I502509

(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 I502509 B**

(45)公告日： 中華民國 104 (2015) 年 10 月 01 日

(21)申請案號：101141357 (22)申請日： 中華民國 101 (2012) 年 11 月 07 日

(51)Int. Cl.： ***G06F9/45 (2006.01)***

(30)優先權：2011/11/07 美國 61/556,782

2012/10/24 美國 13/659,786

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

美國

(72)發明人：孔項勻 KONG, XIANGYUN (US)；王簡仲 WANG, JIAN-ZHONG (US)；格羅佛維諾德 GROVER, VINOD (IN)

(74)代理人：蔡濱陽

(56)參考文獻：

TW I306215 US 2005/0102658A1

US 2011/0078406A1 US 2011/0271170A1

“ A PTX Code Generator for LLVM”, Saarbrücken, Germany, Saarland University, 2010。

審查人員：郭彥鋒

申請專利範圍項數：10 項 圖式數：2 共 19 頁

(54)名稱

一種用於自包括在一電腦程式的高階原始碼中一 64 位元類別轉換表達式擷取一記憶體位址偏置量的方法及系統

SYSTEM AND METHOD FOR EXTRACTING A MEMORY ADDRESS OFFSET FROM A 64-BIT TYPE CONVERSION EXPRESSION INCLUDED IN HIGH-LEVEL SOURCE CODE OF A COMPUTER PROGRAM

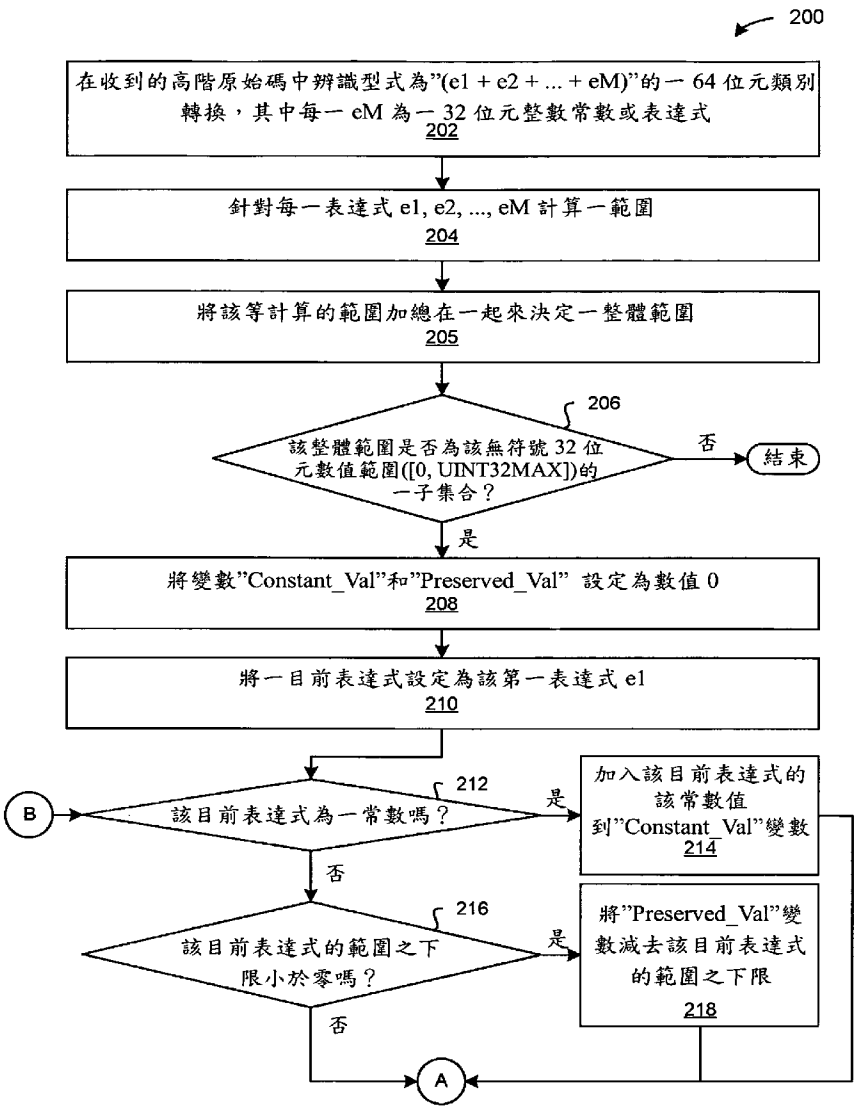
(57)摘要

本發明一具體實施例提出一種自包括在一電腦程式的高階原始碼中的一 64 位元類別轉換表達式中擷取一記憶體位址偏置量的技術。該技術包含接收該 64 位元類別轉換表達式，其中該 64 位元類別轉換表達式包括一或多個 32 位元表達式、針對該等一或多個 32 位元表達式之每一者決定一範圍、藉由加總該等 32 位元表達式的該等範圍來計算一整體範圍、決定該整體範圍為一 32 位元無符號整數的一範圍之子集合、基於該等一或多個 32 位元表達式的該等範圍計算該記憶體位址偏置量、及產生參照該記憶體位址偏置量的至少一組合語言層級的指令。

One embodiment of the present invention sets forth a technique for extracting a memory address offset from a 64-bit type-conversion expression included in high-level source code of a computer program. The technique involves receiving the 64-bit type-conversion expression, where the 64-bit type-conversion expression includes one or more 32-bit expressions, determining a range for each of the one or more 32-bit expressions, calculating a total range by summing the ranges of the 32-bit expressions, determining that the total range is a subset of a range for a 32-bit unsigned integer, calculating the memory address offset based on the ranges for the one or more 32-bit expressions, and generating at least one assembly-level instruction that references the memory address offset.

