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：

※ 申請日期：　　※IPC 分類：　H04L12/56

壹、發明名稱：（中文/英文）

硬體式 TCP/IP 訊務卸載引擎裝置及其方法

FULL HARDWARE BASED TCP/IP TRAFFIC OFFLOAD ENGINE (TOE) DEVICE AND METHOD THEREOF

貳、申 請 人：（共 1 人）

姓名或名稱：（中文/英文）ID ：S00002002A

財團法人工業技術研究院

INDUSTRIAL TECHNOLOGY RESEARCH INSTITUTE

代表人：（中文/英文）翁政義/ WENG, CHENG I

住居所或營業所地址：（中文/英文）

新竹縣竹東鎮中興路四段 195 號/No. 195, Sec. 4, Chung-Hsing Rd., Chu-Tung, Hsinchu, Taiwan, R. O. C.

國　籍：（中文/英文）中華民國

參、發明人：（共 3 人）

姓　名：（中文/英文）　　ID ：

1. 江欣潔 /CHIANG, HSIN CHIEH　　R122663953
2. 戴元邦 /DAI, YUAN PANG　　H122014177
3. 王垂俞 /WANG, CHUEI YU　　L122119124

住居所地址：（中文/英文）

1.2.3.新竹縣竹東鎮中興路四段 195 號/No. 195, Sec. 4, Chung-Hsing Rd., Chu-Tung, Hsinchu, Taiwan, R. O. C.

國　籍：（中文/英文）

1.2.3.中華民國

肆、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：　年　月　日。

◎本案申請前已向下列國家（地區）申請專利　　主張國際優先權：

【格式請依：受理國家（地區）；申請日；申請案號數　順序註記】

1.

2.

3.

4.

5.

主張國內優先權(專利法第二十五條之一)：

【格式請依：申請日；申請案號數　順序註記】

1.

2.

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】

國外微生物　【格式請依：寄存國名；機構；日期；號碼　順序註記】熟習該項技術者易於獲得，不須寄存。

玖、發明說明：

【發明所屬之技術領域】

本發明係為一種有關於封包資料處理的裝置及方法，特別係指一種應用於網路端點上，利用硬體式之訊務卸載引擎(Traffic Offload Engine, TOE)裝置，來改善網路封包輸出/入處理的裝置及方法。

【先前技術】

隨著網路的蓬勃發展，網路端點之間最常見的溝通速度，也由早期的 10Mbps/100Mbps 發展到如今的 1Gbps，而在不久的將來甚至會發展到 10Gbps 之譜。但是相對於網路端點間傳輸速度的不斷快速演進，屬於網路端點的運算處理機制卻沒有等量的成長，使得各個網路端點只有透過不斷的藉由提昇硬體設備的等級，或者是增加運算處理的硬體設備，例如：透過多處理器系統或者是叢集運算系統...，才能夠獲得在運算處理上的能量滿足。

會造成上述問題的主要原因，就在於促成今日網路普及化的最大功臣：TCP/IP 協定架構。TCP/IP 為目前網路中最主要的應用協定架構，在傳統上都係以軟體方式在作業系統核心(OS kernel)中依據 TCP/IP 協定架構來進行有關網路封包的輸出/入處理，然而由於 TCP/IP 協定架構為了提供彈性的模組設計和達到安全的資訊隱藏等功能，因此在架構上包含了非常複雜的層級(Stack)設計，因此造成在處理網路封包輸出/入時必須要耗費掉許多的運算資源(Computing Resource)來執行額外大量且重複性高的層級協定轉換程序(protocol stacking procedure)，造成的影響便是整體網路封包處理的效能不彰，因此透過軟體方式來運作的網路端點運算處理機制是無法滿足快速發展的網路傳輸速度。

在已揭露的習知技術中來看，美國專利早期公開案件中的專利申請號 PN-20020107971、PN-20020087732 與專利號 PN-6591302 中，均採取了利用額外附加的處理器及其中內建的軟體程式來提供訊務卸載(traffic offload)的目的，其中在 PN-20020107971 提到個別負責處理傳送請求，以及接收封包的處理器機制，而在 PN-20020087732 與 PN-6591302 中則進一步針對兩個額外附加處理器的實現方式、周邊輔

助電路設計作進一步的描述。然而此種解決方案雖然能夠保有較高的應用彈性優點，可以快速發展而且能夠輕易達到事後補救的目的，但是相對的其實現上必須要花費相當大的成本，並且在使用時在傳輸處理延遲(latency)所耗用的時間上會較多，因此在實際的應用層面上，如：終端機介面、即時影音...較為不利。

另外，在專利號 PN-6483840 中則提出了一種如何使用純粹硬體架構，來解決訊務卸載的問題。此案係在符合 OSI 網路架構的基礎之下，內建兩個分別用來個別處理第三層(Layer 3)與第四層(Layer 4)的硬體處理單元，來達到輔助訊務卸載的目的。此解決方法，相較於前述硬體搭配軟體程式的做法來說，在傳輸處理延遲上的效能會較佳，但是由於其仍然承襲在最原始的 OSI 網路架構之下，亦即保留具有分層的概念來作硬體模組設計上的區分，因此依然無法達到在解決訊務卸載上的最佳化。

因此，要如何有效提出一套能夠同時兼顧訊務卸載以及傳輸處理延遲效能的解決方案，不但能夠突破既有網路架構的限制，並且能夠真正提供最佳化的效能，應是目前當務之急。

【發明內容】

有鑑於此，本發明提出一種具有最佳化效能的硬體式 TCP/IP 訊務卸載引擎裝置及方法。希望透過完全硬體的實現方式來充分提昇傳輸處理延遲(processing latency)的問題，同時配合無分層的硬體模組設計概念來真正達到訊務卸載(traffic offload)的主要目的。

