# 記憶與速度——談計算機中的記憶層次結構

科學月刊
第十七卷第九期
謝清俊

「儲存」是計算機很重要的功能。不僅是數據資料，程式也是儲存在計算機中的。這個性質，以目前大家對計算機的了解而言，似乎是理所當然的常識。可是，在計算機的發展過程中，儲存程式的觀念（store-program concept）是發展旳里程碑之一。試想，若計算機不能儲存數據或程式，會變成什麼樣兒？那就沒有「計算機」了！它充其量不過是一具複雜的電子儀器罷了。

「儲存」靠一些「記憶體」來執行。在計算機中，記憶體的應用無所不在，並不限於專門做儲存的「記憶單元」、「磁碟」及「磁帶」等等。在計算機裡，凡是資料經過任何一過運作過程的先後，都要用一些暫時儲存的裝置，來存放待處理或已處理的資料。譬如，數學運的前後，數據必須妥為保存。又如在計算機的進出管道（輸入、輸出單位）上，資料亦須留置以待運輸或取用。這些都是儲存的功能，且必須靠些記憶體臨時性安置資料的工作，記錄器（register）或緩衝記憶器（buffer memory）是常用來做這些事的設備。這些記憶體的容量通常都很小，速率非常快。高速率的要求是必然的，因為它必須能匹配以電子速度進行的其他部門，如計算或傳輸，才能獲得較好的資料處理能力。這種速率匹配的問題和記憶體與生俱來，一直到現在還是設計工程師必須小心應付的重要問題。

大量儲存資料的記憶體，可概略分為主記憶體與次級記憶體兩類。主記憶體是計算機工作時主要的資料儲存場所，而次級記憶體則多以週邊設備的形式出現，作為大量儲存資料的倉庫來用。例如：說到計算機結構時所指的記憶單元是主記憶體，而磁碟、磁帶等是次級記憶體。

這篇文章將介紹計算機裡主記憶體和次級記憶體如何組成一個有系統的記憶層資結構（memory hierarchy），以及如何解決它們之間速率匹配的問題。這些討論將有助於對計算機結構的了解。

## 記憶體簡介

目前的主記憶體均由電子電路構成。由於其結構相當規律，是最適合以積體電路技術製作的電路。它是各種記憶體中速率最快的一類，也是單價最貴的。在積體電路發展以前，延時線（delay lines）及磁蕊（magnetic cores）都曾是風光一時的主記憶體元件。以積體電路所做的記憶元件，它的存取速率和電子訊號須經過的途徑長短有密切的關係。所以它的速率約略與晶片的體積成反比。近十幾年來，積體電路製造的密度雖然一直增大，約每二、三年加大四倍，然而每塊晶片上的記憶容量也隨此比例加大，這使得晶片的大小沒有太大的變化。因此，積體電路記憶元件的容量雖然不停地增大，可是主記憶體存取速率的改進就顯得少的多。

記憶體育二個基本功能，是「存入」和「取出」資料。在專業上常作「寫入」和「讀出」。為了使用和設計上的方便，主記憶體存取資料時有一個固定大小單位，稱為「字」（w-ord）。這個字的長度經常是和中央處理機處理資料的單位一致，因此，它代表計算機特性的一個重要指標──處理的能力。我們常見的所謂八位元、十六位元電腦，即指此字長。可是， word 被譯為字，不甚委當。譬如 computer 是一個 word　　，譯作中文時「計算機」是一個三個字的「詞」，不是字。此譯名沿用已久，積習難改。然而明瞭此間之差異將使你深得三昧。

主記憶體做存取工作花的時間是固定的，稱之為記憶週期。在早期的設計中，一個週期包含一次取出和一次存入的動作，而且一定是依先取後存的次序。當計算機開機以後，主記憶體即不停的重複期記憶週期。所以，處理機要

