I498819

(19)中華民國智慧財產局

(12)發明說明書公告本 (11)證書號數：TW I498819 B

(45)公告日： 中華民國 104 (2015) 年 09 月 01 日

(21)申請案號：101145697 (22)申請日： 中華民國 101 (2012) 年 12 月 05 日

(51)Int. Cl.： *G06F9/34 (2006.01)* *G06F9/38 (2006.01)*

(30)優先權：2011/12/06 美國 13/312,954

(71)申請人：輝達公司 (美國) NVIDIA CORPORATION (US)
美國

(72)發明人：邱小剛 QIU, XIAOGANG (US)；肖凱特 傑克 希萊爾 CHOQUETTE, JACK HILAIRE (US)；高索 曼紐爾 奧利維爾 GAUTHO, MANUEL OLIVIER (US)；蕭 明Y（麥可） SIU, MING Y. (MICHAEL) (US)

(74)代理人：蔡濱陽

(56)參考文獻：

TW 200634622A TW 200636573A

US 4367535 US 2006/0253689A1

審查人員：郭子意

申請專利範圍項數：10 項 圖式數：6 共 42 頁

(54)名稱

執行成型記憶體存取作業的系統和方法

SYSTEM AND METHOD FOR PERFORMING SHAPED MEMORY ACCESS OPERATIONS

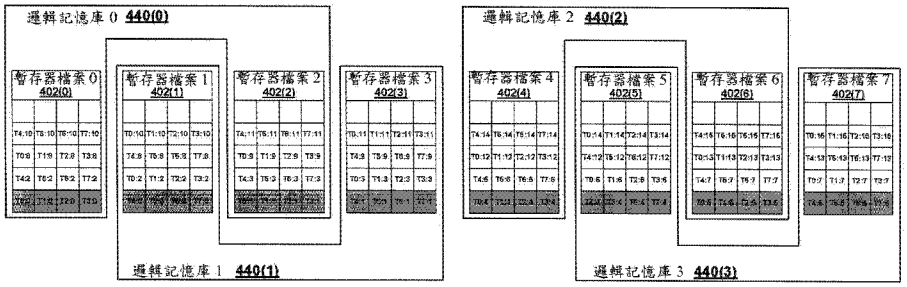
(57)摘要

本發明一具體實施例提出一種技術可提供一種有效率的方式來由一暫存器檔案取得運算元。特別是，該指令分派單元接收一或多個指令，其每一者包括一或多個運算元。該等運算元共同地被組織成可以形成一成型存取的一或多個運算元群組。該等運算元由該暫存器檔案取得，並儲存在一收集器中。一旦所有運算元皆被讀取並收集在該收集器中，該指令分派單元傳送該等指令和相對應的運算元到該串流多處理器內的功能性單元來執行。本發明一項好處在於多個運算元可在一單一暫存器存取作業中自該暫存器檔案取得，而不會有資源衝突。自該暫存器檔案取得運算元的效能藉由形成可有效率地取得呈現出經辨識的記憶體存取樣式之運算元之成型存取而改善。

One embodiment of the present invention sets forth a technique that provides an efficient way to retrieve operands from a register file. Specifically, the instruction dispatch unit receives one or more instructions, each of which includes one or more operands. Collectively, the operands are organized into one or more operand groups from which a shaped access may be formed. The operands are retrieved from the register file and stored in a collector. Once all operands are read and collected in the collector, the instruction dispatch unit transmits the instructions and corresponding operands to functional units within the streaming multiprocessor for execution. One advantage of the present invention is that multiple operands are retrieved from the register file in a single register access operation without resource conflict. Performance in retrieving operands from the register file is improved by forming shaped accesses that efficiently retrieve operands exhibiting recognized memory access patterns.

402・・・暫存器檔案

440・・・邏輯記憶庫



第四B圖

I498819

公告本

發明專利說明書

（本說明書格式、順序，請勿任意更動，※記號部分請勿填寫）

※ 申請案號：101145697

※ 申請日：101. 12. 5　　※IPC 分類：G06F 9/34 (2006.01)
G06F 9/38 (2006.01)

一、發明名稱：（中文/英文）

執行成型記憶體存取作業的系統和方法

SYSTEM AND METHOD FOR PERFORMING SHAPED MEMORY ACCESS OPERATIONS

二、中文發明摘要：

本發明一具體實施例提出一種技術可提供一種有效率的方式來由一暫存器檔案取得運算元。特別是，該指令分派單元接收一或多個指令，其每一者包括一或多個運算元。該等運算元共同地被組織成可以形成一成型存取的一或多個運算元群組。該等運算元由該暫存器檔案取得，並儲存在一收集器中。一旦所有運算元皆被讀取並收集在該收集器中，該指令分派單元傳送該等指令和相對應的運算元到該串流多處理器內的功能性單元來執行。本發明一項好處在於多個運算元可在一單一暫存器存取作業中自該暫存器檔案取得，而不會有資源衝突。自該暫存器檔案取得運算元的效能藉由形成可有效率地取得呈現出經辨識的記憶體存取樣式之運算元之成型存取而改善。

三、英文發明摘要：

One embodiment of the present invention sets forth a technique that provides an efficient way to retrieve operands from a register file. Specifically, the instruction dispatch unit receives one or more instructions, each of which includes one

or more operands. Collectively, the operands are organized into one or more operand groups from which a shaped access may be formed. The operands are retrieved from the register file and stored in a collector. Once all operands are read and collected in the collector, the instruction dispatch unit transmits the instructions and corresponding operands to functional units within the streaming multiprocessor for execution. One advantage of the present invention is that multiple operands are retrieved from the register file in a single register access operation without resource conflict. Performance in retrieving operands from the register file is improved by forming shaped accesses that efficiently retrieve operands exhibiting recognized memory access patterns.

四、指定代表圖：

(一)本案指定代表圖為：第（四B）圖。

(二)本代表圖之元件符號簡單說明：

402 暫存器檔案　　440 邏輯記憶庫

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】

本發明概略關於電腦架構，尤指一種在一暫存器檔案中運算元收集的系統和方法。

【相關技術】

在平行處理系統中常見的作法為設計一種可同時執行多個執行緒的處理器。當這些執行緒皆在執行相同指令序列(基本上每一執行緒有不同資料)時，在該等執行緒之間共用某些資源會有實質上的好處。例如，每一執行緒可以執行一指令來存取要由暫存器檔案的一共用記憶庫取得的一或多個運算元，其中每一執行緒存取該暫存器檔案之記憶庫中一不同的暫存器位址。此種作業可見於單一指令多重執行緒(SIMT, "Single instruction multi-thread")和單一指令多重資料(SIMD, "Single instruction multi-data")處理器當中。

在作業期間，該處理器可以橫跨多個執行緒來執行一指令，其中該指令自該等暫存器檔案的記憶庫存取一或多個運算元，而該等運算元係位在該等暫存器檔案的記憶庫內不同的暫存器位址處。然後該處理器執行一暫存器存取作業來取得該等運算元。例如，如果四個執行緒同時執行每一個需要三個運算元的一指令，則該處理器取得最多到十二個個別的運算元來執行該指令。當在相同的暫存器存取作業內可取得所有十二個運算元時，即可大幅改善效能。

由於多種限制，例如實體記憶體組態，某些暫存器組合可能無法同時存取。當兩個或更多的運算元位於無法同時被存取的暫存器檔案位置時，該處理器即會遭遇一暫存器記憶庫衝突。在這種狀況下，該處理器無法在一單一暫存器存取作業中取得所有運算元。

可避免暫存器檔案衝突的一種方法為針對由該目前指令

存取的每一運算元序列地執行一個別的暫存器存取作業。此種方法由於每一運算元係一次存取一個而可避免暫存器記憶庫衝突。但是，此種方法的一個缺點為該處理器無法使用不會造成一暫存器記憶庫衝突來存取運算元之相同的暫存器存取作業來取得多個運算元。例如，如果四個執行緒正在執行需要三個運算元的一指令，則該處理器將執行十二次個別的暫存器存取作業來避免暫存器記憶庫衝突。但是，該等運算元在整個該等暫存器檔案的記憶庫中的分佈可使得該處理器在少於十二次暫存器存取作業下取得所有運算元。在這種狀況下，關聯於記憶體存取作業可能有的效率將無法實現。

如前所述，本技術中需要的是一種更有效率的方法來由一暫存器檔案收集運算元。

【發明內容】

本發明一具體實施例提出一種用於執行暫存器記憶體作業的電腦實作方法。一指令分派單元接收要被橫跨複數運算元來執行的一指令。該指令分派單元辨識到其中儲存有該等複數運算元的複數暫存器檔案可經由一特定記憶體存取模式來存取。接著，該指令分派單元形成對應於該特定記憶體存取模式的一成型記憶體存取作業。然後該指令分派單元執行該成型記憶體存取作業來由該等複數暫存器檔案存取該等複數運算元。

該揭示技術的一項好處在於多個運算元可在一單一暫存器存取作業中自該暫存器檔案取得，而不會有資源衝突。

【實施方式】

在以下的說明中，許多特定細節係被提出來提供對於本發明之更為完整的瞭解。但是本技術專業人士將可瞭解到本發明可不利用一或多個這些特定細節來實施。

系統概述

第一圖係例示設置成實作本發明一或多種態樣之一電腦系統 100 的方塊圖。電腦系統 100 包括一中央處理單元(CPU)102 與一系統記憶體 104，其經由包括一記憶體橋接器 105 的互連接路徑進行通訊。記憶體橋接器 105 可為例如一北橋晶片，其經由一匯流排或其它通訊路徑 106(例如 HyperTransport 鏈路)連接到一 I/O(輸入/輸出)橋接器 107。I/O 橋接器 107 可為例如一南橋晶片，其接收來自一或多個使用者輸入裝置 108(例如鍵盤、滑鼠)的使用者輸入，並經由通訊路徑 106 及記憶體橋接器 105 轉送該輸入到 CPU 102。一平行處理子系統 112 經由一匯流排或第二通訊路徑 113(例如 PCI(周邊組件互連接) Express, 加速圖形處理埠、或 HyperTransport 鏈路)耦合至記憶體橋接器 105；在一具體實施例中，平行處理子系統 112 為一圖形子系統，其傳遞像素到一顯示器 110(例如一習用陰極射線管或液晶式的監視器)。一系統碟 114 亦連接至 I/O 橋接器 107。一交換器 116 提供 I/O 橋接器 107 與其它像是網路轉接器 118 與多種嵌入卡 120, 121 之其它組件之間的連接。其它組件(未明確顯示)，包括有通用串列匯流排(USB, "Universal serial bus")或其它埠連接、光碟(CD)驅動器、數位視訊碟(DVD)驅動器、薄膜記錄裝置及類似者，其亦可連接至 I/O 橋接器 107。第一圖所示之該等多種通訊路徑(包括特定名稱的通訊路徑 106 與 113)可使用任何適當的協定來實作，例如 PCI(周邊組件互連, Peripheral Component Interconnect)、PCI Express(PCI 快速, PCI-E)、AGP(加速圖形通訊埠, Accelerated Graphics Port)、HyperTransport(超輸送)、或任何其它匯流排或點對點通訊協定、及不同裝置之間的連接，皆可使用如本技術中所知的不同協定。

在一具體實施例中，平行處理子系統 112 加入可針對圖形及視訊處理最佳化的電路，其包括例如視訊輸出電路，且構成一圖形處理器(GPU)。在另一具體實施例中，平行處理子系統

112 加入可針對一般性目的處理最佳化的電路，而可保留底層的運算架構，在此處會有更為詳細的說明。在又另一具體實施例中，平行處理子系統 112 在一單一子系統中可被整合於一或多個其它系統元件，例如結合記憶體橋接器 105、CPU 102、及 I/O 橋接器 107 而形成一系統上晶片(SoC, "System on chip")。

將可瞭解到此處所示的系統僅為例示性，其有可能有多種變化及修正。該連接拓樸，包括橋接器的數目與配置、CPU 102 的數目及平行處理子系統 112 的數目皆可視需要修改。例如，在一些具體實施例中，系統記憶體 104 直接連接至 CPU 102 而非透過一橋接器耦接，而其它裝置透過記憶體橋接器 105 及 CPU 102 與系統記憶體 104 進行通訊。在其它可替代的拓樸中，平行處理子系統 112 連接至 I/O 橋接器 107 或直接連接至 CPU 102，而非連接至記憶體橋接器 105。在又其它具體實施例中，除了做為一或多個分散的裝置之外，I/O 橋接器 107 及記憶體橋接器 105 可被整合到一單一晶片當中。大型具體實施例可包括兩個或更多的 CPU 102，及兩個或更多的平行處理子系統 112。此處所示的該等特定組件皆為選擇性的；例如其可支援任何數目的嵌入卡或周邊裝置。在一些具體實施例中，交換器 116 被省略，且網路轉接器 118 及嵌入卡 120、121 直接連接至 I/O 橋接器 107。

第二圖例示根據本發明一具體實施例之一平行處理子系統 112。如所示，平行處理子系統 112 包括一或多個平行處理單元(PPU, "parallel processing unit")202，其每一者耦合於一局部平行處理(PP, "parallel processing")記憶體 204。概言之，一平行處理子系統包括數目為 U 的 PPU，其中 U≧1。(在此處類似物件的多個實例標示為辨識該物件之參考編號，而括號中的數目辨識所需要的實例)。PPU 202 及平行處理記憶體 204 可以使用一或多個積體電路裝置來實作，例如可程式化處理器，

