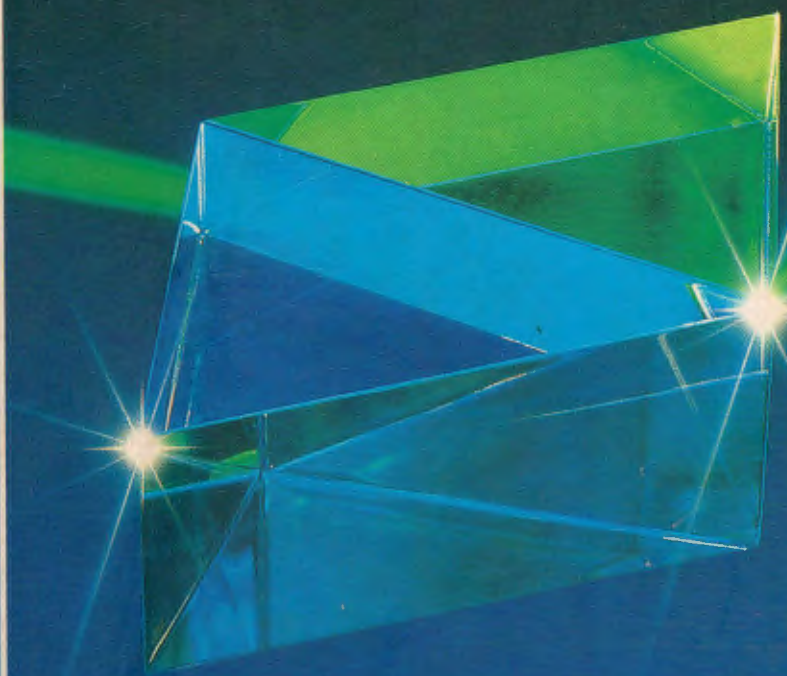
突破640K程式設計技巧

瑩圃電腦軟體研究開發部



SoftTip 尖端電腦

突破640K程式設計技巧

瑩圃電腦軟體研究開發部

<本書簡介>　<書號> 73330

開 數: 16 開　　頁數: 336 頁
所需硬體: IBM PC AT 或 PC/XT 或 PS/2
所需軟體: 無

<本書特色>

電腦軟、硬體功能不斷地進步，但遺憾的是記憶體仍被限制在640K，就像一個大力士，被綁上了雙腳，動彈不得。尤其在中文下更顯得記憶體太小之痛苦。故本書的目的乃在於告訴讀者如何解開電腦技能的枷鎖。

<內容概要>

＜本書簡介＞　＜書號＞ 73330

開 數: 16 開　　頁數: 336 頁
所需硬體: IBM PC AT 或 PC/XT 或 PS/2
所需軟體: 無

＜本書特色＞

電腦軟、硬體功能不斷地進步，但遺憾的是記憶體仍被限制在640K，就像一個大力士，被綁上了雙腳，動彈不得。尤其在中文下更顯得記憶體太小之痛苦。故本書的目的乃在於告訴讀者如何解開電腦技能的枷鎖。

＜內容概要＞

＜相關書籍＞

1. MS-DOS 應用手冊
2. DOS 5.0指令使用手冊
3. DOS 3.3 使用與實例學習手冊
4. DOS 與批次檔之技巧
5. DOS 系統函數呼叫手冊

目　　錄

第一章　原始未擴展之系統

第二章 電腦的心臟

第三章　新一代的記憶體

第四章　更多的記憶體

第五章 擴充記憶體

第六章 擴充記憶體管理

第七章 延伸記憶體

第八章 硬體方面的關連性

第九章 品質升級的抉擇

第十章 最終的改進

第十一章 幾乎是多功

第十二章 多工系統的實際運作

第十三章 眞的需要 *DOS 4.x*嗎?

第十四章 要選擇什麼呢?

第十五章 崩潰 (CRASH)過程

第十六章 視窗及水晶球(Crystal Balls)

附錄A LIM 4.0 EMM 所提供的高階函數

附錄B 16進位的基本運算

附錄C 名詞解釋

第一章　原始未擴展之系統

在進入本書之前，讀者必須對下列之機器—PC,XT 或AT先有粗淺之認識。(爲了分類簡單起見，除了本書所述兩者之主要不同點外，80286 及80386 系統都視爲AT型式的系統。) 然而，跟一些平常的書籍稍微不同的地方在於: 本書並不想涵蓋原始PC及其後續機種的剖析。本書所著重的目標是如何在386 AT型式的機器上充分利用640K以外的記憶體，而不是討論一般在640K之內的動作。所以在實際應用上，讀者可從本書中學到如何將一些程式碼切換到大塊的擴充記憶體之下執行，以及如何操作這些擴充記憶體。)

本書的大部份內容主要在提供一些資料，使讀者能一覽從前未曾發現的部份，或是未曾注意到的細節。當然，讀者也可能注意過擴充或延伸記憶體，以及超越 DOS原始的 1 Mega記憶體限制的問題 (對640K的用者而言) 。讀者可能想要使這640K能採用多出來的幾mega記憶體，而同時保有系統的相容性，以及與其他價值數百萬的軟體之相容性。

對這些問題的討論，本書會先從整體上的瀏覽開始，然

後就很快地進入讀者極有興趣探索的地區 —— 擴充記憶體 (expanded memory) 及延伸記憶體 (extended memory)。如此可以在許多令人混淆不清的記憶體型式中做一番澄清，然而，最重要的還是要瞭解最基礎的東西 —— 基本系統。

1.1 系統的實際限制

BONANZA 螢幕 (R)

(Physical limit of the system)

從物理的角度來看，加入系統內的記憶體數目並無眞正的限制，只要有辦法在電路上不斷地加入記憶體即可。一個Gigabyte行嗎? 當然沒問題，只要你再加上一個超大型的電源供給器以及其他方面能夠配合就行了。

然而在現實世界裡，會有一些現實世界的問題。假設於系統上不斷加上記憶晶片，在超過某一個數量時，電腦就無法找到這些晶片了。使用者可以看見這些晶片插在系統上，並且理直氣壯地指著晶片嚷嚷：「不是在這裡嗎? 不是在這裡嗎 ?」然而，對電腦而言，這些晶片根本不在那裡。說得更清楚一點，當電腦找尋記憶晶片時，只能找到某幾個特定的晶片，而忽略這些超過臨界數量的晶片。

問題就出在電腦不能 "看"， 因爲它沒有視覺，只能偵測(detect)是否有晶片的存在，就如同置身迷宮的老鼠，周圍都是通道，有些通往老鼠窩，有些則通往放置乳酪的地方,但是如果這些通道不相接，則老鼠就無法找到窩及乳酪。

老鼠 (不要跟滑鼠搞混了) 就像微處理器晶片。在我們的電腦內，這些通道就像接在電路板上以通往外界的接腳

(pins) 。在舊型的 8086CPU上，只有 20 根位址接腳——其他則另有功能。每一根位址接腳，與其他位址接腳合用時，就定出許多位址 (address)。在電腦數學上，這些位址的數目可達到 2的 20 次方，或者 2乘以 2，再乘以 2，諸如此類的數學運算，可參閱表 1～1 。這可指出 1048576個特定的記憶位址，從 0000 到 FFFF ，一共 1mega byte，8088 也就只能定址這麼多了。

位址接腳	可能的位址
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024
11	2048
12	4096
13	8192
14	16384
15	32768
16	65536
17	131072
18	262144
19	524288
20	1048576
21	2097152
22	4194304
23	8388608
24	16777216
25	33554432
26	67108864
27	134217728
28	268435456
29	536870912
30	1073741824
31	2147483648
32	4294967296

表1.1: 2的乘方表示了處理器晶片的位址接腳及可能的位址範圍。請注意，有些數目字與電腦系統上經常見到的數目相吻合，如 512位元組在磁碟上爲一磁區 (sector)，1024K 位元組爲一 Mega byte，而 64k區塊則含有 65536位元組。不只位址的計算用到此表，幾乎所有涉及電腦的數目，都與此表有關。

超過 1Mega這個特定的數目時，你仍能插上更多的記憶晶片，一切都隨你。但是 1 Mega byte的位址空間已經是那可憐且過時的 8088CPU所能找到的極限了，無論對於記憶體的需求是否大於此數。

當然，一定有人會問如何只利用 16 位元的位址暫存器來處理 20 根位址接腳。有些人可能在經過徹夜苦思之後，就領悟出是怎麼一回事了。在此長話短說，答案在於把絕對位址 (absolute address) 分成兩個部份 —— 區段及位移 (segmend and offset)。這種位址的格式爲 segment:offset，或由 DEBUG 以 16 進位顯示，如 0000:0000。

雖然我們會稍微用一些算術把絕對位址換算成上述的格式，以配合 16 位元暫存器；以及討論這種換算所造成的特殊情形，但這些東西並非我們的重點，也不是大部份讀者所需要的。然而，對於有興趣的讀者而言，以下所討論的部份已可充份地涵蓋必要的內容。事實上，格式爲 segment:offset的十六進位位址只需四個十六進位 "數字" 表示即可 (注意: 在組合語言內，即是 4個 byte 的大小)，而且我們所關心的是那些絕對位址，而不是一些毫無意義的十六位元世界。

爲了避免使用者產生只有十六位元位址暫存器獨立運作

的錯誤觀念，而對所能定址的空間施加了嚴厲的限制，所以在此澄清：兩個十六位元的位址暫存器一起使用，在理論上可處理 2的 32 次方個獨立的實際位址。即使只有十六位元暫存器，我們仍可利用二個暫存器合作，而不致發生容量不足的情形。

令人可喜的是，先有了 80286，再來是 80386，都加上了更多的位址接腳。參考前列的表 1-1，此表顯示了位址接腳與所能定址的空間之指數關係。加了更多的接腳，就能增加更多的定址空間。

80286 CPU 可處理 16 mega位元組空間 (然而在原始的 IBM AT 上,因其他硬體限制，所以只能定址 4 mega)。80386 打破了 4 mega 位元組的限制，由於 80386的 32 位元硬體架構使得它能定址到 4 giga 位元組，也就是 4000 mega位元組。每增加一根定址接腳，就能多二倍的定址空間，所以這種增加量是很驚人的。

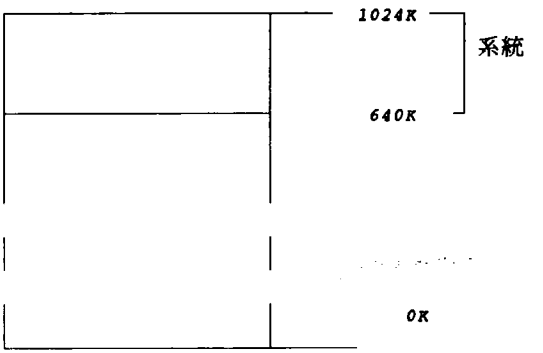


圖 1-1　原始 1 mega 的配置情形

圖1-1 這種配置情形免除了先前系統總空間只能有 64K

的限制。PC的設計者在 1 mega 的位址空間內，保留了384K，供系統使用。

然而，DOS 爲了遷就 8086/8088機種，造成了後述這種進退兩難的局面。80286 被限制只能在 1 mega 以下的空間執行，連 80386也必須受此限制。雖然這有點像只爲了裝載一條麵包而把豐田汽車 (TOYOTA) 加上千匹馬力的引擎。但是，一樣可以正常動作，且 80286及 80386兩者都能有較好的表現，因爲兩者都在較快的時鐘 (clock)下工作。

使用者很難立即注意到其他方面的差異，爲什麼有許多舊型的 8088 軟體可以在 286或 386上執行得很好，但有些則否。不過這些無法正常執行的軟體通常會很快地更新版本，以修正這些不相容的情形。那麼 DOS呢? DOS 3.X 版主要是爲了配合 80286 AT 而推出的。而 DOS 4.X系列版本從 4.0 版開始即包含了特殊的 80386功能支援。DOS 3.X 版與舊型的 8088 機種是往回相容的 (backward compatible)同理，4.X 系列版本一樣與 8088 機種往回相容，至少在理論上是如此。從另一個角度來看，較舊的 DOS版本，從2.X 版本開始也都可以在較新機種的機器上執行。(註: 至少在 4.01版之前的 IBM PC-DOS 4.X,存在著某些嚴重的往回相容問題，包括了無法讀取舊版 DOS所規格化的硬碟，甚至連它們標準的 IMB系列機種都無法避免。不過在出版之前，這些問題似乎已經修正，或者確定會補救。)

這些設法維持相容程度的作法有點不可思議，特別是利用某些牽强附會的作法，或者是爲了配合某些障礙，這些障礙可從許多熱門的遊戲裡看出來。大部份的讀者可能還記得當年蘋果流行的盛況，當蘋果電腦以前發展的程式不再與其他機種相容時，這些遊戲仍可利用某些作法而延續一些時日

，這無疑地打了一劑强心劑。

很不幸地，前段所述可能是一個好消息，但也是一個壞消息。因爲要維持這種相容性，DOS 以及所有在 DOS之下所執行的程式都被限制在 1 mega 的位址空間內。不論是延伸(extended)或者是擴展 (expanded) 記憶體。在 DOS下之任何程式若能超過 1 mega 的限制，純粹是利用某些特殊技巧所達成。現在讀者應該知道了吧！目前之所以無法超過 1 mega 的限制的原因是因爲：事實上，在 DOS看來，超過此限制的部份根本不存在。現在如此，過去之 DOS亦如此，以後也是這樣，只要是在目前的 DOS環境下，就不可能存在。

另一種操作系統又如何呢？在 DOS之外的世界又是什麼樣子呢？的確，OS/2有可能。也許有那麼一天，有其他將會出現、而我們還無法瞧出其端倪的軟體推出時，所有的事情就都成爲可能了，或者是幾乎成爲可能。然而，若眞的有這種作業系統問世，而且能夠眞正地把 DOS推到一旁，則此作業系統必須包含足夠的功能，以便使此作業系統的價值高到足以拋棄我們現有的大部份軟體，以及許多現有的硬體。因爲那些不可思議的往回相容性將必定無存在的價值。

