說明書發明人續頁

發明人 2

姓名：（中文）史蒂芬妮 L. 荷納

（英文）STEPHANIE L. HIRNAK

住居所地址：（中文）美國新罕布夏州貝德佛市紐貝利路 11 號

（英文）11 NEWBURY LANE, BEDFORD, NEW HAMPSHIRE 031102, U.S.A

國籍：（中文）美國 （英文）U.S.A

捌、聲明事項

☐ **本案係符合專利法第二十條第一項☐第一款但書或☐第二款但書規定之期間，其日期為：** _____

☑ **本案已向下列國家（地區）申請專利，申請日期及案號資料如下：**

【格式請依：申請國家（地區）；申請日期；申請案號　順序註記】

1. 美國；2001 年 09 月 28 日；09/966,349
2. _____
3. _____

☑ **主張專利法第二十四條第一項優先權：**

【格式請依：受理國家（地區）；日期；案號　順序註記】

1. 美國；2001 年 09 月 28 日；09/966,349
2. _____
3. _____
4. _____
5. _____
6. _____
7. _____
8. _____
9. _____
10. _____

☐ **主張專利法第二十五條之一第一項優先權：**

【格式請依：申請日；申請案號　順序註記】

1. _____
2. _____
3. _____

☐ **主張專利法第二十六條微生物：**

☐國內微生物　【格式請依：寄存機構；日期；號碼　順序註記】

1. _____
2. _____
3. _____

☐國外微生物　【格式請依：寄存國名；機構；日期；號碼　順序註記】

1. _____
2. _____
3. _____

☐熟習該項技術者易於獲得，不須寄存。

序圖。

發明之詳細說明

參考圖1，通信系統10包括一平行的、硬體式的多重執行緒處理器12。該硬體式多重執行緒處理器12會耦合至一匯流排(例如PCI匯流排14)、一記憶體系統16及一第二匯流排18。該系統10特別適用於能夠分割成數項平行子工作或功能的工作中。明確地說，該硬體式多重執行緒處理器12適用於頻寬導向而非等待時間導向的工作中。該硬體式多重執行緒處理器12具有數個微引擎22，其能夠同時活動，並且能夠作業於獨立作業於一項工作中的多重硬體控制執行緒之上。

該硬體式多重執行緒處理器12亦包括一中央控制器20，其可幫助載入該硬體式多重執行緒處理器12其它資源的微碼控制，並且執行其它一般用途電腦類型的功能，例如操控協定、例外、以及當該些微引擎完成封包傳送之後，為作更細部的處理(例如在邊界條件處)所進行的封包處理額外支援。在其中一具體實施例中，該處理器20係Strong Arm®的架構。該一般用途微處理器20具有一作業系統。透過該作業系統，處理器20便能夠呼叫函數作業於微引擎22a-22f。該處理器20能夠使用任何支援的作業系統，較佳的係即時的作業系統，例如MicrosoftNT即時版、VXWorks。

該硬體式多重執行緒處理器12亦包括複數個函數微引擎22a-22f。功能性微引擎(微引擎)22a-22f各會在硬體中

保持複數個程式計數，以及與該些程式計數相關聯的狀態。有效的方式是，相關的複數組執行緒能夠同時活動於每一個微引擎22a-22f之上，不過，每次僅有一個能夠真正作業。

如圖所示，在其中一具體實施例中共有六個微引擎22a-22f。每個微引擎22a-22f都能夠處理四條硬體執行緒。該六個微引擎22a-22f可與共享資源(包括記憶體系統16，匯流排介面24及28)共同作業。該記憶體系統16包括同步動態隨機存取記憶體(SDRAM)控制器26a及靜態隨機存取記憶體(SRAM)控制器26b。SDRAM記憶體16a及SDRAM控制器26a通常是用於處理大量的資料，例如處理網路封包中的網路酬載。SRAM控制器26b及SRAM記憶體16b通常是使用於低等待時間、快速存取工作(例如存取核心處理器20的查值表、記憶體)等的網路環境中。

該六個微引擎22a-22f會根據資料的特徵存取SDRAM記憶體16a或SRAM記憶體16b。因此，低等待時間、低頻寬資料係儲存於SRAM之中，並且從SRAM中擷取；而等待時間較不重要的高頻寬資料則係儲存於SDRAM之中，並且從SDRAM中擷取。該些微引擎22a-22f都能夠執行記憶體參考指令，用以傳送給SDRAM控制器26a或SRAM控制器26b。

藉由SRAM或SDRAM記憶體存取方式便能夠解釋硬體多重執行緒設計的優點。舉例來說，來自微引擎的Thread_0要求進行SRAM存取時，其便會讓SRAM控制器

26b開始存取SRAM記憶體16b。SRAM控制器便會控制SRAM匯流排的裁定作業、存取SRAM 16b、從該SRAM 16b擷取資料、並且將資料回傳給提出要求的微引擎22a-22b。在SRAM存取期間，如果微引擎(例如22a)僅有單條執行緒能夠作業的話，該微引擎便可能會停止運作，直到資料從該SRAM回傳為止。藉由每個微引擎22a-22f內的硬體式資料內容調換技術，該硬體式資料內容調換技術便能夠在同一個微引擎中執行其它的具有唯一程式計數器的資料內容。因此，當第一條執行緒(例如Thread_0)正在等待時間待所讀取的資料進行回傳的時候，另一條執行緒(例如Thread_1)便能夠繼續作業。在執行期間，Thread_1可能會存取SDRAM記憶體16a。當Thread_1對SDRAM單元進行作業，而且Thread_0正在對SRAM單元進行作業時，另一條新的執行緒(例如Thread_2)可能正在微引擎22a中進行作業。Thread_2可能會作業一段特定的時間，直到其需要存取記憶體或執行其它長等待時間的作業為止，例如存取匯流排介面。所以，處理器12能夠利用一微引擎22a同時完成或進行匯流排作業、SRAM作業及SDRAM作業，而且還有另一條執行緒可用以在資料路徑中處理更多的工作。

硬體式資料內容調換技術亦可同時完成各項工作。舉例來說，可能會有兩條執行緒存取同一個共享資源，如SRAM。當完成其中一個微引擎執行緒資料內容所要求的工作之後，每個分離的功能性單元(例如，FBUS介面28、

SRAM控制器26a及SDRAM控制器26b)便會回報一完成作業的旗標信號。當該微引擎接收到旗標之後，其便會判斷應該開啟哪一條執行緒。

硬體式的多重執行緒處理器12的其中一種實例為網路處理器。以網路處理器來說，硬體式的多重執行緒處理器12會介接各種如媒體存取控制器裝置(例如10/100 BaseT八進位媒體存取控制器13a或Gigabit乙太裝置13b)般的網路裝置。一般來說，該網路連接過程會介接任何類型的通信裝置，或介接可接收/傳送大量資料的介面。作業於網路應用中的通信系統10能夠從裝置13a、13b接收複數個網路封包，並且以平行的方式處理該些封包。利用該硬體式多重執行緒處理器12，便能夠獨立地處理每個網路封包。處理器12的另一種使用實例是幕後排版處理器的列印引擎，或是儲存子系統(即RAID碟片儲存機)的處理器。進一步的使用實例是作為匹配引擎。舉例來說，在業界的安全交易中，電子交易的出現使得需要使用電子匹配引擎來匹配買賣雙方的訂單。所有平行的工作類型都能夠在系統10中完成。

處理器12包括一匯流排介面28，用以將該處理器耦合至該第二匯流排18。在其中一具體實施例中，匯流排介面28會將該處理器12耦合至所謂的FBUS 18(FIFO匯流排)。該FBUS介面28係負責控制該處理器12，並將其耦合至FBUS 18。該FBUS 18係一64位元寬的FIFO匯流排，其是目前可接受的媒體存取控制器(MAC)裝置的最佳匯流排。

該處理器12包括一第二介面(例如PCI匯流排介面24)，用以將該PCI 14匯流排中的其它系統組件耦合至該處理器12。該PCI匯流排介面24可提供記憶體16(例如SDRAM記憶體16a)一條高速的資料路徑24a。透過該資料路徑，便可以直接記憶體存取(DMA)傳輸方式，經由該PCI匯流排14從SDRAM 16a快速地移動資料。此外，該PCI匯流排介面24亦支援目標作業及主作業。目標作業的作業方式是，匯流排14中的從裝置係透過讀取及寫入的方式存取該目標作業中作為從屬裝置的SDRAM。在主作業中，處理器核心20會直接傳送資料給該PCI介面24，或是從該PCI介面24直接接收資料。

每個功能性單元都會耦合至一或多個內部匯流排。該處理器包括一AMBA匯流排，用以將處理器核心20耦合至記憶體控制器26a、26b以及下面所述的AMBA轉譯器30。該處理器亦包括一私有匯流排34，用以將微引擎單元耦合至SRAM控制器26b、AMBA轉譯器30及FBUS介面28。記憶體匯流排38會將記憶體控制器26a、26b耦合至匯流排介面24及28以及記憶體系統16(其包括用以執行開機作業的快閃唯讀記憶體16c)。

