



(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 I564796 B

(45)公告日：中華民國 106 (2017) 年 01 月 01 日

(21)申請案號：104138536

(22)申請日：中華民國 104 (2015) 年 11 月 20 日

(51)Int. Cl. : G06F9/30 (2006.01)

G06F11/22 (2006.01)

(30)優先權：2014/12/23 美國

14/581,815

(71)申請人：英特爾股份有限公司 (美國) INTEL CORPORATION (US)

美國

(72)發明人：柯柏 吉瑟斯 CORBAL, JESUS (ES)；哈內克 羅伯特 HANEK, ROBERT N. (US)；弗格森 華倫 FERGUSON, WARREN E. (US)；巴瑞米 塔蘭 BAHRAMI, TARANEH (US)；泰菲特 艾維 TEVET, AVI A. (US)；布瑞福 丹尼斯 BRADFORD, DENNIS R. (US)；菲瑞 麥可 FERRY, MICHAEL (US)；張經維 ZHANG, JINGWEI (CN)

(74)代理人：林志剛

(56)參考文獻：

TW 201401045A

TW 201403314A

US 6477639B1

WO 2014/128153A1

審查人員：林民安

申請專利範圍項數：25 項 圖式數：16 共 89 頁

(54)名稱

執行用以最佳化指令流的檢查之設備及方法

APPARATUS AND METHOD FOR PERFORMING A CHECK TO OPTIMIZE INSTRUCTION FLOW

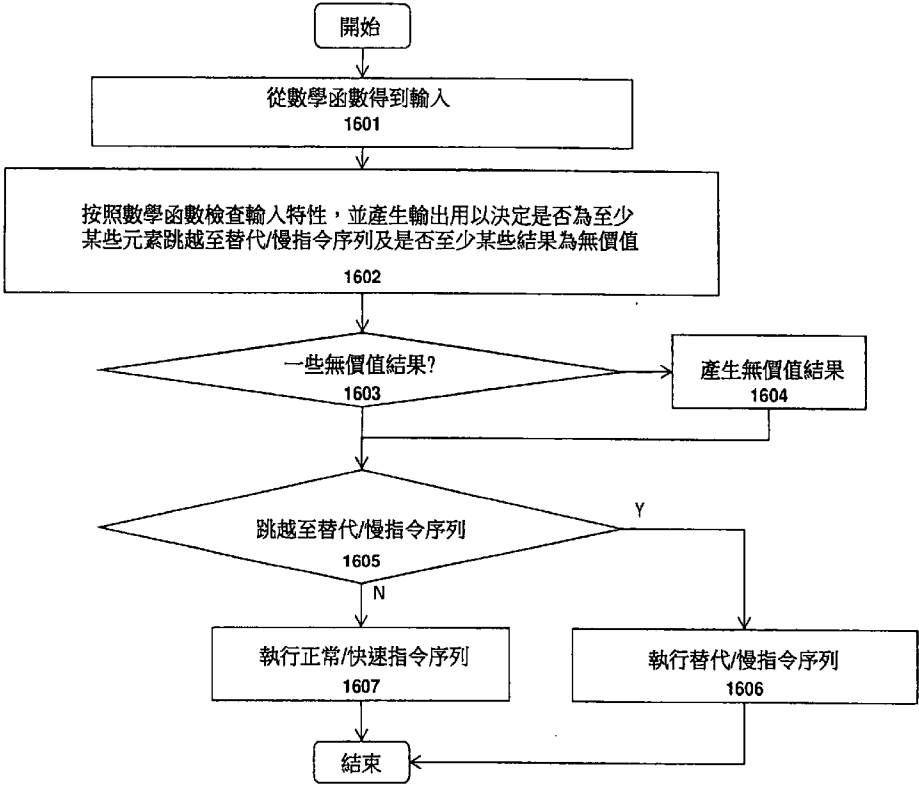
(57)摘要

一種設備及方法，用於對數學指令之輸入實施檢查並選擇有效率地管理處理器之架構狀態的預設序列。例如，處理器的一實施例包含：算術邏輯單元(arithmetic logic unit；ALU)，用以使用一或多個源運算元來執行複數個數學運算；指令檢查邏輯，用以評估用於當下數學指令的源運算元，並依據該評估來決定是執行包括藉由 ALU 來執行當下數學指令之運算的預設序列，或是跳越到運算的替代序列，運算的該替代序列係適合用以提供結果用於具有比運算之該預設序列更有效率的特定類型之源運算元之數學指令。

An apparatus and method for performing a check on inputs to a mathematical instruction and selecting a default sequence efficiently managing the architectural state of a processor. For example, one embodiment of a processor comprises: an arithmetic logic unit (ALU) to perform a plurality of mathematical operations using one or more source operands; instruction check logic to evaluate the source operands for a current mathematical instruction and to determine, based on the evaluation, whether to execute a default sequence of operations including executing the current mathematical instruction by the ALU or to jump to an alternate sequence of operations adapted to provide a result for the mathematical instruction having particular types of source operands more efficiently than the default sequence of operations.

指定代表圖：

圖 16



發明摘要

※申請案號：104138536

※申請日：104 年 11 月 20 日

※IPC 分類：G06F 9/30 (2006.01)

【發明名稱】(中文/英文)

G06F 1/22. (2006.01)

執行用以最佳化指令流的檢查之設備及方法

Apparatus and method for performing a check to optimize instruction
flow

【中文】

一種設備及方法，用於對數學指令之輸入實施檢查並選擇有效率地管理處理器之架構狀態的預設序列。例如，處理器的一實施例包含：算術邏輯單元(arithmetic logic unit; ALU)，用以使用一或多個源運算元來執行複數個數學運算；指令檢查邏輯，用以評估用於當下數學指令的源運算元，並依據該評估來決定是執行包括藉由 ALU 來執行當下數學指令之運算的預設序列，或是跳越到運算的替代序列，運算的該替代序列係適合用以提供結果用於具有比運算之該預設序列更有效率的特定類型之源運算元之數學指令。

【 英文 】

An apparatus and method for performing a check on inputs to a mathematical instruction and selecting a default sequence efficiently managing the architectural state of a processor. For example, one embodiment of a processor comprises: an arithmetic logic unit (ALU) to perform a plurality of mathematical operations using one or more source operands; instruction check logic to evaluate the source operands for a current mathematical instruction and to determine, based on the evaluation, whether to execute a default sequence of operations including executing the current mathematical instruction by the ALU or to jump to an alternate sequence of operations adapted to provide a result for the mathematical instruction having particular types of source operands more efficiently than the default sequence of operations.

【代表圖】

【本案指定代表圖】：第(16)圖。

【本代表圖之符號簡單說明】：無

【本案若有化學式時，請揭示最能顯示發明特徵的化學式】：無

發明專利說明書

(本說明書格式、順序，請勿任意更動)

【發明名稱】(中文/英文)

執行用以最佳化指令流的檢查之設備及方法

Apparatus and method for performing a check to optimize instruction flow

【技術領域】

通言之，本發明與電腦處理器的領域有關。特別是，本發明與用於執行用以最佳化指令流之檢查的設備及方法有關。

【先前技術】

指令集或指令集架構(ISA)係電腦架構中與編程有關的部分，包括原生的資料類型、指令、暫存器架構、定址模式、記憶體架構、中斷與異常處置、以及外部輸入與輸出(I/O)。須注意，用語“指令”在本文中通常意指巨集指令-其為提供給處理器供執行的指令-與微指令或微操作相對-其為處理器的解碼器解碼巨集指令的結果。微指令或微操作可被組構，用以指令處理器上的執行單元實施操作，以實施與巨集指令相關聯的邏輯。

ISA 與微架構有所區別，微架構是用來實施指令集的一組處理器設計技術。具有不同微架構的處理器可共用共同的指令集。例如，Intel® Pentium 4 處理器、Intel®

Core™處理器、及來自加州森尼韋爾之 Advanced Micro Devices, Inc.的處理器都實施幾近相同版本的 x86 指令集(及某些延伸，較新的版本已將其加入)，但具有不同的內部設計。例如，ISA 之相同的暫存器架構可按不同的方式在不同的微架構中使用習知的技術來實施，包括專用的實體暫存器、使用暫存器更名機制(例如，使用暫存器別名表(RAT)、重排序緩衝器(ROB)與止用暫存器檔案)的一或多個動態配置的實體暫存器。除非另有所指，本文所使用的暫存器架構、暫存器檔案、及暫存器等片語意指其對軟體/程式員來說是可見的，及其中指令指定暫存器的方法。在需要區別之處，形容詞“邏輯的”、“架構的”、或“軟體可見的”係用來指示暫存器架構中的暫存器/檔案，而使用不同的形容詞來命名給定之微架構中的暫存器(例如，實體暫存器、重排序緩衝器、止用暫存器、暫存器池)。

指令集包括一或多個指令格式。給定的指令格式定義各種不同的欄位(位元數、位元的位置)，除了別的，其用來指定所要實施的操作，及該操作所要實施的運算元。某些指令格式可經由指令範本(或子格式)的定義被進一步拆解。例如，給定指令格式的指令範本可被定義成具有指令格式之欄位的不同子集(典型上，包括的欄位係按相同的次序，但由於所包括的欄位較少，因此至少某些具有不同的位元位置)，及/或定義成具有不同之解譯的給定欄位。給定的指令使用給定的指令格式來表示(及，如果定義，

在該指令格式之其中一個給定的指令範本中)，並指定操作與運算元。指令流係指令之特定的序列，序列中的每一個指令係以指令格式的指令出現(及，如果定義，該指令格式之其中一個給定的指令範本)。

【圖式簡單說明】

從以上配合以下圖式的詳細描述可更加瞭解本發明，其中：

圖 1A 與 1B 的方塊圖係說明按照本發明實施例之通用向量親和指令格式與其指令範本；

圖 2A-D 的方塊圖係說明按照本發明實施例之例示性的特定向量親和指令格式；

圖 3 的方塊圖係按照本發明一實施例之暫存器架構；以及

圖 4A 的方塊圖係說明按照本發明實施例之例示性循序提取、解碼、止用管線與例示性暫存器更名、亂序發出/執行管線；

圖 4B 的方塊圖係說明按照本發明實施例要被包括在處理器內之循序提取、解碼、止用核心的例示性實施例與例示性暫存器更名、亂序發出/執行架構核心；

圖 5A 係單處理器核心連同其連接到晶粒上之互連網路的方塊圖；

圖 5B 說明按照本發明之實施例，圖 5A 中之處理器核心部分的展開圖；

圖 6 係按照本發明實施例之具有整合式記憶體控制器與圖形之單核心處理器與多核心處理器的方塊圖；

圖 7 說明按照本發明一實施例之系統的方塊圖；

圖 8 說明按照本發明實施例之第二系統的方塊圖；

圖 9 說明按照本發明實施例之第三系統的方塊圖；

圖 10 說明按照本發明實施例之系統上單晶片(SoC)的方塊圖；

圖 11 說明按照本發明實施例，使用軟體指令轉換器將源指令集中的二進制指令轉換成目標指令集中之二進制指令之對比的方塊圖；

圖 12 說明可實施本發明之實施例的例示性處理器；

圖 13 說明用於檢查指令的輸入，並回應地產生一組可用來決定下一指令序列之輸出的指令檢查邏輯；

圖 14 說明用於除法檢查指令之不同類型之輸入值所產生的輸出；

圖 15 說明用於平方根檢查指令之不同類型之輸入值所產生的輸出；

圖 16 說明用於進行檢查操作並選擇第一指令序列或第二指令序列之方法的一實施例。

【發明內容及實施方式】

在以下的描述中，基於解釋之目的，將說明諸多的特定細節，以便提供對下述本發明之實施例的徹底瞭解。不過，熟悉此方面技藝者將明瞭，實施本發明之實施例不用

其中一些的特​​定細節。在其它例中，習知的結構與裝置係以方塊圖之形式來顯示，以避免模糊了本發明之實施例的基本原理。

例示性處理器架構與資料類型

指令集包括一或多個指令格式。給定的指令格式定義各種欄位(位元的數量、位元的位置)，除了別的，其用來指定所要實施的操作(運算碼)及該操作所要實施的運算元。某些指令格式經由指令範本的定義(或子格式)被進一步拆解。例如，給定指令格式的指令範本可被定義成具有指令格式之欄位的不同子集(典型上，所包括的欄位按相同之順序，但由於所包括的欄位較少因而至少某些具有不同的位元位置)，及/或定義成具有不同之解譯的給定欄位。因此，ISA 的每一個指令係使用一給定的指令格式來表示(且如果有定義，係在該指令格式的其中一個給定指令範本中)，且包括用於指定操作及運算元的欄位。例如，例示性的 ADD 指令具有特定的運算碼與指令格式，該指令格式包括用來指定該運算碼的運算碼欄位及用來選擇運算元(源 1/目的與源 2)的運算元欄位；且出現在指令流中的此 ADD 指令，在用以選擇特定運算元的運算元欄位中將具有特定的內容。一組 SIMD 延伸被稱為進階向量延伸(Advanced Vector Extensions; AVX)(AVX1 與 AVX2)且使用向量延伸(Vector Extensions; VEX)編碼方案已是、已發行及/或已出版(例如，見 Intel® 64 and IA-32

Architectures Software Developers Manual, 2011 年 10 月；及見 Intel® Advanced Vector Extensions Programming Reference, 2011 年 6 月)。

例示性指令格式

本文所描述之指令的實施例可具體化成不同的格式。此外，以下詳述例示性的系統、架構、及管線。指令的實施例可在這些系統、架構、及管線上執行，但不限於這些細節。

A. 通用向量親和指令格式

向量親和指令格式係適用於向量指令的指令格式(例如，有某些欄位針對向量運算)。在所描述的實施例中，雖然向量與純量運算兩者皆可經由向量親和指令格式來支援，但另些實施例使用僅只有向量運算的向量親和指令格式。

圖 1A-1B 係說明按照本發明之實施例之通用向量親和指令格式及其指令範本的方塊圖。圖 1A 係說明按照本發明之實施例之通用向量親和指令格式與其類型 A 指令範本的方塊圖；圖 1B 係說明按照本發明之實施例之通用向量親和指令格式與其類型 B 指令範本的方塊圖。精確地說，為通用向量親和指令格式 100 定義了類型 A 與類型 B 指令範本，兩者都包括無記憶體存取 105 指令範本與記憶體存取 120 指令範本。在向量親和指令格式的情況中，通用

一詞意指該指令格式並不限定於任何特定的指令集。

雖然將要描述的本發明實施例中，向量親和指令格式支援如下：具有資料元素寬度(或大小)為 32 位元(4 位元組)或 64 位元(8 位元組)的 64 位元組向量運算元長度(或大小)(且因此，64 位元組向量是由 16 個雙字大小的元素或 8 個 4 字大小的元素構成)；具有資料元素寬度(或大小)為 16 位元(2 位元組)或 8 位元(1 位元組)的 64 位元組向量運算元長度(或大小)；具有資料元素寬度(或大小)為 32 位元(4 位元組)、64 位元(8 位元組)、16 位元(2 位元組)、或 8 位元(1 位元組)的 32 位元組向量運算元長度(或大小)；以及，具有資料元素寬度(或大小)為 32 位元(4 位元組)、64 位元(8 位元組)、16 位元(2 位元組)、或 8 位元(1 位元組)的 16 位元組向量運算元長度(或大小)；但另些實施例可支援具有較多、較少、或不同的資料元素寬度(例如，128 位元(16 位元組)的資料元素寬度)之較多、較少、及/或不同的向量運算元大小(例如，256 位元組的向量運算元)。

在圖 1A 中的類型 A 指令範本包括：1)在無記憶體存取 105 指令範本中顯示無記憶體存取全捨入控制式運算 110 指令範本與無記憶體存取資料轉換式運算 115 指令範本；以及 2)在記憶體存取 120 指令範本中顯示記憶體存取暫時性 125 指令範本與記憶體存取非暫時性 130 指令範本。在圖 1B 中的類型 B 指令範本包括：1)在無記憶體存取 105 指令範本中顯示無記憶體存取寫入遮罩控制部分捨

入控制式運算 112 指令範本與無記憶體存取寫入遮罩控制 `vsize` 式運算 117 指令範本；以及 2) 在記憶體存取 120 指令範本中顯示記憶體存取寫入遮罩控制 127 指令範本。

通用向量親和指令格式 100 包括以下按說明於圖 1A-1B 中之順序所表列的欄位。

格式欄位 140-在此欄位中的特定值(指令格式識別符值)唯一地識別向量親和指令格式，且因此向量親和指令格式的指令出現在指令流中。就此而論，此欄位就以下意義係選用的：只有通用向量親和指令格式的指令集不需要此欄位。

基本運算欄位 142-其內容用來區別不同的基本運算。

暫存器索引欄位 144-其內容直接地或經由位址產生來指定源與目的運算元它們在暫存器或記憶體中應在的位置。這些包括足夠的位元數用以從 $P \times Q$ (例如， 32×512 、 16×128 、 32×1024 、 64×1024) 的暫存器檔案中選擇 N 個暫存器。雖然在一實施例中 N 可到 3 個源與 1 個目的暫存器，但另些實施例可支援較多或較少的源與目的暫存器(例如，可支援到 2 個源，而這些源的其中一個也做為目的，可支援到 3 個源，而這些源的其中一個也做為目的，可支援到 2 個源與 1 個目的)。

修改符欄位 146-其內容用來區別以通用向量指令格式出現的指令，其指定哪些是記憶體存取而哪些則否；亦即，在無記憶體存取 105 指令範本與記憶體存取 120 指令範本之間做區別。記憶體存取操作會讀取及/或寫入到記

憶體階層(在某些情況中使用暫存器中的值指定源及/或目的位址)，而非記憶體存取操作則否(例如，源與目的係暫存器)。在一實施例中，雖然此欄位也在實施記憶體位址計算的 3 種不同方式間做選擇，但另些實施例可支援較多或較少或不同的方式來實施記憶體位址計算。

擴增運算欄位 150-其內容用來區別除了基本運算以外，在多種不同的運算中還要實施哪一項運算。此欄位係與上下文相關。在本發明的一實施例中，此欄位被分割成類型欄位 168、阿爾發欄位 152、及貝它欄位 154。此擴增運算欄位 150 允許共群的運算在單一個指令中實施，而非 2、3、或 4 個指令。

縮放欄位 160-其內容允許用於產生記憶體位址之索引欄位內容的縮放(例如，其使用 $2^{\text{縮放}} \times \text{索引} + \text{基本}$ 來產生位址)。

位移欄位 162A-其內容被用做為產生記憶體位址的一部分(例如，其使用 $2^{\text{縮放}} \times \text{索引} + \text{基本} + \text{位移}$ 來產生位址)。

位移因數欄位 162B(須注意，位移欄位 162A 直接併接於位移因數欄位 162B 之上係指示一個或另一個被使用)-其內容被用做為位址產生的一部分；其指定要被記憶體存取(N)之大小來縮放的位移因數-其中，N 係記憶體存取中的位元組數(例如，其使用 $2^{\text{縮放}} \times \text{索引} + \text{基本} + \text{經縮放的位移}$ 來產生位址)。為了產生用來計算有效位址的最終位移，冗餘的低階位元予以忽略，且因此，位移因數欄位的內容係被記憶體運算元總大小(N)來乘。N 的值係由處

理器硬體於運行時間根據完整運算碼欄位 174(本文稍後描述)與資料調處欄位 154C 來決定。在以下意義上，位移欄位 162A 與位移因數欄位 162B 係選用的：無記憶體存取 105 指令範本不使用此兩欄位，及/或不同的實施例僅實施其中之一或兩者都不使用。