本發明的技術手段，除了具有無分層概念的硬體模組設計之外(特別係針對網路架構中第三層與第四層的最佳化整合)，更進一步包含對作業系統現有功能模組的修改，及對硬體既有附屬記憶體的運用，使本發明能夠完成訊務卸載最佳化的實現結果。不但在功效上較習知技術為佳，更具有成本低廉的優勢，並且能夠充分被應用在各種不同的領域上，故能達到滿足網路傳輸發展需求的功效。

有關本發明具體可行之實施方式，茲就配合圖式說明如下：

【實施方式】

本發明係一種硬體式 TCP/IP 訊務卸載引擎裝置及其方法，其中訊務卸載引擎(Traffic Offload Engine, 以下稱 TOE)裝置主要包含兩個部分，分別為：訊務卸載介面 30(以下稱 TOE 介面)，以及訊務卸載驅動程式 300(以下稱 TOE 驅動程式)。

其中，TOE 介面 30 係裝設於主機 20(即網路端點)之匯流排插槽 29 上用以與外部的網路 10 進行連結，如「第 1 圖」中所示。主機 20 為電腦可執行平台，通常係個人電腦系統(主要組成包含：處理器 21、北橋晶片 22、南橋晶片 23、系統記憶體 24、圖形處理卡 25、附加裝置 26、ATAPI 裝置 27、USB 埠 28 及匯流排插槽 29…)，而裝設本發明之 TOE 介面 30 之匯流排插槽 29 一般係指目前主流的 32-bit 33MHz 之 PCI 區域匯流排，但因 PCI 區域匯流排理論頻寬為 133MB/sec，仍然不足以滿足網路端點間的傳輸需求，因此在其他較佳實施例的狀況之下，可採取具有更高傳輸效能的其他規格匯流排，如：64-bit MHz 之 PCI 區域匯流排、PCI-Express 來搭配本發明之裝置及方法，當然在主機內部的系統匯流排傳輸上也必須要能夠提供相對足夠頻寬的傳輸效能，才能夠具體實現本發明之最佳化功效。惟以上技術內容均非本發明所強調的技術特徵所在，故在此不多作贅述。

本發明的重點技術特徵在於：TOE 介面 30 以及 TOE 驅動程式 300 兩部分，其個別的細部方塊組成將透過「第 2 圖」及「第 3 圖」來作進一步說明。

(1)TOE 介面(TOE interface)30 的部分，請參考到「第 2 圖」，主要包含下列幾個必要組成：

(1-1)TCP/IP 輸出模組 31(TCP/IP output module)，負責根據來自主機 20 應用程式所提出的傳送請求，來產生完整的傳送網路封包，並且在產生傳送網路封包的同時會連結至連線記錄區 33 中，進行對應連線資訊的異動及更新，以確保連線狀態的正常維持。

(1-2)TCP/IP 輸入模組 32(TCP/IP input module)，用以對來自外部網路 10(或稱實體層 40)之接收網路封包進行解析及分類，並且同樣會在儲存接收網路封包的同時連結至連線記錄區 33 中，進行對應連線資訊

的異動及更新，以確保連線狀態的正常維持。

事實上，本 TCP/IP 輸入模組 32 有些時候也會為了維持網路連線，而要求 TCP/IP 輸出模組 31 產生網路封包傳送給實體層 40，以保持網路連線的正常維持。

(1-3)連線記錄區 33(flow scoreboard memory)，主要用來記錄 TOE 介面 30 所負責的所有傳送/接收之網路封包的相關連線資訊。此連線記錄區 33 同時與 TCP/IP 輸出模組 31、TCP/IP 輸入模組 32 連結溝通，因此可以作為整個 TOE 介面 30 在運作過程中各個資料交換溝通及維持資料一致性的地方。

除了上述三個必要組成之外，事實上在 TOE 介面 30 中仍包含有下列其他幾個部分：

(A)暫存記憶區 34(packet memory)，負責用來暫存來自 TCP/IP 輸出模組 31 所等待傳送，及 TCP/IP 輸入模組 32 所解析分類完成後的網路封包資料，藉以提昇整體 TOE 介面 30 的運作效率。

一般來說，暫存記憶區 34 為 TOE 介面 30 中內建的附加記憶體，用來暫存等待進行傳送/接收處理之網路封包資料。在暫存記憶區 34 中更包含有仲裁器(arbitrator)，可用來協助決定有關暫存記憶區 34 的控制權歸屬。

另外，當主機介面 36 或者鏈結層模組 37(link layer module)需要時能夠自本暫存記憶區 34 中讀取對應的網路封包。也因為有暫存記憶體 34 的規劃設計，可使得本發明之實現將可以在不需要增加額外記憶體的情況之下(有別於習知技術)，有效達成訊務卸載的目的功效。

(B)計時器 35(timer)，主要與連線記錄區 33 連結，可以根據各個連線之計時狀態來執行網路封包的傳送/接收處理。

事實上，本計時器 35 在必要的時候將會配合 TCP/IP 輸入模組 32 的請求，驅動 TCP/IP 輸出模組 31 產生網路封包並傳送給實體層 40，以保持網路連線的正常維持。

(C)主機介面 36(host interface)，負責對外與主機 20 連結，並與內部的暫存記憶區 34、TCP/IP 輸出模組 31 及 TCP/IP 輸入模組 32 進行