參考圖2-1至2-4，每個微引擎22a-22f都包括一仲裁器，用以檢查旗標，以決定可於其上作業的執行緒。來自微引擎22a-22f的任一條執行緒都能夠存取SDRAM控制器26a、SRAM控制器26b或FBUS介面28。記憶體控制器26a及26b各具有複數個佇列，用以儲存重要的記憶體參考要

前置充電。如果在奇數及偶數儲存單元之間交替進行記憶體參考的話，便有可能進行前置充電。排列記憶體參考順序以交替存取相反的儲存單元，該處理器12便能改良SDRAM的頻寬。

FBUS介面28對於MAC裝置所支援的每個埠都支援傳輸接收旗標，以及表示服務何時可獲保證的中斷旗標。FBUS介面28亦包括一控制器28a，用以對來自FBUS 18的輸入封包進行標頭處理。控制器28a會抽出封包標頭，並且在SRAM中執行可微控的來源/目的/協定雜湊查詢表(供用於位址平整)。如果無法成功分解該項雜湊的話，便會將該封包標頭送至處理器核心20進行額外的處理。FBUS介面28支援下面的內部資料異動：

從FBUS單元　至處理器核心/從處理器核心至FBUS　(透過AMBA匯流排)。

從FBUS單元　至SRAM單元/從SRAM單元至FBUS　(透過私有匯流排)。

從FBUS單元　至SDRAM/從SDRAM至FBUS　(透過Mbus)。

該FBUS 18係一標準的工業匯流排，其包括一資料匯流排(例如64位元寬)以及一用以進行位址及讀取/寫入控制的側頻帶控制。FBUS介面28提供利用一系列的輸入及輸出FIFO 29a-29b以輸入大量的資料。從FIFO 29a-29b中，微引擎22a-22f便能夠從SDRAM控制器26a擷取資料，或是命令其將資料從所接收的FIFO(其中的資料係來自匯流排18中的裝置)移至FBUS介面28之中。資料可透過記憶體控制器26a，以直接記憶體存取方式傳送至SDRAM記憶體

16a。同樣地，微引擎能夠將資料從SDRAM 26a移動至介面28，透過介面28中的FBUS移出至FBUS 18。

資料功能則是散布於該些微引擎之間。與SRAM 26a、SDRAM 26b及FBUS 28的連接都是透過命令要求的方式進行。一命令要求可能是記憶體要求或FBUS要求。舉例來說，命令要求可將資料從微引擎22a中的暫存器移至共享資源，例如SDRAM處、SRAM處、快閃記憶體或某個MAC位址。該些命令會被外送至每個功能性單元及共享資源。不過，該些共享資源並不需要以區域緩衝的方式保留該資料。相反地，該些共享資源會存取位於該些微引擎內的散佈資料。這使得微引擎22a-22f能夠對資料進行區域存取，而不必進行匯流排中的存取仲裁並且發生競爭匯流排的風險。利用此特點，不必花費任何循環的延遲時間以等待該些微引擎22a-22f的內部資料。

該些用以耦合共享資源(例如記憶體控制器26a及26b)的資料匯流排(例如AMBA匯流排30、SRAM匯流排34及SDRAM匯流排38)必須有足夠的頻寬，方能不產生內部瓶頸。因此，為避免發生瓶頸，處理器12的頻寬條件為每個功能性單元的頻寬至少為內部匯流排最大頻寬的兩倍。舉例來說，SDRAM能夠在83 MHz時於64位元寬的匯流排中運作。該SRAM資料匯流排具有分離的讀取及寫入匯流排，例如運作於166 MHz的32位元寬的讀取匯流排，以及運作於166 MHz的32位元寬的寫入匯流排。也就是，64位元基本上是運作於166 MHz(其為該SDRAM的兩倍頻寬)之

中。

該核心處理器20亦能夠存取該些共享資源。該核心處理器20可透過匯流排32與SDRAM控制器26a、匯流排介面24及SRAM控制器26b進行直接溝通。不過，為存取微引擎22a-22f並且傳輸位於任一微引擎22a-22f中的暫存器，該核心處理器20可透過匯流排34以經由AMBA轉譯器30存取微引擎22a-22f。AMBA轉譯器30實體係屬於FBUS介面28，但是邏輯上卻有區別。該AMBA轉譯器30能夠執行FBUS微引擎傳輸暫存器位置及核心處理器位址(即AMBA匯流排)之間的位址轉譯，讓該核心處理器20能夠存取屬於微引擎22a-22c的暫存器。

該處理器核心20包括一以五階段管線實現的RISC核心50，用以在單循環中執行一個運算元或兩個運算元的單循環移位，提供乘法支援及32位元的桶移位支援。該RISC核心50是標準的Strong Arm®架構，不過為提高效能，其係以五階段管線來實現。該處理器核心20亦包括一16千位元組的指令快取記憶體52、一8千位元組的資料快取記憶體54及一預取流動緩衝器56。該處理器核心20會同時執行算術運算、記憶體寫入及指令擷取。該核心處理器20可透過ARM界定的AMBA匯流排與其它的功能性單元介接。該AMBA匯流排係一32位元的雙向匯流排32。

參考圖3，圖中所示的多重執行緒處理器12正在執行網路路由功能。在其中一實例中，非同步傳輸模式(ATM)、乙太網路及其它類型的封包都會通過網路介面MAC裝

置，並且被傳送至網路處理器12。該些封包係在一般用途微處理器20中或是在透過PCI匯流排介面(未顯示)耦合的另一個處理器中的應用程式內進行處理。為接收及傳輸該些封包，運作於處理器20或是透過PCI匯流排耦合的處理器中的應用程式會使用到網路堆疊72，其包括網路管理、控制及信號處理74，用以管理網路通信。

該網路堆疊72及該應用程式都是在控制該些微引擎的處理器20或是耦合至PCI匯流排的其它處理器中執行。接收、傳輸及資料遞送的路徑代表的是透過該處理器12運送封包。該網路管理、信號處理及該網路堆疊72通常不會專門處理資料遞送。基本上，處理器20會接收及傳輸。該處理器20會產生新的封包，用以透過該網路進行傳輸。在例外的情形中，該處理器20可能會專門處理資料遞送。其可能包含需要特別處置及複合處理的極稀有封包。

對資料遞送處理來說，會使用到微引擎22a-22f。在某些情況中，可能會在一般用途的處理器20等級中進行資料遞送。信號Init是係程式設計者用以啟動微引擎碼的介面。信號Fini則是用以進行終止(將控制資訊放置在已知的狀態中)。微引擎22a-22f能夠提供快速的儲存及遞送能力。該些引擎會使用多層式一般查對處理，用以透過平行硬體支援的處理執行緒執行驗證、分類、政策規劃及過濾。例外及控制封包會傳送至處理器20，用以在網路堆疊72中進行處理。透過PCI埠或裝置埠可將三元網路堆疊(未顯示)放置在晶片外的主機端。藉此便能夠減少處理器20

的負載，或是減少某個地方之集中管理與控制的負載。在某些具體實施例中，該微引擎係一小型的RISC處理器並且具有有限的指令空間。基於該些及其它原因，當執行多協定時，吾人會希望能夠降低指令碼的大小。網路處理器12可實現一般遞送處理，其能夠在不超過指令儲存限制下，處置各種的協定類型(現有的及未來的類型)。

現在參考圖4，所示的係用以遞送儲存在記憶體中的表格結構90的管理安排80。該遞送表格結構管理80包括一控制及管理結構82，其包括一網路堆疊介面84及一表格管理器86。該表格管理器86會管理儲存在SRAM中的路由表格90，並且包括複數個如圖4所示的表格，其包括層4連接表92、層3目的地表94、層2橋接表96及層2連接表98。此外，儲存在記憶體中的資料結構包括儲存在DRAM中的封包緩衝區100。作為封包資料遞送處理器的微引擎可從SRAM中的路由表格90中擷取資訊、儲存以及從DRAM的封包緩衝區中遞送該封包資訊。該些多個表格90都是由控制管理處理器20來設定。舉例來說，層2橋接表96可供ATM虛擬電路、訊框中繼連接MPLS標籤或其它低階連接使用。層2橋接表96亦可供乙太網路橋接使用。層3目的地表94可供依照目的地IP位址進行遞送的網際網路協定(IP)使用。層4連接表92可供依照來源及目的地埠、位址及協定進行遞送的IP使用。所有的表格可能都必須對封包進行解封或加封。

