← H(t1i, e1i)
end for
Cmt ← H(c01, c11, c21, ···, c0N, c1N, c2N)
步驟#2
(Ch1, ···, ChN)∈R {0, 1, 2}
步驟#3
for i = 1 to N
if Chi=0 then Rspi ← (r0i, t1i, e1i, c0i)
if Chi=1 then Rspi ← (r1i, t0i, e0i, c2i)
if Chi=2 then Rspi ← (r1i, t1i, e1i, c1i)
end for
步驟#4
for i = 1 to N
if Chi=0 then
(r0i, t1i, e1i, c0i) ← Rspi
c1i = H(r0i - t1i, F(r0i) - e1i)
c2i = H(t1i, e1i)
if Chi=1 then
(r1i, t1i, e1i, c2i) ← Rspi
c0i = H(r1i, G(t0i, r1i) + e0i)
c1i = H(t0i, e0i)
if Chi=2 then
(r1i, t1i, e1i, c1i) ← Rspi
c0i = H(r1i, y - F(r1i) - G(t1i, r1i) - e1i)
c2i = H(t1i, e1i)
end for
check
Cmt = H(c01, c11, c21, ···, c0N, c1N, c2N)

圖5

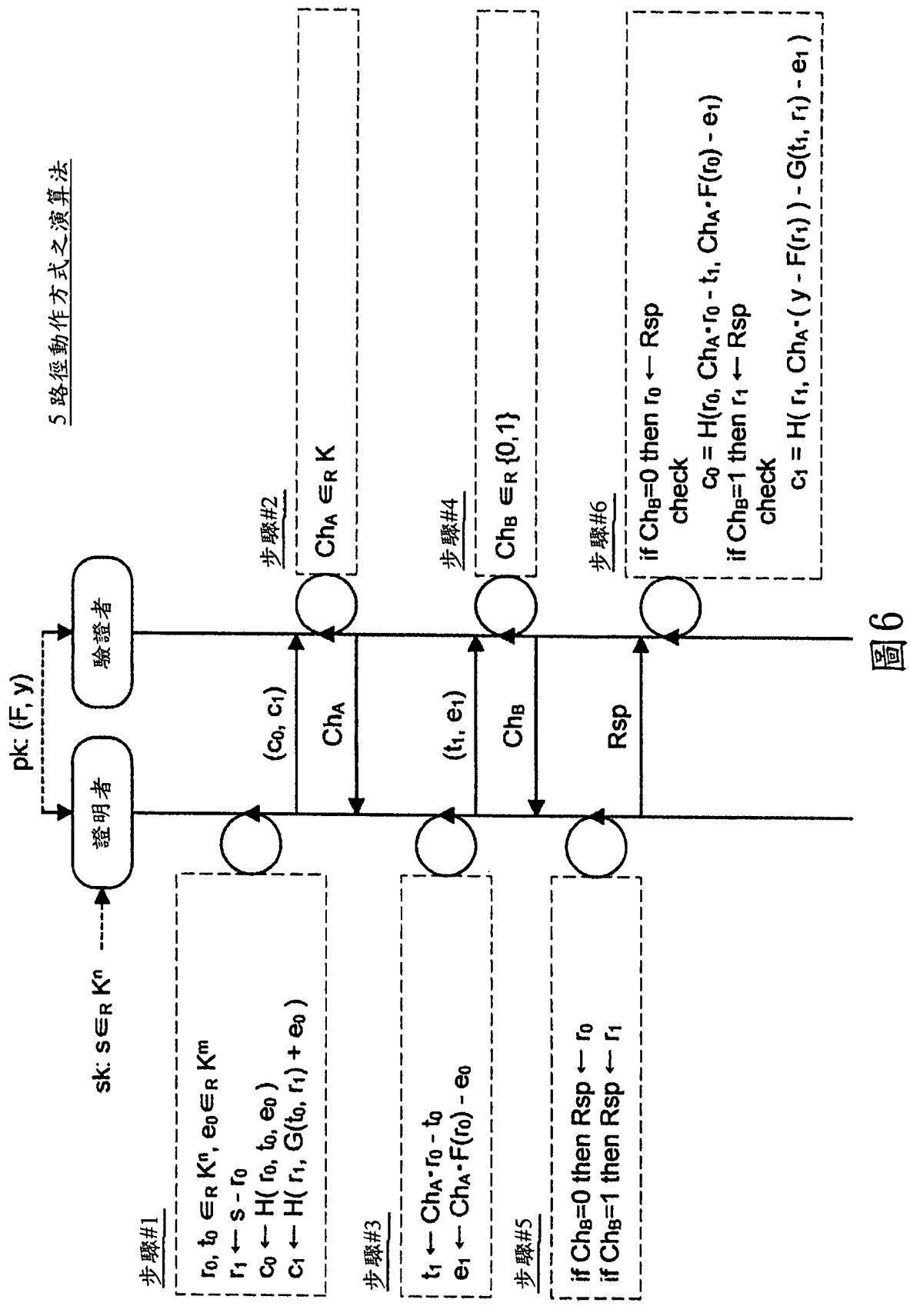

5 路徑動作方式之演算法
證明者
驗證者
pk: (F, y)
sk: s∈R Kn
步驟#1
r0, t0 ∈R Kn, e0∈R Km
r1 ← s - r0
c0 ← H(r0, t0, e0)
c1 ← H(r1, G(t0, r1) + e0)
(c0, c1)
步驟#2
ChA ∈R K
ChA
步驟#3
t1 ← ChA·r0 - t0
e1 ← ChA·F(r0) - e0
(t1, e1)
步驟#4
ChB ∈R {0,1}
ChB
步驟#5
if ChB=0 then Rsp ← r0
if ChB=1 then Rsp ← r1
Rsp
步驟#6
if ChB=0 then r0 ← Rsp
check
c0 = H(r0, ChA·r0 - t1, ChA·F(r0) - e1)
if ChB=1 then r1 ← Rsp
check
c1 = H(r1, ChA·(y - F(r1)) - G(t1, r1) - e1)