200・・・方法

202~218・・・步驟



第二A圖

公告本

104 年 2 月 26 日修正替換頁

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：101141357

※ 申請日：101.11.7　　※IPC 分類：G06F 9/45 (2006.01)

一、發明名稱：（中文/英文）

一種用於自包括在一電腦程式的高階原始碼中一 64 位元類別轉換表達式擷取一記憶體位址偏置量的方法及系統

SYSTEM AND METHOD FOR EXTRACTING A MEMORY ADDRESS OFFSET FROM A 64-BIT TYPE CONVERSION EXPRESSION INCLUDED IN HIGH-LEVEL SOURCE CODE OF A COMPUTER PROGRAM

二、中文發明摘要：

本發明一具體實施例提出一種自包括在一電腦程式的高階原始碼中的一 64 位元類別轉換表達式中擷取一記憶體位址偏置量的技術。該技術包含接收該 64 位元類別轉換表達式，其中該 64 位元類別轉換表達式包括一或多個 32 位元表達式、針對該等一或多個 32 位元表達式之每一者決定一範圍、藉由加總該等 32 位元表達式的該等範圍來計算一整體範圍、決定該整體範圍為一 32 位元無符號整數的一範圍之子集合、基於該等一或多個 32 位元表達式的該等範圍計算該記憶體位址偏置量、及產生參照該記憶體位址偏置量的至少一組合語言層級的指令。

三、英文發明摘要：

One embodiment of the present invention sets forth a

technique for extracting a memory address offset from a 64-bit type-conversion expression included in high-level source code of a computer program. The technique involves receiving the 64-bit type-conversion expression, where the 64-bit type-conversion expression includes one or more 32-bit expressions, determining a range for each of the one or more 32-bit expressions, calculating a total range by summing the ranges of the 32-bit expressions, determining that the total range is a subset of a range for a 32-bit unsigned integer, calculating the memory address offset based on the ranges for the one or more 32-bit expressions, and generating at least one assembly-level instruction that references the memory address offset.

四、指定代表圖：

(一)本案指定代表圖為：第（二A）圖。

(二)本代表圖之元件符號簡單說明：

200　方法　　202～218　步驟

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處理，尤指一種最佳化 64 位元位址模式的運算法。

【相關技術】

開發者使用編譯器來由高階原始碼產生可執行的程式。基本上，一編譯器設置成接收一程式的高階原始碼(例如以 C++ 或 Java 寫成)、決定將要執行該程式的一目標硬體平台(例如一 x86 處理器)、然後將該高階原始碼轉譯成為可在該目標硬體平台上執行的組合語言層級的碼。此種組態的好處是開發者可撰寫一單一高階原始碼程式，然後可將程式在多種硬體平台上來執行，例如行動裝置、個人電腦或伺服器。

概言之，一編譯器包括三個組件：一前端、一中端及一後端。該前端設置成確保該高階原始碼可滿足程式化語言的語法和語意，而據此該前端可產生該高階原始碼的一第一中間表示(IR, “Intermediate representation”)。該中端設置成接收和最佳化該第一 IR，其通常牽涉到例如移除在該第一 IR 中所包括的無法到達的碼(如果有的話)。在最佳化該第一 IR 之後，該中端所產生的一第二 IR 由該後端來做處理。特別是該後端接收該第二 IR，並將該第二 IR 轉譯成為組合語言層級的碼。

為了有效率地產生組合語言層級的碼，需要同時降低該組合語言層級的碼中所參照的暫存器數目以及位址運算碼的數量。要達成這些降低所使用的一種方法稱為「暫存器加偏置量」(“register plus offset”)方法，其中包含產生參照一基礎位址暫存器和一固定偏置量的組合語言層級的指令。例如，在 CUDA™ 架構中，經由該組合語言層級的指令“ld.global.f32 f12, [rd12 + 64]”將一數值自通用記憶體載入到一暫存器”f12”，其中”rd12”為該基礎位址暫存器，而”64”為該固定偏置量。這種方法的優點是多個記憶體位址能夠共用同一基礎位址暫存

器"rd12"。例如，當"rd12"儲存該數值"16"時，該表達式[rd12+64]使得儲存在記憶體位址[80]中的該數值被載入到該通用記憶體位址"f12"。同樣地，當"rd12"儲存該數值"20"時，該表達式[rd12+64]使得儲存在記憶體位址[84]中的該數值被載入到該通用記憶體位址"f12"。因此，編譯器需要能夠辨識出可被降低至實作該「暫存器加偏置量」方法的組合語言層及指令的高階指令。