則將交由TOE驅動程式300中的TOE介面路徑處理模組301來進行處理；反之若非TOE介面30所能夠處理的請求類別時(亦即傳送請求必須要以分層處理的方式進行時)，則交由另一堆疊路徑處理模組53來處理(堆疊路徑處理模組53為習知技術部分)。

(B)指令分配介面304(TOE command dispatch shell)，與TOE介面30連結，並透過網路封包中的TOE介面資訊進行溝通，執行對網路封包的傳送/接收處理。

而為了維持本發明與傳統應用程式之間進行網路封包傳輸的相容性以及彈性，在本發明之TOE驅動程式300中，更包含網路應用程式介面51(Socket API)用以與應用程式60之間作充分的溝通；而為了處理一些特殊必須透過傳統網路堆疊方式進行的傳送請求，本TOE驅動程式300中也提供堆疊路徑處理模組53，其主要係以軟體的方式來執行傳統網路協定之封包處理，以使本發明在遇到特殊請求時仍然能夠運作；另外，為了對記憶體管理運用更加便利，在本TOE驅動程式300之中還包含記憶體管理模組52(Memory Management Unit, MMU)，與指標暫存區塊302連結，用來負責與作業系統核心50中的記憶體管理單元52溝通，處理有關虛擬記憶體(virtual memory)以及實體記憶體(physical memory)之間的映射(mapping)問題，此記憶體管理模組52將視主機20所使用的作業系統不同而有不同介面。

最後，有關本發明傳送及接收方法的部分將分別透過「第4圖」及「第5圖」在以下作說明。

「第4圖」為本發明的傳送流程。首先接收來自主機20中應用程式60所發出的傳送請求，經由網路應用程式介面51接收之後傳送至請求分配介面303中決定處理路徑(步驟100)；判斷是否為TOE介面可處理之請求？(步驟110)如果非TOE介面30可處理之請求時則透過堆疊路徑處理的方式來進行處理(步驟120)，否則若是TOE介面30可處理之請求時，則繼續進行本發明後續的流程。

當確認請求為TOE介面30可處理之請求時，則進一步由TOE介面路徑處理模組301負責轉換傳送請求並產生對應的訊務卸載介面資

訊(以下稱 TOE 資訊)(步驟 130)；完成轉換之後，TOE 介面 30 便會透過 TCP/IP 輸出模組 31 依照連線的計時狀態來讀取連線資訊，並結合 TOE 資訊產生可傳送網路封包(步驟 140)儲存於暫存記憶區 34 中；最後再透過鏈結層模組 37 將傳送網路封包傳送至網路 10(實體層 40)中(步驟 150)。

所有傳送過程中的連線資訊，將記錄在 TOE 介面 30 的連線記錄區 33 中，並且隨時根據異動進行更新。

「第 5 圖」為本發明的接收流程。首先，當網路封包自實體層 40 傳入鏈結層模組 37 後，將由 TCP/IP 輸入模組 32 接收並進行解析(步驟 200)；判斷所收到的接收網路封包是否為 TOE 介面 30 所能夠處理的封包？(步驟 210)如果為非 TOE 介面 30 可處理封包時，則交由堆疊路徑處理模組 53 進行堆疊路徑處理(步驟 220)，否則若為 TOE 介面 30 可處理之網路封包時，則繼續進行後續步驟。

當接收網路封包為 TOE 介面 30 可處理封包時，則將解析後的封包資料存放在暫存記憶區 34 中，並且更新連線記錄區 33 中對應的連線資訊(步驟 230)；然後根據對應連線的計時狀態傳送所接收的網路封包至主機 20 之應用程式 60 中(步驟 240)。

所有接收過程中的連線資訊，會將任何的異動記錄在 TOE 介面 30 的連線記錄區 33 中，以維持連線資料的一致性。

在本發明運作過程當中，為了符合網路協定的規範，將會主動在連線過程的適當時機中自動產生網路封包進行傳送，以維護網路連線的正常運作。

以上所述者，僅為本發明其中的較佳實施例而已，並非用來限定本發明的實施範圍；即凡依本發明申請專利範圍所作的均等變化與修飾，皆為本發明專利範圍所涵蓋。

【圖式簡單說明】

第 1 圖係本發明所提之裝置運作架構示意圖；

第 2 圖係本發明所提之 TOE 介面方塊圖；

第 3 圖係本發明所提之 TOE 驅動程式方塊圖；

第 4 圖係本發明所提之傳送方法流程圖；及

第 5 圖係本發明所提之接收方法流程圖。

【圖式符號說明】

10	網路
20	主機
21	處理器
22	北橋晶片
23	南橋晶片
24	系統記憶體
25	圖形處理卡
26	附加裝置
27	ATAPI 裝置
28	USB 埠
29	匯流排插槽
30	TOE 介面
300	TOE 驅動程式
301	TOE 介面路徑處理模組
302	指標暫存區塊
303	請求分配介面
304	指令分配介面
31	TCP/IP 輸出模組
32	TCP/IP 輸入模組
33	連線記錄區
34	暫存記憶區
35	計時器
36	主機介面
37	鏈結層模組
40	實體層
50	作業系統核心

51 網路應用程式介面

52 記憶體管理模組

53 堆疊路徑處理模組

60 應用程式

步驟 100 根據所接收之一傳送請求，決定一處理路徑

步驟 110 為 TOE 可處理請求

步驟 120 採取一堆疊路徑處理

步驟 130 轉換該傳送請求並產生一 TOE 資訊

步驟 140 依照一計時狀態讀取一連線資訊，並結合該 TOE 資訊產生一傳送網路封包

步驟 150 傳送該傳送網路封包至網路

步驟 200 讀取並解析一接收網路封包

步驟 210 為 TOE 可處理封包

步驟 220 透過一堆疊路徑處理

步驟 230 記錄並更新一連線資訊

步驟 240 依照一計時狀態傳送該接收網路封包至應用程式

伍、中文發明摘要：

一種硬體式 TCP/IP 訊務卸載引擎裝置及其方法，此訊務卸載引擎(Traffic Offload Engine, 以下稱 TOE)裝置主要包含：訊務卸載介面及訊務卸載驅動程式兩部分，配合於作業系統核心中所改良之網路應用程式介面(Socket API)，利用無分層的概念來有效改善網路端點在網路封包輸出/入處理上的負擔。