圖 6

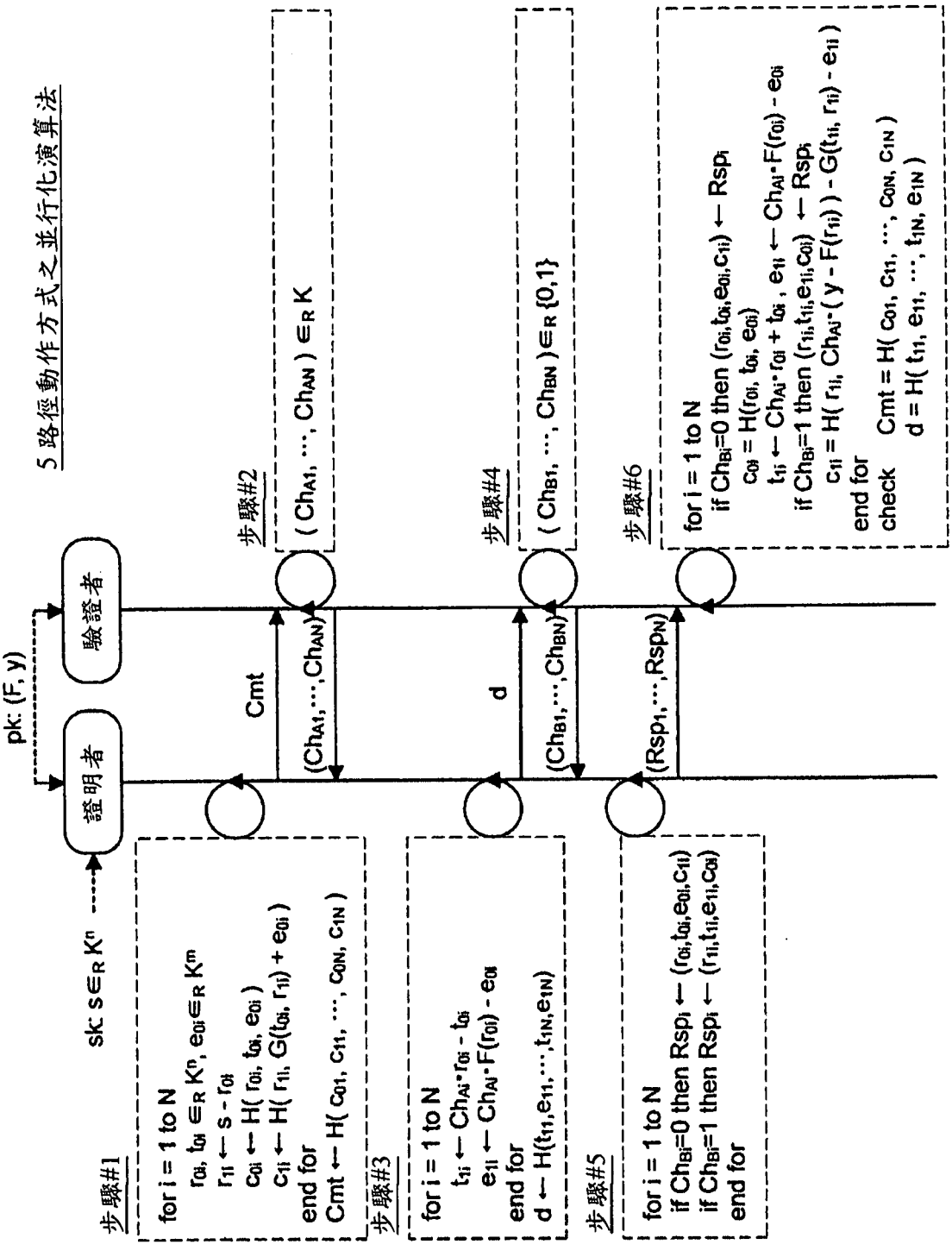

5 路徑動作方式之並行化演算法
pk: (F, y)
sk: s∈R Kn
證明者
驗證者
步驟#1
for i = 1 to N
r0i, t0i ∈R Kn, e0i∈R Km
r1i ← s - r0i
c0i ← H(r0i, t0i, e0i)
c1i ← H(r1i, G(t0i, r1i) + e0i)
end for
Cmt ← H(c01, c11, ···, c0N, c1N)
Cmt
步驟#2
(ChA1, ···, ChAN) ∈R K
(ChA1,···,ChAN)
步驟#3
for i = 1 to N
t1i ← ChAi·r0i - t0i
e1i ← ChAi·F(r0i) - e0i
end for
d ← H(t11,e11,···,t1N,e1N)
d
步驟#4
(ChB1, ···, ChBN)∈R {0,1}
(ChB1,···,ChBN)
步驟#5
for i = 1 to N
if ChBi=0 then Rspi ← (r0i,t0i,e0i,c1i)
if ChBi=1 then Rspi ← (r1i,t1i,e1i,c0i)
end for
(Rsp1,···,RspN)
步驟#6
for i = 1 to N
if ChBi=0 then (r0i,t0i,e0i,c1i) ← Rspi
c0i = H(r0i, t0i, e0i)
t1i ← ChAi·r0i + t0i , e1i ← ChAi·F(r0i) - e0i
if ChBi=1 then (r1i,t1i,e1i,c0i) ← Rspi
c1i = H(r1i, ChAi·(y - F(r1i)) - G(t1i, r1i) - e1i)
end for
check Cmt = H(c01, c11, ···, c0N, c1N)
d = H(t11, e11, ···, t1N, e1N)

