

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： 92/07512

※申請日期： 92-04-02

※IPC 分類： G06F 9/32

壹、發明名稱：(中文/英文)

用以資料轉移之暫存器

REGISTERS FOR DATA TRANSFERS

貳、申請人：(共1人)

姓名或名稱：(中文/英文)

美商英特爾公司

INTEL CORPORATION

代表人：(中文/英文)

大衛 賽門

DAVID SIMON

住居所或營業所地址：(中文/英文)

美國加州聖塔卡拉瓦市米遜大學路 2200 號

2200 MISSION COLLEGE BOULEVARD, SANTA
CLARA, CA 95052, U.S.A.

國籍：(中文/英文)

美國 U.S.A.

參、發明人：(共 4 人)

姓 名：(中文/英文)

1. 吉爾柏特 瓦芮雪
GILBERT WOLRICH
2. 馬克 B. 洛辛保斯
MARK B. ROSENBLUTH
3. 達布拉 布恩斯
DEBRA BERNSTEIN
4. 馬修 J. 亞狄里特
MATTHEW J. ADILETTA

住居所地址：(中文/英文)

1. 美國麻瑟諸塞州菲蒙漢市希得密爾路 4 號
4 CIDER MILL ROAD, FRAMINGHAM,
MASSACHUSETTS 01701, U.S.A.
2. 美國麻瑟諸塞州尤斯布局市奎斯維路 4 號
4 CRESTVIEW DRIVE, UXBRIDGE, MASSACHUSETTS
01569, U.S.A.
3. 美國麻瑟諸塞州沙得伯芮市舊藍斯特路 312 號
312 OLD LANCASTER ROAD, SUDBURY,
MASSACHUSETTS 01776, U.S.A.
4. 美國麻瑟諸塞州保頓市叟亞路 244 號
244 SAWYER ROAD, BOLTON, MASSACHUSETTS
01740, U.S.A.

國 籍：(中文/英文)

- 均美國 U.S.A.

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：

1. 美國 2002 年 04 月 03 日 10/116,670

2.

3.

4.

5.

☒ 主張國際優先權(專利法第二十四條)：

【格式請依：受理國家（地區）；申請日；申請案號數 順序註記】

1. 美國 2002 年 04 月 03 日 10/116,670

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】

本發明係關於計算機中之計算處理，更特別是關於一種在多硬體式資料內容及程式設計引擎內使用用以資料轉移之暫存器以利於高性能之資料處理的系統及方法。

【先前技術】

平行處理為計算處理中處理並行事件資訊的有效形式。與循序處理相比，平行處理要求多個程式的並行執行。平行處理的資料內容中，平行包括同時處理多個事件。與全部工作在單一站臺上依序執行的串列範例或在專門站臺執行工作的管線機器不同，平行處理提供多個站臺，每一站臺皆能同時執行及實施多種工作。多個站臺同時並獨立處理一計算工作之同一或共同元件。因此，平行處理可解決多類型的計算工作，特定問題適於藉由應用數個指令處理單元及數個資料流來解決。

【發明內容】

本發明揭示一種在多硬體式資料內容及程式設計引擎內使用用以資料轉移之暫存器以利於高性能之資料處理的系統及方法。該系統及方法包括一處理器，其包括的程式引擎具有暫存器，用以將資料從位於一執行中程式設計引擎內的該等暫存器之一，轉移至位於一相鄰程式設計引擎內的下一該等暫存器之一。

【實施方式】

結構：

參考圖1，一電腦處理系統10包括一平行硬體式多重執行緒網路處理器12。該硬體式多重執行緒處理器12耦合一記憶體系統或記憶體資源14。記憶體系統14包括動態隨機存取記憶體(dynamic random access memory；DRAM)14a及靜態隨機存取記憶體14b(static random access memory；SRAM)。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潛時導向的工作中。該硬體式多重執行緒處理器12有多個功能微型引擎或程式設計引擎16a至16h(統稱程式設計引擎16)，各帶多個硬體受控執行緒，其對一具體工作可同時啟動並獨立作業。

程式設計引擎16各維護一硬體程式計數器並保持與其相聯之狀態。相應的內容組或執行緒能有效地同時在各微引擎16上啟動，但在任何時間中能實際工作的僅有一個。

在該範例中，八件微引擎16a至16h在圖1內予以說明。每一程式設計引擎16a至16h皆有能力處理八項硬體執行緒或資料內容。該八個程式設計引擎16a至16h操作包括記憶體資源14及匯流排介面(未顯示)的共用資源。硬體式多線處理器12包括一動態隨機存取記憶體(DRAM)控制器18a及一靜態隨機存取記憶體(SRAM)控制器18b。DRAM記憶體14a及DRAM控制器18a通常係用於處理大量的資料，例如處理網路封包中的網路酬載。SRAM記憶體14b及SRAM控制器18b通常係使用於低潛時、快速存取工作(例如存取核心處理器20的查詢表、記憶體)等的網路環境中。

該八個程式設計引擎16a至16h根據資料特性存取DRAM記憶體14a或SRAM記憶體14b。因此，低潛時、低頻寬資料係儲存於SRAM記憶體14b之中，並且從SRAM記憶體14b中擷取，而潛時較不重要的高頻寬資料則係儲存於DRAM 14a之中，並且從DRAM 14a中擷取。程式設計引擎16可對DRAM控制器18a或SRAM控制器18b中任一方執行記憶體參考指令。

硬體式多重執行緒處理器12亦包含一核心處理器20，用以載入該設計引擎16的微碼控制。儘管其他類型的核心處理器可用於本發明之具體實施例，該範例中，核心處理器20係舊金山Intel®公司設計的一基於XScale™之結構。

核心處理器20執行一般用途電腦型功能例如處理協定、異常及封裝處理的特殊支援，其中程式設計引擎16為更詳細之處理如邊界條件而終止該封包。

處理器核心20執行一作業系統(未顯示)。透過該作業系統(operating system; OS)，處理器核心20能夠呼叫在程式設計引擎16a至16h上操作之功能。對於以XScale™結構實施的核心處理器20而言，可以使用的作業系統係例如華盛頓州西雅圖Microsoft®公司的Microsoft® NT即時、加州阿拉米達WindRiver®公司的VxWorks®即時作業系統或網際網路上可用的免費OS。