陸、英文發明摘要：

A full hardware based TCP/IP traffic offload engine (TOE) device and method thereof are disclosed. The device includes a TOE interface and a TOE driver. With the designed Socket API executing in the kernel of operating system, the TOE interface and the TOE driver can effectively improve the Input/Output traffic loading between network nodes without protocol stacking.

柒、指定代表圖：

(一)本案指定代表圖為：第（ 1 ）圖。

(二)本代表圖之元件代表符號簡單說明:

捌、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

10	網路
21	處理器
22	北橋晶片
23	南橋晶片
24	系統記憶體
25	圖形處理卡
26	附加裝置
27	ATAPI 裝置
28	USB 埠
29	匯流排插槽
30	TOE 介面

拾、申請專利範圍：

1. 一種硬體式 TCP/IP 訊務卸載引擎裝置，介於主機與網路實體層之間，可提供進行網路封包之訊務卸載處理，該裝置包含：

一訊務卸載介面，安插於主機之匯流排插槽上，用以進行網路封包資料的傳送/接收處理，更包含下列模組：

一連線記錄區，用以記錄傳送/接收之網路封包的一連線資訊；

一 TCP/IP 輸出模組，用以依照來自主機的一傳送請求產生一傳送網路封包，並同時進行該連線記錄區的異動更新；及

一 TCP/IP 輸入模組，用以將來自網路實體層之一接收網路封包進行解析分類傳送至主機，並同時進行該連線記錄區的異動更新；及

一訊務卸載驅動程式，執行於主機之作業系統核心(kernel)中，用以控制該訊務卸載介面進行網路封包的傳送/接收處理，更包含下列模組：

一指標暫存區塊，用以記錄該訊務卸載介面所使用之複數個記憶體區塊指標(memory block pointer)狀態；及

一訊務卸載介面路徑處理模組，用以進行網路封包的傳送/接收處理，當接收到來自主機應用程式之一傳送請求時能夠進行該傳送請求的轉換及產生一訊務卸載介面資訊，並同時進行該指標暫存區塊的異動更新。

2. 如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載介面更包含一暫存記憶區，用以暫存傳送/接收之網路封包的資料。

3. 如申請專利範圍第 2 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該

暫存記憶區更包含一仲裁器(arbitrator)用以決定該暫存記憶區的控制權歸屬。

4. 如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載介面更包含一計時器，與該連線記錄區連結，用以根據一計時狀態執行網路封包的傳送/接收。

5. 如申請專利範圍第 4 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該計時器更包含定時執行驅動該 TCP/IP 輸出模組產生該傳送網路封包的處理程序。

6. 如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載介面更包含一主機介面，用以處理與主機之間網路封包的傳收及讀寫。

7. 如申請專利範圍第 6 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該主機介面更包含執行該訊務卸載介面與主機之間直接記憶體存取(Direct Memory Access, DMA)的資料傳輸、中斷處理程序。

8. 如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載介面更包含一鏈結層模組(link layer module)，用以處理該暫存記憶區與網路實體層之間網路封包資料的傳收及讀寫。

9. 如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載驅動程式更包含一請求分配介面，用以分配該傳送請求至對應一處理路徑。

10. 如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該

訊務卸載驅動程式更包含一指令分配介面，用以透過該訊務卸載介面資訊進行網路封包的傳送/接收之處理。

11.如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載驅動程式更包含一網路應用程式介面(Socket API)，相容於應用程式並可提供進行溝通。

12.如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載驅動程式更包含一記憶體管理模組(Memory Management Unit, MMU)，用以與該指標暫存區塊及作業系統之記憶體管理單元溝通，進行虛擬記憶體(virtual Memory)與實體記憶體(physical Memory)之間的映射(mapping)。

13.如申請專利範圍第 1 項所述之硬體式 TCP/IP 訊務卸載引擎裝置，其中該訊務卸載驅動程式更包含一堆疊路徑處理模組，用以透過分層堆疊方式進行網路封包的傳送/接收處理。