圖 7

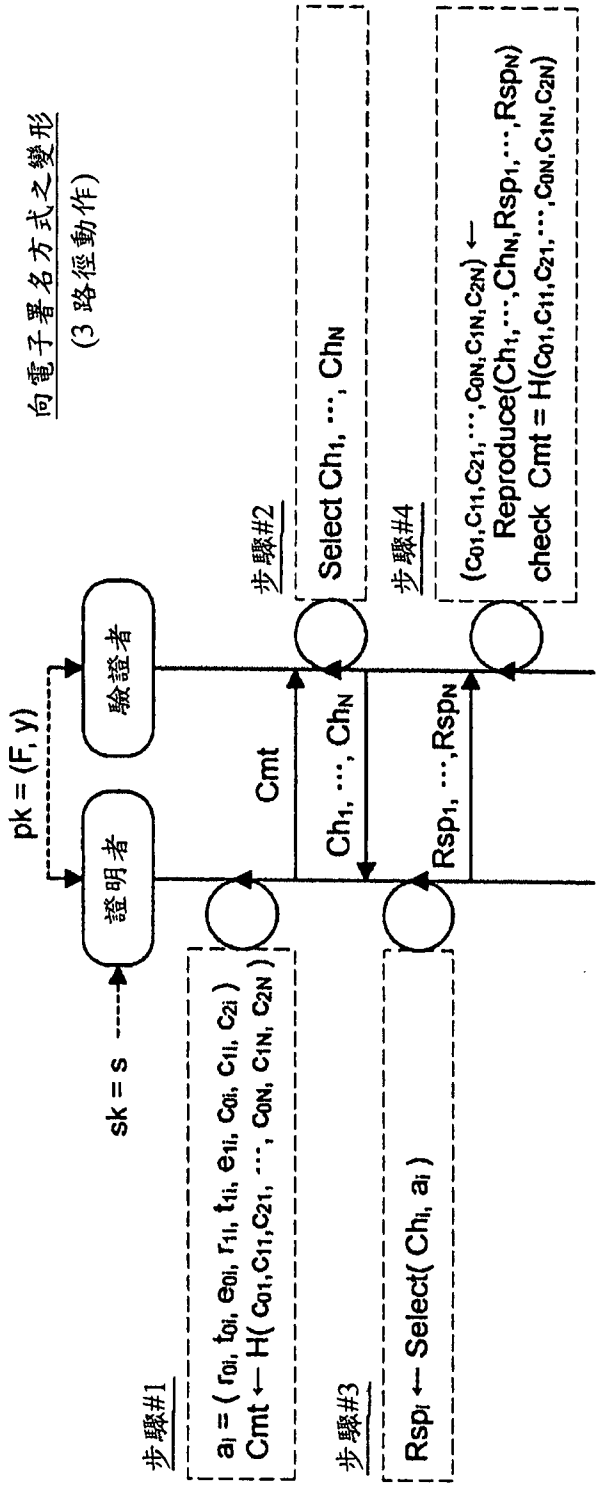

向電子署名方式之變形
(3 路徑動作)
pk = (F, y)
sk = s
證明者
驗證者
步驟#1
$a_i = (r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i}, c_{2i})$
$Cmt \leftarrow H(c_{01}, c_{11}, c_{21}, \cdots, c_{0N}, c_{1N}, c_{2N})$
Cmt
步驟#2
Select $Ch_1, \cdots, Ch_N$
$Ch_1, \cdots, Ch_N$
步驟#3
$Rsp_i \leftarrow Select(Ch_i, a_i)$
$Rsp_1, \cdots, Rsp_N$
步驟#4
$(c_{01}, c_{11}, c_{21}, \cdots, c_{0N}, c_{1N}, c_{2N}) \leftarrow$
Reproduce$(Ch_1, \cdots, Ch_N, Rsp_1, \cdots, Rsp_N)$
check $Cmt = H(c_{01}, c_{11}, c_{21}, \cdots, c_{0N}, c_{1N}, c_{2N})$
文件M
署名生成演算法 Sig
(1) $a_i = (r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i}, c_{2i})$
(2) $Cmt \leftarrow H(c_{01}, c_{11}, c_{21}, \cdots, c_{0N}, c_{1N}, c_{2N})$
(3) $(Ch_1, \cdots, Ch_N) \leftarrow H(M, Cmt)$
(4) $Rsp_i \leftarrow Select(Ch_i, a_i)$
(5) 電子署名: $(Cmt, Rsp_1, \cdots, Rsp_N)$
+
署名驗證演算法 Ver
(1) $(Ch_1, \cdots, Ch_N) \leftarrow H(M, Cmt)$
(2) $(c_{01}, c_{11}, c_{21}, \cdots, c_{0N}, c_{1N}, c_{2N}) \leftarrow$
Reproduce$(Ch_1, \cdots, Ch_N, Rsp_1, \cdots, Rsp_N)$
(3) check $Cmt = H(c_{01}, c_{11}, c_{21}, \cdots, c_{0N}, c_{1N}, c_{2N})$

圖 8

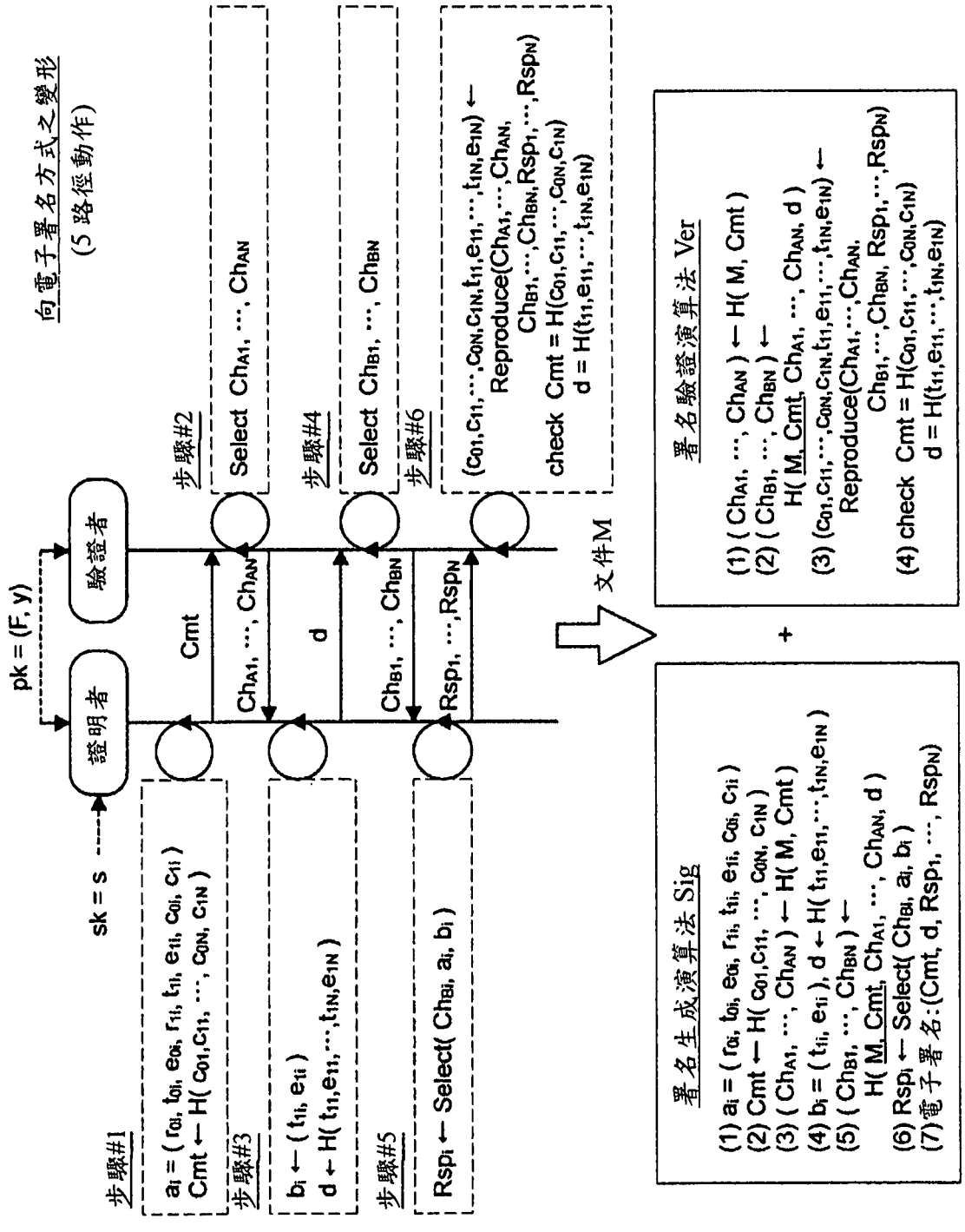
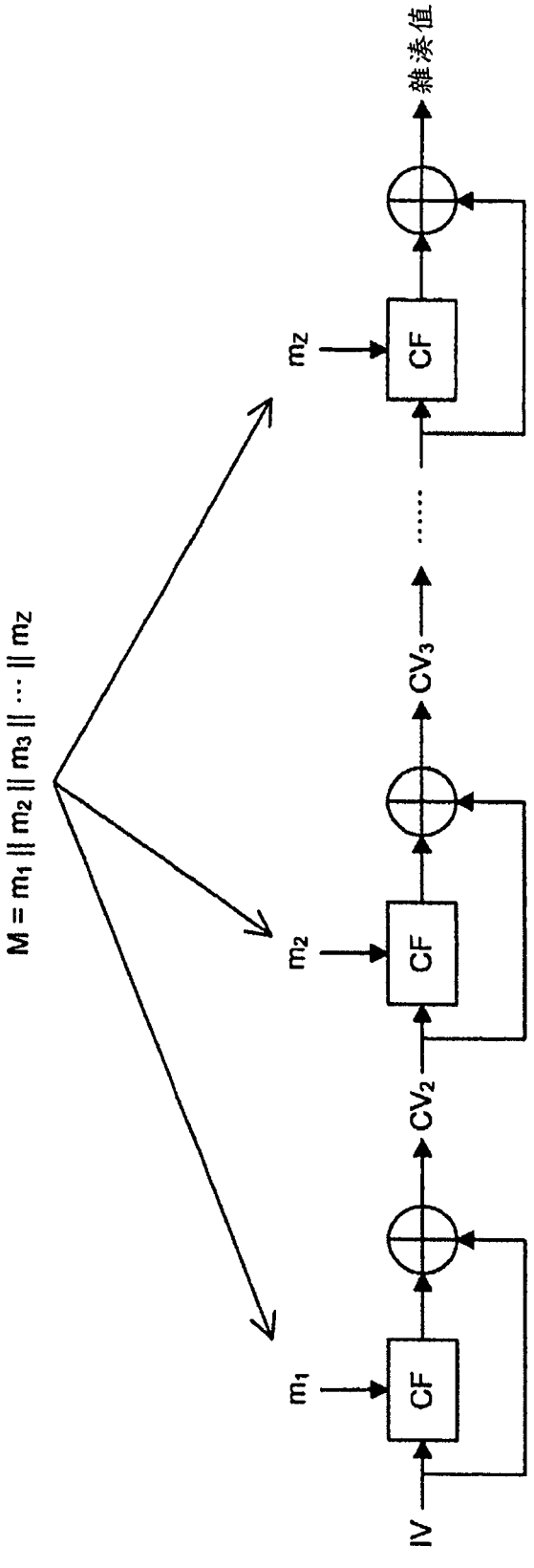

