200844740

發明專利說明書

（本說明書格式、順序及粗體字，請勿任意更動，※記號部分請勿填寫）

※ 申請案號：96150398

※ 申請日期：96.12.26　　※IPC 分類：~~G11C~~ G06F 12/02 (2006.01)

一、發明名稱：（中文/英文）

具有連續邏輯位址空間介面之直接資料檔案系統的使用

USE OF A DIRECT DATA FILE SYSTEM WITH A CONTINUOUS LOGICAL ADDRESS SPACE INTERFACE

二、申 請 人：（共 1 人）

姓名或名稱：（中文/英文）

美商桑迪士克股份有限公司
SANDISK CORPORATION

代表人：（中文/英文）

麗莎　K　托斯
TOTH, LIZA K.

住居所或營業所地址：（中文/英文）

美國加州謬佩塔斯市麥卡錫大道601號
601 MCCARTHY BOULEVARD, MILPITAS, CA 95035, U.S.A.

國　籍：（中文/英文）

美國　U.S.A.

三、發明人：(共 2 人)

姓　名：(中文/英文)

1. 艾倫　威爾斯　辛克萊兒
   SINCLAIR, ALAN WELSH
2. 貝瑞　懷特
   WRIGHT, BARRY

國　籍：(中文/英文)

1. 英國　U.K.
2. 英國　U.K.

四、聲明事項：

☐ 主張專利法第二十二條第二項☐第一款或☐第二款規定之事實，其事實發生日期為：　年　月　日。

☑ 申請前已向下列國家（地區）申請專利：

【格式請依：受理國家（地區）、申請日、申請案號　順序註記】

☑ 有主張專利法第二十七條第一項國際優先權：

1. 美國；2006年12月26日；11/616,242

2. 美國；2006年12月26日；11/616,236

☐ 無主張專利法第二十七條第一項國際優先權：

1.

2.

☐ 主張專利法第二十九條第一項國內優先權：

【格式請依：申請日、申請案號　順序註記】

☐ 主張專利法第三十條生物材料：

☐ 須寄存生物材料者：

國內生物材料　【格式請依：寄存機構、日期、號碼　順序註記】

國外生物材料　【格式請依：寄存國家、機構、日期、號碼　順序註記】

☐ 不須寄存生物材料者：

所屬技術領域中具有通常知識者易於獲得時，不須寄存。

九、發明說明：

【發明所屬之技術領域】

本申請案大體上係關於非揮發性記憶體系統(諸如，可再程式化半導體快閃記憶體)之操作以儲存資料且與連接之主機裝置轉移資料，且更特定言之，係關於其中資料檔案物件之管理。

【先前技術】

在較早代之商用快閃記憶體系統中，將記憶體單元之一矩形陣列劃分為大量的單元群組，每一者儲存一標準磁碟機磁區之資料量，亦即，512個位元組。每一群組中亦通常包括額外量之資料(諸如，16個位元組)來儲存一誤差校正碼(ECC)，且可能包括與使用者資料及/或與其中儲存使用者資料之記憶體單元群組有關的其他附加項資料。每一此群組中之記憶體單元為可一起擦除之最小數目的記憶體單元。亦即，擦除單位有效地為儲存一個資料區段及所包括之任何附加項資料之記憶體單元的數目。在美國專利第5,602,987號及第6,426,893號中描述此類型記憶體系統之實例。需要在用資料再程式化記憶體單元之前擦除記憶體單元為快閃記憶體之一特徵。

最通常以與諸如個人電腦、相機或其類似物之多種主機可移除地連接之記憶卡或隨身碟的形式提供快閃記憶體系統，但亦可在此等主機系統內嵌入快閃記憶體系統。當將資料寫入至記憶體時，主機通常對記憶體系統之一連續虛擬位址空間內的資料之區段、叢集或其他單位指派唯一邏

輯位址。如碟片作業系統(DOS)，主機將資料寫入至記憶體系統之邏輯位址空間內的位址並自該等位址讀取資料。記憶體系統內之控制器將自主機接收之邏輯位址轉譯為記憶體陣列內之實體位址(其中實際上儲存了資料)，且接著記錄此等位址轉譯。記憶體系統之資料儲存容量至少與可在對於記憶體系統所界定之整個邏輯位址空間上定址的資料之量一樣大。

在較後代之快閃記憶體系統中，將擦除單位的大小增大至足夠記憶體單元之區塊以儲存多個資料區段。即使與記憶體系統連接之主機系統可以諸如區段之小的最小單位程式化且讀取資料，亦將大量的區段儲存於快閃記憶體之單個擦除單位中。當主機更新或替換資料之邏輯區段時，一區塊內之資料的一些區段通常變得過時。因為必須在可覆寫區塊中所儲存之任何資料之前擦除整個區塊，所以通常將新資料或經更新資料儲存於已經擦除且具有用於資料之剩餘容量的另一區塊中。此過程使得原始區塊具有過時資料，其佔據記憶體內之寶貴空間。但是若存在剩餘在該區塊中之任何有效資料，則不能擦除該區塊。

因此，為了更好地利用記憶體之儲存容量，通常藉由將有效部分區塊量的資料複製至一已擦除區塊中來合併或收集該等資料，以使得接著可擦除自其複製此等資料之區塊且可重新使用其整個儲存容量。亦需要複製資料以便以一區塊內之資料區段之邏輯位址的次序來群聚該等資料區段，因為此增大了讀取資料並將經讀取資料轉移至主機的

速度。若此資料複製發生得太頻繁，則可降級記憶體系統之操作效能。特定言之，此在記憶體之儲存容量稍大於可經由系統之邏輯位址空間而藉由主機來定址的資料量(典型情況)之情況下影響記憶體系統的操作。在此情況下，在可執行主機程式化命令之前可要求資料合併或收集。接著增大程式化時間。

在逐代之記憶體系統中，區塊之大小不斷增大以便增大可儲存於給定半導體區域中之資料之位元的數目。儲存256個資料區段或256個資料區段以上之區塊變得普通。另外，不同陣列或子陣列之兩個、四個或四個以上區塊經常邏輯上一起連結成元區塊以增大資料程式化及讀取時之並行程度。與此等大容量操作單位一起到來的係有效地操作此等大容量操作單位的挑戰。

【發明內容】

上文交叉引用之專利申請案描述將由一主機提供之資料檔案物件直接儲存於快閃記憶體中之記憶體系統。此與大多數當前商用系統不同，其中在該主機與該記憶體系統之間的介面處存在一連續邏輯位址空間，如上文在先前技術中所描述。利用此"LBA介面"，個別資料檔案物件之資料最普遍地存在於大量記憶體單元區塊中。該記憶體系統未將由該主機提供之該等檔案物件之資料(通常各自為多個資料區段之叢集)與個別資料檔案物件相關聯。相反，該主機僅對供應至該記憶體系統以進行儲存之資料指派當前未對有效資料指派之在該LBA介面內之未用邏輯位址。該

記憶體系統接著指派其各個記憶體單元區塊以用使該記憶體系統有效地操作但不瞭解該等叢集所屬於之該等資料檔案物件的方式儲存經接收資料。一典型結果可為，將個別檔案物件之資料分段為在許多不同記憶體單元區塊中儲存之片段。

在上文交叉引用之專利申請案中之許多者中，另一方面，該記憶體系統不經由LBA介面而自該主機直接接收該等資料檔案物件，使得該記憶體系統可以改良該記憶體系統效能之方式將個別檔案之資料分配給其記憶體單元區塊。舉例而言，因為該資料所屬於之該檔案未知，所以該記憶體系統可限制儲存任一資料檔案之記憶體單元區塊的數目。特定言之，該記憶體系統可約束儲存一檔案物件之資料的亦含有另一檔案物件之資料之記憶體單元區塊的數目。因此，可控制檔案資料之分段。此最小化必須在一共同區塊之外重定位以回收過時資料空間之有效檔案資料的量，該過時資料空間在刪除或修改一儲存於該區塊中之第二檔案的資料時已產生。此導致在該快閃記憶體系統之使用壽命的顯著改良之效能及耐久性。

此改良之效能及耐久性亦可在將直接資料檔案管理系統實施於該主機而並非該記憶體系統的情況下實現。一LBA介面仍可存在於該主機與該記憶體系統之間。但並非將叢集中之檔案資料分配給此單個相連邏輯位址空間，而係將檔案資料分配給對應於在該記憶體系統內之實體區塊的在此空間內之邏輯位址的區塊。在主機內關於在該主機/記

憶體系統介面之該邏輯位址空間內之相連位址的邏輯區塊而替代進行在該快閃記憶體系統內關於實體記憶體單元區塊實施之在上文交叉引用之專利申請案中描述的該等檔案資料管理技術。該記憶體系統可接著為一如當前商業上普及之習知者，其具有一LBA介面。在該主機內之該直接資料檔案管理系統之操作可限制含有來自一個以上檔案之資料之邏輯區塊的數目，正如在該記憶體系統中操作之該直接資料檔案系統限制含有來自一個以上檔案之資料之實體記憶體單元區塊的數目。在實體記憶體單元區塊中之個別檔案物件之資料的分段類似地降低，但係藉由管理映射至實體記憶體單元區塊中之該邏輯位址空間的區塊而實現。

因此，在該LBA介面處之邏輯區塊較佳映射至該記憶體系統之具有相同資料儲存容量及其他相似性的實體區塊中。特定言之，若該直接資料檔案系統在該記憶體系統內操作，則由該主機將該等邏輯區塊組態成似乎其直接資料檔案系統與該等實體區塊將表現之相同。該等實體記憶體區塊之特徵(不通常供應至該主機之資訊)可由該記憶體系統在其初始化時提供至該主機。該主機接著將該連續邏輯位址空間組態成具有對應於該實體記憶體之彼等特徵之特徵的區塊中，且隨後將資料寫入至彼等邏輯區塊內之地址。

作為一替代，該直接資料檔案系統可以與上文描述相同之方式在具有越過記憶體系統之一LBA介面之連續位址空間而界定之邏輯區塊的記憶體系統中操作，而並非在一主

機中實施。即使為該記憶體系統之部分，此直接資料檔案操作仍不同於在上文交叉引用之專利申請案中描述之實例。並非在該記憶體系統之後端處以允許替換該LBA介面之該記憶體系統接收在檔案中之資料的方式來操作(在以上申請案中描述之實例)，該直接資料檔案系統可被添加至在該LBA介面之前的該記憶體系統且以似乎在其LBA介面之前的主機中之上文描述的相同方式操作。此記憶體系統可甚至經組態以提供該LBA介面及該檔案物件介面兩者，該記憶體系統經由該檔案物件介面可與一主機通信，該主機具有該LBA介面或該檔案物件介面但並非該兩種類型之介面。此特別便於在與許多類型之主機裝置可移除地連接之記憶卡中使用。

作為另一替代，一具有處理能力之可移除母卡可具備上文描述之該直接資料檔案系統，以便將直接檔案能力添加至一不具有該直接檔案能力但具有一直接資料檔案介面之主機。當與該主機連接時，該母卡接著操作以提供一在該卡之一輸出處之LBA介面，一具有一LBA介面之標準記憶卡可能可移除地與該卡連接。

本發明之額外態樣、優點及特徵包括於其例示性實例之以下描述中，該等例示性實例之描述應結合隨附圖式而理解。

本文中所引用之所有專利、專利申請案、論文、書籍、說明書、其他公開案、文件及事物茲出於所有目的皆全文以此引用之方式併入本文中。在所併入公開案、文件或事

物與本文件之本文之間的術語之定義或使用中有任何不一致或衝突之情況下，以在本文件中之術語的定義或使用為準。

【實施方式】

快閃記憶體系統之一般描述

參看圖1至圖6描述一種典型快閃記憶體系統。在該系統中，可實施本發明之多種態樣。圖1之主機系統1將資料儲存至快閃記憶體2中且自快閃記憶體2擷取資料。儘管可在主機內嵌入快閃記憶體，但仍將記憶體2說明為經由機械及電連接器之配合零件3及4可移除地連接至主機之卡的更普及形式。當前存在許多不同之市售快閃記憶卡，實例為緊密快閃(CF)、多媒體卡(MMC)、安全數位(SD)、迷你SD、記憶棒、智慧媒體及TransFlash卡。儘管此等卡中之每一者根據其標準化規格而具有唯一機械及/或電介面，但每一者中所包括之快閃記憶體系統為非常類似的。此等卡皆可自本申請案之受讓人SanDisk公司獲得。SanDisk亦在其Cruzer商標下提供一系列隨身碟，該等隨身碟為在小封裝中之手持式記憶體系統，其具有一用於藉由插入主機之USB插座中而與主機連接的通用串列匯流排(USB)插頭。此等記憶卡及隨身碟中之每一者包括與主機介接且控制該等記憶卡及隨身碟內之快閃記憶體之操作的控制器。

使用此等記憶卡及隨身碟之主機系統為許多且各種各樣的。該等主機系統包括個人電腦(PC)、膝上型及其他攜帶型電腦、蜂巢式電話、個人數位助理(PDA)、數位靜態相

機、數位電影相機及攜帶型音訊播放機。主機通常包括一內建式插座，其用於一或多種類型之記憶卡或隨身碟(但是一些要求將記憶卡插入其中之配接器)。

可將圖1之主機系統1視為具有兩個主要部分(在涉及記憶體2之範圍內)，其由電路與軟體之組合構成。該兩個部分為一應用程式部分5及一與記憶體2介接之驅動程式部分6。在個人電腦中，例如，應用程式部分5可包括一用於執行字處理、圖形、控制或其他普及應用軟體之處理器。在相機、蜂巢式電話或主要專用於執行單一組之功能的其他主機系統中，應用程式部分5包括操作相機以獲得並儲存圖像、操作蜂巢式電話以產生並接收呼叫及其類似操作的軟體。

圖1之記憶體系統2包括快閃記憶體7及電路8，該電路8與卡所連接至之主機介接以來回傳遞資料，且控制記憶體7。在資料程式化及讀取期間，控制器8通常在由主機1使用之資料的邏輯位址與記憶體7之實體位址之間轉換。

參看圖2，描述可用作圖1之非揮發性記憶體2之典型快閃記憶體系統的電路。系統控制器通常實施於單個積體電路晶片11上，該積體電路晶片11在系統匯流排13上與一或多個積體電路記憶體晶片並行連接，在圖2中展示單個此記憶體晶片15。所說明之特定匯流排13包括獨立的一組用於載運資料之導體17、一組用於記憶體位址之導體19及一組用於控制及狀態信號之導體21。或者，單一組之導體可在此三個功能之間為時間共用的。另外，可採用系統匯流

排之其他組態，諸如，在2004年8月9日申請之標題為"Ring Bus Structure and It's Use in Flash Memory Systems"的美國專利申請案序號第10/915,039號(公開案第2006/0031593 A1號)中所描述之環狀匯流排。

典型控制器晶片11具有其自身內部匯流排23，該匯流排23經由介面電路25與系統匯流排13介接。通常連接至匯流排之主要功能為處理器27(諸如，微處理器或微控制器)、含有用於初始化("開機")系統之程式碼的唯讀記憶體(ROM)29及主要用於緩衝在記憶體與主機之間轉移之資料的隨機存取記憶體(RAM)31。對經過在記憶體與主機之間的控制器之資料計算且檢查一誤差校正碼(ECC)之電路33亦可連接至匯流排23。亦可包括專用於編碼及解碼經過控制器之資料的電路34。此編碼包括壓縮及安全性加密，但幾乎任何類型之資料變換可以此方式執行。專用電路33及34在被利用時執行原本可由在韌體控制下之處理器27執行的特定演算法。控制器匯流排23經由電路35與主機系統介接，其在記憶卡中含有圖2之系統的情況下係經由為連接器4之部分的卡之外部接觸37而完成。時脈39與控制器11之其他組件中之每一者連接且由控制器11之其他組件中之每一者利用。

記憶體晶片15以及與系統匯流排13連接之任何其他晶片通常含有記憶體單元之陣列，該陣列經組織為多個子陣列或平面，為了簡單起見說明瞭兩個此等平面41及43，但可替代地使用更多(諸如，四個或八個)此等平面。或者，可

不將晶片15之記憶體單元陣列劃分為平面。然而，當如此劃分時，每一平面具有其自身之行控制電路45及47，該等行控制電路可彼此獨立地操作。電路45及47自系統匯流排13之位址部分19接收其各別記憶體單元陣列的位址，且將該等位址解碼以定址各別位元線49及51中之特定一或多者。回應於在位址匯流排19上接收之位址而經由列控制電路55來定址字線53。源電壓控制電路57及59亦與各別平面連接，p型井電壓控制電路61及63同樣與各別平面連接。若記憶體晶片15具有記憶體單元之單個陣列，且若在系統中存在兩個或兩個以上此等晶片，則可與在上述多平面晶片內之平面或子陣列類似地操作每一晶片之陣列。

經由與系統匯流排13之資料部分17連接之各別資料輸入/輸出電路65及67，將資料轉移至平面41及43內且將資料轉移至平面41及43之外。經由藉由各別行控制電路45及47連接至平面之線69及71，電路65及67將程式化資料提供至記憶體單元中且自其各別平面之記憶體單元讀取資料。

儘管控制器11控制記憶體晶片15之操作以程式化資料、讀取資料、擦除且注意各種內務處理事件，但每一記憶體晶片亦含有一些控制電路，其執行來自控制器11之命令以執行此等功能。介面電路73連接至系統匯流排13之控制及狀態部分21。將來自控制器之命令提供至狀態機75，接著該狀態機75提供對其他電路之特定控制以執行此等命令。如圖2中所示，控制線77至81連接狀態機75與此等其他電路。在線83上將來自狀態機75之狀態資訊傳遞至介面73，

以在匯流排部分21上傳輸至控制器11。

儘管亦可替代地使用諸如"反或"(NOR)之其他架構，但記憶體單元陣列41及43之"反及"(NAND)架構當前較佳。藉由參考美國專利第5,570,315號、第5,774,397號、第6,046,935號、第6,373,746號、第6,456,528號、第6,522,580號、第6,771,536號及第6,781,877號及美國專利申請公開案第2003/0147278號，可獲知作為記憶體系統之部分的NAND快閃記憶體及其操作的實例。

由圖3之電路圖說明NAND陣列之一實例，其為圖2之記憶體系統之記憶體單元陣列41的一部分。提供大量全局位元線，在圖2中為解釋簡單起見展示僅四條此等線91至94。若干串聯連接之記憶體單元串97至104連接於此等位元線中之一者與參考電位之間。使用記憶體單元串99作為代表，複數個電荷儲存記憶體單元107至110在串之任一末端處與選擇電晶體111及112串聯連接。當使一串之選擇電晶體導電時，該串連接於其位元線與參考電位之間。接著同時程式化或讀取該串內之一個記憶體單元。

圖3之字線115至118個別地延伸越過若干記憶體單元之串中之每一者中的一個記憶體單元之電荷儲存元件，且閘極119及120控制串之每一末端處之選擇電晶體的狀態。使得共用共同字線及控制閘極線115至120之記憶體單元串形成一起擦除之記憶體單元之區塊123。單元之此區塊含有最小數目之單元，該等單元可同時實體上擦除。每次程式化一列記憶體單元，彼等沿字線115至118中之一者的記憶

體單元。通常，以一規定次序程式化NAND陣列之列，在此狀況下，以沿最接近於串之連接至接地或另一共同電位的末端之字線118的列開始。接著在區塊123中沿字線117程式化記憶體單元之列，如此等等。最後程式化沿字線115之列。

第二區塊125為類似的，其記憶體單元之串與第一區塊123中之串連接至相同全局位元線，但具有一不同組之字線及控制閘極線。由列控制電路55將字線及控制閘極線驅動至其適當操作電壓。若在系統中存在一個以上之平面或子陣列(諸如，圖2之平面1及2)，則一個記憶體架構使用在其間延伸之共同字線。或者可存在共用共同字線之兩個以上平面或子陣列。在其他記憶體架構中，獨立地驅動個別平面或子陣列之字線。

如上文所參考之NAND專利及已公開之申請案中之若干者中所描述，可操作記憶體系統以將電荷之兩個以上可偵測位準儲存於每一電荷儲存元件或區域中，藉此以將資料之一個以上位元儲存於每一者中。如美國專利第6,925,007號中所描述，記憶體單元之電荷儲存元件最通常為導電浮閘，但或者可為非導電介電質電荷捕獲材料。

圖4概念地說明快閃記憶體單元陣列7(圖1)之組織，其在下文之進一步描述中用作一實例。記憶體單元之四個平面或子陣列131至134可處於單個積體記憶體單元晶片上、兩個晶片上(平面中之兩者處於每一晶片上)或四個獨立晶片上。特定配置對於下文之論述並不重要。當然，系統中

可存在其他數目之平面(諸如，1、2、8、16或16個以上)。將平面個別地劃分為在圖4中以矩形展示的記憶體單元之區塊(諸如，區塊137、138、139及140)，該等區塊位於各別平面131至134內。在每一平面中，可存在數打或數百個區塊。如上述，記憶體單元之區塊為擦除之單位，其為可一起實體上擦除的最小數目之記憶體單元。然而，為了增大之並列性，以較大元區塊單位操作區塊。將來自每一平面之一個區塊邏輯上連結在一起以形成一元區塊。將四個區塊137至140展示成形成一個元區塊141。通常一起擦除在一元區塊內之單元中之所有者。無需將用於形成一元區塊之區塊約束在其各別平面中之相同相關位置，如在由區塊145至148構成之第二元區塊143中所展示。儘管通常較佳越過平面中之所有者延伸元區塊，但為了高系統效能，可利用動態地形成在不同平面中之一個、兩個或三個區塊中之任一者或所有者之元區塊的能力來操作記憶體系統。此允許元區塊之大小更接近地與可用於在一程式化操作中儲存之資料量匹配。

出於操作目的，接著將個別區塊劃分為記憶體單元之頁，如圖5中所說明。將區塊131至134中之每一者之記憶體單元(例如)各劃分為八個頁P0至P7。或者，每一區塊內可存在記憶體單元之16個、32個或32個以上頁。頁為一區塊內之資料程式化及讀取的單位，其含有同時經程式化資料的最小量。在圖3之NAND架構中，一頁係由沿一區塊內之一字線的記憶體單元形成。然而，為了增大記憶體系統

操作並列性，可將兩個或兩個以上區塊內之此等頁邏輯上連結為元頁。在圖5中說明一元頁151，其係由來自四個區塊131至134中之每一者之一個實體頁形成。元頁151(例如)包括在四個區塊中之每一者中之頁P2，但元頁之頁無需必須具有在區塊中之每一者內的相同相對位置。儘管較佳越過所有四個平面來程式化且讀取最大量之資料，但為了高系統效能，亦可操作記憶體系統以形成在不同平面之獨立區塊中之一個、兩個或三個頁中之任一者或所有者之元頁。此允許程式化及讀取操作與可便利地並列處理之資料的量適應性地匹配，且減小元頁之部分保持未用資料程式化的機會。

一由多個平面之實體頁形成之元頁(如圖5中所說明)含有沿該等多個平面之字線列的記憶體單元。並非同時程式化一字線列中之所有單元，更通常將其在兩個或兩個以上交錯群組中交替地程式化，每一群組儲存資料之頁(在單個區塊中)或資料之元頁(越過多個區塊)。藉由同時交替地程式化記憶體單元，包括資料暫存器及感應放大器之周邊電路之單位無需為每一位元線提供，而是在鄰近位元線之間時間共用。此節約周邊電路所需要之基板空間的量且允許用沿列增大之密度來封裝記憶體單元。其他方面，較佳同時程式化沿一列之每一單元以最大化可自一給定記憶體系統獲得之並列性。