當表格90依慣用的方式充滿遞送資訊之後，封包資料遞

送處理器便能夠接收封包、執行表格查詢，以便取得資訊，並且如該表格項目所要求般地轉換封包。該控制管理處理器會以共同的格式設定該些表格90，以達到解封及加封的目的。

現在參考圖5，所示的係示範的表格項目，其中的子集係包含於每個表格90之中。該些表格項目包括下面的欄位：

遞送表格格式

Decap Flag.表示應該於何時從封包中剝離該些位元組。如果宣告該旗標的話，Decap Byte Count欄位中便是欲剝離的位元組數量。

Decap To Layer.此欄位規定的是解封至該規定層的標頭層。剖析該封包標頭便可決定該層的長度及解封方式。

Decap Byte Count.此欄位規定的是從該封包前面移除的位元組數量。藉由調整封包緩衝區中的封包起始偏移值便可執行解封。

Current Encap.此欄位規定的是目前封包加封類型的識別符號。

Encap Flag.表示位元組是否應該當作該封包。如果宣告此旗標的話，Encap Byte Count欄位中便是其位元組數量，而Encap Header欄位中的則是欲加封的位元組。

Encap Byte Count.欲當作該封包的位元組數量。

Encap Header.欲當作該封包的實際位元組。

Next Table Type.如果非零值的話，即表示需要一另外的查

詢表。其提供的是表格的類型。舉例來說，層3路由或層4連接表格類型。層3路由查詢可能會使用一種運用到目的地IP位址的最長前置碼匹配查詢表演算法。層4連接查詢表可能會使用一種運用到來源及目的地位址、來源及目的地埠及協定的104位元雜湊演算法。

Next Table Addr.可能會有數個次表格，以及數個相同類型的次表格。此欄位便是規定該表的基位址。

藉由管理過程便可設定或清除該些旗標。信號處理及設定連接都是該網路系統的一部份，其會決定出某一條經過該網路、需要改變標頭的路徑。需要改變標頭的原因有許多種。通常當該協定從一網域改變成另一個網域時便需要改變標頭。

現在參考圖6，所示的係用以加封/解封一般協定的處理程序110。一開始，其中一個微引擎22a-22f會從網路介面接收112一封包。該封包係由一或多個標頭後面加上酬載所構成的。該微引擎(例如22a)會將該封包的酬載部份複製到DRAM的封包緩衝區中，並且其可能會將該封包放置在該緩衝區的偏移區，以便挪出空間供任何新的標頭(其可能是當作封包)使用，用以進行封包遞送。該封包的封包偏移值參數會設定成內定值，其係決定於該緩衝區的偏移區中。該微引擎會讀進114該封包的第一標頭，並且執行層2查詢表。層2查詢表會讀取該表格的層2橋接表及/或層2連接表。該表格會回傳各種參數，例如decap flag或encap flag。該處理程序110會判斷116 decap flag或encap flag是

否已經設定。如果decap flag及encap flag皆已經設定的話，該處理程序便會將decap byte count加入118該封包起始偏移值中，並且從該封包起始偏移值中扣除120 encap byte count，然後將該些encap位元組當作該封包。該處理程序110會藉由檢視目前所讀取之表格中的空白欄位，測試122是否有次表格要檢查。如果有有次表格的話，該處理程序110便會剖析124該次標頭，擷取及讀取該次表格。該處理程序110會繼續運作，以測試欲設定的decap flag或encap flag。

不過，如果該處理程序判斷出並非decap flag及encap flag都已經被設定的話(116上方)，其便會判斷是否encap flag 130或decap flag 132兩者其中之一被設定。如果encap flag被設定的話，其便會從該起始偏移值中扣除120 encap byte count，然後將該些encap位元組當作該封包。相反地，如果僅有decap flag被設定132的話，該處理程序便會將decap byte count加入134該緩衝區偏移值中，並且一定會檢查次表格112。當該處理程序判斷出已經結束檢查表格，接著便會對該封包進行分類，並且以慣用的方式進行遞送136。也就是「no」的情況表示該處理程序可以進行分類及遞送。遞送標頭可讓微引擎接收該標頭，並將其傳送給處理器20或其它地方，因此其能夠與酬載進行重組。遞送該標頭亦包含遞送該封包之類。

現在參考圖7，除了規定從查詢表中所獲得的位元組散佈計數之外，該查詢表可能還會在該表格中設定decap to layer欄位。此欄位規定的是應該解封至某一層的封包前面

部份。吾人熟知的係，封包係定義在OSI(開放系統互連)七層網路協定的協定層中。經過實體層1之後，網路處理層所看見的第一軟體層便是層2，其亦稱為連結層。在該新封包起始層之前剖析該些封包層，便可決定欲解封的位元組長度。該長度可加入到封包起始偏移值中。

圖7所示的係一種變化範例，其中該表格中並未規定其解封長度，必須藉由讀取該封包本身方能決定。換言之，其可能是一組能夠插入圖6處理程序中的標準程序，用以將從該封包所加封的位元組計數代入該偏移值中。

如圖7所示，處理程序140便係用以決定此偏移值。該處理程序140包括讀取表格142；判斷decap to packet layer位元是否已經設定144；如果已經設定的話，便藉由剖析標頭以擷取出欲移除的該層長度146；以及，將該長度加入該封包起始偏移值中148。如果該decap layer並未設定的話，該處理程序便會單純地略過。無論如何，此處理程序都能夠配合圖6所述的處理程序。

一般的decap to layer位元使用方式係用以規定向上解封至層3的IP標頭。如果該封包加封係ATM網路中的多協定的話(例如RFC 1483標準)，那麼，使用RFC 1483長度規則，剖析該層至標頭本身便可決定層2的標頭長度。不過，如果該封包加封係典型的IP的話，便可遵循典型的IP層長度規則決定層2的長度。藉由該埠的類型(其係來自該埠之假設慣用標頭)便可知道該封包加封方式，或是從目前的encap欄位中的第一查詢表亦可知道該封包加封方式。

此項技術提供的是一種一般性的方法，而非提供每一種用以界定分離協定轉換的網路協定。在替代的具體實施例中，此項技術能夠以軟體資料庫標準程序的方式來實現，例如用以進行解封/加封的一般軟體建構區塊，其中，客戶可插入其私有的標頭加封，而且客戶的合作廠商並不需要涉入客戶私有的協定設計中。

已說明本發明的一些具體實施例。但是，應明白，只要不違反本發明的精神與範疇，即可進行各種修改。因此，其它具體實施例都屬於下列申請專利範圍的範疇內。

肆、中文發明摘要

本發明敘述一種遞送封包的方法。該方法包括讀取一份含有複數個旗標的表格，用以決定該等複數個旗標中何者應該設定或清除；並且對該封包執行作業，以便根據該些旗標值解封或加封該封包。

伍、英文發明摘要

A method of forwarding a network packet is described. The method includes reading a table containing a plurality of flags to determine which of the plurality of flags is set or cleared and performing an operation on the packet to decapsulate or encapsulate the packet in accordance with values of the flags.