資料元素寬度欄位 164-其內容識別要使用諸多資料元素寬度中的何者(在某些實施例中用於所有指令；在其它實施例中僅用於某些指令)。在以下意義上，此欄位係選用的：如果僅支援一種資料元素寬度及/或資料元素寬度係使用運算碼的某些態樣來支援，則不需要此欄位。

寫入遮罩欄位 170-其內容係以每一個資料元素位置為基礎來控制目的向量運算元中的該資料元素位置是否反映了基本運算與擴增運算的結果。類型 A 指令範本支援合併寫入遮罩，而類型 B 指令範本支援合併寫入遮罩與歸零寫入遮罩兩者。當合併時，向量遮罩保護目的中之任何組的元素在任何運算(由基本運算與擴增運算所指定)的執行期間不被更新；在另一實施例中，保存目的中與遮罩位元 0 對應之每一個元素的舊值。反之，當歸零時，向量遮罩允許目的中之任何組的元素在任何運算(由基本運算與擴增運算所指定)的執行期間被歸零；在一實施例中，當對應之遮罩位元的值為 0 時，目的的元素被設置成 0。此功能的子集有能力控制要被實施之運算的向量長度(亦即，從第一個到最後一個元素的幅寬可被修改)；不過，被修改的該些元素並不必然連續。因此，寫入遮罩欄位 170 允許

部分的向量運算，包括載入、儲存、算術、邏輯等。在所描述的本發明實施例中，雖然寫入遮罩欄位 170 的內容係在許多寫入遮罩暫存器中選擇包含所要使用之寫入遮罩的一暫存器(且因此寫入遮罩欄位 170 的內容間接地識別所要實施的遮罩)，但替代的實施例代之以或額外允許寫入遮罩欄位 170 的內容直接指定所要實施的遮罩。

立即欄位 172-其內容允許立即的規格。在以下意義上此欄位係選用的：此欄位不存在於不支援立即之通用向量親和格式的實施中，且其不存在於不使用立即的指令中。

類型欄位 168-其內容用來區別不同類型的指令。參考圖 1A-B，此欄位的內容選擇類型 A 或類型 B 指令。在圖 1A-B，使用圓角的方形來指示存在於此欄位中的特定值(例如，圖 1A-B 的類型欄位 168 中分別是類型 A168A 與類型 B168B)。

● 類型 A 的指令範本

在類型 A 之無記憶體存取 105 指令範本的情況中，阿爾發欄位 152 被解譯成 RS 欄位 152A，它的內容用來區別不同類型之擴增運算中的何者要被實施(例如，分別為無記憶體存取捨入型運算 110 指令範本與無記憶體存取資料轉換型運算 115 指令範本指定的捨入 152A.1 與資料轉換 152A.2)，而貝它欄位 154 區別所指定之類型的運算中何者要被實施。在無記憶體存取 105 指令範本中，縮放欄位 160、位移欄位 162A、及位移縮放欄位 162B 不存在。

無記憶體存取指令範本-全捨入控制型運算

在無記憶體存取全捨入控制型運算 110 指令範本中，貝它欄位 154 被解譯成捨入控制欄位 154A，它的內容提供靜態捨入。在所描述的本發明實施例中，雖然捨入控制欄位 154A 包括抑制所有的浮點異常(SAE)欄位 156 與捨入運算控制欄位 158，但在另些實施例中，可支援將此二者概念編碼到相同的欄位內，或僅具有這些概念/欄位的其一或另一(例如，可僅具有捨入運算控制欄位 158)。

SAE 欄位 156-其內容用來區別是否要將異常事件報告去能；當 SAE 欄位 156 的內容指示抑制被致能時，給定的指令不報告任何種類的浮點異常旗標，且不引發任何的浮點異常處理程序。

捨入運算控制欄位 158-其內容區別一組捨入運算中的何者要被實施(例如，捨進、捨去、捨去小數、及捨入最近的數值)。因此，捨入運算控制欄位 158 允許以每個指令為基礎來改變捨入的模式。在本發明的一實施例中，處理器包括用於指定捨入模式的控制暫存器，捨入運算控制欄位 150 的內容置換該暫存器的值。

無記憶體存取指令範本-資料轉換型運算

在無記憶體存取資料轉換型運算 115 指令範本中，貝它欄位 154 被解譯成資料轉換欄位 154B，其內容區別諸多資料轉換中的何者要被實施(例如，無資料轉換、攪

拌、廣播)。

在類型 A 的記憶體存取 120 指令範本的情況中，阿爾發欄位 152 被解譯成逐出提示欄位 152B，其內容區別要使用何者逐出提示(在圖 1A 中，分別為記憶體存取暫時性 125 指令範本與記憶體存取非暫時性 130 指令範本指定暫時性 152B.1 與非暫時性 152B.2)，而貝它欄位 154 被解譯成資料調處欄位 154C，其內容區別諸多資料調處操作(也稱為基元)中的何者要被實施(例如，無調處；廣播；源的上轉換；及目的的下轉換)。記憶體存取 120 指令範本包括縮放欄位 160 及選用的位移欄位 162A 或位移縮放欄位 162B。

向量記憶體指令實施來自記憶體的向量載入及前往往記憶體的向量儲存，支援轉換。就正規的向量指令而言，向量記憶體指令以逐資料元素的方式與記憶體來/往傳送資料，實際被傳送的元素係由被選做為寫入遮罩之向量遮罩的內容所命令。

記憶體存取指令範本-暫時性

暫時性資料係很有可能很快就會被再次使用且快到足以得益於快取的資料。不過，此為提示，且不同的處理器以不同的方式實施此提示，包括完全忽略該提示。

記憶體存取指令範本-非暫時性

非暫時性資料係不太可能很快就被再次使用而快到足

以得益於在第一階快取記憶體中進行快取的資料，且應給予逐出的優先權。不過，此為提示，且不同的處理器以不同的方式實施此提示，包括完全忽略該提示。

類型 B 的指令範本

在類型 B 指令範本的情況中，阿爾發欄位 152 被解譯成寫入遮罩控制(Z)欄位 152C，其內容區別被寫入遮罩欄位 170 所控制的寫入遮罩應為合併或歸零。

在類型 B 之非記憶體存取 105 指令範本的情況中，部分的貝它欄位 154 被解譯成 RL 欄位 157A，其內容區別不同的擴增運算類型中的何者要被實施(例如，分別為無記憶體存取寫入遮罩控制部分捨入控制型運算 112 指令範本與無記憶體存取寫入遮罩控制 VSIZE 型運算 117 指令範本指定捨入 157A.1 與向量長度(VSIZE) 157A.2)，而貝它欄位 154 的其餘部分則區別該指定類型之運算中的何者要被實施。在無記憶體存取 105 指令範本中，縮放欄位 160、位移欄位 162A、及位移縮放欄位 162B 不存在。

在無記憶體存取寫入遮罩控制部分捨入控制型運算 110 指令範本中，貝它欄位 154 的其餘部分被解譯成捨入運算欄位 159A，且異常事件報告被去能(給定的指令不報告任何種類的浮點異常旗標，且不引發任何的浮點異常處置程序)。

捨入運算控制欄位 159A-正如捨入運算控制欄位 158，其內容區別一群捨入運算中的何者要被實施(例如，

捨進、捨去、捨去小數、及捨入最近的數值)。因此，捨入運算控制欄位 159A 允許以每個指令為基礎來改變捨入的模式。在本發明的一實施例中，處理器包括用於指定捨入模式的控制暫存器，以捨入運算控制欄位 150 的內容置換該暫存器的值。

在無記憶體存取寫入遮罩控制 VSIZE 型運算 117 指令範本中，貝它欄位 154 的其餘部分則被解譯成向量長度欄位 159B，其內容區別要實施諸多資料向量長度(例如，128、256、或 512 位元組)中的何者。

在類型 B 之記憶體存取 120 指令範本的情況中，部分的貝它欄位 154 被解譯成廣播欄位 157B，其內容區別是否要實施廣播類型資料的調處操作，而貝它欄位 154 的其餘部分則被解譯成向量長度欄位 159B。記憶體存取 120 指令範本包括縮放欄位 160 及選用的位移欄位 162A 或位移縮放欄位 162B。

關於通用向量親和指令格式 100，所顯示的完整運算碼欄位 174 包括格式欄位 140、基本運算欄位 142、及資料元素寬度欄位 164。雖然一實施例顯示完整運算碼欄位 174 包括了所有這些欄位，但在不支援所有這些欄位的實施例中，完整運算碼欄位 174 並不包括所有這些欄位。完整運算碼欄位 174 提供運算碼。

擴增運算欄位 150、資料元素寬度欄位 164、及寫入遮罩欄位 170 允許以每個指令為基礎將這些特徵指定於通用向量親和指令格式中。

寫入遮罩欄位與資料元素寬度欄位的組合建立了具有類型的指令，因為它們允許依據不同的資料元素寬度來施加遮罩。

在類型 A 與類型 B 中所見到的各種指令範本有利於不同的情況。在本發明的某些實施例中，不同的處理器或處理器內不同的核心可僅支援類型 A、僅支援類型 B，或支援兩者。例如，意欲用於通用計算的高性能通用亂序核心可僅支援類型 B，意欲以用於圖形及/或科學(產出量)計算為主的核心可僅支援類型 A，以及意欲用於兩者的核心可支援兩者(當然，具有混合該兩類某些範本與指令，但並非該兩類型全部之範本與指令的核心也在本發明的範圍之內)。此外，單處理器可包括多個核心，其所有核心都支援相同的類型，或其中不同的核心支援不同的類型。例如，在圖形與通用核心分開的處理器中，其中一個意欲以圖形及/或科學計算為主的圖形核心可僅支援類型 A，而具有亂序執行及暫存器更名意欲用於通用計算之高性能通用核心的一或多個通用核心僅支援類型 B。另一不具有分離之圖形核心的處理器，可包括支援類型 A 與類型 B 兩者的一或多個通用循序或亂序核心。當然，在本發明的不同實施例中，來自某一類的特徵也可在另一類中實施。以高階語言所撰寫的程式可寫成(例如，即時編譯或靜態編譯)多種不同的可執行形式，包括：1)僅具有用於執行之目標處理器所支援之類型的指令之形式；或 2)具有使用所有類型之指令的不同組合所寫成的替代常式，且具有依據

當下執行該碼之處理器所支援之指令來選擇所要執行之常式的控制流程碼之形式。

B. 例示性特定向量親和指令格式

圖 2 係說明按照本發明之實施例之例示性的特定向量親和指令格式的方塊圖。圖 2 顯示的特定向量親和指令格式 200，在以下意義上其為特定的，其指定位置、大小、解譯、及欄位的順序、以及那些欄位中之某些所使用的值。特定向量親和指令格式 200 可用來延伸 x86 指令集，且因此某些欄位與現有 x86 指令集及其延伸(例如，AVX)中使用的那些類似或相同。此格式保持與含延伸之現有 x86 指令集的前置編碼欄位、真實運算碼位元組欄位、MOD R/M 欄位、SIB 欄位、位移欄位、及立即欄位的一致。圖示說明來自圖 2 的欄位映射到來自圖 1 的欄位內。

須瞭解，基於說明之目的，雖然本發明之實施例係參考通用向量親和指令格式 100 情況中的特定向量親和指令格式 200 來描述，但除了申請的部分之外，本發明並不限於特定向量親和指令格式 200。例如，通用向量親和指令格式 100 考慮各不同欄位之各種可能的大小，而特定向量親和指令格式 200 顯示具有特定大小的欄位。藉由特定的例子，雖然特定向量親和指令格式 200 中的資料元素寬度欄位 164 被說明成 1 位元的欄位，但本發明並無此限制(亦即，通用向量親和指令格式 100 考慮其它大小的資料元素寬度欄位 164)。

通用向量親和指令格式 100 包括以下按圖 2A 中說明之順序表列於下的欄位。

EVEX 前置(位元組 0-3)202-被編碼成 4 位元組的形式。

格式欄位 140(EVEX 位元組 0，位元[7:0])-第一位元組(EVEX 位元組 0)係格式欄位 140，且其包含 0x62(唯一值，在本發明的一實施例中用來區別向量親和指令格式)。

第二-第四位元組(EVEX 位元組 1-3)包括提供特定能力的若干位元欄位。

REX 欄位 205(EVEX 位元組 1，位元[7-5])-由 EVEX.R 位元欄位(EVEX 位元組 1，位元[7]-R)、EVEX.X 位元欄位(EVEX 位元組 1，位元[6]-X)及 157BEX 位元組 1，位元[5]-B)構成。如對應的 VEX 位元欄位，EVEX.R、EVEX.X、及 EVEX.B 位元欄位提供相同的功能，且使用 1 的補數形式來編碼，即 ZMM0 被編碼成 1111B、ZMM15 被編碼成 0000B。指令的其它欄位編碼暫存器索引之較低的 3 個位元，如習知技術(rrr、xxx、及 bbb)，以使得 Rrrr、Xxxx、及 Bbbb 可藉由加 EVEX.R、EVEX.X、及 EVEX.B 來形成。

REX'欄位 110-此乃 REX'欄位 110 的第一部分，且是 EVEX.R'位元欄位(EVEX 位元組 1，位元[4]-R')，其被用來編碼延伸之 32 暫存器組的高 16 位元與低 16 位元。在本發明的一實施例中，此位元連同以下指示的其它位元以

位元反置格式被儲存(習知的 x86 32 位元模式)，以與 BOUND 指令區別，它的真實運算碼位元組為 62，但在 MOD R/M 欄位(下文中描述)中不接受值 11 的 MOD 欄位；本發明的替代實施例不以反置格式儲存此與以下指示的其它位元。使用 1 的值來編碼低 16 暫存器。換言之，R'Rrrr 係結合 EVEX.R'、EVEX.R、及來自其它欄位的其它 RRR 所形成。

● 運算碼映射欄位 215(EVEX 位元組 1，位元[3：0]-mmmm)-其內容編碼隱含的前導運算碼位元組(0F、0F 38、或 0F 3)。

資料元素寬度欄位 164(EVEX 位元組 2，位元[7]-W)-係以符號 EVEX.W 來表示。EVEX.W 係用來定義資料類型的粒度(大小)(32 位元資料元素或 64 位元資料元素)。

EVEX.vvvv 220(EVEX 位元組 2，位元[6：3]-vvvv)-EVEX.vvvv 的角色可包括如下：1)EVEX.vvvv 編碼第一源暫存器運算元，以反置(1 的補數)的形式指定，且對具有 2 或更多個源運算元的指令有效；2)EVEX.vvvv 編碼目的暫存器運算元，以 1 之補數的形式指定，用於某些向量移位；或 3)EVEX.vvvv 不編碼任何運算元，此欄位被保留且應包含 1111b。因此，EVEX.vvvv 欄位 220 編碼第一源暫存器區分符(specifier)的 4 個低階位元，以反置(1 的補數)的形式儲存。視指令而定，使用額外不同的 EVEX 位元欄位來將區分符的大小延伸到 32 個暫存器。

EVEX.U 168 類型欄位(EVEX 位元組 2，位元[2]-U)-

如果 EVEX.U=0，其指示類型 A 或 EVEX.U0；如果 EVEX.U=1，則指示類型 B 或 EVEX.U1。

前置編碼欄位 225(EVEX 位元組 2，位元[1：0]-pp)-為基本運算欄位提供額外的位元。除了提供對於 EVEX 前置格式中之舊有 SSE 指令的支援，此欄位也具有緊縮 SIMD 前置的優點(不像 SIMD 前置需要用一個位元組來表示，EVEX 前置僅需要 2 位元)。在一實施例中，為支援舊有的 SSE 指令，在舊有格式與 EVEX 前置格式中都使用 SIMD 前置(66H、F2H、F3H)，這些舊有的 SIMD 前置被編碼到 SIMD 前置編碼欄位內；且在運行時間在被提供給解碼器的 PLA 之前，先被擴展成舊有的 SIMD 前置(如此，PLA 可執行這些舊有指令的舊有與 EVEX 格式而不需要修改)。雖然較新的指令可直接使用 EVEX 前置編碼欄位的內容做為運算碼延伸，但為求一致性某些實施例以類似的方式擴展，但允許藉由這些舊有 SIMD 前置指定不同的意義。替代的實施例可重新設計 PLA 以支援 2 位元的 SIMD 前置編碼，且因此不需要擴展。

阿爾發欄位 152(EVEX 位元組 3，位元[7]-EH；也稱為 EVEX.EH、EVEX.rs、EVEX.RL、EVEX.寫入遮罩控制，及 EVEX.N；也以 α 來說明)-如前所描述，此欄位係與上下文相關。

貝它欄位 154(EVEX 位元組 3，位元[6：4]-SSS，也稱為 EVEX.s₂₋₀、EVEX.r₂₋₀、EVEX.rr1、EVEX.LL0、EVEX.LLB；也以 $\beta\beta\beta$ 來說明)-如前所描述，此欄位係與上

下文相關。

REX'欄位 110-此為 REX'欄位的剩餘部分，且是 EVEX.V'位元欄位(EVEX 位元組 3，位元[3]-V')，其可用來編碼延伸之 32 暫存器組的高 16 或低 16。此位元係以位元反置格式儲存。使用 1 的值來編碼低 16 暫存器。換言之，V'VVVV 係結合 EVEX.V'、EVEX.vvvv 所形成。

寫入遮罩欄位 170-(EVEX 位元組 3，位元[2:0]-kkk)-如前所述，其內容指定寫入遮罩暫存器中之暫存器的索引。在本發明的一實施例中，特定值 EVEX.kkk=000 具有隱含特定之指令不使用寫入遮罩的特殊行為(此可用各種方式來實施，包括使用寫入遮罩硬線到所有特定的指令，或使用旁通遮罩硬體的硬體)。

真實運算碼欄位 230(位元組 4)也稱為運算碼位元組。部分的運算碼被指定在此欄位內。

MOD R/M 欄位 240(位元組 5)包括 MOD 欄位 242、Reg 欄位 244、及 R/M 欄位 246。如前所述，MOD 欄位 242 的內容區別記憶體存取與非記憶體存取操作。Reg 欄位 244 之角色可綜述成兩種情況：編碼目的暫存器運算元或源暫存器運算元，或當作運算碼延伸且不用來編碼任何指令運算元。R/M 欄位 246 之角色可包括如下：參考記憶體位址來編碼指令運算元，或者，編碼目的暫存器運算元或源暫存器運算元。

縮放、索引、基本(SIB)位元組(位元組 6)-如前所述，縮放欄位 150 的內容被用來產生記憶體位址。SIB.xxx 254

與 SIB.bbb 256-這些欄位的內容已於先前關於暫存器索引 Xxxx 與 Bbbb 提及。

位移欄位 162A(位元組 7-10)-當 MOD 欄位 242 包含 10 時，位元組 7-10 係位移欄位 162A，且其工作與舊有的 32 位元位移(disp32)相同，且是以位元組的粒度工作。