參看圖3，將資料同時程式化至沿一列之所有其他記憶體單元中係藉由沿NAND串之至少一末端提供兩列選擇電

晶體(未圖示)而非所展示之單列來最便利地實現。接著，一列之選擇電晶體回應於一控制信號而將區塊內之所有其他串連接至其各別位元線，且另一列之選擇電晶體回應於另一控制信號而將插入之所有其他串連接至其各別位元線。因此，將資料之兩個頁寫入至記憶體單元之每一列中。

在每一邏輯頁中之資料量通常為資料之整數的一或多個區段，按照慣例，每一區段含有資料之512個位元組。圖6展示頁或元頁之資料之兩個區段153及155的邏輯資料頁。每一區段通常含有經儲存之使用者或系統資料的512個位元組之一部分157，及用於與部分157中之資料或與實體頁或其中儲存實體頁之區塊有關之附加項資料的另一數目之位元組159。附加項資料之位元組量通常為16個位元組，從而使區段153及155中之每一者共有528個位元組。附加項部分159可含有在程式化期間自資料部分157計算出之ECC、其邏輯位址、區塊經擦除及再程式化之次數的經歷計數、一或多個控制旗標、操作電壓位準及/或相似內容，加上自此附加項資料159計算出之ECC。或者，附加項資料159或其之一部分可儲存於其他區塊中之不同頁中。

隨著記憶體之並列性增大，元區塊之資料儲存容量增大，且資料頁及元頁之大小亦因此增大。接著，資料頁可含有資料之兩個以上區段。在一資料頁中有兩個區段且每元頁有兩個資料頁的情況下，在一元頁中存在四個區段。

因此，每一元頁儲存資料之2048個位元組。此為高程度之並列性，且可隨著列中之記憶體單元之數目的增大而甚至進一步增大。鑒於此原因，快閃記憶體之寬度正經延伸以便增大一頁及一元頁中之資料量。

上文所識別之實體上較小之可再程式化非揮發性記憶卡及隨身碟為市售的，其具有512百萬位元組(MB)、1十億位元組(GB)、2 GB及4 GB之資料儲存容量，且可變得更高。

用於處理檔案物件之技術

具有邏輯區塊(LBA)記憶體/主機介面之操作

在圖7A、圖8A及圖9A中之每一者中以不同形式說明在主機與記憶體系統之間的共同邏輯介面。由主機對主機產生之資料檔案分配一連續系統位址空間(LBA介面)中之邏輯位址(通常以資料之多區段之叢集為單位)。接著，記憶體系統看見此等邏輯位址並將其映射至實際地儲存資料之記憶體單元之區塊的實體位址。

特別參看圖9A，連續邏輯位址空間161之大小足以為可儲存於記憶體系統中之所有資料提供位址。通常將主機位址空間劃分為資料之叢集的增量。在一給定主機系統中，每一叢集可經設計以含有資料之若干區段，大約在4與64個區段之間為典型的。一標準區段含有512個位元組之使用者資料，加上視需要之若干位元組的附加項資料(通常16個位元組)，共528個位元組。

圖9A說明在主機與(諸如)在記憶卡或隨身碟上得到之大

量記憶體系統之間的最常見介面。主機處理由主機所執行之應用軟體或韌體程式所產生或使用的資料檔案。"檔案"或"檔案物件"意謂由主機針對一些應用或目的而認識為界定實體的資料群組。將一檔案物件之資料作為一單位而管理。字處理檔案之資料為一實例，且電腦輔助設計(CAD)軟體之製圖檔案的資料為另一實例，其主要在諸如PC、膝上型電腦及其類似物之一般電腦主機中得到。以pdf格式之文件的資料亦為此檔案。檔案物件之資料可在執行應用程式期間由主機產生，或在別處產生且接著被提供至主機。靜態數位視訊相機產生用於儲存於記憶卡上之每一圖像的資料檔案。蜂巢式電話利用來自內部記憶卡上之檔案(諸如，電話號碼簿)的資料。PDA儲存且使用若干不同檔案(諸如，地址檔案、行事曆檔案及其類似物)。在任何此應用中，記憶卡亦可含有操作主機之軟體。

在圖9A之實例中，將三個檔案1、2及3展示成已產生。在主機系統上執行之應用程式將每一檔案產生為一有序組之資料且由唯一名稱或其他參考來識別該有序組之資料。由主機對檔案1指派尚未分配給其他檔案之足夠可用的邏輯位址空間。將檔案1展示成已被指派一相連區間之可用邏輯位址。亦為了特定目的共同地分配位址之區間(諸如，用於主機操作軟體之特定區間)，接著即使在主機對資料指派邏輯位址時尚未利用此等位址，該等區間亦避免用於儲存資料。

當由主機稍後產生檔案2時，主機類似地指派在邏輯位

址空間161內之兩個不同區間的相連位址，如圖7中所示。一檔案無需被指派相連邏輯位址，但可被指派在已分配給其他檔案之位址區間之間的位址的分段。接著，此實例展示對由主機產生之又一檔案3分配主機位址空間之其他部分，該等其他部分先前並未分配給檔案1與2或其他資料。

主機藉由維持檔案分配表(FAT)來記錄記憶體邏輯位址空間，在該檔案分配表中維持主機對各種主機檔案指派之邏輯位址。FAT表通常儲存於非揮發性記憶體以及主機記憶體中，且在儲存新檔案、刪除其他檔案、修改檔案及其類似操作時由主機頻繁地更新。舉例而言，當刪除主機檔案時，主機接著藉由更新FAT表來對先前分配給已刪除檔案之邏輯位址解除分配以展示該等邏輯位址現在可用於配合其他資料檔案使用。

主機並不關心記憶體系統控制器選擇來儲存檔案之實體位置。典型主機僅瞭解其邏輯位址空間及已分配給其各種檔案之邏輯位址。另一方面，記憶體系統經由典型LBA主機/卡介面而僅瞭解邏輯位址空間之已寫入資料的部分但不瞭解分配給特定主機檔案之邏輯位址，乃至主機檔案之數目。記憶體系統控制器將由主機所提供用於資料之儲存或擷取的邏輯位址轉換為儲存主機資料之快閃記憶體單元陣列內的唯一實體位址。區塊163表示此等邏輯至實體位址轉換之工作表，其係由記憶體系統控制器來維持。

記憶體系統控制器經程式化以將資料檔案以將系統之效能維持於高位準之方式儲存於記憶體陣列165之區塊及元

區塊中。在此說明中使用四個平面或子陣列。越過由來自平面中之每一者之區塊形成之整個元區塊，用系統允許之最大程度的並列性較佳地程式化且讀取資料。通常將至少一元區塊167分配為一保留區塊，其用於儲存操作韌體及由記憶體控制器所使用之資料。可分配另一元區塊169或多個元區塊以用於儲存主機操作軟體、主機FAT表及其類似物。大多數實體儲存空間保持用於儲存資料檔案。然而，記憶體控制器不瞭解經接收資料如何已由主機分配於其各種檔案物件中。記憶體控制器通常自與主機交互所瞭解的所有知識係：由主機寫入至特定邏輯位址之資料係儲存於對應實體位址中，該實體位址係由控制器之邏輯至實體位址表163來維持。

在典型記憶體系統中，提供儲存容量之少量額外區塊，其少於儲存在位址空間161內之資料量所必須的量。可將此等額外區塊中之一或多者作為冗餘區塊而提供，其用於取代可能會在記憶體之壽命內變得有缺陷之其他區塊。通常可為了各種原因改變個別元區塊內所含有之區塊的邏輯分組，原因包括用冗餘區塊取代對元區塊原始指派之有缺陷區塊。通常將一或多個額外區塊(諸如，元區塊171)維持於已擦除區塊集區中。當主機將資料寫入至記憶體系統時，控制器將由主機所指派的邏輯位址轉換為在已擦除區塊集區中之元區塊內的實體位址。接著擦除未用於儲存邏輯位址空間161內之資料之其他元區塊並將該等元區塊指定為已擦除集區區塊以便在隨後之資料寫入操作期間使

用。

當原始儲存之資料變得過時時，由新資料來頻繁地覆寫儲存於特定主機邏輯位址處之資料。作出回應，記憶體系統控制器將新資料寫入已擦除區塊中並接著改變用於彼等邏輯位址之邏輯至實體位址表以識別彼等邏輯位址處之資料所儲存至的新實體區塊。接著擦除含有在彼等邏輯位址處之原始資料的區塊並使該等區塊可用於儲存新資料。若在寫入開始時，在來自擦除區塊集區之預擦除區塊中不存在足夠的儲存容量，則此擦除經常必須在可完成當前資料寫入操作之前發生。此可不利地影響系統資料程式化速度。記憶體控制器通常瞭解，僅當主機將新資料寫入至其相同邏輯位址時，才藉由主機而使一給定邏輯位址處之資料過時。因此，記憶體之許多區塊可暫時儲存此無效資料。

在商用記憶體系統中利用之區塊及元區塊的大小不斷增大，以便有效使用積體電路記憶體晶片之面積。此導致大部分個別資料寫入儲存少於一元區塊之儲存容量(且在許多狀況下甚至少於一區塊之儲存容量)的資料量。因為記憶體系統控制器通常將新資料引導至已擦除集區元區塊，此可導致元區塊之部分變得未填充。若新資料為儲存於另一元區塊中之一些資料的更新，則亦以邏輯位址次序將來自具有與新資料元頁之邏輯位址相連之邏輯位址之另一元區塊的資料之剩餘有效元頁合意地複製至新元區塊中。舊元區塊可保持其他有效資料元頁。此隨著時間的過去而導

致使個別元區塊之特定元頁之資料過時且無效，且該資料由寫入至不同元區塊之具有相同邏輯位址的新資料替換。

為在整個邏輯位址空間161上維持用於儲存資料之足夠實體記憶體空間，週期性地將該資料壓縮或合併(垃圾收集)以便回收添加至已擦除區塊之集區的區塊。亦需要將資料之區段以與實際一般多之其邏輯位址相同的次序而維持於元區塊內，因為此使得在相連邏輯位址中讀取資料更有效。因此通常以此額外目的執行資料壓縮及垃圾收集。美國專利第6,763,424號中描述了在接收部分區塊資料更新且使用元區塊時管理一記憶體的一些態樣。

資料壓縮通常涉及自元區塊讀取所有有效資料元頁並將該等有效資料元頁寫入至新區塊，在該過程中忽略具有無效資料之元頁。亦較佳以匹配元頁中儲存之資料之邏輯位址次序的實體位址次序來配置具有有效資料之元頁。在新元區塊中所佔據之元頁的數目將少於在舊元區塊中所佔據之元頁的數目，因為未將含有無效資料之元頁複製至新元區塊。接著將舊區塊擦除且將其添加至已擦除區塊集區以可用於儲存新資料。藉由合併所獲得之容量的額外元頁接著可用以儲存其他資料。

在垃圾收集期間，自兩個或兩個以上元區塊聚集具有相連或接近相連邏輯位址之有效資料的元頁，並將該等元頁重新寫入至另一元區塊中(通常為已擦除區塊集區中之一者)。當自原始的兩個或兩個以上元區塊複製所有有效資料元頁時，可將該等有效資料元頁擦除以便將來使用。資

料合併及垃圾收集之發生率隨著在不同區塊中儲存檔案之分段的增大而增大。

資料合併及垃圾收集花費時間且可影響記憶體系統之效能，特定而言，在資料合併或垃圾收集需要在可執行來自主機之命令之前進行時。此等操作通常由記憶體系統控制器排程以儘可能在背景中進行，但執行此等操作之需要可導致控制器對主機提供忙狀態信號，直至完成此操作為止。可延遲主機命令之執行的一實例為：在已擦除區塊集區中不存在用於儲存主機想要寫入至記憶體中之所有資料的足夠預擦除元區塊，且首先需要資料合併或垃圾收集以清空有效資料(其可接著經擦除)之一或多個元區塊。因此，注意力已關注於管理記憶體之控制以最小化此等中斷。在以下美國專利申請案中描述了許多此等技術：在2003年12月30日申請之標題為"Management of Non-Volatile Memory Systems Having Large Erase Blocks"之序號第10/749,831號(現為公開案第2005/0144358 A1號)；在2003年12月30日申請之標題為"Non-Volatile Memory and Method with Block Management System"之序號第10/750,155號(現為專利第7,139,864號)；在2004年8月13日申請之標題為"Non-Volatile Memory and Method with Memory Planes Alignment"之序號第10/917,888號(現為公開案第2005/0141313 A1號)；在2004年8月13日申請之標題為"Non-volatile Memory and Method with Non-Sequential Update Block Management"之序號第10/917,867號(現為公

開案第2005/0141312 A1號)；在2004年8月13日申請之標題為"Non-Volatile Memory and Method with Phased Program Failure Handling"之序號第10/917,889號(現為公開案第2005/0166087 A1號)；在2004年8月13日申請之標題為"Non-Volatile Memory and Method with Control Data Management"之序號第10/917,725號(現為公開案第2005/0144365 A1號)；在2004年12月16日申請之標題為"Scratch Pad Block"之序號第11/016,285號(現為公開案第2006/0161722 A1號)；在2005年7月27日申請之標題為"Non-Volatile Memory and Method with Multi-Stream Update Tracking"之序號第11/192,220號(現為公開案第2006/0155921 A1號)；在2005年7月27日申請之標題為"Non-Volatile Memory and Method with Improved Indexing for Scratch Pad and Update Blocks"之序號第11/192,386號(現為公開案第2006/0155922 A1號)；及在2005年7月27日申請之標題為"Non-Volatile Memory and Method with Multi-Stream Updating"之序號第11/191,686號(現為公開案第2006/0155920 A1號)。

有效控制具有非常多擦除區塊之記憶體陣列之操作的一挑戰在於使在給定寫入操作期間儲存之資料區段的數目與記憶體之區塊的容量及邊界匹配且對準。一種方法為將所使用元區塊組態成利用少於最大數目之區塊來儲存來自主機之新資料，視需要儲存少於填滿整個元區塊之量的一定量資料。在2003年12月30日申請之標題為"Adaptive

Metablocks"之美國專利申請案序號第10/749,189號(現為公開案第2005/0144357 A1號)中描述適應性元區塊之使用。在2004年5月7日申請之標題為"Data Boundary Management"之專利申請案序號第10/841,118號(現為公開案第2005/0144363 A1)及在2004年12月16日申請之標題為"Data Run Programming"之序號第11/016,271號(現為公開案第2005/0144367 A1)中描述在資料區塊之間的邊界與在元區塊之間的實體邊界之配合。

記憶體控制器亦可使用來自FAT表(其由主機儲存於非揮發性記憶體中)之資料，以便更有效地操作記憶體系統。一種此使用在於瞭解資料何時藉由解除其邏輯位址分配而由主機識別為過時。知道此允許記憶體控制器在通常由將新資料寫入至彼等邏輯位址之主機瞭解含有此無效資料之區塊之前排程含有此無效資料之區塊的擦除。此係在2004年7月21日申請之標題為"Method and Apparatus for Maintaining Data in Non-Volatile Memory Systems"之美國專利申請案序號第10/897,049號中描述。其他技術包括監控將新資料寫入至記憶體之主機模式以推斷給定寫入操作為單個檔案還是邊界位於檔案之間的多個檔案。在2004年12月23日申請之標題為"FAT Analysis for Optimized Sequential Cluster Management"之美國專利申請案第11/022,369號描述此類型之技術的使用。

為有效地操作記憶體系統，需要控制器盡其最大努力地瞭解關於由主機對其個別檔案之資料指派的邏輯位址。資

料檔案接著可由控制器儲存於單個元區塊或元區塊之群組內，而並非在不瞭解檔案邊界時散布於大量元區塊中。結果在於減小了資料合併及垃圾收集操作之數目及複雜性。結果，記憶體系統之效能改良。但在主機/記憶體介面包括邏輯位址空間161(圖9A)時，記憶體控制器難以瞭解關於主機資料檔案結構之許多內容，如上文所描述。

直接資料檔案操作

在圖7B、圖8B及圖9B中展示之在主機與用於儲存大量資料之記憶體系統之間的不同類型介面排除邏輯位址空間之使用。而是主機藉由唯一檔案ID(或其他唯一參考)及檔案內之資料之單位(諸如，位元組)的偏移位址來邏輯上定址每一檔案。直接對記憶體系統控制器給出此等位址，該記憶體系統控制器接著保持關於其將每一主機檔案之資料實體上儲存於何處之自身表。此係作為上文交叉引用之專利申請案之主要目標的操作。可利用如上文參看圖2至圖6而描述之相同記憶體系統來實施此檔案介面。在圖7B、圖8B及圖9B之基於檔案之介面與圖7A、圖8A及圖9A之LBA介面之間的主要區別為記憶體系統與主機系統通信且儲存檔案資料之方式。

比較圖8B之基於檔案之介面與圖8A之LBA介面，圖8A之邏輯位址空間及主機維持之FAT表並未在圖8B中出現。相反，藉由檔案號碼及檔案內之資料的偏移來將由主機所產生之資料檔案識別至記憶體系統。記憶體系統接著直接將檔案映射至記憶體單元陣列之實體區塊。

當藉由直接資料檔案儲存技術將新資料檔案程式化至記憶體中時，將資料寫入記憶體單元之已擦除區塊中，以區塊中之第一實體位置開始且按次序順序地進行以穿過區塊之位置。以自主機接收之次序程式化資料，而不管檔案內之資料之偏移的次序。程式化繼續直至已將檔案之所有資料寫入至記憶體中為止。若檔案中之資料量超出單個記憶體區塊之容量，則接著當第一區塊變滿時，程式化在第二已擦除區塊中繼續。以與第一記憶體區塊相同之方式、以自第一位置開始之次序程式化第二記憶體區塊，直至儲存了檔案之所有資料或第二區塊變滿為止。可用檔案之任何剩餘資料程式化第三或額外區塊。用於儲存單個檔案之資料的多個區塊或元區塊不需要實體上或邏輯上相連。為了易於解釋，除非另有說明，否則希望本文中使用之術語"區塊"指擦除之區塊單位或多區塊式"元區塊"，此取決於元區塊是否正用於一特定系統中。

參看圖9B，將檔案1、檔案2及檔案3中之每一者之識別及該等檔案中之資料的偏移直接傳遞至記憶體控制器。接著由記憶體控制器功能173將邏輯位址資訊轉譯為記憶體165之元區塊及元頁之實體位址。並未將檔案資料映射至圖9A之邏輯位址空間161。

快閃最優化檔案系統之原理

圖7C、圖8C及圖9C以不同形式說明併入有圖7B、圖8B及圖9B之直接資料檔案技術且具有包括在圖7A、圖8A及圖9A中之類型的LBA介面的作業系統。圖7C之"快閃最優

化檔案系統"以與圖7B之"直接檔案儲存後端系統"實質上相同之方式操作，除了將檔案之資料映射至在圖7C中之LBA介面之連續位址空間內的邏輯區塊，而不是將檔案之資料映射至在圖7B中之NAND快閃之實體記憶體單元區塊。圖7C之LBA介面及"LBA至實體後端系統"為與圖7A之系統的共同之處。在圖7C之系統中，直接之檔案至區塊位址分配在LBA介面之前發生，但藉由在LBA介面之連續位址空間內之邏輯區塊位址而不是NAND快閃記憶體之實體區塊而工作。

在圖8C中，以一不同形式說明相同觀點。將由主機產生之資料檔案分配給在儲存裝置之邏輯位址空間內的邏輯區塊位址。邏輯位址空間之邏輯區塊接著由記憶體控制器以習知方式映射至實體儲存媒體之區塊。在圖8C中說明此等功能在主機與記憶體系統之間的兩個可能劃分。第一實施例將檔案分配給在主機(在圖中識別為主機1)中之邏輯區塊位址。接著，記憶體1為具有與主機之LBA介面連接之LBA介面的習知記憶卡或其他裝置。或者，圖8C之主機2將資料檔案識別及在檔案內之資料偏移與記憶體系統介接。接著在記憶體2內執行將此等檔案分配給邏輯區塊位址之直接資料檔案功能。

圖8C之記憶體2(最常見形式為記憶卡、隨身碟或其他較小的攜帶型單元)可藉由提供與用於儲存裝置之邏輯位址空間連接之外部連接而另外包括LBA介面。如另一替代，可在包括微處理器之母卡中執行將檔案分配給邏輯區塊位

址之功能。接著使母卡可移除地與主機2連接，且記憶體1將可移除地與母卡連接。

在圖9C中，以一不同方式說明將檔案物件之資料映射至邏輯位址空間之技術。功能173'接收個別檔案之資料，其具有唯一檔案識別符之個別邏輯位址及在該檔案內之資料的偏移位址。由功能173'將此等檔案位址轉換為連續邏輯位址空間161之邏輯區塊內的位址。取決於由實體記憶體利用之單位，將個別邏輯區塊之位址區間界定成具有記憶體陣列165之區塊或元區塊之相同資料儲存容量。圖9C之功能173'實質上與圖9B之功能173相同，除了在圖9C中將檔案映射至位址空間161內之邏輯區塊，而在圖9B中將檔案直接映射至記憶體單元陣列165。與圖9A中實質上相同，接著在圖9C中由功能163將邏輯位址區塊轉譯為記憶體陣列165。功能163可為習知快閃記憶體作業系統，諸如，在前述美國專利第7,139,864號及先前列出之以下已公開之專利申請案中所描述：2005/0141313 A1、2005/0141312 A1、2005/0166087 A1、2005/0144365 A1及2006/0161722 A1。

自圖9C應注意，位址空間161之個別邏輯區塊可含有來自一個以上檔案之資料。又，可對個別檔案之資料指派在一個以上邏輯區塊內之位址。舉例而言，對資料檔案2及3中之每一者指派在兩個或兩個以上邏輯區塊內之位址。邏輯區塊亦可含有兩個不同檔案之資料；圖9C之邏輯區塊2為此之一實例。但較佳地，在可含有給定檔案之資料以及

一些其他檔案之資料的邏輯區塊數目上有至少一極限。可在不同環境下利用不同極限。在一特定實例中，以任一檔案可與另一檔案之資料共用不多於兩個邏輯區塊之方式將檔案之資料分配給位址空間161之若干邏輯區塊。藉由約束允許用檔案之資料僅部分填充之邏輯區塊數目，可在對特定檔案物件之資料指派邏輯區塊位址期間遵循此約束。

此約束使(例如)由於其他檔案之資料隨後變得過時而可變得必要之資料重定位量保持為低。當此發生時，通常將給定檔案之有效資料自含有另一檔案之過時資料的區塊複製至另一區塊中。藉由約束給定檔案與另一檔案之資料共用之區塊數目，此等資料複製操作變得不頻繁。此改良記憶體系統之效能。

參看圖10，以邏輯區塊及實體區塊說明檔案資料之分配。出於說明目的而將實例實體記憶體單元區塊191劃分為四個頁195至199，但實際系統之每個區塊通常含有更多頁。每一頁儲存資料之多個區段。每次程式化資料之一頁，通常在區塊內以195至199之次序。若記憶體系統利用元區塊，則區塊191為元區塊且頁195至199為元頁。

邏輯位址空間161之邏輯區塊193被映射至實體區塊181中。將邏輯區塊193界定成具有與實體區塊191相同之資料儲存容量，且亦劃分為與實體區塊191數目相同之頁201至204，每一邏輯頁具有與實體頁195至199中之每一者相同之資料儲存容量。亦即，較佳使邏輯位址空間之粒度與實體記憶體頁或元頁之資料儲存容量相等。以與在實體區塊