然而，既然現在有了 DOS，而且 DOS存在的日子已有一段不短的時間，再加上是有些新的技巧及技術加進了 DOS之後，使 DOS能做到一些以前被認爲無法達成的功能。所以讀者可以確信：在其他作業系統奪去 DOS光彩之前，還需經過一段很長的時間。

有一件事必須牢記在心：在 DOS之內，有些東西無法趕上現有技術的腳步（特別重申此 "無法" 之存在）。這包含了舊有 1 mega 空間的問題。因此，只要使用了 DOS，若讀

者想用到 1 mega 之外的記憶體，就必須設法愚弄 DOS，以及所有在 DOS之下執行的軟體，使 DOS及這些軟體認爲他們還在 1 mega 之記憶體內執行，以及我們所用的是具有 1 mega記憶體的 8086 機器。這就是擴展記憶體的動作原理。然而，在我們討論這個主題之前，讓我們先看一看我們即將討論的機器實體，以對其餘的討論能有較清楚的瞭解。

1.2 實際的機器

BONANZA 堃圖 (R)

(The Physical Machine)

從一個具有 20 根位址接腳的微處理晶片開始說起，此20根位址接腳足以處理 1 mega 的位址空間。在 DOS還沒出現之時，PC就以此晶片爲基礎而設計出來了。此時，具有1 mega記憶體的桌上型電腦似乎是令人難以置信而且是個非常滑稽的想法。甚至有些早期的 IBM PC 在出售時，竟然只配備了 16K的使用者記憶體，這也是當時的人所買到的配備。

在此情形下，有些無法廢除的關鍵決定就如此定局了，也決定了如何利用這些記憶體的方式。這有點純學術性——有點類似在專賣遊戲或大富翁遊戲中的籌碼——但必須有人去決定它。因此，如圖 1-1所示，640K就如此 "慷慨地" 地被指定了，這對當時的應用軟體來說可能有點荒謬，因爲640K似乎太多了。不論是實際情況所需，或者是專案計劃過的，一大塊的 384K 就保留給系統。當然，他們並不是眞正需要 384K 或其他類似的東西，但是當時連這些天才都無法預估 PC 使用者記憶體的要求，更不用說預知使用者會需求軟碟之類的週邊了。

我們現在非常幸運，在購買電腦時就配備了 640K 的記憶體。然而，我們有了這些記憶體之後，許多工程師（程式設計師）就立刻瓜分這些記憶體，他們瞭解的愈多，市場發展得愈快速，而且賦予了 PC 生命。但是，除了那些 "關鍵性決定" 能在控制範圍之內，其他的就無法控制了，所以也註定了某些無法避免的窘境。

以實用的觀點來說，如果不是一些貪婪的工程師攫取了所有他們不需要用到的記憶體，就不會有擴展記憶體之類的東西出現。因此，就是這關鍵性的 384K 記憶體保留給系統，這些從位址 A000h開始（也就是 640K 之上）的記憶體也就是我們討論的重點之一。

這一段超過 640K 而被積存不用的片段記憶體，也就是 IBM 以及其他有力的公司後來開放給我們存取的部份。這並不只是無用的鉋木屑，如果我們把焦點放在 80386的機器上，我們會發現這些片段是非常有用的，這些片段在經過非常精巧的技術之後，就可以成爲許多 mega 的擴充記憶體。就算有些作法不是非常精巧，但一樣可以正常動作。

很不幸地，經過幾年以後，有許多人注意到了某些高於 A000h 卻還沒用到的位址空間，而且認爲 IBM將不會用到它。因爲有人這麼認爲，他們就作了更深一層的假設，如果 IBM 不用到它，就不會有其他的人注意到這一個部份，於是他們就順理成章地用了這個地方。有些在工業界非常有名的公司，都曾用過這些地方，由於已經根深蒂固地佔用了這些地方，也就很難把它們移到應該屬於它們容身的地方了。因此在往後幾章，我們將處理這些東西存在的事實，並且將說明如何處理這些不甚完美的區域。但是現在只要先看看這些區域到底有什麼東西就可以了，或者假設我們已經到達了系

統內最神祕的部份。

1.3 640K 之外的世界

BONANZA 螢團 (R)

許多讀者對於火星表面的瞭解或許比 640K 以上的部份還要清楚，那也不錯。如果我們都深陷於許多技術的細節內，那麼我們就沒有時間把這些技術應用到我們的工作上了—這就是我們入門的著眼點。然而，當我們把電腦系統推進到超乎原始 PC 設計者想像的地步，並且進一步地觀察目前在理論上所能達到的能力表現時，我們在完美與紛亂之間的距離就會愈來愈小。也就是說，我們對這個部份瞭解得愈多，我們就愈接近峭壁的邊緣。

現在就讓我們看看這些工程師是如何瓜分記憶體這塊大餅。在圖 1-2 當中，我們把焦點聚集到高於 640K的位址範圍內，在此圖內，讀者可見到原始的記憶體是如何指定的。

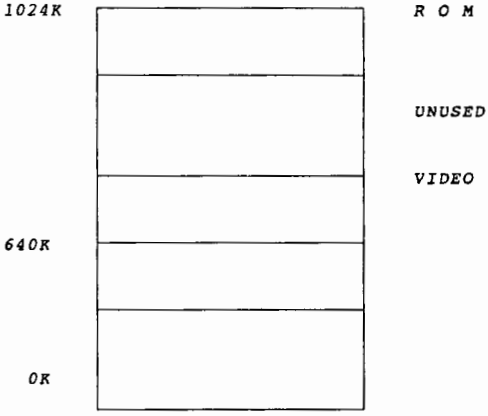


圖 1-2 記憶位址的指定

在圖 1-2當中表示了原始 640K 的部份。在原始被置於一旁供系統使用的 384K 當中，只有頂端的 64K，以及底端的 128K 眞正地指定爲特定用途，頂端 64K是給唯讀記憶體(ROM) 使用，底端的 128K 是給螢幕使用。後者，也就是128K的螢幕記憶體，有許多螢幕只用到 128K 頂端的 32K，而留下了連續的 96K記憶體。這 96K記憶體，連同唯讀記憶體及螢幕記憶體之間的 192K 都浪費掉了。雖然經過多年以後，系統對此區域的運用有所改變，但一樣有許多高於640K的記憶體沒有用到。

從 640K 以上的記憶體開始觀察，首先看到的是從A000h 開始的 128k 是規劃給螢幕記憶體使用。讀者是否對顯示單色文字頁 (monochrome text)眞正用到的記憶體有些概念? 當然，位元對映圖形 (bit mapped graphics)又是另外一回事。但是以目前的顯示器而言，顯示單色文字頁通常只要 4K 即可。這 4K 記憶體是基於 25 行顯示，每行顯示80個字元，而且使用一般的字元產生技術而計算出來的。(註:確實的對映記憶體 (mapping memory) 數量，以及可用的螢幕記憶體隨著不同的顯示卡及不同的製造商而有些微的差異。原始的 IBM單色顯示卡所對映的記憶體爲 B000h到B100h,而彩色顯示卡 CGA則比單色顯示卡多用了 4倍記憶體，且 CGA所對映的記憶體是從 B800h開始的。)

事實上，他們沒有藉口用掉那麼多記憶體，眞正指定給單色顯示卡的位址是從 B000h開始 (704k),有 64k的額外連續記憶體空間誘人地緊鄰在 640k 的使用者空間之後。在實際應用上，現有許多顯示器的位址是從更高的 32k開始，也就是從 B800h開始，比單色顯示卡多留了 32k的連續記憶體未使用。

這個事實並沒有被早期的電腦高手 (hacker, 但是現在hacker有另一種解釋) 所浪費, 特別是當使用者的記憶體需求逐漸盤旋上昇, 並且開始蠶食 640k 這個看不見的障礙時。注意: 在此我們一直不斷地談論位址空間, 而非實際裝設 (install) 的記憶體。通常在那些位址空間並沒有什麼東西, 就像一個新的個體, 只具備了某些空的插槽, 但看不到眞正的擁有者。

這種情形就跟主機板剛出廠時只裝備了一個 bank 的記憶晶片, 但還留下了許多行空插座 (socket) 的情形一樣。以 IBM PC 來說, 在其主機板上總共可支援到 256K 記憶體。因此我們不一定要插上與插座個數相同的晶片, 所以有些記憶體只是具有位址, 但事實上並不存在。然而, 這些位址已經被指定了, 而且是可用, 合法的 (available ,legitimate), 電腦在合理的範圍內會於起動時認得某些裝置(device)的存在, 只要這些裝置能報告他們自己裝置 (installed) 在哪個位址。

即使在當時的 8088 機種—不像現在的 80386電腦, 由於 8088 微處理器本身的限制而無法重新對映 (remap)未用到的位址空間—如果你能把記憶位址的空間設定在高於 A000h 的地方, 那麼把記憶體晶片插入插座, 以獲得更多的使用者空間將是相當簡單而直接的作法。後來有些廠商把這些通稱爲 "擴充板" (expansion board) —— 然而這跟 LIM (Lotus, Intel, Microsoft) 所指定的擴充記憶體特性一點關係也沒有。即使他們提供了最低限度的可選擇性支援—至少有 64K或者在某些情況下有 192K 的 "清道夫" 記憶體 (scavenged memory) 。

因爲在 640K 之上通常沒有連續的 192K區塊 (block), 所以這些奪取 (scavenge) 更多記憶體的方法通常非常粗魯草率, 而且通常要把同位元檢查 (parity check) 抑制, 以達成他們的工作目的。因此 704K 或多或少成爲有缺陷的實際限制。有些相容品製造商 (clone manufacturer) 在 EGA 還未成爲普通的顯示標準時, 眞的就把 704K當成他們機器的標準。

即使在 CGA的問世及流行階段, 仍然沒有造成問題。CGA 這種非常粗糙的裝置, 連同其相當適度的記憶體需求, 就裝設在保留給螢幕的 128K 記憶體的頂端了。

隨著 EGA的上市, 情況就有所改變了。EGA 需要更多的位址空間 (address space)以及記憶體, 這使得 EGA不得不把自己往下拉到從 A000h的位址空間開始定址。這些記憶空間本來就是保留給螢幕顯示用的, 所以 EGA就順理成章地用了這個區域。

非常不幸地, 就如同我們所生存的眞實世界裡, 沒有二個物體能夠在同一個時間內佔有同一個空間, 即使在此例中的" 空間" 只是位址空間。在電腦之內, 位址空間是非常寶貴的, 只要沒有其他的裝置使用了某個位址空間, 一切都沒事; 但是如果有一套軟體以及一套硬體裝置, 或者任何兩個不同的物體試著在同一個時間內使用同一個位址空間, 那麼電腦很有可能當機, 不論是哪一個具有合法的位址使用權, 結果都是一樣的——當機。

由於有些東西只要稍微用點腦筋就可以瞭解, 例如: 瞭解 640K 以上的記憶體在做什麼, 至少在某些工作完成後這個區域會填入那些內容等等。因此, 把這個概念當成第一優

先。讀者還會再一次地遇到這個問題，而且在往後幾章之內，會討論當位址發生抵觸時應該如何解決。

我們已經從歷史的角度對系統保留記憶體底部的三分之一 —— 128K做了一番簡略的介紹。其上及其下也是一樣的道理。有一個稍微矛盾的說法，這些有時可被歸類爲所有權的破壞行動，有時也可解釋爲充分利用其餘 256K 記憶體的動作。然而，其作法在基本上是一樣的，而且就像我們無法從一個人所戴帽子的顏色去分辨此人是好人或壞人一樣。

現在就讓我們好好地研究以下的材料，以便把一些安排不當的區域或片段轉換成某些型式，使其能動作一致。

1.4 入門: 作業系統

BONANZA 壹圓 (R)

(Enter : The Operating System)

從瞭解 DOS的觀點來說，當它與延伸及擴充記憶體之間產生關係時，把我們的作業系統分成幾個基本模組(module)或構成區塊 (building block) 來看是非常重要的。

我們在談到 DOS時，把它視爲作業系統，宛如它是一整套產品，若沒有 DOS的話，我們的電腦只不過是一些搭配不佳的晶片，而且硬體也毫無辦法達成任何有效用的任務。在實際上，DOS 由於這種與機器的搭配關係而逐漸地增加它的名聲，畢竟，在 PC 型式或是其他相容的機器上若無 DOS之類的作業系統就無法運作。

然而從事實的角度來看，DOS 只提供了眞正作業系統的

一部份，爲了證明這個事實，我們只需看看原始的 IBM PC 即可。這些 IBM所出產的機器並不一定需要 DOS才能執行，在打開電源的時候，若機器沒有裝設硬碟或軟碟，則經過幾秒以後，系統即會啓動，螢幕開始動作，鍵盤也開始接受輸入，一點也不需要 DOS或其他外界的作業系統。這些甚至完整地包含了從外界的大量儲存媒體裝置 (mass media device) 當中存入或讀出程式及資料等功能。

大概不會有很多人以此方式使用他們的舊型 PC ，就算有的話，時間也不會太長。然而，重點在於這些 PC 在完成時就已經裝備了完整而獨立的操作系統——此系統能夠設定一些操作的規則，以及管理一些系統的 I/O: 如鍵盤、螢幕，以及大量儲存裝置。甚至還有點誇張地加裝了 ROM BASIC，在起動的時候可以自動從 ROM當中載入。這些獨立的系統稱爲 BIOS (Basic Input/Output Service,基本輸出入服務常式) 。(註: 把 BASIC的核心放在 ROM裡面仍然是 IBM獨有的特性。由 IBM PC-DOS 所提供的 BASIC.COM及 BASICA.COM必須依賴 ROM裡面的 BASIC核心，所以這兩個程式無法在其他的相容機種上執行。由 MS-DOS 所提供的 GWBASIC 就是一個獨立的程式，不需要 ROM裡面的 BASIC核心支援。當然，在 IBM的機器上，GWBASIC 一樣可以執行。

在另一方面，如果 PC 上具有軟式磁碟機，而且有一份 DOS 的話，可以從 DOS來起動電腦。DOS 並不是用來取代內建的作業系統，而是與它相結合。事實上，電腦的起動程序會先執行內部的作業系統，以決定它該執行什麼動作，然後才執行 DOS，由 DOS提供一些額外的功能。