根據上述方法，可被降低的常見型式之高階指令包括參照由一32位元表達式偏置的一64位元基礎記憶體位址的高階指令。此格式的一示例為"64位元基礎位址+(uint64_t)(32位元表達式)"，其中該32位元表達式被型別轉換成64位元(經由"unit64_t"類型轉換記號)，所以該32位元表達式的結果值為64位元。可實作上述格式的一高階指令的示例為"&p + (uint64_t) (-20 * x + 30 * y + 1100)"。關於上述之「暫存器加偏置量」方法，其需要決定是否可由該高階指令中的表達式"(-20 * x + 30 * y + 1100)"中擷取一固定偏置量。可惜地是，該64位元類別的轉換在做出這樣的決定時會引發數個複雜的問題，特別是當該表達式包括無符號整數算數時。特別是，數種程式化語言標準，例如C/C++的標準，其設計無符號的運算來在當發生溢位時產生環繞式數值。因此，習用的編譯器通常無法有效地將合法的高階指令轉譯成組合語言層級的指令來實作該「暫存器加偏置量」方法。

因此，本技術中需要允許自高階指令擷取的固定偏置量用於產生組合語言層級的指令，以實作該「暫存器加偏置量」之方法。

【發明內容】

本發明一具體實施例提出一種自包括在一電腦程式的高階原始碼中的一64位元類別轉換表達式中擷取一記憶體位址偏置量的方法。該方法包括以下步驟： 接收該64位元類別轉

換表達式，其中該 64 位元類別轉換表達式包括一或多個 32 位元表達式、針對該等一或多個 32 位元表達式之每一者決定一範圍、藉由加總該等 32 位元表達式的該等範圍來計算一整體範圍、決定該整體範圍為一 32 位元無符號整數的一範圍內之子集合、基於該等一或多個 32 位元表達式的該等範圍計算該記憶體位址偏置量、及產生參照該記憶體位址偏置量的至少一組合語言層級的指令。

該等揭示具體實施例的一個好處為一編譯器設置成自一電腦程式的高階指令擷取記憶體位址固定偏置量，所以它們可用於組合語言層級的指令中來實作此處所述的該「暫存器加偏置量」方法。因此，其可整體減少該電腦程式所參照的暫存器數目，以及整體地降低其中所包括的該等位址運算的複雜度。較佳地是，該電腦程式可以在處理器上更有效率地執行，並可允許其它電腦程式利用該等經空出的暫存器。

【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 可為一北橋晶片，其經由一匯流排或其它通訊路徑 106(例如 HyperTransport 聯結)連接到一 I/O(輸入/輸出)橋接器 107。I/O 橋接器 107 可為一南橋晶片，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 102 及記憶體橋接器 106 轉送該輸入到 CPU 105。一平行處理子系統 112 經由一匯流排或第二通訊路徑 105(例如 PCI(周邊組件互連接)

Express, 加速圖形埠、或 HyperTransport 鏈路)耦合至記憶體橋接器 113；在一具體實施例中，平行處理子系統 112 為一繪圖子系統，其傳遞像素到一顯示器 110，其可為任何習用的陰極射線管、液晶顯示器、發光二極體顯示器或類似者。一系統碟 114 亦連接至 I/O 橋接器 107，並可設置成儲存 CPU 102 和平行處理子系統 112 所使用的內容、應用程式和資料。系統碟 114 做為儲存應用程式和資料的非揮發性儲存器，並可包括固定式或可移除式硬碟機、快閃記憶體裝置、和 CD-ROM(光碟唯讀記憶體)、DVD-ROM(數位多功能碟片 ROM)、藍光碟、HD-DVD(高解析度 DVD)或任何磁性、光學或固態儲存裝置。

一交換器 116 提供 I/O 橋接器 107 與其它組件（例如一網路轉接器 118 與多種嵌入卡 120 和 121）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、影像錄製裝置及類似者，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 和 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，且構成一圖形處理單元(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")。

在一具體實施例中，平行處理子系統 112 包括一或多個平行處理單元(PPU, "Parallel processing units")，其每一者耦合至一局部平行處理(PP)記憶體。概言之，平行處理子系統 112 包括數目為 U 的 PPU，其中 U >= 1。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 為圖形處理器，其具有顯像管線，其能夠設置成執行相關於依據 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 所供應的圖形資料來產生像素資料的多種作業，與該局部平行處理記憶體 (其能夠做為圖形記憶體，例如做為一習用圖框緩衝器) 互動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，以及用於通用型運算的一或多個其它 PPU。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。

第一圖所示的系統僅為例示性，其有可能有多種變化及修正。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 的數目的連接拓樸皆可視需要修改。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，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，而非分別做為一或多個分散裝置。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。