存取資料時，必須午等待到一個新週期的開始時刻，才能使用記憶體。在這種設計的安排之下，處理機和主記憶體必須依照記憶週期來協調同步，故稱之為「同步型」之主記憶。由於此設計較為簡單，迄今仍有部分機種沿用它。

較新的設計就不這樣做了。由於「取」、「存」的二個動作不一定是非連在一起不可，而且根據統計，「存」的機會比「讀」的機會小得多。因此記憶週期被分為「存」與「取」二個週期，可單獨工作。更由於「存」較「取」略快，所以同步的形態也沒有了。主記憶體空訛閒時，隨時可以聽令做「存」或「取」的動作，這種設計的方式稱為「非同步型」的設計。新式的主記憶體多用這種方法。然而為了討論上的方便，下文中我們不再分「存」與「取」之差異，一概以 T 表示非同步型主記憶體之記憶週期。譬如，讀取 r 個字串時，則 rT 的時間。

主記憶體是屬於隨取記憶（random access memory）的性質。簡稱為ＲＡＭ，它的特徵（已如上述）是無論存取什麼位置的字，其花費的時間是一定的，與該字的「地址」無關。與ＲＡM相對的是ＳＡM，是順取記憶（Sequential accwss nenory）的簡稱。這種記憶存取的時間是存放地址的函數，因為其資料的存取必須按一定的次序。譬如磁帶就是典型的例子。磁碟的結構是介於北二者之間的，其儲存軌道的尋取是類似ＲＡM的結構，而軌道內的資料仍然是ＳＡM的方式安排。這種結構稱為直接取用，而此類設備稱為直接取用的儲裝置（direct access storage device, 簡稱ＤＡＳＤ）。

本文要談的是處理機、主記憶體和磁碟類記憶裝置（ＤＡＳＤ）之間的關係。磁帶在早期曾扮演磁碟的角色，可是目前漸淪為作備分資料或離線交換資料的媒體，與計算機處理資料的工作關係，逐漸疏離了。

磁碟和磁帶都是以電磁特性儲存資料的裝置，所以它的速度較慢，約在幾十分之一秒的範圍內，而主記憶體的記憶週期則以千萬分之一秒的單位計算，因此，北二者之速率有數十邁倍之差距。再者，處理器內運算的速率更以億分

心一秒來算，又約較主記憶體快將十倍，如何安排一個有效率的整體記憶體架構，以使得記憶體的整體效率像ＲＡＭ，而價格像ＳＡＭ，一直是設計者面臨的課題，也是計算機內記憶結構安排的中心思想。

## 記憶體的層次結構

目前，計算機裡的記憶體結構分為三個層次：第一層為隱藏記憶（cache memory），這是直接和處理機相連的；第二層是主記憶體，它介於隱藏記憶與次級記憶體之間；第三層是資級記憶體，共主要的設備是直接取用記憶裝置（ＤＡＳＤ）。此結構如圖一所示。

隱藏記憶 cache 一詞是由法文而來，原意指隱藏東西的地方。由於使用者用計算機時，對有關記憶體的問題之思考仍是以主記憶體為對象，對 cache 視而不見（像是透明的），故稱之為藏記憶。

cache 是一個小而高速的ＲＡＭ，通常只有數千至數萬位元組（bytes）大小，用的是最快速的電子記憶元件，約比主記憶體快十倍上下。通常，一個處理機裝置一個 cache 。c-ache 的構想最由見於英國劍橋大學韋爾克斯教授（M. Wilkes）1965 年的研究報造，它的主要角的是解決處理機與主記憶體之間的速率匹配問題。

cache 和主記憶體之間存取資料之單位較字為大，稱為框（frame），通常框之大小在 4～128 位元組之間，而常用的大小是 32 位元組。

當處理機向 cache 拿資料時，要是資料已在 cache 中，則直接取用，若不在 cache 中，則必須到主記憶體中，將該資料所有的那一框資料一起搬到 cache 中來，再予取用。因此在電路上較為複雜，它必須維持一個 cache 的目錄（directory），以便隨時查閱所要的資料是否在 cache 中。同時還要些管理的邏