何謂 "額外" ？ 軟式磁碟機即是一個例子。原始的 PC 並不一定需要軟碟——事實上，PC的設計者並不期望大部份

的使用者都必需具備一部昂貴的軟式磁碟機 (以當時來說，軟碟是很昂貴的) 。他們看上的是卡式錄音機，以此當成PC上最受歡迎的大量儲存媒體，這也是他們提供這類輸出入常式——連同一組用在特殊版本的 ROM BASIC上的 MOTOR指令——建立到 PC 上的原因。而這一版本的 BASIC的綽號叫做卡帶 BASIC (CASSETTE BASIC) 也就不足爲奇了。

如果你想用到磁碟機的話，那麼就應該準備一份 DOS (Disk Operation System, 磁碟作業系統) 。

現在先等一會兒。你既然需要 DOS，就必須從磁碟載入DOS,以使用磁碟。這表示所需要的資料及指令，甚至連從磁碟讀入指令的動作都必須放在磁碟上的某一特殊位置，以便使一個功能不甚完整之系統 (dumb system)也能夠逐步處理這些資料。這些資料必須儘量精簡，使其在一小段 (gulp) 的讀取之後便能提供足夠的 I/O指令，使得系統能繼續載入DOS 的其他部份，並且能包含所有與磁碟作業相關的服務常式。這也就是 DOS的靈魂及精髓之所在。

首先讀入的一小段 (gulp) 通常就是所謂的起動記錄 (boot record),而且總是位於軟式磁碟的第 0磁軌 (track) 的第 1個磁區 (sector),若磁碟片是雙面的，則此起動記錄必定位於第 1面。這是必然的現象，如果起動記錄置於磁碟的其他位置，那麼還處於半醒狀態的電腦 (half-awake computer) 就無法找到起動記錄了——即使電腦已經讀到此起動記錄，它一樣無法認得這就是它所要的起動記錄。總而言之，在起動階段的電腦可以說是非常愚蠢的。

只要電腦讀進了第 1個磁軌的第 1磁區，就好像你在早晨喝下了一杯咖啡一樣——系統雖然仍處於惺惺忪忪的半醒

狀態，但早已準備就緒，以迎接更龐大、更艱難的任務了。（註：在硬式磁碟機上，起動記錄位於 DOS劃分部份——DOS partition 的第 1個磁區）。

接下來是 BIOS(基本輸出入系統),如果你瞭解的話，你應該知道電腦內早就有 BIOS 了。這一個燒錄在 ROM裡面的BIOS在出廠時就已經裝好了，如果沒有此 BIOS 的話，電腦甚至連讀入起動記錄的能力都沒有。所以電腦內部有一內建之 BIOS（事實上是插入一個唯讀記憶晶片——ROM CHIP)，但是我們現在要討論的部份有點不一樣。由 DOS所提供的部份，替內建的 BIOS 加入了更多的功能，而且在某些情形之下可能會把內建 BIOS 之內的部份指令改寫，以適應環境的改變。（也許有人會問：ROM 裡面的指令如何改寫？事實上，並不是改寫 ROM的內容，而是改寫某一服務常式，並修改插斷向量表。有興趣的讀者可自行參閱組合語言方面的書籍。）

然而，有一件事情必須牢記在心，而且這件事情會愈來愈重要，這件事就是：DOS 或是其他可視爲 DOS的軟體並無法獨自完成任務。它必須依賴某些構成電腦的個體，並且與他們一起動作——如大多數作業系統的基本部份。除了下列的作業系統之外—— DOS, XENIX, PC-MOS/386, OS/2 等等，都必須與此基礎相結合，並且要包容此基礎，還有，要與此基礎完全相容。

作業系統的基礎部份——此基礎部份必須在起動程序時載入記憶體，它通常包含在某種稱爲 ROM晶片的特殊晶片之內。因爲如此，所以我們稱之爲 ROM BIOS 。以現代的標準來看，若沒有某種型式的內建指令，沒有一種功能式的電腦(functional computer) 能夠起動，更不用說執行特定功能

了。至少，在打開電源時，電腦內部必須包含足夠的資訊以便起動。

然而，上一段所提的 "東西" 到底是什麼呢？其實也很簡單，只不過定義了某些必須出現於螢幕的字元，例如當你按了某些鍵之後，螢幕就會依你的按鍵而做各種反應。若不定義這些事情——若沒有一組基本的指令做爲某些標準，那麼程式就明顯地無法獨立於機器的實際裝置之外，如此一來程式必須根據各種機器組態來設計，也會顯得非常複雜。

接下來的就是 DOS的開頭，比起前述的部份多了一點東西，但一樣有點不夠，而且在事實上是非常不夠。然而它的確提供了磁碟 I/O ——可支援單面 8磁區的磁碟格式，並且提供了比 CP/M 好得多的目錄結構。(CP/M 是當時替極受歡迎的 8080 晶片所設計之作業系統)。

DOS 1.0 版的設計包括檔案屬性管理 (file attribute management)，而且能顯示檔案的大小及其變動的日期 (modification date)。它也提供了一個 AUTOEXEC 的批次檔案 (batch file) 以提供起始化動作 (initialization)。然而，其最主要的功能還是在於提供 ROM BIOS 未包含的磁碟服務常式。

但是有許多和 DOS有關的基本服務常式以及一些被認爲是理所當然的功能在 1.0版的 DOS裡都還沒有。這也就是我們爲什麼常看到此注意事項: " 需要 DOS 2.0或更高版本" 的原因了。現在大部份軟體認爲非常基本的服務常式在當時都還不存在——既不在 ROM BIOS,也不在 DOS 1.0當中。

IBM 是唯一供應 DOS 1.0的廠商 —— 但是在當時也還

沒有其他競爭者的加入，其他的競爭者還未成氣候。然而，即將成爲巨人的嬰兒戰戰競競地跨出了第一步，即使在 1.1 版發行的時候也只多了幾家販賣商。而在此時，平行的 MS-DOS也已經開始問世了。

在 DOS 2.0版就可看到很大的改變，也是首次出現看起來較像樣的作業系統。除了 1.0原有的功能之外，2.0 更提供了 I/O轉向 (redirection)，管線功能 (pipe),過濾器 (filter)，印表機緩衝區 (print spooling),磁碟標記 (volume lable)，擴充的檔案屬性 (expanded attribute)，以及經由 CONFIG.SYS 檔案提供了更强大的組態選項。它也增加了 ANSI 顯示驅動器 (display driver),允許程式動態地控制記憶體 (dynamic control)，以及程式環境區塊維護 (environment block mainteance)，它甚至支援了使用者自定的命令處理器 (command processor)。

DOS 還在日漸茁壯當中，雖然在過去及現在都曾有過瑕疵，但它想呈現其多樣化，以適應不同的環境及不同的使用者，甚至不完全相容的電腦。DOS 已經設法包容許多不相容性，不僅是從使用者方面，也從軟體的相容性著手。雖然 DOS 的速度不是很快，未來也可能如此，但是對許多使用者而言，DOS 提供了最佳的解決辦法。所以，DOS 還能持續一段很長的時間。

1.5 進階: 雙向道

BONANZA 曼圖 (R)

(Evolution: A two way street)

經過多年，就連 DOS所提供的磁碟服務常式也經歷了可

觀的改變。新的磁碟服務常式隨著需要而增加，有時必須支援未經預期而出現的裝置，或者是 ROM BIOS 未支援的裝置。有一個例子就是：一下子就造成流行的 720K 、三吋半磁碟，舊機種的 ROM BIOS 或是 3.1版的 DOS都無法預期其出現。(事實上，三吋半磁碟在 DOS 2.0以上就可充分利用其容量，只要與第三團體 —— third party 的裝置驅動器 (device driver) 合用即可。DOS 3.2 版提供了支援，而不必用到外部的裝置驅動器，但是如果需要特殊 I/O時，必須在 CONFIG.SYS 檔案裡加上 DRIVPARM /X /X 的敘述，以配合每一台特殊的磁碟機。) 3.2 版的 DOS加上了三吋半磁碟的支援，使得舊型的機種不僅能容易地升級支援三吋半磁碟，更能支援舊版本未能支援的其他特性及功能，以避免版本老是跟不上時代。

然而，在 DOS 3.3出版之前，大部份販售商已經把這些支援以及其他必要的服務常式併入他們各種不同的常式內。這種提供權宜之計以解決三吋半高密度磁碟支援問題的重擔就不落在 DOS之上了。所以 DOS中止這些服務常式一點也不令人感到意外。這只不過是冰山的一角，DOS 以及許多相容電腦之間的相互關係正不斷地進展，兩者隨著市場及裝置改變而有增有減。

1.6 那裡還藏有什麼呢?

BONANZA 璽圖 (R)

(What else is in there ?)

到目前爲止，我們見到的 DOS服務常式大部份都是位於 2 個不曾在目錄列表中出現的檔案內。只用 DOS的 DIR指令是無法發現它們的。當然，它們是可定位的 (locatable)，

而且不只是由半醒的電腦逐步讀取。然而，它們是隱藏的，因爲有一個檔案屬性設定於這兩個檔案，所以 DIR無法找到它們。即使你是一個喜好追根究底的人而找到這兩個檔案，DOS 一樣有其他鎖定機制，以防止意外的損害：也就是利用另外一種屬性，把這兩個檔案設成唯讀模式 (read only)。(註：即使 DOS的 DIR指令無法列出隱藏檔，但一樣可利用 TYPE指令把它們叫出來，並查看它們的內容，或者你也可以使用大部份的編輯器或文書處理器來做到，但大前提是：必須知道這些隱藏檔的完整檔名。前述的兩個隱藏檔一般是 IBMBIO.COM及 IBMDOS.COM 。雖然能以 TYPE 之類的方法觀察這兩個檔案，此兩個檔案的所有內容並不能完全地顯示在螢幕上，不過，這也足以證明這兩個隱藏檔眞的存在，而且是可載入的檔案。一般來說，唯讀屬性可避免因干擾或粗心而造成損壞，特別是以一般的編輯器或文書編輯軟體開啓時。然而，這個保護屬性可以很容易地改變 —— 最近幾個版本的 DOS都提供了 ATTRIB 這個公用程式。而且 DEBUG這種能顯示檔案所有內容的工具程式，不論它所做的動作爲何，都不考慮屬性，所以進行追根究底時必須謹慎行事。)

在這兩個平常爲隱藏檔的檔案當中，有一個IBMBIO.COM，這通常稱爲 BIOS 模組 (BIOS moudle)，由它的名字即可猜測出它的主要功能在於提供輔助 I/O服務常式，以填補 ROM BIOS 之不足。(註：在 MS-DOS 當中，與 IBMBIO.COM 及 IMBDOS.COM 等效的檔名爲 IO.SYS 及 MSDOS.SYS，在 MS-DOS 的版本當中出現了一個有趣的特例 ，因爲它用了 .SYS 的延伸檔名，而且 .SYS 及 .COM的延伸檔名在 DOS的平常規則之下是無法交換使用的。然而，不論它們檔名爲何，這兩個檔案如果沒有經過變動的話，其屬性一定是系統 (system) 、隱藏 (hidden),以及唯讀 (read-only)。)

另外一個，通常被稱爲 DOS核心 (DOS kernel),提供了所有使用者現有的應用軟體所必須的軟體介面。IBMDOS.COM 或它的等效檔案 MSDOS.SYS提供了一群與硬體獨立的服務常式，稱爲系統函數 (system functions)。 這包括下列幾項：

[1] 檔案及記錄 (record) 管理。
[2] 記憶體管理 (只包含原始的記憶體 —— conventional memory only)。
[3] 字元裝置的輸出入。
[4] 有時被稱爲 spawning 的功能 (也就是在程式內執行另一個程式)。
[5] 存取即時時鐘 (real time clock)。

應用軟體在執行這些功能函數時，只要設定好函數所規定的暫存器，然後就可轉移給作業系統執行了。

1.7 命令處理器 ——

BONANZA 螢圖 (R)

Command processor:COMMAND.COM

我們與 DOS接觸最多的可能就是命令處理器: command.com 這個檔案，這有時被指爲 shell (殼，就好像 UNIX上 cshell 之類的 shell一樣),但請讀者不要把這個 shell 與 DOS 4.0版才引進的圖形介面 DOS (graphic interface DOS)相混淆，因爲此圖形介面也稱爲 SHELL (字母全部大寫) 。爲了避免混淆，在本書一提到 shell，就是指命令處理器或是 COMMAND.COM。

COMMAND.COM 這個名稱大概可以說明它的作用了。然而它是一個多重的模組 (complex module),它不僅包含了一些內建的命令 (built-in command,internal command), 還可叫用其他的外部命令函數, 包括批次檔 (.BAT) 的處理。

在此時, 是分辨 "internal (內部)"及 "external(外部)"的時機了。 internal 是指在起動程序結束, 且載入命令處理器之後即可直接執行的指令。當我們說到一個原始的系統 (raw system),是指沒有載入其他程式, 也沒有執行AUTOEXEC批次檔的系統, 螢幕上只有我們熟悉的 A> 或 C> 的提示符號出現。

如果系統內沒有其他東西 (如常駐程式之類的),那麼你能執行的就是 COPY,DELETE檔案, 獲取目錄列表 (DIR), 建立次目錄 (MD),變換次目錄 (CD),刪除次目錄 (RD) 等等。剛才所列的指令以及其他指令對 COMMAND.COM來說都是內部的。

是否包括 FORMAT,MODE及 CHKDSK 之類的指令呢? 爲了怕列出的指令表太長, 所以趕快回答 "NO!", 因爲這些都是外部指令, 這表示這些命令會起動 (invoke) 某些獨立的公用程式, 而且這些公用程式必須位於內定的磁碟及目錄之上, 或者在路徑之內 (PATH),如果都不是, 那麼在命令列上可加上指向此命令的路徑, 一般地範例如下:

```
C:> \DOS\FORMAT
```