最佳化 64 位元位址模式

如此處更為詳細地說明者，本發明之具體實施例包含在第一圖的電腦系統 100 上執行的一編譯器 150，其設置成實做一運算法來自動地將一電腦程式的高階指令自動地轉譯成組合語言層級的指令來實作先前所述的「暫存器加偏置量」方法。特定而言，編譯器 150 接收該電腦程式的高階原始碼，並辨識出參照由一 32 位元表達式所偏置的一 64 位元基礎位址的至少一高階指令。在辨識時，編譯器 150 決定一固定偏置量是否能自該 32 位元表達式安全地擷取，其中該固定偏置量係包括在實作該「暫存器加偏置量」方法而產生的組合語言層級的指令中。如上所述，執行此技術可以整體地減少被該電腦程式所參照的暫存器之數目，以及整體地降低其中所包括的該等位址運算的複雜度。較佳地是，由一系統的觀點而言，該電腦程式可更有效率地執行，並可允許在該系統內執行的其它電腦程式來利用由本發明揭示技術所空出的該等暫存器。

第二 A 到二 B 圖例示根據本發明一具體實施例中用於自一電腦程式的一高階指令擷取使用在組合語言層級的指令中的一固定偏置量來實作該「暫存器加偏置量」方法的方法步驟 200 之流程圖。雖然該等方法步驟係配合第一圖之系統做說明，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，一方法 200 開始於步驟 202，其中編譯器 150 辨識由編譯器 150 接收的高階原始碼中一 64 位元類別轉換，其型式為：“(e1 + e2 + ... + eM)”，其中每一 eM 為一 32 位元整數常數或表達式。64 位元類別轉換中的一高階指令的一示例為“&p + (uint64_t) (-20 * x + 30 * y + 1100)”，其中&p 為一 64 位元基礎位址、“-20 * x”和“30 * y”為 32 位元整數表達式、

“1100”為一 32 位元整數常數、及“(uint64_t)”為該 64 位元類別轉換。

在步驟 204，編譯器 150 針對每一表達式 e1, e2, ... ,eM 計算一範圍。在一具體實施例中，編譯器 150 設置成辨識每一表達式 eM 的最不可能的數值以及針對每一表達式 eM 的最有可能的數值。當該表達式 eM 為一 32 位元常數時，該範圍簡單地就等於該 32 位元常數值，例如”1100”，而該最不可能的數值等於該最有可能的數值。但是，當該表達式 eM 為一實際的 32 位元表達式時，例如“-20 * x”，編譯器 150 解析該高階碼來藉由同時決定”x”的最不可能的數值及”x”的最有可能的數值來決定該範圍，然後執行由該 32 位元表達式所定義的該算術來建立一最終範圍。例如，如果編譯器 150 決定該 32 位元表達式“-20 * x”中”x”的範圍為[0,100]，則該 32 位元表達式的範圍為[-20 * 100, -20 * 0] = [-200, 0]。如以下進一步的詳細說明，這些計算出的範圍由編譯器 150 用來決定一固定偏置量是否可由該 64 位元類別轉換來安全地擷取，如果是，則決定該固定偏置量的該正確數值。

在步驟 205，編譯器 150 將 e1, e2, ... , eM 的該等範圍加總在一起，以決定包括在該經接收的高階原始碼中該表達式的整體範圍(即 e1, e2, ... , eM 的整體範圍)。在步驟 206，編譯器 150 決定該整體範圍是否為該無符號 32 位元數值範圍([0, UINT32MAX])的一子集合。如果在步驟 206 編譯器 150 決定該整體範圍為該無符號 32 位元數值範圍([0, UINT32MAX])的一子集合時，則方法 200 進行到步驟 208，如下所述。否則，高階方法 200 即結束，因為並無固定偏置量可自該 64 位元類別轉換安全地擷取。

在步驟 208，編譯器 150 將該等變數“Constant_Val”和“Preserved_Val” 設定為數值 0。在步驟 210，編譯器 150 該第一表達式 e1 設定為一目前表達式。在一具體實施例中，該目

前表達式被實作成一指向至一表達式 eM 的指標，所以該編譯器可以有效率地解析包括在該 64 位元類別轉換中的每一 eM。

在步驟 212，編譯器 150 決定該目前表達式是否為一常數。請注意在第一次通過步驟 212 時，該目前表達式指向至該表達式 e1。如果在步驟 212 編譯器 150 決定該目前表達式為一常數(例如"1100")，則方法 200 進行到步驟 514，其中編譯器 150 將該目前表達式的該常數數值加入到該"Constant_Val"變數。

如果在步驟 212 編譯器 150 決定該目前表達式並非常數，則編譯器 150 知道該目前表達式事實上為一表達式(例如"-20 * x")。因此，方法 200 進行到步驟 216，其中編譯器 150 決定該目前表達式的該範圍之下限是否小於零。如果在步驟 216 編譯器 150 決定該目前表達式的該範圍之下限為小於零，則方法 200 進行到步驟 218。在步驟 218，編譯器 150 自該變數"Preserved_Val"減去該目前表達式的該範圍之下限。然後方法 200 進行到步驟 220，如下述。

請注意到如果步驟 212 和步驟 216 的決定結果皆為否，則該目前表達式對於該"Constant_Val"變數和該"Preserved_Val"變數的該等數值皆沒有影響。此時方法進行到步驟 220，其中編譯器 150 分析包括在該 64 位元類別轉換中的下一個表達式 eM(如果有的話)。