藉由SRAM或DRAM記憶體存取方式便能夠解釋硬體多重執行緒設計的優點。例如，從程式設計引擎16(如程式設計引擎16a)其中之一發出之資料內容(如Thread_0)SRAM存取

請求會引起SRAM控制器18b啟動對SRAM記憶體14b之存取。SRAM控制器18b存取SRAM記憶體14b，從SRAM記憶體14b中擷取資料並把資料傳回提出請求之程式設計引擎16。

在SRAM存取期間，如果程式設計引擎16a至16h之一僅有單條執行緒能夠運作的話，該程式設計引擎可能會停止運作，直到資料從該SRAM記憶體14b回傳為止。

藉由使用每個程式設計引擎16a至16h內的硬體式資料內容調換，該硬體式資料內容調換便能夠在同一個程式設計引擎中執行其他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DRAM記憶體14a。當Thread_1在該DRAM單元上運作並且Thread_0在SRAM單元上運作時，一新執行緒如Thread_2現在則能在程式設計引擎16內運作。Thread_2可以運作一定時間，直到其需要存取記憶體或執行一些其他長潛時運作，例如對一匯流排介面進行存取。所以，多重執行緒處理器12能夠利用一程式設計引擎16同時完成或進行匯流排運作、SRAM運作及DRAM運作，而且還有另一條執行緒或資料內容可用以處理更多的工作。

硬體式資料內容調換亦可同時完成各項工作。舉例來說，可能會有兩條穿線存取同一個共用資源，如SRAM記憶體14b。當完成其中一個程式設計引擎執行緒或資料內容所要

求的工作之後，每個分離的功能性單元(例如，SRAM控制器 18b及DRAM控制器 18a)便會回報一完成運作的旗標信號。當程式設計引擎 16a至16h收到該旗標，該程式設計引擎 16a至16h能決定該接通哪一條執行緒。

硬體式的多重執行緒處理器 12的其中一種應用實例為網路處理器。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(Media Access Controller；MAC)裝置(例如10/100 BaseT八進位媒體存取控制器或適應IEEE 802.3的Gigabit乙太裝置)般的網路裝置。通常，以網路處理器來說，硬體式的多線處理器 12能夠介接任何類型收發大量資料的通訊裝置或介面。在網路應用中作業的電腦處理系統 10能夠接收網路封包並以平行方式處理該等封包。

程式設計引擎內的暫存器：

參考圖 2，所顯示的係程式設計引擎 16之一示範程式設計引擎 16a。程式設計引擎 16a包括一控制儲存器 30，在一範例中其包括4096指令之RAM，每一指令寬為40位元。該RAM儲存程式設計引擎 16a執行的一微程式。控制儲存器 30內的微程式可由處理器核心 20(圖 1)載入。

除事件信號外(一執行中的執行緒之局部)，程式設計引擎 16a使用整體發信狀態。使用發信狀態，一執行中的執行緒可向全部程式設計引擎 16a至16h播送信號狀態。程式設計引擎內所有執行緒可分成該等發信狀態。

如上所述，程式設計引擎 16a支援八個資料內容的多重執

行緒之執行。這使一執行緒可在另一執行緒發出一記憶體參考後開始執行，並且必須等待該參考完成才可做更多工作。由於記憶體潛時非常重要，多重執行緒執行對維持程式設計引擎16a之有效硬體執行極其重要。多重執行緒之執行使程式設計引擎16可藉由在數個執行緒上執行有用的獨立工作而隱藏記憶體潛時。

可進行有效資料內容調換的程式設計引擎16a具有其本身之暫存器組、程式計數器及資料內容特定局部暫存器。每一資料內容具有一副本，因此無需為每一次資料內容調換在共用記憶體與程式設計引擎暫存器間移動資料內容具體資訊。快速資料內容調換使一資料內容執行計算時，其他資料內容等待輸入/輸出(input-output; I/O)，通常是等待外部記憶體存取完成，或等待來自另一資料內容或硬體單元的信號。

一般用途暫存器：

程式設計引擎16a藉由保持八個程式計數器及八個資料內容相關暫存器組來執行該等八個資料內容。有許多不同類型的資料內容相關暫存器，例如一般用途暫存器(general purpose register; GPR) 32，內部程式設計代理暫存器(未顯示)，靜態隨機存取記憶體(SRAM)輸入轉移暫存器34，動態隨機存取記憶體(DRAM)輸入轉移暫存器36，SRAM輸出轉移暫存器38，DRAM輸出轉移暫存器40。

GPR 32用以一般程式設計之目的。GPR 32專門在程式控制下讀取及寫入。GPR 32當用作一指令之源極時，對一執

行資料路徑44供應運算元。

執行資料路徑44可接收一或二個運算元、執行一運作及選擇性寫回一結果。執行資料路徑44包括一可定址內容記憶體(content addressable memory; CAM)45。CAM 45之每一項儲存一32位元值，可與源極運算元相比較。全部項目以平行方式比較，查詢結果為6位元值。

當GPR 32用作一指令之目的地時，其寫入執行資料路徑44之結果。程式設計引擎16a亦包括I/O轉移暫存器34、36、38及40，其用以在程式設計引擎16a及其外部位置(例如DRAM記憶體14a、SRAM記憶體14b等等)間轉移資料。

轉移暫存器

程式設計引擎16a亦包括轉移暫存器34、36、38及40。轉移暫存器34、36、38及40用以在程式設計引擎16a及其外部位置(例如DRAM、SRAM等等)間轉移資料。圖2所說明的轉移暫存器有四種類型，即輸入轉移暫存器及輸出轉移暫存器。

當輸入轉移暫存器用作一指令之源極時，對執行資料路徑44供應運算元，而輸出轉移暫存器用作一指令之源極時，其寫入執行資料路徑44之結果。

局部控制及狀態暫存器 (Local Control and Status Register; CSR)

局部控制及狀態暫存器(CSR)37係在執行資料路徑44外部，控制特定用途資訊。其藉由特殊指令(local_csr_rd及local_csr_wr)來讀取並寫入，通常存取頻率小於資料路徑暫

存器。

下一相鄰暫存器：

程式設計引擎16a還包括一百二十八(128)個下一相鄰(NN)暫存器，統稱為NN暫存器35。每一NN暫存器35當用作一指令之源極時，亦對執行資料路徑44供應運算元。每一NN暫存器35之寫入係藉由不限於相鄰程式設計引擎的一外部實體，或藉由各NN暫存器35駐存的同一程式設計引擎16a。特定暫存器之選擇係藉由一資料內容相關運作，其中暫存器序號編碼入指令中，或作為一環運作藉由例如CSR暫存器內的NN_Put (NN寫入位址)及NN_Get (NN讀取位址)選擇。