你能分辨出有何不同嗎? 只要查看你的 DOS目錄, 或者是查看你存放 DOS的磁片 —— 但此磁片的內容必須是源自

DOS 的 distribution disk (散佈磁片) 。此磁片中或此目錄中的檔案如: FORMAT.COM; MODE.COM; CHKDSK.COM, 任何在此 DOS目錄下的名稱都很像一個合法的 DOS命令, 但已經加上了 .COM 或 .EXE 的延伸檔名, 以表示其爲可執行檔。事實上, 在提示符號下打入這類命令, 在經由 COMMAND.COM 處理之後, 就會執行相對應的檔案。事實上, 你可以毫無顧忌地從你的硬碟之內刪除大部份位於 DOS目錄之下的檔案, 這樣也不會造成什麼大問題, 只要你確定你不執行這些檔案。然而, 如果你未來需要改變系統設定, 最好把這些檔案放在 distribution disk當中, 好好保存。

把這些再加入 COMMAND.COM的函數控制以及批次檔的執行, 你就能一窺 DOS全貌。至少到目前爲止我們都已經完成進入下一章的準備動作了。

還有一件重要的事: DOS 實際上只不過是一些有用的公用程式之集合。即使有新的擴充記憶體裝置驅動程式 (expanded memory device driver),如 386EMM.SYS (或者是 IBM所提供的 XMA2EMS.SYS以及 386系統專用的 XMAEM.SYS)是從 DOS 4.0開始提供的。它們都只是公用程式 (utility):並非神聖而不可侵犯, 也不一定是最好的, 因爲在 DOS之下還有很多同類的程式以完成同樣的功能。(註 :其實連 COMMAND.COM本身也不是不可或缺的。不過對大部份的 PC 使用者來說, 此現象並不是以熟悉的型式出現。Hewlett-Packard 的 MS-DOS 電腦在剛開始的時候, 是搭配其公司所專有之螢幕導向殼層 (screen-oriented shell)出售, 此 shell稱爲 Personal Applications Manager。它的功能很像 DOS提供的選擇性圖型使用者介面 (GUI —— graphic user interface) 或是 "Presentation Manager", (畫面管理程式)DOS是從 4.0開始才提供 PM 。另外, 剛剛

所提到 HP 專有的版本，是爲了他們公司生產的Touch Screen HP-150 所特別設計，但是在HP或其他的 DOS 機器上，一樣可以執行。）我們將於往後幾章討論更多的公用程式或驅動程式。

第二章 電腦的心臟

嚴格來講，電腦只有兩個主要的部分：

1. 微處理器 (microprocessor)—— 實際執行所有的動作的地方。
2. 記憶體 (memory)—— 包括一般記憶體、延伸(extended)記憶體，及擴充 (expanded) 記憶體。

不管電腦的結構有多麼複雜，如果我們將焦點集中在微處理器與記憶體這兩個部分，那麼，許多複雜的元件都會比較容易了解。微處理器之所以重要，乃是因爲它控制了在記憶體中執行的一些規則，以及記憶體的定址 (address)方式。本章就先對微處理器加以討論。

基本上，本書主要在探討三個不同，但是卻有極度關聯的微處理器 (chip): 1) 8088, 2) 80286,及 3) 80386。80286 及 80386是往後發展的核心。但是，就像飛機無法淘汰掉汽車一樣，新近的電腦工業只能開展新的領域，而不能取代舊有的東西。

在軟體及特定的輔助硬體 —— 特別是記憶體設備，均有重大的革新。即，目前技術的突破，使得後來發展的微處理器，均能與較早的微處理器相容，也就是說：原來在舊有機器上能處理的，在新的機器上仍然能夠處理。但是，有些應用只能在 286及 386上處理，也有些應用只適用於 386。這點是應該加以注意的。

與早期的微處理相容，這個曾被許多人奉爲圭臬的性質，已經不再被看得那麼重要，因爲技術發展的腳步不能爲了那些早期原始的產品而放慢下來。不過，在許多方面我們仍免不了執著於這種傳統的想法。因此，我們還是要先來看看它們彼此間的差異與關係。

2.1 8088

8088的發展可追溯到 1972 年的 8008 。8008是 Intel 公司所發售的第一個八位元微處理機。在這裡我們並不打算對 8088 或 8086 多做介紹。但是，爲了要清楚地了解位元(bits)的觀念，我們還是要介紹更早一代的 Intel 4004 。就如同它的名字所暗示的，4004是一個四位元的微處理機。

在二進位系統 (binary system)中，要能表示 0到 9，至少需要四個位元，而四位元的重要性也就在此，我們簡單的解釋一下：因爲一個位元可以數到 2，兩個位元可以數到 2 的兩倍，也就是 4，三個位元只能數到 8。因此，我們必須要有四位元才能數到 16(2 的 4次方等於 16)，也才能數到10 。

假如你想做的只是做些簡單的數字運算，那麼四位元便

十分足夠了。例如，4004多半用在簡單的計算機裡，它能做加減乘除的運算。跟辦公室中一些傳統機械式的事務機比起來，4004還算得上是科技的奇蹟。

接下來的問題，就是要如何表示 "文字" 了。四位元只能表示出 16 個 (0123456789ABCDEF),無法表示完整的英文字母及其他符號。正因爲二進位系統中每增加一位元其範圍就成爲兩倍這個簡單的事實，八位元微處理機便應運而生了。

八位元表示可以有 256種組合，這已經超過了表示 10 個數字、所有大小寫英文字母，以及一些標點符號及特殊符號 (如希臘字母及數學符號) 所需要的空間。

因此，Intel 公司發展了 8008,它與 4004 比起來只不過多了幾個位元罷了。8008不久後便被 8080 所取代。但是，這些發展仍然開啓了一個新紀元。在此之前，我們只有四位元可用，所能做的就是簡單的加減乘除，不需要眞的程式設計。所有的工作都即時 (real time)完成，沒有任何輔助記憶體。

但早期的八位元微處理機仍面臨了一些問題。如同今日的大型電腦，某些程式仍是需要的，但當時並沒有一種可用的程式語言 —— 即使已經有了一些 "程式",也沒有什麼比較簡單的方法來儲存或取用。

有一群認爲 8080 難以發展下去的工程師便離開了，他們設計了另一種微處理機 — Z80 。Z80 仍是一個八位元的微處理機，它使用十六位元位址，故可以定址 64K的記憶體。Z80 最重要的貢獻，在於其促進了第一套專爲微處理機設

計的簡易作業系統的發展。這套系統就是 CP/M (Control Program for Microcomputers)。

在此同時，Intel 繼續改良 8080,且發展新的衍生產品以開拓市場。他們設計出了十六位元的 8086,然後又將之改成八位元的架構而成爲 8088 。Intel 這樣做是基於對作業系統的需要，因爲當時沒有十六位元的作業系統。在往後退了一步之後，8088在設計上使十分接近早期的 8008(其實更接近它的對手 Z80),如此一來，便可使許多在 CP/M 上設計的程式可很容易地轉換到 8088 上執行。

就在這個市場一片混亂的時候，IBM 開始設計第一部 PC (IBM Personal Computer)。如此一來，大家似乎開始由混亂中趨於一致，然後又再度分散。事實上，當時 IBM曾把 CP/M-86 與 DOS皆列爲早期 PC 的作業系統，由用戶自行選擇。

接下來我們就來看看 DOS。DOS 事實上是由 CP/M 的一個衍生版本發展而來。這個衍生版是由一家叫 Seattel Computer的公司設計的，後來由 Tim Paterson 及當時剛成立不久的 Microsoft取得版權後，繼續開發成 DOS。請注意一件事，這個時候的 PC 仍像個玩具，而且沒有清楚的發展方向與目標。

爲了與 CP/M 維持一個相當程度的相容性，Intel 把 8088上 1 megabyte 的定址空間分解成一些 64K的區段 (Segments)，這是爲了要配合 Z80及原來 CP/M 所容許的最大記憶空間。事實上，早期 PC 的標準配備只有 16K，若不加擴充卡，主機板上也只能擴充到 64K。

DOS 起初與 CP/M-86同時銷售，但只賣大約一半的價錢。此外，DOS 有較好用且較合邏輯的使用者界面與命令語法，因此，很快地將 CP/M-86趕出了 IBM PC 的市場。但是，DOS 仍深受 CP/M 的影響，至今仍是 64K區段，且還是保留了一些 CP/M 的特質。

PC發展出來之後，即使是最 "原始" 的版本，都已遠遠超出設計者的期待。但 Intel公司仍繼續設計更新且更好的晶片，現在新的兩代機器 —— 80286,80386 早已技術成熟，而最新的 80486機型，也開始在市場上銷售。不過，技術雖快速轉變，卻還有一些事情依舊不改。我們的作業系統仍是以 64K區段組成 1 mega 的記憶空間，不論我們的機器是早期的 PC,或是其它更快更好的後續機種。

這些限制妨礙了 80286，特別是 80386功能的發揮，但8088本身仍有能力做類似多工處理 (multitasking) 的功能。現在使用者已能經由一些如 Software Carousel的程式切換 (Context-Swtiching)工具，或類似 Microsoft Windows 及 Quarterdeck Desqview 等視窗環境 (windowing enviroments) 來做多工處理。

8088也可以使用擴充 (expanded) 或延伸 (extended) 記憶體，但其限制就比 286和386 多得多。8088的架構並不提供如保護模式 (protected mode) 之類的功能。這對一個一般的使用者來說並不是什麼問題，大多數需要用到延伸記憶體來執行他的應用軟體的使用者，大概都會買功能較强的 286 或386 機型。

對大部分的應用而言，8088最大的限制在於它仍是一個八位元的微處理機。有些人稱之爲十六位元，但它的確不是

。8088有兩個八位元處理機一同工作，故能一次處理十六位元資料。在設計上，它與外界溝通時是以八位元爲一單位的，因此，PC的標準開放架構匯流排 (standard open architecture bus) 及其它同類產品與衍生機型都是八位元匯流排，如此一來，所有要與此機器完全相容的作業系統或軟體，都必須設計成八位元的模式。

由於 1 mega DOS 及八位元作業系統與軟體的限制太大，因而促使了一套能配合 80286十六位元工作能力的新作業系統的開發，這就是 OS/2 的起源。OS/2也不再侷限於 DOS 1 megabyte的線性定址空間。在此同時，32位元的 80386已發展完成。但是我們還是有一些八位元作業系統及大量發展成熟的八位元軟體可用。

因此，8088在很長一段時間內仍不會被淘汰，即使在一些需要最新技術來應付他們複雜工作的地方，仍有不少 "老式" 的 PC 做著如文書處理之類的簡單工作。以今天的軟體技術，它們也可以做多工處理。到了眞的不敷使用時，仍有加速卡 (accelerator board)升級套件可供選用，此部分本書將做介紹。

2.2 80286

80286 的出現使得一些覺得 PC 與 8088 無法滿足他們的需求的使用者重新燃起了希望。它的出現也帶來了一些新名詞，例如 "延伸記憶體" (extended memory)，"眞實" (real) 及 "保護" (protected) 模式。事實上，80286 更像是多了另一種記憶體磁碟 (RAM disk) 的快速 PC。

80286 可以定址到 16 megabytes,這和 8088 比起來是大得多。80286 也提供了所謂的 "保護模式" 用來確保資料的安全。這種保護模式，及一些用在 80286上的概念，都可以追溯到 1960 年代中期，一個由奇異電子 (General Electric)，MIT，及貝爾實驗室 (Bell Labs)所領導的聯合計畫。

不幸的是，當初 DOS在設計時，似乎認爲 8088 將永遠是桌上電腦科技的主流，因而沒有提供保護模式或 1 megabytes 以後的記憶體。

因此，當 DOS還是唯一的作業系統時，便存在著一個問題：雖然有一些軟體發展者想要很小心的去使用這塊延伸記憶體，但仍充滿著困難與危險。結果就是限制了用途，以及不少書籍所散佈的一些錯誤觀念。

這些誤解之一，就是無論有多大的延伸記憶體，我們都無法執行一個以上的程式，因爲第二個程式很可能蓋過第一個，而破壞了第一個程式的資料等等。例如，我們不能同時使用 VDISK及 disk cache,或其它如 AutoCad 等較早使用延伸記憶體以解決記憶空間不足問題的軟體。或許讀者會問，80286 不是提供了保護模式來防止這些問題嗎？不錯，但如果沒有作業系統提供協助，我們仍然不能用保護模式，我們只能說 286有這個 "能力" 而已。

目前 DOS仍是我們主要的作業系統，但是 DOS延伸程式 (DOS Extenders) 的開發已經使 286上延伸記憶體難以使用的問題解決了一半。基本上，DOS 延伸程式就是補足一般 DOS 所沒有的，它提供了支援延伸記憶體的軟體管理系統。

現在已經出現了一個 "工業標準" 來規範，如 DOS Extenders 及其他一些用到延伸記憶體的軟體。這個叫做 " 虛擬控制程式界面" (Virtual Control Program Interface, VCPI)的 "標準",目前還沒有被所有的設計師或設計部門所接受，但至少是一個好的開始。它訂了一些準則，若控制程式遵守這些準則，便可以在延伸記憶體中同時並存且不互相衝突。

在做了這麼多努力之後，仍無法完全解決在 80286上使用延伸記憶體所遭遇的所有問題。這是因爲 80286本身的一些問題。

基本上，80286 晶片的問題在於我們可以很容易地由眞實模式進入保護模式，卻沒有一個簡單的方法可以由保護模式回到眞實模式。一個程式可以因爲由足夠的資料而一直在保護模式中執行，但當它要存取磁碟，檔案管理，或其他 DOS 服務時，它仍須回到眞實模式。請記住，DOS 只能用到 1 megabyte (FFFFH)，當你必須超出這個範圍時，就必須離開 DOS，直到再次需要 DOS服務爲止。

有幾個方法可以使 80286 (或 80386)由保護模式回到實際模式，還有一種特殊的裝置必須在模式切換時小心處理的。這種叫做 A20 閘 (gate) 的裝置，除非將之關掉，否則它會把超過 1024K 的位址參用，都轉回到位址空間的底部，也就是會把第 20 個位元以上的位元都清爲 0，不允許用到延伸記憶體的範圍。

很明顯的，會有些程式設計師想要回到 DOS時，會故意使用超過 1024K的位址來達到目的。若不用命令把 A20閘關掉，A20 閘便會完成這個 "技巧" 。因此，如果眞要用到延