位移因數欄位 162B(位元組 7)-當 MOD 欄位 242 包含 01 時，位元組 7 係位移因數欄位 162B。此欄位的位置與舊有 x86 指令集 8 位元位移(disp8)的位置相同，其以位元組的粒度工作。由於 disp8 係正負號延伸，其僅可在 -128 與 127 位元組的偏移之間定址；就 64 位元組快取線而論，disp8 使用 8 位元，其僅可設定到 4 個真正有用的值 -128、-64、0、及 64；由於通常需要較大的範圍，因此使用 disp32；不過，disp32 需要 4 位元組。disp8 與 disp32 相較，位移因數欄位 162B 係 disp8 的重新解譯；當使用位移因數欄位 162B 時，實際的位移係由位移因數欄位的內容乘以記憶體運算元存取的大小(N)來決定。此類型的位移被稱為 $\text{disp8} * N$ 。此縮短了平均的指令長度(位移使用單個位元組，但具有更大的範圍)。此經壓縮的位移，係基於有效的位移係記憶體存取之粒度之倍數的假設，且因此，位址偏移之冗餘的低階位元不需要被編碼。換言之，位移因數欄位 162B 取代舊有 x86 指令集的 8 位元位移。因此，位移因數欄位 162B 的編碼方式，僅除了 disp8 被超載到 $\text{disp8} * N$ 之外，其餘與 x86 指令集的 8 位元位移相同(因此，在 ModRM/SIB 編碼規則方面沒有改變)。換言

之，在編碼規則或編碼長度方面沒有改變，僅藉由硬體來解譯位移值(其需要以記憶體運算元的大小來縮放位移，以得到逐位元組的位址偏移)。

立即欄位 172 操作如前所述。

完整運算碼欄位

圖 2B 的方塊圖說明按照本發明之實施例，組成完整運算碼欄位 174 之特定向量親和指令格式 200 的欄位。特別是，完整運算碼欄位 174 包括格式欄位 140、基本運算欄位 142、及資料元素寬度(W)欄位 164。基本運算欄位 142 包括，前置編碼欄位 225、運算元映射欄位 215、及真實運算碼欄位 230。

暫存器索引欄位

圖 2C 的方塊圖說明按照本發明之實施例，組成暫存器索引欄位 144 之特定向量親和指令格式 200 的欄位。特別是，暫存器索引欄位 144 包括 REX 欄位 205、REX'欄位 210、MODR/M.reg 欄位 244、MODR/M.r/m 欄位 246、VVVV 欄位 220、xxx 欄位 254、及 bbb 欄位 256。

擴增運算欄位

圖 2D 的方塊圖說明按照本發明之實施例，組成擴增運算欄位 150 之特定向量親和指令格式 200 的欄位。當類型(U)欄位 168 包含 0 時，其表示 EVEX.U0(類型 A

168A)；當其包含 1 時，其表示 EVEX.U1(類型 B 168B)。

當 $U=0$ 且 MOD 欄位 242 包含 11 時(表示無記憶體存取操作)，阿爾發欄位 152(EVEX 位元組 3，位元[7]-EH)被解譯成 rs 欄位 152A。當 rs 欄位 152A 包含 1 時(捨入 152A.1)，貝它欄位 154(EVEX 位元組 3，位元[6:4]-SSS)被解譯成捨入控制欄位 154A。捨入控制欄位 154A 包括 1 位元的 SAE 欄位 156 及 2 位元的捨入運算欄位 158。當 rs 欄位 152A 包含 0 時(資料轉換 152A.2)，貝它欄位 154(EVEX 位元組 3，位元[6:4]-SSS)被解譯成 3 位元的資料轉換欄位 154B。當 $U=0$ 且 MOD 欄位 242 包含 00、01、或 10 時(表示記憶體存取操作)，阿爾發欄位 152(EVEX 位元組 3，位元[7]-EH)被解譯成逐出提示(EH)欄位 152B，及貝它欄位 154(EVEX 位元組 3，位元[6:4]-SSS)被解譯成 3 位元的資料調處欄位 154C。

當 $U=1$ 時，阿爾發欄位 152(EVEX 位元組 3，位元[7]-EH)被解譯成寫入遮罩控制(Z)欄位 152C。當 $U=1$ 且 MOD 欄位 242 包含 11 時(表示無記憶體存取操作)，部分的貝它欄位 154(EVEX 位元組 3，位元[4]- S_0)被解譯成 RL 欄位 157A；當其包含 1 時(捨入 157A.1)，貝它欄位 154 的其餘部分(EVEX 位元組 3，位元[6-5]- $S_{2..1}$)被解譯成捨入操作欄位 159A，然而當 RL 欄位 157A 包含 0 時(VSIZE 157.A2)，貝它欄位 154 的其餘部分(EVEX 位元組 3，位元[6-5]- $S_{2..1}$)被解譯成向量長度欄位 159B(EVEX 位元組 3，位元[6-5]- $L_{1..0}$)。當 $U=1$ 且 MOD 欄位 242 包含 00、

01、或 10 時(表示記憶體存取操作)，貝它欄位 154(EVEX 位元組 3，位元[6：4]-SSS)被解譯成向量長度欄位 159B (EVEX 位元組 3，位元[6-5]-L_{1,0})與廣播欄位 157B(EVEX 位元組 3，位元[4]-B)。

C. 例示性暫存器架構

圖 3 係按照本發明一實施例之暫存器架構 300 的方塊圖。在說明的實施例中，有 32 個寬度 512 位元的向量暫存器 310；這些暫存器稱為 zmm0 到 zmm31。下 16 個 zmm 暫存器的低階 256 位元被疊覆在暫存器 ymm0-16 上。下 16 個 zmm 暫存器的低階 128 位元(ymm 暫存器的低階 128 位元)被疊覆在暫存器 xmm0-15 上。特定向量親和指令格式 200 在這些疊覆之暫存器檔案上的操作如下表之說明。

可調整的向量長度	類型	操作	暫存器
不包括向量長度欄位 159B 的指令範本	A(圖 1A；U=0)	110, 115, 125, 130	zmm 暫存器(向量長度係 64 位元組)
	B(圖 1B；U=1)	112	zmm 暫存器(向量長度係 64 位元組)
包括向量長度欄位 159B 的指令範本	B(圖 1B；U=1)	117, 127	zmm、ymm、或 xmm 暫存器(向量長度係 64 位元組、32 位元組、或 16 位元組)，視向量長度欄位 159B 而定

換言之，向量長度欄位 159B 在最大長度與一或多個

其它較短長度之間做選擇，在此，每一個此等較短之長度係前一長度之長度的一半；且沒有向量長度欄位 159B 的指令範本以最大向量長度運算。此外，在一實施例中，特定向量親和指令格式 200 的類型 B 指令範本對封裝的或純量的單/雙精度浮點資料及封裝的或純量的整數資料運算。純量運算係對 zmm/ymm/xmm 暫存器中最低階之資料元素位置實施的運算；較高階資料元素位置被保持與送入指令之前相同或被歸零，視實施例而定。

寫入遮罩暫存器 315-在說明的實施例中，有 8 個寫入遮罩暫存器(k0 到 k7)，每一個的大小為 64 位元。在替代的實施例中，寫入遮罩暫存器 315 的大小係 16 位元。如前所述，在本發明的一實施例中，向量遮罩暫存器 k0 不能用做為寫入遮罩；當編碼通常指示 k0 被用於寫入遮罩時，其選擇 0xFFFF 的硬接線寫入遮罩，有效地將該指令的寫入遮罩去能。

通用暫存器 325-在說明的實施例中，其有 16 個 64 位元的通用暫存器，其可連同現有的 x86 定址模式用來定址記憶體運算元。這些暫存器係藉由名稱 RAX、RBX、RCX、RDX、RBP、RSI、RDI、RSP、及 R8 到 R15 來引用。

純量浮點堆疊暫存器檔案(x87 堆疊)345，MMX 封裝整數平面暫存器檔案 350 疊置於其上-在說明的實施例中，x87 堆疊係 8 元素堆疊，使用 x87 指令集延伸對 32/64/80 位元浮點資料實施純量浮點運算；而 MMX 暫存

器被用來對 64 位元的封裝整數資料實施運算，以及保持 MMX 與 XMM 暫存器之間所實施之某些運算的運算元。

本發明之替代的實施例可使用較寬或較窄的暫存器。此外，本發明之替代的實施例可使用較多、較少、或不同的暫存器檔案與暫存器。

D. 例示性核心架構、處理器、及電腦架構

在不同的處理器中處理器核心可按不同之目的用不同的方式實施。例如，此等核心之實施可包括：1)意欲用於通用計算的通用循序核心；2)意欲用於通用計算的高性能通用亂序核心；3)意欲主要用於圖形及/或科學(產出量)計算的特殊用途核心。不同處理器的實施可包括：1)包括一或多個意欲用於通用計算之通用循序核心及/或一或多個意義用於通用計算之通用亂序核心的 CPU；以及 2)包括一或多個意欲主要用於圖形及/或科學(產出量)之特殊用途核心的協同處理器。此等不同的處理器導致不同的電腦系統架構，其可包括：1)在與 CPU 分離之晶片上的協同處理器；2)在與 CPU 同一封裝內之獨立晶粒上的協同處理器；3)在與 CPU 同一晶粒上的協同處理器(在此情況中，此等協同處理器有時稱為特殊用途邏輯，諸如整合式圖形及/或科學(產出量)邏輯，或諸如特殊用途核心)；以及 4)系統單晶片，其可包括位在同一晶粒上之上述的 CPU(有時被稱為應用核心或應用處理器)、上述的協同處理器、及附加功能。接下來描述例示性的核心架構，接著是例示

性處理器及電腦架構的描述。

圖 4A 的方塊圖說明按照本發明之實施例的例示性循序管線與例示性暫存器更名、亂序發出/執行管線兩者。圖 4B 的方塊圖說明包括在按照本發明實施例之處理器中之循序架構核心的例示性實施例與例示性暫存器更名、亂序發出/執行架構核心兩者。圖 4A-B 中的實線框說明循序管線與循序核心，而選用的額外虛線框說明暫存器更名、亂序發出/執行管線與核心。基於循序態樣係亂序態樣的子集，因此將描述亂序態樣。

在圖 4A 中，處理器管線 400 包括提取級 402、長度解碼級 404、解碼級 406、配置級 408、更名級 410、排程(也稱為調度或發出)級 412、暫存器讀取/記憶體讀取級 414、執行級 416、寫回/記憶體寫入級 418、異常處置級 422、與提交級 424。

圖 4B 顯示處理器核心 490 包括前端單元 430，耦接至執行引擎單元 450，且兩者都耦接至記憶體單元 470。核心 490 可以是精減指令集計算(RISC)核心、複雜指令集計算(CISC)核心、極長指令字(VLIW)核心、或混合或替代核心類型。如另一選項，核心 490 可以是特殊用途核心，例如，諸如網路或通訊核心、壓縮引擎、協同處理器核心、通用計算圖形處理單元(GPGPU)核心、圖形核心、或類似物。

前端單元 430 包括耦接至指令快取單元 434 的分支預測單元 432，指令快取單元 434 耦接至指令轉譯後備緩衝

區(TLB)436，TLB436 耦接至指令提取單元 438，指令提取單元 438 耦接至解碼單元 440。解碼單元 440(或解碼器)可解碼指令，並產生諸如一或多個微操作、微碼登錄點、微指令、其它指令、或其它控制信號的輸出，這些係解碼自原始指令，或以其它方式反映或導出自原始指令。解碼單元 440 可使用各種不同的機制來實施。適合之機制的例子包括但不限於查找表、硬體實施、可程式邏輯陣列(PLA)、微碼唯讀記憶體(ROM)、等。在一實施例中，核心 490 包括微碼 ROM 或其它媒體，其儲存用於某些微指令的微碼(例如，在解碼單元 440 中或在前端單元 430 中)。解碼單元 440 耦接至執行引擎單元 450 中的更名/配置器單元 452。

執行引擎單元 450 包括更名/配置器單元 452，其耦接至止用單元 454 與一組一或多個排程器單元 456。排程器單元 456 代表任何數量之不同的排程器，包括保留站、中央指令視窗等。排程器單元 456 耦接至實體暫存器檔案單元 458。每一個實體暫存器檔案單元 458 代表一或多個實體暫存器檔案，每一個不同的實體暫存器檔案儲存一或多個不同的資料類型，諸如純量整數、純量浮點、封裝的整數、封裝的浮點、向量整數、向量浮點、狀態(例如，指令指標，其為下一個要被執行之指令的位址)、等。在一實施例中，實體暫存器檔案單元 458 包含向量暫存器單元、寫入遮罩暫存器單元、及純量暫存器單元。這些暫存器單元可提供架構向量暫存器、向量遮罩暫存器、及通用

暫存器。實體暫存器檔案單元 458 被止用單元 454 重疊用以說明其中可實施暫存器更名或亂序執行之各種不同的方式(例如，使用重排序緩衝器與止用暫存器檔案；使用未來檔案、歷史緩衝器、及止用暫存器檔案；使用暫存器映射與暫存器池等)。止用單元 454 與實體暫存器檔案單元 458 耦接至執行叢集 460。執行叢集 460 包括一組一或多個執行單元 462 與一組一或多個記憶體存取單元 464。執行單元 462 可對各種不同類型的資料(例如，純量浮點、封裝的整數、封裝的浮點、向量整數、向量浮點)實施各種操作(例如，移位、加、減、乘)。雖然某些實施例可包括若干個專用於特定功能或功能組的執行單元，但其它實施例可包括僅一個執行單元或多個執行單元全部用來實施所有的功能。所顯示的排程器單元 456、實體暫存器檔案單元 458、與執行叢集 460 可能有複數個，這是因為某些實施例為某些類型的資料/操作建立獨立的管線(例如，純量整數管線、純量浮點/封裝的整數/封裝的浮點/向量整數/向量浮點管線、及/或記憶體存取管線，其每一個具有它們自己的排程器單元、實體暫存器檔案單元、及/或執行叢集-且在獨立之記憶體存取管線的情況中，某些實施例中實施僅具有記憶體存取單元 464 之管線的執行叢集)。亦須瞭解，在使用分離之管線的情況中，這些管線中的一或多個管線可亂序發出/執行而其它管線則循序。

記憶體存取單元 464 組耦接至記憶體單元 470，其包括資料 TLB 單元 472 耦接至資料快取單元 474 再耦接至 2

階(L2)快取單元 476。在一例示性的實施例中，記憶體存取單元 464 可包括載入單元、儲存位址單元、及儲存資料單元，每一個單元都耦接至記憶體單元 470 中的資料 TLB 單元 472。指令快取單元 434 進一步耦接至記憶體單元 470 中的 2 階(L2)快取單元 476。L2 快取單元 476 耦接至一或多個其它階的快取，且最終耦接至主記憶體。

舉例來說，例示性的暫存器更名、亂序發出/執行核心架構可按如下來實施管線 400：1)指令提取 438 實施提取及長度解碼級 402 與 404；2)解碼單元 440 實施解碼級 406；3)更名/配置器單元 452 實施配置級 408 與更名級 410；4)排程器單元 456 實施排程級 412；5)實體暫存器檔案單元 458 與記憶體單元 470 實施暫存器讀取/記憶體讀取級 414；執行叢集 460 實施執行級 416；6)記憶體單元 470 與實體暫存器檔案單元 458 實施寫回/記憶體寫入級 418；7)在異常處置級 422 中可包括各種單元；以及 8)止用單元 454 與實體暫存器檔案單元 458 實施提交級 424。

核心 490 可支援包括本文所描述之指令的一或多個指令集(例如，x86 指令集(含某些延伸，在較新的版本中已加入)；加州森尼韋爾之 MIPS Technologies 的 MIPS 指令集；加州森尼韋爾之 ARM Holdings 的 ARM 指令集(含選用的附加延伸，諸如 NEON))。在一實施例中，核心 490 包括用以支援封裝之資料指令集延伸(例如，AVX1、AVX2)的邏輯，藉由允許很多多媒體應用程式所使用的操作可使用封裝資料來實施。

須理解，核心可支援多執行緒(執行兩或多個平行的操作或執行緒組)，且可用多種方式來做到，包括時間切割多執行緒、同步多執行緒(在此，單實體核心為每一個執行緒提供邏輯核心，該實體核心係同步多執行緒)，或它們的組合(例如，時間分割提取與解碼並於之後同步多執行緒，諸如 Intel® 的超執行緒技術(Hyperthreading technology)。

雖然是在亂序執行的環境來描述暫存器更名，但須瞭解，暫存器更名可用於循序架構中。雖然所說明的處理器實施例也包括各自獨立的指令與資料快取單元 434/474 和共享的 L2 快取單元 476，但替代的的實施例可具有用於指令與資料兩者的單一內部快取記憶體，諸如，例如，1 階(L1)內部快取記憶體，或多階的內部快取記憶體。在某些實施例中，系統可包括內部快取記憶體與外部快取記憶體的組合，外部快取記憶體係在核心及/或處理器的外部。或者，所有的快取都可在核心及/或處理器的外部。

圖 5A-B 說明更特定的例示性循序核心架構的方塊圖，其核心是晶片中之數個邏輯方塊(包括相同類型及/或不同類型的其它核心)的其中之一。邏輯方塊通過高帶寬互連網路(例如，環狀網路)與某些固定的功能邏輯、記憶體 I/O 介面、及其它必要的 I/O 邏輯通訊，視應用而定。

圖 5A 的方塊圖係按照本發明之實施例的單處理器核心，連同其與晶粒上之互連網路 502 的連接與它的 2 階(L2)快取 504 的本地子集。在一實施例中，指令解碼器

500 支援具有封裝資料指令集延伸的 x86 指令集。L1 快取 506 允許進入到純量與向量單元之對快取記憶體的低延遲存取。雖然在一實施例中(為簡化設計)，純量單元 508 與向量單元 510 使用個別的暫存器組(分別為純量暫存器 512 與向量暫存器 514)，且它們之間的資料傳送係寫入到記憶體並接著從 1 階(L1)快取 506 中讀回，但本發明的替代實施例可使用不同的方法(例如，使用單個暫存器組，或包括允許在兩暫存器檔案之間傳送資料的通訊路徑，而不寫入與讀回)。

L2 快取 504 的本地子集係整體 L2 快取的一部分，其被分割成分開的本地子集，每個處理器核心一個。每一個處理器核心具有到它自己之 L2 快取 504 本地子集的直接存取路徑。藉由處理器核心所讀取的資料被儲存在它的 L2 快取子集 504 中，且可被快速地存取，與其它處理器核心平行存取它們自己的本地 L2 快取子集。處理器核心所寫的資料被儲存在其自己的 L2 快取子集 504 中，且如有需要，清除其它子集中的此資料。環狀網路確保共享資料的一致性。環狀網路係為雙向以允許代理，諸如處理器核心、L2 快取、及其它邏輯方塊在晶片內彼此通訊。每一環狀資料徑在每個方向的寬度為 1012 位元。

圖 5B 係按照本發明之實施例，圖 5A 中之處理器核心的部分展開圖。圖 5B 包括 L1 快取 504 的 L1 資料快取 506A 部分，以及關於向量單元 510 及向量暫存器 514 的更多細節。特別是，向量單元 510 係 16 位元寬的向量處

理單元(VPU)(見 16 位元寬 ALU 528)，其執行一或多個整數、單精度浮點、及雙精度浮點指令。VPU 支援以攪拌單元 520 攪拌暫存器的輸入，以數值轉換單元 522A-B 做數值轉換，及以複製單元 524 複製在記憶體輸入上。寫入遮罩暫存器 526 允許預測所得到的向量寫入。

圖 6 係按照本發明之實施例之處理器 600 的方塊圖，其具有一個以上的核心，可具有整合式記憶體控制器、且可具有整合式圖形。圖 6 中的實線方塊說明處理器 600 具有單核心 602A、系統代理 610、一組一或多個匯流排控制器單元 616，而選用的附加虛線方塊說明替代的處理器 600，具有多個核心 602A-N、系統代理單元 610 中的一或多個整合式記憶體控制器單元 614，以及特殊用途邏輯 608。