特殊應用積體電路(ASIC, "Application specific integrated circuits")，或記憶體裝置，或以任何其它技術上可行的方式來實作。

請再次參照第一圖以及第二圖，在一些具體實施例中，平行處理子系統 112 中部份或所有的 PPU 202 為圖形處理器，其具有顯像管線，其能夠設置成執行關於自 CPU 102 及/或系統記憶體 104 經由記憶體橋接器 105 及第二通訊路徑 113 所供應的圖形資料產生像素資料的多種作業，與本地平行處理記憶體 204 進行互動(其能夠做為圖形記憶體，其包括例如一習用像框緩衝器)，以儲存及更新像素資料，傳遞像素資料到顯示器 110 及類似者。在一些具體實施例中，平行處理子系統 112 可以包括可操作為圖形處理器的一或多個 PPU 202，及用於通用型運算的一或多個其它 PPU 202。該等 PPU 可為相同或不同，且每個 PPU 可以具有一專屬的平行處理記憶體裝置或並無專屬的平行處理記憶體裝置。在平行處理系統 112 中一或多個 PPU 202 可以輸出資料到顯示器 110，或在平行處理系統 112 中每個 PPU 202 可以輸出資料到一或多個顯示器 110。

在作業中，CPU 102 為電腦系統 100 的主控處理器，其控制及協調其它系統組件的作業。特別是 CPU 102 發出控制 PPU 202 之作業的命令。在一些具體實施例中，CPU 102 對每一 PPU 202 寫入一命令串流至一資料結構(未明確示於第一圖或第二圖中)，其可位於系統記憶體 104、平行處理記憶體 204 或可同時由 CPU 102 與 PPU 202 存取的其它儲存位置。指向至每一資料結構的一指標被寫入至一推入緩衝器來啟始在該資料結構中該命令串流之處理。PPU 202 自一或多個推入緩衝器讀取命令串流，然後相對於 CPU 102 的該作業非同步地執行命令。執行優先性可針對每一推入緩衝器藉由一應用程式經由裝置驅動器 103 來指定，以控制該等不同推入緩衝器的排程。

現在請回頭參照第二圖以及第一圖，每個 PPU 202 包括一

I/O(輸入/輸出)單元 205，其經由通訊路徑 113 與電腦系統 100 的其它部份進行通訊，其連接至記憶體橋接器 105(或在另一具體實施例中直接連接至 CPU 102)。PPU 202 與電腦系統 100 的其餘部份之連接亦可改變。在一些具體實施例中，平行處理子系統 112 係實作成一嵌入卡，其可被插入到電腦系統 100 的一擴充槽中。在其它具體實施例中，PPU 202 可利用一匯流排橋接器整合在一單一晶片上，例如記憶體橋接器 105 或 I/O 橋接器 107。在又其它的具體實施例中，PPU 202 之部份或所有元件可與 CPU 102 整合在一單一晶片上。

在一具體實施例中，通訊路徑 113 為一 PCI-EXPRESS 鏈路，其中如本技術中所熟知具有專屬的線路會分配給每個 PPU 202。其亦可使用其它通訊路徑。一 I/O 單元 205 產生封包(或其它信號)在通訊路徑 113 上傳輸，且亦自通訊路徑 113 接收所有進入的封包(或其它信號)，導引該等進入封包到 PPU 202 的適當組件。例如，關於處理工作的命令可被導引到一主控介面 206，而關於記憶體作業的命令(例如自平行處理記憶體 204 讀取或寫入其中)可被導引到一記憶體交叉開關單元 210。主控介面 206 讀取每個推入緩衝器，並輸出儲存在該推入緩衝器中的該命令串流至一前端 212。

每一 PPU 202 較佳地是實作一高度平行的處理架構。如詳細所示，PPU 202(0)包括一處理叢集陣列 230，其包括數目為 C 的通用處理叢集(GPC, “General processing clusters”)208，其中 $C \geq 1$。每一 GPC 208 能夠同時執行大量(例如數百或數千)的執行緒，其中每個執行緒為一程式的一實例。在多種應用中，不同的 GPC 208 可分配來處理不同種類的程式，或執行不同種類的運算。GPC 208 的分配可根據每種程式或運算所提升的工作負荷而改變。

GPC 208 由一任務/工作單元 207 內的一工作分配單元接收要被執行的處理任務。該工作分配單元接收指向至被編碼成

任務中介資料(TMD)且儲存在記憶體中的處理任務的指標。該等指向至 TMD 的指標被包括在儲存成一推入緩衝器且由前端單元 212 自主控介面 206 接收的該命令串流中。可被編碼成 TMD 的處理任務包括要被處理之資料的索引，以及定義了該資料要如何被處理的狀態參數和命令(例如那一個程式要被執行)。任務/工作單元 207 自前端 212 接收任務，並確保 GPC 208 在由該等 TMD 之每一者所指定的該處理啟始之前被設置成一有效狀態。一優先性可針對用於排程該處理任務之執行的每一 TMD 來指定。處理任務亦可自處理叢集陣列 230 接收。視需要，該 TMD 可包括一參數，其控制該 TMD 是否要被加入一處理任務清單(或指向至該等處理任務的指標清單)的頭端或尾端，藉此提供在優先性之上的另一控制層級。

記憶體介面 214 包括數目為 D 的區隔單元 215，其每一者被直接耦合至平行處理記憶體 204 的一部份，其中 $D \geq 1$。如所示，區隔單元 215 的該數目大致上等於動態隨機存取記憶體(DRAM) 220 的數目。在其它具體實施例中，區隔單元 215 的數目可能不等於記憶體裝置的數目。本技術專業人士將可瞭解到 DRAM 220 可由其它適當儲存裝置取代，並可為一般的習用設計。因此可省略詳細說明。顯像目標，例如圖框緩衝器或紋路地圖，其可儲存在不同 DRAM 220 中，其允許區隔單元 215 平行地寫入每個顯像目標之不同部份而有效率地使用平行處理記憶體 204 之可使用頻寬。

GPC 208 之任何一者可處理要被寫入到平行處理記憶體 204 內 DRAM 220 中任一者的資料。交叉開關單元 210 設置成導引每個 GPC 208 之輸出到任何區隔單元 215 的輸入或到另一個 GPC 208 做進一步處理。GPC 208 經由交叉開關單元 210 與記憶體介面 214 進行通訊，以自多個外部記憶體裝置讀取或寫入其中。在一具體實施例中，交叉開關單元 210 具有到記憶體介面 214 的一連接來與 I/O 單元 205 進行通訊，以及到局部

平行處理記憶體 204 的一連接，藉此使得不同 GPC 208 內該等處理核心能夠與系統記憶體 104 或並非位在 PPU 202 局部之其它記憶體進行通訊。在第二圖所示的該具體實施例中，交叉開關單元 210 直接連接於 I/O 單元 205。交叉開關單元 210 可使用虛擬通道來隔開 GPC 208 與區隔單元 215 之間的流量串流。

再次地，GPC 208 可被程式化來執行關於許多種應用之處理工作，其中包括但不限於線性及非線性資料轉換、影片及/或聲音資料的過濾、模型化作業(例如應用物理定律來決定物體的位置、速度及其它屬性)、影像顯像作業(例如鑲嵌遮影器、頂點遮影器、幾何遮影器及/或像素遮影器程式)等等。PPU 202 可將來自系統記憶體 104 及/或局部平行處理記憶體 204 的資料轉移到內部(晶片上)記憶體、處理該資料、及將結果資料寫回到系統記憶體 104 及/或局部平行處理記憶體 204，其中這些資料可由其它系統組件存取，包括 CPU 102 或另一個平行處理子系統 112。

一 PPU 202 可具有任何數量的局部平行處理記憶體 204，並不包括局部記憶體，並可用任何的組合來使用局部記憶體及系統記憶體。例如，一 PPU 202 可為在一統一記憶體架構(UMA, "Unified memory architecture")具體實施例中的一圖形處理器。在這些具體實施例中，將可提供少數或沒有專屬的圖形(平行處理)記憶體，且 PPU 202 將專有地或大致專有地使用系統記憶體。在 UMA 具體實施例中，一 PPU 202 可被整合到一橋接器晶片中或處理器晶片中，或提供成具有一高速鏈路(例如 PCI-EXPRESS)之一分離的晶片，其經由一橋接器晶片或其它通訊手段連接 PPU 202 到系統記憶體。

如上所述，任何數目的 PPU 202 可以包括在一平行處理子系統 112 中。例如，多個 PPU 202 可提供在一單一嵌入卡上，或多個嵌入卡可被連接至通訊路徑 113，或一或多個 PPU 202

可被整合到一橋接器晶片中。在一多 PPU 系統中 PPU 202 可彼此相同或彼此不相同。例如，不同的 PPU 202 可具有不同數目的處理核心、不同數量的局部平行處理記憶體等等。當存在有多個 PPU 202 時，那些 PPU 可平行地作業而以高於一單一 PPU 202 所可能的流量來處理資料。加入有一或多個 PPU 202 之系統可實作成多種組態及型式因子，其中包括桌上型、膝上型、或掌上型個人電腦、伺服器、工作站、遊戲主機、嵌入式系統及類似者。

多並行任務排程

多個處理任務可在 GPC 208 上並行地執行，且一處理任務於執行期間可以產生一或多個「子」(child)處理任務。任務/工作單元 207 接收該等任務，並動態地排程該等處理任務和子處理任務來由 GPC 208 執行。

第三 A 圖係根據本發明一具體實施例中第二圖之任務/工作單元 207 的方塊圖。任務/工作單元 207 包括一任務管理單元 300 和工作分配單元 340。任務管理單元 300 基於執行優先性程度組織要被排程的任務。針對每一優先性程度，任務管理單元 300 儲存一指標清單至對應於在排程器表 321 中該等任務的該等 TMD 322，其中該清單可利用一鏈接串列來實作。TMD 322 可被儲存在 PP 記憶體 204 或系統記憶體 104 中。任務管理單元 300 接受任務並儲存該等任務在排程器表 321 中的速率與任務管理單元 300 排程任務來執行的速率相脫離。因此，任務管理單元 300 可基於優先性資訊或使用其它技術來收集數個任務，例如循環式排程。

工作分配單元 340 包括一任務表 345，其具有位置，而每一位置可由將要被執行的一任務之 TMD 322 佔用。任務管理單元 300 在當任務表 345 中有空的位置時即可排程任務來執行。當沒有空位置時，不會佔用一位置的一較高優先性的任務可以逐出佔用一空位的一較低優先性的任務。當一任務被逐出

時，該任務即停止，且如果該任務的執行尚未完成，則指向至該任務的一指標被加入到要被排程的一任務指標清單，所以該任務的執行將在稍後恢復。當於一任務執行期間產生一子處理任務時，指向至該子任務的一指標被加入到要被排程的一任務指標清單。一子任務可由在處理叢集陣列230中執行的一TMD 322來產生。

不像是任務/工作單元207自前端212接收的一任務，子任務係自處理叢集陣列230接收。子任務不會被插入到推入緩衝器中或被傳送至該前端。當產生一子任務或該子任務的資料被儲存在記憶體中時，不會通知CPU 102。經由推入緩衝器提供的該等任務和子任務之間另一個差別在於經由該等推入緩衝器提供的該等任務由該應用程式定義，然而該等子任務係於該等任務的執行期間被動態地產生。

任務處理概述

第三B圖為根據本發明一具體實施例中第二圖之該等PPU 202中之一者內一GPC 208的方塊圖。每個GPC 208可構形成平行地執行大量的執行緒，其中術語「執行緒」(thread)代表在一特定組合的輸入資料上執行的一特定程式之實例。在一些具體實施例中，使用單一指令、多重資料(SIMD, "Single-instruction, multiple-data")指令發行技術來支援大量執行緒之平行執行，而不需要提供多個獨立指令單元。在其它具體實施例中，單一指令多重執行緒(SIMT, "Single-instruction, multiple-thread")技術係用來支援大量概略同步化執行緒的平行執行，其使用一共用指令單元設置成發出指令到GPU 208之每一者內一組處理引擎。不像是一SIMD執行方式，其中所有處理引擎基本上執行相同的指令，SIMT的執行係允許不同的執行緒經由一給定執行緒程式而更可立即地遵循相異的執行路徑。本技術專業人士將可瞭解到一SIMD處理規範代表一SIMT處理規範的一功能子集合。

GPC 208 的作業較佳地是經由一管線管理員 305 控制，其可分配處理任務至串流多處理器(SM, "Streaming multiprocessor")310。管線管理員 305 亦可設置成藉由指定 SM 310 輸出之已處理資料的目的地來控制一工作分配交叉開關 330。

在一具體實施例中，每個 GPC 208 包括 M 個數目的 SM 310，其中 M≥1，每個 SM 310 設置成處理一或多個執行緒群組。同時，每個 SM 310 較佳地是包括可被管線化的相同組合的功能性執行單元(例如執行單元合載入儲存單元，如第三 C 圖中所示的執行單元 302 和 LSU 303)，允許在一先前指令已經完成之前發出一新指令，其為本技術中已知。並可提供任何功能性執行單元的組合。在一具體實施例中，該等功能單元支援多種運算，其中包括整數及浮點數算術(例如加法及乘法)，比較運算，布林運算(AND, OR, XOR)、位元偏位，及多種代數函數的運算(例如平面內插、三角函數、指數、及對數函數等)；及相同的功能單元硬體可被利用來執行不同的運算。