伸記憶體的的範圍，就必須關掉 A20閘，要回到實際模式時，再將之打開。

假如所有相關指令都能小心使用，一個由保護模式回到實際模式的技巧，就是善加利用 80286上一些由以前的高手們所發現，而文件上沒有記載的特性。如 LOADALL功能，這似乎是製造商爲了內部測試或品質管制的需要而加上去的，但也確實是一個簡單且直接的方法。問題在於 Intel從沒有把它記錄在文件上，也不保証以後的晶片仍然會有這個特性。因此，雖然它的確能用來從保護模式回到實際模式，但對想用這個功能的程式師來說，仍要非常注意其潛在的危險。

還有一個更普遍，更能接受，且只用到 80286文獻所提到的特徵的方法 —— 三次錯誤 (triple fault) 詳細的討論已超出本書範圍，我們只做簡單的介紹:

當一個在保護模式中執行的程式需要回到實際模式時，它只要做一件 "一定不合法" 的事 — 把一個一定被認爲是個錯誤的數值放入暫存器中。系統發現這個 "錯誤" 之後，便開始尋找一些指令來處理這個 "錯誤",然後找出另一個會在這一層中造成第二個 "錯誤" 的數值，這第二個錯誤使 80286 的系統進入第三層也是最後一層的錯誤保護中，接著它會發現又是一個不合法的傳回值，然後你就被 "三振出局" 了。

第三次錯誤之後，80286 便準備重新啓動以清除所有的錯誤，當然，原來正在做的事也會遺失。接下來它會檢查一些暫存器，卻發現一些位元資料的樣式指出它並不是剛剛開機，所以它不做重新啓動。在重設一些特定暫存器的值後，便回到 DOS。

是否覺得有些不可思議呢？但這的確是最可靠的，也是大多數 286程式設計師所用的方法。不過，通往延伸記憶體的大門已經被 80386敞開，就比較沒有人去想這個問題了。

80286 有一些嚴重的缺點，在 80386出現之後變得更爲明顯。有些人就因爲 80286缺乏某些能力，而一直稱其爲 " 傷腦筋 "的晶片。

不過，286 機器仍然被大量使用，且用戶仍在增加中。許多使用者認爲現在（甚至永遠）不需要用到 80386所提供的功能，也就不願意多花錢買 386機器。事實上，買部 386機器來做家庭預算或打學校的期末報告，就像買一輛高性能跑車卻只用來開到街角的超級市場買東西一樣的不切實際。

最近這些缺點已經有了不少解決的方法，這些方法大概都需要添加一些硬體設備以提供原來 286晶片所缺少的功能。就像 8088 一樣，286 也有加速卡升級套件可供選用，我們將在後面的章節裡討論。

2.3 性能超强的 80386

BONANZA 螢圖 (R)

在 80486機型尚未大量出現前，80386 機器仍然是主角。與別的單項發展比起來，80386 使得桌上型電腦有了革命性的進展，且很可能因而影響我們的生活。但直到目前爲止，80386 的能力只被發揮了一小部分，就如同一個剛睡醒的巨人一般。

有不少人將 386機型視爲老 AT 的新版本，然後貼上一

個較高的價碼。也就因爲這種 AT 的印象，使得 386機型一直沒有一個屬於自己的歸類，大多數的將其歸爲 AT類，並當做是 PC 家族的一員。

如果一定要因近似的程度而將之歸爲 PC 類或 AT 類，後者是比較接近一些；不過，這的確是部完全不同的機器，本書也將堅持這種看法。事實上，在本書中大多數的部分都避免使用 "PC" 或 "AT" 這兩個字眼，而完全依照所使用的微處理機來將機器分類。但我們要提醒讀者，即使這樣分類，仍是一個不安全的做法，因爲所有的 386機器並不是用相同的方式設計生產出來的。因此某些在此描述的特徵，在另外一些機器上就可能沒有。不過，這至少是個開始。

80386 有不少其祖先們所沒有的特徵：第一、有更多的位址線，使其定址能力可以達到 4 gigabytes。第二、通常較高的時鐘速度 (clock speed)。這兩個大概是多數人眼中，80386 比較重要的特性。但對大部分的使用者來說，這兩個特性似乎都抵不過 386機型比 286高出的價錢。

這兩個特性對 386整體的性能有一定的貢獻，也是 386最爲人知之處，但的確不是 80386最與眾不同之處。80386微處理機最特別的在於其有三種不同的操作模式。

在實際模式中，386 機器就確實只是一部較大、較可愛的 PC。386機器多半看來很壯觀，放在桌上會人印象深刻，而 386機器執行的速度也常令人驚歎不已。但除此之外，它們的確只是大 PC,其三十二位元處理能力完全浪費掉了。這就是在實際模式的情況。

換到保護模式，差別就比較明顯。與 286不同的，386

可以隨時由保護模式回到實際模式不用如 286一般使用一些令人困惑的技巧。事實上，一些 OS/2 的版本已有效利用這個特性，使得原來在 MS-DOS 實際模式下的程式，可以和在OS/2保護模式下執行的應用軟體同時執行。

即使不用 OS/2 或其他尚未上市的新作業系統，80386的保護模式依然可以工作。這表示 DOS的程式可以載入延伸記憶體並在其中執行；除了可以充分利用這塊延伸記憶體外，三十二位元處理能力也能完全發揮。這如果和一些能在十六位元環境下可以做相同工作的產品比較起來，其效率大約在五倍以上。

DOS Extenders 的技術的確被應用在一些以十六位元 80286 系統爲主要目標的軟體上。但是，我們可以說，目前許多用在 286系統上的 DOS Extenders技術，都是擷取在 386 系統上發展的技術，而非專爲 286開發的。事實上，80386 已被公認是較具吸引力的發展環境，一些暢銷的套裝軟體也都發展專爲三十二位元系統設計的版本。還有越來越多的軟體專爲三十二位元環境開發，有些甚至已不考慮 286的市場。

到目前爲止仍只是 386的一部分而已。386 還有一個從前的晶片所沒有的操作模式，叫做虛擬86(virtual 86,V86)模式，這個模式使 80386具有以前的晶片所沒有的能力。在虛擬86模式下，80386 晶片可以模擬無限數目個獨立的8086系統，這些 8086 系統每個都可以擁有 640K 記憶體，並可以一起執行。

這個獨立虛擬機器模擬做得十分徹底，事實上，使用者甚至可以在不同的虛擬機器上使用不同的作業系統。當然，

這些作業系統都必須支援虛擬模式下的操作，且和 8086 微處理機機家族相容。

這就是多工處理 (multitasking) —眞正的多工處理，允許多個程式 "眞的" 執行，而不是載入記憶體後，就開始坐冷板凳，程式碼是眞的 "共同" (concurrently)執行。(我們很容易犯下說是 "同時" (simultaneously)執行的錯誤，我們不久後就會看到) 但不論是否眞的爲 "同時",80386 仍爲多工處理開了一扇門，這在以前幾乎是不切實際且不可能的。

不過，多工處理還有一些資料保護以外的問題。一部虛擬機器少了一些很重要，而我們看不到的東西，我們只能評估其性能而已。例如，所有的虛擬機器必須共用一個時鐘，但在每個特定時間，只有一部虛擬機器能得到時鐘。

2.4 分段時鐘

時間切割 (time slicing) 的意思是把一時間長度依電腦時鐘切割成數段 (ticks) — 例如 A分兩段 (ticks)，B分一段等等。在本例中，在兩段時鐘內由 A取得微處理機的使用，但在接下來一段內，A 就必須等 B，如此輪流下去。

本例中，我們把時鐘依 2:1 的比例切割，因此，A 機器的有效時鐘速度只有眞實時鐘速度的三分之二，而 B機器只有三分之一。對一部 16 MHz 的 386 系統來說，B機器的速度只比 5MHz 多一點。但別忘了，早期 8088 PC的執行速度只有 4.77MHz ,每個人都嫌太慢了。

如果把時鐘分給更多虛擬機器，情況就變得更糟，速度減慢得很快。至於如何分配時段，可以依自己的意思。要分多或分少，視個別的需要而訂。

假設你分到了很大的一個時鐘段準備盡情使用，卻有一個報告必須在五點前印出來。把這件事丟到後面去，讓它在"背景"(background)中靜靜地執行到結束爲止。在此同時你仍有相當大的時鐘段可以在"前景"(foreground)中做別的事 — 即使可能還有另一件事仍在背景中執行。

但是，時鐘段仍是像記憶體或終端機數一樣的有限資源。除非用到了多工處理，否則可能永遠不會注意到這點。你或許會說，"這沒什麼關係，只是每件事都跑得慢一點罷了，我想我會習慣的。"能這樣想最好，事實上也非這樣不可。有些軟體你可能不願意給太多的時鐘段，但其確實要更多一些否則難以正常工作，例如在背景中執行的高速資料通訊。

雖然大部分新的 386機器時鐘速度都相當驚人，但請記住電腦的時鐘段仍是有限的資源，只允許一個程式有足夠的時間去做一定量的工作。如果從來不用多工處理，這似乎無關緊要，一旦用到了，就成了一個很大的問題。

還好現在已經有了一些處理這個問題的方法，雖然還不能完全解決，但至少減輕了一些。方法之一，背景程式可以被"選出"(polled)。例如，一個程式若沒有執行程式碼或正在做輸入輸出的動作，則這個程式分配到的時間可以被縮短以分給其他程式。在某些情況下，甚至可能略過一群停滯(idle)了一段時間的程式 (包括前景程式),而集中所有的時間在一個需要時間的工作上。例如，一個在前景的文書處理

程式等待一個鍵按下等了半個小時，此時另一個資料庫系統就可以利用這些時間做排序這種需要大量時間的工作。

無論如何切割，時間仍是有限資源，不管用什麼方法，總要設法加以妥善運用。其實，只要有一點幫助，我們還是可以把多工處理做得很好。

2.5 其他的問題

BONANZA 堂國 (R)

目前的電腦系統當然還圍繞著一些硬體上的問題，我們已試著把我們討論的範圍限定在一些特別重要的問題上，特別是關於記憶體方面的問題。事實上，這些關於記憶體與記憶體運用的問題只有在考慮到記憶體用法與管理時才較爲顯著，而這正是本書的主題，將在後續幾章討論。

第三章　新一代的記憶體

不論用的是何種微處理機晶片，電腦看起來都很像，今年的電腦與去年甚至前幾年的機器看來也沒什麼差別，裡面總是有些記憶晶片插在某個地方，而我們只要知道這些記憶體共有多少 K，我們就算已經知道了關於記憶體我們所需要知道的一切了，對嗎?

在以前這樣或許眞的夠了，但現在這的確不是個正確的觀念。至少關於延伸與擴充記憶體的技術就不是如此。而本書的主要目標也在於 DOS屏障 "之外" 所發生的事。不論你使用何種記憶體或以何種方式存取，其餘的你的硬體應該提供一些基礎來建構。

但有些基礎似乎不是十分穩固。我們連考慮該用 80 或 200 奈秒 (nanosecond) 記憶晶片的機會都沒有，因爲當我們正準備開始思考這個問題時，我們會發現，不論是那一種系統，裡面的記憶晶片都是正確的規格。

記憶體也不是只有容量多大或速度多快而已，特別是與

擴充記憶體的使用扯上關係時。在很多情況下一些其他的因素嚴重限制了使用者對擴充記憶體的使用。事實上，根據一項估計，今天市場上大約有百分之九十的機器，對記憶體管理的程度都做了或多或少的一些強迫性限制，且無論再使用何種硬體或軟體，都無法去除這些限制。換句話說，現在銷售中大多數的擴充記憶卡，雖然廣告說完全可以，實際上只能支援 LIM 4.0 EMS現格要求的一部分。

在 PC 當道時，並沒有記憶體管理這種問題。對記憶體只有 "有" 和 "沒有",其結果就很簡單的如圖 3-1所表示的，傳統上記憶體與中央處理單元 (Central Processing Unit, CPU)之間的關係。

基本上過去的機器都是這樣做的，但時至今日仍有不少的電腦製造商這樣做。這個方法簡單、直接，而且容易了解：CPU 擁有主機板上所有記憶體 (或至少全部記憶體中的256K),且要用時隨時可以使用。

在圖 3-1中我們看到了相同大小的記憶體區塊 (block)。這是一群 "自由代理人" 。而在圖 3-2，我們看到整個記憶體都是 "自由代理人" 。經由圖中的方塊以某種方式聯結在一起，這個方塊我們叫做 "管理器" (manager) 。基本上我們已經有了 CPU和記憶體，我們也有兩種不同的方式把兩個東西結合起來。

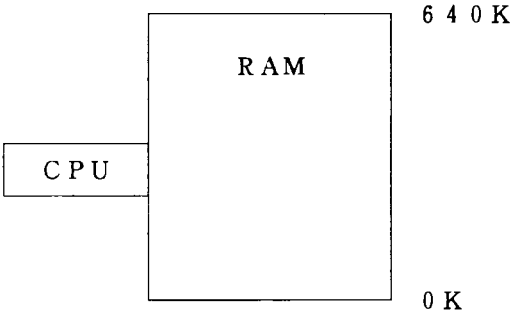


圖 3-1　固定記憶體 (fixed memory) 與 CPU的關係。就如同我們原先所了解的，640K以下的 RAM (在某些 EGA以前的系統有 704K 到 736K)直接與 CPU接線。

請注意與第一個圖不同之處，後兩圖並沒有將位址指派給任何記憶體。我們稱之爲 "自由代理" (free agent)系統，假設 CPU準備對某特定位址工作 (通常是 CPU "被指定")，這個要求便會送給管理器，管理器便在其記錄中查看，該位址是否爲可用 (in use),若是，則將代表那些位址的記憶體送到發出要求之處。

這些聽來似乎有些混亂且可能只適用於某些範圍，但其實非常精確，建構良好，且依循著一些不在本書討論範圍內的規則。我們只要記住這個概念：每個記憶晶片都不需要佔著某個特別的實際位置以回應其位址叫用。換句話說，一記憶晶片不用對某個今天的工作中與昨天工作中相同的位址有所回應。