陸、(一)、本案指定代表圖為：第_____圖

(二)、本代表圖之元件代表符號簡單說明：

柒、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

拾壹、圖式

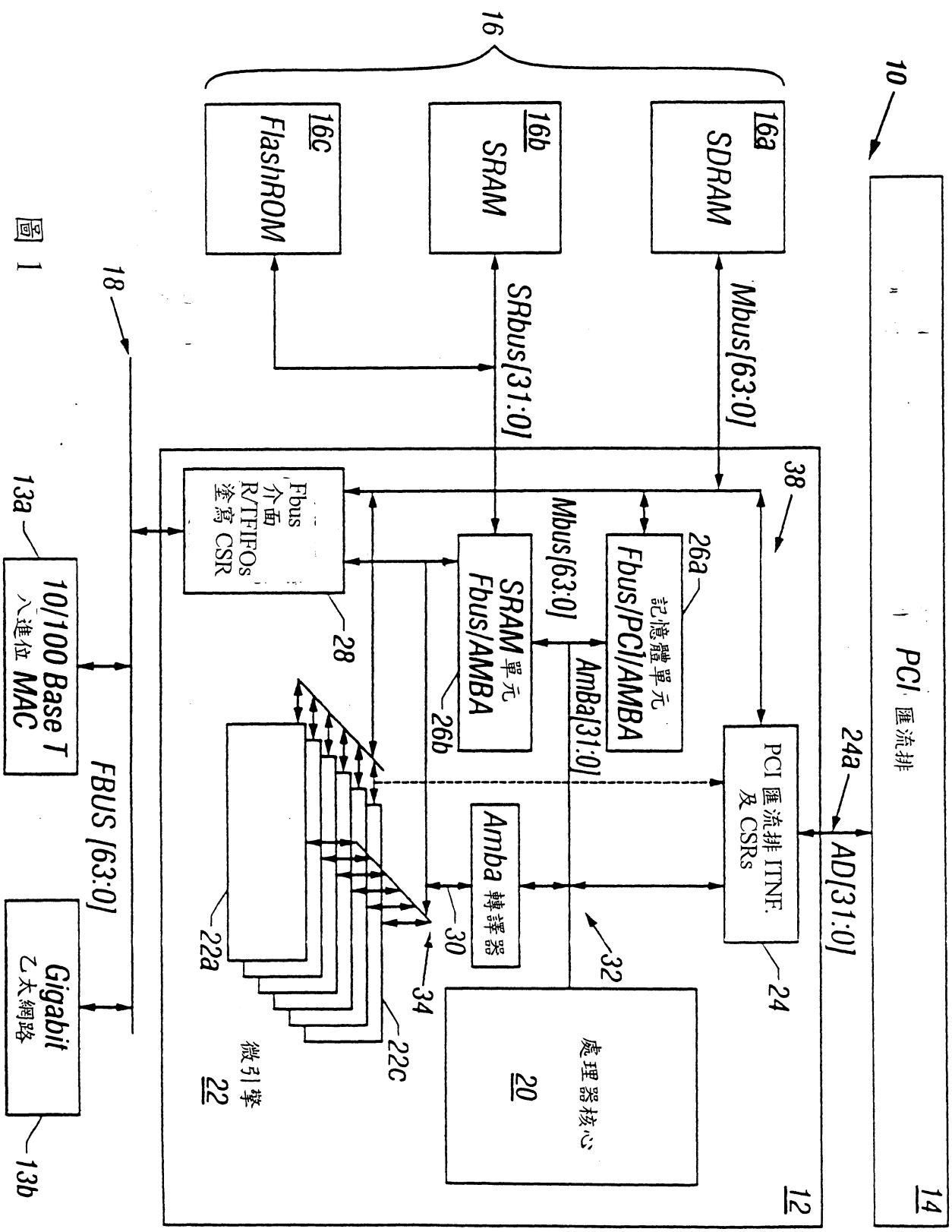

10
PCI 匯流排
14
AD[31:0]
24a
PCI 匯流排 ITNF. 及 CSRs
24
12
38
16
16a SDRAM
16b SRAM
16c FlashROM
Mbus[63:0]
SRbus[31:0]
26a 記憶體單元 Fbus/PCI/AMBA
Mbus[63:0]
AmBa[31:0]
SRAM 單元 Fbus/AMBA
26b
Fbus 介面 R/TFIFOs 塗寫 CSR
28
Amba 轉譯器
30
32
處理器核心 20
34
微引擎 22
22a
22c
18
FBUS [63:0]
13a
10/100 Base T 入進位 MAC
Gigabit 乙太網路
13b

圖 1

圖式續頁

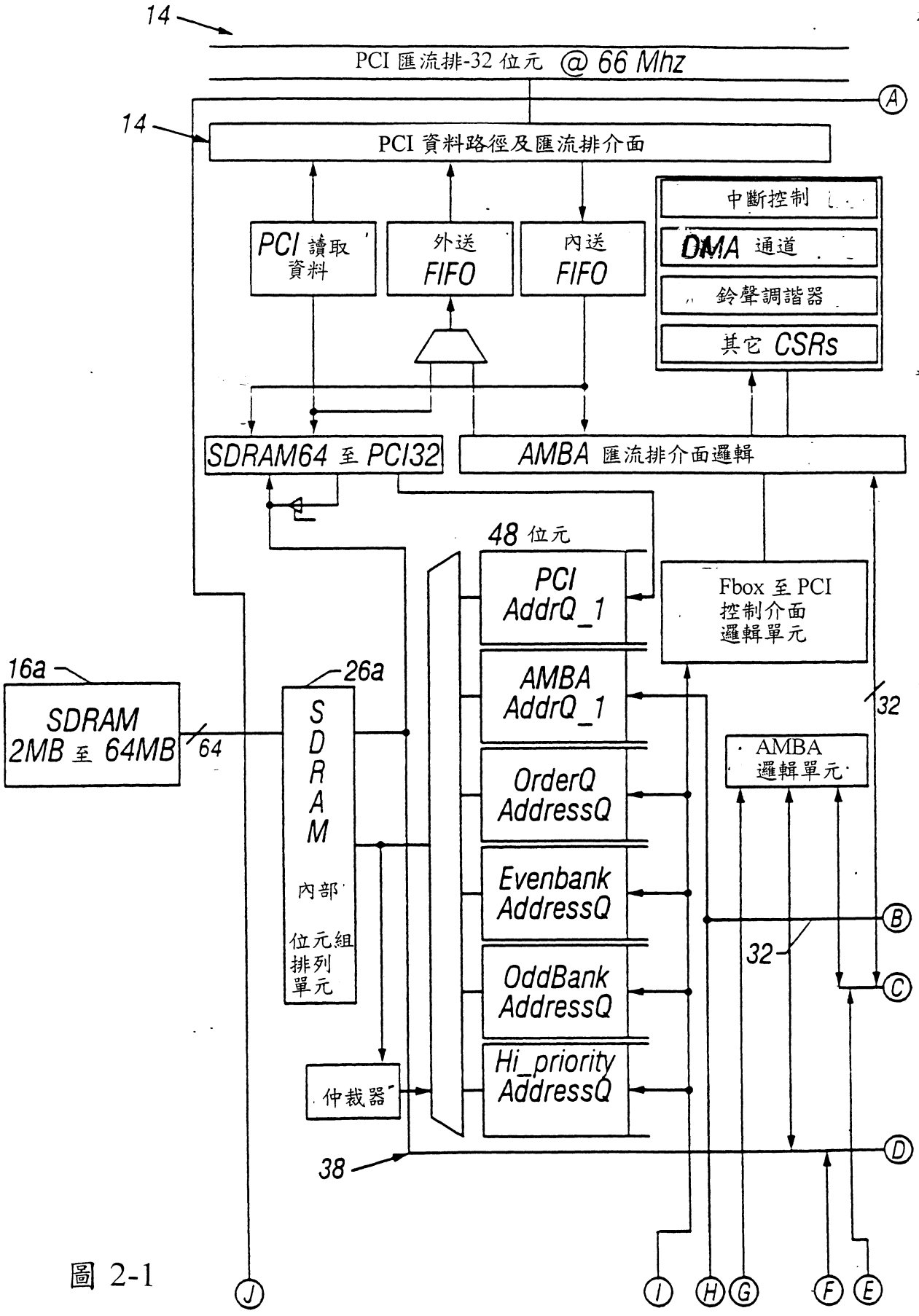

14
PCI 匯流排-32 位元 @ 66 Mhz
A
14
PCI 資料路徑及匯流排介面
中斷控制
DMA 通道
鈴聲調諧器
其它 CSRs
PCI 讀取資料
外送 FIFO
內送 FIFO
SDRAM64 至 PCI32
AMBA 匯流排介面邏輯
48 位元
PCI AddrQ_1
AMBA AddrQ_1
OrderQ AddressQ
Evenbank AddressQ
OddBank AddressQ
Hi_priority AddressQ
Fbox 至 PCI 控制介面邏輯單元
AMBA 邏輯單元
32
32
16a
SDRAM 2MB 至 64MB
64
26a
SDRAM 內部 位元組排列單元
仲裁器
38
B
C
D
J
I
H
G
F
E