傳送到一特定 GPC 208 之該等系列的指令構成一執行緒，如先前此處所定義者，橫跨一 SM 310 內該等平行處理引擎(未示出)並行地執行某個數目之執行緒的集合在此稱之為「包繞」(warp)或「執行緒群組」(thread group)。如此處所使用者，一「執行緒群組」代表同步地對於不同輸入資料執行相同程式的一執行緒的群組，該群組的每一執行緒被指定給一 SM 310 內的一不同處理引擎。一執行緒群組可包括比 SM 310 內處理引擎的數目要少的執行緒，其中當該執行緒群組正在被處理的循環期間一些處理引擎將為閒置。一執行緒群組亦可包括比 SM 310 內處理引擎之數目要更多的執行緒，其中處理將發生在連續的時脈循環之上。因為每個 SM 310 可並行地支援最多到 G 個執行緒群組，因此在任何給定時間在 GPC 208 中最高可執行 G * M 個執行緒群組。

此外，在相同時間於一 SM 310 內可以啟動複數相關的執行緒群組(在不同的執行階段)。此執行緒群組的集合在此處稱之為「協同執行緒陣列」(CTA, "Cooperative thread array")或「執行緒陣列」(thread array)。一特定 CTA 之大小等於 m*k，其中 k 為在一執行緒群組中並行地執行的執行緒之數目，其基本上為 SM 310 內平行處理引擎數目之整數倍數，而 m 為在 SM 310 內同時啟動的執行緒群組之數目。一 CTA 的大小概略由程式師及該 CTA 可使用之硬體資源(例如記憶體或暫存器)的數量所決定。

每一 SM 310 包含一階(L1)快取(如第三 C 圖所示)，或使用在 SM 310 外部一相對應 L1 快取中用於執行載入與儲存作業的空間。每個 SM 310 亦可存取到所有 GPC 208 之間共用的二階(L2)快取，並可用於在執行緒之間傳送資料。最後，SM 310 亦可存取到晶片外的「通用」記憶體，其可包括例如平行處理記憶體 204 及/或系統記憶體 104。應瞭解到在 PPU 202 外部的任何記憶體皆可做為通用記憶體。此外，一 1.5 階(L1.5)快取 335 可包括在 GPC 208 之內，設置成由 SM 310 要求經由記憶體介面 214 接收及保持自記憶體提取的資料，其中包括指令、一致性資料與常數資料，並提供該要求的資料至 SM 310。在 GPC 208 中具有多個 SM 310 的具體實施例較佳地是共用被快取在 L1.5 快取 335 中的共通指令和資料。

每一 GPC 208 可包括一記憶體管理單元(MMU, "Memory management unit")328，其設置成將虛擬位址映射到實體位置。在其它具體實施例中，MMU 328 可存在於記憶體介面 214 內。MMU 328 包括一組頁表項(PTE, "Page table entries")，用於將一虛擬位置映射到一瓷磚的一實體位址，或是一快取線索引。MMU 328 可以包括位址轉譯旁看緩衝器(TLB, "Translation lookaside buffer")或可以存在於多處理器 SM 310 或 L1 快取或 GPC 208 內的快取。該實體位址被處理成分佈表面資料存取局

部性，以允許在區隔單元215之間有效率的要求交叉。該快取線索引可用於決定一快取線的一要求為一命中或錯失。

在圖形和運算應用中，一GPC 208可設置成使得每個SM 310耦合於一紋路單元315，用於執行紋路映射作業，例如決定紋路樣本位置、讀取紋路資料及過濾該紋路資料。紋路資料自一內部紋路L1快取(未示出)讀取，或是在一些具體實施例中自SM 310內的L1快取讀取，且視需要自一L2快取、平行處理記憶體204或系統記憶體104提取。每一SM 310輸出已處理的任務至工作分配交叉開關330，藉以提供該已處理的任務至另一GPC 208進行進一步處理，或是將該已處理的任務經由交叉開關單元210儲存在由所有GPC 208之間共用的一L2快取、平行處理記憶體204或系統記憶體104中。一preROP(預先掃描場化作業)325設置成自SM 310接收資料、導引資料到隔間單元215內的ROP單元、並進行色彩混合的最佳化、組織像素色彩資料、並執行位址轉譯。

將可瞭解到此處所示的核心架構僅為例示性，其有可能有多種變化及修正。在一GPC 208內可包括任何數目的處理單元，例如SM 310或紋路單元315、preROP 325。再者，如第二圖所示，一PPU 202可以包括任何數目的GPC 208，其較佳地是在功能上彼此類似，所以執行行為並不會根據是那一個GPC 208接收一特定處理任務而決定。再者，每個GPC 208較佳地是與其它使用分開且不同的處理單元、L1快取的GPC 208獨立地運作，以針對一或多個應用程式來執行任務。

本技術專業人士將可瞭解到在第一、二、三A和三B圖中所述之該架構並未以任何方式限制本發明之範圍，而此處所教示的技術可以實作在任何適當設置的處理單元上，其包括但不限於一或多個CPU、一或多個多核心CPU、一或多個PPU 202、一或多個GPC 208、一或多個圖形或特殊目的處理單元或類似者，其皆不背離本發明之範圍。

在本發明之具體實施例中，需要使用 PPU 202 或一運算系統的其它處理器來使用執行緒陣列執行一般性運算。在該執行緒陣列中每一執行緒被指定一唯一執行緒識別(thread ID)，其可在該執行緒的執行期間由該執行緒存取。可被定義成一維或多維度數值的執行緒 ID 控制該執行緒的處理行為之多種態樣。例如，一執行緒 ID 可用於決定一執行緒要做處理的是該輸入資料集的那一部份，及/或決定一執行緒要產生或寫入的是在一輸出資料集的那一部份。

每個執行緒指令的一序列可以包括至少一指令來定義該代表性執行緒和該執行緒陣列的一或多個其它執行緒之間的一協同行為。例如，每個執行緒的該指令序列可以包括一指令來在該序列中一特定點處中止該代表性執行緒之作業的執行，直到當該等其它執行緒中一或多者到達該特定點為止，該代表性執行緒的一指令係儲存資料在該等其它執行緒中一或多者可存取的一共用記憶體中，該代表性執行緒的一指令係基於它們的執行緒 ID 原子性地讀取和更新儲存在該等其它執行緒中一或多者可存取的一共用記憶體中的資料，或類似者。該 CTA 程式亦可包括一指令來運算資料在該共用記憶體中要被讀取的一位址，利用該位址為執行緒 ID 的函數。藉由定義適當的函數和提供同步化技術，資料可藉由一 CTA 的一執行緒被寫入到共用記憶體中一給定的位置，並以一可預測的方式由該相同 CTA 的一不同執行緒自該位置讀取。因此，即可支援可在執行緒當中共用任何需要的資料型式，且在一 CTA 中任何執行緒能夠與該相同 CTA 中任何其它執行緒共用資料。如果有的話，在一 CTA 的執行緒當中資料共用的程度係由該 CTA 程式決定；因此，應瞭解到在使用 CTA 的一特定應用中，根據該 CTA 程式，一 CTA 的該等執行緒可以或不需要實際地彼此共用資料，該等術語"CTA"和「執行緒陣列」在此處為同義地使用。

第三 C 圖為根據本發明一具體實施例中第三 B 圖的 SM 310 的方塊圖。SM 310 包括一指令 L1 快取 370，其設置成經由 L1.5 快取 335 自記憶體接收指令和常數。一包繞排程器和指令單元 312 自指令 L1 快取 370 接收指令和常數，並根據該等指令和常數控制局部暫存器檔案 304 和 SM 310 功能性單元。SM 310 功能性單元包括 N 個執行(執行或處理)單元 302 和 P 個載入儲存單元(LSU)303。

SM 310 提供具有不同程度存取性的晶片上(內部)資料儲存。特殊暫存器(未示出)可由 LSU 303 讀取但不能寫入，並用於儲存定義每一執行緒的「位置」之參數。在一具體實施例中，特殊暫存器包括每一執行緒(或 SM 310 內每一執行單元 302)的一暫存器，用於儲存一執行緒 ID；每一執行緒 ID 暫存器僅可由執行單元 302 的個別單元存取。特殊暫存器亦可包括額外的暫存器、其可由執行由儲存一 CTA 識別的一 TMD 322(或由所有 LSU 303)所代表的相同處理任務的所有執行緒來讀取、該等 CTA 維度、該 CTA 所屬的一格柵的該等維度(或如果 TMD 322 編碼一佇列任務而非一格柵任務時的佇列位置)，以及該 CTA 被指定到的 TMD 322 之一識別。

如果 TMD 322 為一格柵 TMD，TMD 322 的執行造成固定數目的 CTA 被啟動及執行，以處理儲存在佇列 525 中固定數量的資料。CTA 的數目係界定成是格柵的寬、高與深的乘積。該固定數量的資料可以儲存在 TMD 322 中，或者 TMD 322 可以儲存指向至將由該等 CTA 處理的資料之一指標。TMD 322 亦儲存由該等 CTA 執行的該程式之一開始位址。

如果 TMD 322 為一佇列 TMD，則使用 TMD 322 的一佇列特徵，代表要被處理的資料量並不一定是固定的。佇列項目儲存資料來由指定給 TMD 322 的該等 CTA 做處理。該等佇列項目亦可代表於一執行緒的執行期間由另一 TMD 322 產生的一子任務，藉此提供巢化的平行度。基本上，該執行緒或包括

該執行緒的 CTA 之執行被中止，直到該子任務的執行完成為止。該佇列可儲存在 TMD 322 中或隔離於 TMD 322，其中 TMD 322 儲存指向至該佇列的一佇列指標。較佳地是，當代表該子任務的 TMD 322 正在執行時，由該子任務產生的資料可被寫入到該佇列。該佇列可實作成一圓形佇列，所以資料的總量並不限於該佇列的大小。

屬於一網格的 CTA 具有隱式格柵寬度、高度和深度參數以指明該網格內個別 CTA 的位置。特殊暫存器回應於經由前端 212 自裝置驅動器 103 接收的該等命令而被寫入，且於一處理任務的執行期間不會改變。前端 212 排程每一處理任務來執行。每一 CTA 關聯於一特定 TMD 322 來用於一或多個任務的並行執行。此外，一單一 GPC 208 可以並行地執行多個任務。

一參數記憶體(未示出)儲存運行時間參數(常數)，其可被相同 CTA(或任何 LSU 303)內任何執行緒讀取但無法寫入。在一具體實施例中，裝置驅動器 103 在導引 SM 310 開始使用這些參數的一項任務之執行之前提供參數至該參數記憶體。任何 CTA(或 SM 310 內任何執行單元 302)內的任何執行緒能夠經由一記憶體介面 214 存取共通記憶體。共通記憶體的一些部份可被儲存在 L1 快取 320 中。

局部暫存器檔案 304 由每一執行緒使用做為暫存空間；每一暫存器被分配做為一執行緒的專屬使用，而在任何局部暫存器檔案 304 中的資料僅可由該暫存器被分配到的該執行緒來存取。局部暫存器檔案 304 可實作成被實體或邏輯性地區分成 P 條線路的一暫存器檔案，其每一者具有某個數目的項目(其中每個項目可以儲存例如 32 位元的字元)。一條線路被指定給 N 個執行單元 302 和 P 個載入儲存單元 LSU 303 的每一者，且在不同線路中的相對應的項目可存在有執行相同程式的不同執行緒之資料來實施 SIMD 執行。該等線路的不同部份可被分配給該等 G 個並行執行緒群組之不同的執行緒群組，所以在

局部暫存器檔案 304 中一給定項目僅可由一特定執行緒存取。在一具體實施例中，局部暫存器檔案 304 內某些項目被保留來儲存執行緒識別及實作該等特殊暫存器之一。此外，一一致性 L1 快取 375 儲存 N 個執行單元 302 和 P 個載入儲存單元 LSU 303 之每一線路的一致性或常數值。

共用的記憶體 306 可由一單一 CTA 內的執行緒存取；換言之，在共用記憶體 306 中任何位置可由該相同 CTA 內任何執行緒存取(或可由 SM 310 內任何處理引擎存取)。共用的記憶體 306 可實作成利用一互連接的一共用暫存器檔案或共用的晶片上快取記憶體，其可允許任何處理引擎可讀取或寫入到該共用記憶體中任何位置。在其它具體實施例中，共用的狀態空間可映射到晶片外記憶體的一每一 CTA 的區域之上，且被快取在 L1 快取 320 中。該參數記憶體可被實作成該相同共用的暫存器檔案或實作共用記憶體 306 的共用快取記憶體內一指定的區段，或是實作成一獨立的共用暫存器檔案或 LSU 303 具有唯讀性存取的晶片上快取記憶體。在一具體實施例中，實作該參數記憶體的該區域亦用於儲存該 CTA ID 與任務 ID，以及 CTA 與網格維度或佇列位置，實作該等特殊暫存器的某些部份。在 SM 310 中每一 LSU 303 耦合至一統一位址映射單元 352，其轉換提供用於載入與儲存在一統一記憶體空間中指定的指令之一位址成為在每一不同記憶體空間中的一位址。因此，一指令可用於藉由指定在該統一記憶體空間中一位址來存取該等局部、共用、或共通記憶體空間之任一者。

在每一 SM 310 中的 L1 快取 320 可用於快取私密的每一執行緒之局部資料，以及每一應用程式的共通資料。在一些具體實施例中，該由每一 CTA 共用的資料可被快取在 L1 快取 320 中。LSU 303 經由一記憶體和快取互連接 380 耦合至共用記憶體 306 和 L1 快取 320。