14.一種硬體式 TCP/IP 訊務卸載引擎方法，運作於主機與網路實體層之間，可提供進行網路封包傳送之訊務卸載處理，該方法包含下列步驟：

根據所接收之一傳送請求，決定一處理路徑；及

當該傳送請求為訊務卸載介面可處理請求時，更包含執行下列步驟：

轉換該傳送請求並產生一訊務卸載介面資訊；

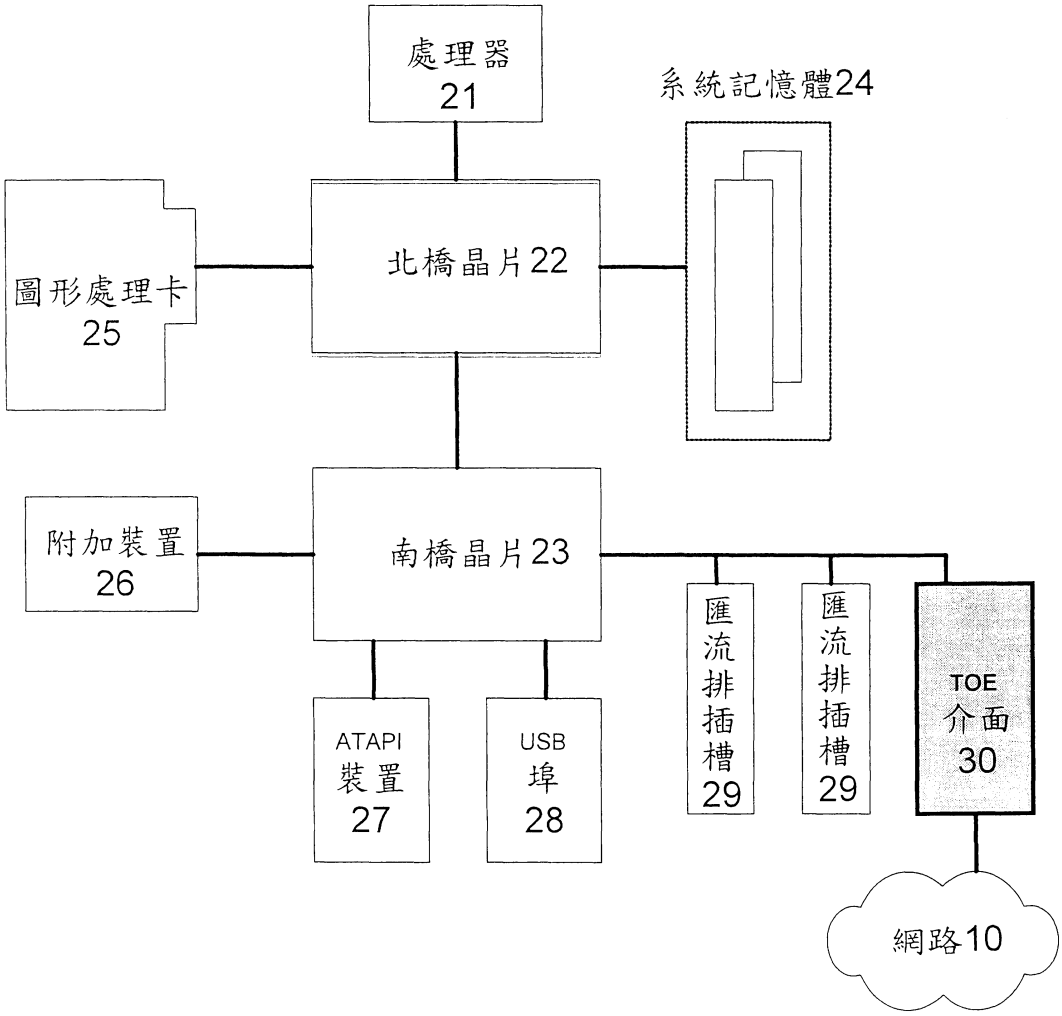
依照一計時狀態讀取一連線資訊，並結合該訊務卸載介面資訊產生一傳送網路封包；及

傳送該傳送網路封包至網路。

15.如申請專利範圍第 14 項所述之硬體式 TCP/IP 訊務卸載引擎方法，其中該方法更包含當該傳送請求為訊務卸載介面不可處理請求時，採取一堆疊路徑處理的步驟。

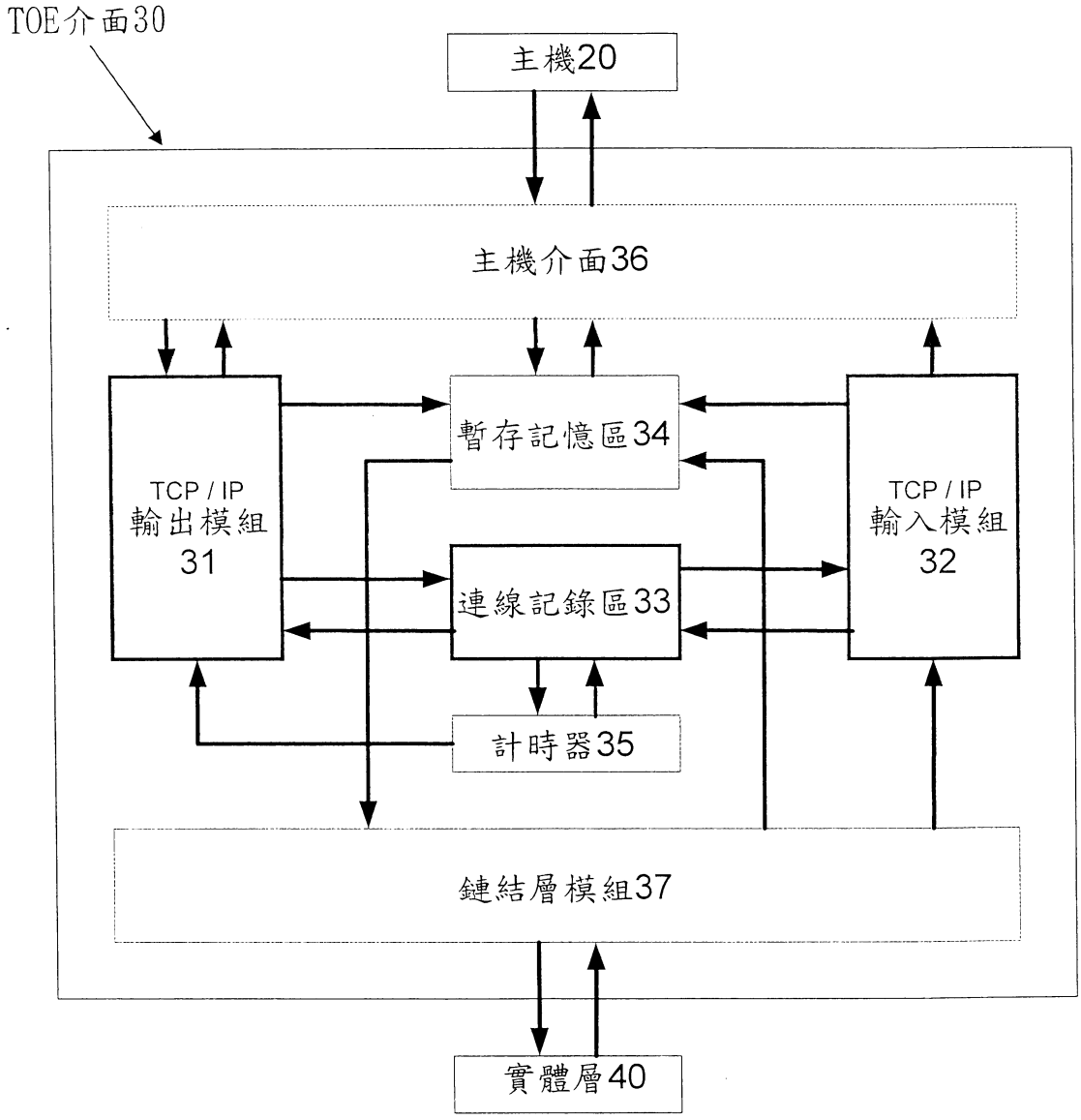
16.如申請專利範圍第 14 項所述之硬體式 TCP/IP 訊務卸載引擎方法，其中該計時狀態更包含定時執行驅動，以產生該傳送網路封包的處理步驟。