191中寫入資料之頁相同之順序對資料指派在邏輯區塊193內之邏輯頁。在邏輯區塊193之第一頁201開始處寫入資料使得在實體區塊191之第一頁195開始處開始寫入資料。

為維持此邏輯功能與實體功能之協調，進行檔案至邏輯區塊轉譯之主機需要瞭解與該主機一起操作之記憶體的實體特徵。在使用元區塊之記憶體系統之一實例中，此等特徵可由以下參數界定：

1. 實體頁之大小，以實體頁所儲存之資料之區段的數目計；
2. 元頁之大小，以連結在一起以形成個別元頁之頁的數目計；
3. 每個元區塊之頁的數目；及
4. 映射至實體元區塊之第一頁之最低邏輯位址。

利用此資訊，主機可組態其邏輯位址空間161之邏輯區塊結構以用由圖10所說明之方式操作。若僅一種類型之記憶體(諸如，嵌入主機之記憶體)待由特定主機使用，則僅需維持主機之邏輯位址空間之一種組態。但更典型情況為，具有不同實體特徵之攜帶型記憶體裝置與給定主機裝置可移除地連接，實際上與許多不同主機裝置可移除地連接。因此，在主機內提供使該主機之邏輯區塊組態適應於與該主機連接之特定攜帶型記憶體裝置之實體區塊配置的能力。為實現此，以資料可由主機讀取之方式在記憶體裝置本身中儲存上文列出之記憶體參數的資料。記憶體系統之控制器通常改變任何特定邏輯區塊將映射至之實體區

塊，但主機並不瞭解此情況且此情況不影響主機對邏輯區塊指派檔案資料之位址。

圖11說明記憶體裝置207，其在可由主機211經由內連匯流排213存取之非揮發性儲存空間209中含有此等參數資料。存在可由主機讀取此等參數之許多方式。一實例為界定一供應商特定命令，其係在初始化記憶體裝置207期間由主機211發給記憶體裝置。記憶體裝置207接著操作以將經儲存參數值返回至該主機。另一實例為，此等參數可包括於記憶體裝置207回應於來自主機之現有標準命令而已返回至主機211的現有欄位之未用部分中。此命令之一實例為識別驅動(Identify Drive)命令。

例示性快閃最優化檔案系統

在此部分中提供將個別檔案映射至連續邏輯位址空間之邏輯區塊之技術的實例實施的進一步細節。其之某些態樣已參考實質上相同之以下功能而描述：圖7C之"快閃最優化檔案系統"、圖8C之"將檔案分配給邏輯區塊位址"及圖9C之"檔案/偏移至邏輯位址轉換"173'。

在此部分中描述之用於將檔案映射至邏輯區塊位址之許多內容利用在上文交叉引用之專利申請案中描述之將檔案映射至實體記憶體單元區塊位址的相同技術。主要不同在於，檔案映射係(諸如)由主機裝置越過LBA介面而進行，而並非繞過LBA介面藉由直接將資料檔案映射至實體記憶體區塊(如在先前交叉引用之專利申請案中所描述)而進行。可將先前申請案之實體記憶體區塊映射技術替代地應

用於將資料檔案物件映射至LBA位址空間之邏輯區塊，本文中描述其之一些實例。

在本文中對邏輯映射檔案物件之描述中，稱將資料"寫入至"或"程式化至"LBA介面之區塊。當然，與實體記憶體區塊相反，此等邏輯區塊實際上並不儲存資料，所以此指對特定邏輯區塊指定資料之位址。類似地，稱邏輯區塊在未被分配給資料時為"已擦除"。"已擦除"邏輯區塊為不含有資料之位址的邏輯區塊，所以其完全可被指派資料之位址。其他邏輯區塊可為"部分擦除"，此意謂該邏輯區塊之一部分可用於接收資料之額外位址。

快閃最優化檔案系統之一般操作

當新資料檔案待程式化至記憶體中時，將資料寫入至以在區塊中之第一位置開始之未佔據邏輯區塊中，且按次序順序地進行以穿過區塊之位置。以自主機接收之次序程式化資料，而不管檔案內之資料之偏移的次序。繼續程式化直至已寫入檔案之所有資料為止。若檔案中之資料量超過單個邏輯區塊之容量，則接著在第一區塊變滿時，程式化在第二空白(已擦除)區塊中繼續。以與第一邏輯區塊相同之方式、以自第一位置開始之次序程式化第二邏輯區塊，直至分配了檔案之所有資料或第二區塊變滿為止。可用檔案之任何剩餘資料程式化第三或額外區塊。用於儲存單個檔案之資料的多個邏輯區塊或元區塊無需相連。為了易於解釋，除非另有說明，否則希望本文中使用之術語邏輯"區塊"指具有與在記憶體系統內擦除之實體區塊最小單位

相同之容量的邏輯區塊或對應於通常一起擦除之多區塊式實體元區塊的多區塊式邏輯"元區塊"，此取決於元區塊是否正用於一特定系統中。

圖12之表說明快閃最優化檔案系統之全面功能。個別邏輯區塊可視為處於三種狀態中之一者中。此三種狀態為已擦除區塊641、儲存有效檔案資料而無可回收容量之區塊643及可含有一些有效檔案資料但亦具有來自未程式化(已擦除)頁面及/或儲存於其中之過時(無效)資料之可回收容量的區塊645。由功能647將資料寫入至已擦除邏輯區塊，藉此導致寫入至在種類643或645中之區塊，此取決於所得經程式化區塊是否保持任何可回收容量。當刪除檔案時，如由功能649所指示，含有檔案之資料的區塊643轉換為具有可回收容量之區塊645。在功能650中將資料自可回收區塊複製至其他區塊之後，區塊645之未用儲存容量由功能651回收，此導致使彼等區塊返回至可將新資料寫入至其之已擦除區塊641的狀態。

參看圖13A，其說明將資料檔案寫入至邏輯位址空間。在此實例中，資料檔案181大於一個區塊或元區塊183之儲存容量，將該資料檔案181展示成在垂直實線之間延伸。因此，亦將資料檔案181之一部分184寫入至第二區塊185中。將此等邏輯區塊展示成具有相連位址但並非必需。將來自檔案181之資料如同其為自主機接收之串流一樣寫入直至已將檔案之所有資料寫入至邏輯位址空間為止。在圖13A之實例中，資料181為檔案之初始資料。

一種用於記憶體系統管理且記錄經儲存資料之較佳方法為使用可變大小之資料群組。亦即，將檔案之資料儲存為複數個資料群組，可將該等資料群組以一界定次序鏈接在一起以形成完整檔案。當正寫入來自主機之資料流時，只要在檔案資料之邏輯偏移位址中或在分配給資料之邏輯位址空間中存在不連續性，即開始一新資料群組。此邏輯位址空間不連續性之一實例為在檔案之資料填滿一個邏輯區塊且開始寫入至另一區塊中時。此係在圖13A中說明，其中第一資料群組填滿第一區塊183，檔案之剩餘部分184作為第二資料群組儲存於第二區塊185中。第一資料群組可由(F0,D0)來表示，其中F0為資料檔案開始處之邏輯偏移，且D0為在邏輯區塊183內檔案開始處之位置。可將第二資料群組表示為(F1,D1)，其中F1為儲存於第二區塊185開始處之資料的檔案偏移，且D1為第二區塊開始處之對應邏輯位址。

經由主機記憶體介面而轉移之資料量可以資料之位元組的數目、資料之區段的數目或用一些其他粒度來表達。當經由當前邏輯位址介面與大容量記憶體系統進行通信時，主機最經常用位元組粒度來界定其檔案之資料，但接著將位元組分組為每一者512個位元組之區段或分組為每一者多個區段之叢集。通常進行此動作以簡化記憶體系統之操作。儘管本文中描述之基於檔案之主機記憶體介面可使用資料之一些其他單位，但原始主機檔案位元組粒度通常較佳。亦即，資料偏移、長度及其類似物較佳以位元組(資

料之最小合理單位)而非藉由區段、叢集或其類似物來表達。此允許用本文中描述之技術來更有效地使用快閃記憶體儲存器之容量。

接著將以在圖13A中說明之方式寫入至邏輯位址空間之新檔案在檔案索引表(FIT)中表示為關於資料群組之以如下次序的一連串索引項(F0,D0)、(F1,D1)。亦即，只要主機系統想要存取特定檔案，則主機產生其檔案ID或其他識別，接著存取其FIT以識別構成該檔案之資料群組。為了便於操作記憶體系統，個別資料群組之長度<length>亦可包括於其個別項中。

只要主機將圖13A之檔案維持於已打開狀態，則較佳地亦維持寫入指標P以界定用於針對該檔案寫入自主機接收之任何另外資料的邏輯位址。將用於檔案之任何新資料在邏輯區塊中之檔案的末端處寫入，而不管檔案內之新資料的邏輯位置。記憶體系統允許多個檔案(諸如，4或5個此等檔案)同時保持打開，且針對該等檔案中之每一者維持一寫入指標P。用於不同檔案之寫入指標指向不同邏輯區塊中之位置。若在關於打開檔案之數目之系統限制已存在時，主機系統想要打開一新檔案，則首先關閉已打開檔案中之一者且接著打開新檔案。

圖13B說明由主機將資料附加至圖13A之先前寫入但仍打開之檔案的末端。將資料187展示成由主機系統添加至檔案之末端，亦在第二區塊185中在該檔案之資料的末端處寫入該資料187。經附加資料成為資料群組(F1,D1)之部

分，因此該資料群組現含有更多資料，因為在現有資料群組184與經附加資料189之間不存在檔案不連續性且不存在邏輯位址不連續性。因此，仍將完整資料表示為在FIT中之一連串索引項(F0,D0)、(F1,D1)。亦將指標P之位址改變為儲存經附加資料之末端的位址。

在圖13C中展示將資料191之區塊插入至圖13A之先前寫入檔案中的一實例。儘管主機將資料191插入至檔案中，但快閃最優化檔案系統在先前寫入檔案資料之末端處的位置193處附加經插入資料。無需在將資料插入至打開檔案中時以其邏輯次序重寫入檔案之資料，儘管此可稍後在主機關閉檔案之後在背景中進行。由於將經插入資料完全儲存在第二邏輯區塊185內，故其形成單個新群(F1,D3)。但進行此插入導致將圖13A之前一資料群組(F0,D0)劃分為兩個群組，一個在插入之前的群組(F0,D0)及一個在插入之後的群組(F2,D1)。此因為，只要存在資料之檔案不連續性(諸如，在插入之開始F1處及在插入之末端F2處出現之不連續性)，則需要形成新資料群組。群組(F3,D2)係作為第二區塊185開始之邏輯位址D2的結果。即使群組(F1,D3)及群組(F3,D2)儲存在同一邏輯區塊中，但仍將群組(F1,D3)及群組(F3,D2)維持為獨立，因為在該等群組中儲存之資料的檔案偏移存在不連續性。接著在FIT中由以如下次序之資料群組索引項(F0,D0)、(F1,D3)、(F2,D1)、(F3,D2)表示具有插入之原始檔案。自圖13A、圖13B及圖13C之實例應注意，可在並不使由邏輯區塊位址表示之任

何資料變得過時的情況下寫入用於新或現有檔案之新資料。

作為在圖13C中說明之將資料插入至現有檔案中的替代，只要已插入資料，則可由主機將檔案作為獨立檔案而重寫入。此獨立檔案接著可由記憶體系統作為新檔案而處理。接著由主機刪除舊檔案，且系統可藉由回收對經儲存舊檔案(其之資料現為過時)指派之邏輯位址空間而回應。

圖13D說明另一實例，其中更新以在圖13A中展示之方式原始寫入之資料的特定部分。將資料檔案之一部分195展示成待更新。將檔案之一更新部分197附加至先前寫入之資料，而並非重寫入具有更新之整個檔案。先前寫入之資料的一部分199現為過時。在更新之後，檔案在系統FIT中係由以如下次序之資料群組索引項(F0,D0)、(F1,D3)、(F2,D1)、(F3,D2)表示。在圖13D中，再次將圖13A之單個資料群組(F0,D0)劃分為幾塊：一個在經更新部分之前的群組、經更新部分及一個在經更新部分之後的群組。需要回收由過時資料佔據之位址空間199，但此較佳稍後進行而不作為寫入檔案資料之部分而進行。此回收通常導致儲存用於特定檔案之更少量之資料群組。

為了進一步說明可變長度資料群組之使用，由圖14A至圖14E按次序展示涉及相同檔案之一連串若干寫入操作。首先將原始檔案資料W1寫入至連續位址空間之兩個邏輯區塊中，如圖14A中所示。接著由兩個資料群組界定檔案，第一群組在邏輯區塊之開始處開始，且在邏輯區塊邊

界之後需要第二群組。圖14A之檔案接著由用於資料群組之索引項的以下序列描述：(F0,D0)、(F1,D1)。

在圖14B中，主機使在圖14A中寫入之檔案資料更新。緊接在前一群組(F1,D1)之後寫入經更新檔案資料U1，其中經更新資料之前一版本變為過時。圖14A之前一群組(F0,D0)縮短為圖14B之經修改群組(F0,D0)，且前一群組(F1,D1)縮短為群組(F4,D2)。因為經更新資料與邏輯區塊之邊界重疊，因此在兩個群組(F2,D3)與(F3,D4)中寫入經更新資料。在第三邏輯區塊中儲存一些資料。檔案現由用於資料群組之索引項的以下序列描述：(F0,D0)、(F2,D3)、(F3,D4)、(F4,D2)。

在圖14C中，圖14B之檔案由使新檔案資料I1之插入的主機進一步修改。因為經插入資料與邏輯區塊之邊界重疊，所以將新資料I1作為圖14C之新群組(F5,D6)及(F6,D7)而寫入至緊接在圖14B之前一群組(F4,D2)之後的邏輯區塊中。使用第四邏輯區塊。在圖14C中，因為新資料I1之插入，所以將圖14B之前一群組(F0,D0)分割為縮短群組(F0,D0)及(F7,D5)。檔案現由用於資料群組之索引項的以下序列描述：(F0,D0)、(F5,D6)、(F6,D7)、(F7,D5)、(F8,D3)、(F9,D4)、(F10,D2)。

圖14D展示對圖14C之資料檔案的進一步修改：將新資料W2附加至該檔案之末端。將新資料W2作為圖14D之新群組(F11,D8)而緊接在圖14C之前一群組(F10,D2)之後寫入。檔案現由用於資料群組之索引項的以下序列描述：

(F0,D0)、(F5,D6)、(F6,D7)、(F7,D5)、(F8,D3)、(F9,D4)、(F10,D2)、(F11,D8)。

在圖14E中展示對打開檔案之第二更新，其中將經更新檔案資料U2寫入至圖14D之檔案中。在圖14E中，經更新資料U2緊接在圖14D之前一群組(F11,D8)之後寫入，其中該資料之前一版本變為過時。圖14D之前一群組(F9,D4)縮短為在圖14E中之經修改群組(F9,D4)，前一群組(F10,D2)變得完全過時，且前一群組(F11,D8)經縮短以形成新群組(F14,D9)。在圖14E之新群組(F12,D10)及(F13,D11)中寫入與邏輯區塊邊界重疊之經更新資料。現在，檔案需要第五邏輯區塊。檔案現由用於資料群組之索引項的以下序列描述：(F0,D0)、(F5,D6)、(F6,D7)、(F7,D5)、(F8,D3)、(F9,D4)、(F12,D10)、(F13,D11)、(F14,D9)。

在根據前文描述之檔案建立或修改之後，較佳以正確邏輯次序使每一檔案之資料的偏移維持連續。因此，舉例而言，作為將資料插入至檔案中之操作的一部分，由主機提供之經插入資料的偏移與緊接在該插入之前的偏移連續，且已在該插入之後的檔案中之資料增大經插入資料之量。更新現有檔案最通常導致在現有檔案之給定位址區間內的資料由相同量之經更新資料所替換，因此一般無需替換檔案之其他資料之偏移。

可將如此儲存之資料之粒度或解析度維持為與主機之粒度或解析度相同。舉例而言，若主機應用用一位元組粒度寫入檔案資料，則在邏輯區塊中亦可用一位元組粒度來表

示檔案。接著在若干位元組中量測資料在個別資料群組內之量及位置。亦即，可在主機應用檔案內獨立地定址之資料的相同偏移單位亦可在該檔案儲存於快閃記憶體中時在該檔案內獨立地定址。接著，在FIT中將在一邏輯區塊內之同一檔案之資料群組之間的任何邊界指定給最接近之位元組或其他主機偏移單位。類似地，以主機偏移之單位界定在一邏輯區塊內之不同檔案之資料群組之間的邊界。

術語"區段"在本文中結合大區塊記憶體使用，以表示與ECC相關聯之經儲存資料的單位。因此，區段為在此誤差校正碼由記憶體系統之控制器產生且用資料儲存時，傳遞至快閃記憶體且自快閃記憶體傳遞之資料的最小單位。當參考實體記憶體時，"頁"用於表示在一區塊內之記憶體單元的單位。頁為程式化之最小單位。在邏輯區塊內之邏輯"頁"為含有與實體頁相同之資料量的頁。術語"元頁"用於表示具有一元區塊之完全並行性的頁。元頁為程式化之最大單位。

自圖14B及圖14E應注意，經更新命令導致由檔案所占之邏輯位址空間大於在檔案中之資料量。原因為留存用於已由更新替換之資料的邏輯位址。因此，非常需要藉由消除過時之無效資料而將檔案之資料合併(垃圾收集)至較少邏輯位址空間中。因此，更多邏輯位址空間變得可用於其他資料。

亦可注意，除了圖14B及圖14E之檔案資料更新，圖14C之資料插入亦導致檔案資料之位址失序。亦即，雖然幾乎

始終在檔案內某處定位更新及插入，但更新及插入在其進行時被添加至檔案之末端。此為圖14B、圖14C及圖14E之實例的情況。因此，可能會需要越過邏輯位址空間對檔案之資料重排序以匹配偏移在檔案內之次序。因為順序地讀取頁及區塊將給予以其偏移次序之檔案的資料，所以此接著改良讀取經儲存資料之速度。此亦提供檔案之最大可能之重組。但對檔案資料重排序以使讀取更有效對記憶體系統之效能的重要性並不如檔案資料合併，檔案資料合併潛在地釋放用於其他資料之位址的一或多個邏輯區塊。因此，對檔案中之資料重排序一般並非由其自身進行(其中優點不值添加之操作負擔)，但可作為具有極少或不具有添加之操作負擔之許多垃圾收集操作的一部分而進行。

因為已進行兩個資料更新U1及U2，所以圖14E之檔案包括過時檔案群組(灰色部分)。結果，自圖14E顯而易見，分配給檔案之邏輯位址空間量之實質上大於檔案之大小。因此，垃圾收集係合適的。圖15提供垃圾收集圖14E之資料檔案之結果的說明。在垃圾收集之前，該檔案佔據位址空間之接近五個邏輯區塊(圖14E)，而在垃圾收集之後，同一檔案在稍多於三個區塊(圖15)內。作為垃圾收集操作之部分，將資料自其初始寫入至之邏輯區塊複製至其他已擦除邏輯區塊中，且接著擦除原始區塊。若整個檔案經垃圾收集，則可將其資料複製至新區塊中，該等新區塊具有與在該檔案內之資料邏輯偏移次序相同之邏輯次序。舉例而言，更新U1及U2以及插入I1在垃圾收集之後(圖15)以與

其在主機檔案中出現之相同次序儲存。

基於檔案之垃圾收集亦通常導致在合併之檔案內形成新的且不同的資料群組。在圖15之情況下，檔案係由用於新資料群組之索引項的以下新序列描述：(F0,D12)、(F1,D13)、(F2,D14)、(F3,D15)。此比在圖14E中展示之檔案的情形下存在之資料群組的數目少得多。現在針對區塊中之每一者存在一個資料群組，已將檔案之資料複製至該等區塊中。作為垃圾收集操作之部分，更新FIT以反映形成檔案之新資料群組。

當在圖14E之狀態中時，回收固持檔案之資料的區塊係對區塊個別地操作而並非對儲存同一檔案之資料的多個區塊操作。舉例而言，若考慮在某給定時間對圖14E之含有位址空間之任何區塊之最少量有效資料的第二區塊002進行回收操作，則接著將其單個資料群組複製至另一已擦除區塊。新區塊接著含有單個資料群組(F8,D16)，且該區塊之剩餘部分為已擦除容量，可將新資料寫入至該已擦除容量中。已自區塊(在圖14E中，資料儲存於該區塊中)回收該已擦除容量。檔案接著由用於構成該檔案之資料群組之索引項的以下序列描述：(F0,D0)、(F5,D6)、(F6,D7)、(F7,D5)、(F8,D16)、(F9,D4)、(F12,D10)、(F13,D11)、(F14,D9)。在圖14E中展示之其他區塊保持不變，直至其個別地符合用於回收操作之標準為止。

檔案區塊管理

基於儲存於區塊中之檔案資料之結構來認識邏輯區塊之

特定類型。接著注意具有在連續位址空間中之位址的每一檔案處於若干狀態中之一者，每一檔案狀態係由儲存檔案之資料之區塊的數目及類型而界定。對一檔案而言，當有資料待寫入時，該檔案當前狀態及自一狀態至另一狀態之准許轉變較佳經控制以約束含有用於特定檔案之資料的區塊數目，該特定檔案亦含有一或多個其他檔案之資料。此促進邏輯區塊之有效利用，且降低維持用於接受新資料或經複製資料之足夠已擦除區塊所必需之稍後回收操作的頻率。

在此實例中認識之含有檔案資料之邏輯區塊的核心類型如下：

"檔案區塊"已完全程式化，且表示單個檔案之有效資料。其亦可含有一些過時資料之位址。

"程式化區塊"已部分程式化，且表示僅單個檔案之有效資料。一些已擦除容量仍留存在該區塊中。其亦可含有一些過時資料之位址。

"共同區塊"已部分程式化，且表示兩個或兩個以上檔案之有效資料。仍留存一些已擦除容量。其亦可含有一些過時資料之位址。

"滿的共同區塊"已完全程式化，且表示兩個或兩個以上檔案之有效資料。其亦可表示一些過時資料。

另一類型之區塊為"已擦除區塊"，其中在該區塊中不存在資料位址，因此該區塊之全部容量可用於接受資料。當LBA介面之邏輯位址空間變滿或幾乎充滿資料位址時，通

常藉由連續地回收存在於正使用之邏輯區塊內之未用容量來維持特定最小數目之已擦除區塊的集區。

"碎形區塊"為一集體術語，指程式化區塊、共同區塊或滿的共同區塊。用於檔案之碎形區塊含有該檔案之有效資料，以及未經程式化之儲存容量、用於其他檔案之有效資料或該兩者。本文中描述之技術的主要目的為藉由管理經指定以接收檔案之資料之作用區塊的類型來最小化在位址空間中之碎形區塊的數目。此減少需要在邏輯位址空間中執行垃圾收集及資料合併(區塊回收操作)以維持特定最小數目之已擦除邏輯區塊的情況。因為花費更少時間來內部複製資料以回收在先前經程式化區塊中之未用容量的分段，所以接著使可將資料寫入至記憶體中之速率增大。