暫存器檔案中的運算元收集

如第四 A 到四 D 圖所示，暫存器檔案 402 包括有暫存器，其可儲存該處理器在執行指令時所存取的資料，例如指令運算元。每一個別的暫存器胞由一執行緒位址和一暫存器位址來辨識。例如，在暫存器檔案 0 402(0)的左上角之暫存器為執行緒 4 的暫存器 10，所以由該命名 T4:10 來辨識。同樣地，在暫存器檔案 0 402(0)的右下角之暫存器為執行緒 3 的暫存器 0，所以該命名 T3:0 來辨識。每一暫存器檔案被組織成使得在每一暫存器檔案中一列的暫存器於一給定的暫存器存取作業期間可被同時地存取。例如，暫存器 T0:0、T1:0、T2:0 和 T3:0 在一單一暫存器存取作業中可被同時地存取。暫存器檔案 402 根據於一特定暫存器存取作業期間被存取之運算元種類而被組織成邏輯記憶庫。當運算元自該等暫存器檔案讀取時，該等暫存器存取作業橫跨該等暫存器檔案形成樣式，在此處辨識為「成型存取」(shaped accesses)。將暫存器放置在該等暫存器檔案內以及於暫存器存取作業期間形成的該等成型存取樣式可具有共通運算元組態之優點。因此，使用成型存取相對於針對每一運算元之個別的序列暫存器存取作業，可降低自該等暫存器檔案收集指令運算元的潛時，因而可改善效能。第四 A 到四 D 圖之每一圖例示了不同的示例性成型存取樣式。

第四 A 圖例示根據本發明一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫。如所示，該等暫存器檔案的記憶庫包括暫存器檔案 402 和邏輯記憶庫 420。在此例中，邏輯記憶庫 420 可以形成使得邏輯記憶庫 0 420(0)包括暫存器檔案 0 402(0)和暫存器檔案 2 402(1)，邏輯記憶庫 1 420(1)包括暫存器檔案 1 402(2)和暫存器檔案 3 402(3)，依此類推。利用此配置，暫存器檔案 402 被最佳化來取得單一寬度運算元。例如，如果執行緒 0 到 7 皆存取儲存在暫存器 0 中的一單一寬度運算元，則該處理器可以形成邏輯記憶庫 0 420(0)內一成型存取，以取得該底列的暫存器，包括 T0:0 到 T7:0。該處理器可以形

成邏輯記憶庫 1 420(1)內一成型存取以取得一相同群組執行緒的一不同的單一寬度運算元，例如 T0:1 到 T7:1。同樣地，該處理器可形成邏輯記憶庫 2 420(2)和邏輯記憶庫 3 420(3)內一成型存取，以取得相同組合的八個執行緒之暫存器 4 和 5 的一單一寬度運算元。在一單一暫存器存取作業期間，該例示的成型存取取得八個執行緒之每一者的四個單一寬度運算元，如第四 A 圖中陰影區域所示。

第四 B 圖例示根據本發明另一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫。如所示，該等暫存器檔案的記憶庫包括暫存器檔案 402 和邏輯記憶庫 440。在此例中，邏輯記憶庫 440 可以形成使得邏輯記憶庫 440(0)包括暫存器檔案 0 402(0)和暫存器檔案 2 402(2)，邏輯記憶庫 1 440(1)包括暫存器檔案 1 402(1)和暫存器檔案 3 402(3)，依此類推。利用此配置，暫存器檔案 402 被最佳化來取得雙重寬度運算元。例如，如果執行緒 0 到 3 皆存取儲存在暫存器配對 0-1 中的一雙重寬度運算元，則該處理器可以形成邏輯記憶庫 0 440(0)內一成型存取，以取得該底列的暫存器，包括 T0:0-1 到 T3:0-1。該處理器可以形成邏輯記憶庫 1 440(1)內一成型存取以取得一不同群組執行緒的一雙重寬度運算元，例如 T4:0-1 到 T7:0-1。同樣地，該處理器可以形成邏輯記憶庫 2 440(2)和邏輯記憶庫 3 440(3)內一成型存取，以由每一邏輯記憶庫取得四個執行緒的一雙重寬度運算元。例如，在一單一暫存器存取作業期間，該例示的成型存取取得八個執行緒之每一者的兩個雙重寬度運算元，如第四 B 圖中陰影區域所示。

第四 C 圖例示根據本發明又另一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫。如所示，該等暫存檔案的記憶庫包括暫存器檔案 402 和邏輯記憶庫 460。在此例中，邏輯記憶庫 460 可形成為使得兩個邏輯記憶庫 460(2) 460(3)被最佳化來取得單一寬度運算元，而兩個邏輯記憶庫 460(0) 460(1)

被最佳化來取得雙重寬度運算元。例如，於一單一暫存器存取作業期間，該例示的成型存取自邏輯記憶庫 0 460(0)和邏輯記憶庫 1 460(1)取得八個執行緒之每一者的一個雙重寬度運算元，也就是暫存器 T0:0-1 到 T7:0-1。在相同的暫存器存取作業期間，該例示的成型存取自邏輯記憶庫 2 460(2)和邏輯記憶庫 3 460(3)取得八個執行緒之每一者的兩個雙重寬度運算元，也就是執行緒 0-7 的暫存器 4 和 5。此示例性成型存取由第四 C 圖中的陰影區域所示。

第四 D 圖例示根據本發明再又另一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫。如所示，該等暫存器檔案的記憶庫包括暫存器檔案 402 和邏輯記憶庫 480。在此例中，邏輯記憶庫 480 可形成為使得邏輯記憶庫 480 被最佳化來取得四重寬度運算元。例如，於一單一暫存器存取作業期間，該例示的成型存取自邏輯記憶庫 0/1 480(0)取得四個執行緒之每一者的一個四重寬度運算元，也就是暫存器 T0:0-3 到 T3:0-3。在相同的暫存器存取作業期間，該例示的成型存取自邏輯記憶庫 2/3 480(1)取得該等相同四個執行緒的一第二四重寬度運算元，也就是 T0:4-7 到 T3:4-7。此示例性成型存取由第四 D 圖中的陰影區域所示。

第五圖例示根據本發明一具體實施例第三 C 圖的該包繞排程器和指令單元 312 與局部暫存器檔案 304 之方塊圖。如所示，包繞排程器和指令單元 312 包括一包繞排程器 502 和一指令分派單元 504。包繞排程器 502 自指令 L1 快取 370 取得指令和常數，並排程該指令來當做一執行緒群組內一或多個執行緒來執行。指令分派單元 504 根據自指令 L1 快取 370 取得的該等指令和常數來控制局部暫存器檔案 304 和 SM 310 功能性單元。指令分派單元 504 評估該等指令運算元來決定該等運算元是否可配合在一經識別的成型存取樣式之內。然後指令分派單元 504 選擇一成型存取來由暫存器檔案 506 讀取一組運算

元。然後指令分派單元 504 產生該等位址和讀取致能來由暫存器檔案 506 讀取該等運算元。

局部暫存器檔案 304 包括一暫存器檔案 506 和一收集器 508。暫存器檔案 506 根據該成型存取的樣式於該暫存器存取作業期間取得該等運算元，並傳送該等運算元至收集器 508。收集器 508 根據在該等原始指令中該等運算元的位置來對準該等運算元，並依此儲存該等運算元。如果所有運算元皆於該第一暫存器存取作業期間被取得，則該等指令和運算元被傳送至 SM 310 功能性單元，其中包括執行單元 302 和載入儲存單元 303。如果部份運算元未在該第一暫存器存取作業期間被取得，則收集器 508 儲存每一成型存取的結果，直到所有運算元皆被收集為止。一旦所有運算元被儲存在收集器 508 之內，該等指令和運算元被傳送至 SM 310 功能性單元，其中包括執行單元 302 和載入儲存單元 303。

為了形成一成型存取，指令分派單元 504 自包繞排程器 502 接收一或多個指令。每一指令係關聯於一或多個運算元。該等運算元共同地被組織成可以形成一成型存取的一或多個運算元群組。指令分派單元 504 設置成可辨識由該等一或多個運算元群組所利用的該等不同的記憶體存取樣式。指令分派單元 504 形成成型存取來有效率地取得呈現一或多個這些記憶體存取樣式的運算元群組。在一具體實施例中，指令分派單元 504 將該等暫存器檔案的記憶庫分開成兩個分區，即暫存器檔案 0-3 402(0)-402(3)和暫存器檔案 4-7 402(4)-402(7)。針對這兩個暫存器檔案分區，指令分派單元 504 產生一或多次成型存取。指令分派單元 504 辨識出該等暫存器檔案的記憶庫內每一運算元的位置。指令分派單元 504 記錄需要該運算元的該等特定執行緒、保持這些執行緒之每一者的該運算元之暫存器檔案、和該暫存器檔案內運算元所在的該列。接著，指令分派單元 504 選擇該等運算元之一，並決定該運算元係為一單一寬

度、雙重寬度或四重寬度運算元。指令分派單元根據該運算元寬度形成該等暫存器檔案的該分區內的邏輯記憶庫。然後指令分派單元 504 形成橫跨該邏輯記憶庫的一成型存取，其中儲存有該經選擇的運算元，其基本上設置成讀取橫跨該邏輯記憶庫的相同列位址。同樣地，針對該等暫存器檔案的該分區內其它的邏輯記憶庫選擇一運算元，且亦形成該邏輯記憶庫的一成型存取。然後該程序針對暫存器檔案的其它分區來重複。因為該等暫存器檔案的兩個分區被獨立地處理，該等邏輯記憶庫的配置和該成型存取類型在該等暫存器檔案的兩個分區之每一分區中可以不同。

一旦該等成型存取被適當地辨識和設置時，指令分派單元 504 使得暫存器檔案 506 來讀取關聯於該等成型存取的該等暫存器，如上所述。指令分派單元 504 將該等經取得的運算元對準於該等相對應的指令，並傳送該等經對準的運算元至該等運算元被儲存的收集器 508。接著，指令分派單元 504 決定關聯於該目前指令集的所有運算元是否已被讀取、已自暫存器檔案 506 收集、並儲存在收集器 508 中。如果有額外的運算元要讀取，則指令分派單元 504 停止 SM 310 內該管線，並重複上述的該程序來針對該等剩餘的運算元形成額外的成型存取。該程序繼續直到所有關聯於該目前指令集的所有運算元被讀取和收集，此時指令分派單元 504 解除停止 SM 310 內的該管線，藉此使得該等指令可被執行。

前述程序的一示例如下所述。假設指令分派單元 504 能夠選擇位在該等暫存器檔案的第一分區內的暫存器 0-1 處的執行緒 0 之一雙重寬度運算元。該運算元可位在暫存器檔案 0 402(0)和暫存器檔案 2 402(2)的左下方胞中，如第四 C 圖所示。因此，指令分派單元 504 將形成設置成包括暫存器檔案 0 402(0)和暫存器檔案 2 402(2)的邏輯記憶庫 0 460(0)。同樣地，指令分派單元 504 將形成設置成包括暫存器檔案 1 402(1)和暫存器檔案

3 402(3)的邏輯記憶庫 1 460(1)。所得到的成型存取將針對四個執行緒之每一者存取兩組的雙重寬度運算元。同樣地，指令分派單元 504 能夠選擇位在該等暫存器檔案的第二分區內的暫存器 4 處的執行緒 0 之一單一寬度運算元。該運算元可位在暫存器檔案 4 402(4)的左下方胞中，如第四 C 圖所示。因此，指令分派單元 504 將形成設置成包括暫存器檔案 4 402(4)和暫存器檔案 5 402(5)的邏輯記憶庫 2 460(2)。同樣地，指令分派單元 504 將形成設置成包括暫存器檔案 6 402(6)和暫存器檔案 7 402(7)的邏輯記憶庫 3 460(3)。所得到的成型存取將針對八個執行緒之每一者存取兩組的單一寬度運算元。一旦完成該等成型存取，指令分派單元 504 將決定是否仍有額外的運算元。如果有更多的運算元要被收集，指令分派單元 504 將停止 SM 310 管線、形成額外的成型存取來收集任何剩餘的運算元、然後解除停止 SM 310 管線。

第六圖係根據本發明一具體實施例用於收集暫存器檔案運算元的方法步驟之流程圖。雖然該等方法步驟係配合第一到五圖之該等系統做說明，本技術專業人士將可瞭解到設置成以任何順序執行該等方法步驟的任何系統皆在本發明之範圍內。

如所示，方法 600 開始於步驟 602，其中指令分派單元 504 自包繞排程器 502 接收一組一或多個指令。每一指令可包括一或多個運算元，且被排程來在一或多個執行緒上執行。在步驟 604，指令分派單元 504 評估由該等一或多個指令存取的該組運算元，藉以決定該等運算元是否可配合到數個經辨識的成型存取樣式中之一個樣式。視一成型存取樣式被辨識的程度，在步驟 606，指令分派單元 504 形成一成型暫存器檔案存取來取得該等指令運算元。在步驟 608，指令分派單元 504 藉由傳送該等相對應暫存器位址和讀取致能至暫存器檔案 506 來執行該成型暫存器檔案存取。在步驟 610，指令分派單元 504 將該等經取得的運算元對準於該等相對應指令。在步驟 612，

指令分派單元 504 寫入該等經取得的運算元到收集器 508。

在步驟 614，指令分派單元 504 決定是否已經取得所有的運算元。如果尚未取得所有運算元，方法 600 繼續進行到步驟 614，其中指令分派單元 504 停止該管線來防止進一步的指令進入到包繞排程器和指令單元 312。然後方法 600 回到步驟 606，其中指令分派單元 504 形成另一個成型暫存器存取。方法 600 繼續進行直到指令分派單元 504 在步驟 614 決定已經取得所有運算元。然後方法 600 進行到步驟 618，其中指令分派單元 504 解除停止該管線，此時方法 600 中止。