圖式


處理器
21
系統記憶體24
圖形處理卡
25
北橋晶片22
附加裝置
26
南橋晶片23
ATAPI
裝置
27
USB
埠
28
匯流排插槽
29
匯流排插槽
29
TOE
介面
30
網路10

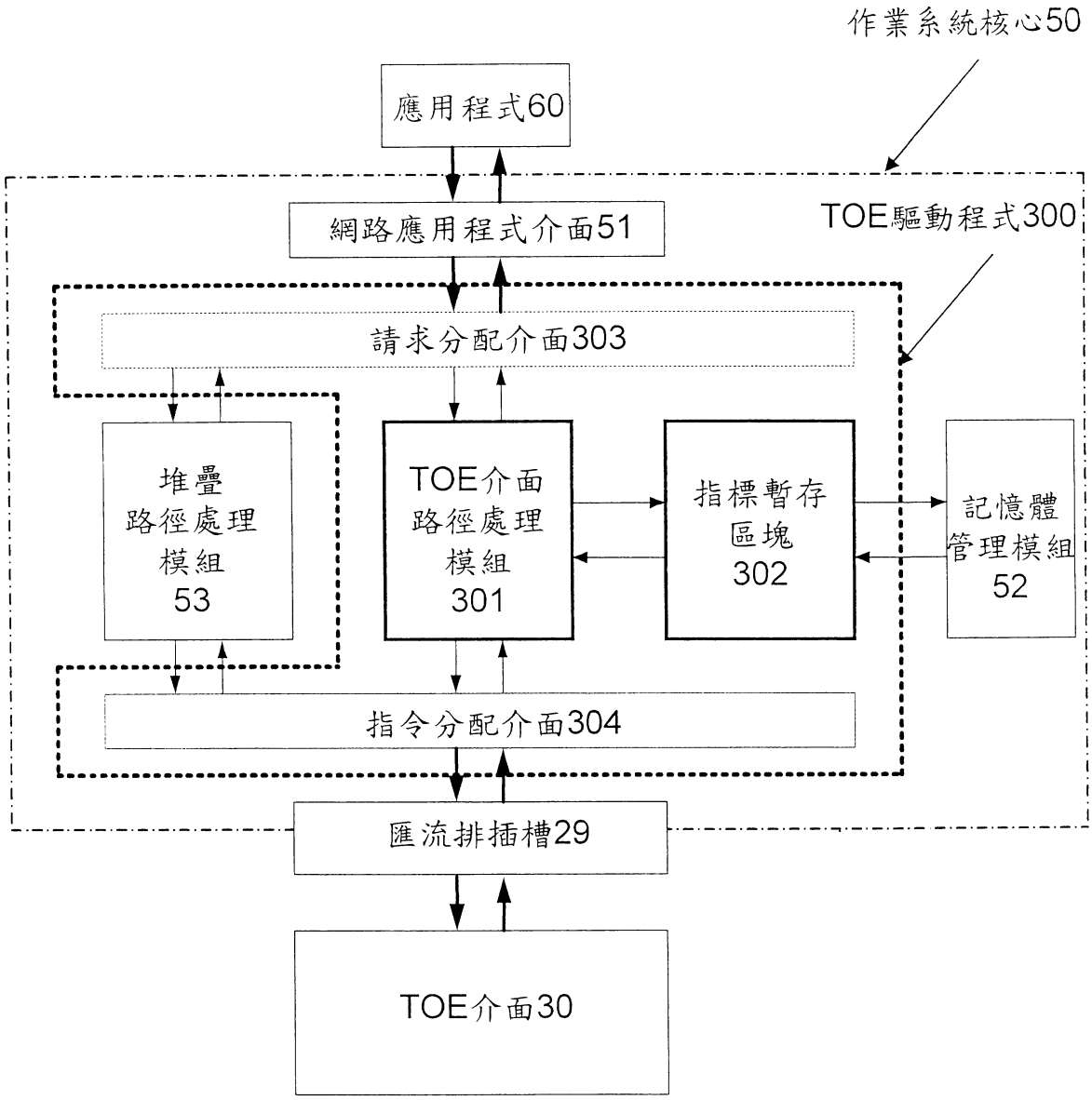
第 1 圖

圖式


TOE介面30
主機20
主機介面36
TCP / IP
輸出模組
31
暫存記憶區34
連線記錄區33
計時器35
TCP / IP
輸入模組
32
鏈結層模組37
實體層40