亦在本文中使用額外術語以集中描述區塊之其他類型：

"部分區塊"含有一些未經程式化容量、一或多個檔案之有效資料之位址，且可表示一些過時資料。程式化區塊及共同區塊為部分區塊之實例。

"過時區塊"為含有一些過時資料之位址的檔案區塊或滿的共同區塊。過時區塊不具有任何已擦除容量，且表示有效資料及無效資料兩者。

"無效區塊"不含有有效資料。無效區塊含有至少一些過時資料之位址，且可含有已擦除容量但不表示任何有效資料。

圖16A至圖16D說明使用上文定義之邏輯區塊之類型的一些實例。在圖16A中，檔案A之資料已填滿區塊661及

663，且部分填充第三區塊665。在此實例之每一區塊中從左至右地寫入資料，首先填充區塊661，接著填充區塊663且隨後寫入至區塊665之一部分中。區塊665之剩餘部分為可儲存額外資料之未經程式化之已擦除容量。藉由上文列出之定義，區塊661及663為檔案區塊，且區塊665為程式化區塊。在程式指標P處開始將任何新資料寫入至區塊665中。指標P隨著將資料寫入至區塊而從左至右地移動，以始終指向在區塊中之下一可用儲存位置。為保持未經程式化之已擦除容量之個別區塊(無論當前是否為作用)保持該指標，使得始終瞭解待寫入至區塊之任何其他資料的邏輯位址。

圖16B之實例包括為共同區塊之區塊669，因為該區塊699含有另一檔案B之資料以及當前檔案A之資料加上一些未經程式化容量。在展示程式指標P處開始，在檔案A之末端處將新資料寫入至區塊669中。區塊669為檔案A之作用區塊。該區塊699亦可為用於檔案B之作用區塊，在此情況下，可在程式指標P處寫入檔案A或檔案B之額外資料。或者，一獨立區塊(未圖示)可為檔案B之作用區塊。

可將檔案之資料直接寫入至已含有其他檔案之資料之部分區塊的已擦除容量中，而並非寫入至已擦除區塊中，以便較佳此形式使用未經程式化容量。此在待寫入小於一滿區塊容量之已知量的檔案資料時特別有用。搜尋現有部分區塊以找到適合待寫入之已知資料量的適量已擦除容量。資料之頁(或在使用元區塊時之元頁)的數目與在部分區塊

中之未經程式化容量之頁的數目比較。當以此方式程式化程式化區塊之未用已擦除空間，則該區塊轉換為共同區塊。

在圖16C中，檔案A儲存於檔案區塊661、區塊671之一部分及區塊673之一部分中。區塊671為滿的共同區塊，因為該區塊671充滿兩個檔案A及B之資料。區塊673為類似於圖16A之區塊665的程式化區塊。區塊673為檔案之作用區塊，且指標P指向在區塊673內首先寫入額外資料處之未用容量的位置。

在圖16D之實例中，將檔案A寫入至滿的共同區塊671之一部分及共同區塊675中。區塊675含有第三檔案C之資料。指標P指向作用區塊675之寫入額外資料處之未用部分的第一位置。

儘管圖16A至圖16D之實例展示儲存於多個區塊中之檔案A的資料以說明區塊之若干不同類型，但在許多情況下，檔案可足夠小以儲存於更少數目之區塊(甚至單個區塊)中。本文中描述之技術亦可應用於此等小檔案。又，較大檔案可佔據三個以上區塊中之頁。

應注意，邏輯區塊665、669、671、673及675為碎形區塊。需要最小化由任一檔案之資料佔據之碎形區塊的數目，因為其存在增大需要回收在該等碎形區塊中之未用容量之可能性且因此不利地影響系統效能。未用已擦除容量存在於部分邏輯區塊665、669、673及675中，但除非已知檔案之未寫入資料量且該已知量匹配此等區塊中之一者的

未用容量，否則將新資料自主機直接寫入至此空間內並沒有效率。最常見地，未知特定資料之來自主機的資料量，因此不易於填充容量之此等位元。因此，可需要在回收操作期間將資料自另一區塊移動至未用空間內，以有效使用記憶體容量。區塊669、671及675含有一個以上檔案之資料，其意謂，當刪除該等檔案中之一者或其儲存於共同區塊中之資料變得過時，則可能進行資料回收以回收由過時資料之位址佔據之區塊容量。

因此，為降低資料回收操作花費之時間量，允許特定檔案之資料在任一時間儲存於僅一個、兩個或某個另一數目之碎形區塊中。在判定准許之碎形區塊數目時，權衡能夠使用該等碎形區塊之優點與具有該等碎形區塊之不良影響。在本文中描述之特定實例中，任一檔案之資料可儲存於兩個或少於兩個碎形區塊中，但不多於兩個。因此，指定新作用區塊以儲存檔案之資料的處理受約束。對每一檔案指派一組准許檔案狀態中之一者，其係由儲存檔案之資料之區塊類型所界定。當需要指派將新作用區塊以接收特定檔案之資料時，諸如當一現有區塊變為滿時，如此指定之區塊類型取決於檔案之狀態，且在許多情況下亦取決於其他因素。

在圖17之表中給出在一特定實施中根據含有用於檔案之資料之碎形區塊的組合之七個准許檔案狀態00至20的定義。准許檔案狀態中之每一者允許在不超過兩個碎形區塊中儲存檔案。對可儲存檔案之資料之檔案區塊的數目不存

在約束。檔案之狀態為用於控制對將用作檔案之作用區塊之區塊的選擇的特性。已選擇為回收區塊之區塊並不視為碎形區塊，因為存在於該回收區塊中之用檔案的任何資料並不有助於判定檔案之狀態，此係由於回收區塊為暫時的。在FIT中監控且記錄存在於裝置中之每一檔案的狀態以及檔案資料索引資訊。只要發生任何狀態轉變，則更新針對檔案記錄之狀態。

取決於檔案狀態轉變與程式化資料、與變過時之資料還是與選定之回收區塊相關聯，將檔案狀態轉變再劃分為三類。在圖18之狀態圖中說明由於掛起或完成之資料程式化操作的檔案狀態中之准許轉變。由具有識別來自圖17中之表之數字的檔案狀態之圓來指示七個檔案狀態。

在圖18之狀態轉變上的標籤具有以下含義：

A-將已擦除區塊作為檔案之作用區塊而分配；

B-已填充部分區塊；

C-將部分區塊作為檔案之作用區塊而分配；

D-將此檔案之部分區塊作為另一檔案之作用區塊而分配；

E-對作為作用區塊而分配之已擦除區塊進行資料轉變；及

F-對作為作用區塊而分配之部分區塊進行資料轉變。

多數狀態轉變在分配一區塊或一區塊變為滿時自動發生。然而，一些界定狀態轉變亦併入有特定資料自一個區塊至另一區塊之再定位。資料作為單個不間斷操作而再定位，且狀態轉變視為僅在完成資料再定位之後發生。此等

轉變被指定為"資料轉變"。參看圖18之狀態圖，圖19之表提供准許狀態轉變之細節。

當待寫入之資料的長度已知時，可將部分區塊作為作用區塊而分配。在此情況下，自裝置中之全體部分區塊選擇"最適合"部分區塊。"最適合"定義為具有待寫入之已知量資料可有效利用之適量已擦除容量的部分區塊。在一些情況下，若不存在"最適合"部分區塊，則"最大"部分區塊可作為替代而選擇。此為具有最高量之可用未用容量的部分區塊。

圖20為說明由於資料變為過時之檔案狀態轉變的狀態圖。特定檔案之此等狀態轉變在儲存於含有特定檔案之資料之碎形區塊中的檔案之所有資料變為過時時發生。資料已變為過時之檔案無需為特定檔案。可由四個事件中之任一者使資料過時：

1.由主機刪除一檔案；

2.由主機刪除在檔案內之資料；

3.由主機更新先前對檔案寫入之資料；或

4.在回收操作期間重定位檔案之資料。

在圖20之狀態轉變上的標籤具有以下含義：

G-在部分區塊中之此檔案的所有資料已變為過時；

H-在滿的共同區塊中之此檔案或所有其他檔案之所有資料已變為過時；及

I-在部分區塊中之所有其他檔案之所有資料已變為過時。

圖21之表提供由於在圖20中說明之過時資料之檔案狀態轉變的細節。在此等環境中之任一者下使資料為過時導致定位有過時資料之區塊的類型改變，從而導致檔案之狀態的所得改變。

當將一區塊選擇為回收區塊時，不再將該區塊視為檔案之碎形區塊(檔案之資料存在於該區塊中)。此導致由圖22之狀態圖說明之檔案狀態轉變。在圖22之狀態轉變上的標籤具有以下含義：

J-將部分區塊選擇為回收區塊；及

K-將滿的共同區塊選擇為回收區塊。

在圖23之表中給出由於回收區塊之選擇的檔案狀態轉變的細節。

存在兩個替代機制以用於將檔案之資料與連續邏輯位址空間之邏輯區塊對準。在實體記憶體單元區塊上操作之直接資料檔案系統的情況下，如在上文交叉引用之專利申請案中所描述，新檔案之開始較佳對準已擦除記憶體單元區塊之開始。此亦可在直接資料檔案系統以邏輯區塊操作時進行，如圖24中所說明。說明儲存於邏輯區塊1至7中之三個檔案A、B及C。當寫入此等檔案中之一者的所有資料時，自圖24應注意，檔案之最後部分佔據部分區塊之一部分。

圖25之表提供用於判定作為作用區塊而分配以儲存檔案之資料之邏輯區塊類型的標準。如所指示，此取決於檔案之現有狀態(由圖17之表所定義)及待程式化之資料的主導

狀況。即使當在此基礎上選擇分配情況中之一者時，必須取決於可用性而自一可能性之有限集合進一步選擇區塊之類型，如圖25之右手行所指示。舉例而言，對分配情況B而言，部分區塊優先用於接收已知長度之資料。首先搜尋具有恰足夠可用(已擦除)容量以儲存此已知量資料之部分區塊。但若不存在該部分區塊，則判定是否存在具有最大未經程式化空間之部分區塊。否則，作為第三優先級，接著指定完全未分配(已擦除)區塊以接收資料，此將導致一部分區塊，因為在此實例分配情況B中待寫入之資料的已知量小於將填滿一滿區塊之量。

可刪除與在首次寫入一檔案(諸如，在圖24中之檔案A、B或C中之一者)時之狀態保持相同之該檔案，而隨之無需重定位任何不相關檔案之資料。但若回收操作已合併檔案之部分區塊之資料與另一檔案之資料，則可按需要來刪除檔案以自僅一個區塊重定位另一檔案之資料。舉例而言，若來自區塊2之檔案的資料A已與在區塊7中之檔案C的資料合併，則接著，可隨後按需要來刪除檔案A或檔案C以自僅一個區塊7重定位資料。

區塊回收為與寫入檔案資料之處理交錯的處理，其中自正經歷回收之區塊重定位有效資料，以便允許待擦除之區塊(其所有容量指定為未分配)回收在該區塊中之未用容量。可出於兩種原因中之一者選擇回收之區塊：

1.區塊含有由於已刪除或更新檔案之過時資料；或

2.區塊為部分區塊，且含有未經程式化容量。

分配給回收處理之時間比例較佳不變，使得可維持寫入新檔案資料之速度不變。此難以達成，因為檔案寫入處理產生必須由回收處理來處理之不可預測數目的部分區塊。

在圖24中展示之檔案至區塊之映射機制的優點在於，其允許含有最新近寫入之檔案資料的部分區塊保持儘可能長，直至該等部分區塊被選擇為用於回收操作之源區塊或目的區塊為止。此增大關於檔案能夠在重定位其資料中之任何資料或在共同區塊中之不相關檔案的資料之前被刪除的概率。此係因為該檔案之資料包含在專用於該檔案之區塊中。因為無需回收操作、無需複製資料之時間，且因此記憶體系統更有效地操作。

圖24之映射機制的一缺點在於，通常對每一寫入之檔案產生一個部分區塊，且一般需要合併許多部分區塊之資料以便回收其可用未經程式化(已擦除)容量。另外，若記憶體控制器以自動合併來自部分寫入區塊之資料以回收新的已擦除區塊容量之方式管理實體記憶體，則可由圖24之映射機制產生大量消耗時間之資料複製。因此，可能會需要實施圖26之映射機制的替代機制。此機制之主要特徵在於新檔案開始處之資料鄰接先前佔據部分區塊之不相關檔案的資料。當已寫入檔案之所有資料時，最後寫入資料最通常佔據部分區塊之一部分，但僅為暫時地。在該部分區塊中之未經程式化空間立即填充有經寫入以鄰接完成檔案之新檔案的資料。

在圖27之表中給出圖26之機制之針對為寫入資料分配作

用區塊的實施。在圖25之表中的分配情況A在圖27之表中由界定用於新檔案及現有檔案之獨立情況的分配情況A1及A2替代。

在圖26中，用於檔案之開始及檔案之末端之資料通常與不相關檔案共用一區塊，且在刪除一檔案時必須自兩個區塊重定位資料。舉例而言，若刪除檔案B，則自區塊2重定位檔案A之資料，且自區塊3重定位檔案C之資料。此接著使區塊2及區塊3能夠被擦除且添加至已擦除(未分配)區塊的集區，可將其他資料稍後寫入至該等區塊中。

圖26之檔案至區塊映射機制之一優點在於，其消除經部分程式化區塊之累積。因為在部分區塊中待寫入之新檔案的資料開始鄰接不相關現有檔案之資料，所以含有最新近寫入之檔案資料的部分區塊無法保持長時間週期，且因此在裝置中存在非常少之部分區塊。此限制關於必須在回收操作期間合併來自部分區塊之資料的機會，且允許建立不變回收速率以使用於新檔案資料之不變寫入速度得以維持。

然而，圖26之映射機制的一缺點在於，其增大關於在刪除檔案時需要重定位不相關檔案之資料的機率，且增大關於必須對每一機會重定位之資料量。在圖26之機制中之關於在刪除檔案時增大之資料重定位的此缺點否定了超越圖24之機制之關於降低在回收操作期間來自部分區塊之資料之合併的發生率的優點。

回收區塊容量

如上文所描述，區塊管理之部分包括回收在區塊中之未用容量以儲存新資料。此在儲存於記憶體系統中之資料量遠小於記憶體系統容量時並不特別重要，但較佳將記憶體系統設計為仿佛其充滿資料般地操作。其意謂，可以回收此未用容量之方式處理僅含有過時資料之區塊及含有有效資料但亦具有一些過時資料及/或未寫入頁之其他區塊。目標為儘可能完全地利用記憶體系統之儲存容量，而同時最小化對系統之效能的不利影響。

將在指定用於回收操作之區塊(源區塊)中之有效資料複製至具有用以儲存該有效資料之充分未分配(已擦除)容量的一或多個區塊(目的區塊)中。根據上文描述之區塊管理技術來選擇目的區塊。將儲存於源區塊中之每一檔案的資料複製至基於檔案之狀態及其他因素(如上文所描述)而選擇的區塊類型。在圖28A至圖28D中給出作為回收操作之部分的在不同類型檔案之間複製資料的實例。

在圖28A中，作為實例說明對兩個部分區塊681及683之回收操作。區塊681為程式化區塊，其中儲存檔案A之有效資料，同時亦含有並未儲存資料之已擦除容量。取決於檔案A之狀態的一個可能回收操作為，將區塊681之檔案A的資料複製至已包括不同檔案B之資料的另一部分區塊685的可用已擦除容量中，因此使該部分區塊685成為共同區塊。接著在FIT中不再引用在區塊681中之資料群組，且將該區塊記錄為過時。當儲存於區塊681中時，檔案A具有包括程式化區塊之狀態(參看圖17)中之一者。可接著將資料

移動至另一碎形區塊，而使檔案保持寫入至最多兩個碎形區塊中。在複製至區塊685之後，取決於儲存檔案之其他資料之區塊類型，檔案A已轉變為包括儲存於共同區塊中之檔案之資料的狀態(參看圖17)中之一者。

圖28A之區塊683為共同區塊，該區塊683係藉由將其檔案C及D之經儲存資料複製至含有檔案E之資料之程式化區塊687的已擦除容量中而回收，該程式化區塊687接著變為共同區塊。在區塊683中之檔案C及D之資料接著過時，如該區塊自身。檔案C及D中之每一者的狀態並未改變，因為資料已自一個共同區塊移動至另一共同區塊。然而，檔案E之狀態已改變。或者，檔案C及D中之每一者之資料可移動至彼此不同之區塊且無需必要地複製至共同區塊之可用空間。檔案之狀態可接著可能地轉變至其他狀態。

在圖28B中說明對實例區塊689及691之回收操作。此等區塊中之每一者為過時區塊，因為其充滿有效資料及過時資料。區塊689為含有檔案F之資料的檔案區塊，該檔案F之一部分為過時且剩餘部分為有效。舉例而言，此可在更新檔案F期間發生，其中將新資料寫入至在檔案末端處的具有與檔案之現有資料相同之邏輯偏移的位址，且現有資料接著變為過時。在此實例中，將檔案F之資料複製至含有檔案G之資料的程式化區塊693之已擦除容量中，從而導致區塊693之類型改變為共同區塊。或者，可將檔案F之有效資料寫入至已擦除區塊，此接著導致該區塊為程式化區塊。

圖28B之區塊691為含有檔案H之無效資料及檔案I之有效資料的滿的共同區塊。在此實例中，將檔案I之有效資料自區塊691複製至已擦除區塊695中。區塊695接著變為程式化區塊。或者，若可得到良好適合性，則可將檔案I之資料寫入至含有另一檔案之資料的部分區塊。目的區塊將取決於檔案I在回收操作時之狀態。

由於在圖28A及圖28B中展示之回收操作之四個特定實例中之每一者，儲存於兩個部分區塊中之資料組合為一個，藉此使該兩個區塊中之另一者僅具有過時資料。該等區塊接著為無效區塊。原始區塊681、683、689及691中每一者之整個空間接著藉由擦除區塊而回收，如圖28C中所說明。已擦除區塊為回收無效區塊之結果。

圖28D展示儲存檔案J之資料之檔案區塊697的實例。當檔案J由主機刪除時，使在區塊697中且可能亦在其他區塊中之檔案J的資料為過時。區塊697接著變為無效。回收無效區塊為系統已擦除區塊集區提供已擦除區塊。

自記憶體刪除檔案亦通常導致在一或多個碎形區塊(諸如，共同區塊或滿的共同區塊)中之檔案的資料變為過時。因為另一檔案之剩餘有效資料小於區塊之儲存容量且可為少量，所以該區塊接著經受回收操作。

由圖29之流程圖概括展示回收操作。取決於特定實施例，為部分、過時及無效區塊維持一或多個清單，如由步驟701所指示。根據一技術，在啟動記憶體系統時(諸如，當首先施加電源時)建立區塊之此清單。此清單可包括使

每次選擇一個回收區塊之區塊的其他資訊，諸如，在每一區塊中之有效資料量及在每一區塊中之已擦除空間量。此等量通常根據區塊之頁的數目或在使用元區塊時之元頁的數目而量測。一較佳替代技術為在非揮發性記憶體中維持此等表，且只要一區塊狀態改變便在清單中添加或更新該區塊之項。利用此技術，當初始化記憶體系統時，無需掃描區塊及建立清單。作為將所有部分、過時及無效區塊保持於清單上之替代，僅包括彼等具有低於一些設定臨限量的少量有效資料之區塊，因為選擇回收區塊之一特徵在於，該區塊具有極少或不具有需要複製之有效資料。在許多回收操作中必須將資料自一個區塊複製至另一區塊，此花費顯著量之時間，因此一般對彼等具有較少待複製資料量之區塊執行此操作。

此等區塊之清單隨著寫入、更新、移動、刪除等而不斷改變。導致區塊類型改變為部分、過時且無效且導致區塊類型自部分、過時及無效改變的改變導致由圖29之步驟701所維持之清單有所改變。有效資料量之改變個別地儲存於此等區塊中，且已擦除容量之量的改變亦記錄於區塊清單中。

在步驟703中，較佳自彼等在經更新清單上之區塊識別單個回收區塊作為下一個以便進行回收。在部分或過時區塊之情況下，其為待複製至稱為目的區塊之另一區塊中之有效資料的源。下文描述可用於選擇源區塊之若干特定技術。

考慮到需要回應主機之命令而執行之記憶體操作，圖29之下一步驟705接著判定在當前時間執行回收操作是否合適。若主機已發布閒置命令或指示將存在主機不會期望記憶體系統執行特定操作之某個時間週期的類似命令，接著系統可在前景中自由地進行負擔操作，其包括回收操作。即使主機忙於將資料寫入至記憶體系統或自記憶體系統讀取資料，回收操作(特定言之，其資料複製)仍可與資料寫入及讀取操作交錯進行。在2005年10月25日申請之Alan Sinclair之美國專利申請案序號第11/259,423號及2005年12月19日申請之Alan Bennett等人之序號第11/312,985號中描述應用至實體記憶體單元區塊之此交錯。

若由圖29之步驟705判定可進行回收操作，則該處理取決於經識別回收區塊是否含有有效資料，且若含有則是否含有一個以上檔案之有效資料而不同。在部分區塊或過時區塊之情況下，按定義該區塊含有有效資料，且在共同區塊或滿的共同區塊之情況下，該區塊含有兩個或兩個以上檔案之有效資料。由步驟707判定在回收區塊中是否存在有效資料。若存在必須移動之有效資料，則在下一步驟709中識別單個檔案之資料且識別目的區塊以接收該資料。目的區塊藉由上文參看圖17至圖19而描述之處理來識別，以便維持檔案(有效資料屬於該檔案)之所有資料儲存於兩個或兩個以下碎形區塊中(在此實例中)。接著開始將一個檔案之有效資料自源回收區塊複製至目的區塊，如步驟711所指示。在複製此等資料之後，處理返回至步驟707

以判定另一檔案之資料是否剩餘。若剩餘，則對額外資料重複步驟709及711之處理。目的區塊係獨立於對不同檔案之資料的較早選擇而選擇。此繼續，直至在步驟707中判定在源區塊中不再存在待移動之資料為止，在此情況下，可按步驟713擦除源區塊。接著可將此區塊放於已擦除區塊集區中以用於儲存新資料。

返回至圖29之步驟707，若源區塊不含有有效資料(為無效區塊之情況)，則不存在待移動之有效資料。僅需要擦除源區塊。因此，在該情況下之處理繞過如圖29中展示之步驟709及711。

在圖29之處理的第一實施例中，由步驟701維持部分、過時及無效區塊之單個清單。在清單上之個別項中包括在區塊中之有效資料量。在步驟703中，自清單選擇為回收區塊之區塊為具有最少有效資料之區塊。若在清單上存在一個無效區塊，則首先選擇該區塊，因為該區塊不具有有效資料。若在清單上存在許多無效區塊，則選擇在此處時間最長之區塊。若在清單上不存在無效區塊，則接著將具有最少量有效資料之區塊選擇為回收區塊。藉由在清單上選擇具有所有區塊之最少量有效資料的區塊，回收操作接著花費比當存在待自一個區塊複製至另一區塊之更多有效資料時更短之時間。結果，記憶體系統之其他操作(諸如，將資料寫入至記憶體及自記憶體讀取資料之速度)得以維持於高速率。以較少之記憶體效能成本獲得新的已擦除區塊。

此用於基於在碎形區塊中之有效資料量而在單個清單上選擇源區塊之圖29之處理的第一實施例具有實施相對簡單之優點。然而，此處理可藉由亦考慮部分區塊之值而改進。部分區塊具有可寫入資料之已擦除容量，而過時區塊及無效區塊均不含有任何已擦除容量。在可將過時區塊用於儲存新資料之前，將任何有效資料自該等過時區塊移出且移動至另一區塊中，使得該等過時區塊可接著經擦除且可用於儲存新資料。但部分區塊具有可寫入資料之已擦除容量，而並非必須承受回收操作之負擔。舉例而言，此可能會無益於回收一部分區塊，僅因為在部分區塊亦含有可寫入資料之大量已擦除容量時含有最少量之有效資料。