總而言之，該揭示技術提供一種有效率的方式來由一暫存器檔案取得運算元。特定而言，指令分派單元 504 自包繞排程器 502 取得一或多個指令，其每一者包括一或多個運算元來橫跨一或多個執行緒被執行。該等運算元共同地被組織成可以形成一「成型存取」的一或多個運算元群組。指令分派單元 504 設置成可辨識由該等運算元群組所利用的不同記憶體存取樣式。指令分派單元 504 形成對應於該等記憶體存取樣式的一成型存取。對應於由該成型存取涵蓋的暫存器之該等運算元係自暫存器檔案 506 取得，並儲存在一收集器 508 中。如果在該成型存取之後所有指令運算元並未被讀取並收集在收集器 508 中，則指令分派單元 504 停止 SM 310 內該管線，並形成另一個成型存取來由暫存器檔案 506 取得該等額外的運算元，並儲存該等運算元在收集器 508 中。一旦所有運算元皆被讀取並收集在收集器 508 中，指令分派單元 504 解除停止該管線，並傳送該等指令和相對應的運算元到 SM 310 內的功能性單元來執行。

較佳地是，多個運算元在一單一暫存器存取作業中由暫存器檔案 506 取得，而不會有資源衝突。當指令運算元利用經辨識的記憶體存取樣式時，由暫存器檔案 506 取得運算元的效能可藉由形成有效率地取得呈現這些存取樣式的運算元之成型

存取所改善。另外，在該等暫存器檔案的記憶庫內的暫存器檔案 402 可被彈性地配置到邏輯記憶庫當中來取得在一組指令之內的多個運算元。因此，每一暫存器存取作業可具有一不同的成型存取，其係利用由該組運算元所擔保的一不同邏輯記憶庫配置。指令分派單元 504 使用成型存取來執行一或多個暫存器存取作業，直到所有的運算元皆被讀取且被收集為止。

前述係關於本發明之具體實施例，本發明之其它及進一步的具體實施例皆可進行，而並不背離其基本範圍，且其範圍由以下的申請專利範圍所決定。

【圖式簡單說明】

所以，可以詳細瞭解本發明上述特徵之方式當中，本發明之一更為特定的說明簡述如上，其可藉由參照具體實施例來進行，其中一些例示於所附圖式中。但是應要注意到，該等附屬圖式僅例示本發明的典型具體實施例，因此其並非要做為本發明之範圍的限制，其可允許其它同等有效的具體實施例。

第一圖例示設置成實作本發明一或多種態樣之電腦系統的方塊圖；

第二圖為根據本發明一或多種態樣中第一圖之電腦系統的一平行處理子系統之方塊圖；

第三 A 圖為根據本發明一具體實施例中第二圖的該前端的方塊圖；

第三 B 圖為根據本發明一具體實施例中第二圖之該等平行處理單元中之一者內一通用處理叢集的方塊圖；

第三 C 圖為根據本發明一具體實施例中第三 B 圖的該串流多處理器之一部份的方塊圖；

第四 A 圖例示根據本發明一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫；

第四 B 圖例示根據本發明另一具體實施例設置用於運算

元收集的一暫存器檔案之記憶庫；

第四 C 圖例示根據本發明又另一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫；

第四 D 圖例示根據本發明再又另一具體實施例設置用於運算元收集的一暫存器檔案之記憶庫；

第五圖例示根據本發明一具體實施例第三 C 圖的該包繞排程器和指令單元與該局部暫存器檔案之方塊圖；以及

第六圖係根據本發明一具體實施例用於收集暫存器檔案運算元的方法步驟之流程圖。

【主要元件符號說明】

100 電腦系統
102 中央處理單元
103 裝置驅動器
104 系統記憶體
105 記憶體橋接器
106 通訊路徑
107 輸入/輸出橋接器
108 輸入裝置
110 顯示器
112 平行處理子系統
113 通訊路徑
114 系統碟
116 交換器
118 網路轉接器
120, 121 嵌入卡
202 平行處理單元
204 平行處理記憶體
205 輸入/輸出單元
206 主控介面
207 任務/工作單元
208 通用處理叢集
210 交叉開關單元
212 前端
214 記憶體介面
215 區隔單元
220 動態隨機存取記憶體
230 處理叢集陣列
300 任務管理單元
302 執行單元
303 載入儲存單元
304 局部暫存器檔案
305 管線管理員
306 共用記憶體
310 串流多處理器
312 包繞排程器和指令

單元
315 紋路單元
320 L1 快取
321 排程器表
322 任務中介資料
325 預先掃描場化作業
328 記憶體管理單元
330 工作分配交叉開關
335 L1.5 快取
340 工作分配單元
345 任務表
352 統一位址映射單元
370 指令 L1 快取
380 記憶體和快取互連接
402 暫存器檔案
420, 440, 460, 480 邏輯記憶庫
502 包繞排程器
504 指令分派單元
506 暫存器檔案
508 收集器

~~104年3月11日修正替換頁~~

(104年3月12日修正劃線頁(本))