當前一相鄰程式設計引擎將帶有NN_Put之指令作為一目的地執行時使用NN_Put暫存器。寫入由此暫存器內該值選擇的NN暫存器，NN_Put之值即會增加(由127繞回0)。比較此暫存器之值與NN_Get暫存器之值，以決定何時判定NN_Full與NN_Empty狀態信號。

當NN暫存器35作為一源極存取時使用NN_Get暫存器，其在指令之源極區域內指定。讀取由此暫存器內該值選擇的NN暫存器，NN_Put之值即會增加(由127繞回0)。比較此暫存器之值與NN_Put暫存器之值，以決定何時判定NN_Full與NN_Empty 狀態信號。

具體而言，當各NN暫存器35用作一指令之目的地時，指令結果資料由程式設計引擎16a發出，通常係發至另一相鄰程式設計引擎。另一方面，當NN暫存器35用作一指令之目

管線階段60a係例如一規則時間間隔，其間一具體處理功能(例如功能62a)應用於資料封包54之一。一處理功能62可持續一個或多個管線階段60。例如，功能64持續兩個管線階段，即管線階段60b及60c。

一單一程式設計引擎，例如程式設計引擎16a，可構成一功能管線單元50。在功能管線單元50內，功能62a、64或62p經由功能管線單元50從一個程式設計引擎(例如，程式設計引擎16a)移至另一個程式設計引擎(例如，程式設計引擎16b)，以下將要說明。

參考圖4，資料封包54依順序指派給程式設計引擎資料內容58。因此，若「n」執行緒或資料內容58在程式設計引擎16a內執行，第一資料內容58，「PE0.1」在來自「PE0.n」之資料封包54到達前完成資料封包54之處理。使用此方法程式設計引擎16b可開始處理「n+1」封包。

將程式設計引擎16a之執行時間劃分為例如功能管線階段60a至60c，其結果係多個程式設計引擎16以平行方式執行一相同功能管線單元70。功能管線階段60a分佈於兩個程式設計引擎16a及16b，程式設計引擎16a及16b各執行八個資料內容。

在運作中，每一資料封包54較長時間地保留在資料內容58之一，等待更多程式設計引擎16加入功能管線單元50與70。在此範例中，資料封包54保留在一資料內容十六個資料封包到達時間(8資料內容x2程式設計引擎)，因為未要求資料內容PE0.1接受另一資料封包58，直到其他資料內容58已接

收他們地資料封包。

在此範例中，功能管線階段60a之功能62a可由程式設計引擎16a傳送至程式設計引擎16b。功能62a之傳送使用下一相鄰暫存器完成，如圖4內虛線80a至80c所說明。

功能管線階段60a至60m之序號等於功能管線單元50與70內程式設計引擎16a及16b之序號。此可確保一具體管線階段每一次只在一個程式設計引擎16內執行。

資料內容管線操作：

每一程式設計引擎16支援八個資料內容的多重執行緒之執行。其原因之一係使一執行緒可在另一執行緒發出一記憶體參考後開始執行，並且必須等待該參考完成才可做更多工作。由於記憶體潛時非常重要，此行為對維持程式設計引擎16a至16f之有效硬體執行極其重要。另一種說法是，如僅支援一單一執行緒執行，程式設計引擎會將大量迴圈時間耗費於等待參考完成，因而減少全部計算量。多重執行緒之執行使程式設計引擎可藉由在數個執行緒上執行有用的獨立工作而隱藏記憶體潛時。

每一程式設計引擎16a至16h(圖1)都有八個可用資料內容。為有效進行資料內容調換，程式設計引擎內八個資料內容中每一項具有其自身地暫存器組、程式計數器及資料內容特定局部暫存器。每一資料內容具有一副本，因此無需為每一次資料內容調換在共用記憶體與程式設計引擎暫存器間移動資料內容具體資訊。快速資料內容調換使一資料內容執行計算時，其他資料內容等待I/O，通常是等待外

部記憶體存取完成，或等待來自另一資料內容或硬體單元的信號。

參考圖 5，具體指定工作之資料內容利用 CAM 45a至 45c 保持在程式設計引擎 16a至 16c。封包之處理係以類似裝配線的管線方式利用 NN暫存器 35a至 35c將資料由一程式設計引擎傳送至其後相鄰之程式設計引擎。資料由管線一階段 90a傳送至其後一階段 90b，再由階段 90b至階段 90c，依此類推。換言之，資料向管線下一階段傳送使處理器迴圈之步驟可交疊。具體而言，當一指令在執行時，可擷取下一指令，其意味著「管」上一次可有多個指令。各位於處理之不同階段。

例如，資料可在管線上利用 NN暫存器 35a至 35c 將資料由一程式設計引擎 16傳送至下一程式設計引擎 16，如圖 5藉由範例所說明。此實施管線式處理之方法的優點係每一階段 90a至 90c的 CAM 45a至 45c包括之資訊對管線階段的全部八個資料內容始終有效。當來自處理最小資料封包必須穿過資料管線時可以利用資料管線方法。

參考圖 6，如上所述，資料管線操作要求一管階段(例如管階段 P)的結果資料發送至下一管階段，例如管階段 P+1 (100)。接著，將管階段 P (102)內程式設計引擎 16a之輸出寫入下一相鄰暫存器，該下一相鄰暫存器可作為一源極運算元在管階段 P+1 (104)由下一程式設計引擎 16b讀取。

參考圖 7，決定將下一相鄰暫存器之位元址寫入程式設計引擎 16b可使用兩種方法。入同樣的下一相鄰暫存器。另一

種方法係，為實施一內部處理引擎FIFO (302)可使用(300)程式設計引擎16a內一寫入指示暫存器及程式設計引擎16b內一讀取指示暫存器。程式設計引擎16a內寫入指示暫存器值及程式設計引擎16b內讀取指示暫存器值用以產生一完整指示，程式設計引擎16將資料插入FIFO (304)前會檢查該指示，並產生一空指示，供程式設計引擎16b從FIFO (306)移除資料使用。FIFO之下一相鄰組態可對管階段P與P+1間的資料內容提供彈性。當管階段P+1內一資料內容發現該下一相鄰FIFO為空，該資料內容可執行非運作功能，使該管階段保持一預定執行速率，或即使前一管階段未提供輸入亦以該相同速率「跳動」。