因此，處理器 600 之不同的實施可包括：1)CPU，具有係為整合式圖形及/或科學(產出量)邏輯(其可包括一或多個核心)的特殊用途邏輯 608，以及係為一或多個通用核心(例如，通用循序核心、通用亂序核心、或兩者組合)的核心 602A-N；2)協同處理器，具有意欲主要用於圖形及/或科學(產出量)之大量特殊用途核心的核心 602A-N；以及，3)協同處理器，具有係為大量通用循序核心的核心 602A-N。因此，處理器 600 可以是通用處理器、協同處理器或特殊用途處理器諸如，例如，網路或通訊處理器、壓縮引擎、圖形處理器、通用圖形處理單元(general purpose graphics processing unit; GPGPU)、高產出量多重整合式

核心(MIC)協同處理器(包括 30 個以上的核心)、內嵌式處理器、或類似物。處理器可在一或多個晶片上實施。處理器 600 可以是一或多片基板的一部分及/或使用許多處理技術其中任何技術諸如，例如，BiCMOS、CMOS、或 NMOS 在一或多片基板上實施。

記憶體階層包括核心內的一或多階快取、一組一或多個共用快取單元 606、以及耦接至整合式記憶體控制器單元 614 組的外部記憶體(未顯示)。共用快取單元 606 組可包括一或多個中階快取，諸如 2 階(L2)、3 階(L3)、4 階(L4)、或其它階快取、末階快取(LLC)、及/或它們的組合。雖然在一實施例中以環狀式互連單元 612 來互連整合式圖形邏輯 608、共用快取單元 606 組、及系統代理單元 610/整合式記憶體控制器單元 614，但替代的實施例可使用許多習知技術來互連這些單元。在一實施例中，一或多個快取單元 606 與核心 602A-N 之間保持一致性。

在某些實施例中，一或多個核心 602A-N 具有多執行緒能力。系統代理 610 包括協調與操作核心 602A-N 的那些組件。系統代理單元 610 可包括例如電力控制單元(PCU)與顯示單元。PCU 可為或可包括調節核心 602A-N 與整合式圖形邏輯 608 之電力狀態所需的邏輯與組件。顯示單元用於驅動一或多平台外部連接的顯示器。

就架構指令集而言，核心 602A-N 可以是同質或異質的；亦即，兩或多個核心 602A-N 有能力執行相同的指令集，而其它的核心則僅有能力執行該指令集的子集或不同

的指令集。

圖 7-10 係例示性電腦架構的方塊圖。習知技術中用於膝上型電腦、桌上型電腦、手持式 PC、個人數位式助理、工程工作站、伺服器、網路裝置、網路集線器、交換機、內嵌式處理器、數位信號處理器(DSP)、圖形裝置、遊戲機、機上盒、微控制器、行動電話、可攜式媒體播放機、手持式裝置、及各種各樣其它電子裝置的其它系統設計與組構也都適用。通言之，有能力結合如本文所揭示之處理器及/或其它執行邏輯的各種系統或電子裝置也一體適用。

現參閱圖 7，圖中顯示按照本發明一實施例之系統 700 的方塊圖。系統 700 可包括耦接至控制器集線器 720 的一或多個處理器 710、715。在一實施例中，控制器集線器 720 包括圖形記憶體控制器集線器(GMCH)790 與輸入/輸出集線器(IOH)750(其可以在不同的晶片上)；GMCH 790 包括記憶體與圖形控制器，記憶體 740 與協同處理器 745 與其耦接；IOH 750 將輸入/輸出(I/O)裝置 760 耦接至 GMCH 790。或者，記憶體與圖形控制器其中之一或兩者都整合在處理器內(如本文之描述)，記憶體 740 與協同處理器 745 直接耦接至處理器 710，且控制器集線器 720 與 IOH 750 位在單晶片上。

附加處理器 715 的選用性質在圖 7 中以虛線指示。每一個處理器 710、715 可包括本文所描述的一或多個處理器核心，且可以是處理器 600 的某些版本。

記憶體 740 例如是動態隨機存取記憶體(DRAM)、相變記憶體(PCM)、或該兩種的組合。關於至少一實施例，控制器集線器 720 經由多點匯流排與處理器 710、715 通訊，諸如前端匯流排(FSB)、點對點介面，諸如快速通道互連(QPI)，或類似的連接 795。

在一實施例中，協同處理器 745 係特殊用途處理器，諸如，例如，高產出量 MIC 處理器、網路或通訊處理器、壓縮引擎、圖形處理器、GPGPU、內嵌式處理器、或類似物。在一實施例中，控制器集線器 720 可包括整合式圖形加速器。

就度量一系列優點而論，實體資源 710、715 之間有各種差異，包括架構、微架構、熱、功耗特性等。

在一實施例中，處理器 710 執行的指令用來控制通用類型之資料處理操作。協同處理器指令可內嵌在該等指令內。處理器 710 識別這些協同處理器指令係應由附屬之協同處理器 745 來執行的類型。於是，處理器 710 在協同處理器匯流排或其它互連上將這些協同處理器指令(或代表協同處理器指令的控制信號)發送給協同處理器 745。協同處理器 745 接受並執行所接收的協同處理器指令。

現參閱圖 8，圖中顯示的方塊圖係按照本發明實施例之第一更特定的例示性系統 800。如圖 8 所示，多處理器系統 800 係點對點互連系統，且包括經由點對點互連 850 耦接的第一處理器 870 與第二處理器 880。處理器 870 與 880 每一個皆是處理器 600 的某版本。在本發明的一實施

例中，處理器 870 與 880 分別是處理器 710 與 715，而協同處理器 838 係協同處理器 745。在另一實施例中，處理器 870 與 880 分別是處理器 710 與協同處理器 745。

所顯示的處理器 870 與 880 分別包括整合式記憶體控制器(IMC)單元 872 與 882。處理器 870 也包括做為其一部分的匯流排控制器單元點對點(P-P)介面 876 與 878；同樣地，第二處理器 880 也包括 P-P 介面 886 與 888。處理器 870、880 可使用 P-P 介面電路 878、888 經由點對點(P-P)介面 850 互換資訊。如圖 8 所示，IMC 872 與 882 將處理器分別耦接至各自的記憶體，即記憶體 832 與記憶體 834，其可以是主記憶體的一部分，局部地附接到各自的處理器。

處理器 870 與 880 每一個都可使用點對點介面電路 876、894、886、898 經由個別的 P-P 介面 852、854 與晶片組 890 交換資訊。晶片組 890 可經由高性能介面 839 與協同處理器 838 隨意地交換資訊。在一實施例中，協同處理器 838 係特殊用途處理器，諸如，例如，高產出 MIC 處理器、網路或通訊處理器、壓縮引擎、圖形處理器、GPGPU、內嵌式處理器、或類似物。

共享快取(未顯示)可包括在處理器內或兩處理器的外部，此外還經由點對點互連與處理器連接，使得如果處理器被置於低電力模式，其中之一或兩個處理器的本地快取資訊可儲存在共享快取中。

晶片組 890 可經由介面 896 耦接至第一匯流排 816。

在一實施例中，第一匯流排 816 可以是周邊組件互連 (PCI)匯流排，或諸如 PCI Express 匯流排或其它第三代 I/O 互連匯流排的匯流排，然而本發明的範圍不限於此。

如圖 8 所示，各種的 I/O 裝置 814 連同將第一匯流排 816 耦接至第二匯流排 820 的匯流排橋接器 818 可耦接至第一匯流排 816。在一實施例中，一或多個附加的處理器 815，諸如協同處理器、高產出 MIC 處理器、GPGPU、加速度(諸如，例如，圖形加速器或數位信號處理器(DSP)單元)、現場可程式閘極陣列、或任何其它的處理器都可耦接至第一匯流排 816。在一實施例中，第二匯流排 820 可以是低接腳數(LPC)匯流排。各種裝置可耦接至第二匯流排 820，在一實施例中，例如，包括鍵盤及/或滑鼠 822、通訊裝置 827 與儲存單元 828，諸如磁碟機或可包括指令/碼與資料 830 的其它大量儲存裝置。此外，音頻 I/O 824 也可耦接至第二匯流排 820。須注意，也可以是其它的架構。例如，系統可實施多點連接匯流排或其它此類的架構來取代圖 8 的點對點架構。

現參閱圖 9，圖中顯示的方塊圖按照本發明之實施例之第二更特定的例示性系統 900。圖 8 與 9 中相似的元件使用相似的參考數字，且為了避免模糊了圖 9 的其它態樣，圖 9 省略了圖 8 中的某些態樣。

圖 9 說明的處理器 870、880 分別包括整合式記憶體與 I/O 控制邏輯(“CL”)872 與 882。因此，CL 872、882 包括整合式記憶體控制器單元且包括 I/O 控制邏輯。圖 9 說

明不僅記憶體 832、834 耦接至 CL 872、882，且 I/O 裝置 914 也耦接至控制邏輯 872、882。傳統 I/O 裝置 915 耦接至晶片組 890。

現參閱圖 10，圖中顯示按照本發明實施例之 SoC 1000 的方塊圖。與圖 6 中類似的元件使用相似的參考數字。此外，虛線方塊係在更先進之 SoC 上選用的特徵。在圖 10 中，互連單元 1002 耦接至：包括一組一或多個核心 202A-N 與共享快取單元 606 的應用處理器 1010；系統代理單元 610；匯流排控制器單元 616；整合式記憶體控制器單元 614；一組或一或多個協同處理器 1020，其可包括整合式圖形邏輯、影像處理器、音頻處理器、及視訊處理器；靜態隨機存取記憶體 (SRAM) 單元 1030；直接記憶體存取 (DMA) 單元 1032；及用於耦接一或多台外部顯示器的顯示單元 1040。在一實施例中，協同處理器 1020 包括特殊用途處理器，諸如，例如，網路或通訊處理器、壓縮引擎、GPGPU、高產出 MIC 處理器、內嵌式處理器、或類似物。

本文所揭示之機制的實施例可在硬體、軟體、韌體、或這些實施方法的組合中實施。本發明的實施例可實施為電腦程式或程式碼，可在包含至少一個處理器、儲存系統 (包括揮發性與非揮發性記憶體及/或儲存元件)、至少一個輸入裝置、及至少一個輸出裝置的可程式系統上執行。

程式碼，諸如圖 8 中說明的碼 830，可施加於輸入指令以實施本文所描述的功能，並產生輸出資訊。輸出資訊

可按習知的方式施加於一或多個輸出裝置。基於此應用之目的，處理系統包括任何具有處理器的系統，諸如，例如，數位信號處理器(DSP)、微控制器、特定應用積體電路(ASIC)、或微處理器。

程式碼可在高階程序或物件導向程式語言中實施，以與處理系統通訊。如有需要，程式碼也可在組合或機器語言中實施。事實上，本文所描述的機制不限於任何特定程式語言的範圍。無論如何，該語言係可被編譯或解譯的語言。

至少一實施例的一或多個態樣可藉由儲存在機器可讀取媒體中的代表指令來實施，其代表處理器內的各種邏輯，當這些指令被機器讀取時，致使機器製造用以實施本文所描述之技術的邏輯。這些表示法是習知的“IP 核心”，可儲存在有形的機器可讀取媒體上，並供應給各不同的客戶或製造工廠，用以載入到實際製造邏輯或處理器的製造機器內。

此機器可讀取儲存媒體可包括但不限於：藉由機器或裝置所製造或形成之物件的非暫時性有形配置，包括諸如硬式磁碟機的儲存媒體；任何其它類型的碟片，包括軟碟、光碟、光碟唯讀記憶體(CD-ROM)、可覆寫光碟(CD-RW)、及磁光碟；半導體裝置，諸如唯讀記憶體(ROM)、隨機存取記憶體(RAM)，諸如動態隨機存取記憶體(DRAM)、靜態隨機存取記憶體(SRAM)、可抹除可程式唯讀記憶體(EPROM)、快閃記憶體、電可抹除可程式唯讀記

憶體 (EEPROM)、相變記憶體 (PCM)、磁卡或光學卡；或適合儲存電子指令之任何其它類型的媒體。

因此，本發明之實施例也包括非暫時性有形機器可讀取媒體，其內包含指令或包含設計資料，諸如硬體描述語言 (Hardware Description Language; HDL)，其定義本文所描述的結構、電路、裝置、處理器、及/或系統特徵。這些實施例也稱為程式產品。

在某些情況中，可用指令轉換器將指令從源指令集轉換到目標指令集。例如，指令轉換器可轉譯(例如，使用靜態二進制轉譯、包括動態編譯的動態二進制轉譯)、變形、仿真、或以其它方式將指令轉換成能被核心處理的一或多種其它指令。指令轉換器可在軟體、硬體、韌體、或它們的組合中實施。指令轉換器可在處理器上、處理器外、或部分在處理器上而部分在處理器外。

圖 11 的方塊圖係按照本發明之實施例，使用軟體指令轉換器將源指令集中的二進制指令轉換到目標指令集中之二進制指令之對照。在說明的實施例中，指令轉換器係軟體指令轉換器，然而，可替代地在軟體、韌體、硬體、或以上這些的各種組合中實施指令轉換器。圖 11 顯示，可使用 x86 編譯器 1104 編譯高階語言 1102 的程式，以產生可被具有至少一個 x86 指令集核心 1116 之處理器原生執行的 x86 二進制碼 1106。具有至少一個 x86 指令集核心 1116 的處理器代表可實質地實施與具有至少一個 x86 指令集核心之英特爾處理器相同功能的任何處理器，其為

了獲致與具有至少一個 x86 指令集核心之英特爾處理器實質相同的結果，藉由相容地執行或以其它方式處理(1)英特爾 x86 指令集核心之指令集的實質部分，或(2)以在具有至少一個 x86 指令集核心之英特爾處理器上運行為目標之應用程式或其它軟體的目的碼版本。x86 編譯器 1104 代表一編譯器，無論有或無額外的鏈結處理，其可操作以產生可在具有至少一個 x86 指令集核心 1116 之英特爾處理器上執行的 x86 二進制碼 1106(例如目標碼)。同樣地，圖 11 顯示高階語言 1102 的程式可使用替代的指令集編譯器 1108 來編譯，以產生可被不具有至少一個 x86 指令集核心之處理器 1114(例如，具有執行 MIPS Technologies of 森尼韋爾，CA 之 MIPS 指令集及/或 ARM Holdings of 森尼韋爾，CA 之 ARM 指令集的核心之處理器)原生執行之替代的指令集二進制碼 1110。指令轉換器 1112 被用來將 x86 二進制碼 1106 轉換成可被無 x86 指令集核心之處理器 1114 原生執行的碼。此經轉換的碼不太可能與替代的指令集二進制碼 1110 相同，原因是此指令轉換器的能力很難做到；不過，經轉換的碼將可完成一般的操作，並從替代的指令集組成指令。因此，指令轉換器 1112 代表軟體、硬體、韌體、或它們的組合，透過仿真、模擬、或任何其它處理，允許不具有 x86 指令集處理器或核心的處理器或其它電子裝置執行 x86 二進制碼 1106。

用於實施的方法及設備

最佳化指令流的檢查

IEEE 正確的捨入功能諸如除法與平方根可用兩種方法來實施：使用基於基數的硬體除法器以原生地提供功能性，或者，實施軟體指令/微碼(*u*code)序列，典型的牛頓-瑞佛生演算法之形式/形變，用以迭代種子以得到最終的結果。

軟體序列有一問題，雖然演算法能非常有效率地處理大部分的輸入，但對於某些隅點情況(*corner case*)則需要額外的處理，隅點情況導致的序列會慢到一個數量級(例如，當處理非正常的輸入時)。此外，當輸入具有無價值的結果時(諸如被零除)，即需要最終的後處理以修補結果，其增加該序列之關鍵性的潛時。

在一實施例中，指令序列假設最佳的情況，檢查出預設序列的關鍵路徑，且當輸入中有隅點情況時，其跳越至替代的或“慢”序列。本發明的實施例包括“檢查”類型的指令或微操作(*uop*)，其比主預設計算先計算分離之路徑中的此條件性跳越。在某些實施例中，指令檢查操作也為無價值的輸入提供預設的答案。雖然以下描述的實施例專注於除法與平方根(*sqrt*)的實施，但本發明的基本原理並不限於任何特定類型的數學功能。更特定地說，一實施例係基於封裝之平方根/除法的 *AVX-512* 實施(例如，使用預測/遮罩)，但觀念可延伸到任何指令集架構。

一實施例的“檢查”指令/*uop* 根據輸入決定是以具有預設/快速流程的演算法處理，或是替代地跳越到慢流程，

留下布林向量進入遮罩。此外，一實施例產生預先計算的異常旗標(無效的、非正常的、及被零除)，且如果它們未被遮罩，即放棄執行。最後，本發明的一實施例為無價值的輸入(諸如被無限大除)預計算預設的答案。

如圖 12 之說明，可在其上實施本發明之實施例的例示性處理器 1255 包括處理器管線之解碼級 1230 內用於解碼指令檢查指令的指令檢查解碼邏輯 1231，以及管線之執行級 1240 內用於執行指令檢查指令的指令檢查執行邏輯 1241。在一實施例中，指令檢查解碼邏輯 1231 包含用於將指令檢查指令解碼成複數個微操作的硬體，接著，指令檢查執行邏輯 1241 執行這些微操作。不過，本發明的基本原理並不限於微碼的處理器。如上所述，在一實施例中，該等指令包括除法檢查指令(DIVCHK())與平方根檢查(SQRTCHK())。不過，本發明的基本原理並不限於這些特定的實施。

圖 12 中所示的其它處理器組件包括一組通用暫存器(GPR)1205、一組向量暫存器 1206、一組遮罩暫存器 1207、及一組控制暫存器 1208。在圖 12 中雖未明示，但可經由獨立的控制暫存器匯流排(CRBus)介面存取控制暫存器 1208。在一實施例中，遮罩暫存器 1207 包括 8 個 64 位元的運算元遮罩暫存器，用來對儲存在向量暫存器 1206 中的值實施位元遮罩操作(例如，按前文描述的遮罩暫存器 k0-k7 來實施)。不過，本發明的基本原理並不限於任何特定類型/大小的遮罩暫存器。在一實施例中，多

個向量資料元素被封裝到每一個向量暫存器 1206 中，其可具有 512 位元的寬度，用以儲存 2 個 256 位元的值、4 個 128 位元的值、8 個 64 位元的值、16 個 32 位元的值等。不過，本發明的基本原理並不限於任何特定大小/類型的向量資料。

為簡化緣故，圖 12 中說明單一個處理器核心(“核心 0”)的細節。不過，須瞭解，圖 12 中所顯示的每一個核心可具有與核心 0 相同的邏輯組。例如，每一個核心可包括專用的 1 階(L1)快取 1212 與 2 階(L2)快取 1211，用於按照特定的快取管理政策來快取指令與資料。L1 快取 1212 包括獨立的指令快取 1220 用於儲存指令，與獨立的資料快取 1221 用於儲存資料。儲存在各不同處理器快取內的指令與資料以快取線的粒度被管理，其可以是固定的大小(例如，長度為 64、128、512 位元組)。此例示性實施例的每一個核心具有用於從主記憶體 1200 及/或共享的 3 階(L3)快取 1216 中提取指令的指令提取單元 1210；用於解碼該等指令(例如，將程式指令解碼成微操作或“uops”)的解碼單元 1220；用於執行該等指令的執行單元 1240；及用於止用該等指令並將結果寫回的寫回單元 1250。