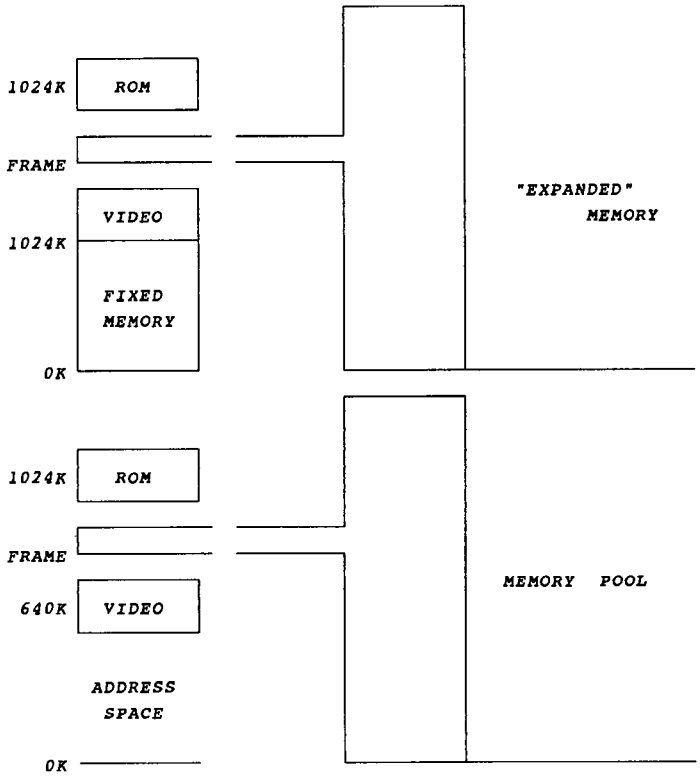


圖 3-2 自由代理記憶體 (Free agent memory)。上圖表示在 EEMS/LIM4.0 之前的傳統 EMS。系統，同時有擴充記憶體及固定傳統記憶體，也就是最早LIM 的規格。下則沒有固定記憶體， RAM 可做爲傳統記憶體及擴充記憶體，由一個共同的記憶體資源庫提供。這個方式爲多工處理及許多 4.0 EMS。 增加了不少彈性。

事實上，今天有一些與 DOS相容的新機種，其主機板上沒有使用者隨機存取記憶體 —— 一點也沒有，例如 AST Premium/286 。其所有的記憶體都設在一個或更多的插入式

(plug-in) 記憶卡上，這表示假如你沒有至少一塊插了一些記憶體的記憶卡，你連啓動都沒有辦法，因爲連個載入 DOS 或其它作業系統的地方都沒有。當然不會有人賣你動都不能動的機器，所以不用對 AST最初只在機器上裝一塊記憶卡 (最小 512K) 便出售而感到驚訝。但這的確証明了一些觀點。

AST 及其新機種當然是極端了一些，其設計師也希望你能相信他們的做法。這的確很極端，但若讀者已仔細看了上述觀點，了解記憶體實際配置在何處並不是件重要的事，也就不會奇怪。我們只管存取與使用就夠了。但若要使用到需要完全支援 LIM 4.0 EMS規格的範圍，我們就需要更多超過多數業者所允許的自由存取。

這也就是我們要開始做些區別的地方，因爲這些在貨架上看來都很光滑的盒子並不是以相同的方式製造的。區分的方式，不是用處理機晶片區分 286,386，也不是用時鐘速度，或其它大部分業務員所知道的方式。關鍵是 "記憶體管理" (memory manage) 。

雖然記憶體管理才是關鍵，但很令人驚訝的是，多數製造商在接受這個觀念及將之加入其設計電腦技術上的速度仍十分緩慢。因此，我們可以發現許多機器的使用者可能永遠無法了解這項技術的潛力，一些使用者可能因此而浪費了幾個 megabytes的記憶體。

PS/2 model 50 及 60 就屬於很浪費的這類。當我們想自 640K 開始向下回填 (backfilling)以獲得完整的 LIM 4.0 EMS 的能力時，我們會因系統的標準而浪費掉一整個 megabyte 的記憶體。這在多工處理中是很重要的，以類

似下述的方式進行:

首先整個傳統記憶體的內容必須拷貝到擴充記憶體中。使用軟體開關 (至少 PS/2 model 50及 60 在記憶體部分是以軟體控制的, 而非 PC 及其他早期機型所使用的 DIP開關) 使主機板記憶體禁能 (disable)。我們必須將之關掉, 因爲系統不允許將其指派給擴充記憶體庫, 且我們也不能讓兩塊實際記憶體 (physical memory)對同一個位址皆有反應。

主機板記憶體禁能之後, 那塊含有拷貝資料的擴充記憶體就被換回以取代主機板記憶體。從此我們就擁有完整的 LIM 4.0 EMS 功能 — 包括以回填方式清除到 0h.不過, 建在主機板上的所有記憶體就浪費掉了。

有很多從記憶體中間開始提供的不同程度記憶體管理, 通常不是由 0h 開始, 而在 256K 附近。由此至 640K, 我們可以將傳統記憶體換出以置入擴充記憶體。這使很多使用者推出一個錯誤的結論: 換出傳統記憶體及回填的潛力限制在 384K 的區塊, 很多著名的電腦著作也支持這種看法, 但這是錯的。這沒有什麼理由, 而只是一個簡單的事實, 大概不到百分之十之機器允許被禁能的記憶體在低於 256K 的部分重作對映 (mapping)。因此, 這只是一個基於統計數字的傳統, 沒有別的。

其實我們可能會想, 由 640K 開始往下做, 而不是由任意一點開始往上做, 只因是 "齊頭平等"(雖然不完全是)。在最初的 Compaq Deskpro 上, 128K以上的記憶體可被禁能, 以前 8088 PC也允許 64K以上的記憶體被禁能。Epson Equity plus 允許任意所有建在主機板上的記憶體以開關向下功能至 (包括) 0h。IBM PS/2 家族的所有成員可以軟體

開關將主機板記憶體禁能。

請注意 "禁能" 這個詞，就如同前面所述，禁能通常是指 "停工" (locked out)。當然，這不一定就表示我們一定要在主機板上插滿 640K 的記憶體 (假設有足夠的插座) 只爲了要能將其關掉且浪費掉。在這種情況下我們通常只需要在主機板上插很少的記憶體 (甚至一點也沒有),其餘的部分就留給延伸或擴充記憶體的硬體與軟體來填補剩下可觀的空隙。

AST 似乎總是能將技術帶領到某些新的方向。在其一些較先進的產品中，具有對映記憶體向下清除至 0h 的能力，且若其中沒有別的，則將可供對映的區域顯示出來。至少以目前技術的情況而言，這反映出的是一種 "抽象" 的狀況而非眞實可行的東西。在眞實世界中，還有些如怎樣指派位址給埠 (ports)及 DOS本身要處理些什麼事等現實的考慮。

在本書中，我們將把記憶體視爲與 CPU完全脫離的整體。記憶體非 CPU所有，而屬於一個 "中間人" — 某種記憶體管理器。這是一個軟體與硬體的組合，通常是一個可以蓋過 ROM BIOS 輸入輸出常式的可裝設週邊設備。此時記憶體管理器假設已控制了部分或全部的記憶體。在記憶體管理器的控制下，CPU 只能看到記憶體管理器要它看到的地方。

3.1 管理器的功能

BONANZA 曼圖 (R)

無論一些內部或外部的設備是否取得全部或部分系統的記憶體資源，CPU 與作業系統 (DOS 或其它) 及應用軟體都必須完全 "忘記" 記憶體管理器以及所有在記憶體上玩的把

戲的存在。

如果我們面對的是延伸記憶體，則我們用的是 DOS延伸程式 (DOS Extender) 這個辭，但不論用的是什麼名字，其功能都是在管理 DOS所無法管理的記憶體。且無論是那一種，對我們與我們的軟體而言，這些記憶體的管理工具應該都是不覺其存在的。事實上，就做一個使用者而言，我們需要知道其存在及其所扮演的角色，以確定我們所買的硬體及其支援的軟體有足夠的能力應付我們現在及以後的需要。而在PC仍是主角的時候，這些管理的工作都必須自己做，也就不必考慮這個問題。

爲了要了解記憶體管理器如何完成其工作，我們必須使自己完全脫離以晶片大小來看記憶體的方法，例如 64K，256K，1Mb 等等，但請忘了這些數字。一萬六千位元組 — 16K，這是一個有趣的數字。記憶晶片的大小可以隨你喜好，但 16K是本書討論的內容中，記憶體最大 — 在大部分的情況下也是最小的區塊 (block) 或單位。

一萬六千位元組：一 64K SIMM 或晶片組有四塊這樣大小的記憶體，1 megabyte SIMM 或晶片組則有 64 個 16K。好了，我們已能處理了，是嗎？現在我們試試看：取一個 64K 的晶片，將其四塊 16K記憶體放在四個位址不連續的不同地方，如圖 3-3所示。這有點像將四塊冰塊放入一池水中，不停攪動使其分開，直到互不接近爲止。

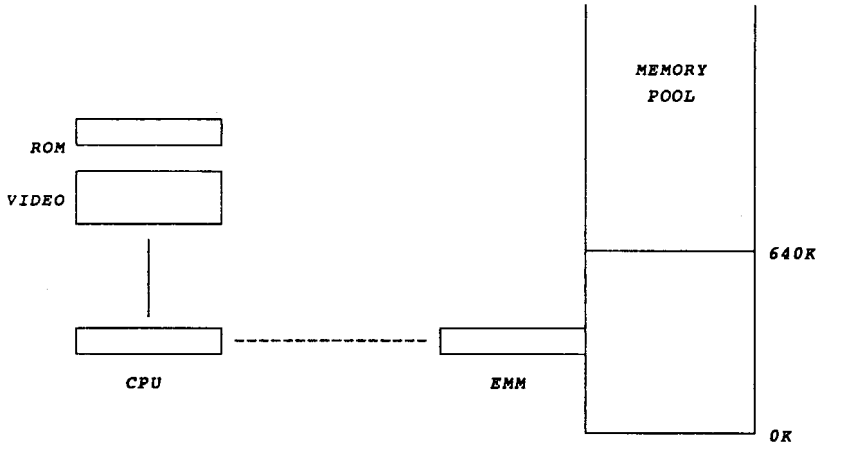


圖 3-3　有管理器的自由代理記憶體，CPU 仍以系統位址直接使用 ROM及視訊記憶體，而只能 "看見" 記憶體管理器所分配的 RAM。這是多工處理與功能完整的擴充記憶體的一個關鍵。

記憶體管理的實際情形並不如同冰塊在一池水中這般散亂，且相去甚遠。事實上位址配置係遵循著一些由系統的設計者或程式設計人所訂下的嚴格規則。但將每塊 16K記憶體以隨機方式處理的能力仍必須加入硬體中，因爲記憶體管理程式絕對會將大部分可用的記憶體資源取走。

事實上，這是一個處理電腦記憶體時，最常發生誤解的觀念。今天許多買了原廠配備 1 megabyte 記憶體的使者就認爲記憶體是由 0h 開始到 FFFFh 爲止完全連續的，且深信不移。

除非製造商眞的沒有在 640K 到 1024K (A000h 到 FFFFh) 間動了手腳，你才可能有原廠配備的 1 megabyte

記憶體。因爲視訊記憶體完全是另一個獨立的東西且通常由另一塊視訊界面卡 (在某些機器上是與主機板結合在一起) 所提供，故我們先將之忽略。事實上，除此之外，系統在這個範圍內並沒有任何隨機存取記憶體，有的只是通常由 F000h 開始的唯讀記憶體 (ROM)。

這表示如果我們眞的有了整個 megabyte 的記憶體，那麼後面的 384K 必定是跳過了 640K 到 1024K之間的範圍，而由 1024K開始 —— 也就是 384K 的延伸記憶體。不久後我們將看到，延伸記憶體不只可做延伸記憶體，只要有正確的軟體配合，常常都用做擴充記憶體。其實無論那一種情況都必須有適當的系統設計及軟體。

到目前爲止，擴充記憶體仍是大多數使用者所希望，或至少是他們的軟體所希望的，延伸記憶體仍只被視爲一項極具潛力的有力工具而已。現在我們已有了足夠的基礎，也知道了一些基本規則，可以來看看傳統 640K 記憶體之後究竟有些什麼玄機。我們將先從擴充記憶體看起，再看延伸記憶體，最後我們會看到二者是如何一同工作的。

第四章　更多的記憶體

到目前爲止，讀者應該已經了解各晶片及最基本的系統 —— 晶片如何在各個系統中使用，以及它如何在加裝的記憶體上動作 —— 的功能了。所以，現在是我們研究研究記憶體的時候了。附加 (additional) 記憶體有兩種：實際的 (real)及 "僞造的" (fake)。當然，我們並不是這樣稱呼它，而說它是 "實際的" —— 表示記憶體是由隨機存取記憶體 (Random Access Memory;RAM)晶片所組成的，它是實際存在的記憶體。或是稱它爲 "虛擬的" (virtual)—— 雖然實際上不存在，但卻是很有功用的。

雖然虛擬記憶體可能會使我們產生幻覺，但它並不是全然 "僞造的"。 也就是說，當沒有足夠的空間可以存放某些資料時，存放在虛擬記憶體的資料不能被竄改，也不能在這個地方執行程式碼。虛擬記憶體只是一個給定特定名稱的磁碟儲存區。如此，便可將資料置換 (swap) 到磁碟上，而不必提供較多的 RAM，可能需要一筆超乎我們所能夠想像的數目。因此，記憶體被分或兩種 —— 不管你是怎麼稱呼它。

若對這兩種附加記憶體再加以細分，又可以分成兩類。

因爲實際記憶體比較重要，所以在此先分析實際記憶體。它的附加記憶體包括延伸 (extended) 記憶體及擴充 (expanded)記憶體。

在這兩種記憶體當中，讀者對於擴充記憶體應該不會感到陌生 —— 即使到目前爲止，你可能還不太清楚這兩種記憶體的差別。在這兩種記憶中，只有擴充記憶體能直接從 DOS 存取，因爲不管記憶體有多少 megabytes，DOS 只能處理位址在 1 megabyte 之前的部分。而延伸記憶體則剛好相反，其所有的記憶體位址，均在 1 megabytes之後，這些位址只能透過實際上架空的 DOS來存取。

4.1 擴充記憶體究竟是什麼?