七、申請專利範圍：

1. 一種用於執行暫存器記憶體作業的電腦實作方法，該方法包括：

接收要橫跨複數運算元來執行的一指令；

辨識出其中儲存有該等複數運算元的複數暫存器檔案，其可經由一既定記憶體存取模式來存取；

形成對應於該既定記憶體存取樣式的一成型記憶體存取作業；以及

執行該成型記憶體存取作業，以由該等複數暫存器檔案存取該等複數運算元。

2. 一種用於執行暫存器記憶體作業的子系統，其包含：

一指令分派單元，其設置成：

接收要橫跨複數運算元來執行的一指令；

辨識出其中儲存有該等複數運算元的複數暫存器檔案，其可經由一既定記憶體存取模式來存取；

形成對應於該既定記憶體存取樣式的一成型記憶體存取作業；以及

執行該成型記憶體存取作業，以由該等複數暫存器檔案存取該等複數運算元。

3. 如申請專利範圍第 2 項之子系統，其中該指令分派單元進一步設置成根據該指令對準該等複數運算元；以及儲存該等複數運算元在一運算元收集器中。

4. 如申請專利範圍第 2 項之子系統，其中形成一成型記憶體存取作業包含形成該等複數暫存器檔案中至少一部份所屬於的一或多個邏輯記憶庫。

5. 如申請專利範圍第 2 項之子系統，其中該成型記憶體存取作業設置成存取單一寬度運算元。

6. 如申請專利範圍第 2 項之子系統，其中該成型記憶體存取作業設置成存取雙重寬度運算元。

7. 如申請專利範圍第 2 項之子系統，其中該成型記憶體存取

作業設置成存取四重寬度運算元。

8. 如申請專利範圍第 2 項之子系統，其中該成型記憶體存取作業設置成存取具有不同運算元寬度的運算元。

9. 如申請專利範圍第 2 項之子系統，其中該指令分派單元進一步設置成傳送該指令和至少該等複數運算元至一功能性單元來執行。

10. 一種運算裝置，其包含：

一子系統，其包括一指令分派單元設置成：

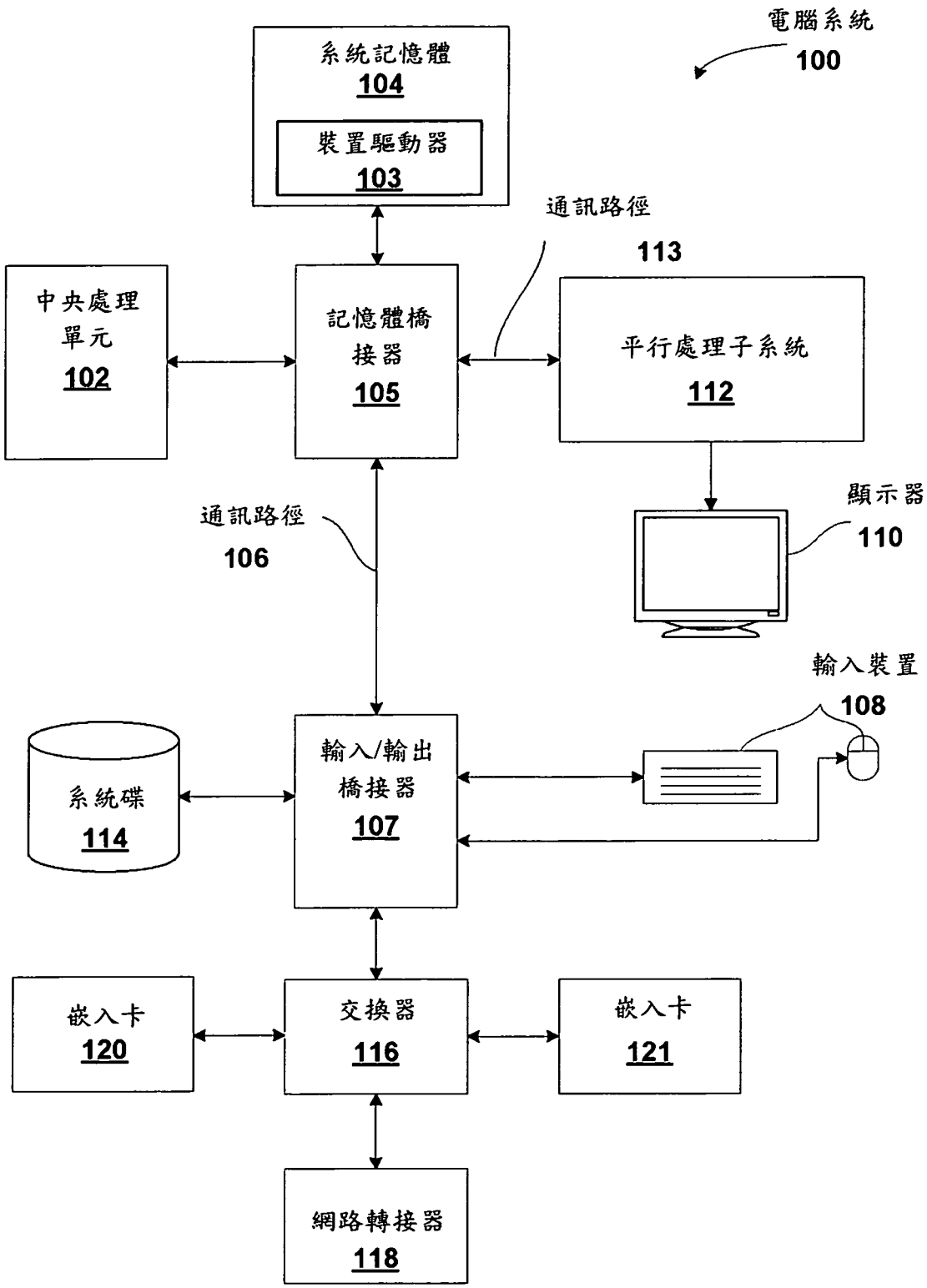
接收要橫跨複數運算元來執行的一指令；

辨識出其中儲存有該等複數運算元的複數暫存器檔案，其可經由一既定記憶體存取模式來存取；

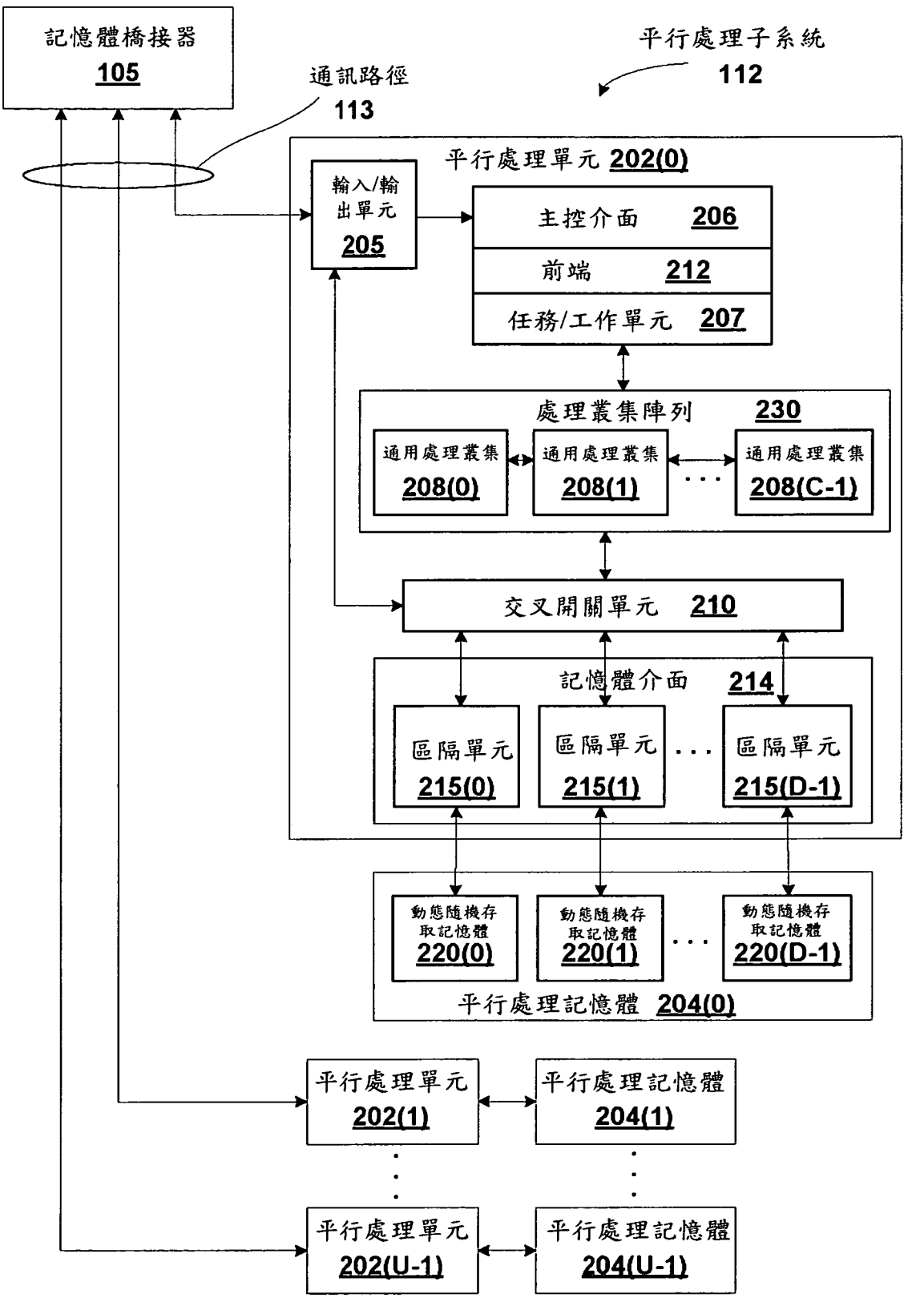
形成對應於該既定記憶體存取樣式的一成型記憶體存取作業；以及

執行該成型記憶體存取作業，以由該等複數暫存器檔案存取該等複數運算元。

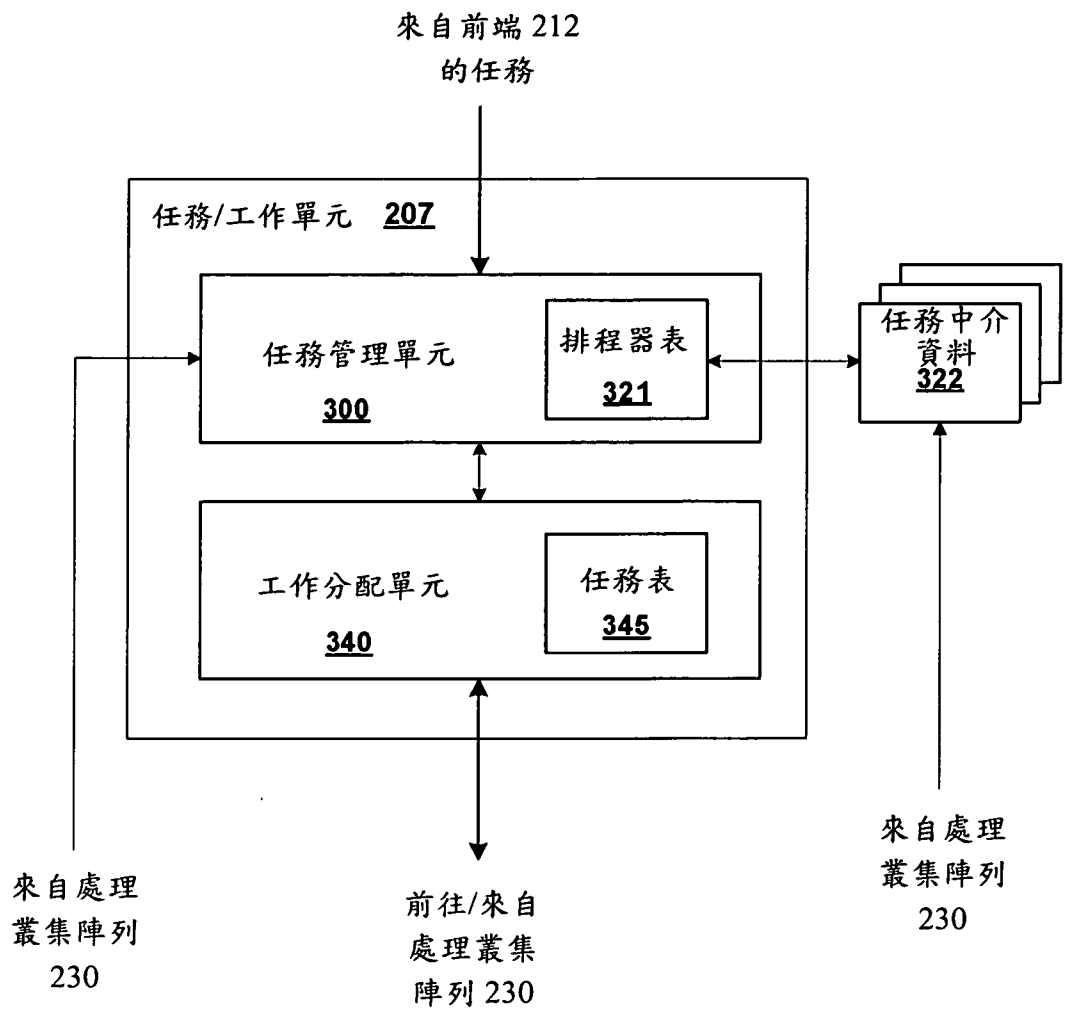
八、圖式：


電腦系統
100
系統記憶體
104
裝置驅動器
103
通訊路徑
113
中央處理單元
102
記憶體橋接器
105
平行處理子系統
112
顯示器
110
通訊路徑
106
輸入裝置
108
輸入/輸出橋接器
107
系統碟
114
嵌入卡
120
交換器
116
嵌入卡
121
網路轉接器
118

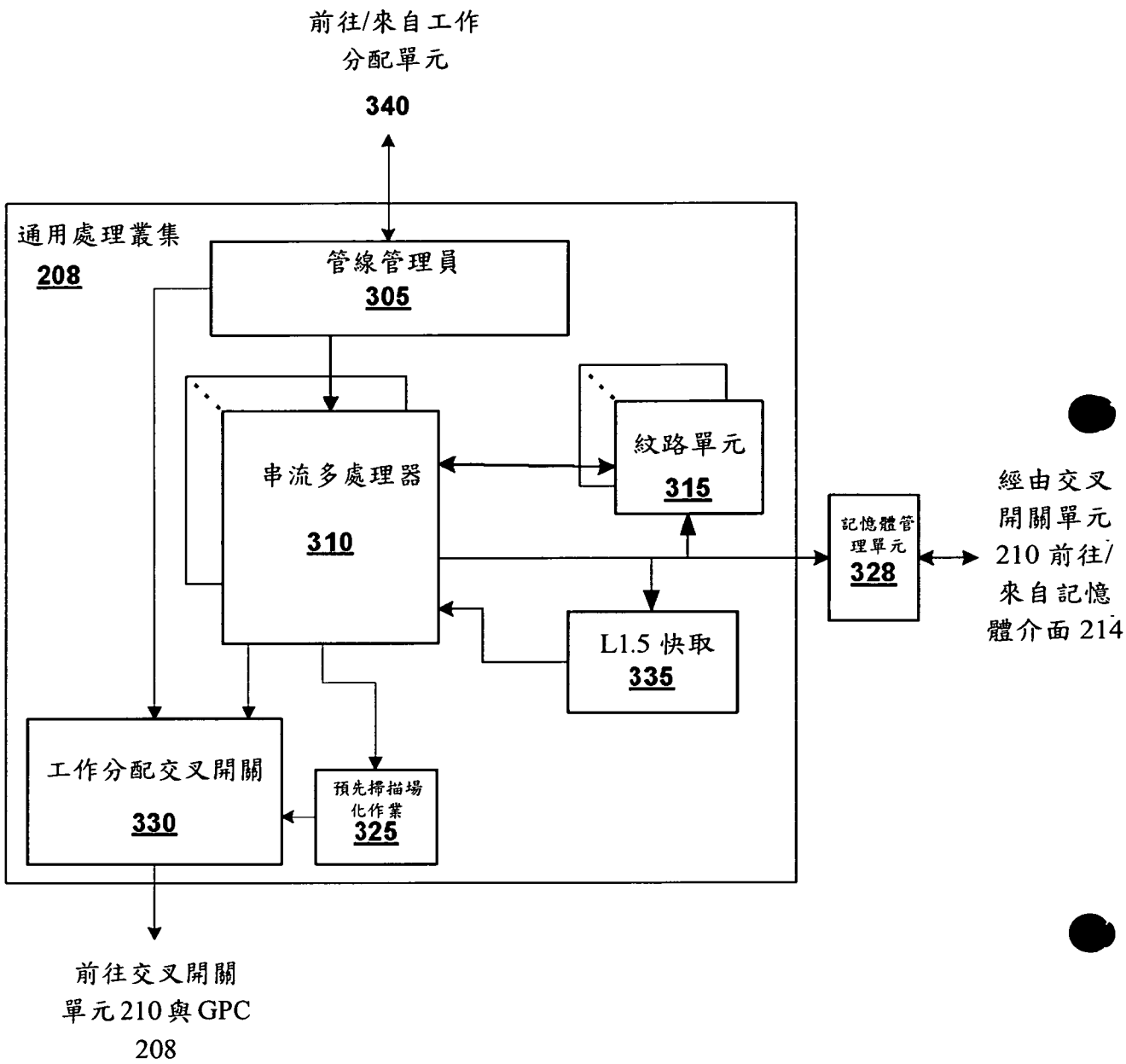
第一圖


記憶體橋接器
105
通訊路徑
113
平行處理子系統
112
平行處理單元 202(0)
輸入/輸出單元
205
主控介面 206
前端 212
任務/工作單元 207
處理叢集陣列 230
通用處理叢集
208(0)
通用處理叢集
208(1)
通用處理叢集
208(C-1)
交叉開關單元 210
記憶體介面 214
區隔單元
215(0)
區隔單元
215(1)
區隔單元
215(D-1)
動態隨機存取記憶體
220(0)
動態隨機存取記憶體
220(1)
動態隨機存取記憶體
220(D-1)
平行處理記憶體 204(0)
平行處理單元
202(1)
平行處理記憶體
204(1)
平行處理單元
202(U-1)
平行處理記憶體
204(U-1)

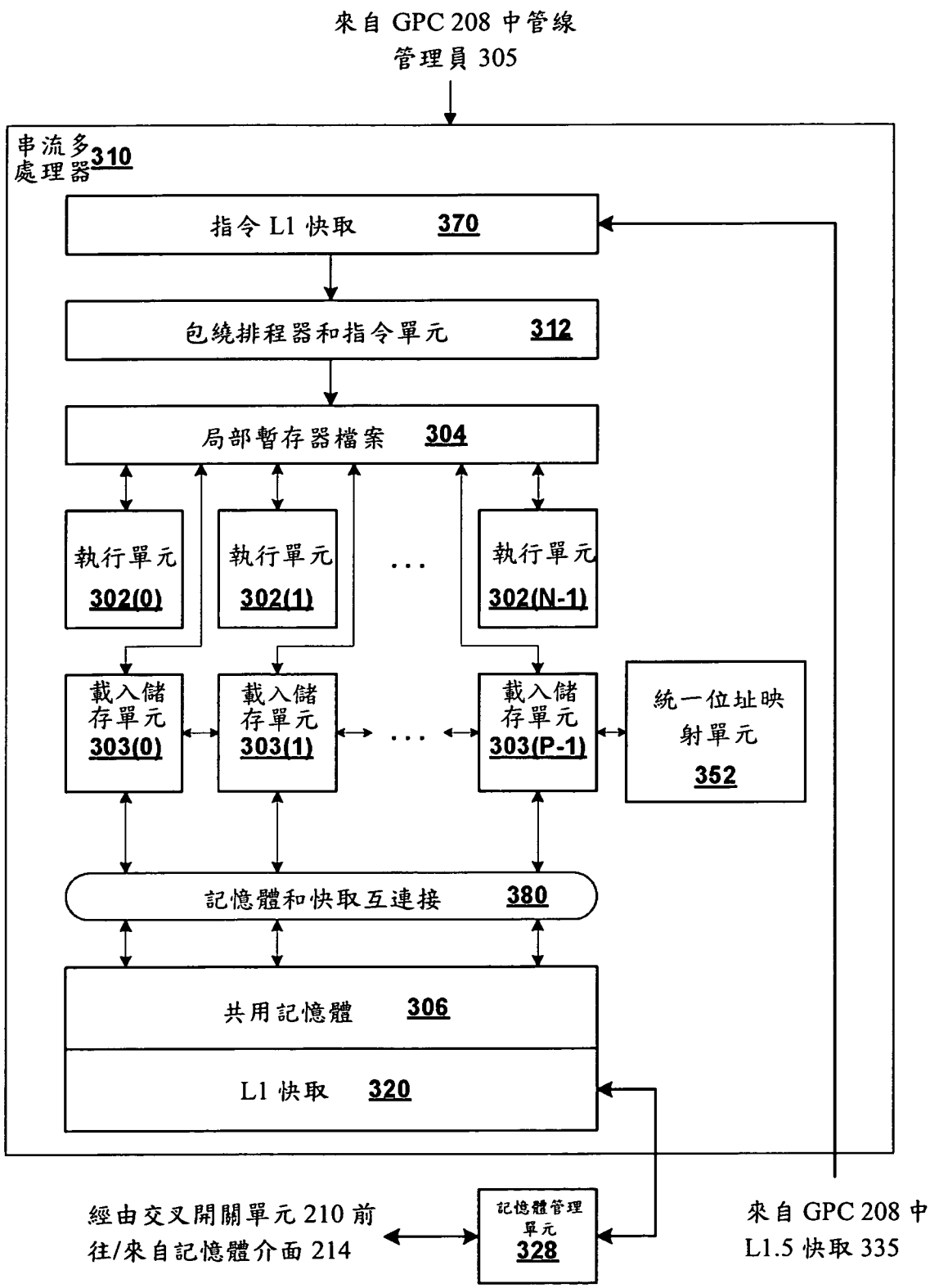
第二圖


來自前端 212
的任務
任務/工作單元 207
任務管理單元
300
排程器表
321
任務中介
資料
322
工作分配單元
340
任務表
345
來自處理
叢集陣列
230
前往/來自
處理叢集
陣列 230
來自處理
叢集陣列
230