因此，在圖29之處理之其他實施例中，基於存在於部分區塊中之有效資料量及已擦除容量之量兩者而將部分區塊選擇為用於回收源區塊之候選者。在圖30中展示在部分區塊中之資料的成份。區塊(可為元區塊)具有含有有效資料之特定數目之一或多個頁(可為元頁)及經擦除且可寫入資料之一或多個其它頁。部分區塊亦可含有一或多個其它頁，其含有過時資料，如圖30之實例中所示。

在圖29之處理之此等其他實施例中，較佳由步驟701將部分區塊維持於與過時及無效區塊之清單分離之清單中。當部分區塊具有較少已擦除容量(其意謂該等部分區塊在其當前狀態下並不非常有用)及需要移動之少量有效資料時，使該等部分區塊朝其清單之頭部移動以用於回收操作。此等區塊將主要含有過時資料。相反地，具有大量已

擦除容量(其意謂該等部分區塊潛在地有用於儲存資料)及待移動之大量有效資料之部分區塊最不可能被識別為用於回收區塊之候選者。回收具有已擦除容量之部分區塊並不如同回收一過時區塊般將儲存容量之一些量添加至邏輯位址空間。無效區塊明顯為針對回收之最具吸引力之區塊，因為此等無效區塊並不具有有益之已擦除容量且並無需要複製之有效資料。

在圖29之回收區塊識別步驟703之第二實施例中，由步驟701維持三個獨立清單，其中部分區塊、過時區塊及無效區塊中之每一者針對一個清單。若存在無效區塊，則自無效區塊之清單選擇回收區塊直至該清單上不再存在區塊為止。除了可能以先進先出(FIFO)次序以使在清單上時間最長之無效區塊得以首先選擇，不存在列出無效區塊之特定次序。接著，若不存在無效區塊，則自過時區塊清單選擇具有在該清單上之所有區塊之最少量有效資料的區塊。

若在無效清單及過時清單中之任一者上均不存在區塊，則接著在步驟703中將在部分區塊清單上之區塊選擇為回收區塊。儘管可將一部分區塊選擇為具有最少量有效資料之部分區塊，但較佳以認識其已擦除容量之益處之方式來對部分區塊分等級。為此目的，可對每一部分區塊計算"回收增益"，如下：

$$\text{回收增益}=(S-kE)/V \qquad (1)$$

其中，S為根據區塊之資料儲存頁之總數目的區塊大小，E為可寫入資料之已擦除容量之頁的數目，且V為含有需要

移動至另一區塊之有效資料之頁的數目。包括常數k以權衡區塊之已擦除容量之有利效應，但可將其設為1。隨著kE之值增大，所得回收增益降低。隨著V之值升高，回收增益亦降低。在步驟703中將具有回收增益之最高值之部分區塊選擇為回收區塊。其他數學表達式可替代地用於定義根據E及V之權衡對含有有效資料之系統操作的損失與具有已擦除容量之益處的回收增益。亦可在區塊中每次發生改變(諸如，每次將資料寫入至其已擦除容量中)時計算回收增益，且將該回收增益作為由檔案目錄或FIT維持之資訊的一部分而儲存。

在圖31中說明此第二實施例，其展示自獨立之部分區塊清單、過時區塊清單及無效區塊清單選擇(如由圖29之步驟701所維持)選擇回收區塊(圖29之步驟703)的方法。步驟721首先判定是否存在於無效區塊清單上列出之區塊。若存在多個此等區塊，則由步驟723將在清單上最長時間之區塊選擇為回收區塊。若在無效區塊清單上不存在區塊，則接著由步驟725判定在過時區塊清單上是否存在項。若存在，則在一個以上區塊在過時區塊清單上之情況下，由步驟727將具有最少量有效資料之區塊選擇為回收區塊。若由步驟725判定在過時區塊清單上不存在項，則接著在步驟729中查閱部分區塊清單。當在部分區塊清單上存在一個以上區塊時，將具有最高回收增益者選擇為回收區塊。回收增益諸如藉由使用上文方程式(1)來考慮在區塊中之有效資料量及已擦除容量。若在部分區塊清單上不存在

內容，則藉由返回至步驟721來重複該處理，直至在清單中之一者上出現區塊為止。在選擇回收區塊之後，該處理進行至圖29之步驟705。

由圖32之流程圖展示第三實施例。執行圖29之步驟703亦開始於在由圖29之步驟701所維持之無效區塊清單上尋找一項的步驟741。若在無效區塊清單上存在一個以上項，則由圖32之步驟743將最舊者選擇為回收區塊。若在無效區塊清單上不存在項，則下一步驟745判定在過時區塊清單上是否存在項。若存在，隨後步驟與圖31之實施例之不同之處在於，若在部分區塊清單上亦存在至少一項，則將判定最佳係自過時區塊清單還是部分區塊清單選擇回收區塊。

圖32之步驟747識別在過時區塊清單上之含有最少量有效資料的區塊。接著由步驟749判定在部分區塊清單上是否存在至少一區塊，且若存在，則在步驟751中識別具有最少量有效資料之區塊。下一步驟753接著在自過時區塊清單識別之一個區塊與在部分區塊清單上識別之一個區塊之間進行選擇。為此目的，對在步驟751中自部分區塊清單識別之區塊計算量(V+kE)，項V、E及k與上文之使用相同。將此量與在於步驟747中自過時區塊清單識別之區塊中之有效資料的量V進行比較。若關於部分區塊之量(V+kE)大於過時區塊之V，則接著在步驟755中將該過時區塊選擇為回收區塊。但若過時區塊之V大於經識別部分區塊之量(V+kE)，則接著在步驟757中將該部分區塊選擇為

回收區塊。

藉由在與經識別過時區塊之僅有效資料V比較之前將經識別部分區塊之已擦除容量之量kE添加至其有效資料V，該處理傾向於贊同選擇過時區塊。保持具有與經識別過時區塊相同之有效資料量的經識別部分區塊，因為該部分區塊仍可能潛在地用於在其已擦除容量中儲存資料。實際上，將保持具有比過時區塊之有效資料量小一量kE之有效資料量的部分區塊。

返回至圖32之步驟745，若在過時區塊清單上不存在項，則接著在步驟759中判定是否存在於部分區塊清單上列出之區塊。若不存在，則該處理返回至步驟741以重複，直至在三個清單中之一者上放有一區塊為止。若存在列出之多個部分區塊，則接著在步驟761中，將具有最少量有效資料之區塊選擇為回收區塊。或者，可藉由使用如參考第二實施例(圖31)之步驟731而描述之回收增益來選擇部分區塊。

或者，第三實施例可利用僅兩個清單。第一清單為過時區塊清單，其含有用於含有過時資料且不含有已擦除容量之區塊的項。將無效區塊及過時區塊兩者放在單個"過時"區塊清單上，而並非使用如圖32中展示之獨立無效區塊清單。區塊可視情況含有有效資料。在清單中之每一項具有一含有用於定義與該項有關之在該區塊中之有效資料量的值的欄位。在清單中之項根據在此等欄位中之值而排序。因此，含有過時資料且不含有有效資料之區塊(無效區塊)

在此第一清單之頭部處集合在一起。

在此對第三實施例之替代中之第二清單為部分區塊清單，其含有用於含有一些已擦除儲存容量之區塊的項。區塊可視情況含有有效資料。在清單中之每一項具有一含有用於定義與該項有關之在該區塊中之有效資料量的值的欄位。在清單中之項根據在此等欄位中之值而排序。可由圖32之步驟753之技術自第一清單或第二清單之頭部(具有最少量無效資料之區塊)選擇一區塊。

圖33之表闡述放於部分區塊清單及過時區塊清單上以用於根據第三實施例之此修改之回收操作的區塊類型的細節。為放於部分區塊清單上，一區塊含有有效資料及已擦除容量兩者。其與在區塊中是否存在任何過時資料無關。為放於過時區塊清單上，區塊含有過時資料，以及有效資料或已擦除容量中之一者。

結論

雖然已參考本發明之例示性實施例描述了本發明之各種態樣，但應瞭解，本發明有權在隨附申請專利範圍之全部範疇內受到保護。

【圖式簡單說明】

圖1示意性地說明主機及所連接非揮發性記憶體系統；

圖2為用作圖1之非揮發性記憶體之實例快閃記憶體系統的方塊圖；

圖3為可在圖2之系統中使用之記憶體單元陣列的代表性電路圖；

圖4說明圖2之系統之實例實體記憶體組織；

圖5展示圖4之一部分實體記憶體之放大圖；

圖6展示圖4及圖5之一部分實體記憶體的另一放大圖；

圖7A、圖7B及圖7C展示且比較操作可再程式化記憶體系統之三種方法；

圖8A、圖8B及圖8C以不同格式展示且比較與分別在圖7A、圖7B及圖7C中所展示相同之操作可再程式化記憶體系統之三種方法以及與主機系統之介面；

圖9A、圖9B及圖9C以不同格式展示且比較與分別在圖8A、圖8B及圖8C中所展示相同之操作可再程式化記憶體系統之三種方法及與主機之介面；

圖10說明可用於進行圖9C之技術之邏輯至實體區塊映射的一實例；

圖11展示用以設定參數以進行在圖9C及圖10中所說明之技術的在主機與記憶體系統之間的交互；

圖12說明直接資料檔案系統之操作週期；

圖13A至圖13D展示寫入檔案資料之四個不同實例；

圖14A至圖14E展示寫入單個資料檔案之順序；

圖15展示回收圖14E之區塊的結果；

圖16A至圖16D展示儲存於區塊類型之多種允許組合中之資料檔案的實例；

圖17為給出根據一特定實例之檔案之准許狀態的表；

圖18為展示由於程式化資料之准許檔案狀態轉變的狀態圖；

圖19為描述在圖18中展示之檔案狀態轉變之表；

圖20為展示由於過時資料之准許檔案狀態轉變的狀態圖；

圖21為描述在圖20中展示之檔案狀態轉變之表；

圖22為展示由於回收區塊之准許檔案狀態轉變的狀態圖；

圖23為描述在圖22中展示之檔案狀態轉變的表；

圖24展示資料檔案與邏輯區塊之對準的一實施例；

圖25為展示在用於圖24之資料對準實施例之多種狀況下的作用區塊之分配的表；

圖26展示資料檔案與邏輯區塊之對準的一替代實施例；

圖27為展示在用於圖26之資料對準實施例之多種狀況下的作用區塊之分配的表；

圖28A至圖28D展示區塊回收操作之實例；

圖29為概括說明回收操作之流程圖；

圖30說明儲存於典型部分記憶體單元區塊中之資料的類型；

圖31提供進行圖29之流程圖之步驟中之一者的特定實施例的細節；

圖32提供執行圖29之流程圖之相同步驟的替代實施例的細節；及

圖33為定義放在又一實施例之兩個區塊清單上之區塊類型的表。

【主要元件符號說明】

1	主機系統
2	快閃記憶體
3	配合零件
4	配合零件
5	應用程式部分
6	驅動程式部分
7	快閃記憶體
8	電路
11	控制器
13	系統匯流排
15	記憶體晶片
17	導體
19	導體
21	導體
23	內部匯流排
25	介面電路
27	處理器
29	唯讀記憶體(ROM)
31	隨機存取記憶體(RAM)
33	電路
35	電路
37	外部接觸
39	時脈
41	平面

43	平面
45	行控制電路
47	行控制電路
49	位元線
51	位元線
53	字線
55	列控制電路
57	源電壓控制電路
59	源電壓控制電路
61	p型井電壓控制電路
63	p型井電壓控制電路
65	輸入/輸出電路
67	輸入/輸出電路
71	線
73	介面電路
75	狀態機
77	控制線
78	控制線
79	控制線
80	控制線
81	控制線
83	線
91	線
92	線

93	線
94	線
97	記憶體單元串
98	記憶體單元串
99	記憶體單元串
100	記憶體單元串
101	記憶體單元串
102	記憶體單元串
103	記憶體單元串
104	記憶體單元串
107	電荷儲存記憶體單元
108	電荷儲存記憶體單元
109	電荷儲存記憶體單元
110	電荷儲存記憶體單元
111	選擇電晶體
112	選擇電晶體
115	字線
116	字線
117	字線
118	字線
119	閘極
120	閘極
123	區塊
125	區塊

131	平面/子陣列
132	平面/子陣列
133	平面/子陣列
134	平面/子陣列
137	區塊
138	區塊
139	區塊
140	區塊
141	元區塊
143	元區塊
145	區塊
146	區塊
147	區塊
148	區塊
151	元頁
153	區
155	區
157	資料部分
159	附加項部分/位元組
161	連續邏輯位址空間
163	區塊/邏輯至實體位址表
165	記憶體單元陣列
167	元區塊
169	元區塊

171	元區塊
173'	功能
181	實體區塊
183	區塊
184	部分/資料群組
185	區塊
187	資料
189	附加項資料
191	區塊
193	邏輯區塊
195	實體頁/資料檔案之一部分
197	實體頁/檔案之一更新部分
199	實體頁/先前寫入之資料的一部分
201、202、203、204	頁
207	記憶體裝置
209	非揮發性儲存空間
211	主機
213	內連匯流排
641	區塊
643	區塊
645	區塊
661	區塊
663	區塊

665	區塊
669	區塊
671	區塊
673	區塊
675	區塊
P0、P1…P7	頁

五、中文發明摘要：

資料檔案被指派位址，該等位址在具有實體記憶體單元區塊之一平常類型之快閃記憶體系統之一連續邏輯位址空間介面(LBA介面)的一或多個邏輯區塊內。此指派可由通常但並非必要產生該等資料檔案之主機裝置進行。以一降低在該等實體記憶體區塊內之檔案資料的分段量之方式控制含有任一檔案之資料之邏輯區塊的數目，藉此維持良好記憶體效能。該主機可回應於瞭解其所連接之一記憶體之實體特徵而組態該位址空間之該等邏輯區塊。

六、英文發明摘要：

Data files are assigned addresses within one or more logical blocks of a continuous logical address space interface (LBA interface) of a usual type of flash memory system with physical memory cell blocks. This assignment may be done by the host device which typically, but not necessarily, generates the data files. The number of logical blocks containing data of any one file is controlled in a manner that reduces the amount of fragmentation of file data within the physical memory blocks, thereby to maintain good memory performance. The host may configure the logical blocks of the address space in response to learning the physical characteristics of a memory to which it is connected.

十、申請專利範圍：

1. 一種識別在一系統內之檔案物件之資料的方法，其包含：

維持一劃分為邏輯區塊之系統邏輯位址空間，

對在該等邏輯區塊中之一或多者內之該等檔案物件中之個別者指派唯一位址，其中該等邏輯區塊可個別地含有一個以上檔案物件之位址，及

限制對一個別檔案物件指派的亦含有一第二檔案物件之資料之邏輯區塊的數目。

2. 如請求項1之方法，其另外包含：

在一主機系統中進行該方法，該主機系統另外提供該等檔案物件之該資料且在一適於與一記憶體系統連接之外部介面處利用該邏輯位址空間。

3. 如請求項1之方法，其另外包含：

在一記憶體系統中進行該方法，該記憶體系統另外自一外部源接收該等檔案物件之該資料且在一與非揮發性資料儲存媒體之介面處在該記憶體系統內利用該邏輯介面位址空間。

4. 如請求項1之方法，其另外包含：

在一處理單元中進行該方法，該處理單元適於與一供應該等資料檔案物件之主機可移除地連接，且適於與一在一與該處理單元之介面處利用該邏輯位址空間的記憶體系統可移除地連接。

5. 如請求項1至4中任一項之方法，其中限制邏輯區塊之該

數目包括將寫入一個別資料檔案物件的亦含有一第二檔案物件之資料的邏輯區塊之數目限於一預定最大數目。

6. 如請求項5之方法，其中該預定最大數目為2。

7. 一種關於一主機系統經由一介面轉移檔案物件之資料的方法，該介面適於與一具有可在再程式化之前一起擦除之記憶體單元之區塊的類型之一非揮發性記憶體系統連接，該方法包含：

在該介面處維持一劃分為邏輯區塊之邏輯位址空間，及

以該等邏輯區塊可個別地含有一個以上檔案物件但僅部分填充有一個別檔案物件之資料的邏輯區塊的一數目限於小於至少一預設極限的一方式指定在該等邏輯區塊中之一或多者內之該等檔案物件中之個別者內之資料的位址，以使一或多個其他檔案物件之資料被寫入或可被寫入至該等部分填充區塊中。

8. 如請求項7之方法，其另外包含：

將該邏輯位址空間之該等邏輯區塊的大小組態成個別地具有與一記憶體系統之記憶體單元之該等個別區塊相同的資料儲存容量，該主機系統適於與該記憶體系統連接。

9. 如請求項7之方法，其中對經由該介面發送該等檔案物件之資料的該主機指定該等邏輯區塊中之一或多者內之該等個別檔案物件的資料之該等位址。

10. 一種在一具有記憶體單元之區塊之非揮發性記憶體系統中儲存檔案物件之資料的方法，該等記憶體單元可在用

該資料再程式化之前一起擦除，其中：

將一邏輯位址空間劃分為邏輯區塊，該等邏輯區塊個別地具有對應於該等個別記憶體單元區塊之特徵的至少一特徵，

對個別檔案物件之資料指派在該等邏輯區塊中之一或多者內的位址，其中該等邏輯區塊可個別地含有一個以上檔案物件之資料的位址，但含有一特定個別檔案物件之資料加上另一檔案物件之資料的邏輯區塊數目為有限的，且

該等邏輯區塊之位址在該記憶體系統內映射至記憶體單元區塊之位址。

11. 如請求項10之方法，其中對該特定檔案物件之資料指派位址包括將僅部分填充有該特定檔案物件之資料之邏輯區塊的數目限於一預定數目。

12. 如請求項11之方法，其中該預定數目為2。

13. 如請求項10之方法，其中該至少一對應特徵包括該等個別邏輯區塊的一資料儲存容量與該等個別記憶體單元區塊之資料儲存容量相同。

14. 如請求項10之方法，其中該至少一對應特徵包括：

該等個別邏輯區塊之該資料儲存容量與該等個別記憶體單元區塊之該資料儲存容量相同，

將該等個別邏輯區塊劃分為用於在其中寫入資料之多個頁，該等頁具有與該等記憶體單元區塊之個別多個頁相同之資料儲存容量，且

將該等個別邏輯區塊之最低頁位址映射至該等個別記憶體單元區塊之最前頁中。

15. 一種關於一主機系統經由一介面轉移檔案物件之資料的方法，該介面適於與一具有可在再程式化之前一起擦除之記憶體單元之單位的類型之一非揮發性記憶體系統連接，該方法包含：

在該介面處維持一劃分為邏輯區塊之邏輯位址空間，

根據一在要指定之個別區塊內定址之檔案資料的結構指定一組複數個邏輯區塊類型，

根據一儲存要指定之個別檔案之資料的一或多個邏輯區塊之類型的一組合指定一組複數個准許檔案狀態，

維持一在該等邏輯區塊內定址之個別資料檔案之該等檔案狀態的記錄，及

對根據在該記錄中之該個別檔案之當前狀態選擇的一類型之邏輯區塊指定一個別檔案之資料的位址。

16. 如請求項15之方法，其中該等經指定複數個資料區塊類型包括第一複數個類型，其中僅一個檔案之資料儲存於一單個邏輯區塊中；及第二複數個類型，其中兩個或兩個以上檔案之資料儲存於一單個邏輯區塊中，且其中該等准許檔案狀態限制可被指定該等檔案中之一單個者的位址的該第二複數個類型之區塊的一最大數目。

17. 一種電腦系統，其包含：

一介面，其中利用在一劃分為不同區塊之連續邏輯位址空間內之位址識別資料檔案，及

一處理器，其與該介面操作地連接以導致個別檔案之資料由該等邏輯區塊中之一或多者的位址所識別，該等邏輯區塊經選擇以維持部分含有該等檔案中之任一者之資料的區塊的一數目低於一預設極限。

18. 如請求項17之系統，其中該處理器進一步操作以取決於部分填充有該檔案之資料之位址的邏輯區塊的數目而對該等個別檔案指派複數個經指定狀態中之一者。

19. 如請求項17之系統，其中該處理器進一步操作以對具有在該等邏輯區塊中定址之資料的每一檔案指派一狀態，且選擇一給定檔案之資料將根據該給定檔案之該狀態而定址的該等邏輯區塊中之一者。

20. 如請求項19之系統，其中該處理器進一步操作以至少部分基於部分填充有該檔案之資料之位址的邏輯區塊的一數目來對具有在該等邏輯區塊中定址之資料的每一檔案指派一狀態。

21. 如請求項20之系統，其中該處理器進一步操作以另外基於部分填充有該檔案之資料之位址的亦含有可被指定額外位址之未用容量的邏輯區塊的一數目來對具有在該等邏輯區塊中定址之資料的每一檔案指派一狀態。

22. 如請求項20之系統，其中該處理器進一步操作以另外基於部分填充有該檔案之資料之位址的亦含有一或多個其他檔案之資料之位址的區塊的一數目來對具有在該等邏輯區塊中定址之資料的每一檔案指派一狀態。