圖 2-1

圖式頁

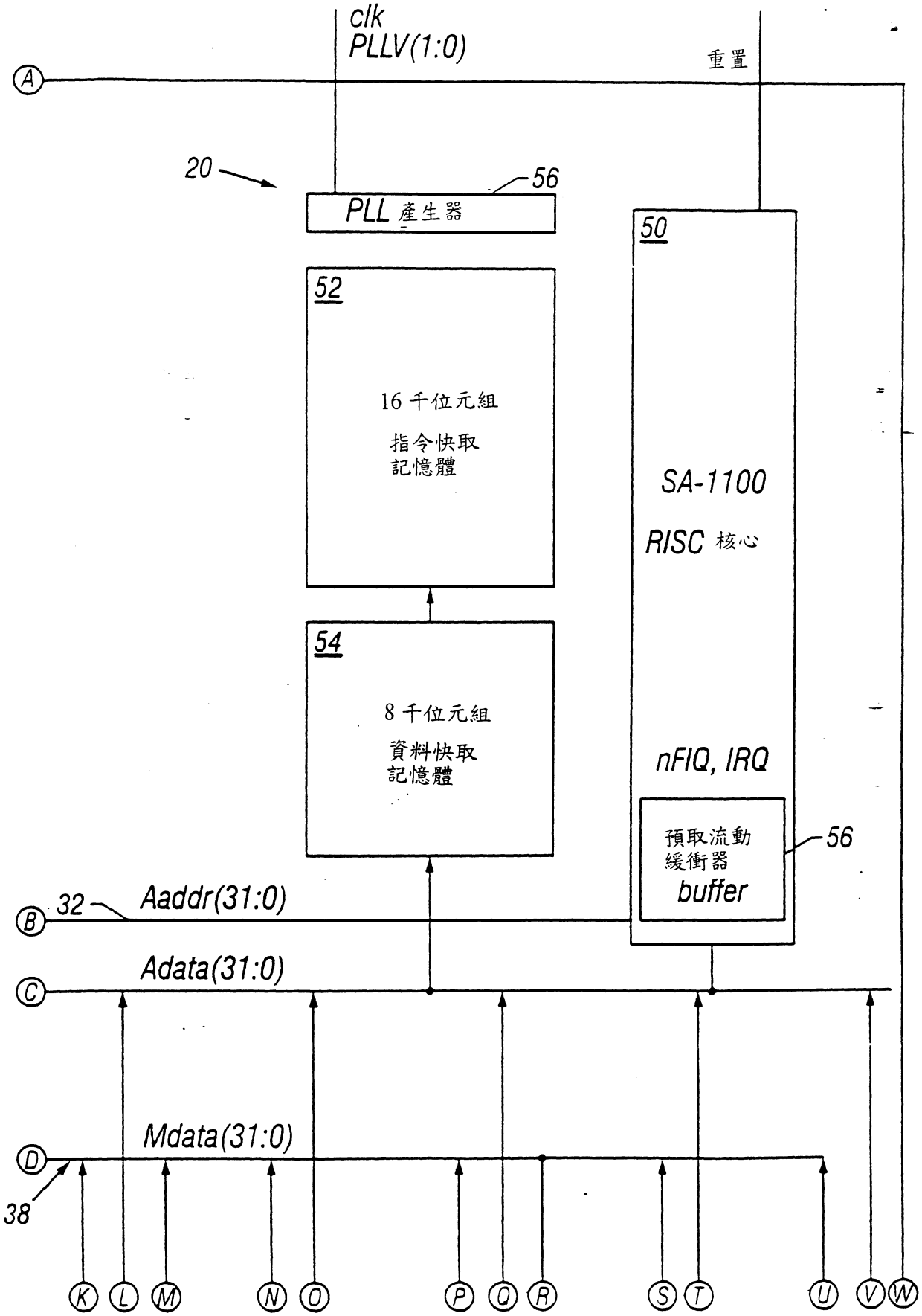

clk
PLLV(1:0)
重置
A
20
56
PLL 產生器
50
52
16 千位元組
指令快取
記憶體
SA-1100
RISC 核心
54
8 千位元組
資料快取
記憶體
nFIQ, IRQ
預取流動
緩衝器
buffer
56
B
32
Aaddr(31:0)
C
Adata(31:0)
Mdata(31:0)
D
38
K
L
M
N
O
P
Q
R
S
T
U
V
W

圖 2-2

圖式續頁

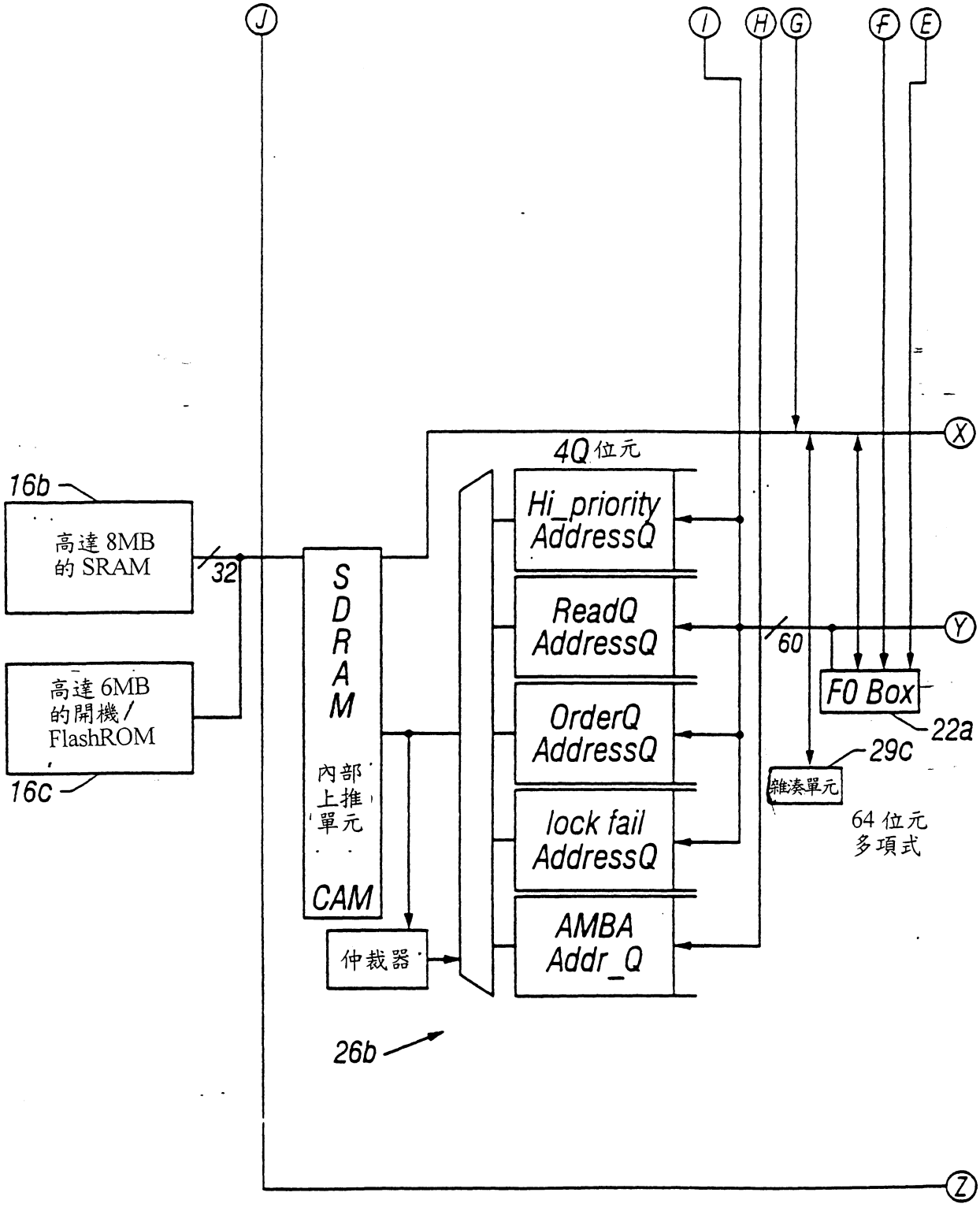

J
I
H
G
F
E
X
Y
Z
4Q 位元
16b
高達 8MB
的 SRAM
32
高達 6MB
的開機/
FlashROM
16c
SDRAM
內部
上推
單元
CAM
仲裁器
26b
Hi_priority
AddressQ
ReadQ
AddressQ
OrderQ
AddressQ
lock fail
AddressQ
AMBA
Addr_Q
60
F0 Box
22a
29c
雜湊單元
64 位元
多項式