向電子署名方式之變形
(5 路徑動作)
pk = (F, y)
sk = s
證明者
驗證者
步驟#1
$a_i = (r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i})$
$Cmt \leftarrow H(c_{01}, c_{11}, \cdots, c_{0N}, c_{1N})$
Cmt
步驟#2
Select $Ch_{A1}, \cdots, Ch_{AN}$
$Ch_{A1}, \cdots, Ch_{AN}$
步驟#3
$b_i \leftarrow (t_{1i}, e_{1i})$
$d \leftarrow H(t_{11}, e_{11}, \cdots, t_{1N}, e_{1N})$
d
步驟#4
Select $Ch_{B1}, \cdots, Ch_{BN}$
$Ch_{B1}, \cdots, Ch_{BN}$
步驟#5
$Rsp_i \leftarrow Select(Ch_{Bi}, a_i, b_i)$
$Rsp_1, \cdots, Rsp_N$
步驟#6
$(c_{01}, c_{11}, \cdots, c_{0N}, c_{1N}, t_{11}, e_{11}, \cdots, t_{1N}, e_{1N}) \leftarrow$
Reproduce($Ch_{A1}, \cdots, Ch_{AN}$, $Ch_{B1}, \cdots, Ch_{BN}, Rsp_1, \cdots, Rsp_N$)
check $Cmt = H(c_{01}, c_{11}, \cdots, c_{0N}, c_{1N})$
$d = H(t_{11}, e_{11}, \cdots, t_{1N}, e_{1N})$
文件M
署名生成演算法 Sig
(1) $a_i = (r_{0i}, t_{0i}, e_{0i}, r_{1i}, t_{1i}, e_{1i}, c_{0i}, c_{1i})$
(2) $Cmt \leftarrow H(c_{01}, c_{11}, \cdots, c_{0N}, c_{1N})$
(3) $(Ch_{A1}, \cdots, Ch_{AN}) \leftarrow H(M, Cmt)$
(4) $b_i = (t_{1i}, e_{1i})$, $d \leftarrow H(t_{11}, e_{11}, \cdots, t_{1N}, e_{1N})$
(5) $(Ch_{B1}, \cdots, Ch_{BN}) \leftarrow$
H(M, Cmt, $Ch_{A1}, \cdots, Ch_{AN}$, d)
(6) $Rsp_i \leftarrow Select(Ch_{Bi}, a_i, b_i)$
(7)電子署名:(Cmt, d, $Rsp_1, \cdots, Rsp_N$)
+
署名驗證演算法 Ver
(1) $(Ch_{A1}, \cdots, Ch_{AN}) \leftarrow H(M, Cmt)$
(2) $(Ch_{B1}, \cdots, Ch_{BN}) \leftarrow$
H(M, Cmt, $Ch_{A1}, \cdots, Ch_{AN}$, d)
(3) $(c_{01}, c_{11}, \cdots, c_{0N}, c_{1N}, t_{11}, e_{11}, \cdots, t_{1N}, e_{1N}) \leftarrow$
Reproduce($Ch_{A1}, \cdots, Ch_{AN}$,
$Ch_{B1}, \cdots, Ch_{BN}, Rsp_1, \cdots, Rsp_N$)
(4) check $Cmt = H(c_{01}, c_{11}, \cdots, c_{0N}, c_{1N})$
$d = H(t_{11}, e_{11}, \cdots, t_{1N}, e_{1N})$

圖9

雜湊函數之構造例



・IV：初始值
・CVi：鏈接值
・CF：壓縮函數

圖10

通常之安裝方法

(基於3路徑動作方式之電子署名方式)

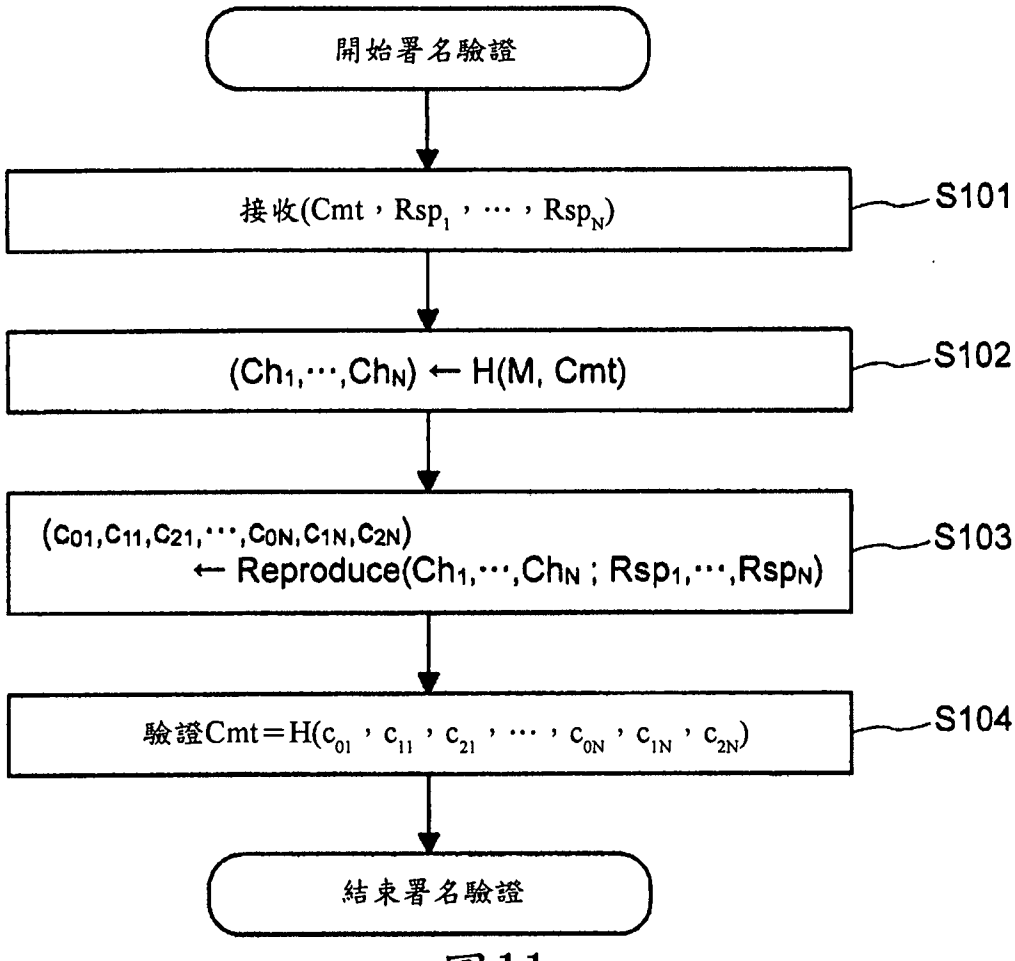


圖11

記憶體削減方法

(基於3路徑動作方式之電子署名方式)