在步驟 220，編譯器 150 決定該類別轉換要求中是否包括額外的表達式。如果在步驟 220 編譯器 150 決定該類別轉換要求中包括額外的表達式，則方法 200 進行到步驟 222。否則，方法 200 進行到步驟 224，如下述。

在步驟 222，編譯器 150 將該目前表達式設定為包括在該類別轉換要求中的下一個表達式。例如，如果該目前表達式指向到 e1，則在步驟 222 該目前表達式被更新為指向到 e2。

在步驟 224，編譯器 150 決定該"Constant_Val"變數是否小於或等於該"Preserved_Val"變數。如果在步驟 224 編譯器 150

決定該"Constant_Val"變數小於或等於該"Preserved_Val"變數，則方法200進行到步驟226，其中編譯器150傳回一零的固定偏置量值。否則，方法200進行到步驟228，其中編譯器150擷取一固定偏置量值，其等於該Preserved_Val變數減去該Constant_Val變數後之值。

因此，如果透過方法步驟200，編譯器150能夠擷取一固定偏置量值，則該固定偏置量值可被包括在實作該「暫存器加偏置量」方法的組合語言層級的指令中。

總而言之，本發明之具體實施例提出一種自包括在一電腦程式的高階原始碼中的一64位元類別轉換表達式中擷取一記憶體位址偏置量的技術。編譯器150接收該64位元類別轉換表達式，其包括一或多個32位元表達式，例如**"-20 * x", "30 * y"**和**"1100"**。編譯器1500根據上述之方法200的步驟204針對該等一或多個32位元表達式之每一者來決定一範圍，然後決定該等一或多個32位元表達式之每一者的該範圍是否為一32位元無符號整數的該範圍的一子集合。假設該等範圍符合此子集合要求，編譯器150基於該等一或多個32位元表達式的該等範圍計算一記憶體位址偏置量，如方法200之步驟212-228中所述。

此處所揭示之該等技術的一項好處為編譯器150設置成自動地轉譯一電腦程式的高階指令成為組合語言層級的指令來實作該「暫存器加偏置量」方法，其如此處所述可提供數種優點。再次地，這些優點可包括可整體減少該電腦程式所參照的暫存器數目，以及整體地降低其中所包括的該等位址運算的複雜度。因此，該電腦程式可以在處理器上更有效率地執行，並可允許其它電腦程式利用該等經空出的暫存器，其增加並行執行的可能性。

本發明一具體實施例可以實作成由一電腦系統使用的一程式產品。該程式產品的程式定義該等具體實施例的功能(包括此處所述的方法)，並可包含在多種電腦可讀取儲存媒體

上。例示性的電腦可讀取儲存媒體包括但不限於：(i)不可寫入儲存媒體(例如在一電腦內唯讀記憶體裝置，例如可由CD-ROM 讀取的光碟唯讀記憶體(CD-ROM)碟片，快閃記憶體，ROM 晶片，或任何其它種類的固態非揮發性半導體記憶體)，其上可永久儲存資訊；及(ii)可寫入儲存媒體(例如在一磁碟機內的軟碟片、或硬碟機、或任何種類的固態隨機存取半導體記憶體)，其上可儲存可改變的資訊。

本發明已經參照特定具體實施例在以上進行說明。但是本技術專業人士將可瞭解到在不背離附屬申請專利範圍所提出之本發明的廣義精神與範圍之下可對其進行多種修正與改變。因此前述的說明及圖面係在以例示性而非限制性的角度來看待。

因此，本發明之具體實施例的範圍係在以下的申請專利範圍中提出。

【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。

第一圖例示設置成一種實作本發明一或多種態樣之電腦系統的方塊圖。

第二 A 到二 B 圖例示根據本發明一具體實施例中用於自一電腦程式的一高階指令擷取使用在組合語言層級的指令中的一固定偏置量來實作該「暫存器加偏置量」方法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	圖形處理單元
113	通訊路徑
114	系統碟
116	交換器
118	網路轉接器
120, 121	嵌入卡
150	編譯器
200	方法
202～228	步驟