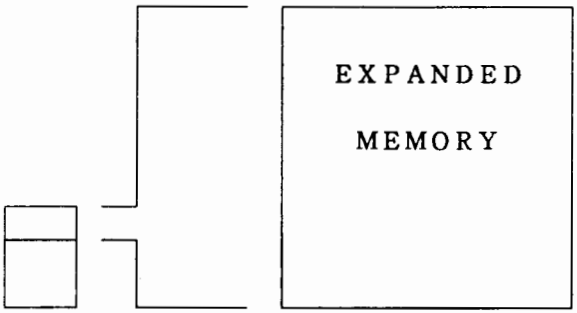


圖 4-1 4.0 版。使用傳統的置換 (swap) 方式。原始的擴記憶體是完全分開的記憶體區塊(block)—— 最多可達 8 megabytes。透過在高於 640K上的未使用位址空間的頁框，可以一次存取 64K(4個 16K的頁)。

圖 4-1是擴充記憶體的簡圖。透過一個在轉換處理基礎

(revolving access basis)上的 64K頁框 (page frame) 可以處理八 megabytes的空間。而轉換處理基礎則是由擴充記憶體管理程式 (Expanded Memory Manager)所管理。

這只是一個簡單的開始，讓我們從主要架構中，借用一點技巧 —— 記憶庫切換 (bank - switching),以便舉出所有的可能，並且讓 DOS更加生動活潑。關於 LIM 4.0 EMS的介紹，它的範圍以及可能性已經戲劇性的改變 (擴充) 了。但基本的觀念是不變的。爲了不再使讀者更加混淆，下面就開始加以詳細說明。

擴充記憶體有時被比喻成像是一盤在晚餐桌中間的火雞大餐。這個比喻其實有一個問題：在擴充記憶體中並不會發生眞實的移動。不過，這個比喻還算適當。

假設你坐在有一份火雞大餐放在中央旋轉餐桌上的桌子旁，橄欖與梅子醬就在你面前，而火雞則在較遠的一邊。你轉動這盤火雞大餐，然後用叉子取下一塊火雞肉。這時橄欖與梅子醬仍在餐盤上，但已轉到了搆不到的另一邊去了。當你準備用這些調味料時，只要再轉動一次餐桌就行了。就像增加更多的擴充記憶體一樣，這個旋轉餐桌越大，每次旋轉後，手臂範圍內所能拿到的東西也越多。

如果所有的東西都放在固定的位置，即使是個盲人，只要經過一些練習，也能找到他所要找的東西。當然，即使在載入時可以有彈性地使擴充記憶體，幾個順序不一定相同，或不一定會開啓相同的檔案的應用程式，我們也不必在意記憶體管理程式在昨天的工作中的什麼地方塞了什麼東西，也不用擔心它們這一分鐘在什麼地方，因爲記憶體管理程式知道在什麼地方，且能將其以飛快的速度 (RAM 的速度) 取出

，並放入某個 DOS能夠辨識的地方（如某種 "窗" (Window) 或" 框" (frame) ）。

這個類似用來拿取旋轉餐桌上某些東西的湯勺的框，叫做 "頁框" (page frame)。更詳細的細節將在下章中集中注意力於擴充記憶體時討論。目前我們只要說，頁框是一塊 640K以上，1024K 以下，DOS 可觸及的位址就足夠了。

在 LIM 4.0 EMS的規格下，擴充記憶體已不再是對一大塊額外的記憶體一次只能存取一小部分而已。與目前在其上所能做的事比較，擴充記憶體已發生戲劇性的改變。在新的規格標準下，沒有一塊 DOS的 1 megabyte 定址空間是超出限制的。依 4.0 EMS的規格，擴充記憶體可在一瞬間對任一部分做切換。見圖 4-2。

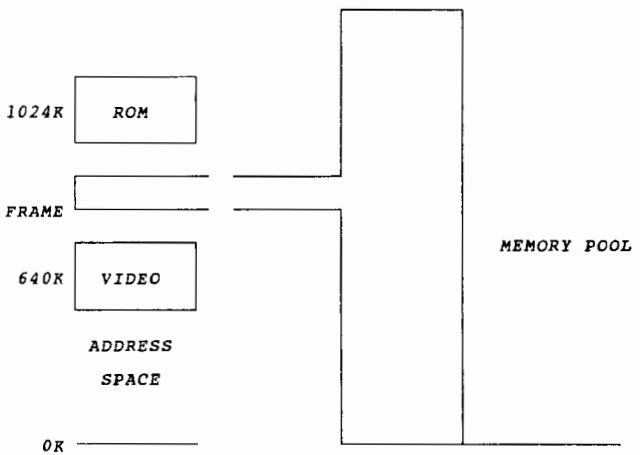


圖 4-2如果製作完全，在 4.0 EMS下，載入與執行程式所需的 RAM 也由一個共用資源庫中取得，如果這個資源庫夠大，則可同時載入數個程式，且利用快速的記憶庫切換，使它們看起來是同時執行。這在只支援 3.2版擴充記憶體的硬體

上無法做到。

對位址可做無限制的存取一旦被做出來，肯定是LIM 4.0 EMS 最令人興奮之處，但也是大多數使用者最難以了解之處，無論是在觀念上或是其對電腦使用上各方面的影響。這將是下兩章中，我們專注於各有關的軟體與硬體時要討論的焦點。

除了提供執行較大程式的能力之外，擴充記憶體另一個較大的貢獻，是提供了產生多重同時工作環境 (multiple simultaneous work enviroments)的能力。這表示在注意時間內，我們可以一次載入一個以上的應用程式，且可依需要在其間來回切換 (當一個暫停時，馬上用另一個接替) 。如此一來，386 系統上的多工處理已不再只是可能，而是許多應用程式的有力工具。

不過，還是有兩件事沒有改變。第一就是類似旋轉餐桌的觀念，對記憶體的存取仍須輪流完成以能將其區別。第二是一些目前仍在市場銷售的硬體設備，雖然宣稱完全支援LIM 4.0 EMS 的功能，實際上卻只提供 3.2版的單一 64K頁框。因此，如何使消費者覺醒就成了另外一件非常重要且不曾改變的事。

4.2　延伸記憶體：一個沈睡的巨人

BONANZA 金礦 (R)

如何喚醒一個沈睡中的巨人？＂非常小心＂是一個答案。直到最近，延伸記憶體仍是一個有很多人談論，只有少部分人眞的使用，更少的人能夠了解的東西。後兩句話仍是對的，但最近發生了一些極令人興奮的事，所以現在應該是仔

細看看延伸記憶體的時候了。

如果擴充記憶體像是旋轉餐桌，延伸記憶體就 像是一幢十層建築中，穿過屋頂而直指外太空的電梯。我們的屋頂是 1 megabyte,而延伸記憶體以實際線性位址一路直上，用 80386 可達 4 gigabytes (4000000000 位元組)，用 80286 則是 16 megabytes 。

大部分是因爲我們在第二章中討論過的，由 80286而來的一些問題，延伸記憶體的起步十分緩慢。其實在延伸記憶體上能完成的事比在擴充記憶體上能做的事多得多。例如，如果沒玩什麼花樣，在 DOS下的擴充記憶體不能執行超過 1 megabyte的程式 (甚至只能 "接近" 1 megabyte) 。但在延伸記憶體下，" 天空"(還有你的荷包) 才是你的界限。這可能要歸功於虛擬 (僞造的) 記憶體。

即使將來的趨勢眞是延伸記憶體 —— 而目前一些跡象也顯示對許多使用者來說確實如此 —— 也不表示我們應該絕望，只須作壁上觀，等著看結果會走上那一條路。很幸運的，在目前或是可見的將來，我們並不需要在延伸記憶體或擴充記憶體間馬上做一個選擇。目前眞的應該要做的選擇是：我們是否能活在 DOS正常的 640K 之內。如果覺得不能，接下來要做的就是選一個能提供兩種記憶體之一或二者俱備的硬體。

本書後面將討論購買硬體時該注意些什麼事。現在我們只要知道好的硬體可依需要提供擴充記憶體或延伸記憶體。實際上你現在正使用的擴充記憶體，可能原是延伸記憶體，被一個高明的週邊驅動程式轉換而來，而使你的應用程式以爲是擴充記憶體。

因爲延伸記憶體是線性的，位址由 1024K處開始，且由此向上發展，也就是說所有的位址都在 DOS允許的 1024K之外，所以在 DOS中無法進入這個區域。現在有一個有趣的問題，我們是否能以一些如 OS/2 或其他定址空間不限於 1024K 的作業系統來取代 DOS ？

到目前爲止，對大部分的使用者來說，還沒有一套可用來代替的作業系統出現。不過還是有一種叫做 DOS延伸程式(extender)的軟體。DOS 延伸程式提供了以有規則的方式存取較高位址所需要的界面。在此 "規則" 是個重點。除了 80286 本身的限制及缺乏一套可用延伸記憶體的作業系統之外，長久以來，延伸記憶體的應用發展緩慢的另一個原因，就是缺乏在 1 megabyte 以上一些可被接受的規則。

這對一個以上應用程式同時操作來說是產生了一個不安全的環境。例如，我們正把延伸記憶體拿來做 VDISK (或其他 RAM disk)，又試著想同時在延伸記憶體中裝入印表機驅動器，如此一來其中一個可能會蓋過另一個。在這種情況下做虛擬磁碟的結果，光用想的就很恐怖。保護模式原就是用來防止這些問題的，但在 DOS下無法工作。DOS 無法辨識所有超過 1024K的東西也是另一個好理由。所以，DOS 根本不需要任何用來管理 1 megabyte 以上記憶體的規則。

4.3 DOS 延伸程式 (EXTENDER)

80286 及 80386雖具有保護模式操作的能力，但直到 DOS 延伸器的到來才爲 1024K以上的工作找出一條路。

不論有沒有規則，4 gigabytes 或80286 的 16 megabytes 都需要大量的記憶晶片。爲了便於討論，假設每一 megabyte 要美金五百元，那麼一部 286光是 RAM晶片就要八千美元。如果裝上所有 80386系統所能支援的記憶晶片，估計要八百萬元。以 80386的情況爲例，即使付得起這個價錢，實際上也沒有一個可以連接這麼多晶片的方法被設計出來。RAM 現已隨著科技的進步大幅下降。

這數字看起來雖然很荒謬，但卻不是沒有意義的，而且很實際。因爲當我們沒有足夠的記憶晶片可用時，我們還有另一個方法 —— 虛擬記憶體 (virtual memory)。這是用其他種類的資料儲存設備來模擬隨機存取記憶體，通常是以硬式磁碟機來模擬。

4.4 非傳統式記憶體

BONANZA 螢圖 (R)

近幾年來，人類發明的能力總是不會讓阻擋在路中的障礙存在太久。即使是 PC 也開始掙脫束縛並擺脫那種 "玩具" 的印象。程式設計師已經發現了一些方法，其中不少是由大型電腦的世界中借用而來，可以使每一種可得的有限資源發揮最大效用，使其看來似乎不只這些。

這對記憶體來說特別眞切。因爲無論如何分割，記憶體仍是有限的資源且超過限制後硬體便無法動作。因此，隨著程式大小的增長，程式設計師便被迫尋找解決的方法，看是否能使程式的大小超過傳統使用者系統上可用記憶空間的數量。

其實記憶體可以有很多種型式，近幾年在這方面也有不

少成果。例如將磁性物質串成蜘蛛網狀的矩陣，或精密的光學設備如使用雷射的寫入一次讀出多次光碟 (Write Once Read Many disk , WORM)，且沒有人知道下一種是什麼。

在一般資源的發明中，使用者記憶體大致可分爲主要的兩類。第一是隨機存取記憶體 (RAM)，這是一種 "易變的" (volatile，指容易因電源等因素而改變) 儲存媒體，通常用來做暫時性的儲存與處理。另一種型式是 "不易變的" (nonvolatile)大量儲存媒體，這在今天通常是指硬式磁碟，其容量往往遠超過所有隨機存取記憶體的總和 —— 即使包括延伸記憶體或擴充記憶體。

就如同一些經過巧妙程式規畫的隨機存取記憶體看起來像儲存磁碟 (如 RAM disk)，我們也可以反過來，使部分磁碟的儲存空間看起來像是更多的隨機存取記憶體。在許多情況下，這些 RAM就是你眞的很希望你有，卻買不起的部分。

當然，對眞實的磁碟存取的速度要比 RAM慢上許多。不過，經使用一些不同的技巧，部分很複雜的虛擬記憶體套裝軟體，可以切掉大部分時間，使對多數的應用程式而言，幾乎沒有感覺。因此，我們不必爲了怕有損顏面而將之摒除在外，特別是考慮到增加 4 gigabytes其他種類記憶體的花費及其他問題時。

順便提一下另一個推薦使用虛擬記憶體的有趣理由：虛擬記憶體可以支援眞實記憶體以應付超大型的應用程式。在使用虛擬記憶體的情況下，80286 的能力一躍而至 4 gigabytes ，而非原來限制的 16 megabytes 。而80386 的虛擬記憶體更可達 64 terabytes 。

讓我們回過來看，虛擬記憶體一個很簡單的形式下，我們可以將整個應用程式及所有開啓的檔案拷貝回我們的硬碟虛擬記憶體裡的一個特殊檔案中。我們還可以將視訊資料拷貝進去，如此我們可以空出整個工作的空間（在 DOS及所有常駐程式載入的部分以上）來做其他的用途，而這個空間仍被保留，所以我們可以跳回來，此時游標依然閃動，就如同離開時一樣。Software Carousel 就是一個典型的這種程式，無論給它那一段記憶體都能工作。

當然，" 跳回來" 這句話不是很嚴謹，因爲拷貝到磁碟需要一些時間，且這個技術也不是永遠適用。但雖然有些限制，仍然增添了不少便利。對許多很需要多工處理的使用者來說，這又使他們更接近了一些，特別是需要超過 640K 記憶體的限制時。