其他具體實施例：

在結合圖1至7說明的上述範例中，電腦處理系統10可使用各種網路處理器實施程式設計引擎16。

應瞭解雖然本發明已結合其具體說明而描述，以上說明係用於解釋而非限制本發明之範疇，本發明由隨附的申請專利範圍之範疇定義。其他方面、優點及修改部屬於下列申請專利範圍的範疇內。

【圖式簡單說明】

圖1係一處理系統的方塊圖。

圖2係圖1的有利地實踐本發明具體實施例之一的處理系統的詳細方塊圖。

圖3係圖1之處理系統的功能管線單元的方塊圖。

圖4係圖1的有利地實踐本發明具體實施例之一的處理系

統的詳細方塊說明圖。

圖5係一資料內容處理的簡化方塊圖。

圖6係有利地實踐本發明具體實施例之一的資料內容管線程式之流程說明圖。

圖7係決定下一相鄰暫存器位元址之程式的流程說明圖。

【圖式代表符號說明】

- 10 電腦處理系統
- 12 網路處理器
- 14 記憶體資源
- 16 程式設計引擎
- 20 核心處理器
- 30 控制儲存器
- 32 一般用途暫存器
- 34 SRAM輸入轉移暫存器
- 35 NN暫存器
- 36 DRAM輸入轉移暫存器
- 37 控制及狀態暫存器
- 38 SRAM輸出轉移暫存器
- 40 DRAM輸出轉移暫存器
- 42 局部記憶體
- 44 執行資料路徑
- 45 可定址內容記憶體
- 50 功能管線單元
- 52 資料單元

- 54 網路封包
- 56 局部暫存器單元
- 58 資料內容
- 60 管線階段
- 62 處理功能
- 64 管線功能
- 70 功能管線單元
- 14a 動態隨機存取記憶體
- 14b 靜態隨機存取記憶體
- 18a 動態隨機存取記憶體控制器
- 18b 靜態隨機存取記憶體控制器
- 60a 管階段
- 60b 管線階段
- 60c 管線階段
- 60m 管線段
- 62a 管線功能
- 60p 管線功能
- 90a 階段
- 90b 階段
- 90c 階段

伍、中文發明摘要：

本發明揭示一種在多硬體式資料內容及程式設計引擎內使用用以資料轉移之暫存器以利於高性能之資料處理的系統及方法。該系統及方法包括一處理器，其包括的程式引擎具有暫存器，用以將資料從位於一執行中程式設計引擎內的該等暫存器之一，轉移至位於一相鄰程式設計引擎內的下一該等暫存器之一。

陸、英文發明摘要：

A system and method for employing registers for data transfer in multiple hardware contexts and programming engines to facilitate high performance data processing. The system and method includes a processor that includes programming engines with registers for transferring data from one of the registers residing in an executing programming engine to a subsequent one of the registers residing in an adjacent programming engine.