七、申請專利範圍：

1. 一種用於自包括在一電腦程式的高階原始碼中一64位元類別轉換表達式擷取一記憶體位址偏置量之方法，該方法包含：

接收該 64 位元類別轉換表達式，其中該 64 位元類別轉換表達式包括一或多個 32 位元表達式；

針對該等一或多個 32 位元表達式之每一者決定該 32 位元表達式的一範圍；

藉由加總該等32位元表達式的該等範圍來計算一整體範圍；

決定該整體範圍為一32位元無符號整數的一範圍內的一子集合；

基於該等一或多個32位元表達式的該等範圍計算該記憶體位址偏置量；以及

產生參照該記憶體位址偏置量的至少一組合語言層級的指令。

2. 如申請專利範圍第 1 項之方法，其中包括在該等一或多個 32 位元表達式中的一第一 32 位元表達式包含一常數值。

3. 如申請專利範圍第 2 項之方法，其中決定該第一 32 位元表達式的該範圍包含擷取關聯於該第一32位元表達式的該常數值。

4. 如申請專利範圍第 2 項之方法，其中包括在該等一或多個 32 位元表達式中的一第二 32 位元表達式包括至少一變數。

5. 如申請專利範圍第 4 項之方法，其中決定該第二 32 位元表達式的該範圍包含：

解析該高階原始碼來決定將在該電腦程式的執行期間被指定給該至少一變數的一最小值；以及

將該第二32位元表達式的該範圍之一下限設為該最小值。

6. 如申請專利範圍第 5 項之方法，其中基於該等一或多個 32

位元表達式的該等範圍計算該記憶體位址偏置量包含：

建立一第一變數和一第二變數成為具有等於零的數值；

將關聯於該第一32位元表達式的該常數值加入到該第一變數；

自該第二數值減去關聯於該第二32位元表達式的該下限；以及