指令提取單元 1210 包括了各種習知的組件，包括有下一指令指標符 1203，用於儲存要被從記憶體 1200(或其中一個快取)中提取之下一個指令的位址；指令轉譯後備緩衝區(ITLB)1204，用於儲存最近使用之虛擬到實體之指令位址的映射，以提升位址轉譯的速度；分支預測單元

1202，用於推測地預測指令分支位址；以及分支目標緩衝區(BTB)1201，用於儲存分支位址與目標位址。指令一旦被提取，接著被串流到指令管線的其餘級，包括解碼單元 1230、執行單元 1240、及寫回單元 1250。這些單元每一個的結構與功能係為所屬技術領域中具有通常知識者相當瞭解的，且在此不再詳細描述，以避免模糊了關於本發明之不同實施例的態樣。

● 圖 13 說明指令檢查邏輯 1300，其產生向量輸出與遮罩輸出，用以根據與當下指令相關聯的複數個輸入值 1301-1303(例如，用於除法或平方根指令的輸入)來選擇特定的指令序列 1311(例如，預設/快速序列或替代/慢序列)。以下提供用於除法及平方根指令之向量與遮罩輸出的各種特定例(見圖 14(除法)與圖 15(平方根))。此外，在某些實施例中，指令檢查邏輯 1300 可為特定類型之輸入(例如，無價值的輸入，諸如被無限大除)產生結果 1312，及可產生可供後續指令序列 1311 使用的異常旗標 1313。舉例來說，異常旗標可包括浮點異常旗標，諸如被零除(ZE)、無效的運算(IE)、及非正常化的運算元(DE)。

如以上之討論，指令檢查邏輯 1300 可包括解碼級 1230 內的邏輯(例如，指令檢查解碼邏輯 1231)與執行級 1240(例如，指令檢查執行邏輯 1241)，前者將指令解碼成複數個微操作，後者執行該等微操作以實施本文所描述的技術。不過，須注意，本發明的基本原理並不限於微碼的處理器架構。

現將描述除法檢查 (DIVCHK) 指令的一實施例，其對與除法指令相關聯的輸入與其它變數實施檢查，並回應地選擇適當/有效率的指令序列來實施除法運算。在一實施例中，用於除法的微碼流策略如下：

```

division(x,y)
{
    start_fast_sequence(x,y);
    if(DIVCHK())
        jump slow_flow;
    else
        finish_fast_flow(x,y)
        return res;
slow_flow:
    do_slow_flow(x,y)
    return res;
}

```

在以上的碼中，DIVCHK 評估輸入(x, y)，以決定是否跳越至替代的“slow_flow”指令序列，或完成正常的“finish_fast_flow”序列。在一實施例中，使用以下的 DIVCHK 規格。

在一實施例中，使用微操作 $\text{UOP}_{\{\text{DIVCHK}\}\{\text{PS},\text{PD}\}}$ ，其中，第一參數 (Param 1) 檢查除法輸入，及第二參數 (Param 2) 指示運算是否包括單精度或雙精度輸入。UOP 可使用第一源 (srcA) 做為分子，及第二源 (scrB) 做為分母，為除法序列檢查輸入的特性，並返回 3 個不同的資訊：

1. 具有特殊情況輸入預設回應的向量輸出(例如，被零除返回 INF)。在一實施例中，如果需要產生實際的輸

出，其使用慢速路徑演算法輸出特殊的指示(例如，+1.0)。

2. 遮罩輸出信令那些可使用快速路徑演算法處理的元素。在一實施例中，零的遮罩意指結果為無價值或需要經由慢流程序列來發生計算(如由諸如+1.0 之特殊指示所指定)。

3. 產生預先計算的異常旗標。例如，此可包括浮點異常旗標，諸如被零除(ZE)、無效的運算(IE)、及非正常化的運算元(DE)。

圖 14 顯示的表綜述輸出係兩個不同輸入類型(X 係 INPUT1 而 Y 係 INPUT2)之函數的 3 個輸出。在此表中，X 輸入顯示於列中，而 Y 輸入顯示於行中。所得到的輸出係按(kdst, vdst, flags)配置其中，“kdst”包含遮罩輸出、“vdst”包含向量輸出、“flags”包含異常旗標。輸入“denorm”代表非正常的運算元、“normal”代表正常的運算元、及“Inf”代表無限大。NaN 代表非數字，QNaN 代表安靜的 NaN，SNaN 代表發信的 NaN。QNaN 係具有最有效部分位元設定的 NaN；SNaN 係具有最有效部分位元清除的 NaN。

在一實施例中，在兩輸入皆正常數字(在表中“見下文”來指示)的情況係按如下處置：

```

xExp = x的指數偏置 ; // 指數的記憶體形式
yExp = y的指數偏置 ; // 指數的記憶體形式
eDiff = xExp - yExp;
if (    (eDiff ≥ bias + 1) || // 商數上溢
        (eDiff ≤ (1 - bias)) || // 商數下溢
        (yExp ≥ (2*bias - 1)) || // 倒數下溢
        (xExp ≤ 2*precision) // 餘值下溢 (*)
    ) {
    return (0,+1.0,None);
} else
    return (1,+0.0,None);
}

```

因此，以上係用於商數上溢/下溢、倒數下溢、與餘值下溢的碼測試。如果檢測到這些情況的任一者，則向量輸出為+1.0；若否，則向量輸出為+0.0。關於餘值測試，來自 ISA 之輸入與輸出的非正常在內部計算中被清空。

在一實施例中，DIVCHK 操作採以下之形式：

```

// UOP_DIVCHK_PS (.INPUT1(src_a), .INPUT2(src_b))
for(n=0; n<2; n+=1)
{
    i = n*32
    {MASK_OUTPUT[n], VECT_OUTPUT[i+31:i]}
        = divchk(INPUT1[i+31:i], INPUT2[i+31:i]) // (x, y)
}

```

為非正常(如果 DAZ=0)產生異常，為單精度與雙精度浮點資料元素兩者產生無效與被零除。

現描述平方根檢查(SQRTCHK)指令的一實施例，其對與平方根指令相關聯的輸入與其它變數實施檢查，並回應地選擇適當/有效率的指令序列來實施平方根運算。在一

實施例中，用於平方根的微碼流策略如下：

```

sqrt(x)
{
    start_fast_sequence(x);
    if(SQRTCHK())
        jump slow_flow;
    else
        finish_fast_flow(x)
        return res;
Slow_flow:
    do_slow_flow(x)
    return res;
}

```

在以上的碼中，SQRTCHK 評估輸入 x ，以決定是否跳越至替代的“slow_flow”指令序列，或完成正常的“finish_fast_flow”序列。在一實施例中，使用以下的 SQRTCHK 規格。

UOP_{SQRTCHK}{PS,PD} 係在一實施例中使用的微操作，在此，第一參數 {SQRTCHK} 檢查平方根指令輸入，及第二參數 {PS,PD} 指示運算是否包括單精度或雙精度輸入。UOP 檢查平方根序列輸入的特性，並返回 3 個不同的資訊：

1. 具有特殊情況輸入預設回應的向量輸出(例如，負數的平方根返回 NaN)。在一實施例中，如果需要產生實際的輸出，其使用快速路徑演算法輸出特殊的指示(例如，+1.0)。

2. 遮罩輸出信令那些必須使用慢速路徑演算法處理

的元素。

3. 產生預先計算的異常旗標。例如，此可包括浮點異常旗標，諸如無效的運算(IE)、及非正常化的運算元(DE)。

圖 15 顯示的表中列出 x 之各種不同類型的輸入(第一行)與所得到之按(kdst, vdst, flags)配置的輸出，其中，“kdst”包含遮罩輸出、“vdst”包含向量輸出、“flags”包含異常旗標。輸入包括 -norm (包括-無限大)、+/-非正常化的運算元、+/-0、+無限大、qNaN、與 sNaN。限制偏置指數 $(x) \leq 2 * \text{精度}$ (在表中以條件(a)來識別)係意欲防止 $d = x - s * s$ 之計算中的下溢。

```
// UOP_SQRTCHK_PS (.INPUT1(src_b))
for(n=0; n<2; n+=1)
{
    i = n*32
    {MASK_OUTPUT[n], VECT_OUTPUT[i+31:i]}
    = sqrtchk(INPUT1[i+31:i]);
}
```

為非正常(如果 DAZ=0)產生異常，及為單精度與雙精度浮點資料元素產生無效。

圖 16 說明按照本發明一實施例的方法。該方法在上述架構的環境中實施，但並不限於任何特定架構。

在 1601，從數學函數得到輸入。例如，如果數學函數係除法，則輸入包含分子(x)與分母(y)。如果數學函數係平方根，則輸入包含需要平方根的數字(x)。在 1602，按照數學函數來評估輸入的各不同特性，以產生用以指示

是否需跳越到替代/慢指令序列的輸出。例如，如以上的討論，該輸出可包括具有特殊情況輸入預設回應的向量輸出(例如，被零除則返回 INF)、信令那些能使用快速路徑演算法處理之元素的遮罩輸出(例如，在遮罩 K2 中)、及預先計算的異常旗標。此外，在 1602 做出關於是否至少某些數學函數之結果將會無價值的決定。

在 1603，如果決定至少某些結果無價值，則在 1604，無價值的結果被計算並被送回。在某些例中，除了無價值的結果不需要其它結果。如果需要額外的結果，在 1605，根據 1602 的結果，做出關於是否要跳越到替代/慢指令序列的決定。如果在 1604 的結果僅只有無價值的結果，則無替代/慢指令序列被許可。如果替代/慢指令序列被許可，則在 1606 執行替代/慢指令序列。如果在 1605 的決定係替代/慢指令序列不被許可，則在 1607 執行正常/快速指令序列(例如，使用正常的運算元值來計算除法、平方根、或其它數學函數的結果)。

在以上的說明書中，已參考某特定的例示性實施例描述了本發明的實施例。不過，很明顯，可對其做各種的修改與改變，不會偏離如所附申請專利範圍中所陳述之本發明較廣的精神與範圍。因此，說明與圖式被視為說明而無限制之意。

前文已描述了本發明的實施例可包括各不同步驟。該等步驟可具體化成機器可執行的指令，其可用來致使通用或特殊用途處理器實施該等步驟。或者，這些步驟可藉由

特定的硬體組件來實施，其包含用於實施該等步驟的硬線邏輯，或藉由程式化的電腦組件及客制的硬體組件的任何組件來實施。

如本文之描述，指令可意指硬體的特定組構，諸如被組構成用以實施特定操作或具有預定功能性的特殊用途積體電路(ASIC)，或儲存在具體化成非暫時性電腦可讀取媒體之記憶體中的軟體指令。因此，圖中所顯示的技術可使用在一或多個電子裝置(例如，終端站、網路元件等)上儲存與執行的碼與資料來實施。此等電子裝置使用電腦機器可讀取媒體來儲存與通訊(內部及/或在網路上與其它電子裝置)碼與資料，諸如非暫時性電腦機器可讀取媒體(例如，磁碟；光碟；隨機存取記憶體；唯讀記憶體；快閃記憶體裝置；相變記憶體)，及暫時性電腦機器可讀取通訊媒體(例如，電、光、聲、或傳播信號的其它形式-諸如載波、紅外線信號、數位信號等)。此外，典型上，此等電子裝置包括一組一或多個處理器耦接至一或多個其它組件，諸如一或多個儲存裝置(非暫時性機器可讀取儲存媒體)、使用者輸入/輸出裝置(例如，鍵盤、觸控螢幕、及/或顯示器)、及網路連接。處理器組與其它組件之耦接典型上是經由一或多個匯流排與橋接器(也稱為匯流排控制器)。儲存裝置與用來負載網路流量的信號分別代表一或多個機器可讀取儲存媒體與機器可讀取通訊媒體。因此，典型上，給定之電子裝置的儲存裝置儲存碼及/或資料供在該電子裝置的一組一或多個處理器上執行。當然，本發

明之實施例的一或多個部分可使用軟體、韌體、及/或硬體的不同組合來實施。基於解釋之目的，為了徹底瞭解本發明，此詳細描述的全文陳述了諸多的特定細節。不過，熟悉此領域之技藝者將可明瞭，實行本發明可不需其中的一些特定細節。為了避免模糊了本發明的標的，在某些例中未詳細描述熟知的結構與功能。因此，本發明的範圍與精神應由就以下的申請專利範圍來判斷。

【符號說明】

- 100：通用向量親和指令格式
- 105：無記憶體存取指令範本
- 110：無記憶體存取全捨入控制式運算指令範本
- 115：無記憶體存取資料轉換式運算
- 120：記憶體存取指令範本
- 125：記憶體存取暫時性指令範本
- 112：無記憶體存取寫入遮罩控制部分捨入控制式運算指令範本
- 130：記憶體存取非暫時性指令範本
- 117：無記憶體存取寫入遮罩控制 `vsize` 式運算指令範本
- 127：記憶體存取寫入遮罩控制指令範本
- 140：格式欄位
- 142：基本運算欄位
- 144：暫存器索引欄位
- 146：修改符欄位
- 150：擴增運算欄位

- 152：阿爾發欄位
- 154：貝它欄位
- 168：類型欄位
- 160：縮放欄位
- 162A：位移欄位
- 162B：位移因數欄位
- 164：資料元素寬度欄位
- 170：寫入遮罩欄位
- 172：立即欄位
- 174：完整運算碼欄位
- 152A：RS 欄位
- 152A.1：捨入
- 152A.2：資料轉換
- 154A：捨入控制欄位
- 156：浮點異常欄位
- 158：捨入運算控制欄位
- 154B：資料轉換欄位
- 154C：資料調處欄位
- 152B：逐出提示欄位
- 152B.1：暫時性
- 152B.2：非暫時性
- 125：記憶體存取暫時性指令範本
- 130：記憶體存取非暫時性指令範本
- 159A：捨入運算控制欄位

159B：向量長度欄位

157A：RL 欄位

157A.1：捨入

157A.2：向量長度

157B：廣播欄位

200：特定向量親和指令格式

202：EVEX 前置

205：REX 欄位

210：REX'欄位

215：運算碼映射欄位

225：前置編碼欄位

230：真實運算碼欄位

240：MOD：R/M 欄位

242：MOD 欄位

244：Reg 欄位

246：R/M 欄位

300：暫存器架構

310：向量暫存器

315：寫入遮罩暫存器

325：通用暫存器

345：純量浮點堆疊暫存器檔案

350：MMX 封裝整數平面暫存器檔案

400：處理器管線

402：提取級

- 404：長度解碼級
- 406：解碼級
- 408：配置級
- 410：更名級
- 412：排程級
- 414：暫存器讀取/記憶體讀取級
- 416：執行級
- 418：寫回/記憶體寫入級
- 422：異常處置級
- 424：提交級
- 430：前端單元
- 450：執行引擎單元
- 470：記憶體單元
- 490：處理器核心
- 432：分支預測單元
- 434：指令快取單元
- 436：指令轉譯後備緩衝區
- 438：指令提取單元
- 440：解碼單元
- 450：執行引擎單元
- 452：更名/配置單元
- 454：止用單元
- 456：排程器單元
- 458：實體暫存器檔案單元

- 460：執行叢集
- 462：執行單元
- 464：記憶體存取單元
- 470：記憶體單元
- 472：資料 TLB 單元
- 474：資料快取單元
- 476：2 階(L2)快取單元
- 500：指令解碼器
- 502：晶粒上的互連網路
- 504：2 階(L2)快取單元
- 506：1 階(L1)快取
- 508：純量單元
- 510：向量單元
- 512：純量暫存器
- 514：向量暫存器
- 520：攪拌單元
- 522：數值轉換單元
- 524：複製單元
- 526：寫入遮罩暫存器
- 528：向量處理單元
- 600：處理器
- 602：核心
- 604：快取單元
- 606：共用快取單元

- 608：特殊用途邏輯
- 610：系統代理
- 612：環狀式互連單元
- 614：整合式記憶體控制器單元
- 616：匯流排控制器單元
- 700：系統
- 710：處理器
- 715：處理器
- 720：控制器集線器
- 740：記憶體
- 745：協同處理器
- 750：輸入/輸出集線器
- 760：輸入/輸出裝置
- 790：圖形記憶體控制器集線器
- 800：多處理器系統
- 850：點對點互連
- 870：第一處理器
- 880：第二處理器
- 838：協同處理器
- 876：匯流排控制器單元點對點介面
- 878：匯流排控制器單元點對點介面
- 886：點對點(P-P)介面
- 888：點對點(P-P)介面
- 850：點對點(P-P)介面

- 832：記憶體
- 834：記憶體
- 872：整合式記憶體控制器
- 882：整合式記憶體控制器
- 894：點對點介面電路
- 898：點對點介面電路
- 852：點對點(P-P)介面
- 854：點對點(P-P)介面
- 890：晶片組
- 839：高性能介面
- 896：介面
- 816：第一匯流排
- 814：輸入/輸出裝置
- 820：第二匯流排
- 818：匯流排橋接器
- 815：附加的處理器
- 824：音頻輸入/輸出
- 822：鍵盤及/或滑鼠
- 827：通訊裝置
- 828：儲存單元
- 830：指令/碼與資料
- 900：系統
- 914：輸入/輸出裝置
- 915：傳統輸入/輸出裝置

872：整合式記憶體與 I/O 控制邏輯

882：整合式記憶體與 I/O 控制邏輯

1000：系統單晶片

1010：應用處理器

1020：協同處理器

1030：靜態隨機存取記憶體單元

1032：直接記憶體存取單元

1040：顯示單元

1002：互連單元

1102：高階語言

1104：x86 編譯器

1106：x86 二進制碼

1108：替代的指令集編譯器

1110：替代的指令集二進制碼

1112：指令轉換器

1114：不具有 x86 指令集核心的處理器

1116：具有 x86 指令集核心的處理器

1200：主記憶體

1201：分支目標緩衝區

1202：分支預測單元

1203：下一指令指標符

1204：指令轉譯後備緩衝區

1205：通用暫存器

1206：向量暫存器

- 1207 : 遮罩暫存器
- 1208 : 控制暫存器
- 1210 : 指令提取單元
- 1211 : 2 階 (L2) 快取
- 1212 : 1 階 (L1) 快取
- 1216 : 3 階 (L3) 快取
- 1220 : 解碼單元
- 1221 : 資料快取
- 1230 : 解碼級
- 1231 : 指令檢查解碼邏輯
- 1240 : 執行級
- 1241 : 指令檢查執行邏輯
- 1250 : 寫回單元
- 1255 : 處理器
- 1301 : 輸入值
- 1302 : 輸入值
- 1303 : 輸入值
- 1300 : 指令檢查邏輯
- 1311 : 指令序列
- 1312 : 結果
- 1313 : 異常旗標

申請專利範圍

1.一種處理器，包含：

算術邏輯單元(arithmetic logic unit; ALU)，用以使用一或多個源運算元來執行複數個數學指令；

指令檢查邏輯，用以評估用於當下數學指令的該等源運算元，並依據該評估用以決定是執行包括藉由該 ALU 來執行該當下數學指令之運算的預設序列，或是跳越到運算的替代序列，運算的該替代序列係適合用以提供結果用於具有比運算之該預設序列更有效率的特定類型之源運算元之數學指令。

2.如申請專利範圍第 1 項之處理器，其中，該當下的數學指令包含具有藉由該源運算元所指定之分子與分母的除法指令，且其中，該指令檢查邏輯回應該分子或分母係非正常化之運算元、係等於無限大、係非數值(not-a-number; NaN)的運算元、及/或導致被零除而致使到運算之該替代序列的跳越。