23. 如請求項17之系統，其中該介面包括一外部介面。

十一、圖式：

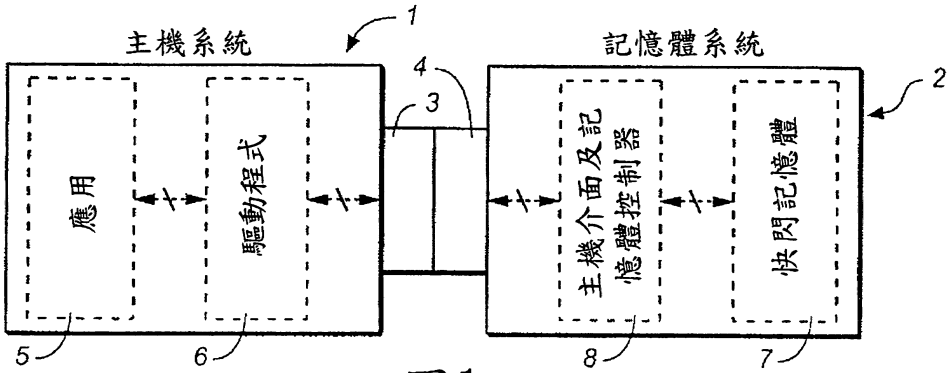

圖1

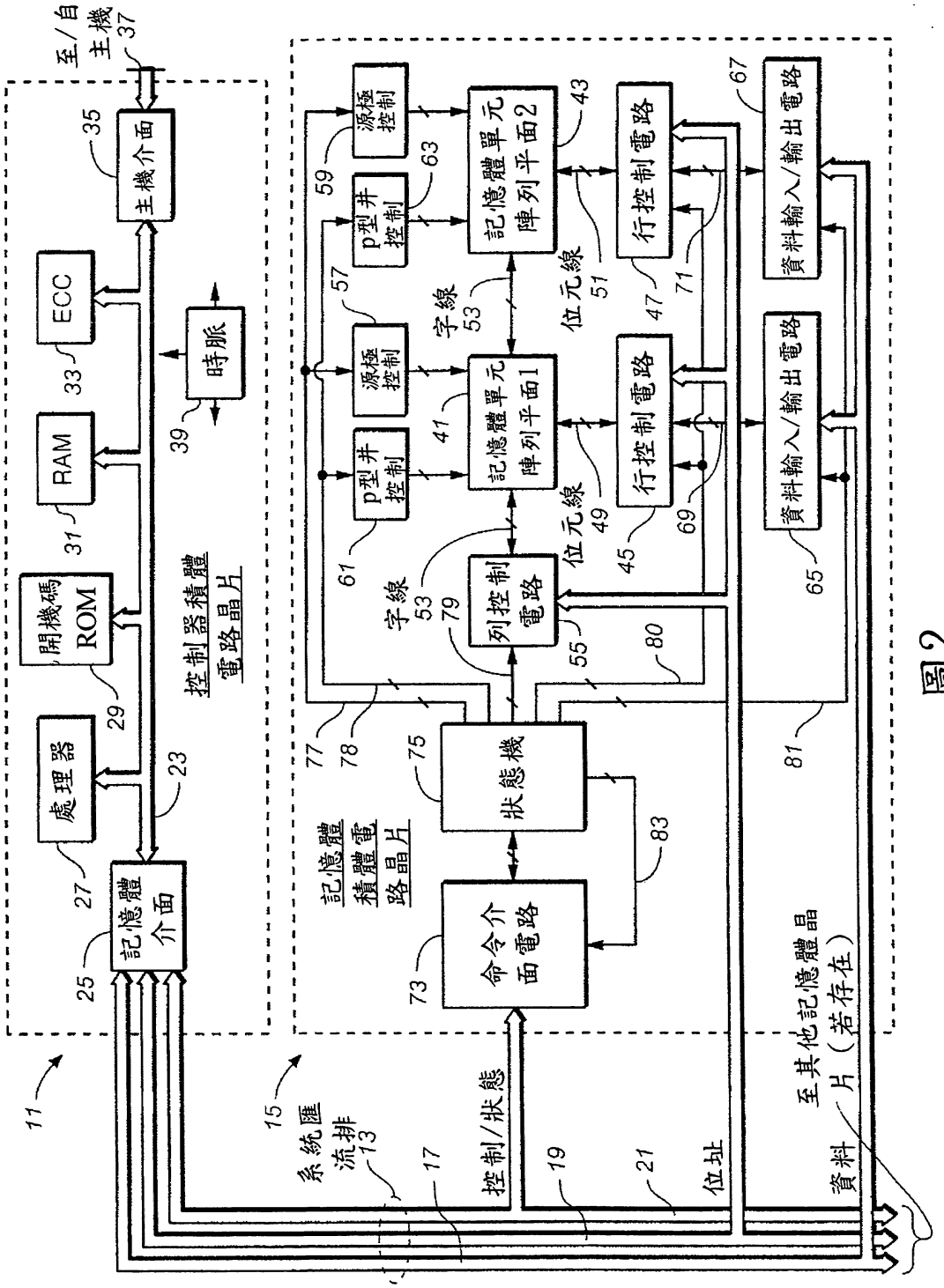
至/自主機 37
主機介面 35
ECC 33
RAM 31
開機碼 ROM 29
處理器 27
記憶體介面 25
時脈 39
控制器積體電路晶片
23
11
15
系統匯流排 13
控制/狀態 17
位址 19
資料 21
記憶體積體電路晶片
命令介面電路 73
狀態機 75
列控制電路 79
字線 53
p型井控制 61
源極控制 57
記憶體單元陣列平面1 41
位元線 49
行控制電路 45
資料輸入/輸出電路 65
記憶體單元陣列平面2 43
位元線 51
行控制電路 47
資料輸入/輸出電路 67
77
78
80
55
69
71
81
83
至其他記憶體晶片（若存在）

圖2

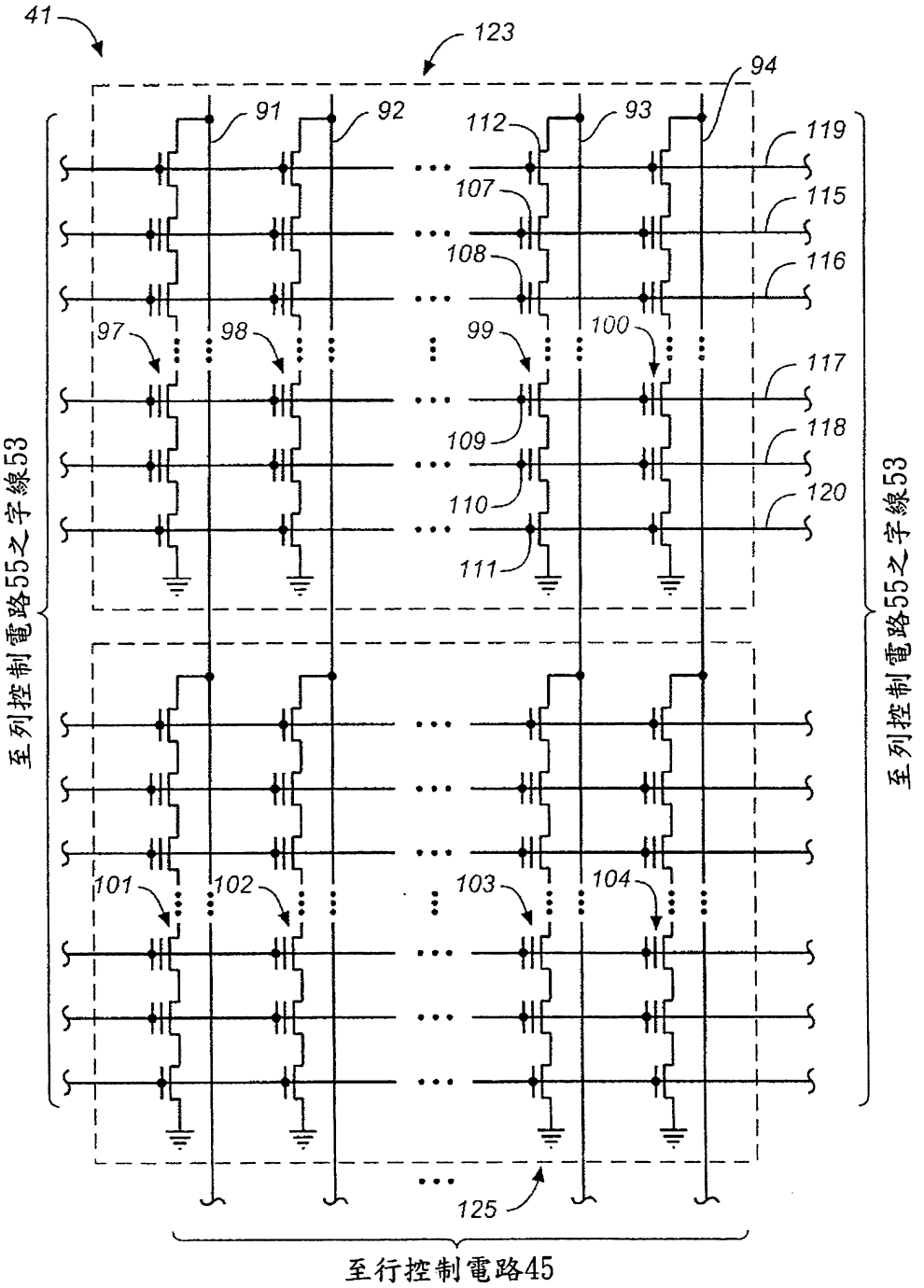
41
123
94
91
92
112
93
119
107
115
108
116
97
98
99
100
117
109
118
110
120
111
至列控制電路55之字線53
至列控制電路55之字線53
101
102
103
104
125
至行控制電路45

圖3

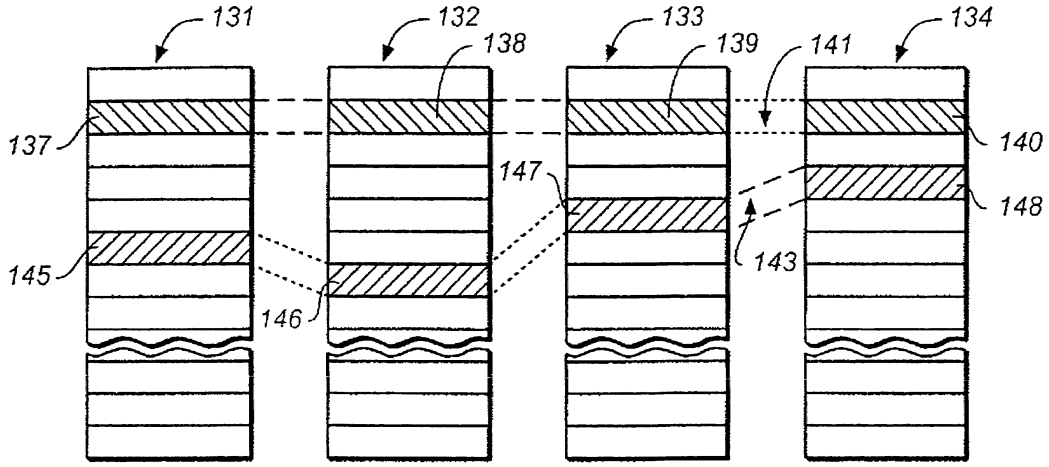
131
132
138
133
139
141
134
137
140
147
148
145
143
146

圖4

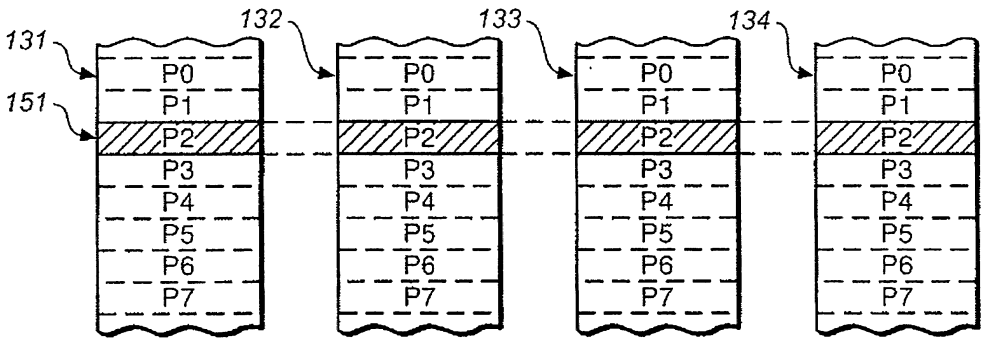
131
132
133
134
151
P0
P1
P2
P3
P4
P5
P6
P7

圖5

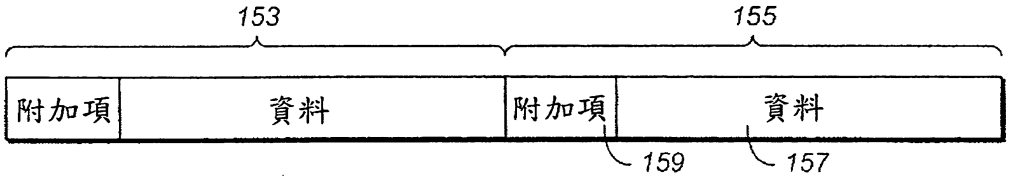
153
155
附加項
資料
附加項
資料
159
157

圖6

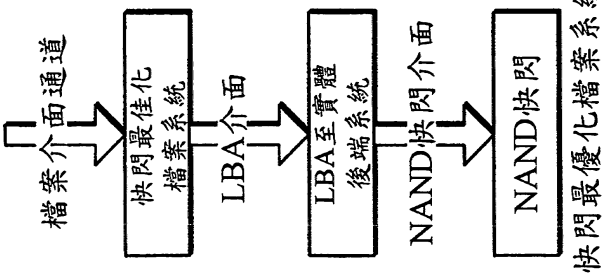
檔案介面通道
習知檔案系統
LBA介面
LBA至實體後端系統
NAND快閃介面
NAND快閃
習知檔案系統

圖7A

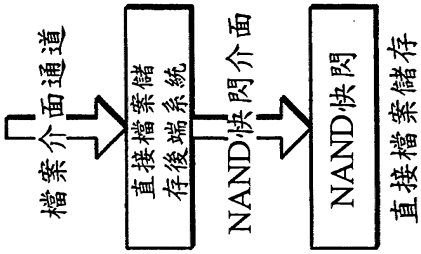
檔案介面通道
直接檔案儲存後端系統
NAND快閃介面
NAND快閃
直接檔案儲存

圖7B

檔案介面通道
快閃最佳化檔案系統
LBA介面
LBA至實體後端系統
NAND快閃介面
NAND快閃
快閃最優化檔案系統

圖7C

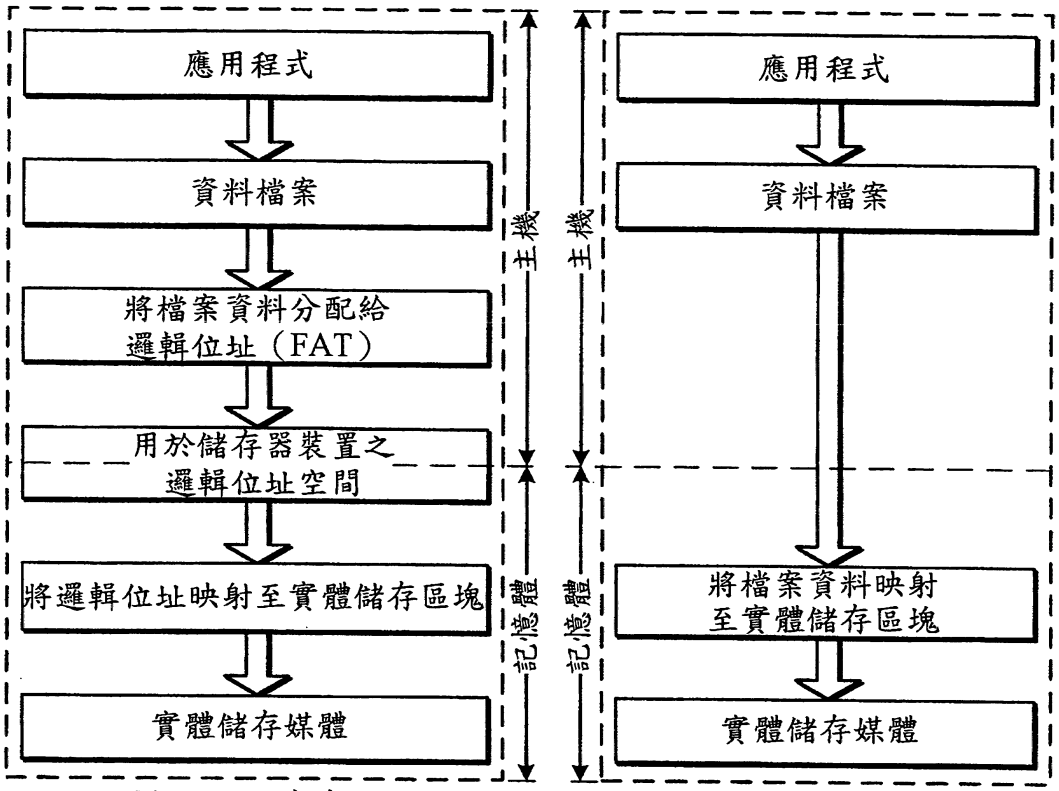
應用程式
資料檔案
將檔案資料分配給
邏輯位址（FAT）
用於儲存器裝置之
邏輯位址空間
將邏輯位址映射至實體儲存區塊
實體儲存媒體
主機
記憶體
邏輯位址儲存系統
應用程式
資料檔案
將檔案資料映射
至實體儲存區塊
實體儲存媒體
主機
記憶體
直接資料檔案儲存系統

圖8A

圖8B

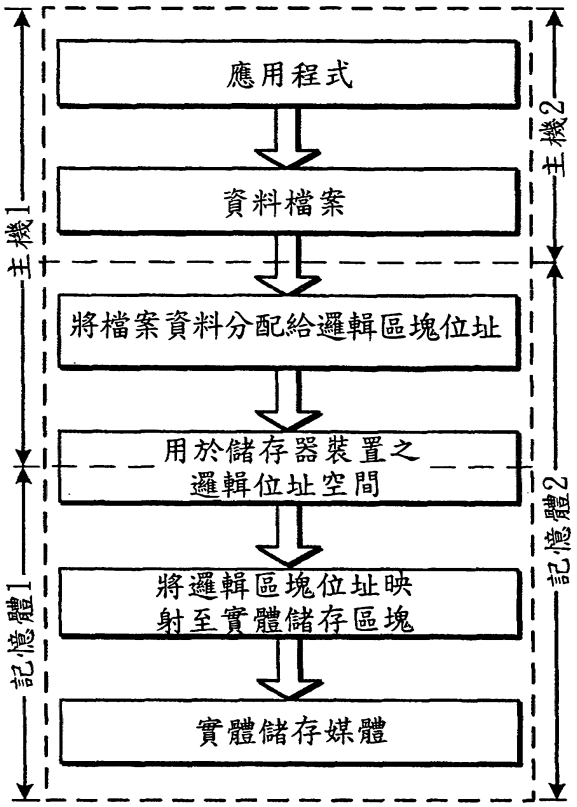
應用程式
資料檔案
將檔案資料分配給邏輯區塊位址
用於儲存器裝置之
邏輯位址空間
將邏輯區塊位址映
射至實體儲存區塊
實體儲存媒體
主機1
主機2
記憶體1
記憶體2

圖8C

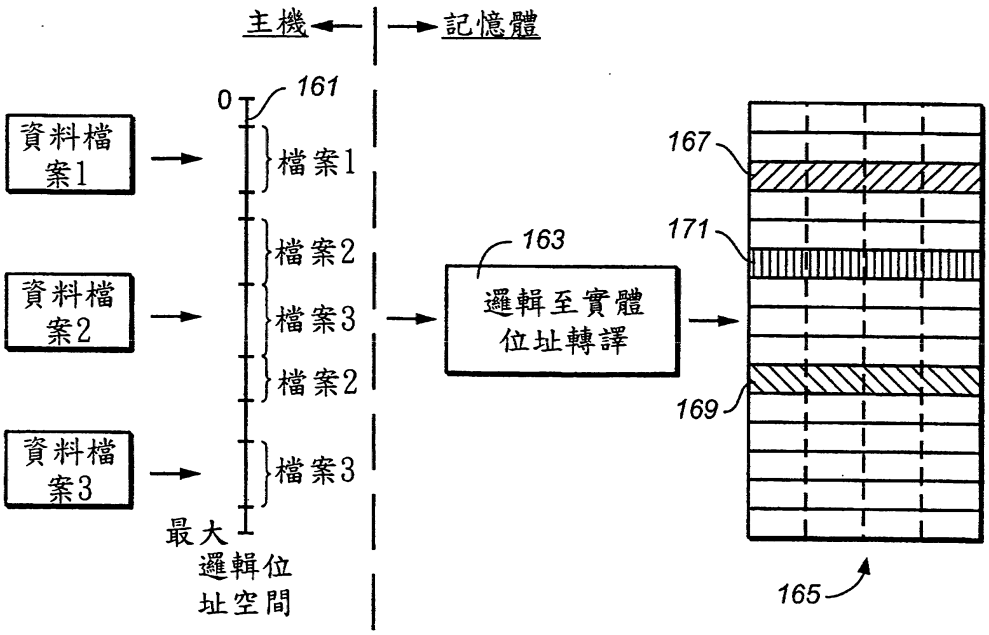
主機
記憶體
0
161
資料檔案1
檔案1
檔案2
資料檔案2
檔案3
檔案2
資料檔案3
檔案3
最大邏輯位址空間
163
邏輯至實體位址轉譯
167
171
169
165

圖9A

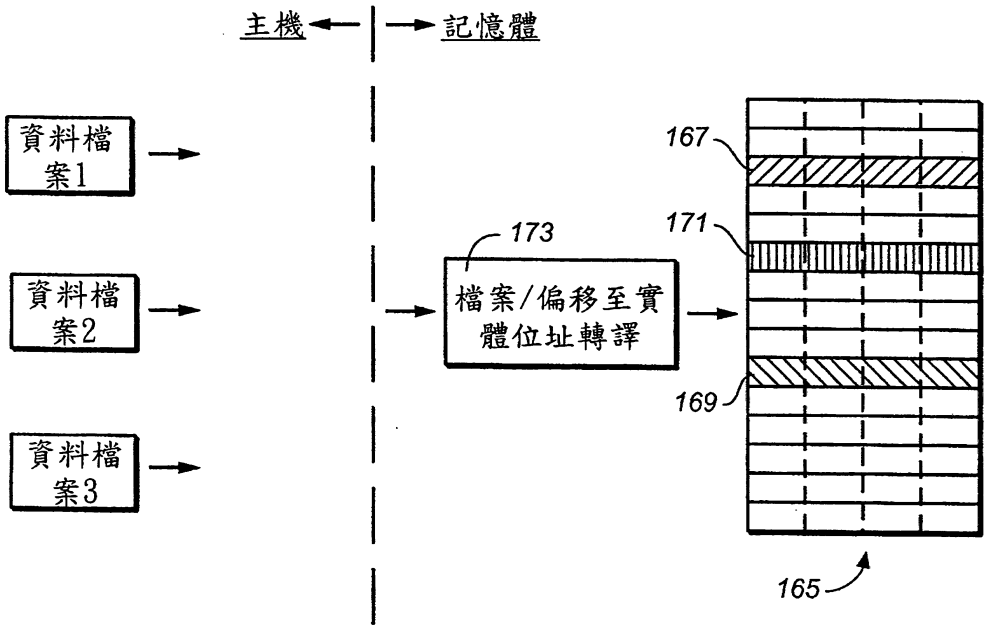
主機
記憶體
資料檔案1
資料檔案2
資料檔案3
173
檔案/偏移至實體位址轉譯
167
171
169
165