當該第一數值小於或等於該第二數值時將該記憶體位址偏置量設定為該第二數值和該第一數值之間的差值；或

當該第一數值大於該第二數值時將該記憶體位址偏置量設定為零。

7. 如申請專利範圍第 1 項之方法，其中該記憶體位址偏置量係在該至少一組合語言層級的指令之行內做參照。

8. 如申請專利範圍第 1 項之方法，其中該組合語言層級的指令在由一處理器執行時使得該處理器進行：

從一基礎暫存器載入一第一數值；

將該第一數值加入到該記憶體位址偏置量來產生一第二數值；以及

參照由該第二數值所索引的一記憶體位址。

9. 一種用於自包括在一電腦程式的高階原始碼中一64位元類別轉換表達式擷取一記憶體位址偏置量之系統，該系統包含：

一處理器，其設置成：

接收該 64 位元類別轉換表達式，其中該 64 位元類別轉換表達式包括一或多個 32 位元表達式；

針對該等一或多個 32 位元表達式之每一者決定該 32 位元表達式的一範圍；

藉由加總該等32位元表達式的該等範圍來計算一整體範圍；

決定該整體範圍為一32位元無符號整數的一範圍的一

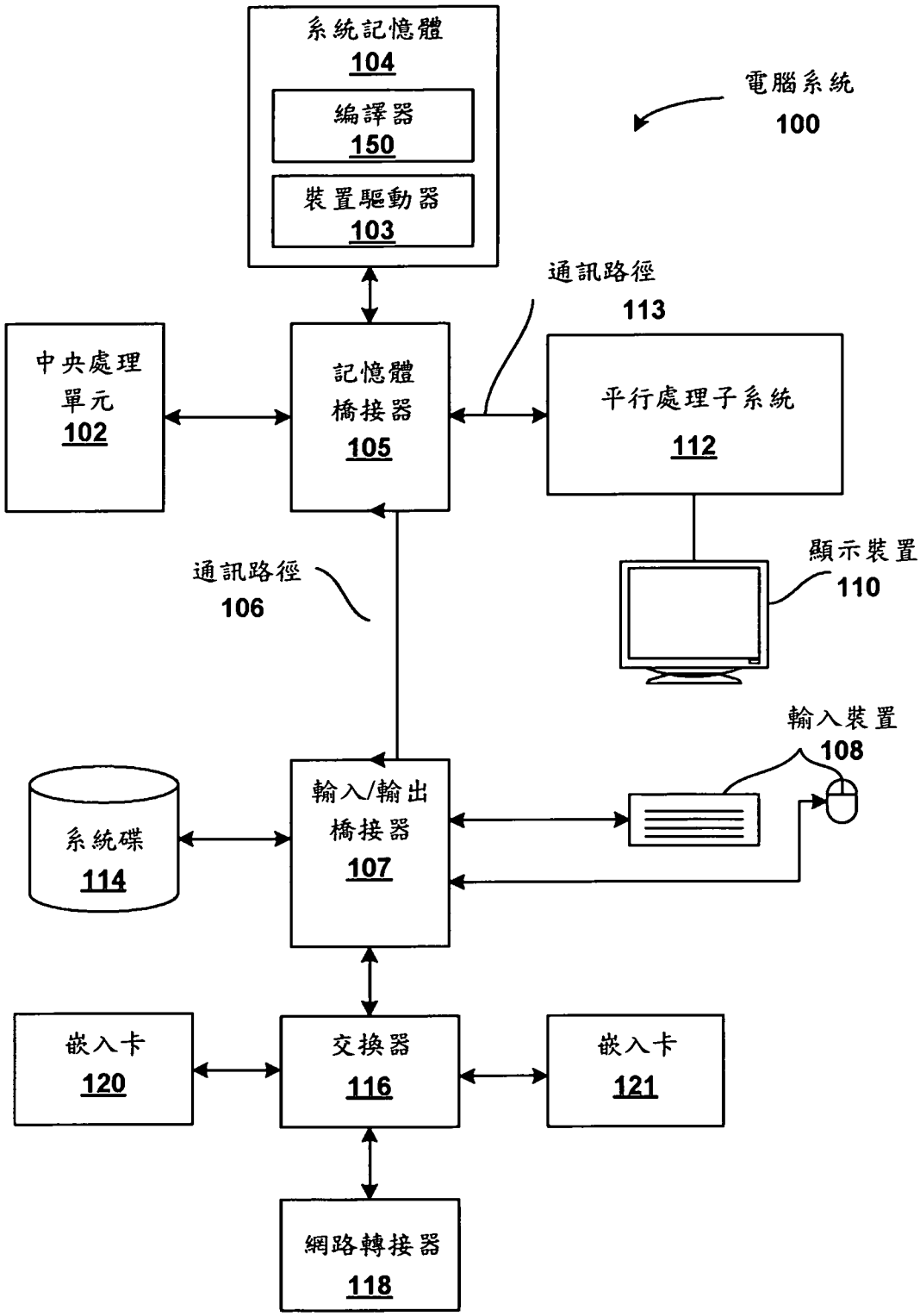
子集合；

基於該等一或多個32位元表達式的該等範圍計算該記憶體位址偏置量；以及

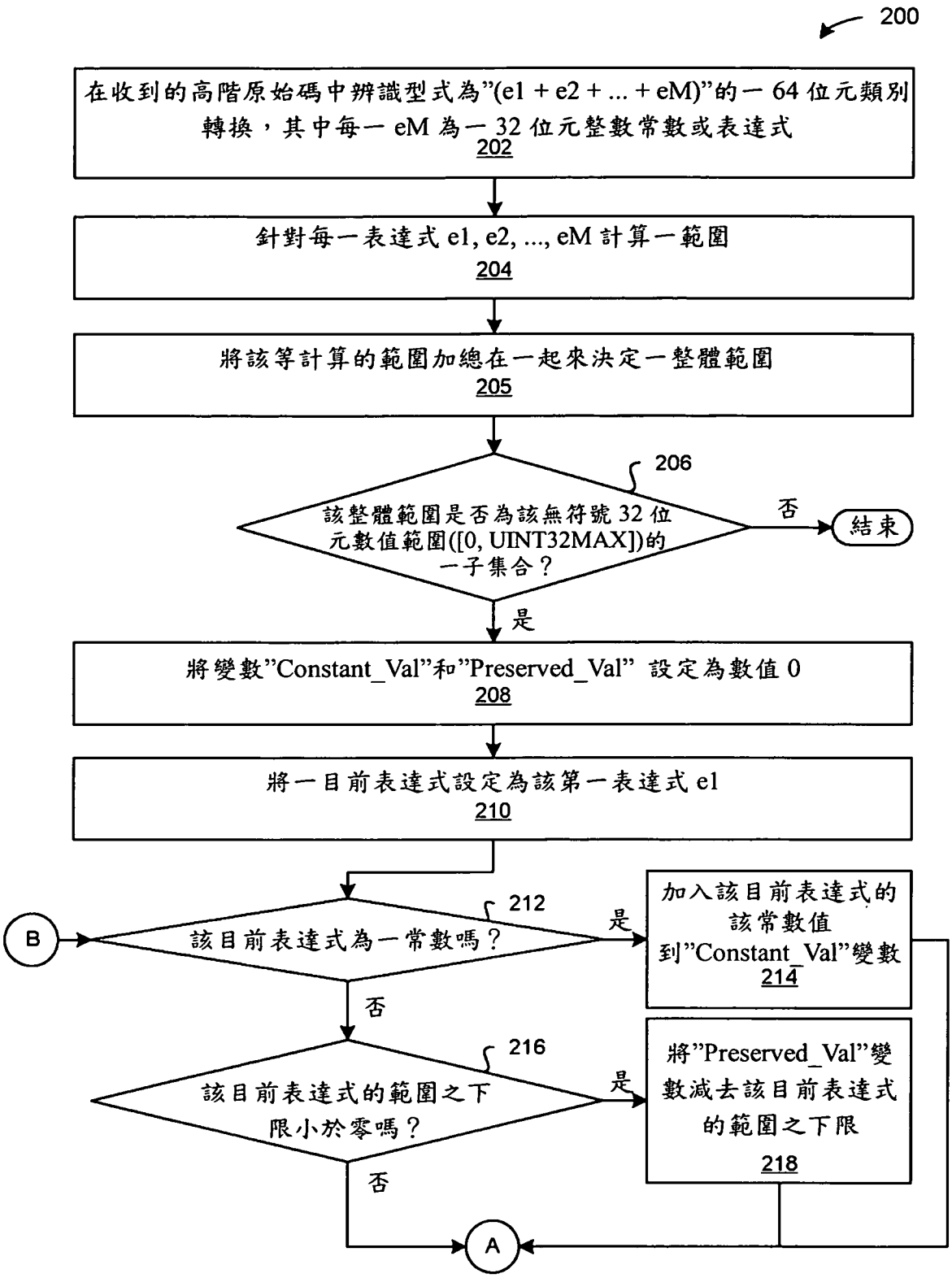
產生參照該記憶體位址偏置量的至少一組合語言層級的指令。

10. 如申請專利範圍第 9 項之系統，其中包括在該等一或多個 32 位元表達式中的一第一 32 位元表達式包含一常數值。

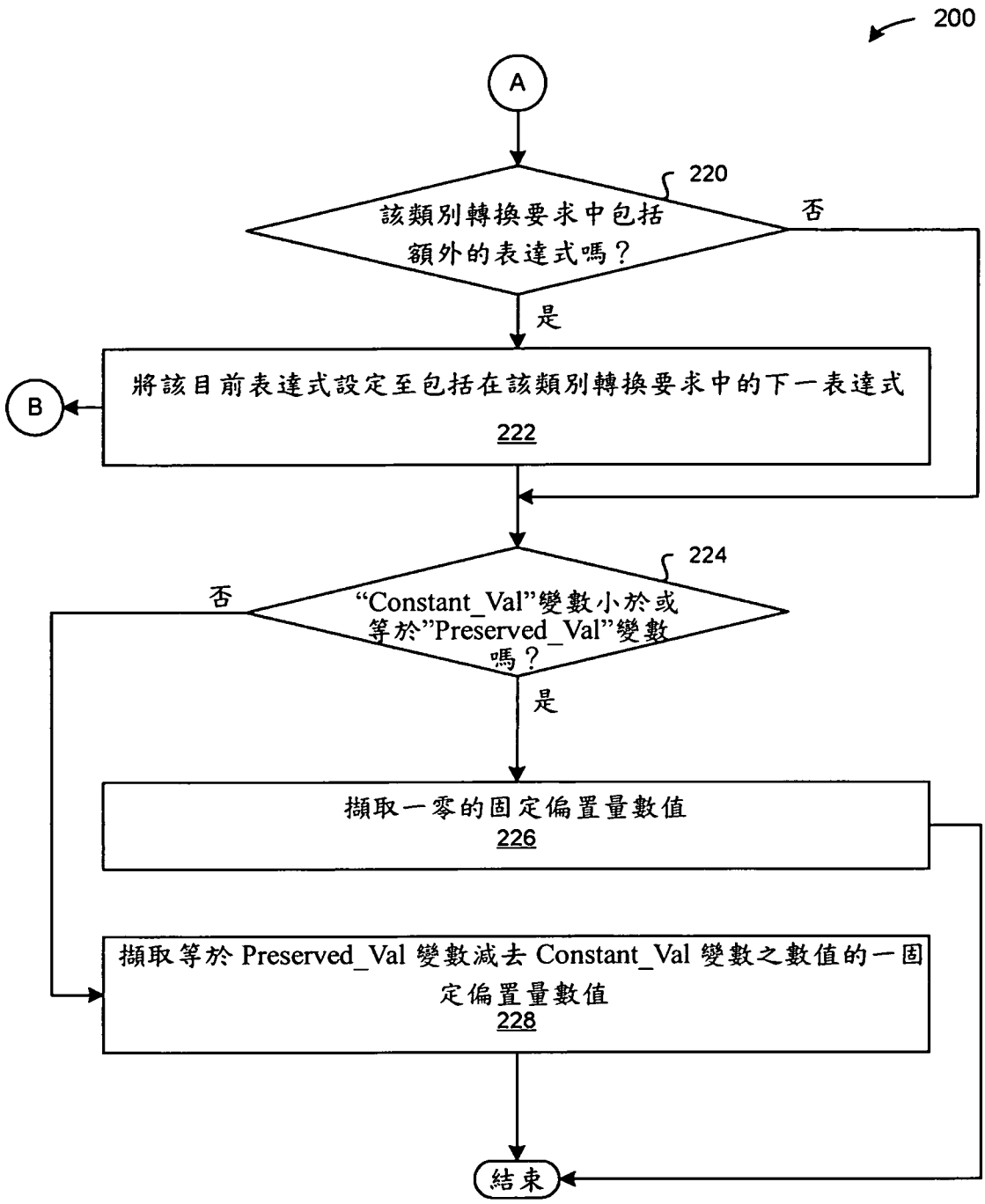
八、圖式：


系統記憶體
104
編譯器
150
裝置驅動器
103