柒、指定代表圖：

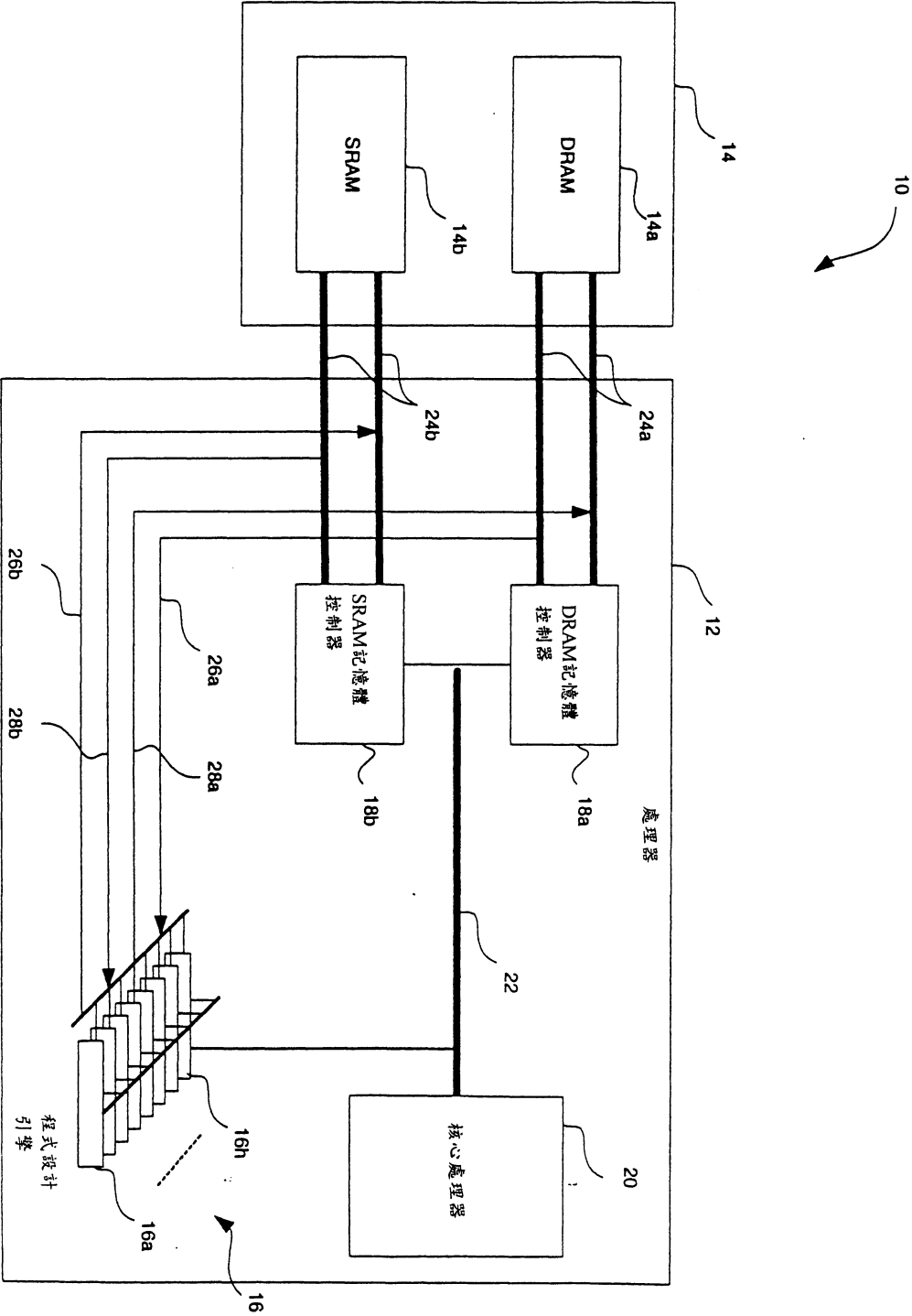
(一)本案指定代表圖為：第（ 2 ）圖。

(二)本代表圖之元件代表符號簡單說明：

- 16 程式設計引擎
- 30 控制儲存器
- 32 一般用途暫存器
- 34 SRAM輸入轉移暫存器
- 35 NN暫存器
- 36 DRAM輸入轉移暫存器
- 37 控制及狀態暫存器
- 38 SRAM輸出轉移暫存器
- 40 DRAM輸出轉移暫存器
- 42 局部記憶體
- 44 執行資料路徑
- 45 可定址內容記憶體