圖9B

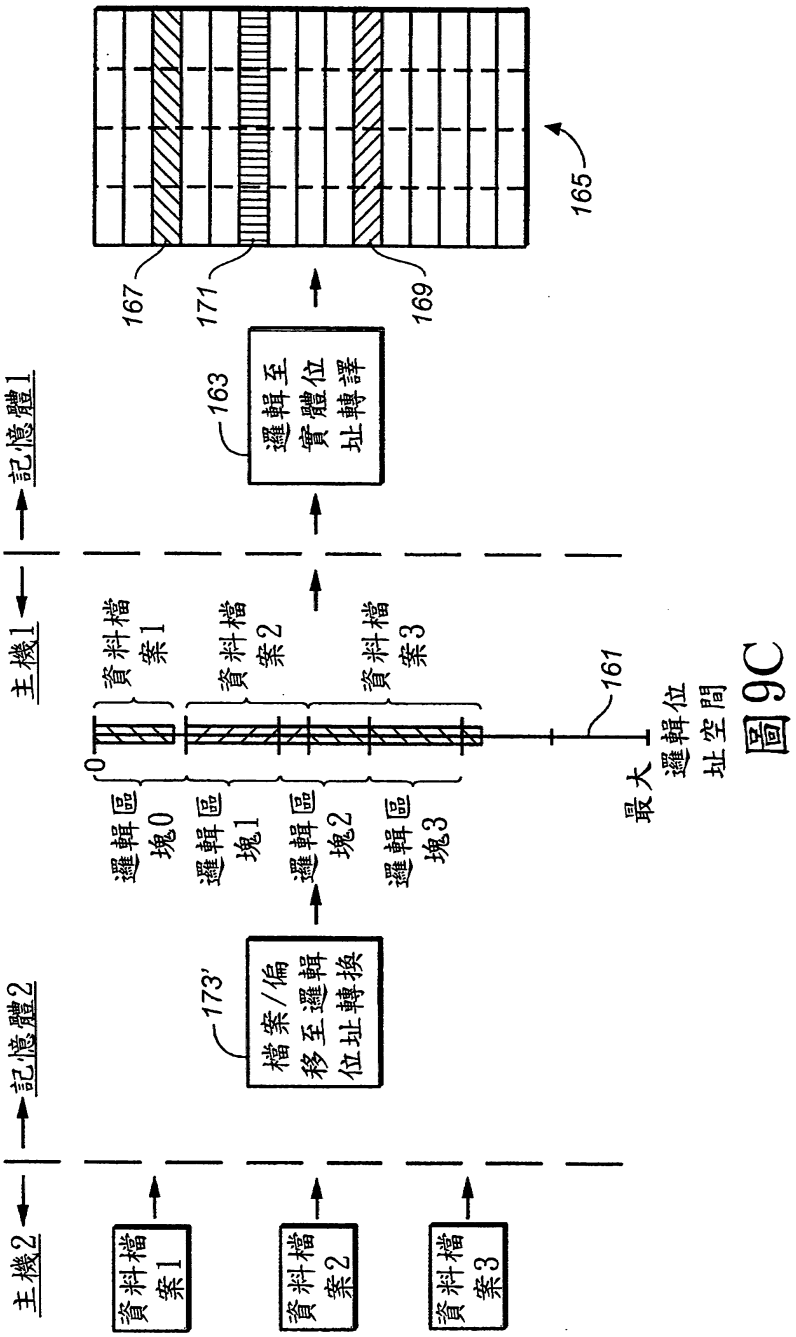
主機2
記憶體2
主機1
記憶體1
資料檔案1
資料檔案2
資料檔案3
檔案/偏移至邏輯位址轉換
173'
邏輯至實體位址轉譯
163
邏輯區塊0
邏輯區塊1
邏輯區塊2
邏輯區塊3
0
最大邏輯位址空間
161
167
171
169
165

圖 9C

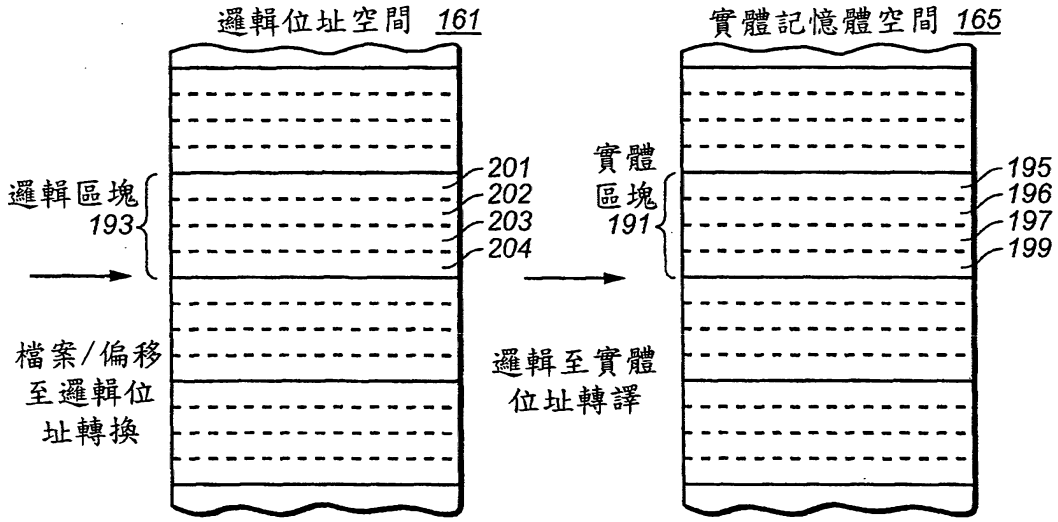
邏輯位址空間 161
實體記憶體空間 165
邏輯區塊
193
201
202
203
204
實體
區塊
191
195
196
197
199
檔案/偏移
至邏輯位
址轉換
邏輯至實體
位址轉譯

圖10

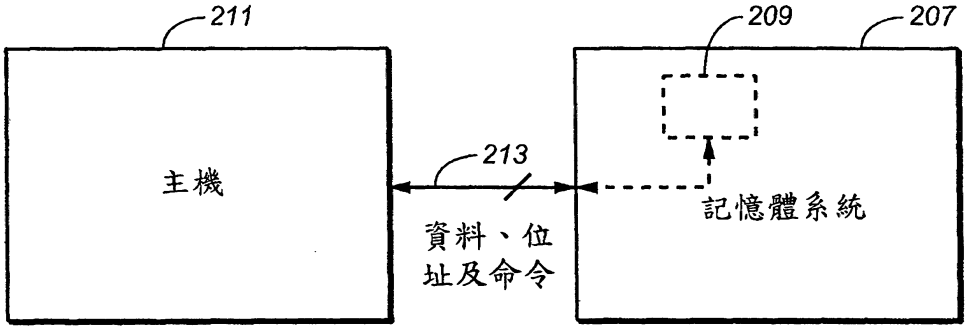
211
209
207
主機
213
資料、位
址及命令
記憶體系統

圖11

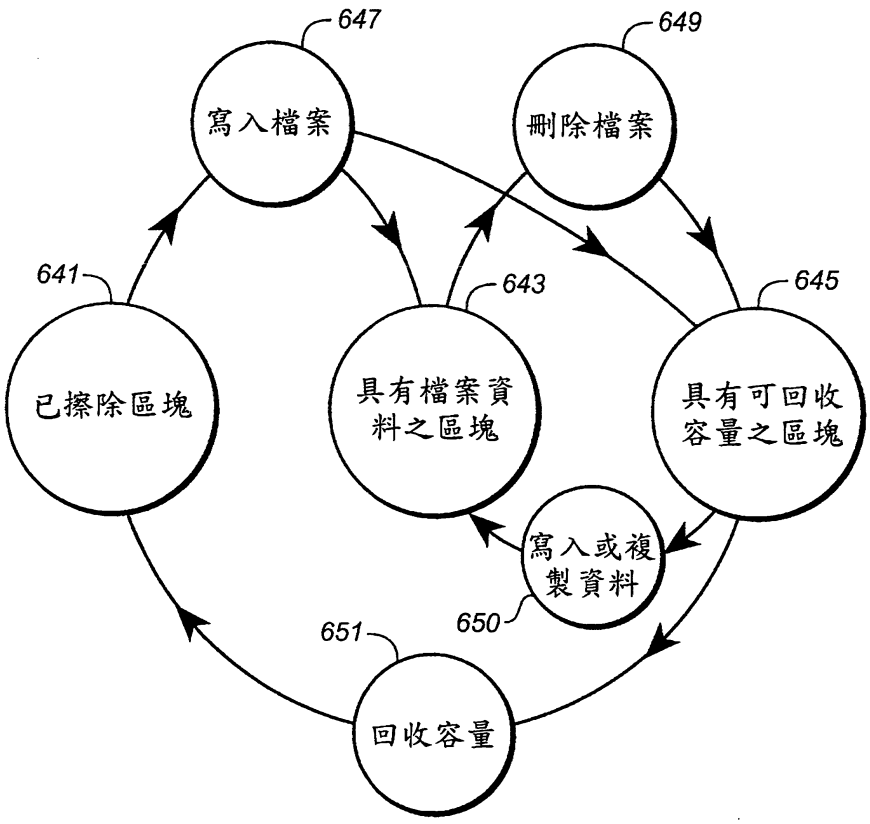
647
寫入檔案
649
刪除檔案
641
已擦除區塊
643
具有檔案資料之區塊
645
具有可回收容量之區塊
寫入或複製資料
650
651
回收容量

圖12

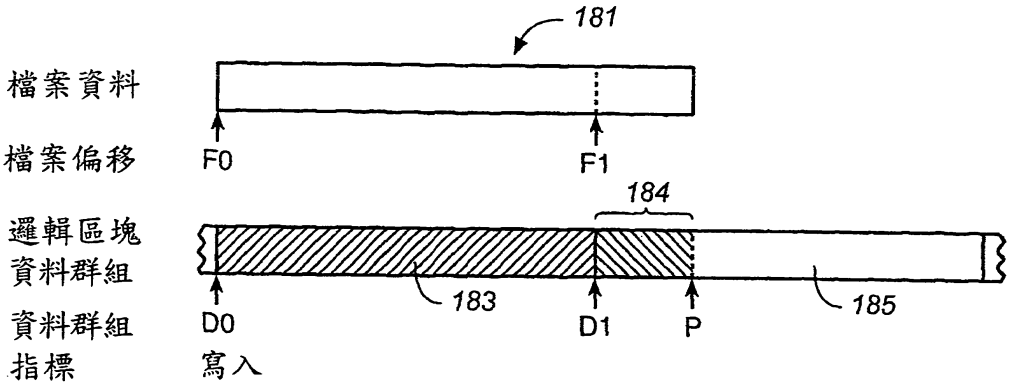

圖13A

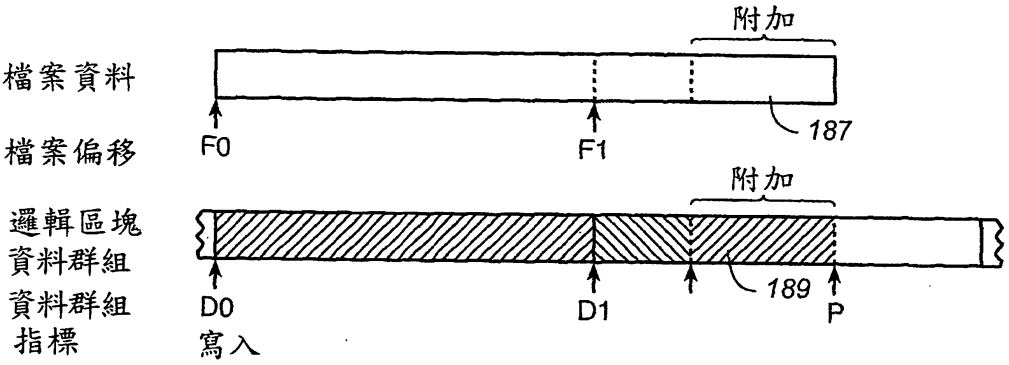

圖13B

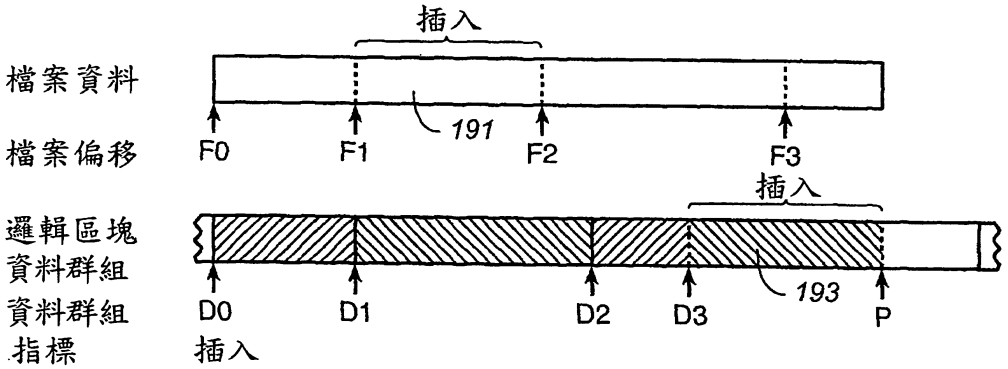

圖13C

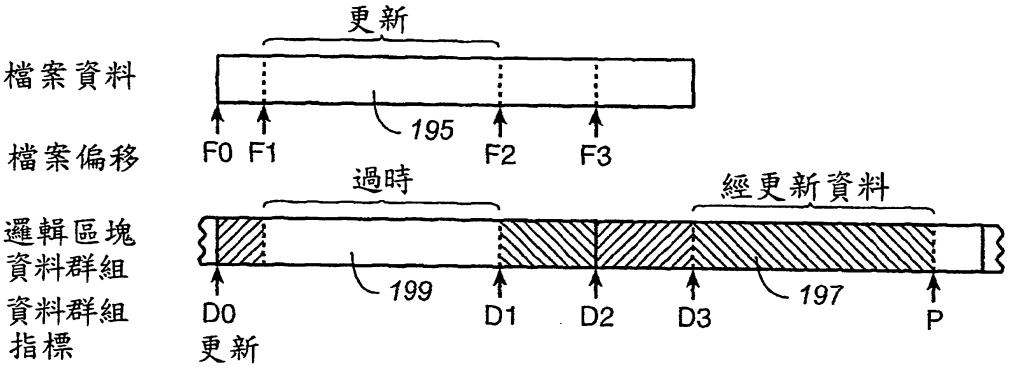

圖13D

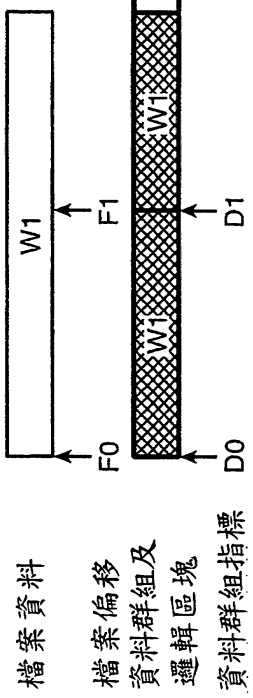

圖14A

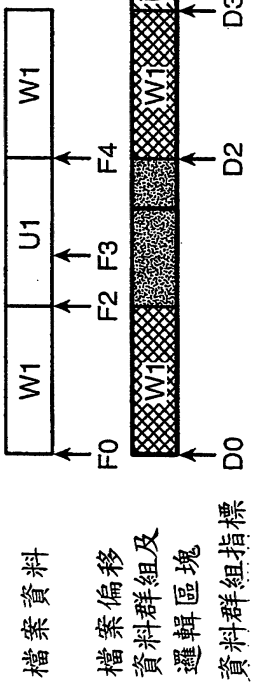

圖14B

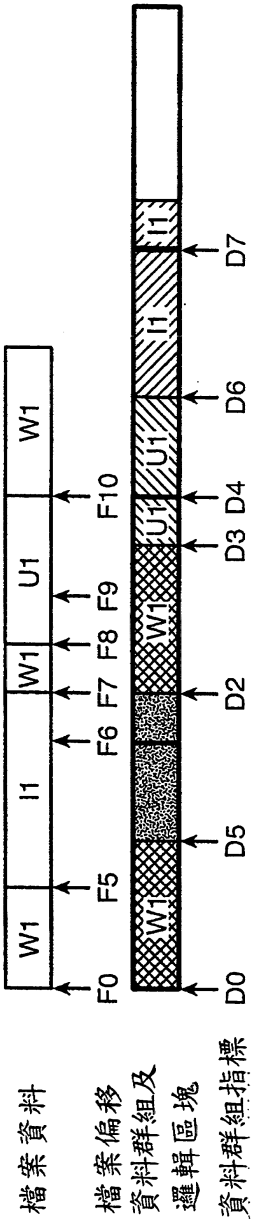

圖14C

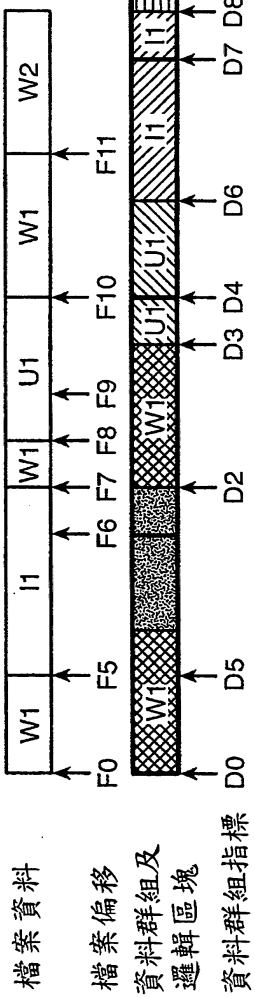

圖14D

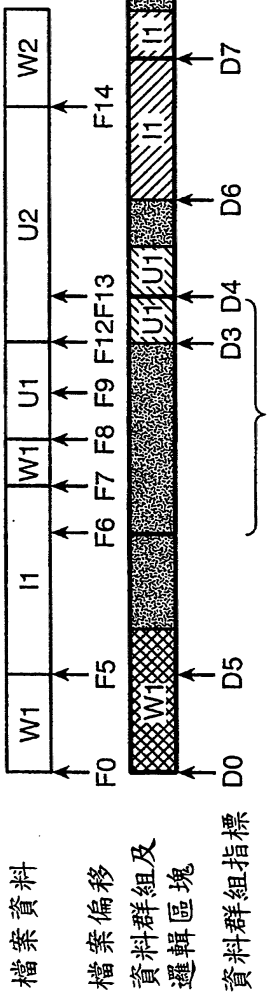

圖14E

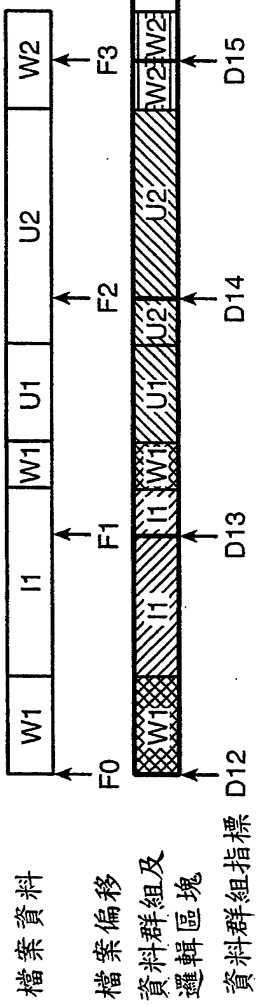

圖15

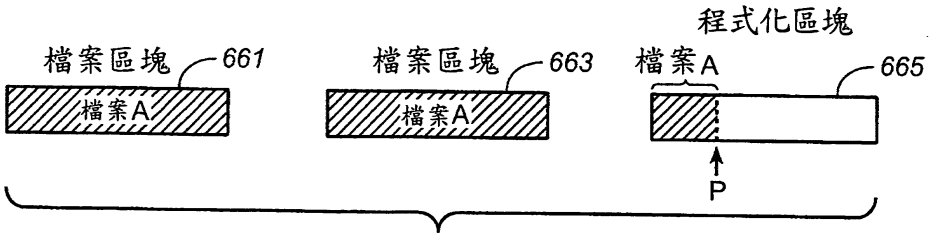

圖16A

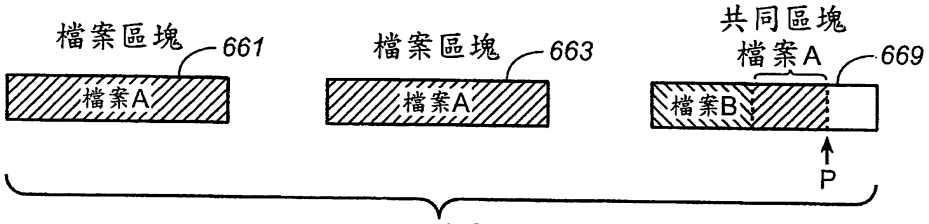

圖16B

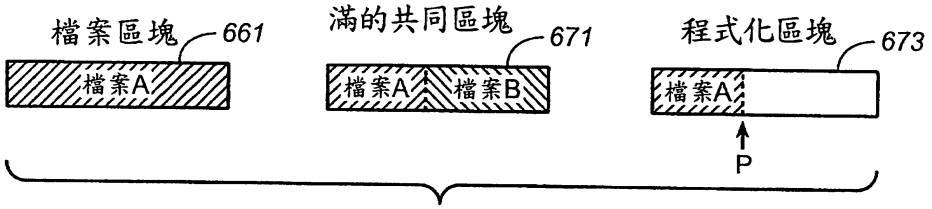

圖16C

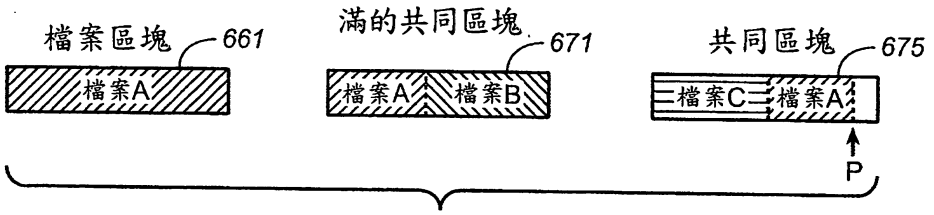

圖16D

准許檔案狀態

檔案狀態	碎形區塊	
	滿的共同區塊	部分區塊
00	無	無
01	無	程式區塊
02	無	共同區塊
10	滿的共同區塊	無
11	滿的共同區塊	程式區塊
12	滿的共同區塊	共同區塊
20	兩個滿的共同區塊	無

圖17

由於程式化資料之檔案狀態轉變

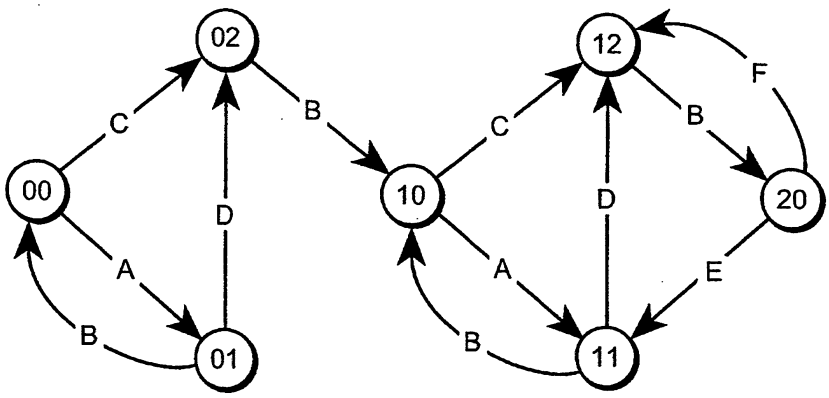

圖18

由於程式化資料之檔案狀態轉變

狀態轉變	觸發事件	描述
00至01	將已擦除區塊作為作用區塊而分配	將已擦除區塊作為程式化區塊而分配，以程式化用於檔案之未知長度的資料或在部分區塊不可用時。
00至02	將部分區塊作為作用區塊而分配	自部分區塊清單選擇部分區塊以程式化用於檔案之已知長度的資料。其變為用於檔案之共同區塊。
01至00	程式化區塊變為滿	用於檔案之程式化區塊變為滿且被指定為檔案區塊。檔案接著不具有作用區塊。
01至02	將部分區塊作為用於另一檔案之作用區塊而分配	自部分區塊清單選擇用於檔案之程式化區塊以程式化用於另一檔案之已知長度的資料。其變為用於檔案之共同區塊。
02至10	共同區塊變為滿	用於檔案之共同區塊變為滿且被指定滿的共同區塊。檔案接著不具有作用區塊。
10至11	將已擦除區塊作為作用區塊而分配	將已擦除區塊作為程式化區塊而分配以程式化用於檔案之未知長度的資料或在部分區塊不可用時。
10至12	將部分區塊作為作用區塊而分配	自部分區塊清單選擇部分區塊以程式化用於檔案之已知長度的資料。其變為用於檔案之共同區塊。
11至10	作為程式化區塊變為滿	用於檔案之程式化區塊變為滿且被指定為檔案區塊。檔案接著不具有作用區塊。
11至12	將程式化區塊作為用於另一檔案之作用區塊而分配	自部分區塊清單選擇用於檔案之程式化區塊以程式化用於另一檔案之已知長度的資料。其變為用於檔案之共同區塊。
12至20	共同區塊變為滿	用於檔案之共同區塊變為滿且被指定第二滿的共同區塊。檔案接著不具有作用區塊。
20至11	將已擦除區塊作為作用區塊而分配	將已擦除區塊作為用於檔案之程式化區塊而分配。檔案已含有兩個碎形區塊，且不可分配另一區塊。因此，狀態轉變必須併入有資料轉變，其中分配已擦除區塊且將用於檔案之所有資料自一滿的共同區塊移動至該已擦除區塊。其變為用於檔案之程式化區塊。資料轉變必須在狀態轉變可視為完成之前作為單個操作而完成。
20至12	將部分區塊作為作用區塊而分配	將部分區塊作為用於檔案之共同區塊而分配。檔案已含有兩個碎形區塊，且不可分配另一區塊。因此，狀態轉變必須併入有資料轉變，其中分配部分區塊且將用於檔案之所有資料自一滿的共同區塊移動至該部分區塊。其變為用於檔案之共同區塊。資料轉變必須在狀態轉變可視為完成之前作為單個操作而完成。