電腦系統
100
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
顯示裝置
110
通訊路徑
106
輸入裝置
108
系統碟
114
輸入/輸出
橋接器
107
嵌入卡
120
交換器
116
嵌入卡
121
網路轉接器
118

第一圖


200
在收到的高階原始碼中辨識型式為"(e1 + e2 + ... + eM)"的一 64 位元類別轉換，其中每一 eM 為一 32 位元整數常數或表達式
202
針對每一表達式 e1, e2, ..., eM 計算一範圍
204
將該等計算的範圍加總在一起來決定一整體範圍
205
206
該整體範圍是否為該無符號 32 位元數值範圍([0, UINT32MAX])的一子集合？
否
結束
是
將變數"Constant_Val"和"Preserved_Val" 設定為數值 0
208
將一目前表達式設定為該第一表達式 e1
210
B
212
該目前表達式為一常數嗎？
是
加入該目前表達式的該常數值到"Constant_Val"變數
214
否
216
該目前表達式的範圍之下限小於零嗎？
是
將"Preserved_Val"變數減去該目前表達式的範圍之下限
218
否
A

第二 A 圖


200
A
220
該類別轉換要求中包括額外的表達式嗎？
否
是
B
將該目前表達式設定至包括在該類別轉換要求中的下一表達式
222
224
否
“Constant_Val”變數小於或等於”Preserved_Val”變數嗎？
是
擷取一零的固定偏置量數值
226
擷取等於 Preserved_Val 變數減去 Constant_Val 變數之數值的一固定偏置量數值
228
結束

第二B圖