捌、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

圖 1





回 2

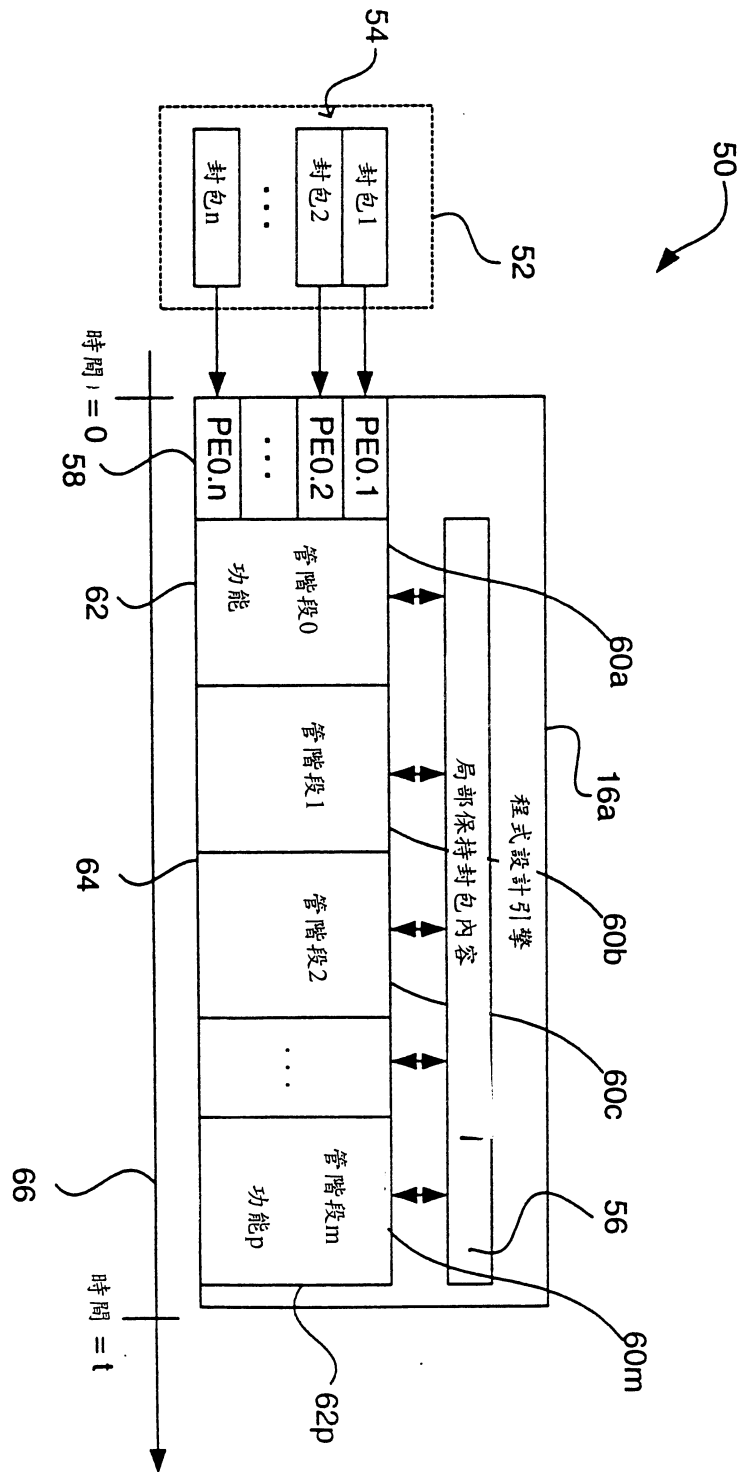


圖 3

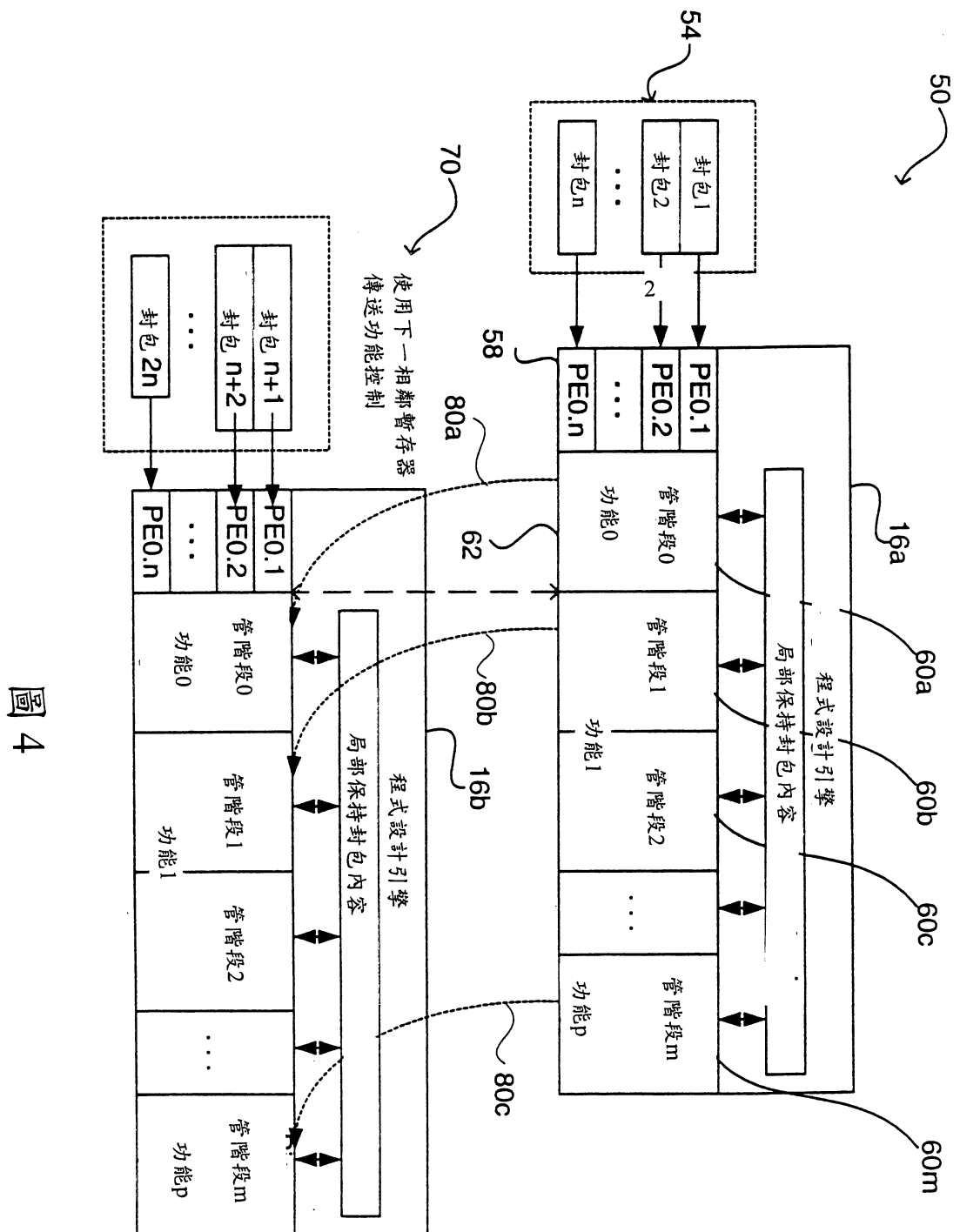


圖 4

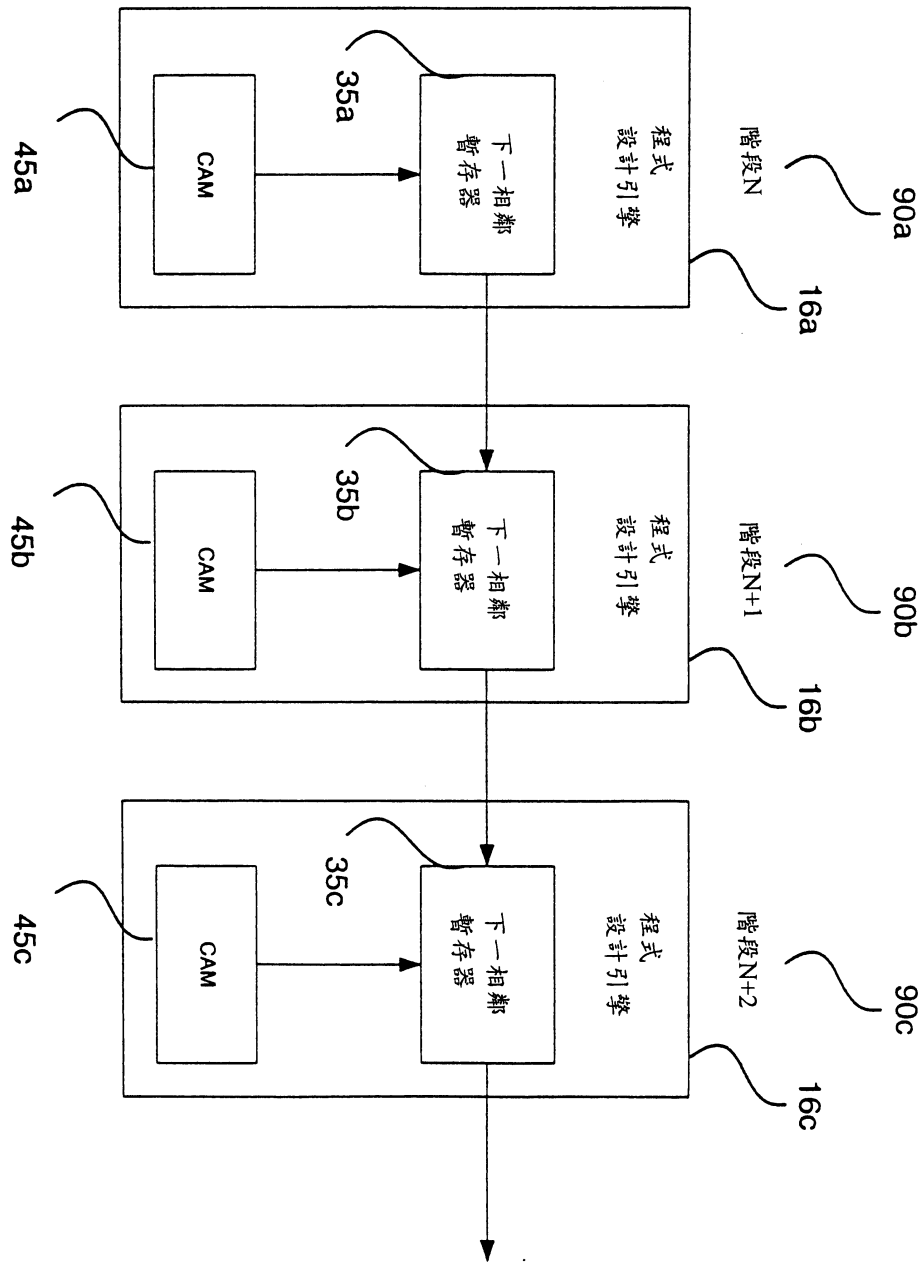


圖 5

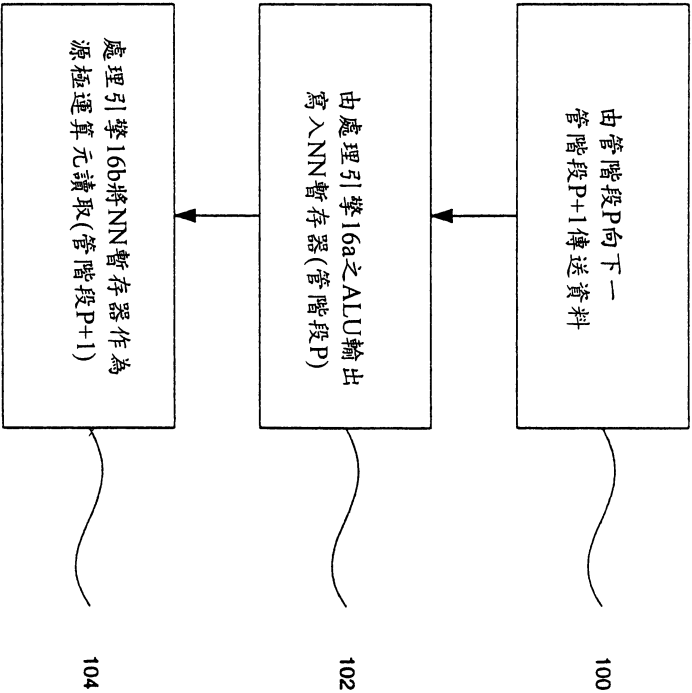


圖 6

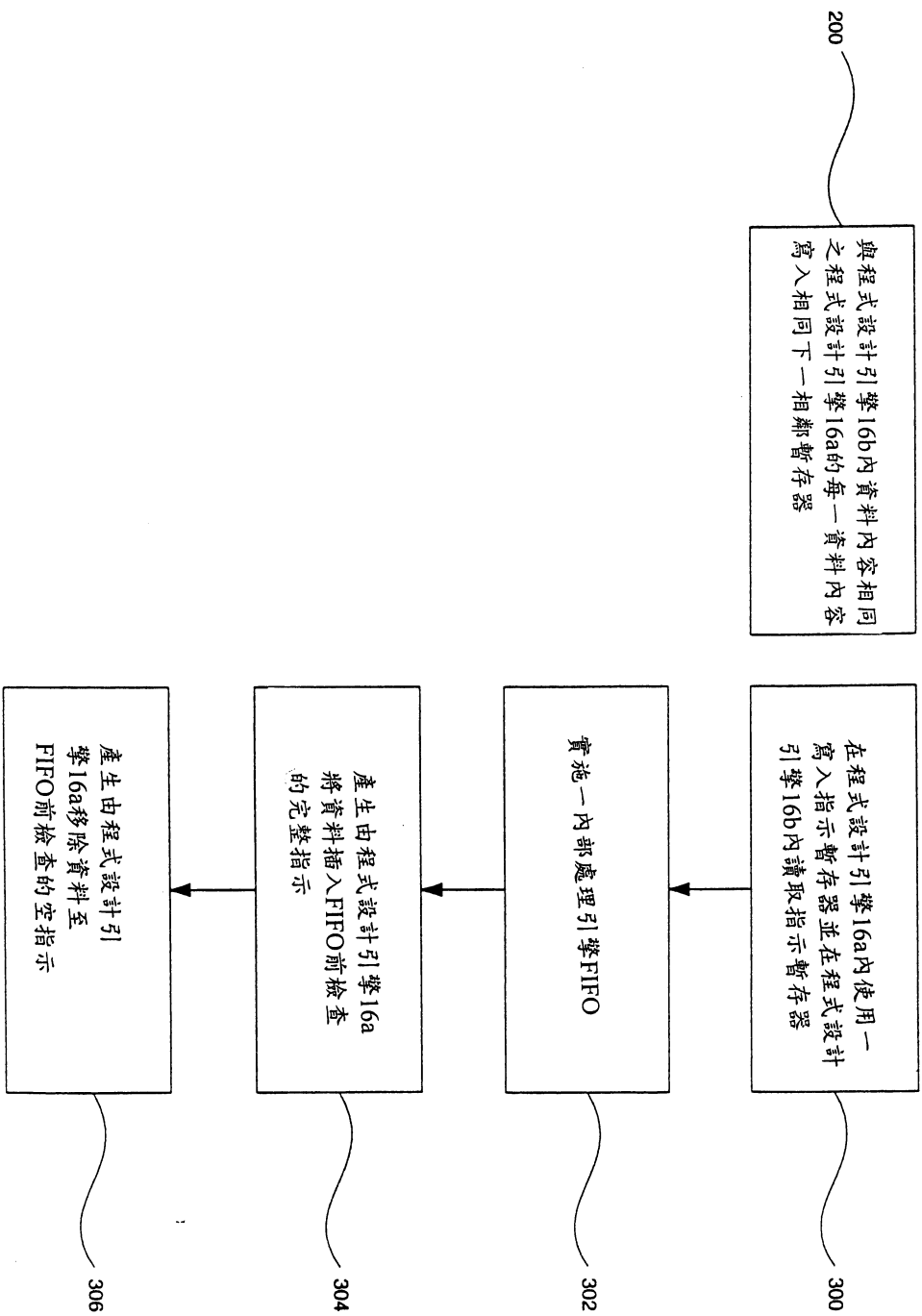


圖 7

的地時，指令結果資料寫入程式設計引擎16a內所選擇之NN暫存器35。當NN暫存器35用作一目的地時，資料不會像原先一樣由程式設計引擎16a發出。每一NN暫存器35以資料內容管線方法使用，如下文所述。

亦會使用一局部記憶體42。局部記憶體42包括位於程式設計引擎16a內之可定址記憶體。局部記憶體42專門在程式控制下讀取及寫入。局部記憶體42亦包括由全部程式設計引擎16共用之變數。共用變數由以下說明之程式設計引擎16a至16h在功能管線階段修正各指定工作。共用變數包括一重要部分，用以定義讀取-修正-寫入時間。重要部分在計算處理系統10內的執行與使用亦會在下面說明。

功能管線操作及管線階段：

參考圖3，所顯示的係程式設計引擎16a之功能管線單元50。功能管線單元50包括程式設計引擎16a及一資料單元52，其包括由程式設計引擎運作的資料，如網路封包54。程式設計引擎16a顯示其具有一局部暫存器單元56。局部暫存器單元56儲存來自資料封包54的資訊。