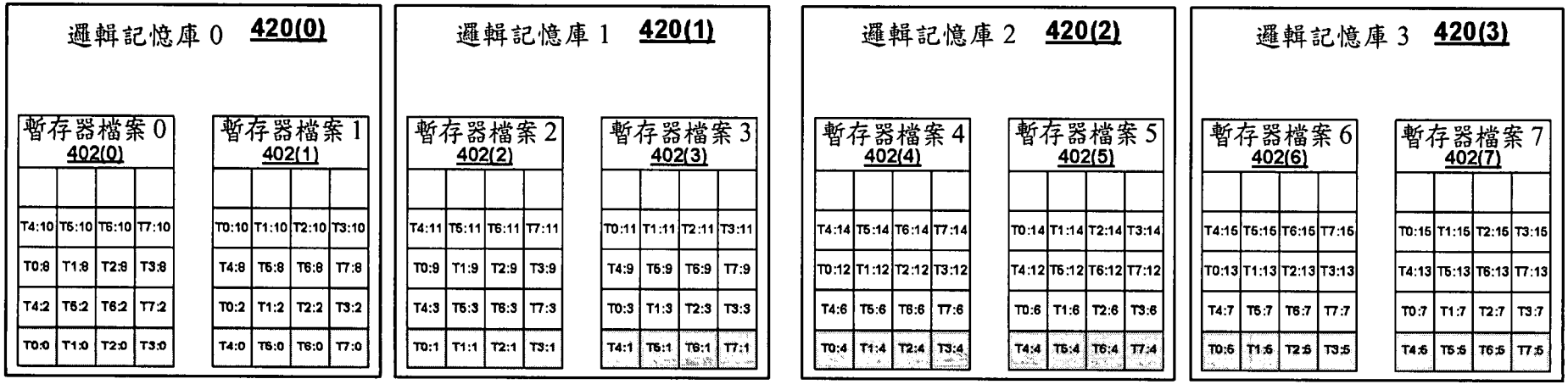
第三 A 圖


前往/來自工作
分配單元
340
通用處理叢集
208
管線管理員
305
串流多處理器
310
紋路單元
315
記憶體管
理單元
328
經由交叉
開關單元
210 前往/
來自記憶
體介面 214
L1.5 快取
335
工作分配交叉開關
330
預先掃描場
化作業
325
前往交叉開關
單元 210 與 GPC
208

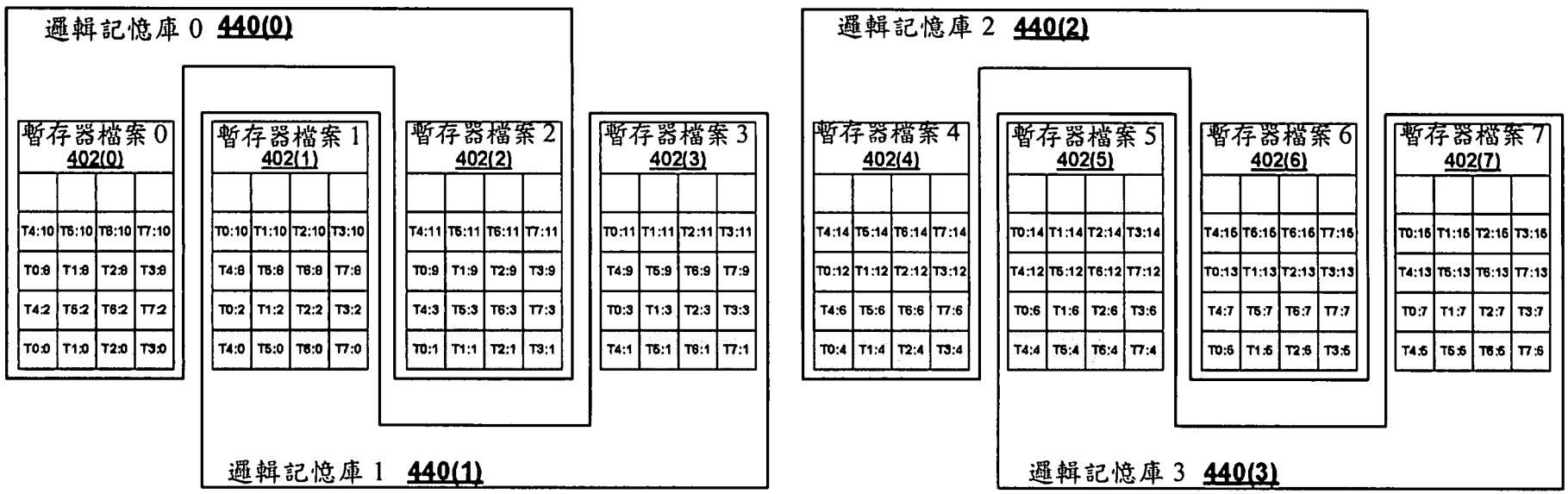
第三 B 圖


來自 GPC 208 中管線
管理員 305
串流多處理器 310
指令 L1 快取 370
包繞排程器和指令單元 312
局部暫存器檔案 304
執行單元 302(0)
執行單元 302(1)
...
執行單元 302(N-1)
載入儲存單元 303(0)
載入儲存單元 303(1)
...
載入儲存單元 303(P-1)
統一位址映射單元 352
記憶體和快取互連接 380
共用記憶體 306
L1 快取 320
經由交叉開關單元 210 前往/來自記憶體介面 214
記憶體管理單元 328
來自 GPC 208 中 L1.5 快取 335

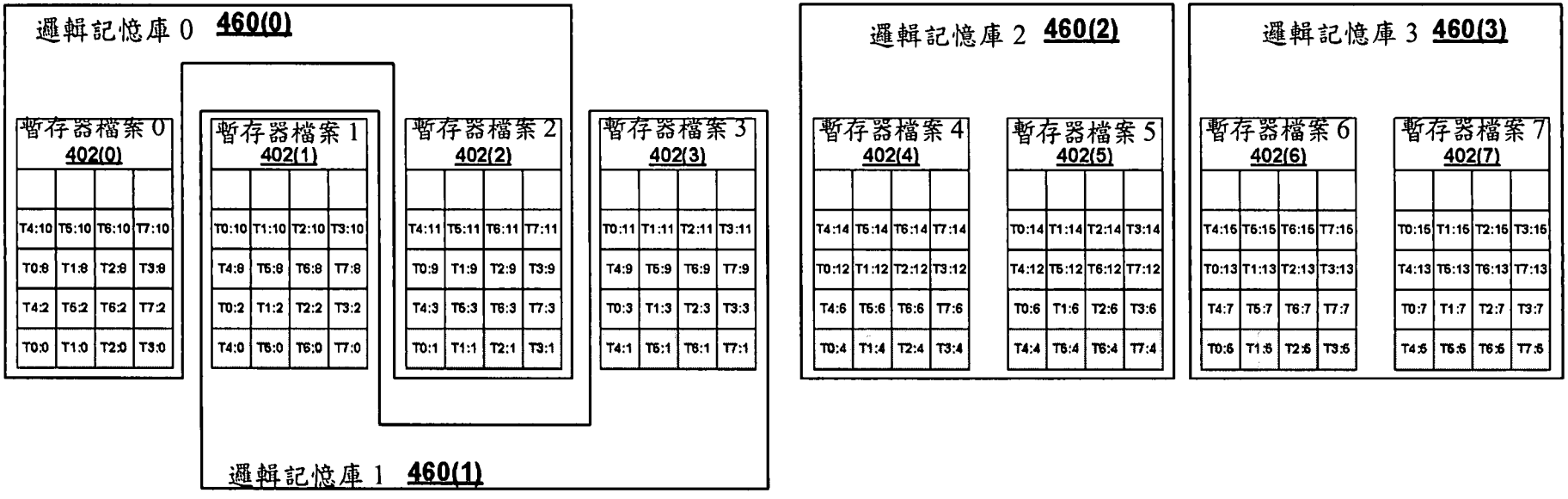
第三C圖


邏輯記憶庫 0 420(0)
暫存器檔案 0 402(0)
T4:10 T5:10 T6:10 T7:10
T0:8 T1:8 T2:8 T3:8
T4:2 T5:2 T6:2 T7:2
T0:0 T1:0 T2:0 T3:0
暫存器檔案 1 402(1)
T0:10 T1:10 T2:10 T3:10
T4:8 T5:8 T6:8 T7:8
T0:2 T1:2 T2:2 T3:2
T4:0 T5:0 T6:0 T7:0
邏輯記憶庫 1 420(1)
暫存器檔案 2 402(2)
T4:11 T5:11 T6:11 T7:11
T0:9 T1:9 T2:9 T3:9
T4:3 T5:3 T6:3 T7:3
T0:1 T1:1 T2:1 T3:1
暫存器檔案 3 402(3)
T0:11 T1:11 T2:11 T3:11
T4:9 T5:9 T6:9 T7:9
T0:3 T1:3 T2:3 T3:3
T4:1 T5:1 T6:1 T7:1
邏輯記憶庫 2 420(2)
暫存器檔案 4 402(4)
T4:14 T5:14 T6:14 T7:14
T0:12 T1:12 T2:12 T3:12
T4:6 T5:6 T6:6 T7:6
T0:4 T1:4 T2:4 T3:4
暫存器檔案 5 402(5)
T0:14 T1:14 T2:14 T3:14
T4:12 T5:12 T6:12 T7:12
T0:6 T1:6 T2:6 T3:6
T4:4 T5:4 T6:4 T7:4
邏輯記憶庫 3 420(3)
暫存器檔案 6 402(6)
T4:15 T5:15 T6:15 T7:15
T0:13 T1:13 T2:13 T3:13
T4:7 T5:7 T6:7 T7:7
T0:5 T1:5 T2:5 T3:5
暫存器檔案 7 402(7)
T0:15 T1:15 T2:15 T3:15
T4:13 T5:13 T6:13 T7:13
T0:7 T1:7 T2:7 T3:7
T4:5 T5:5 T6:5 T7:5

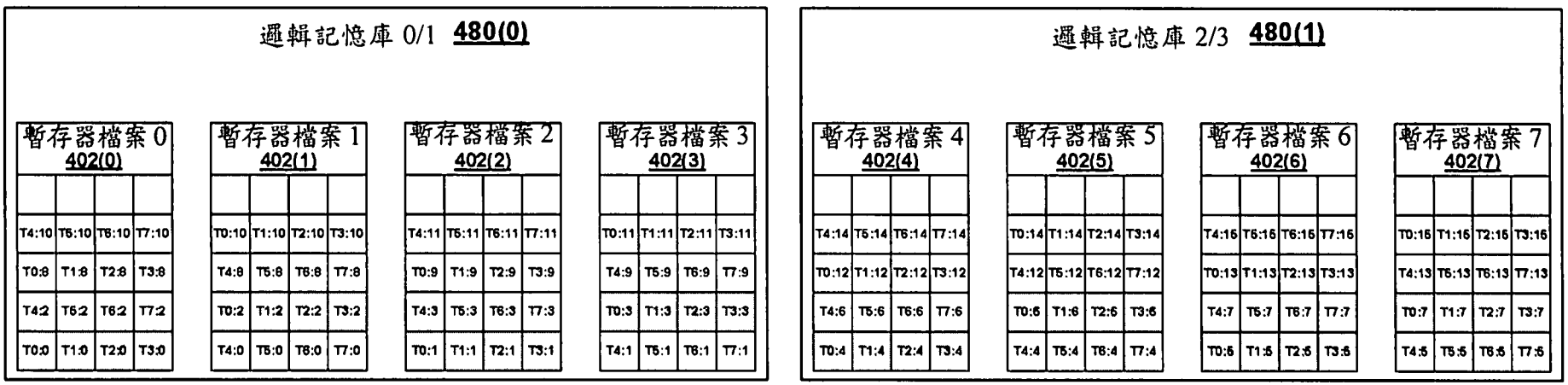
第四A圖


邏輯記憶庫 0 440(0)
暫存器檔案 0 402(0)
T4:10 T5:10 T6:10 T7:10
T0:8 T1:8 T2:8 T3:8
T4:2 T5:2 T6:2 T7:2
T0:0 T1:0 T2:0 T3:0
暫存器檔案 1 402(1)
T0:10 T1:10 T2:10 T3:10
T4:8 T5:8 T6:8 T7:8
T0:2 T1:2 T2:2 T3:2
T4:0 T5:0 T6:0 T7:0
暫存器檔案 2 402(2)
T4:11 T5:11 T6:11 T7:11
T0:9 T1:9 T2:9 T3:9
T4:3 T5:3 T6:3 T7:3
T0:1 T1:1 T2:1 T3:1
暫存器檔案 3 402(3)
T0:11 T1:11 T2:11 T3:11
T4:9 T5:9 T6:9 T7:9
T0:3 T1:3 T2:3 T3:3
T4:1 T5:1 T6:1 T7:1
邏輯記憶庫 1 440(1)
邏輯記憶庫 2 440(2)
暫存器檔案 4 402(4)
T4:14 T5:14 T6:14 T7:14
T0:12 T1:12 T2:12 T3:12
T4:6 T5:6 T6:6 T7:6
T0:4 T1:4 T2:4 T3:4
暫存器檔案 5 402(5)
T0:14 T1:14 T2:14 T3:14
T4:12 T5:12 T6:12 T7:12
T0:6 T1:6 T2:6 T3:6
T4:4 T5:4 T6:4 T7:4
暫存器檔案 6 402(6)
T4:15 T5:15 T6:15 T7:15
T0:13 T1:13 T2:13 T3:13
T4:7 T5:7 T6:7 T7:7
T0:5 T1:5 T2:5 T3:5
暫存器檔案 7 402(7)
T0:15 T1:15 T2:15 T3:15
T4:13 T5:13 T6:13 T7:13
T0:7 T1:7 T2:7 T3:7
T4:5 T5:5 T6:5 T7:5
邏輯記憶庫 3 440(3)