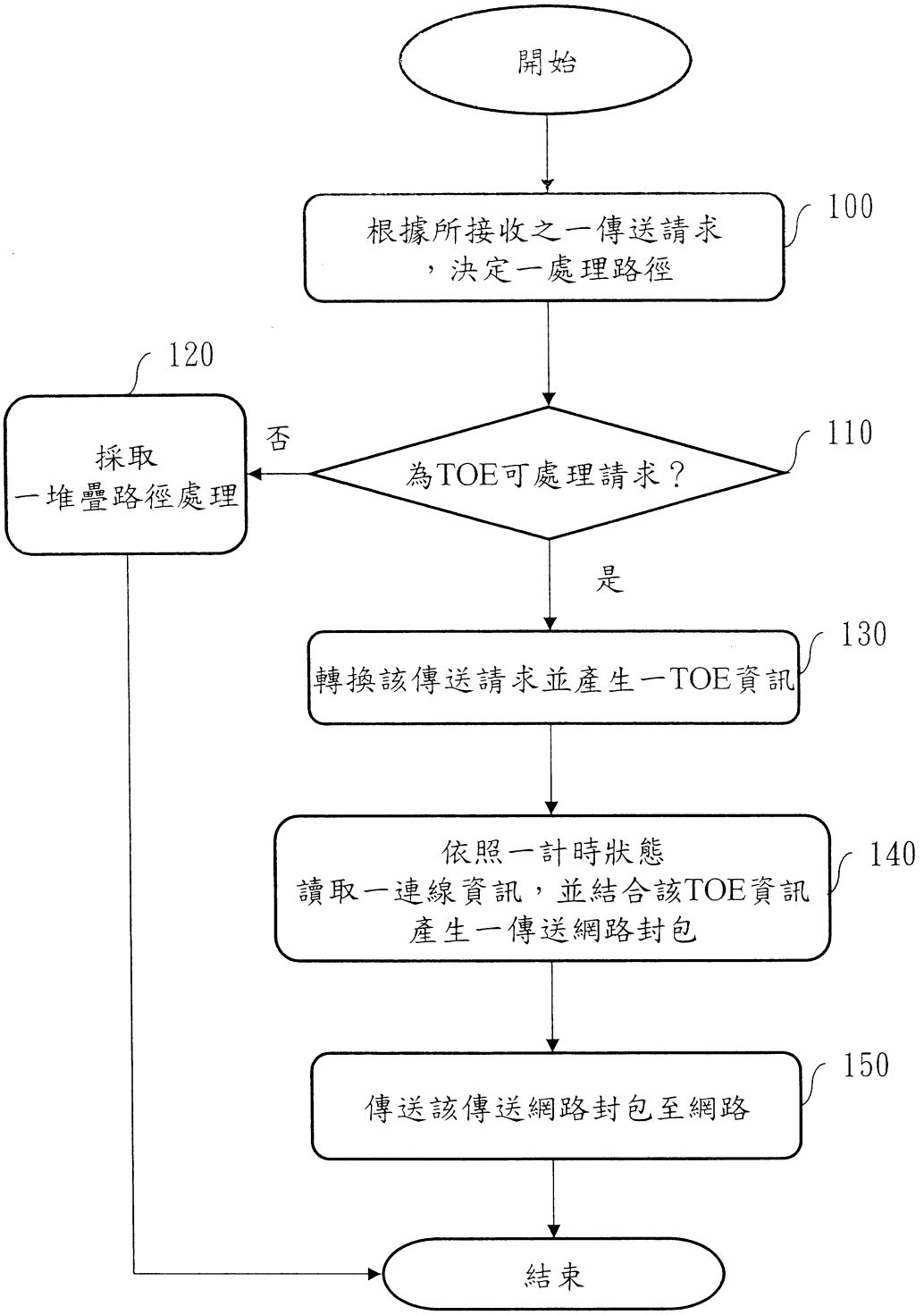
第 2 圖

圖式


作業系統核心50
應用程式60
網路應用程式介面51
TOE驅動程式300
請求分配介面303
堆疊
路徑處理
模組
53
TOE介面
路徑處理
模組
301
指標暫存
區塊
302
記憶體
管理模組
52
指令分配介面304
匯流排插槽29
TOE介面30