3.如申請專利範圍第 2 項之處理器，其中，該指令檢查邏輯用以在該處理器中設定包括被零除(ZE)、無效之運算(IE)、及/或非正常化之運算元(DE)的一或多個異常旗標。

4.如申請專利範圍第 1 項之處理器，其中，該當下的指令包含具有在其上實施平方根運算之源運算元值的平方根指令，其中，該指令檢查邏輯回應該源運算元係負數、係非正常化的運算元、係等於無限大、及/或係非數值

(not-a-number; NaN)的運算元而致使往運算之該替代序列的跳越。

5.如申請專利範圍第 1 項之處理器，其中，該當下的指令包含具有在其上實施平方根運算之源運算元值(x)的平方根指令，其中，該指令檢查邏輯回應使用 x 實施比較而致使到運算之該替代序列的跳越。

6.如申請專利範圍第 1 項之處理器，其中，當實施該源運算元的該評估時，該指令檢查邏輯產生用以指示執行該預設序列或該替代序列的向量輸出。

7.如申請專利範圍第 6 項之處理器，其中，該指令檢查邏輯進一步產生遮罩輸出，用以指令要以該替代序列處理的元素。

8.如申請專利範圍第 7 項之處理器，其中，該指令檢查邏輯進一步產生預計算異常旗標輸出，用以指示起因於對該源運算元實施該數學運算的一或多個異常。

9.如申請專利範圍第 1 項之處理器，其中，運算的該預設序列包含指令或微操作的預設序列，且其中，運算的該替代序列包含指令或微操作的替代序列。

10.一種方法，包含：

得到用於當下之數學指令的一或多個源運算元；

評估用於該當下數學指令的該等源運算元；

依據該評估，決定是執行包括執行該當下數學指令之運算的預設序列，或是跳越到運算的替代序列，運算的該替代序列係適合用以提供結果用於具有比運算之該預設序

列更有效率的特定類型之源運算元之數學指令。

11.如申請專利範圍第 10 項之方法，其中，該當下的數學指令包含具有藉由該源運算元所指定之分子與分母的除法指令，且其中，該方法包含回應該分子或分母係非正常化之運算元、係等於無限大、係非數值(NaN)的運算元、及/或導致被零除而跳越到運算的該替代序列。

12.如申請專利範圍第 11 項之方法，進一步包含在該方法中設定一或多個異常旗標，包括有被零除(ZE)、無效之運算(IE)、及/或非正常化之運算元(DE)。

13.如申請專利範圍第 10 項之方法，其中，該當下的指令包含具有在其上實施平方根運算之源運算元值的平方根指令，該方法進一步包含回應該源運算元係負數、係非正常化的運算元、係等於無限大、及/或係非數值(NaN)的運算元而跳越到運算的該替代序列。

14.如申請專利範圍第 10 項之方法，其中，該當下的指令包含具有在其上實施平方根運算之源運算元值(x)的平方根指令，該方法進一步包含回應使用 x 實施比較而跳越到運算的該替代序列。

15.如申請專利範圍第 10 項之方法，進一步包含當實施該源運算元的該評估時，產生用以指示執行該預設序列或該替代序列的向量輸出。

16.如申請專利範圍第 15 項之方法，進一步包含產生遮罩輸出，用以信令要以該替代序列處理的元素。

17.如申請專利範圍第 16 項之方法，進一步包含產生

預計算異常旗標輸出，用以指示起因於對該源運算元實施該數學運算的一或多個異常。

18.如申請專利範圍第 10 項之方法，其中，運算的該預設序列包含指令或微操作的預設序列，且其中，運算的該替代序列包含指令或微操作的替代序列。

19.一種系統，包含：

記憶體，用以儲存指令與資料，包括數學指令與圖形指令；

複數個核心，用以執行該數學指令及處理該資料；

圖形處理器單元，用以回應該圖形指令實施圖形運算；

網路介面，用以在網路上接收及傳送資料；

介面，用以接收來自滑鼠或游標控制裝置的使用者輸入；以及

算術邏輯單元(ALU)，用以使用一或多個源運算元來執行複數個數學指令；

指令檢查邏輯，用以評估用於當下數學指令的該等源運算元，並依據該評估用以決定是執行包括藉由該 ALU 來執行該當下數學指令之運算的預設序列，或是跳越到運算的替代序列，運算的該替代序列係適合用以提供結果用於具有比運算之該預設序列更有效率的特定類型之源運算元之數學指令。

20.如申請專利範圍第 19 項之系統，其中，該當下的數學指令包含具有藉由該源運算元所指定之分子與分母的

除法指令，且其中，該指令檢查邏輯回應該分子或分母係非正常化之運算元、係等於無限大、係非數值(NaN)的運算元、及/或導致被零除而致使到運算之該替代序列的跳越。

21.如申請專利範圍第 20 項之系統，其中，該指令檢查邏輯用以在該處理器中設定包括被零除(ZE)、無效之運算(IE)、及/或非正常化之運算元(DE)的一或多個異常旗標。

22.如申請專利範圍第 19 項之系統，其中，該當下的指令包含具有在其上實施平方根運算之源運算元值的平方根指令，其中，該指令檢查邏輯回應該源運算元係負數、係非正常化的運算元、係等於無限大、及/或係非數值(NaN)的運算元而致使到運算之該替代序列的跳越。

23.如申請專利範圍第 19 項之系統，其中，該當下的指令包含具有在其上實施平方根運算之源運算元值(x)的平方根指令，其中，該指令檢查邏輯回應使用 x 實施比較而致使到運算之該替代序列的跳越。

24.如申請專利範圍第 19 項之系統，其中，當實施該源運算元的該評估時，該指令檢查邏輯產生用以指示執行該預設序列或該替代序列的向量輸出。

25.如申請專利範圍第 24 項之系統，其中，該指令檢查邏輯進一步產生遮罩輸出，用以信令要以該替代序列處理的元素。

圖式

圖 1A

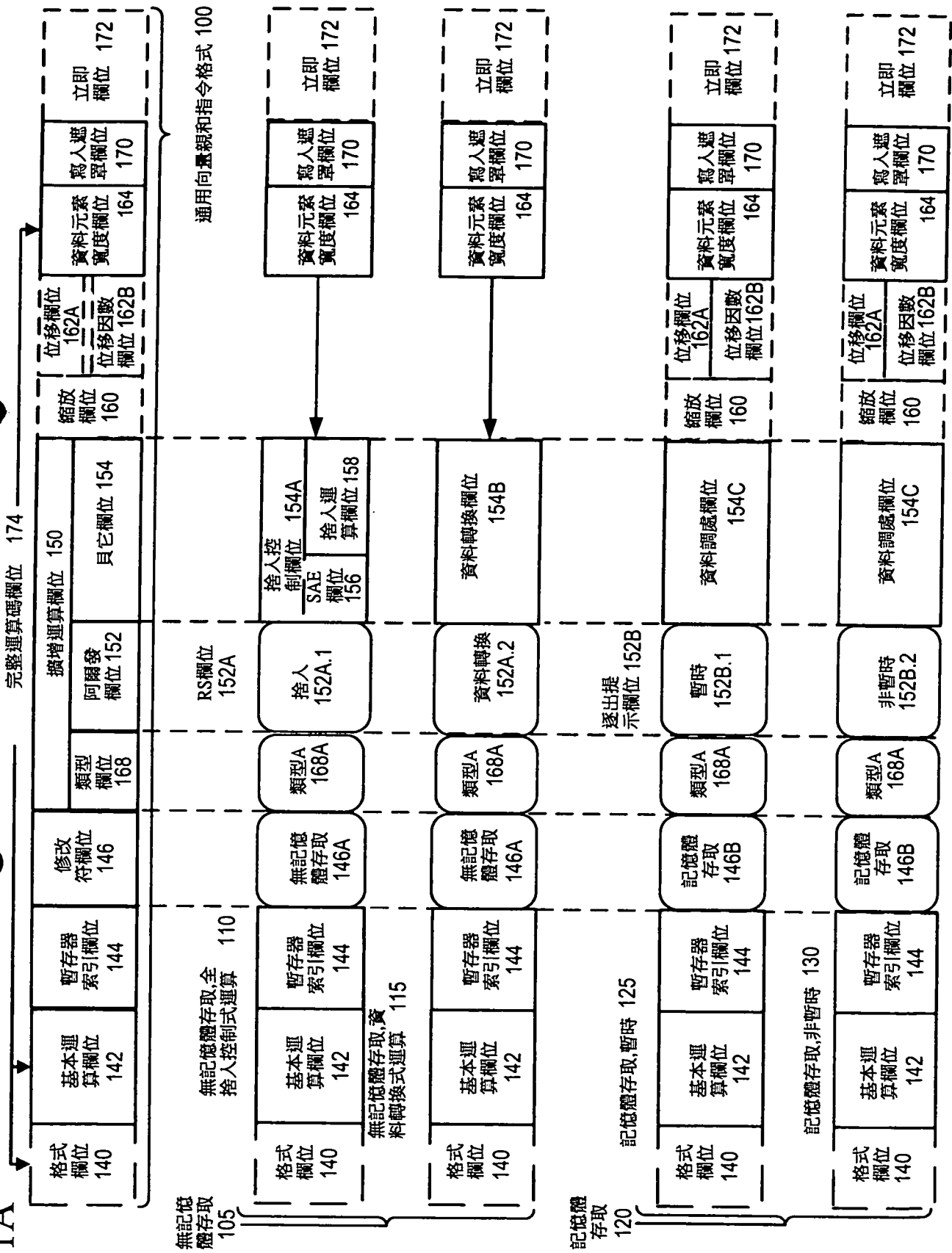
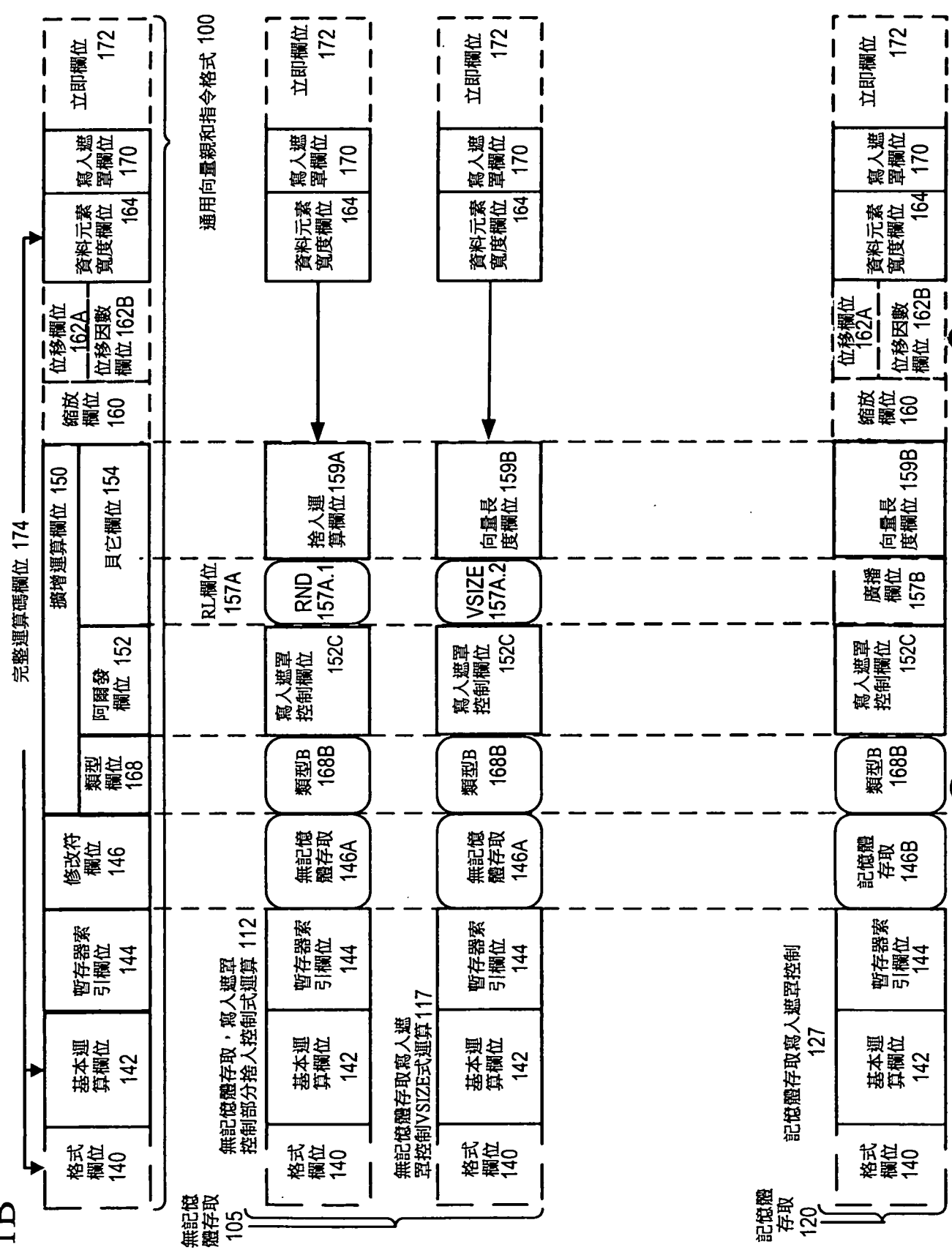


圖 1B



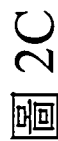
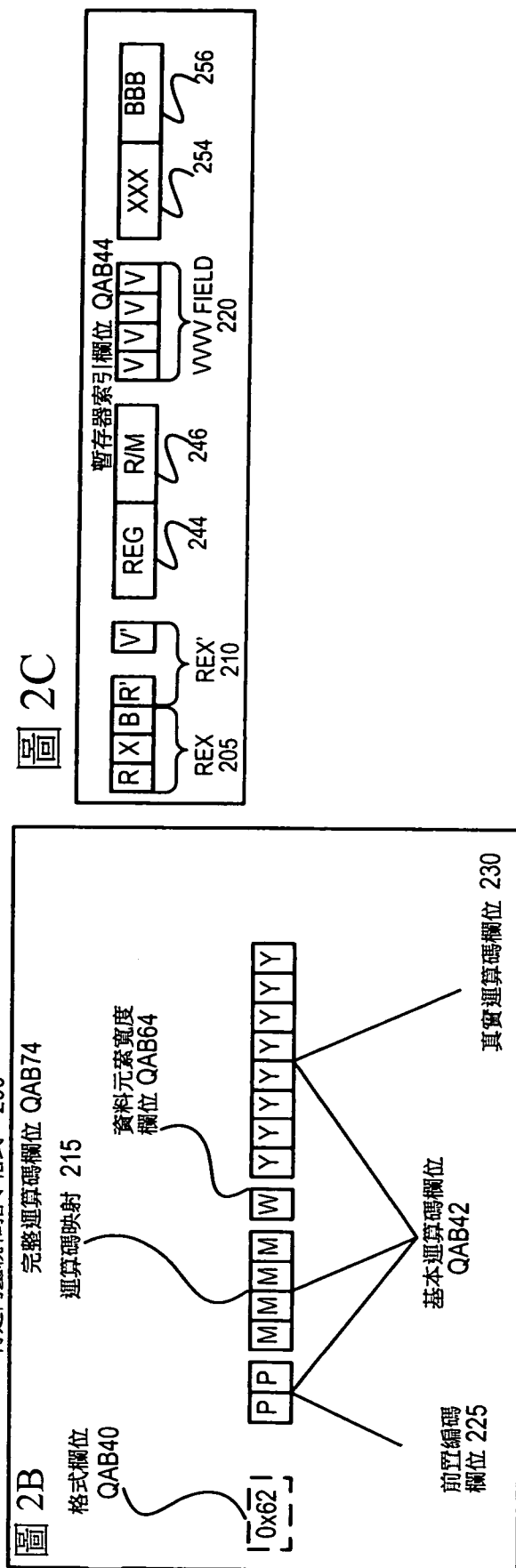
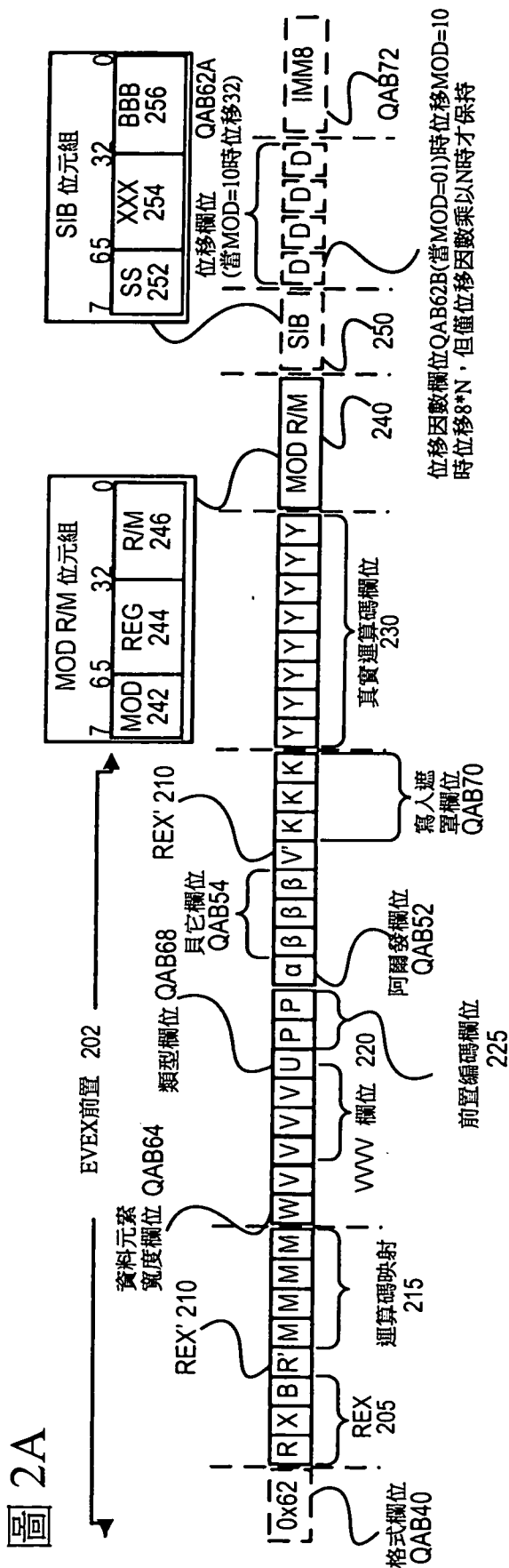