圖 2-3

圖式續頁

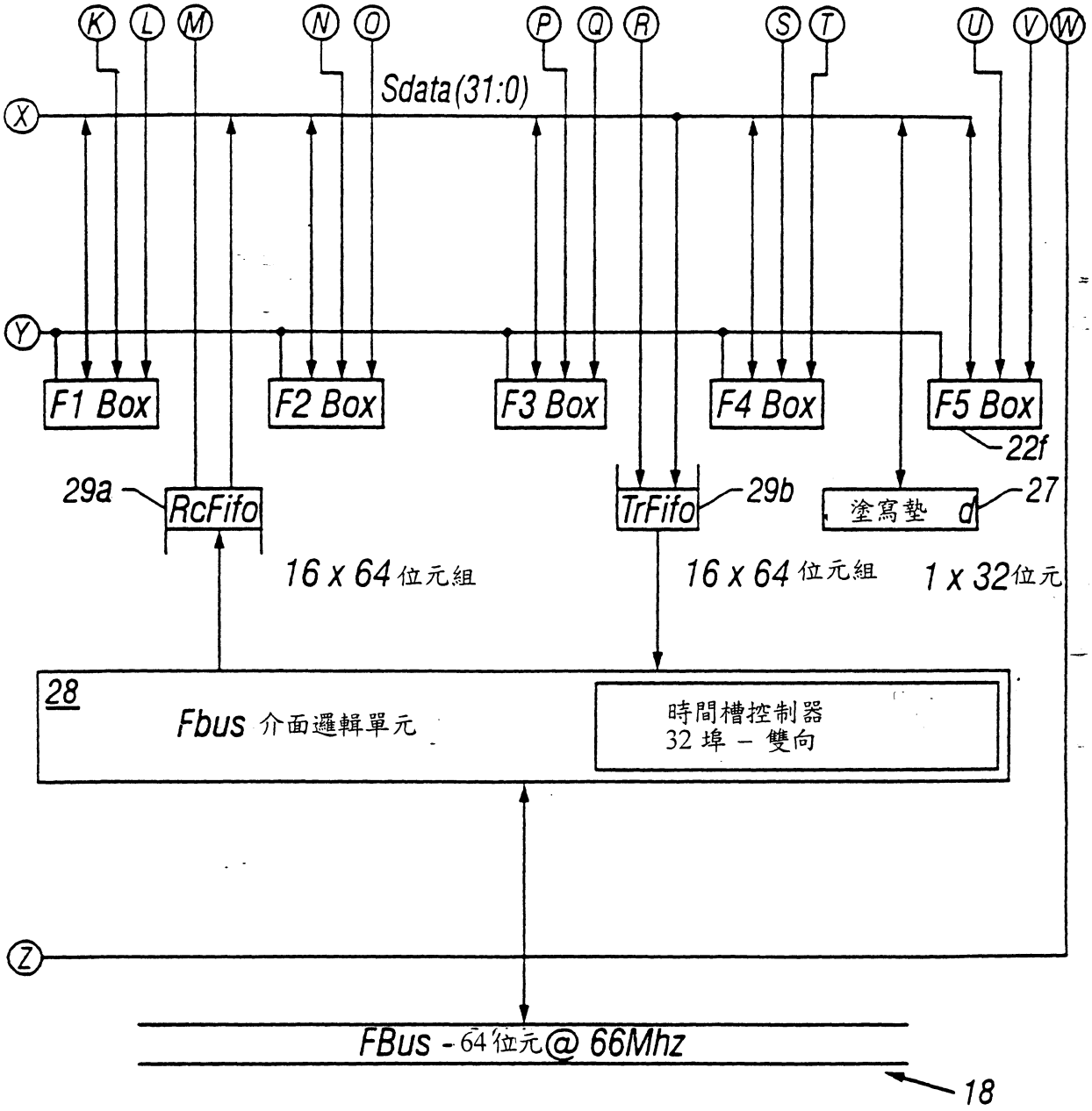
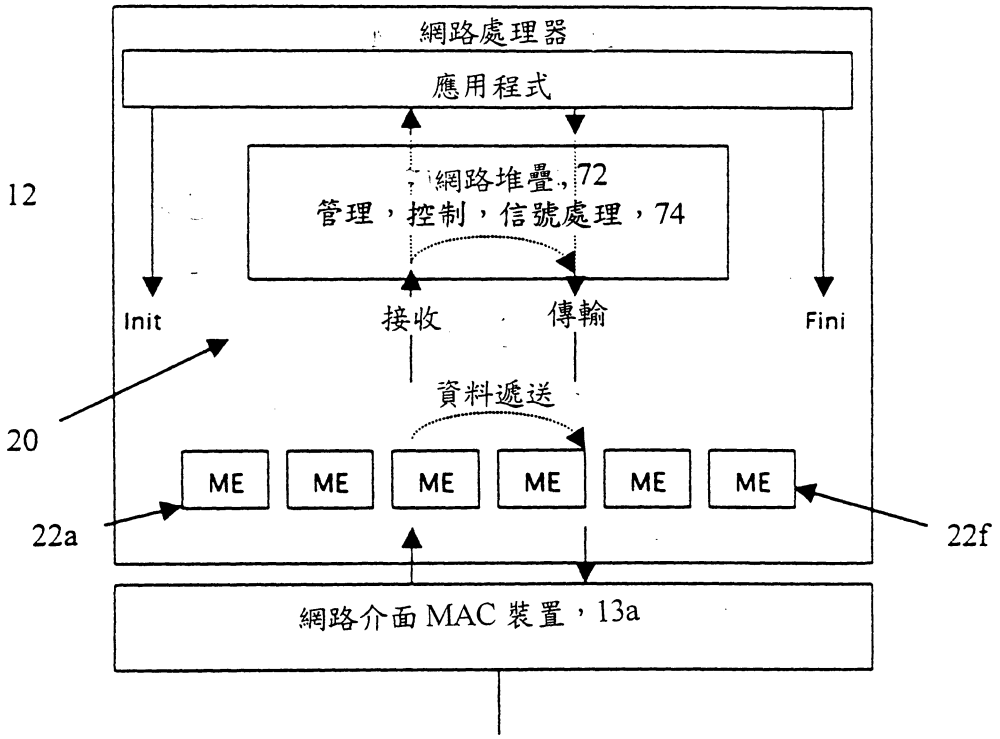

K
L
M
N
O
P
Q
R
S
T
U
V
W
Sdata(31:0)
X
Y
F1 Box
F2 Box
F3 Box
F4 Box
F5 Box
22f
29a
RcFifo
TrFifo
29b
塗寫墊
27
16 x 64 位元組
16 x 64 位元組
1 x 32位元
28
Fbus 介面邏輯單元
時間槽控制器
32 埠 – 雙向
Z
FBus - 64位元@ 66Mhz
18

圖 2-4

圖式頁



非同步傳輸模式，同步光網，
乙太網路等

圖 3

圖式續頁

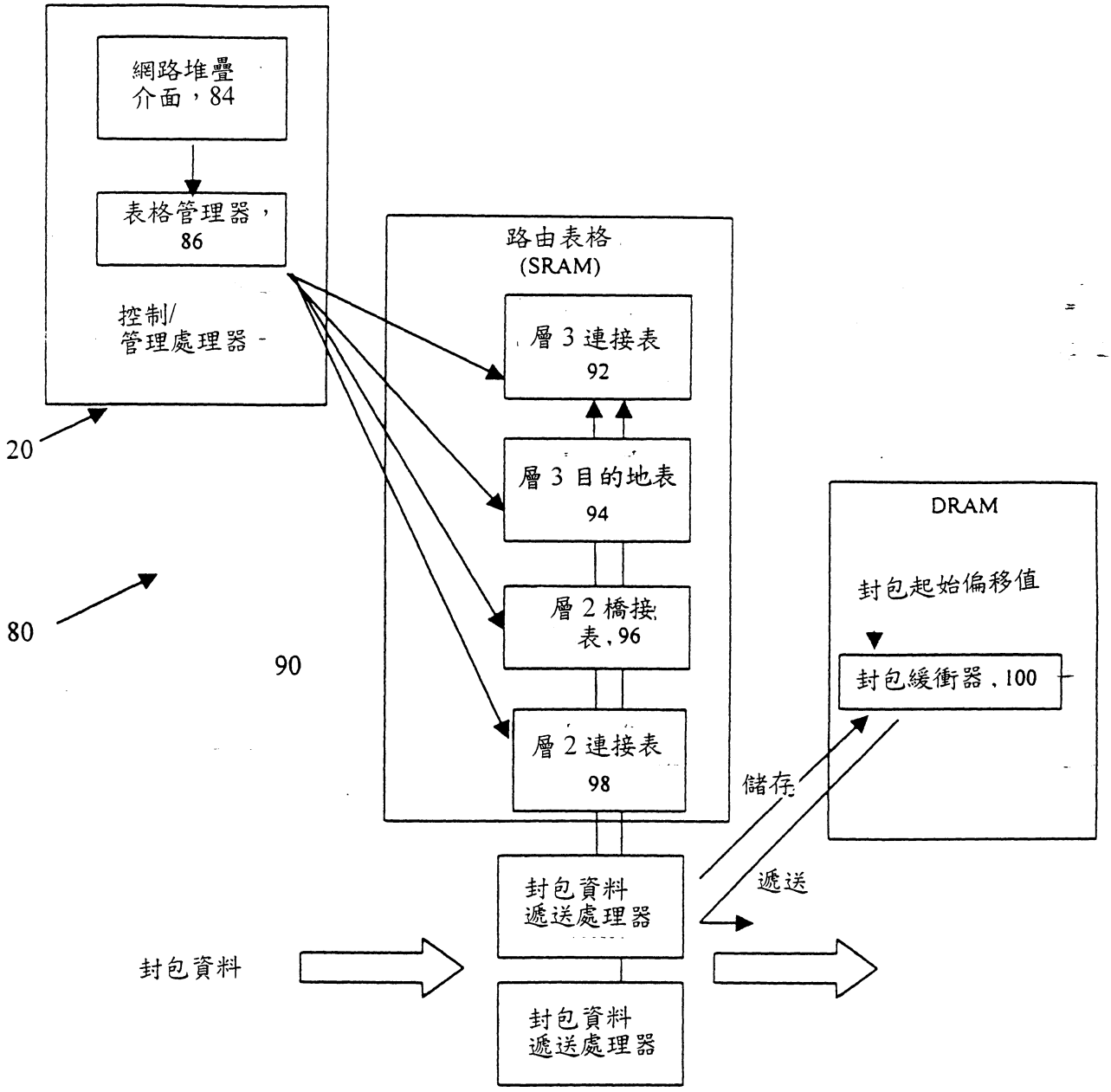

網路堆疊介面，84
表格管理器，86
控制/管理處理器
20
80
90
路由表格 (SRAM)
層 3 連接表 92
層 3 目的地表 94
層 2 橋接表，96
層 2 連接表 98
DRAM
封包起始偏移值
封包緩衝器，100
儲存
遞送
封包資料遞送處理器
封包資料
封包資料遞送處理器