輯，以快定在不同情況時，採取適當的尋取或存入資料的步驟。請參考圖二中的例子說明。

主記憶體和次級記憶體之間，以頁（page）為存取資料的單位，頁又較框大，這樣的系統稱為分頁系統（paging system）。通常，頁的大小配合磁碟的結構，將一段（sector）貝料作為頁心大小，其數值常在一千位元組左右。雖然此二者心間速率差在數十萬倍，但是很少見到利用另一個 cache 調協二者間之差距。然而在去年（1985），ＩＢM公司宣布推出在ＤＡＳＤ中加 Cache 的結構，在想的情況下，可將磁碟的平均存取速率自 35ms 降下 10ms 以下。相信這類系統，不久便會被廣泛採用在各種不同的磁碟系統中。

## 無 cache 的系統

讓我們思考一種最單純的記憶架構：處理機和主記憶體直接連接的狀況，沒有 cache ，也沒有次級記憶。

通常，主記憶體是由許多完全相同的記憶庫（banks）構成的，而不是整塊的ＲＡM。每個ＲＡM做的記憶庫都有獨自的存取功能。這些記憶庫又稱記憶模組（memory module），通常以數目編號來識別之。例如，若有M個模組，則由 0，1，....編至（M-1）號。為有效利用電路，M常為 2 的整數倍數。這樣模組化的目的有二，其一是加快速度以配合中央處理機心需。因為M個模組可以同時工作，所以存取速度可以有快約M倍之潛力。其二是當記憶系統出毛病時，有問題的模組可以不用，在中央處理機的控制之下，可以重新將沒有毛病模組編組使用，使計算機能繼續工作。

資料存取的地址是經過電路的選擇來控制的。換言之，地址將被選擇電路轉換為記憶模組裡的位置。這轉換的方式有二種：一、交織式（interlaving）；二、分割式（partitio-ning）。在交織式的方式下，連續的址依次指定在不同的模組中。例如：第○、Ｍ、２M.....等地址放在第○個模組中，第 1，Ｍ＋１，

２Ｍ＋１，.....等地址放在第一個模組中，餘此類推。在這樣的安排下，一個地址的指令，就可以取出M個連續的「字」。所以，平均每個字的記憶週期就降至約Ｔ／Ｍ。這是一個平行的結構。

在分割式的方式下，一群連續的地址放在同一個模組中。若每個模組容量是Ｃ個字，則可能第 0 個模組的位置是由 0 至（Ｃ-１），第一個模組的位置是Ｃ至（２Ｃ－１），.....餘此類推。由此觀之，交織式的安排有較高之效率，若處理機要處連續ｒ個地址資料時，在分割式的記憶下約需ｒＴ的時間，而在交織式的結構下只需〔ｒ／Ｍ〕・Ｔ（註一）的時間。然而在電路上，交織式的將較分割式的複雜得多。

為了建立處理機與記憶模組間的溝通，需要一個管理的電路稱為記憶控制器（memorycontroller），它將處理機存取的要求傳至記憶模組，並將記繶模組之反應和結果送回給處理器。此外，它還需應付對同一記憶模組幾乎同時提出存取要求的情況。處理器與記憶模組之間的連接通常有二種方法：一、用交換機（switching mechanism），二、用匯流排（bus）。用交換機可將任何一個處理器和任何一個記憶模組直接相連，它可使任何一個處理器借此通道，與任何一個記憶模組完成一次存取的動作。而這個匯流排是大家共用的。匯流排的控制部分，將負責匯流排使用權的分配工作。成敦的匯流排設計，並不用匯流排把一個處理器與對應的記憶模組完全連接在一起，而是將處理器的要求和記憶器的答覆分別處理。