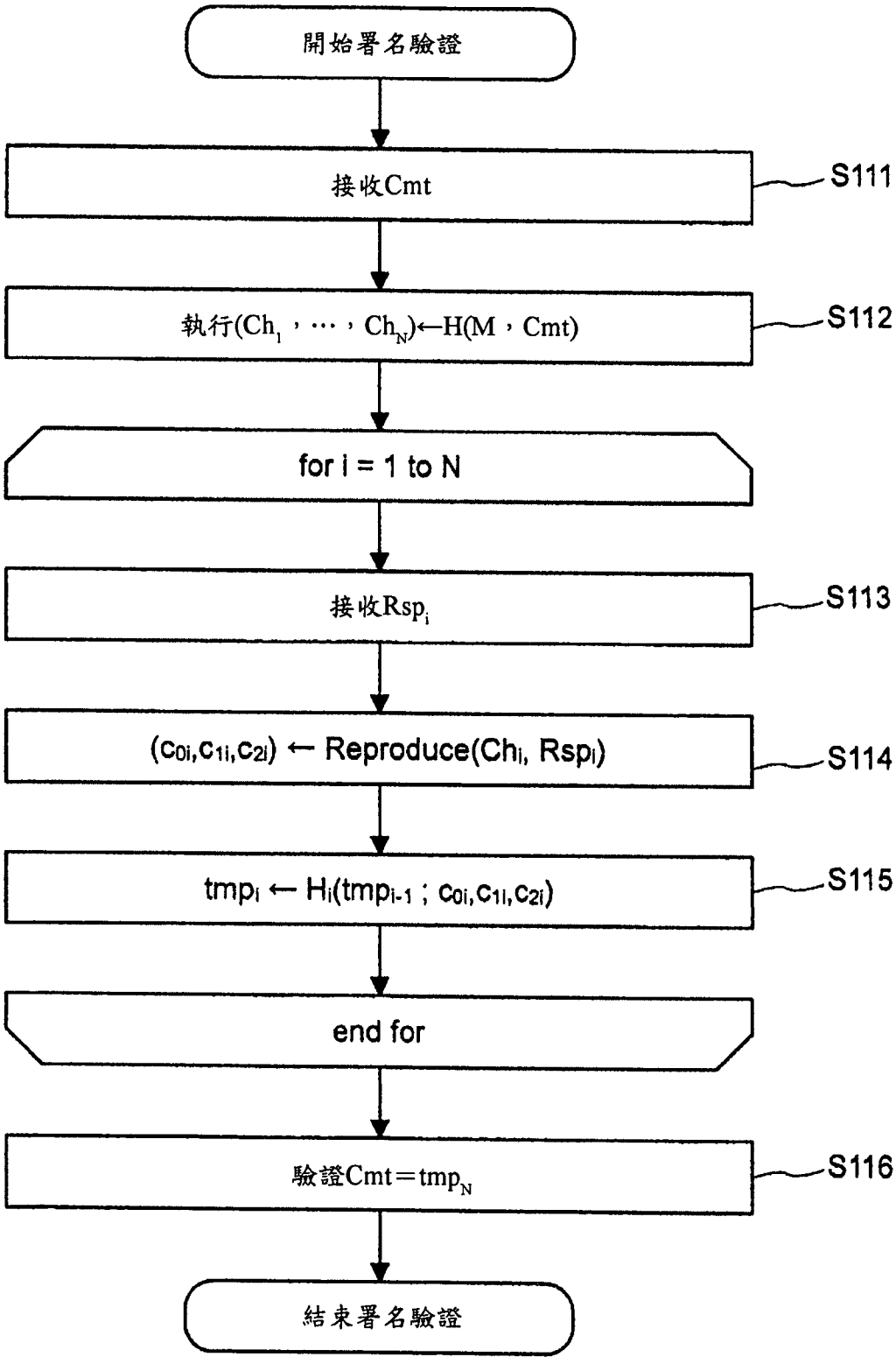


圖12

通常之安裝方法

(基於5路徑動作方式之電子署名方式)