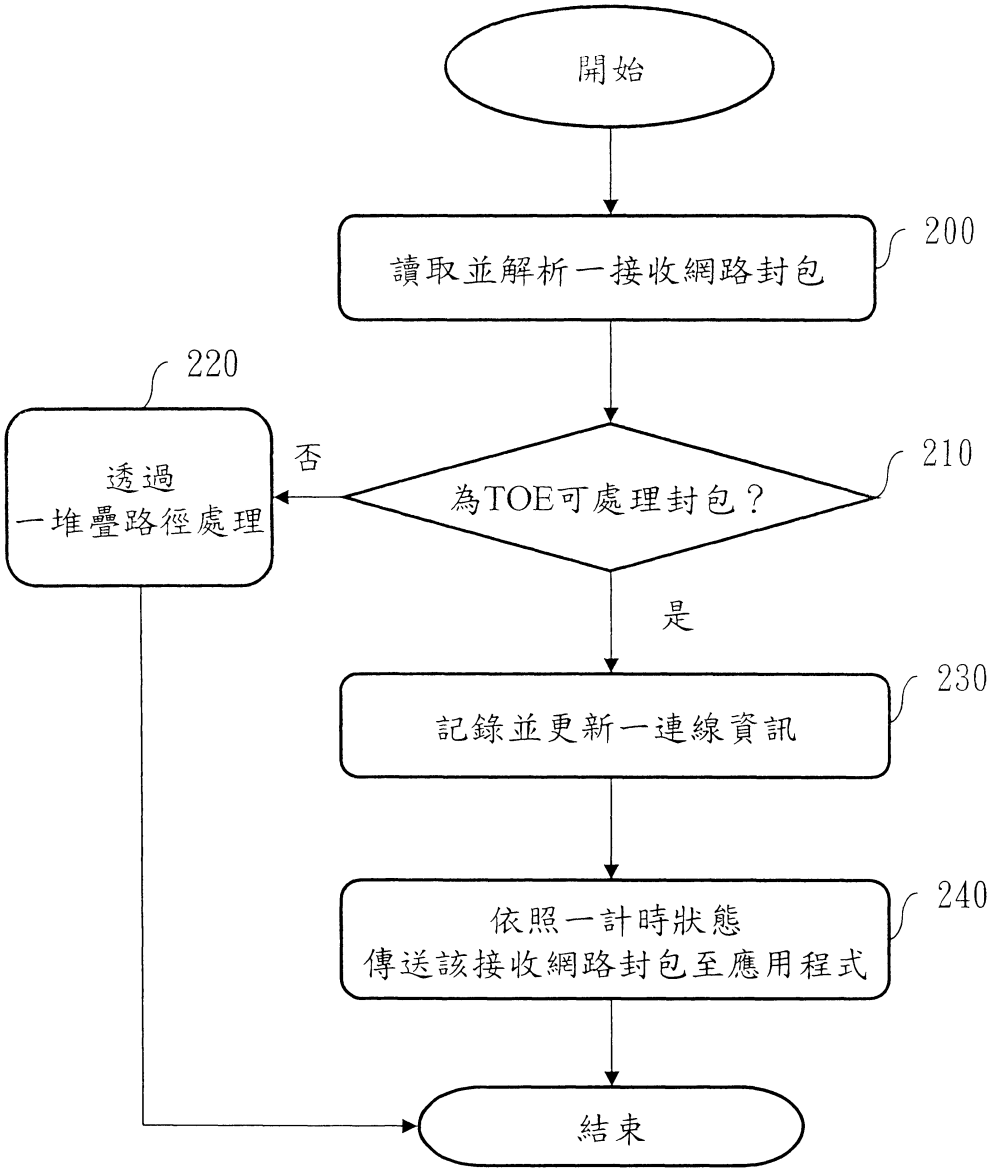
第 3 圖

圖式


開始
根據所接收之一傳送請求，決定一處理路徑
100
為TOE可處理請求？
110
否
採取一堆疊路徑處理
120
是
轉換該傳送請求並產生一TOE資訊
130
依照一計時狀態讀取一連線資訊，並結合該TOE資訊產生一傳送網路封包
140
傳送該傳送網路封包至網路
150
結束

第 4 圖

圖式


開始
讀取並解析一接收網路封包
200
為TOE可處理封包？
210
否
透過
一堆疊路徑處理
220
是
記錄並更新一連線資訊
230
依照一計時狀態
傳送該接收網路封包至應用程式
240
結束

第 5 圖

連結溝通。

其主要的運作內容包括：讀取暫存記憶區 34 中的網路封包傳送給主機 20 或 TCP/IP 輸出模組 31；將接收自主機 20 或 TCP/IP 輸入模組 32 之網路封包寫入暫存記憶區 34；負責進行與主機 20 之間直接記憶體存取(Direct Memory Access, DMA)的資料傳輸程序、中斷處理程序…等等。

(D)鏈結層模組 37，負責對外與外部網路 10(或實體層 40)連結，用以處理網路封包資料的傳送。

其主要的運作內容包括：自暫存記憶區 34 中所讀取之網路封包資料的傳送；及將自外部實體層 40 所接收之網路封包寫入至暫存記憶區 34 中並傳送給 TCP/IP 輸入模組 32。

(2)TOE 驅動程式(TOE driver)300 的部分，則請參考「第 3 圖」，執行於主機 20 之作業系統核心 50(kernel)中用來控管 TOE 介面 30 的各項運作，其主要包含下列幾個必要組成：

(2-1)TOE 介面路徑處理模組 301(TOE path processing module)，用以進行網路封包的傳送/接收處理。

當接收到傳送請求時，能夠進行傳送請求的轉換，及產生對應的 TOE 介面資訊，反之當接收由 TOE 介面 30 所傳來的資料時，將會進行接收處理。此外，於處理傳送/接收處理的同時，將進一步進行指標暫存區塊 302 中指標狀態的異動更新。

(2-2)指標暫存區塊 302(memory pool)，與 TOE 介面路徑處理模組 301 連結，用以隨時記錄 TOE 介面 30 所使用之記憶體區塊指標(memory block pointer)狀態。

除了上述兩個必要組成之外，事實上在 TOE 驅動程式 300 中仍包含有下列其他幾個部分：

(A)請求分配介面 303(request dispatch shell)，用以分配來自主機 20 中應用程式 60 之傳送請求至對應的處理路徑。

至於處理路徑決定的原則，則係依照應用程式 60 所提出的傳送請求類型來決定，當傳送請求為 TOE 介面 30 所能夠處理的請求類別時，