圖19

由於過時資料之檔案狀態轉變

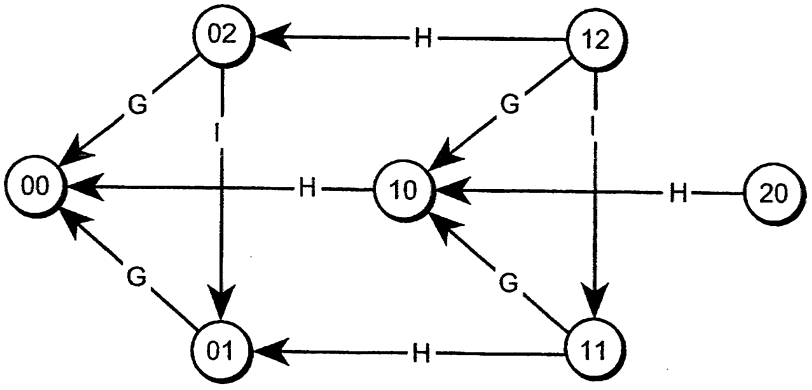

圖20

由於過時資料之檔案狀態轉變

狀態轉變	描述
01至00	用於在程式化區塊中之檔案的所有資料由主機刪除且變為過時。
02至00	用於在共同區塊中之檔案的所有資料變為過時。
02至01	用於在共同區塊中之其他檔案的所有資料變為過時。共同區塊變為程式化區塊。
10至00	情況1 用於在滿的共同區塊中之檔案的所有資料變為過時。
10至00	情況2 用於在滿的共同區塊中之其他檔案的所有資料變為過時。滿的共同區塊變為檔案區塊。
11至01	情況1 用於在滿的共同區塊中之檔案的所有資料變為過時。
11至01	情況2 用於在滿的共同區塊中之其他檔案的所有資料變為過時。滿的共同區塊變為檔案區塊。
11至10	用於在程式化區塊中之檔案的所有資料由主機刪除且變為過時。
12至02	情況1 用於在滿的共同區塊中之檔案的所有資料變為過時。
12至02	情況2 用於在滿的共同區塊中之其他檔案的所有資料變為過時。滿的共同區塊變為檔案區塊。
12至10	用於在共同區塊中之檔案的所有資料變為過時。
12至11	用於在共同區塊中之其他檔案的所有資料變為過時。共同區塊變為程式化區塊。
20至10	情況1 用於在一個滿的共同區塊中之檔案的所有資料變為過時。
20至10	情況2 用於在一個滿的共同區塊中之其他檔案的所有資料變為過時。共同區塊變為檔案區塊。

圖21

由於回收區塊選擇之檔案狀態轉變

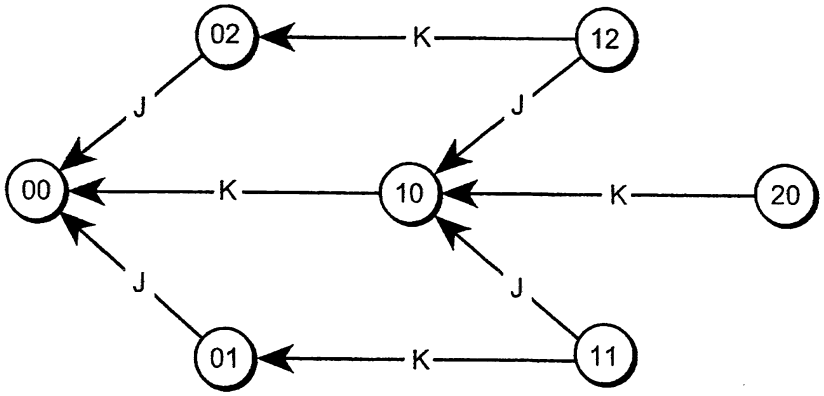

圖22

由於回收區塊選擇之檔案狀態轉變

狀態轉變	描述
01至00	將程式化區塊選擇為回收區塊。
02至00	將共同區塊選擇為回收區塊。
10至00	將滿的共同區塊選擇為回收區塊。
11至01	將滿的共同區塊選擇為回收區塊。
11至10	將程式化區塊選擇為回收區塊。
12至02	將滿的共同區塊選擇為回收區塊。
12至10	將共同區塊選擇為回收區塊。
20至10	將滿的共同區塊選擇為回收區塊。

圖23

將檔案對準至區塊開始

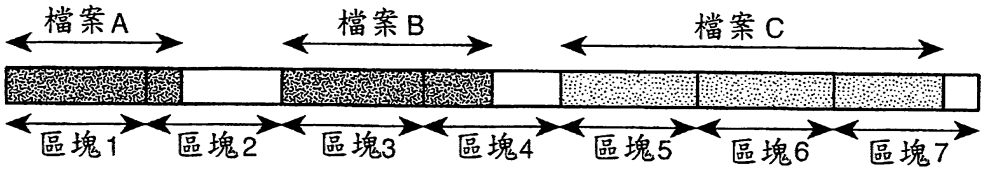

圖24

在將檔案對準至區塊之開始時分配作用區塊

分配情況	現有檔案狀態	主導狀況	待分配之區塊類型之優先級的次序
A	00	未知長度或大於一區塊之已知長度的資料待程式化	1. 已擦除區塊 2. 最大部分區塊
B	00	小於一區塊之已知長度資料待程式化	1. 最適合部分區塊 2. 最大部分區塊 3. 已擦除區塊
C	10	未知長度或大於一區塊之已知長度的資料待程式化	1. 已擦除區塊 2. 最大部分區塊
D	10	小於一區塊之已知長度的資料待程式化	1. 最適合部分區塊 2. 已擦除區塊
E	20	未知長度或大於一區塊之已知長度的聚集資料待程式化，其中聚集資料為待在資料轉變期間重定位之資料與待自另一源程式化之資料的和	1. 已擦除區塊
F	20	小於一區塊之已知長度的聚集資料待程式化，其中聚集資料為待在資料轉變期間重定位之資料與待自另一源程式化之資料的和	1. 最適合部分區塊 2. 已擦除區塊

圖25

檔案在區塊內之鄰接

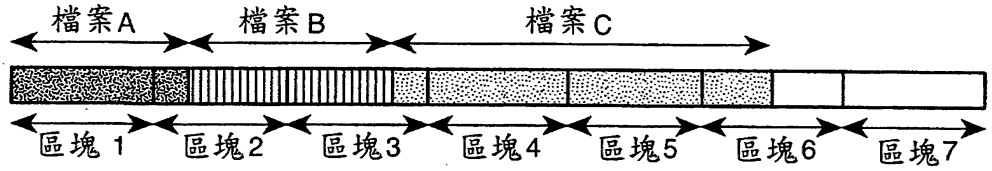

圖26

分配作用區塊以使檔案在區塊內鄰接

分配情況	現有檔案狀態	主導狀況	待分配之區塊類型之優先級的次序
A1	00	用於未知長度或大於一區塊之已知長度之新檔案的資料待程式化。檔案之開始鄰接在區塊內之其他資料。	1. 最大部分區塊 2. 已擦除區塊
A2	00	用於未知長度或大於一區塊之已知長度之現有檔案的資料待程式化	1. 已擦除區塊 2. 最大部分區塊
B	00	小於一區塊之已知長度的資料待程式化	1. 最適合部分區塊 2. 最大部分區塊 3. 已擦除區塊
C	10	未知長度或大於一區塊之已知長度的資料待程式化	1. 已擦除區塊 2. 最大部分區塊
D	10	小於一區塊之已知長度的資料待程式化	1. 最適合部分區塊 2. 已擦除區塊
E	20	未知長度或大於一區塊之已知長度的聚集資料待程式化，其中聚集資料為待在資料轉變期間重定位之資料與待自另一源程式化之資料的和	1. 已擦除區塊
F	20	小於一區塊之已知長度的聚集資料待程式化，其中聚集資料為待在資料轉變期間重定位之資料與待自另一源程式化之資料的和	1. 最適合部分區塊 2. 已擦除區塊

圖27

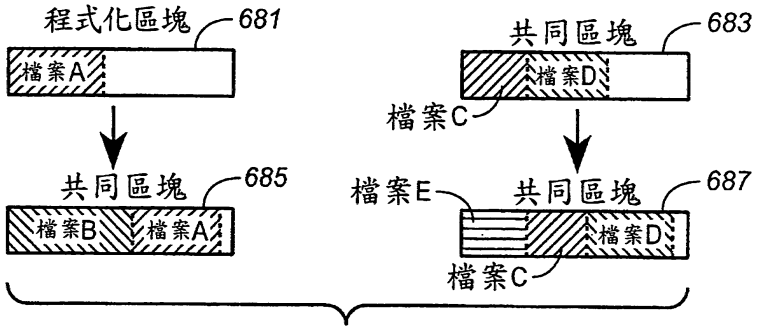

圖28A

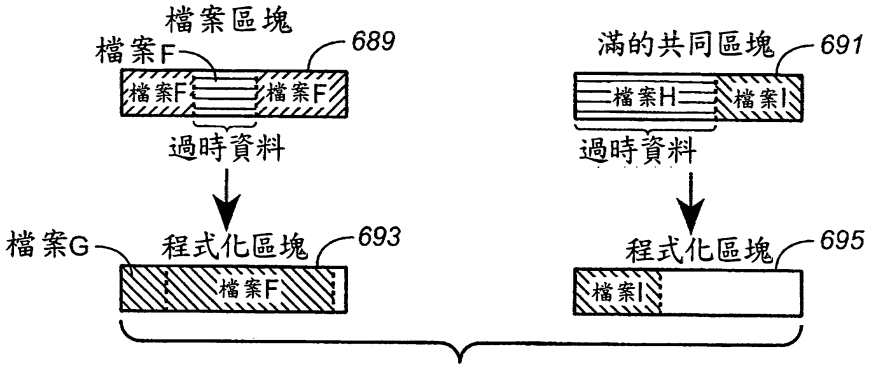

圖28B

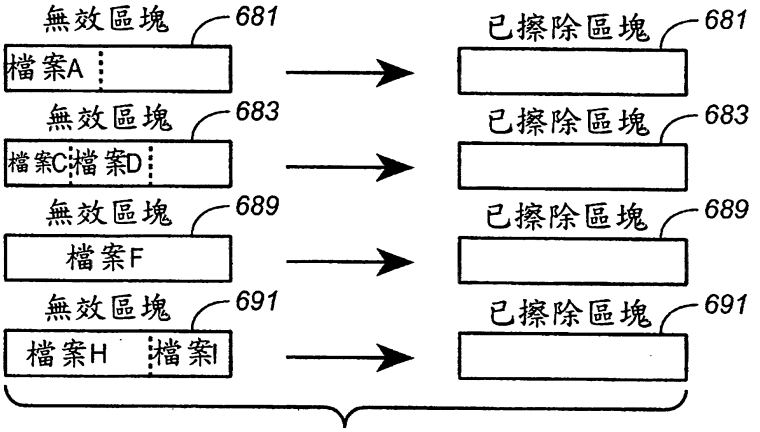

圖28C

圖28D

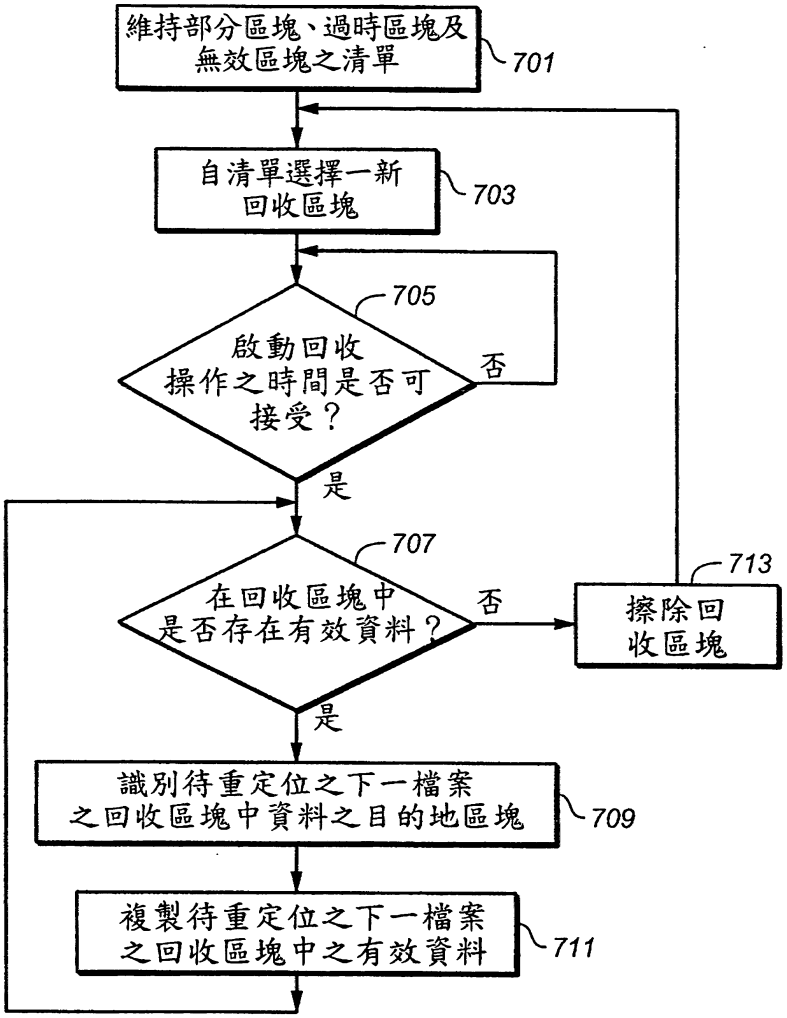
維持部分區塊、過時區塊及無效區塊之清單
701
自清單選擇一新回收區塊
703
705
啟動回收操作之時間是否可接受？
否
是
707
在回收區塊中是否存在有效資料？
否
713
擦除回收區塊
是
識別待重定位之下一檔案之回收區塊中資料之目的地區塊
709
複製待重定位之下一檔案之回收區塊中之有效資料
711

圖29

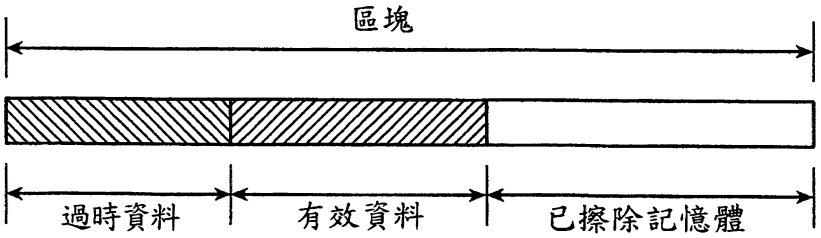
區塊
過時資料
有效資料
已擦除記憶體

圖30

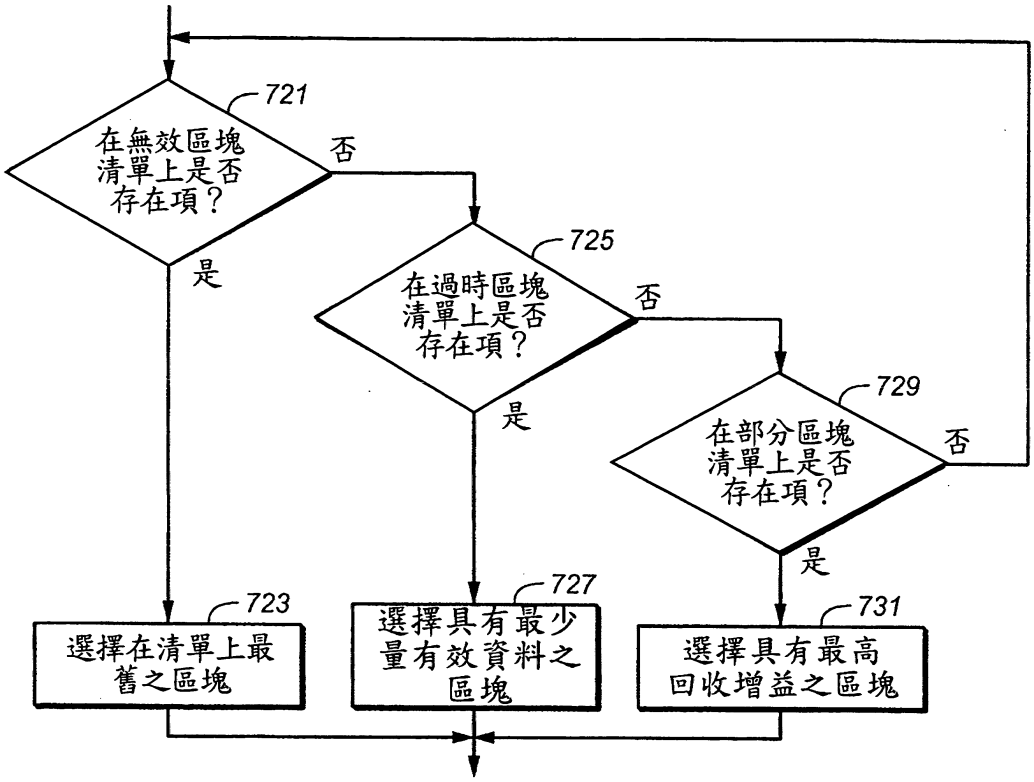

圖31

區塊類型	區塊內容			區塊清單
	有效資料 (V)	已擦除容量 (E)	過時資料 (O)	
程式	是	是	無關	部分
共同	是	是	無關	部分
滿共同	是	否	否	無
			是	過時
檔案	是	否	否	無
			是	過時
無效	否	是	是	過時
已擦除	否	是	否	已擦除

圖33

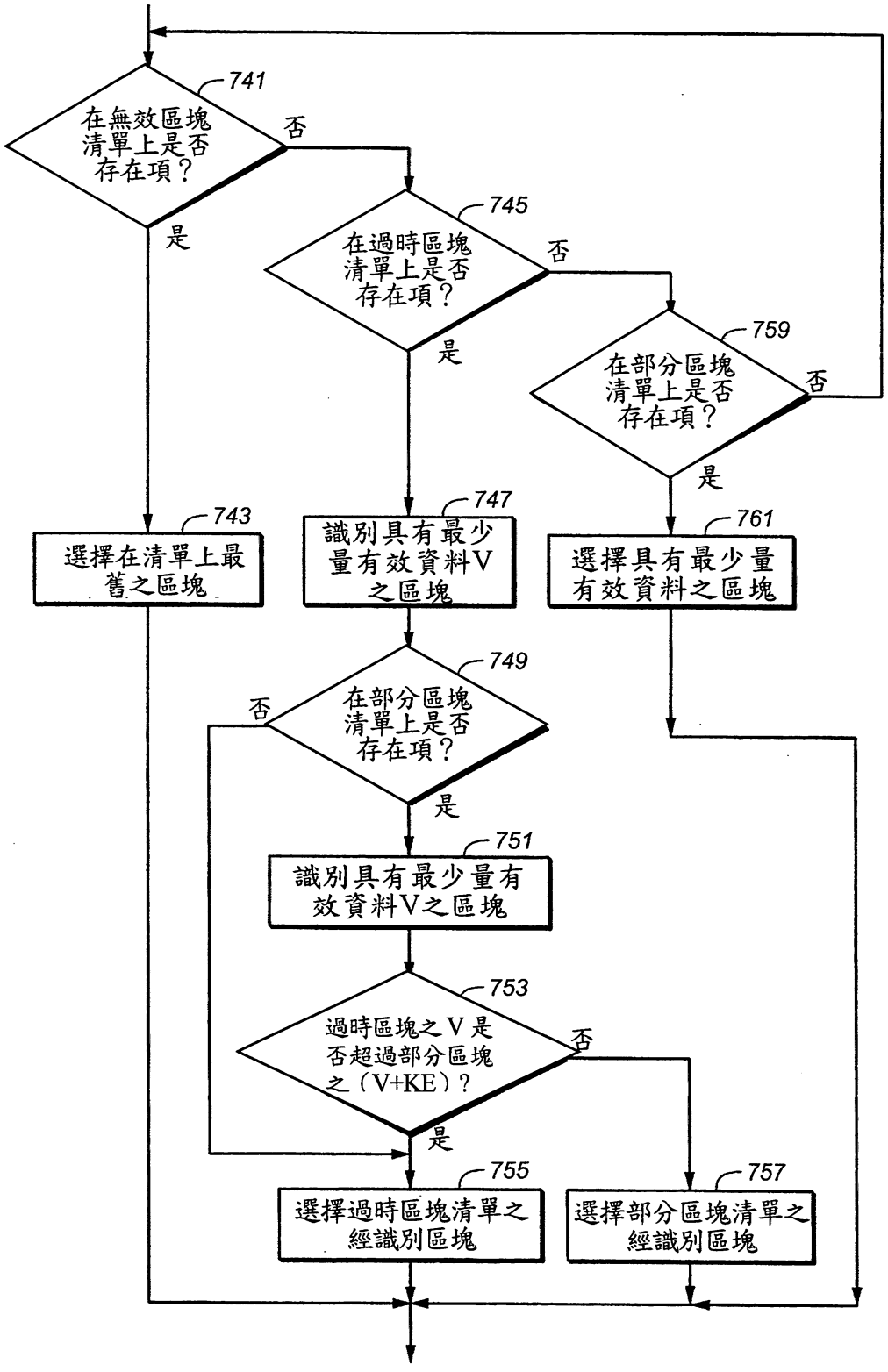
741
在無效區塊清單上是否存在項？
否
是
745
在過時區塊清單上是否存在項？
否
是
759
在部分區塊清單上是否存在項？
否
是
743
選擇在清單上最舊之區塊
747
識別具有最少量有效資料V之區塊
761
選擇具有最少量有效資料之區塊
749
在部分區塊清單上是否存在項？
否
是
751
識別具有最少量有效資料V之區塊
753
過時區塊之V是否超過部分區塊之（V+KE）？
否
是
755
選擇過時區塊清單之經識別區塊
757
選擇部分區塊清單之經識別區塊

圖32

七、指定代表圖：

(一)本案指定代表圖為：第（ 9C ）圖。

(二)本代表圖之元件符號簡單說明：

161	連續邏輯位址空間
163	區塊/邏輯至實體位址表
165	記憶體單元陣列
167	元區塊
169	元區塊
171	元區塊
173'	功能

八、本案若有化學式時，請揭示最能顯示發明特徵的化學式：

（無）