當一次存取工作在匯流排中占用的時間遠較Ｔ／Ｍ為小時，匯流排的效率將趨近於交換機的。若有Ｎ個處理器，Ｍ個記意模組，交換機的複雜程度將與ＭＮ成正比，而匯流排之複雜度僅與Ｍ＋Ｎ成正比。然而當Ｍ與Ｎ都很大時，這二個系統都會性生問題，對交換機而言將需太多的電路，而匯流排的反應時間將會加長。這種Ｎ個處理器分用Ｍ個記憶模組的安排方式，已是平行處理的結構，這種想法在硬體系統裡早已存在。

Ｎ個處理器能使Ｍ個記憶模組忙碌到什麼程度呢？這是個困難的問題，因為許多處理器若要同時利用同一個記憶模組，會產生相當大的干擾。

因為一個程式（經常）會產生連續的地址，所以，若是Ｎ=１，可選擇Ｔ／Ｍ之值較處器產生地址之間隔時間略短，就可匹配處理器之速度和記憶體的資訊處理流量（through p-ut）。

當Ｎ＞１時，處理器之間性生地址的情形將是混合的隨機，而近似於亂數尋址的狀況。在此情形下，根據白斯凱特（F. Baskett）和史密斯（A. Smith）在 1976 年的分析，記憶體使用心效率為。

上式的Ｕ是使用率，它表示每記憶週期中平均有效的使用程度。若Ｎ＝Ｍ，則Ｕ＝0.586，即約有 60%的使用率。若Ｎ＝２Ｍ，則使用率為 76%。若處理器產地地址的間隔是ＴＮ／ＵＭ，則處理器與記憶模組之間可得一完全之匹配。

這情況似乎不壞，然若使用 cache，則會更好。

## 使用 cache 之分析

如果在上述的例子中，在主記憶體與處理機心間加了一個 cache，情形會變成怎樣呢？當處理機向 cache 要資料時，若是該地址已在 cache 中，則直接由 cache 中取用。如果地址不在 cache 裡，則須至主記憶體中尋取，找到該投記憶框後將之存入 cache 中，並將該字送到處理機去。由於 cache 與主記憶體之間是以框為單位傳輸，而框中心地址又是連續的，因此使用效率將提高。

設若框架心大小為Ｂ位元組，而主記憶體為Ｍ個交織式的記憶模組，則需（Ｍ－１）／２次來尋找所需取用的第一個位元組，再加上Ｂ次的時間來取其餘，因此，其使用效率為

若Ｂ為３２位元組，Ｍ＝Ｂ，則Ｕ約為９０％。故 cache 可提高記憶模組的使用率，Ｎ之數目無關。

除上述的理由外，cache 還有許多其他好處。舉些數字，就可明白

一個典型的主記憶體約在１Mbyte～６４Mbytes 之間。常用的 cache 是４～３２Kbyt-es，典型的主記憶體記憶週期是 500ns（註二），而 cache 是 50ns。在一般的應用中，不中的機會（miss ratio，也就是指處理機向 cache 要求，而 cache 中無該記意位置的情形）是５％，這是在 cache 有２０Kbytes 之估計，然而針對某些應用時，此比例可以少到１％。在此情形之下，平均的取存週期是　　0.59*50+0.05*500=72.5nscache 雖貴，但由於量少，僅占全記憶體價格之 1/10 以下。然而整個記憶體的效能，卻幾乎與 cache 之效能相差無幾。

以上的討論說明一個不錯的記憶結構模式是：在處理機外的記憶體是交織式記憶模組與 cache 之綜合，並且可以調整 cache 的大小，使達到 95%９５％以上的使用率。

存在 cache 中的資料，我們可稱之為程式的工作集（working set）。通常它們是程式中最常用的一群數據，無論它是指令或是 data。為了減少尋不中的比例，程式設計人員必須了解工作集的觀念，在程式中大要經常更易其常使用心數據。這觀念稱之為程式的區域化（programming for locality），這是很容易做到的，亦能使程式人員充分發揮計算機的威力。

註一：〔ｒ／Ｍ〕表示較ｒ／Ｍ大的最小整數。

註二：ns 為 10 秒（nanoseconds）。