功能管線單元50內，當不同的功能在時間66由time=0至time=t過程中在資料封包54執行時，程式設計引擎16a之資料內容58，即程式設計引擎0.1(PE0.1)之程式設計引擎0.n(PE0.n)，保留在程式設計引擎16a。程式設計執行時間分為「m」個功能管線階段或管階段60a至60m。管線階段60a至60m之每一管線階段對管線內的資料執行不同的管線功能62a、64或62p。

拾、申請專利範圍：

1. 一種處理器，其包括：

複數個程式設計引擎，其包括複數個暫存器，用以將資料由位於執行中之一程式設計引擎內的該等複數個暫存器之一轉移至位於一相鄰程式設計引擎內的隨後該等複數個暫存器之一。

2. 如請求項1之處理器，其中該等複數個暫存器包含下一相鄰暫存器。

3. 如請求項1之處理器，其中該等複數個暫存器組態設定成為，為該等複數個程式設計引擎之封包處理指定工作。

4. 如請求項1之處理器，其中該等複數個暫存器組態設定成為，建立對應該等複數個程式設計引擎之程式設計階段。

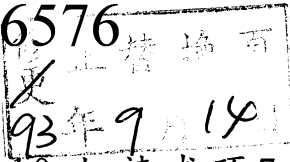
5. 如請求項1之處理器，其中該等複數個暫存器在該等程式設計階段之間建立複數個管線。

6. 如請求項1之處理器，其中該等複數個暫存器保持該管線之當前運作程式設計階段，且該相鄰程式設計引擎組態設定成為，保持該等複數個管線之一隨後程式設計階段。

7. 如請求項1之處理器，其中該等複數個暫存器藉由傳送該等複數個程式設計引擎中之功能資料的一功能管線控制單元支援一功能管線。

8. 如請求項7之處理器，其進一步包括一穿過該功能管線單元之一同步單元。

9. 如請求項7之處理器，其中該功能管線控制單元包括複數個功能管線階段。



- 10.如請求項7之處理器，其中每一該等複數個功能管線階段執行一不同的系統功能。
- 11.如請求項7之處理器，其中該等複數個程式設計引擎依順序處理一資料封包。
- 12.如請求項7之處理器，其中該資料封包被指定至該等複數個程式設計引擎之多個資料內容。
- 13.如請求項7之處理器，其中該等複數個程式設計引擎使用該系統之該功能管線單元執行一資料封包處理功能。
- 14.如請求項7之處理器，其中該等複數個程式設計引擎執行內部執行緒發信。
- 15.一種在程式設計引擎間轉移資料之方法，該方法包括：
從該等程式設計引擎之複數個資料暫存器讀取資料，用以在一管線單元之一平行處理器內處理該資料，其支援多個資料內容在每一該等程式設計引擎內執行；以及
寫入資料至該等程式設計引擎之該等複數個資料暫存器，該等複數個資料暫存器位於一執行中的程式設計引擎。
- 16.如請求項15之方法，其進一步包括為從該等複數個資料暫存器讀取資料及寫入資料至該等複數個資料暫存器提供下一相鄰暫存器。
- 17.如請求項15之方法，其進一步包括為從該等複數個程式設計引擎之封包處理指定工作。
- 18.如請求項15之方法，其進一步包括建立對應該等複數個程式設計引擎之程式設計階段，及建立該等程式設計階段間的複數個管線。

- 19.如請求項15之方法，其進一步包括保持該管線之該當前運作程式設計階段，且將該相鄰程式設計引擎組態設定為保持該等複數個管線之一隨後程式設計階段。
- 20.如請求項15之方法，其中資料之該讀取及該寫入包括藉由傳送該等複數個程式設計引擎中之功能資料的一功能管線控制單元支援一功能管線。
- 21.一種具有指令儲存於其上之電腦可讀取媒體，當為一平行處理器所執行時，用以使該平行處理器：
- 從程式設計引擎之複數個資料暫存器讀取資料，用以在一管線單元之一平行處理器內處理該資料，其支援多個資料內容在每一該等程式設計引擎內執行；以及
- 寫入資料至該等程式設計引擎之該等複數個資料暫存器，該等複數個資料暫存器位於一執行中的程式設計引擎。
- 22.如請求項21之電腦可讀取媒體，其進一步包括指令，用以使該處理器為從該等複數個資料暫存器讀取資料及寫入資料至該等複數個資料暫存器以提供下一相鄰暫存器。
- 23.如請求項21之電腦可讀取媒體，其中該等複數個暫存器使用該管線單元執行資料轉移。
- 24.如請求項21之電腦可讀取媒體，其進一步包括指令，用以使該處理器將該等複數個暫存器組態設定為為該等複數個程式設計引擎之封包處理指定工作。
- 25.一種電腦可讀取媒體，具有指令記錄其上，當藉由一機器執行時，會使該機器：

從一執行中程式設計引擎內第一複數個暫存器中之一，轉移資料至位於一相鄰程式設計引擎內隨後第二複數個暫存器中之一：

為該等複數個程式設計引擎之封包處理指定工作；

建立對應該等複數個程式設計引擎之程式設計階段；以及

建立該等程式設計階段間的複數個管線。

26.如請求項25之電腦可讀取媒體，進一步包括指令以使該機器從位於該等程式設計引擎之一當前執行程式設計引擎內一下相鄰暫存器轉移資料至位於鄰近該當前執行程式設計引擎之程式設計引擎內一隨後下一相鄰暫存器。

27.如請求項25之電腦可讀取媒體，進一步包括指令，使該等複數個暫存器保持該管線之該當前運作程式設計階段，該相鄰程式設計引擎保持該等複數個管線之一隨後程式設計階段。

28.一種多處理系統，其包括：

複數個程式設計引擎，其包括複數個下一相鄰暫存器，組態設定為從一執行中程式設計引擎內第一複數個暫存器之一轉移資料至位於一相鄰程式設計引擎內隨後第二複數個暫存器之一，以及穿過一功能管線單元處理資料封包之一同步單元。

29.如請求項28之多處理系統，其進一步包括該等複數個程式設計引擎之複數個程式設計階段利用的共用記憶體位置，該等共用變數包括定義該等記憶體位置之該讀取-修正-寫

I226576 替換頁
更
93年9月18日

入時間的一重要部分。

- 30.如請求項28之多處理系統，其中每一該等複數個程式設計引擎進一步包括一可定址內容記憶體(content addressable memory；CAM)。