圖 2D

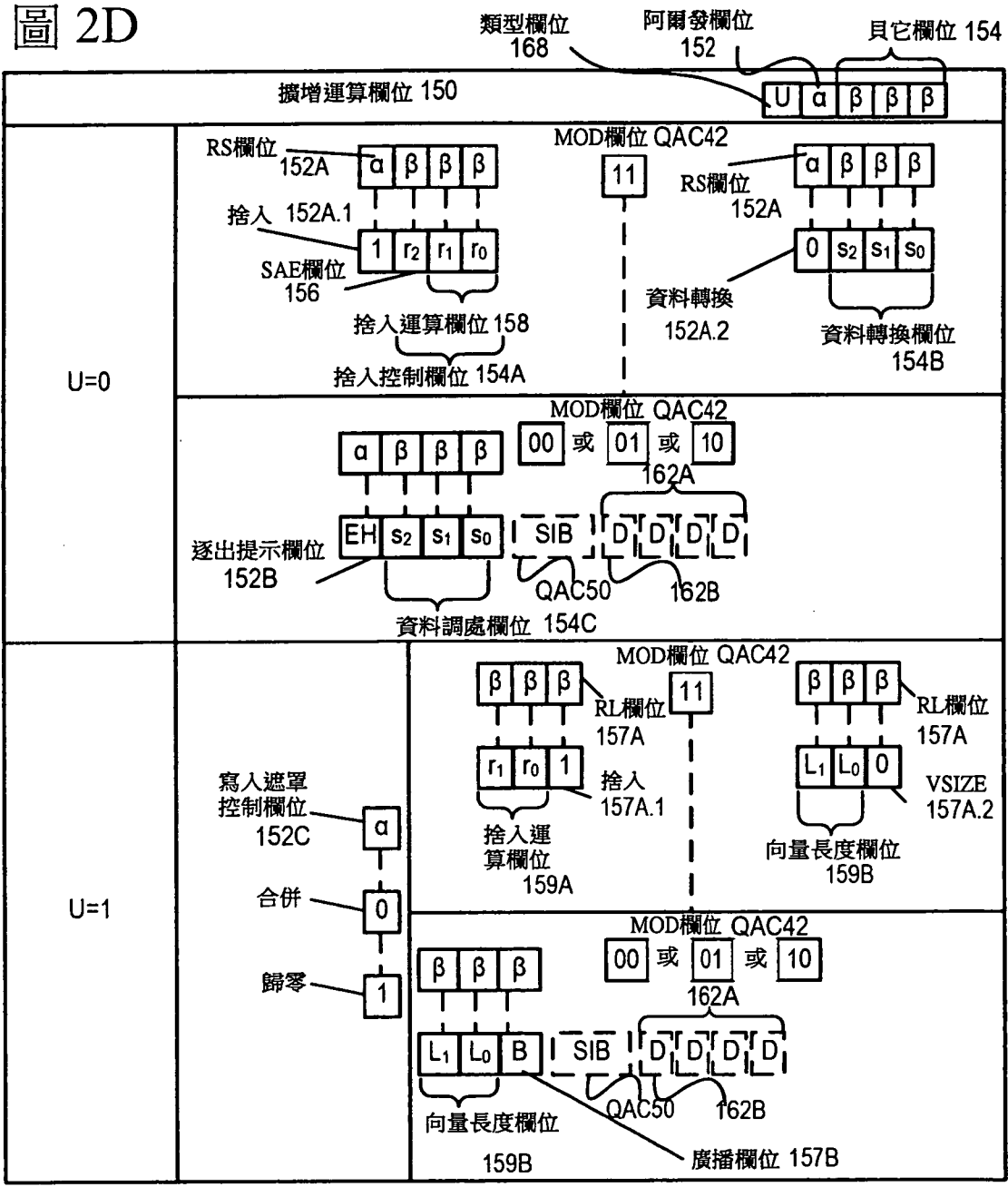
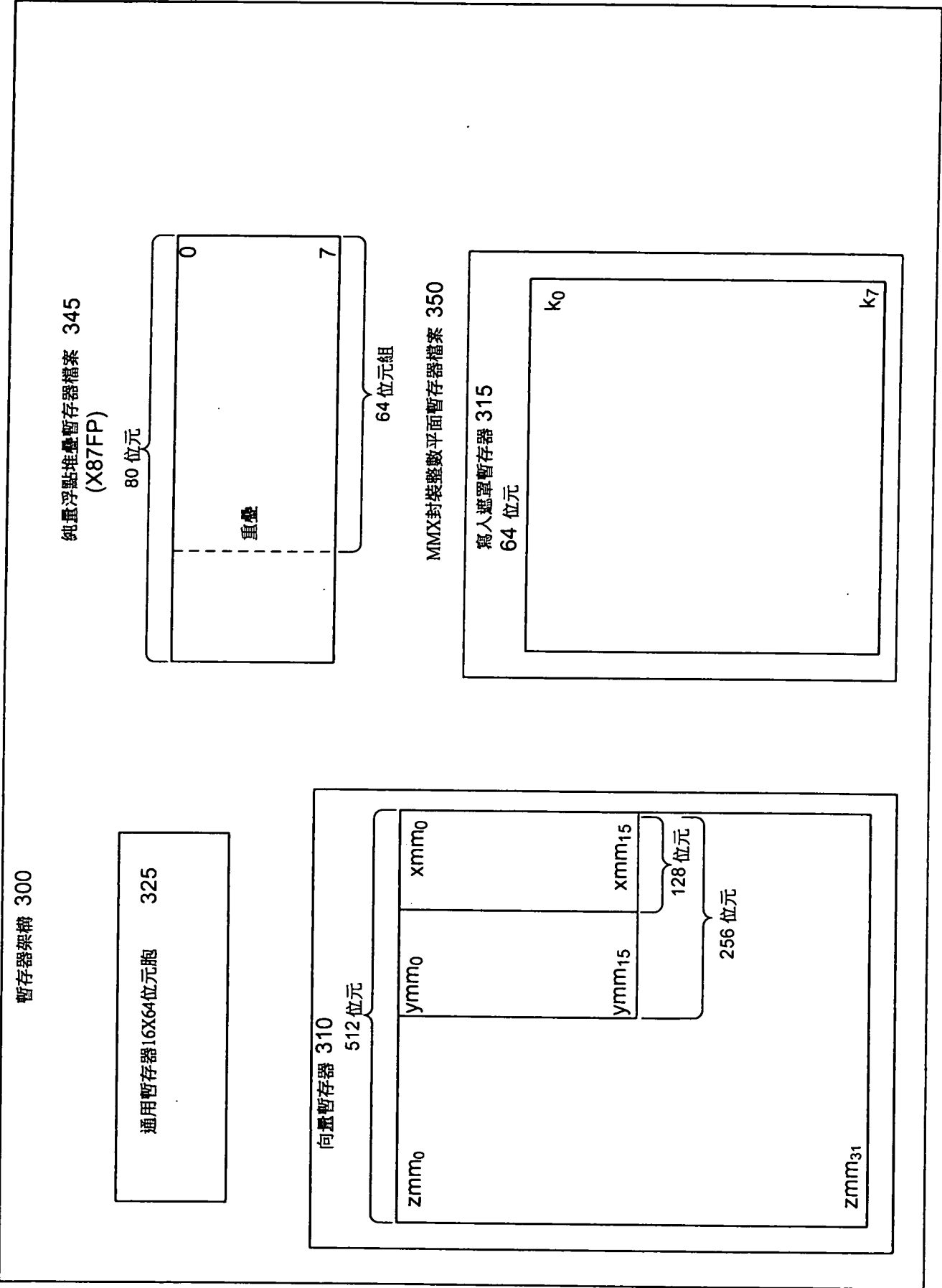


圖 3



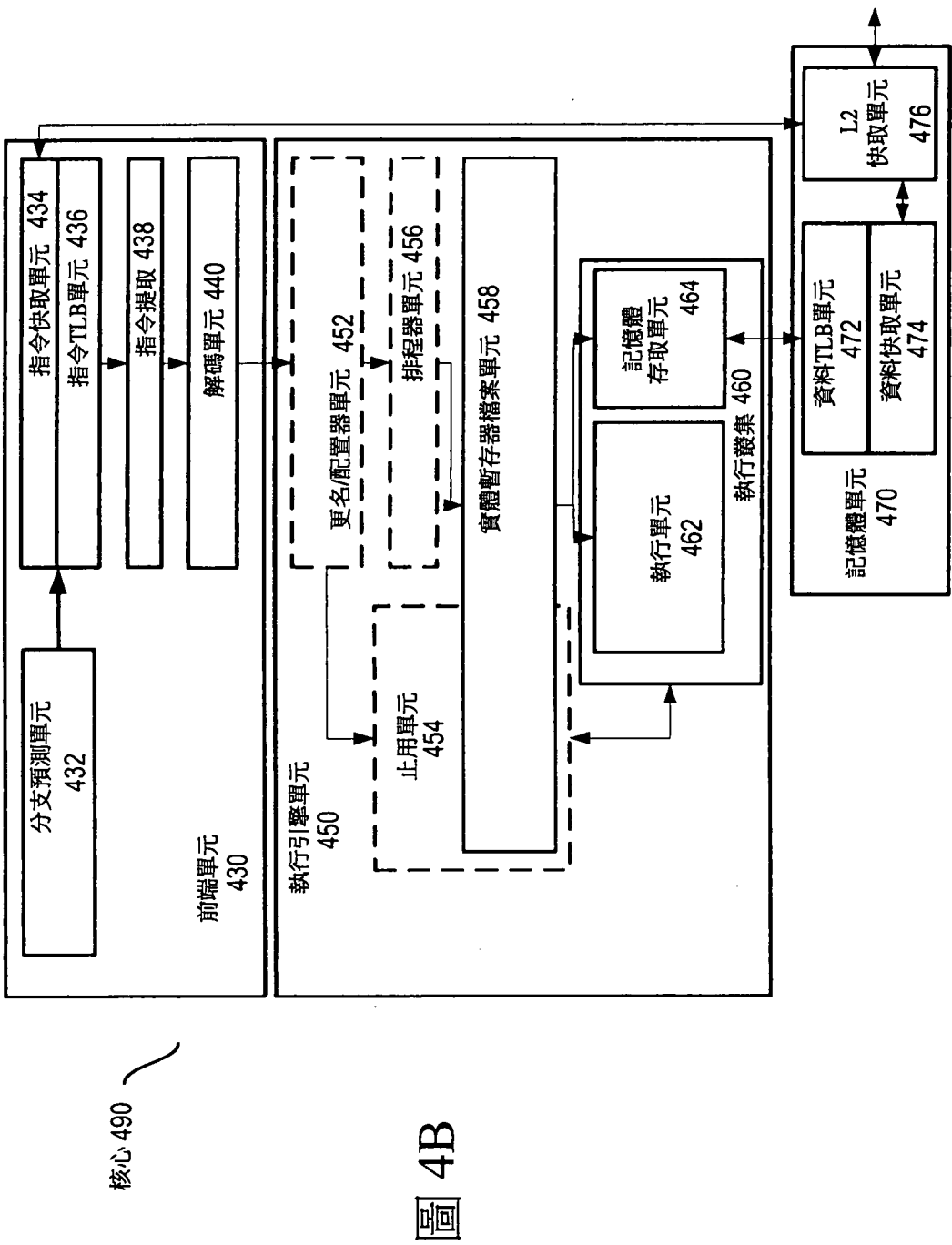
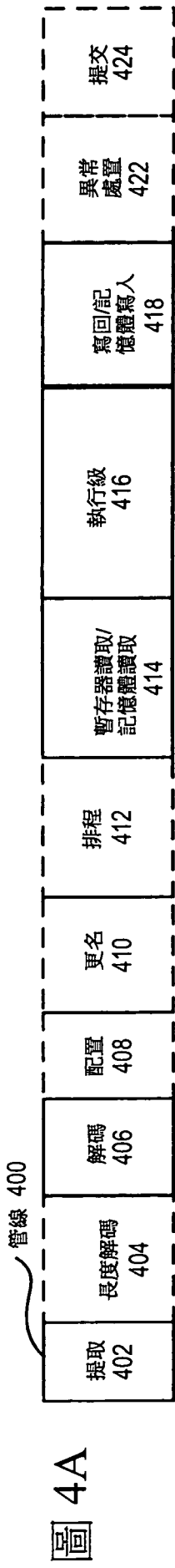