圖 4

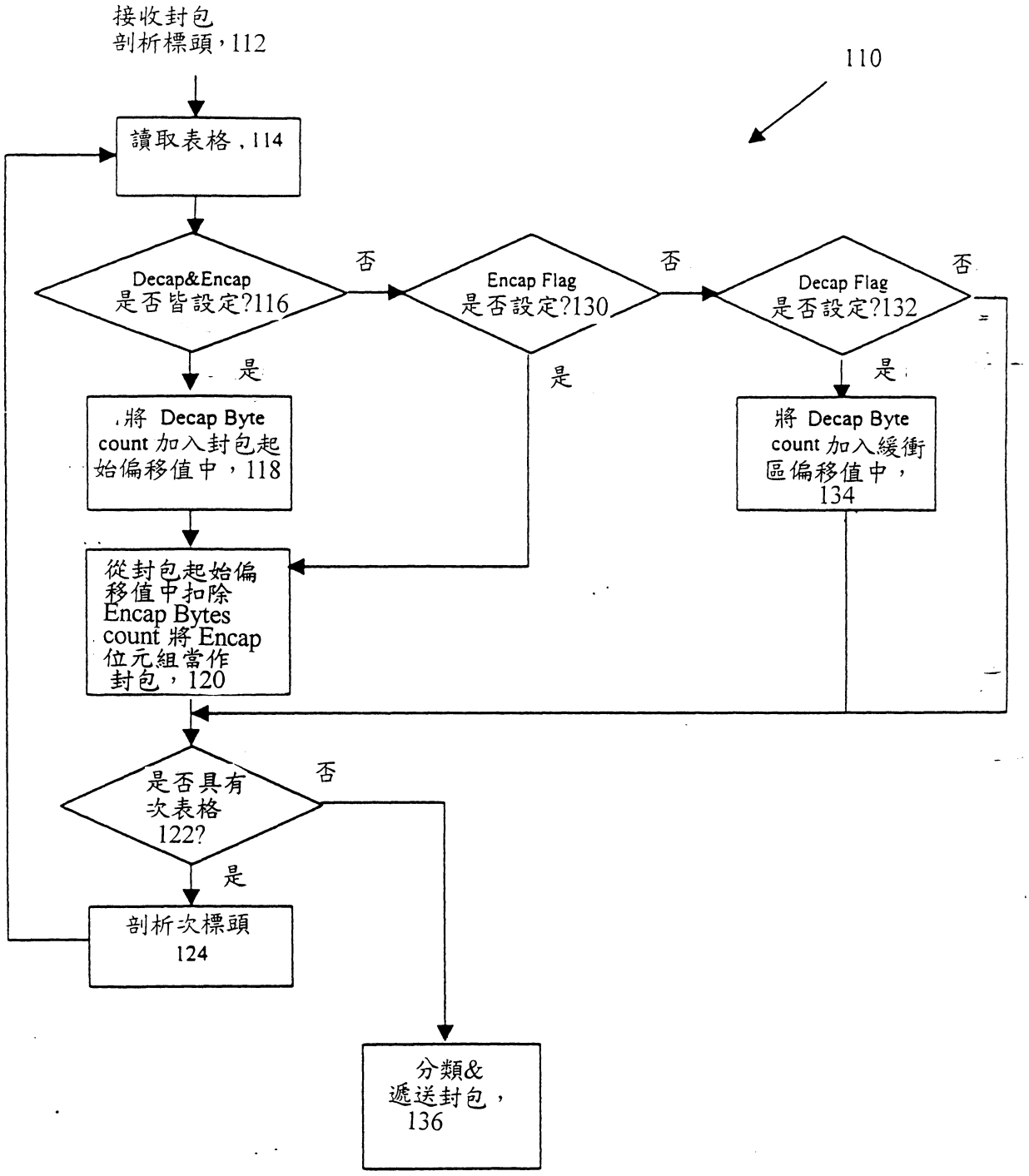

接收封包
剖析標頭，112
110
讀取表格，114
Decap&Encap
是否皆設定?116
否
Encap Flag
是否設定?130
否
Decap Flag
是否設定?132
否
是
是
是
將 Decap Byte count 加入封包起始偏移值中，118
將 Decap Byte count 加入緩衝區偏移值中，134
從封包起始偏移值中扣除 Encap Bytes count 將 Encap 位元組當作封包，120
是否具有次表格 122?
否
是
剖析次標頭 124
分類& 遞送封包，136

圖 6

圖式簡單

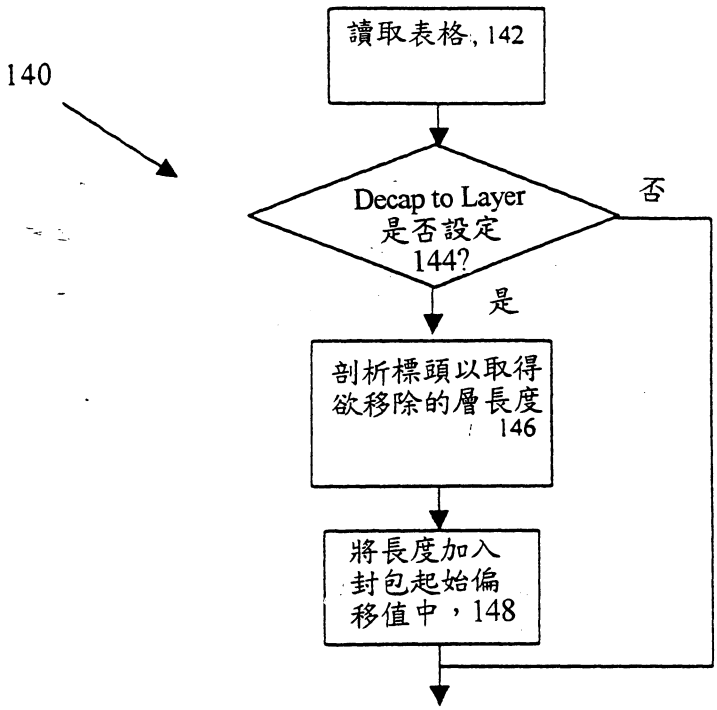

140
讀取表格，142
Decap to Layer
是否設定
144?
否
是
剖析標頭以取得
欲移除的層長度
146
將長度加入
封包起始偏
移值中，148

圖 7

公告本

修正補充

I239164

發明專利說明書

中文說明書替換頁(92年3月)

（填寫本書件時請先行詳閱申請書後之申請須知，作※記號部分請勿填寫）

※ 申請案號：091122365　※IPC分類：H04L 12/46

※ 申請日期：91、9、27

壹、發明名稱

（中文）多協定解封/加封控制結構及封包協定轉換方法

（英文）MULTIPROTOCOL DECAPSULATION/ENCAPSULATION CONTROL STRUCTURE AND PACKET PROTOCOL CONVERSION METHOD

貳、發明人（共 2 人）

發明人 1 （如發明人超過一人，請填**說明書發明人續頁**）

姓名：（中文）丹諾 F. 虎波

（英文）DONALD F. HOOPER

住居所地址：（中文）美國麻州舒伯瑞市主要環道19號

（英文）19 MAIN CIRCLE, SHREWSBURY, MASSACHUSETTS 01545, U.S.A.

國籍：（中文）美國　（英文）U.S.A

參、申請人（共 1 人）

申請人 1 （如申請人超過一人，請填**說明書申請人續頁**）

姓名或名稱：（中文）美商英特爾公司

（英文）INTEL CORPORATION

住居所或營業所地址：（中文）美國加州聖塔卡拉瓦市米遜大學路2200號

（英文）2200 MISSION COLLEGE BLVD., SANTA CLARA, CA. 95052, U.S.A.

國籍：（中文）美國　（英文）U.S.A

代表人：（中文）大衛 賽門

（英文）DAVID SIMON

玖、發明說明

(發明說明應敘明：發明所屬之技術領域、先前技術、內容、實施方式及圖式簡單說明)

發明背景

本發明與在網域之間遞送封包有關。

封包都係透過一連串的路由裝置加以發送，各路由裝置都可儲存該些封包，並且將其從發源地遞送至目的地。舉例來說，封包剛發出時可能是網際網路封包，其可透過ATM(非同步傳輸模式路徑)遞送，然後回到企業網路的乙太網路上，再抵達其最終所希望送達的接收者。當該網路經過該些網域的時候，可能會在該封包中加入各種標頭加封，或是從該封包中移除各種標頭加封。某些連接使用的是點對點協定(PPP)，不過，某些連接則使用多協定標簽交換(MPLS)、第二層穿隧協定(LTTP)、非同步傳輸模式(ATM)等。

圖式之簡單說明

圖1所示的係採用硬體式多重執行緒處理器之通信系統方塊圖。

圖2-1至2-4所示的係圖1之硬體式多重執行緒處理器之詳細方塊圖。

圖3所示的係圖2之多重執行緒處理器之功能安排方塊圖。

圖4所示的係使用於圖1之處理器中的記憶體之資料結構方塊圖。

圖5所示的係使用於圖4之遞送表格的格式方塊圖。

圖6所示的係一般封包遞送過程之處理程序圖。

圖7所示的係圖6之封包遞送過程之另一觀點的處理程

求。該些佇列可保持記憶體參考順序或排列記憶體參考方式，以最佳化記憶體頻寬。舉例來說，thread_0與thread_1無關，因此，thread 1及0當然可以隨意的順序對SRAM單元進行記憶體參考。微引擎22a-22f都能夠發出記憶體參考要求給記憶體控制器26a及26b。微引擎22a-22f會對記憶體控制器26a及26b進行充分的記憶體參考作業，使得記憶體控制器26a及26b變成處理器12作業時的瓶頸。