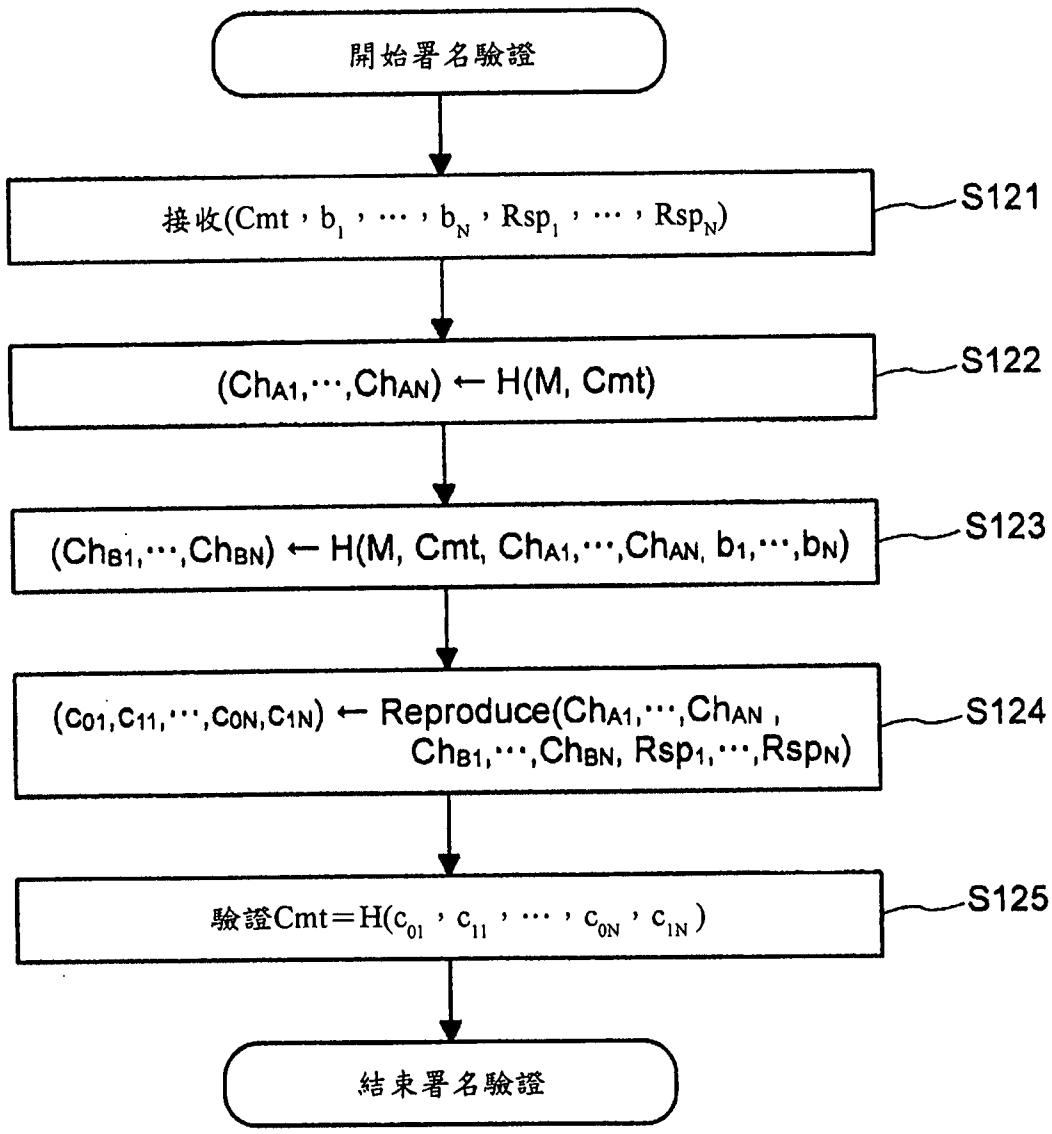


圖13

記憶體削減方法
(基於 5 路徑動作方式之電子署名方式)

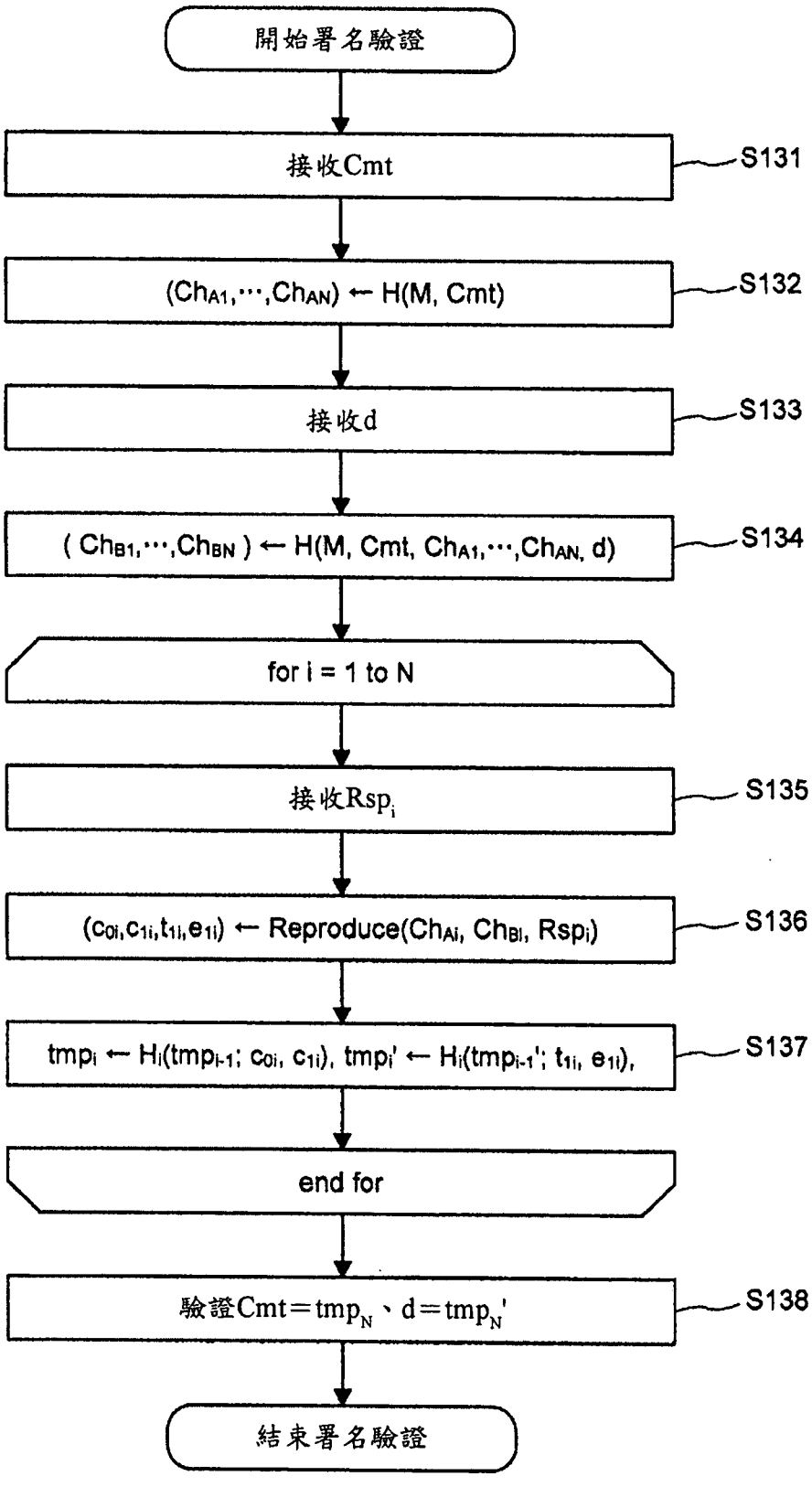
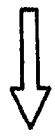


圖14

抽選方法#1

(2進制M記號之均勻隨機數)

X_{2M}： 0110110010101100 10 ⋯ 0101100100



(2位元劃分：2進制2記號→3進制1記號)

01	10	11	00	10	10	11	00	10	⋯	01	01	10	01	00
‖	‖	‖	‖	‖	‖	‖	‖	‖		‖	‖	‖	‖	‖
X_{22}： 1	2	×	0	1	2	×	0	2		1	1	2	1	0

將X_{22}＞2除外　　將X_{22}＞2除外

圖15

抽選方法#2

(2進制M記號之均勻隨機數)

X_{2M}： 0110110010101100 10 ⋯ 0101100100

(不劃分：2進制M記號→3進制L記號)