圖 5A

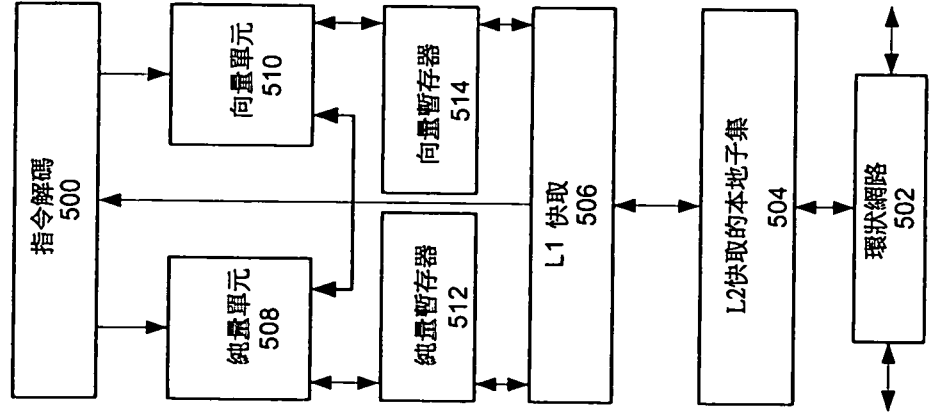


圖 5B

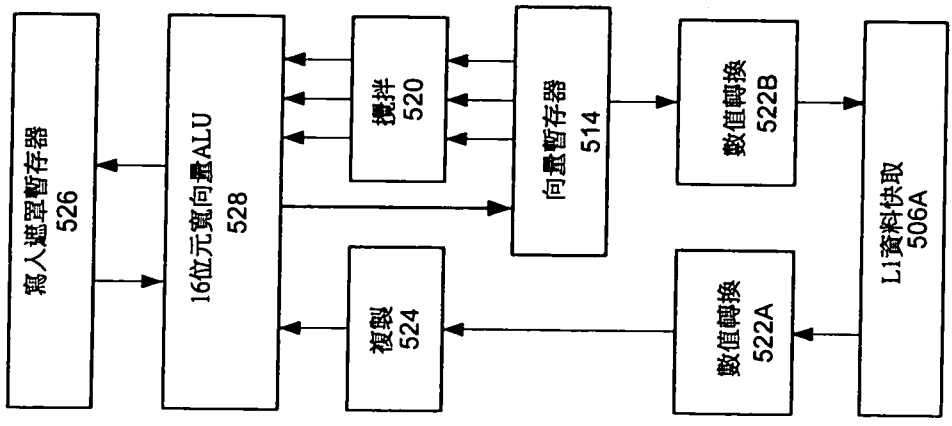


圖 6

處理器 600

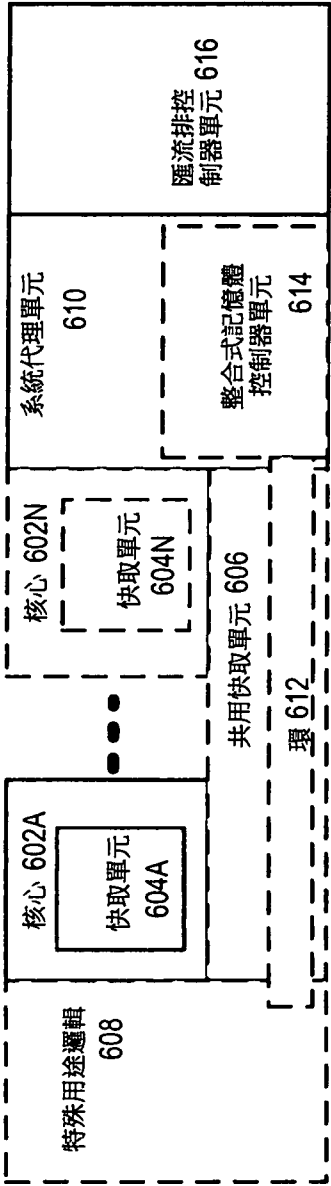


圖 7

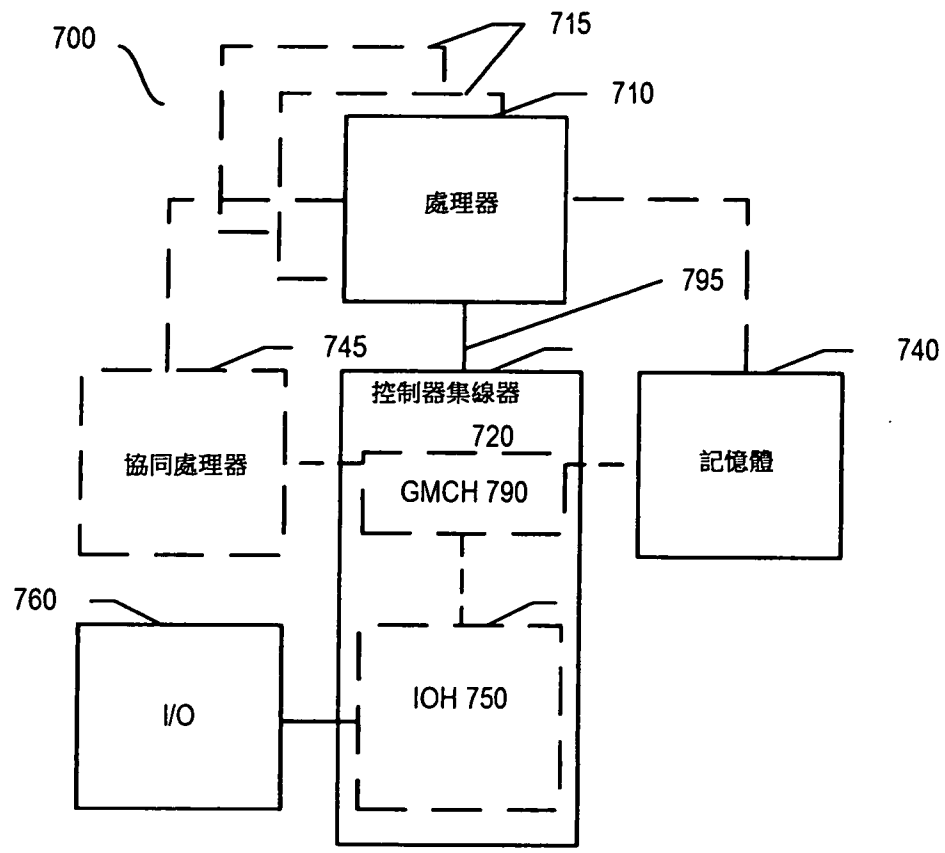


圖 8

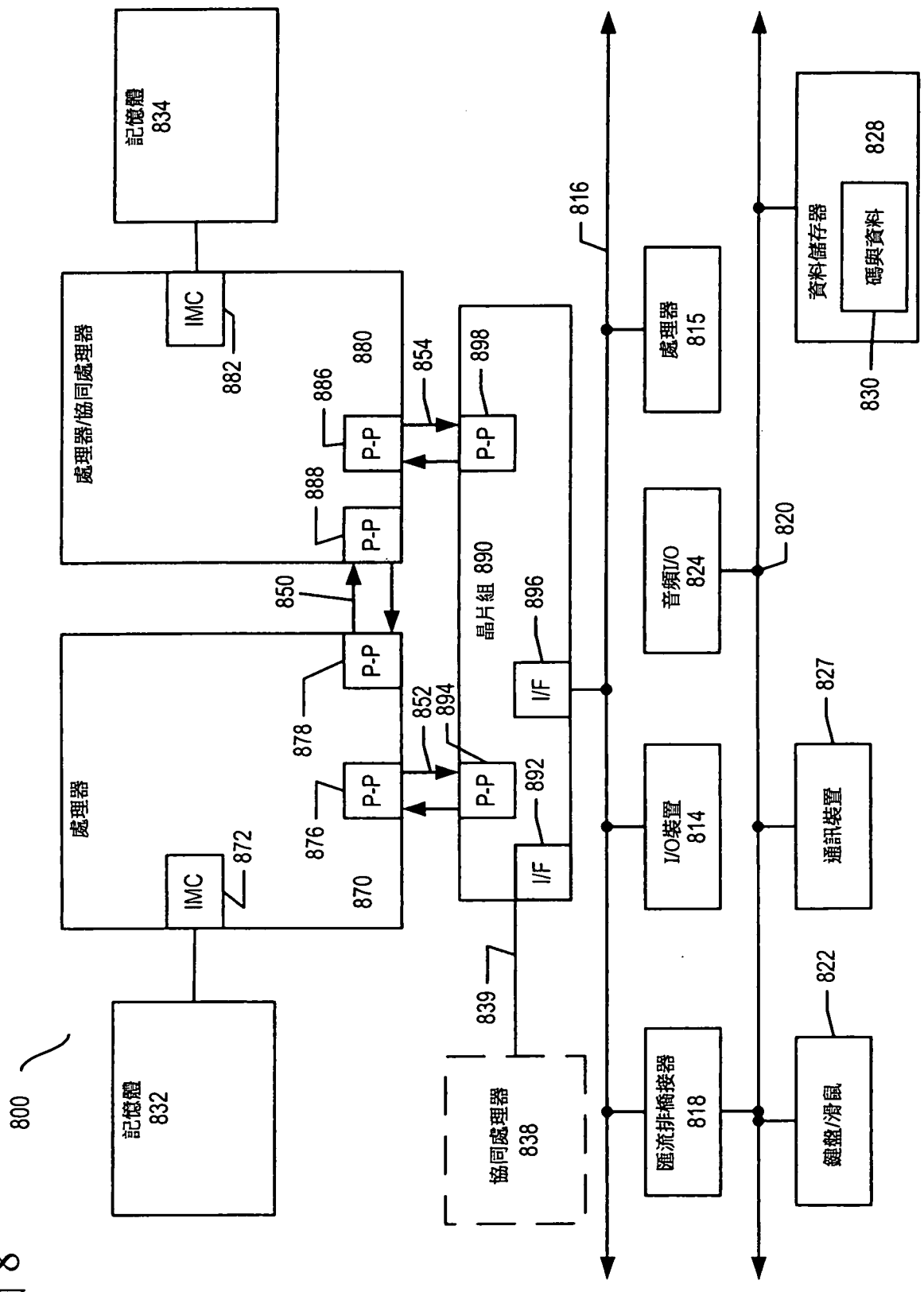


圖 9

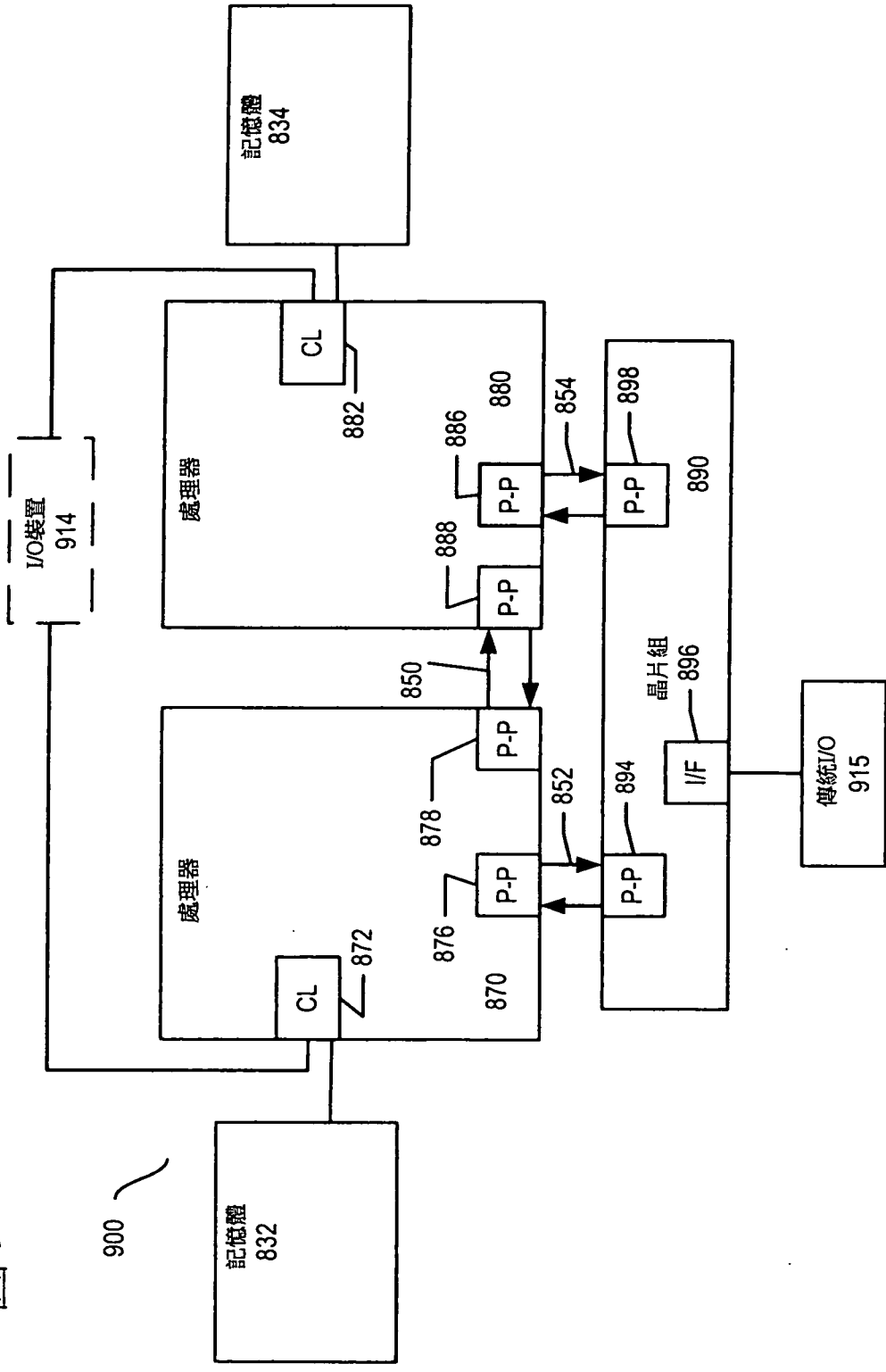


圖 10

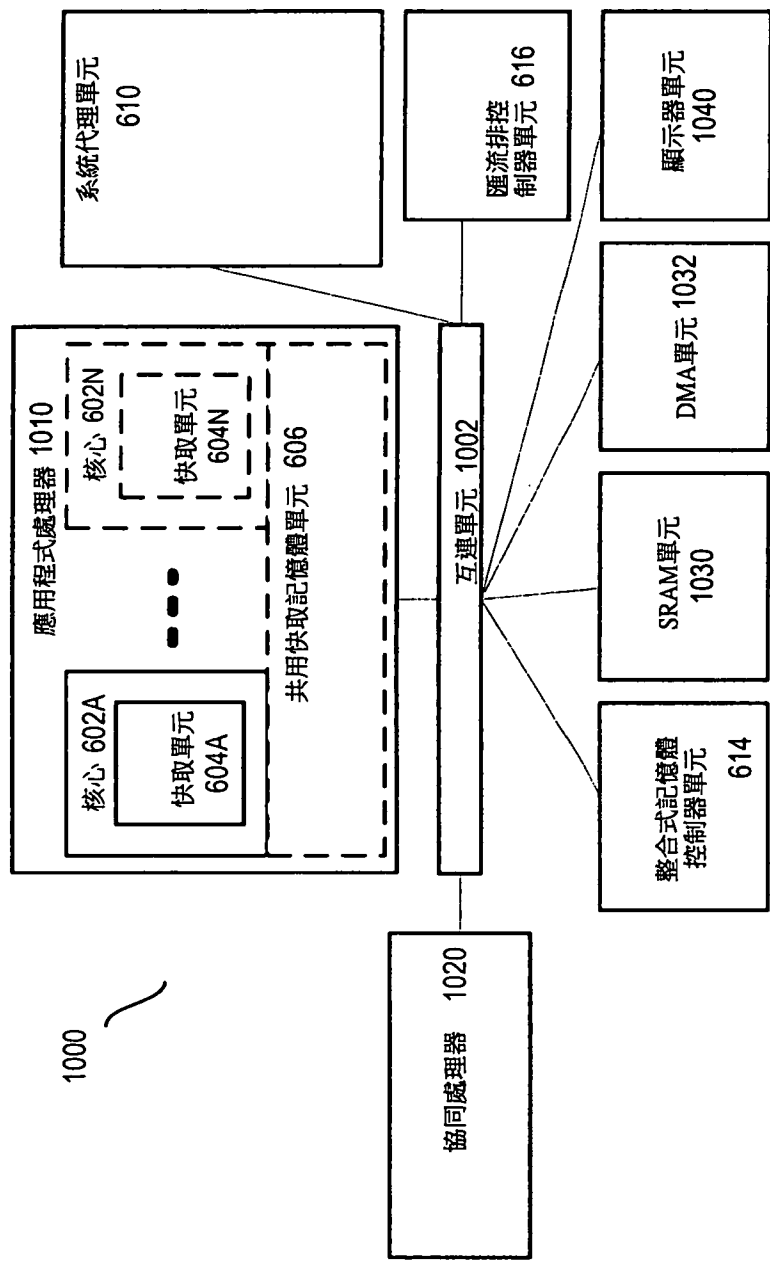


圖 11

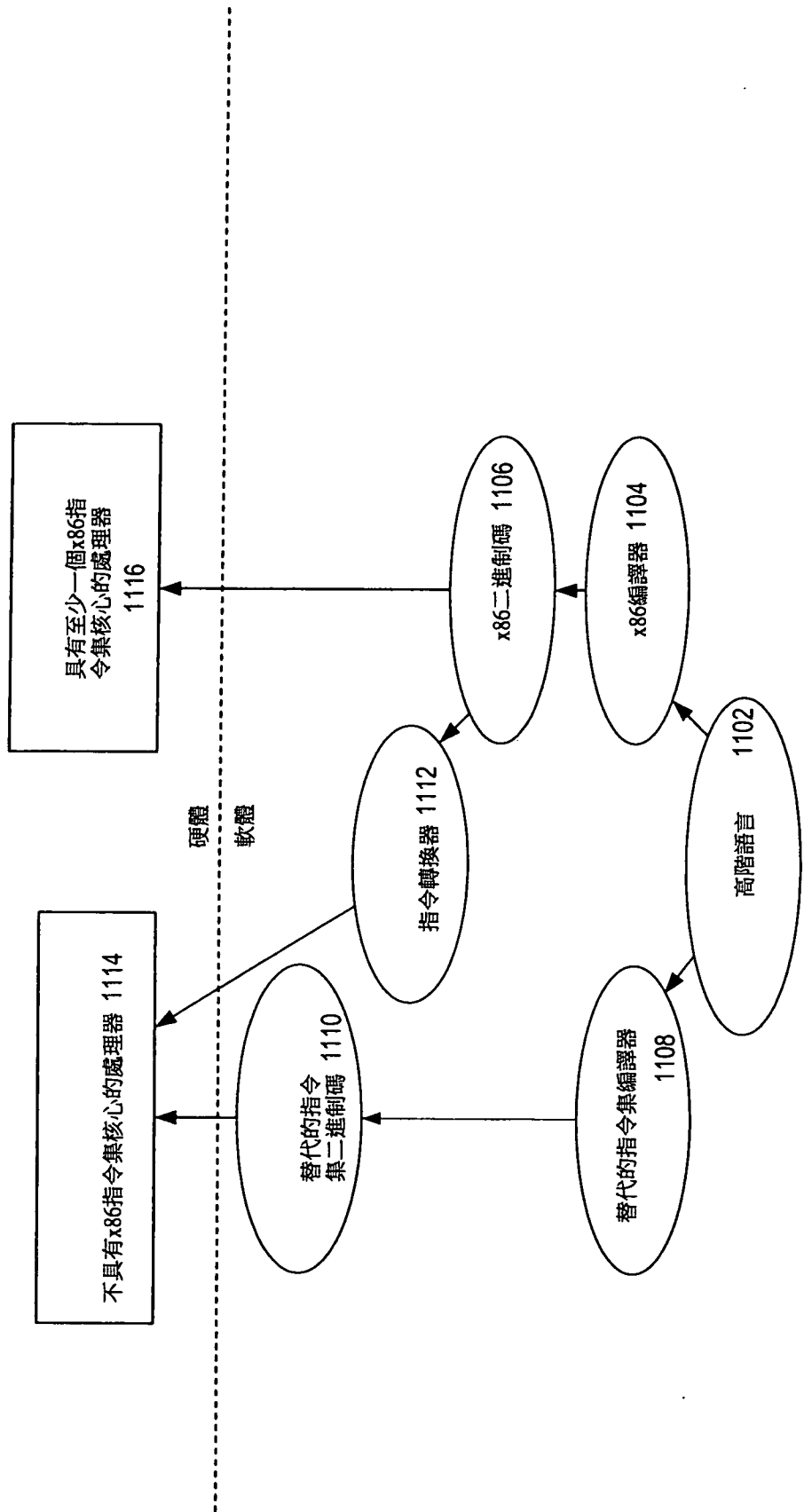


圖 12

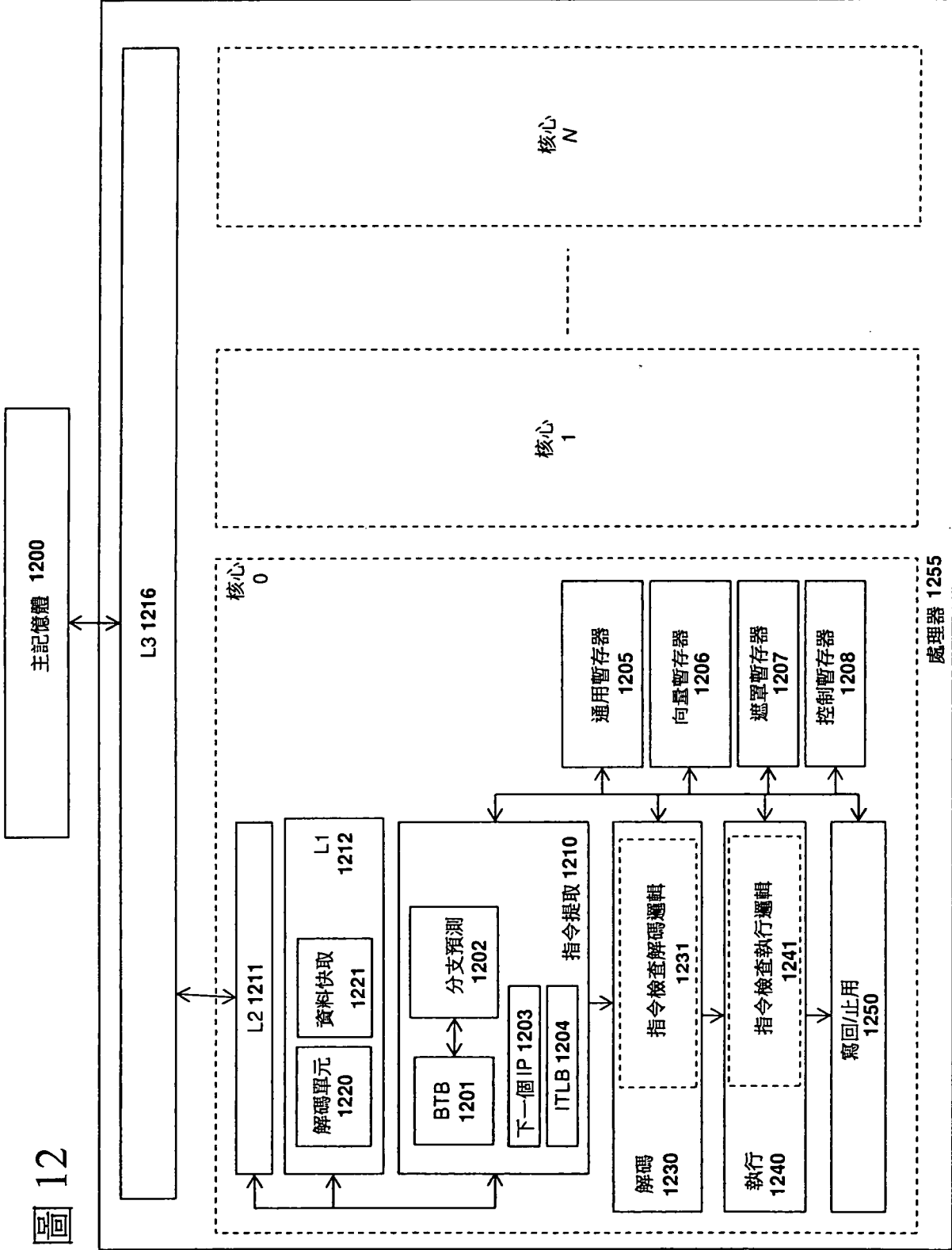


圖 13

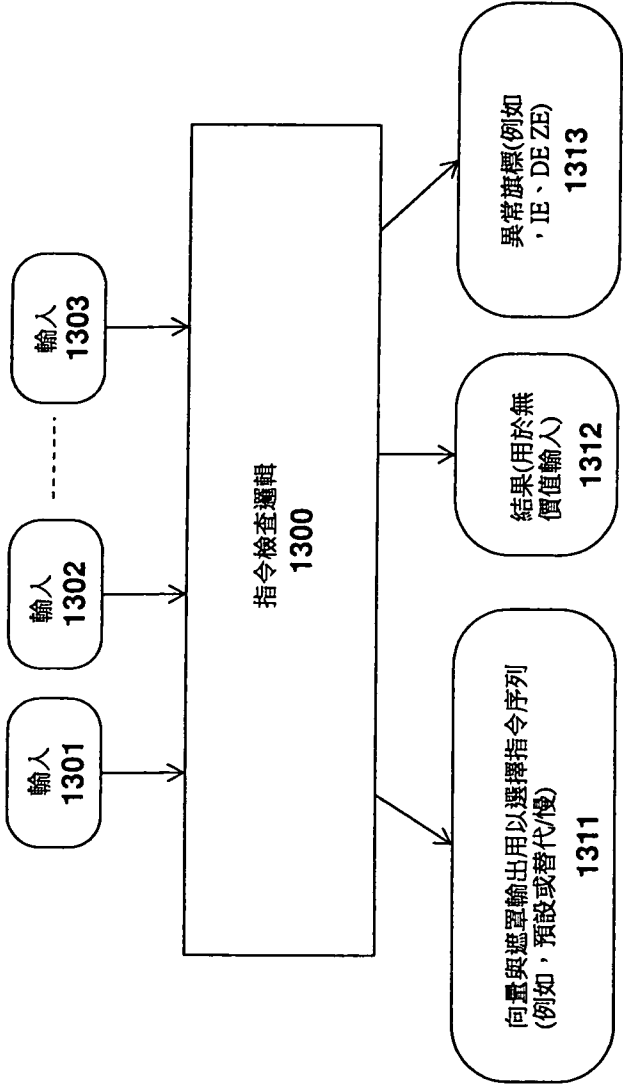


圖 15

kdst, vdst, 旗標			類型
輸入 (x)			
· norm(包括-無限大)	(0, NaN, #IE)		特殊數字
- 無限大	(0, NaN, #IE)		特殊數字
-0	(0, x, None)		特殊數字
+0	(0, x, None)		特殊數字
+ 無限大	(0, +1.0, #DE)		慢流程
+ 無限大	(0, x, None)		特殊數字
qNaN	(0, x, None)		特殊數字
sNaN	(0, qNaN(x), #IE)		特殊數字
偏置指數(x)≤2*精度 ^(a)	(0, +1.0, None)		慢流程
偏置指數(x)=2*偏置 ^(b)	(0, +1.0, None)		慢流程
其它	(1, +0.0, None)		快速流程

圖 16