如果記憶體子系統16充滿本質不相關的記憶體要求的話，處理器12便能夠執行記憶體參考排序。記憶體參考排序會改良可達成的記憶體頻寬。如下面敘述，記憶體參考排序可降低存取SRAM時所產生的空滯時間或磁泡。對SRAM進行記憶體參考，切換讀取及寫入之間信號的電流方向時便會產生磁泡或空滯時間，以等待將SRAM 16b耦合至SRAM控制器26b的導體中的電流設定完成。

也就是，用以驅動匯流排上的電流的驅動器必須在改變狀態之前便設定完成。因此，不斷地進行讀寫循環可能會降低峰值頻寬。記憶體參考排序可讓處理器12組織記憶體的參考方式，使得一長串的讀取之後才會進行一長串的寫入。如此便能夠最小化管線中的空滯時間，更有效地達成最大可用頻寬。參考排序有助於保持平行硬體資料內容執行緒。在SDRAM中，參考排序可隱藏儲存單元(bank)之間的前置充電。明確地說，如果將記憶體系統16b組織成奇數儲存單元及偶數儲存單元的話，當處理器作業於奇數儲存單元時，記憶體控制器便能夠開始對偶數儲存單元進行

元件符號說明

10	通信系統
12	多重執行緒處理器
13a	10/100 BaseT 八進位媒體存取控制器
13b	十億位元(Gigabit)乙太裝置
14	週邊控制器介面(PCI)匯流排
16	記憶體系統
16a	同步動態隨機存取記憶體
16b	靜態隨機存取記憶體
16c	快閃唯讀記憶體
18	先進先出(FIFO)匯流排(FBUS)
20	中央控制器
22,22a-22f	微引擎
24	PCI 匯流排介面
24a	高速資料路徑
26a	同步動態隨機存取記憶體(SDRAM)控制器
26b	靜態隨機存取記憶體(SRAM)控制器
27	塗寫墊
28	FBUS 介面
29a,29b	先進先出
29c	雜湊單元
30	AMBA 轉譯器
32	匯流排
34	SRAM 匯流排
38	SDRAM 匯流排
50	RISC 核心
52	指令快取記憶體

54	資料快取記憶體
56	預取流動緩衝器
72	網路堆疊
74	網路管理、控制及信號處理
80	管理安排
84	網路堆疊介面
86	表格管理器
90	表格結構
92	層 4 連接表
94	層 3 目的地表
96	層 2 橋接表
98	層 2 連接表
100	封包緩衝區
110	處理程序

拾、申請專利範圍

1. 一種遞送網路封包之方法，其包括：

讀取一含有複數個旗標的表格，用以決定應該設定或清除該複數個旗標中的哪一個旗標；以及

根據該些旗標數值對該封包執行運算，以便解封或加封該封包。

2. 如申請專利範圍第1項之方法，其中，該些表格充滿遞送資訊。

3. 如申請專利範圍第1項之方法，其中，該遞送表格結構包括一控制及管理結構，其包括一網路堆疊介面及表格管理器。

4. 如申請專利範圍第3項之方法，其中，該表格管理器會管理路由表格，並且包括複數個表格，其包括層4連接表、層3目的地表、層2橋接表及層2連接表。

5. 如申請專利範圍第1項之方法，其中，該些表格包含一旗標，表示是否應該從該封包中剝離該些位元組；以及一欄位，表示欲剝離的位元組數量。

6. 如申請專利範圍第1項之方法，其中，該些表格包含一欄位，用以規定向上解封至該指定層的標頭層。

7. 如申請專利範圍第1項之方法，其中，該些表格包含一欄位，用以規定目前封包加封類型的識別符號。

8. 如申請專利範圍第1項之方法，其中，該些表格包含一旗標，表示位元組是否應該當作該封包；以及一欄位，規定其位元組數量以及欲加封的位元組。

9.如申請專利範圍第1項之方法，其中，該些表格包含一Next Table Type欄位，用以表示需要一另外的查詢表，以及確認該表格的類型。

10.一種用以加封/解封封包之方法，其包括：

接收一封包；

讀進該封包的第一標頭，並且執行層2查詢表，該查詢表會讀取一連接表以回傳參數；以及

判斷該表是否回傳decap flag或encap flag。

11.如申請專利範圍第10項之方法，其中，如果decap flag及encap flag都已經設定的話，

將decap byte count加入封包起始偏移值中，並且從該封包起始偏移值中扣除encap byte count；以及

將該些encap位元組當作該封包。

12.如申請專利範圍第10項之方法，進一步包括：

藉由檢視目前所讀取之表格中的空白欄位，判斷是否有次表格要檢查。

13.如申請專利範圍第12項之方法，其中，如果有次表格的話，

剖析該次標頭，擷取及讀取該次表格。

14.如申請專利範圍第11項之方法，其中，如果decap flag及encap flag並非皆已經設定的話，

判斷是否 encap flag 或 decap flag 兩者其中之一被設定。

15.如申請專利範圍第11項之方法，其中，如果encap flag被設定的話，其便會從該起始偏移值中扣除encap byte

count，然後將該些encap位元組當作該封包。

16. 如申請專利範圍第11項之方法，其中，如果decap flag被設定的話，便將decap byte count加入該起始偏移值中，並且檢查次表格。

17. 如申請專利範圍第11項之方法，其中，該封包係由一或多個標頭後面加上酬載所構成的，該方法進一步包括：

將該封包的酬載部份複製到封包緩衝區中。

18. 如申請專利範圍第17項之方法，其中，複製可能會將該封包放置在該緩衝區的偏移區中，以便挪出空間供任何新的標頭(其可能是當作封包)使用，用以進行封包遞送。

19. 一種常駐於電腦可讀取媒體中用以遞送網路封包之電腦程式產品，其包括指令讓電腦進行下面工作：

讀取一含有複數個旗標的表格，用以決定應該設定或清除該複數個旗標中的哪一個旗標；以及

根據該些旗標數值對該封包執行運算，以便解封或加封該封包。

20. 如申請專利範圍第19項之電腦程式產品，其中，該些表格充滿遞送資訊。

21. 如申請專利範圍第19項之電腦程式產品，其中，該遞送表格結構包括一控制及管理結構，其包括一網路堆疊介面及表格管理器。

22. 一種常駐於電腦可讀取媒體中用以遞送網路封包之電腦程式產品，其包括指令讓電腦進行下面工作：

接收一封包；

讀進該封包的第一標頭，並且執行層2查詢表，該查詢表會讀取一連接表以回傳參數；以及

判斷該表是否回傳decap flag或encap flag。

23.如申請專利範圍第22項之電腦程式產品，其中，如果decap flag及encap flag都已經設定的話，該電腦程式便會執行指令以進行下面工作：

將decap byte count加入封包起始偏移值中，並且從該封包起始偏移值中扣除encap byte count；以及

將該些encap位元組當作封包。

24.如申請專利範圍第22項之電腦程式產品，其進一步包括指令以進行下面工作：

藉由檢視目前所讀取之表格中的空白欄位，判斷是否有次表格要檢查。

25.如申請專利範圍第24項之電腦程式產品，其中，如果有次表格的話，該電腦程式便會執行指令以進行下面工作：

剖析該次標頭，擷取及讀取該次表格。

26.如申請專利範圍第22項之電腦程式產品，其中，該封包係由一或多個標頭後面加上酬載所構成的，該電腦程式產品會進一步執行指令以進行下面工作：

將該封包的酬載部份複製到封包緩衝區中。

27.如申請專利範圍第26項之電腦程式產品，其中，用以複製的指令可能會將該封包放置在該緩衝區的偏移區

中，以便挪出空間供任何新的標頭(其可能是當作封包)使用，用以進行封包遞送。

28.一種處理網路封包之處理器，其包括：

一儲存指令的電腦儲存媒體，用以讓電腦進行下面工作：

讀取一含有複數個旗標的表格，用以決定應該設定或清除該複數個旗標中的哪一個旗標；以及

根據該些旗標數值對該封包執行運算，以便解封或加封該封包。

29.如申請專利範圍第28項之處理器，其中，該些表格包含遞送資訊。

30.一種解封網路封包之方法，其包括：

讀取一含有複數個旗標的表格，用以決定應該設定或清除該複數個旗標中的哪一個旗標；以及

根據該些旗標數值對該封包執行解封運算。

31.如申請專利範圍第30項之方法，其中，該些表格包含遞送資訊。

解封旗標	解封至層	解封位組數量	目前加封	加封旗標	加裝位元數量	加封位元組	次表格型態	次表格位址

圖 5