X_{2M}： 0110110010101100 10 ⋯ 0101100100

將X_{2M}＞3^L除外

圖16

抽選方法#3

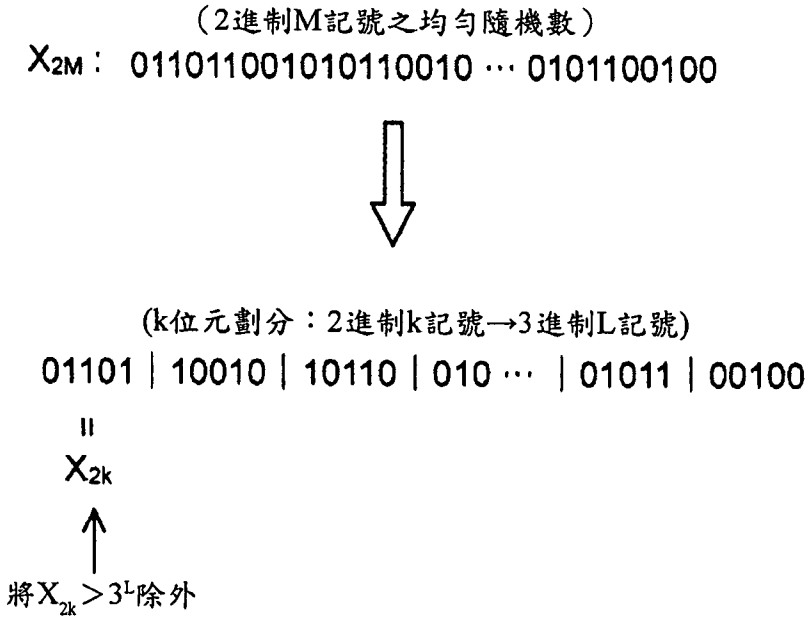


圖17

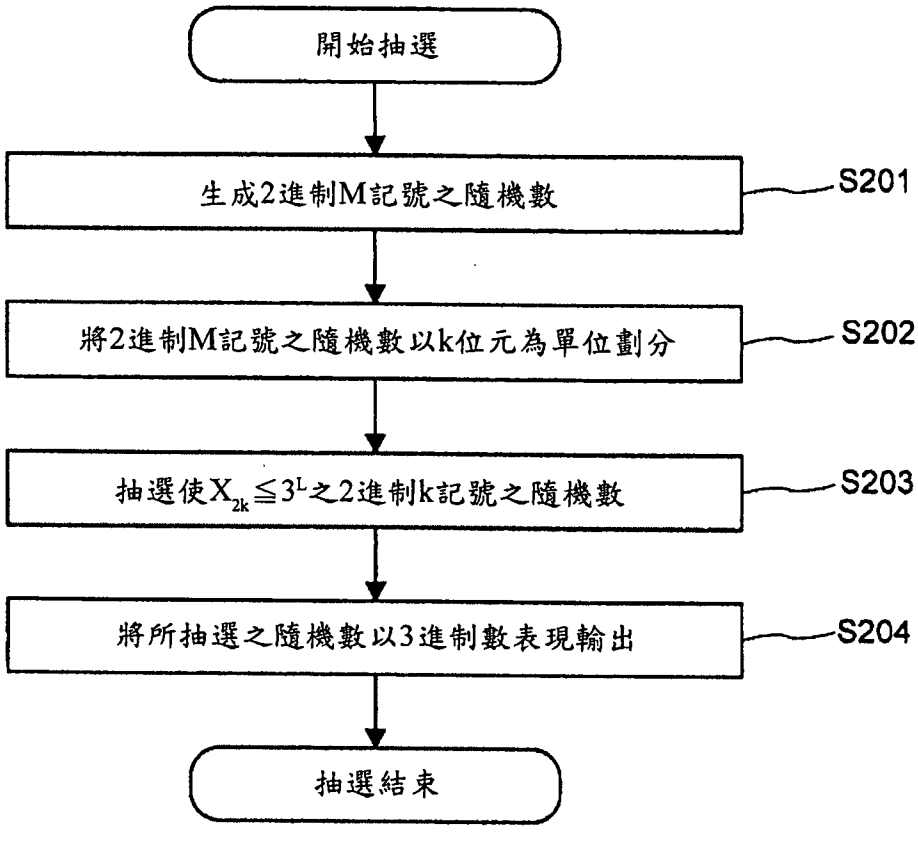


圖18

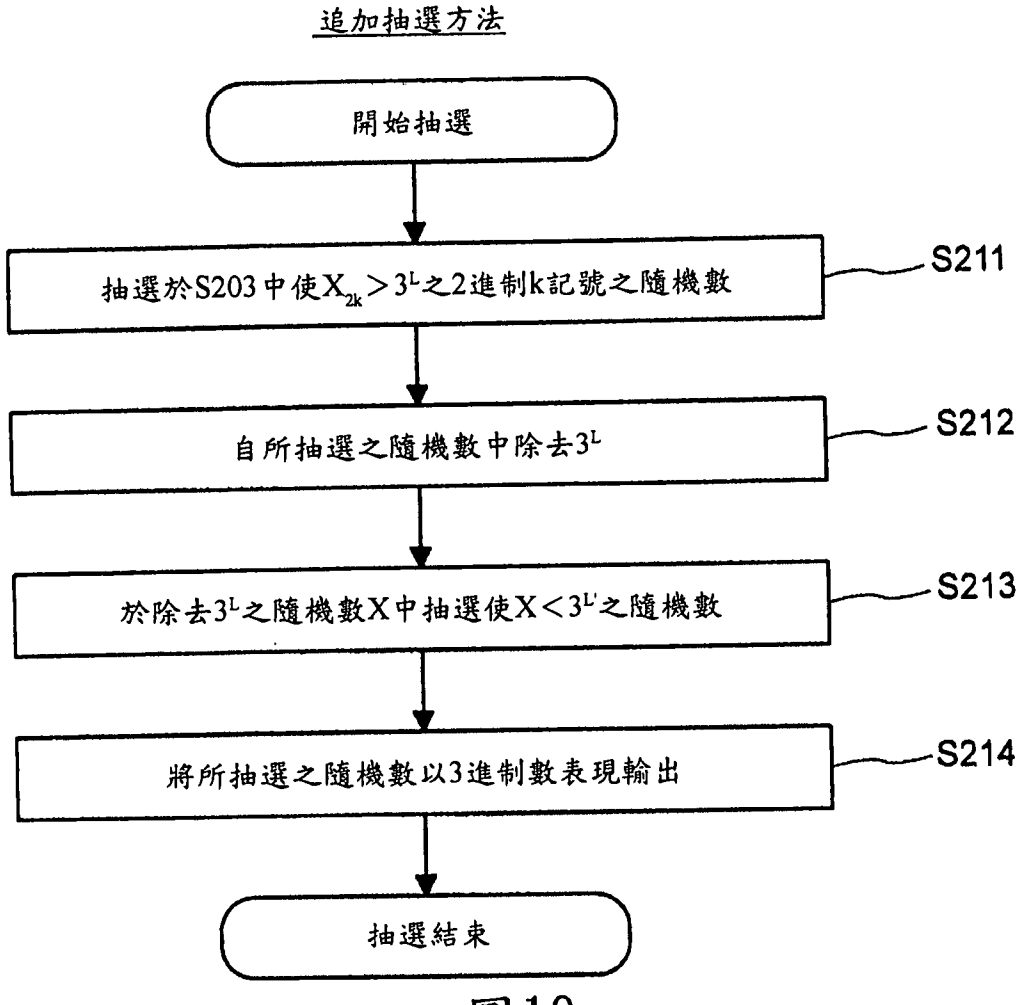

追加抽選方法
開始抽選
抽選於S203中使$X_{2k} > 3^L$之2進制k記號之隨機數
S211
自所抽選之隨機數中除去3^L
S212
於除去3^L之隨機數X中抽選使$X < 3^{L'}$之隨機數
S213
將所抽選之隨機數以3進制數表現輸出
S214
抽選結束