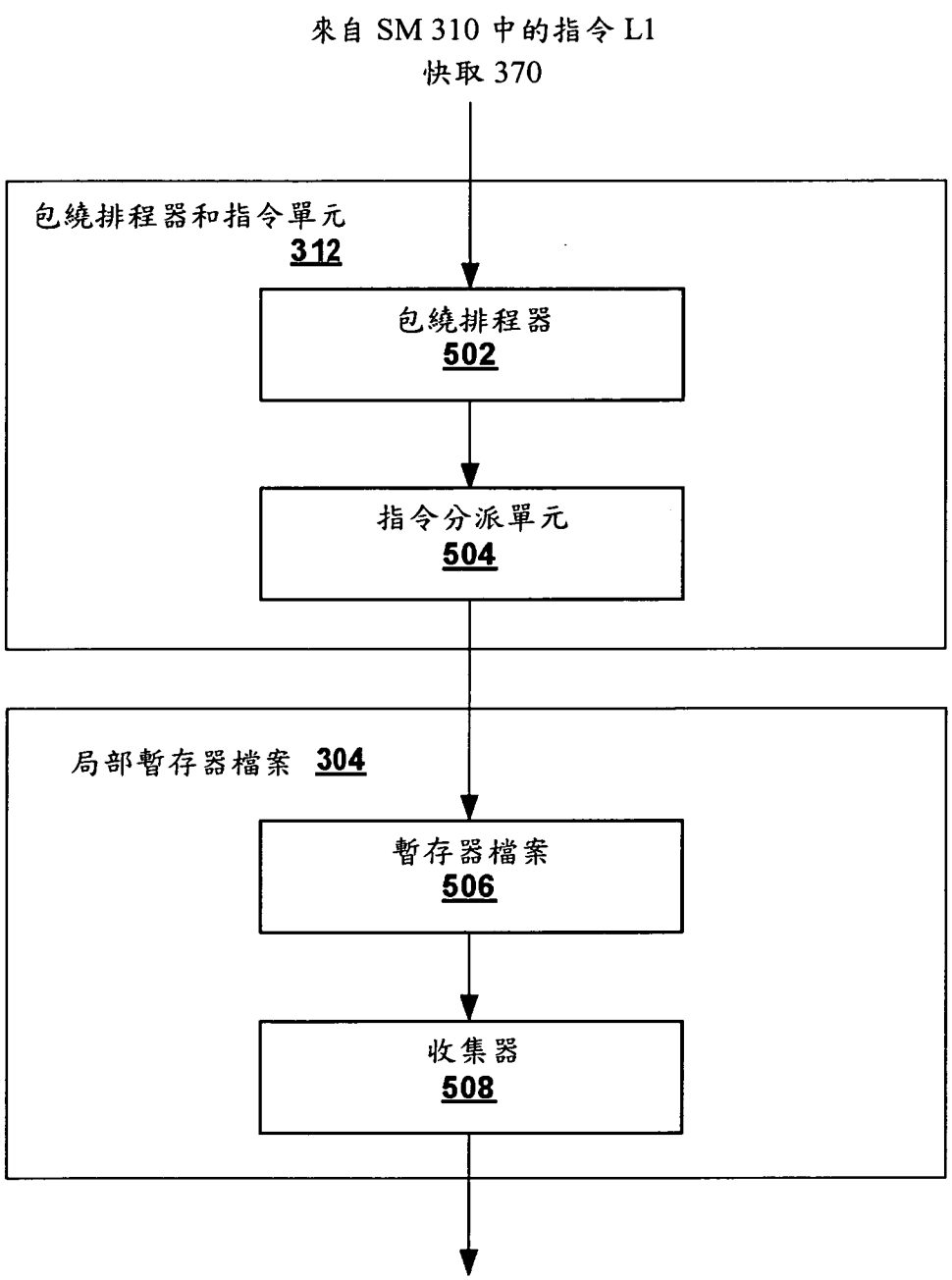
第四 B 圖


邏輯記憶庫 0 460(0)
暫存器檔案 0 402(0)
T4:10 T5:10 T6:10 T7:10
T0:8 T1:8 T2:8 T3:8
T4:2 T5:2 T6:2 T7:2
T0:0 T1:0 T2:0 T3:0
暫存器檔案 1 402(1)
T0:10 T1:10 T2:10 T3:10
T4:8 T5:8 T6:8 T7:8
T0:2 T1:2 T2:2 T3:2
T4:0 T5:0 T6:0 T7:0
暫存器檔案 2 402(2)
T4:11 T5:11 T6:11 T7:11
T0:9 T1:9 T2:9 T3:9
T4:3 T5:3 T6:3 T7:3
T0:1 T1:1 T2:1 T3:1
暫存器檔案 3 402(3)
T0:11 T1:11 T2:11 T3:11
T4:9 T5:9 T6:9 T7:9
T0:3 T1:3 T2:3 T3:3
T4:1 T5:1 T6:1 T7:1
邏輯記憶庫 1 460(1)
邏輯記憶庫 2 460(2)
暫存器檔案 4 402(4)
T4:14 T5:14 T6:14 T7:14
T0:12 T1:12 T2:12 T3:12
T4:6 T5:6 T6:6 T7:6
T0:4 T1:4 T2:4 T3:4
暫存器檔案 5 402(5)
T0:14 T1:14 T2:14 T3:14
T4:12 T5:12 T6:12 T7:12
T0:6 T1:6 T2:6 T3:6
T4:4 T5:4 T6:4 T7:4
邏輯記憶庫 3 460(3)
暫存器檔案 6 402(6)
T4:15 T5:15 T6:15 T7:15
T0:13 T1:13 T2:13 T3:13
T4:7 T5:7 T6:7 T7:7
T0:5 T1:5 T2:5 T3:5
暫存器檔案 7 402(7)
T0:15 T1:15 T2:15 T3:15
T4:13 T5:13 T6:13 T7:13
T0:7 T1:7 T2:7 T3:7
T4:5 T5:5 T6:5 T7:5

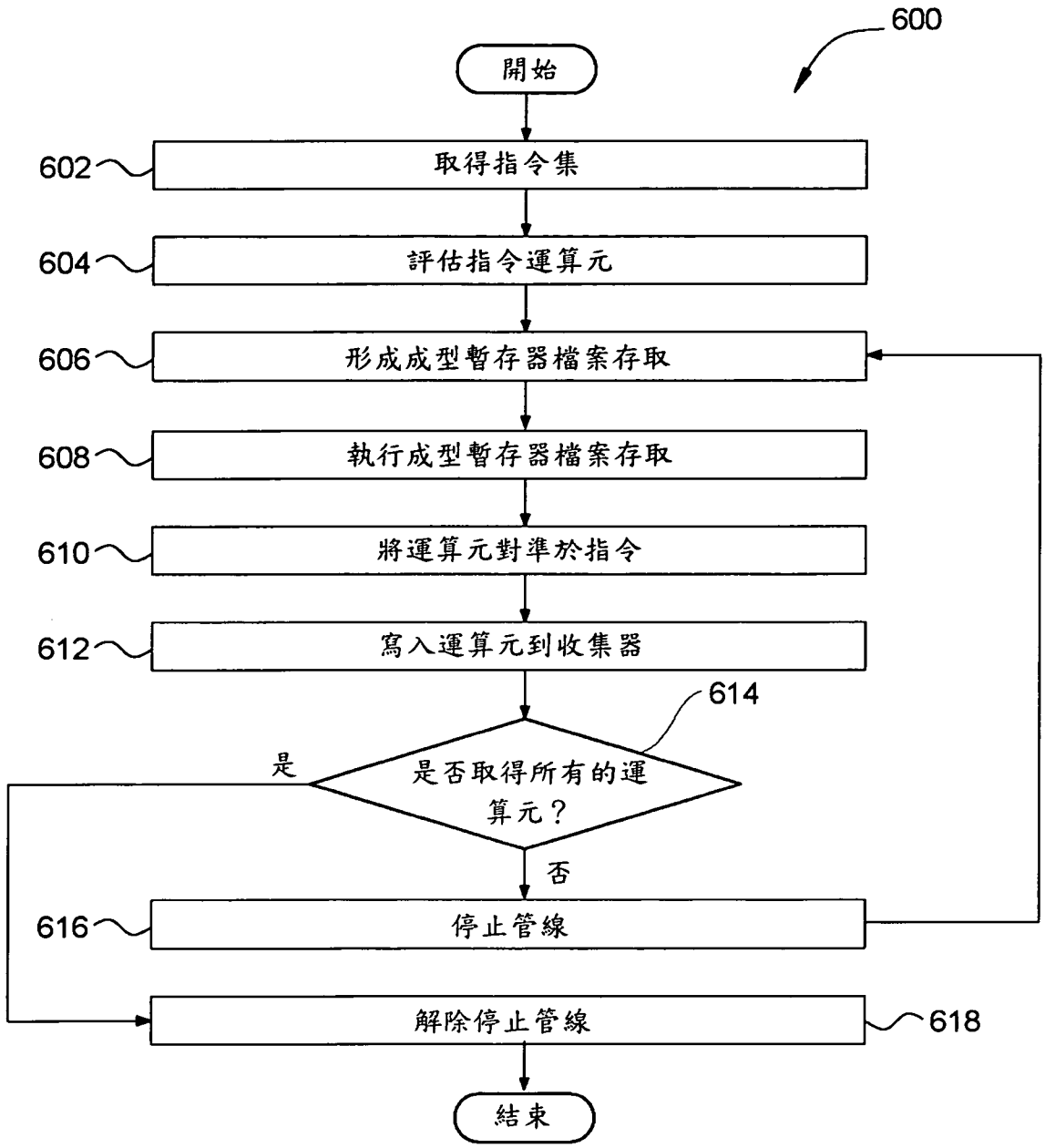
第四C圖


邏輯記憶庫 0/1 480(0)
暫存器檔案 0 402(0)
T4:10 T5:10 T6:10 T7:10
T0:8 T1:8 T2:8 T3:8
T4:2 T5:2 T6:2 T7:2
T0:0 T1:0 T2:0 T3:0
暫存器檔案 1 402(1)
T0:10 T1:10 T2:10 T3:10
T4:8 T5:8 T6:8 T7:8
T0:2 T1:2 T2:2 T3:2
T4:0 T5:0 T6:0 T7:0
暫存器檔案 2 402(2)
T4:11 T5:11 T6:11 T7:11
T0:9 T1:9 T2:9 T3:9
T4:3 T5:3 T6:3 T7:3
T0:1 T1:1 T2:1 T3:1
暫存器檔案 3 402(3)
T0:11 T1:11 T2:11 T3:11
T4:9 T5:9 T6:9 T7:9
T0:3 T1:3 T2:3 T3:3
T4:1 T5:1 T6:1 T7:1
邏輯記憶庫 2/3 480(1)
暫存器檔案 4 402(4)
T4:14 T5:14 T6:14 T7:14
T0:12 T1:12 T2:12 T3:12
T4:6 T5:6 T6:6 T7:6
T0:4 T1:4 T2:4 T3:4
暫存器檔案 5 402(5)
T0:14 T1:14 T2:14 T3:14
T4:12 T5:12 T6:12 T7:12
T0:6 T1:6 T2:6 T3:6
T4:4 T5:4 T6:4 T7:4
暫存器檔案 6 402(6)
T4:15 T5:15 T6:15 T7:15
T0:13 T1:13 T2:13 T3:13
T4:7 T5:7 T6:7 T7:7
T0:5 T1:5 T2:5 T3:5
暫存器檔案 7 402(7)
T0:15 T1:15 T2:15 T3:15
T4:13 T5:13 T6:13 T7:13
T0:7 T1:7 T2:7 T3:7
T4:5 T5:5 T6:5 T7:5

第四D圖


來自 SM 310 中的指令 L1
快取 370
包繞排程器和指令單元
312
包繞排程器
502
指令分派單元
504
局部暫存器檔案 304
暫存器檔案
506
收集器
508
前往執行單元 302 和載入
儲存單元 303

第五圖


600
開始
602
取得指令集
604
評估指令運算元
606
形成成型暫存器檔案存取
608
執行成型暫存器檔案存取
610
將運算元對準於指令
612
寫入運算元到收集器
614
是否取得所有的運算元？
是
否
616
停止管線
618
解除停止管線
結束

第六圖