圖19

構造化方法#1

$$f_1(x_1,\cdots,x_n)=\sum_{i,j}a_{1ij}x_ix_j+\sum_i b_{1i}x_i$$

$$\vdots$$

$$f_m(x_1,\cdots,x_n)=\sum_{i,j}a_{mij}x_ix_j+\sum_i b_{mi}x_i$$

資料構造A　　資料構造B

圖20

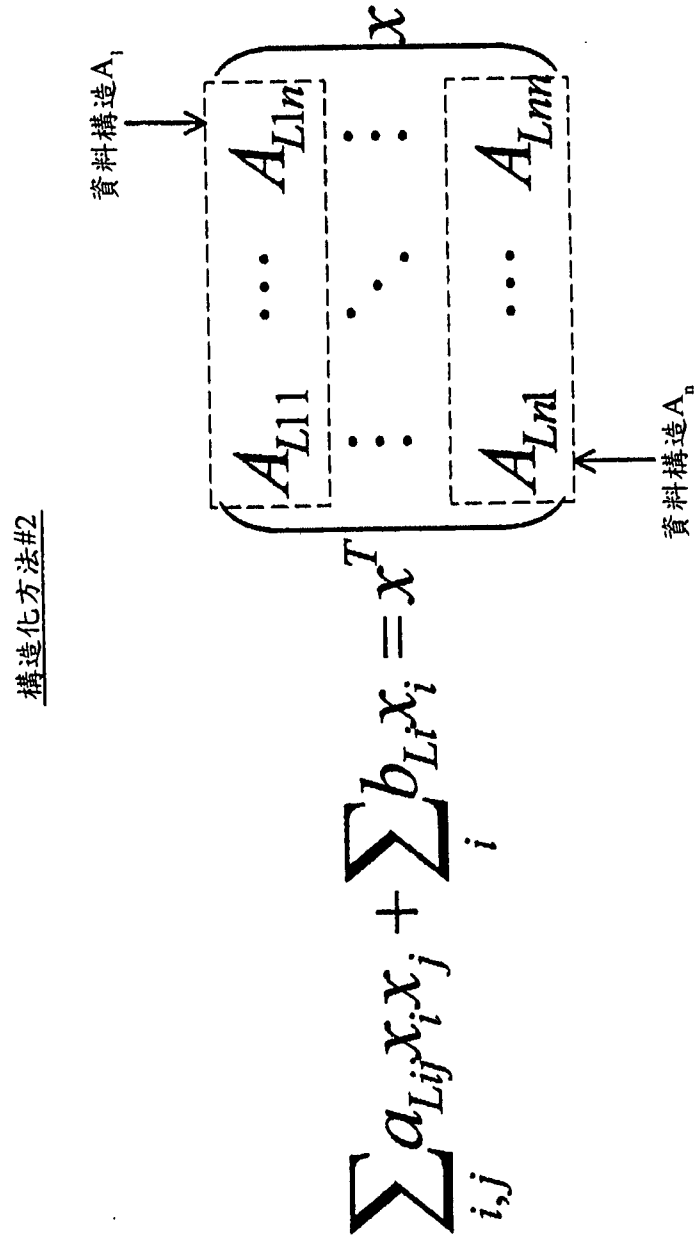

構造化方法#2
資料構造A1
資料構造An
$\sum_{i,j} a_{Lij} x_i x_j + \sum_i b_{Li} x_i = x^T \begin{pmatrix} A_{L11} & \cdots & A_{L1n} \\ \vdots & \ddots & \vdots \\ A_{Ln1} & \cdots & A_{Lnn} \end{pmatrix} x$

圖21

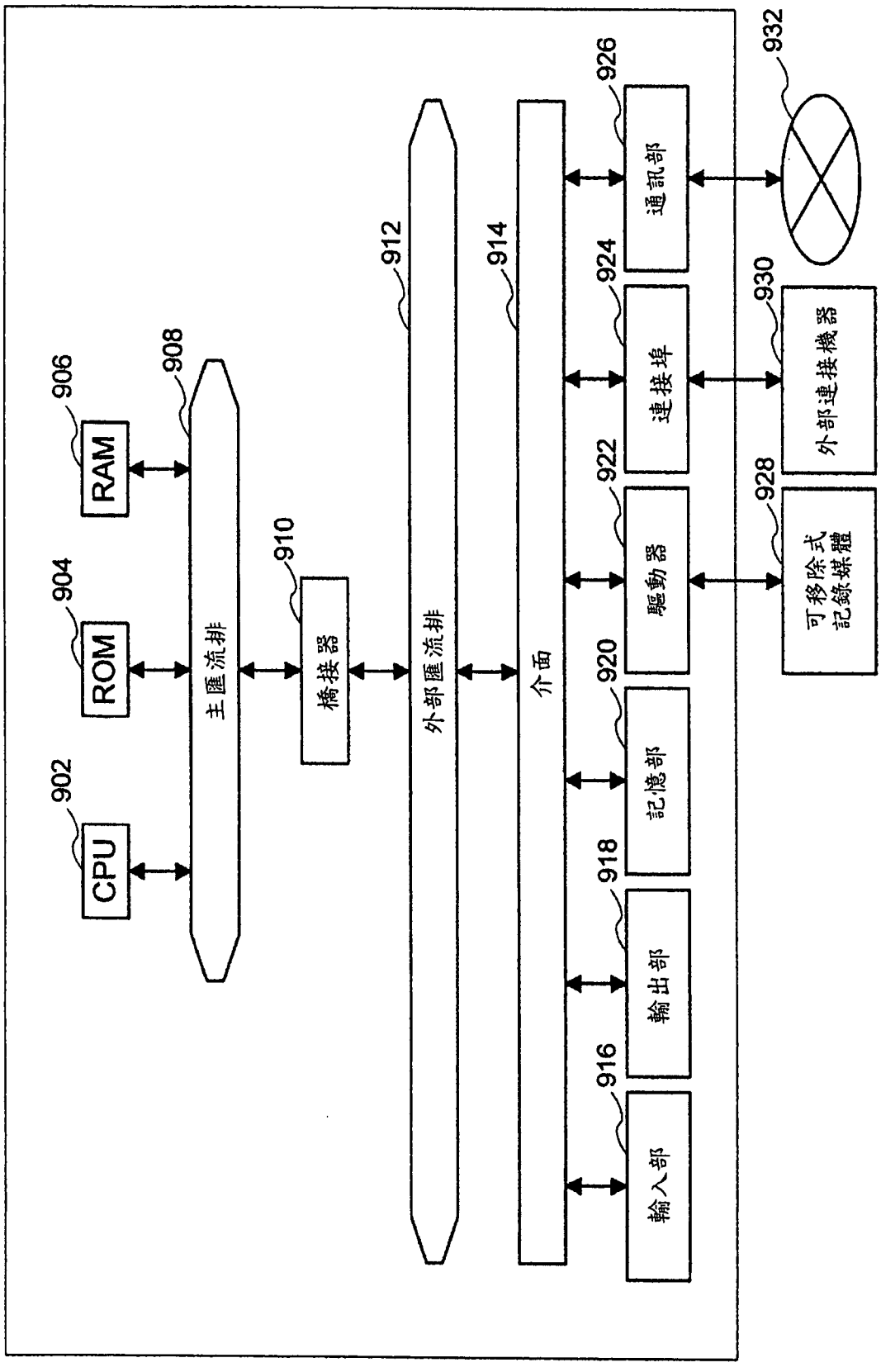

902
CPU
904
ROM
906
RAM
908
主匯流排
910
橋接器
912
外部匯流排
914
介面
916
輸入部
918
輸出部
920
記憶部
922
驅動器
924
連接埠
926
通訊部
928
可移除式記錄媒體
930
外部連接機器
932

圖22