4.5 使虛擬記憶體更近眞實

BONANZA 螢幕 (R)

在某些方面嚴重降低速度的東西都會變得令人難以忍受。這是在這個高度競爭社會中生存的第一守則之一。請記住，任何一種可能造成減慢磁碟存取速度的要求都應該拒絕接受，特別是當我們在討論一個大小達數百萬位元組，且在延伸記憶體中執行的程式時。

很明顯的，實際上的程式就這麼大而已，但就技術層面而言，正在執行或最近一次執行的程式碼留在隨機存取記憶體中，而已經不用，或已有一段時間未用的部分則被換出 (swapped out)至磁碟上的一塊區域去，而這個區域程式看來仍像是隨機存取記憶體。因此，如果有足夠的 RAM以支援大多數程式的需求，在整個執行過程中，將幾乎無法察覺虛

擬記憶體的存在。

讀者可能對 "覆疊" (overlay)的想法十分熟悉，但虛擬記憶體與覆疊的觀念有很大的不同。使用覆疊觀念的有一個中央核心部分，這個核心最早被載入並一直留在記憶體中，用自磁碟中載入並叫用其他部分的覆疊程式以完成特定的工作。簡單的說，" 覆疊" 就是把舊的部分 "倒掉" (更正確的說是將之蓋過) 。然後依需要載入其他部分以完成工作。無論一段覆疊程式被使用的頻率有多高，一旦被蓋過，就一定要再次從磁碟中叫出。這需要程式設計師將程式分成一些能互相呼叫的小程式 (通常用 OVL爲附加檔名以資區別) 。

而虛擬記憶體不僅較快，且在作業系統管理下 (或在 DOS 下經由某較大的指令集以取代原作業系統直接提供者)，虛擬記憶體是 "無形" 的。其對使用者與程式設計師來說都是看不見的。且眞實的系統大小限制被去除後，基本上程式設計師可自由地撰寫任何大小的程式。

當然，虛擬記憶體也有如眞實的 RAM一般的大小限制。不過這個界限大得多，80286 的虛擬界限是 4 gigabytes (與 80386 的眞實定址界限相同),而 80386的虛擬限制更可達到 64 terabytes (64000000000000,64兆位元組),我們當然可以找到是夠的硬碟把這些裝進去，但我們的 RAM只裝得下幾個 megabytes而已。這些數字看起來是否似乎不太可能？或許，但變成可能的時間也不會太久了。

在多工處理 (時間切割) 環境下 (在第二章中有很詳細的討論) 使用虛擬記憶體時，有一點需要特別注意的，與電腦本身的各部分比起來，磁碟存取的速度是相當慢的。而多

工處理其實是因爲微處理機能在兩個或更多的工作間做高速來回切換所產生的 "幻覺" 。如果分配給某個使用虛擬記憶體的工作的時間太短，可能連在磁碟上尋找資料及讀出 (或寫入) 資料所需的時間都不夠。

這可能造成一種叫 "振盪"(thrashing)的結果，也就是說，一個應用程式每次輪到其工作時都存取磁碟，卻永遠沒有足夠的時間做完。很明顯的，沒有一個好的作業系統或多工處理系統會容許這種事情發生，多少都會設法避免。讀寫動作的時間其實可以有效減低，但如果磁頭仍必須不停地在內外軌道間來回移動，在這種情況下，時間將大部分都消耗在此處。

4.6 其他有關速度的因素

對硬碟做換入及換出資料的工作很明顯的要比在 RAM上工作慢得多。不過還有一些差異沒這麼明顯的，如所有的隨機存取記憶體存取速度不是完全相同的。隨機存取記憶體有不同的速度 (當然價錢也隨著不同),爲了要得到最佳的效果，最好選擇能與各特定系統匹配的。

動態隨機隨取記憶體 (Dynamic Random Access Memory ,DRAM,這是最常使用的一種) 速度的範圍通常在 80 到 250 奈秒，也就是數十億分之一秒之間。儲存在 DRAM 中的資料是以微量電荷的形式存在的。這些電荷必須週期性地 "復新" (refreshed),否則這些電荷 (也就是你的資料) 將慢慢地流失，最後完全消逝。因此每一定時間內復新資料是每一部電腦中一項主要且不斷進行的功能。

這些電荷會隨時間緩慢消耗並不是惟一的問題。每當我們需要存取資料，每個讀取的動作都會抽掉每個位元一些寶貴的電荷。如果不在其復新之前再次讀取資料便不會發生什麼問題，否則可能會有一些令人不太愉快的結果 —— 你拿到的資料是錯的。在廣告上標示的時間 (通常也會打在每個晶片上),是這個晶片被讀取後，下次可被讀取之前，復新所需的時間長度。

實際上我們處理的頂多是數十億分一秒，但即使如此，市面上只有少數的 DRAM 能跟得上全速運作 (如 16Mhz) 的系統。有另外一種記憶晶片 —— 靜態隨機存取記憶體 (Static Random Access memory ,SRAM)，其速度便快得多。這是個好消息，壞消息則是，與 DRAM 比起來，其價格貴得多，且儲存容量通常也較小。

還有另一個我們常常提到的選擇，叫做 "等待狀態" (wait state)。我們可以這樣解釋：電腦必須等候以使其記憶體能跟得上。而所謂幾個等待狀態是在每個記憶體週期中電腦插入 "暫停" 的數目。

其動作的大概情形解釋如下：一個正常的記憶體週期包括兩個微處理機週期；一個是微處理機送出對資料的要求，另一個則是資料送回。加入一個等待狀態 (在本例中等於加入第三個微處理機週期) 將使電腦的有效速度減少三分之一。這在某些情況下特別明顯，例如一些高速的 80386機器加了兩個或更多的等待狀態後，其實際的表現與我們所期盼的比起來會慢了許多。因此，時鐘速度只是整體性能表現的一部分而已。

設計人員還可使用一些技巧使系統實際上以較高的速度

執行，用 SRAM 做資料快取記憶體 (cache memory) 是其中之一。否則當我們去買更多的記憶晶片 (或插有記憶晶片的附加卡) 時，應該留意自己需要的是什麼，晶片本身的規格又是什麼。(多快? 多大?)

事實上，設計人員用了很多技巧，而由我們看來這些全歸於數種不同種類的記憶體，每一種都有一些與眾不同的特點。我們將進入另一階段，以我們能用記憶體做些什麼的觀念，來仔細研究記憶體。

第五章 擴充記憶體

擴充記憶體使我們首次能使用超過 640K 的記憶體。依原來的定義，擴充記憶體只是用來增加記憶體，使應用程式有更多空間儲存資料。不過，經過後來一些修改，今天的定義已增加了一些功能，可提供動態的操作環境，也就是可以在其上執行程式碼，而不只是用來儲存資料。

關於擴充記憶體的發展，最早只有 Lotus公司。許多 1-2-3 的使用者非常需要記憶體以裝入較大的試算表，且常需要超過 640K 。此時 PC 脫離了玩具的階段。Lotus 的使用者不是惟一需要更多記憶體來工作的人，但當時似乎只有 Lotus 注意到這個問題，也因而成爲擴充記憶體的發展早期主要推動者之一。

Lotus 注意到了且馬上有所行動。不像一大堆只會說 "沒有問題，DOS 在接近頂部處還有很多未用的位址區塊。" 的供應商。 (我們在本章中提到系統當機時會遇到這些問題。) Lotus 找了 Intel公司共同試著尋找一個能滿足大數人的實際解決方法，由此推出了擴充記憶體的第一個規格標準

。

Microsoft 最初的確沒有加入。但 Microfoft的名字既然能加入其中 (LI "M"), 當然是對這個雛形有一些重要的貢獻。不用說, 其間必然徵詢過 IBM的看法, 也必然加入了一些 IBM的意見。最後, 某些關於位址區塊的部分, IBM 擁有大大的 "保留" 字樣。但無論他們曾扮演什麼角色, IBM 的名字還是沒有加進去。事實上, 在 DOS 4.0 推出之前, IBM 從未正式地承認過擴充記憶體。

最初 3.0 版的規格少了不少東西, 無法滿足許多需求若渴的應用程式及使用者, 如 Lotus的大試算表使用者。3.0 版很快就被修改。後來 3.1 版加入了一些 Lotus所需要的特徵。最後 3.2 版推出。3.2 基本上仍是以早期的版本爲輪廓, 然後加入一點東西, 因此許多人認爲 3.2 版沒什麼進展。

認爲 LIM規格沒什麼進展的人, 主要是一些其他的軟硬體供應商。當時持這種觀點的供應商比使用者還多, 是因爲 EMS 雖然完全是一個軟體標準, 仍必須有硬體的配合, 這表示我們必須先有硬體。對硬體供應商來說, 擴充記憶體無疑是個大金礦, 是一種全新的產品。

Intel 公司當然是其中之一, 在早期的階段他們推出了 AboveBoard, 隨後衍生出一系列 Intel AboveBoard 。後來又出現了一些相似或相容的產品。由於 Intel曾參與 LIM EMS 規格標準的發展, 他們的產品當然爲標準提供了足夠的硬體支援。

AST 研究公司也很快地加入這場競爭, 但他們的

RAMpage 卡走到了一個相當不同的方向。基本上 AST 遵循 3.2 版規格所設定的規則，且完全與其相容，不過向前邁進了一步。AST 注意到了那些認爲 LIM標準進步不夠多的人，因而發展了一個較大的版本 —— 加强擴充記憶體規格 (Enhanced Expanded Memory Specification, EEMS)。EEMS 加入了一些 LIM使用者所期盼的額外特徵與功能，AST 也讓他們的產品提供了 EEMS 標準所需要的硬體支援。

最初，包括 AST本身，沒有人知道實際的運用如何與這些吸引顧客的先進特徵相結合。但過了沒多久，一個自稱爲 Quarterdeck 辦公室系統的無名團體出現了，並展示了他們所擅長的多工處理。這也該是多工處理有所發展的時候了。此處的多工處理並非只如 MS-WINDOW或 Software Carousel 所提供的程式切換 (context switching)方式，而是眞的多工處理，且即使是舊的 8088PC 也能做到。但我們必須先有一塊支援 EEMS 的卡，及總和超過 640K 的任意程式組合。

EMS 的規格標準顯然並不夠好，所以 Lotus ,Intel ，及 Microfoft三家公司又推出了一個新的版本: LIM 4.0 EMS 。許多 Lotus的使用者對 3.2版規格標準不表滿意，因爲在實際使用上，只能存取到該版所承諾的 8 megabytes的一半。至於 Intel，無論他們所銷售的附加卡或這個 EMS標準，都無法做到其所聲稱的支援多工處理。

更糟的是，Intel 有一個新的微處理機 —— 80386,非常適合做多工處理，卻只有 EEMS 能支援。因此，一個加入了先前 AST的 EEMS 卡上大部分特徵的新規格標準誕生了。

早期的 LIM擴充記憶體規格標準有一些優點，而 4.0版則更能有效地使用擴充記憶體，例如:

- 使試算表的使用者能更有效地使用擴充記憶體。
- 具有在擴充記憶體中載入與執行記憶體常駐程式 (如TSR)的能力。
- 一群應用程式可使用放在擴充記憶中的共用資料。
- 可做較大的虛擬磁碟，印表機驅動器，及磁碟快取(cache)
- 大小限制由 8-megabyte 增大爲 32-megabyte。
- 允許多個程式同時在擴充記憶體中，且有較佳的性能表現。
- 允許一次定址 64K以上。
- 允許定址 640K 以上及以下的記憶體。
- 可做高速記憶庫切換。

4.0 版規格可與早期版本相容，以便使那些以 EMS 3.2版及 EEMS (AST公司) 爲標準設計的程式能在其上工作。而新標準的完全製作版亦可在支援 EMS 3.2及 EEMS 的硬體上執行。在很多情況下，新的週邊驅動程式的性能都被要求能跟上支援後來 4.0版標準的硬體，但除了支援 EEMS 標準的之外，大部分都做不到。

4.0 版規格爲軟體發展者提供了 15 個新的功能及 39個新的子功能，這個的好處對使用者來說並不特別明顯。但就整體而言，這些其實才是 4.0版規格標準的精神與重心所在。這些新功能包括:

- 多頁映射 (multiple page mapping)眞正提昇一般情況及資料保護上的性能表現。
- 擴充記憶體的配置量可動態地增長或縮減　這使記憶體在加入新的應用程式，或想增加原來的配置數量時，不

必再回到記記憶庫 (pool) 中。

- 模擬遠程跳躍 (far jump)及遠程呼叫 (far call) 使程式設計師可以在擴充記憶體中執行程式。
- 具有拷貝或交換一區域記憶體的能力 可由傳統記憶體到擴充記憶體，或由擴充記憶體到傳統記憶體，或由擴充記憶體到擴充記憶體。
- 具有一次對映 64K以上且/ 或映射至傳統記憶體的能力以某些方面來說這可能是 4.0版最重要的特徵。

由此開始有 "回填" (backfilling)的能力及可置換整塊記憶體 (256K以上的區域，但未做嚴格限制) ——包括正在執行中的程式碼。由此也可做多工處理，使能在背景中執行。

由最初的 LIM EMS規格標準及其最早的目的開始，擴充記憶體至今已走了很長的路。1-megabyte記憶體的電腦已可充分利用。但請記住，所有 LIM 4.0 EMS中所提到的，完全是一個軟體規格，而沒有談到硬體。當然，我們要有擴充記憶體，軟硬體二者缺一不可，但事實上 EMS的硬體規格標準一直都不存在，支援或不支援這個軟體標準，完全由各製造商視其需要而自行決定，其中有些重要的不同。爲了加快速度，我們將軟硬體分開討論。

5.1 基礎

就如同我們已經看到的，DOS 因歷史的及一些無關緊要的理由，將記憶體分成十六個 64K的區段，十個給使用者 (0h 到 9h)，六個留給系統 (Ah到 Fh)，設成頁框 (我們到擴充記憶體的窗) 的區塊也因而成了 64k的區塊，特別是在

3.2 版規格下，區塊可由 C000h ,D000h ,E000h 開始，見圖 5-1。

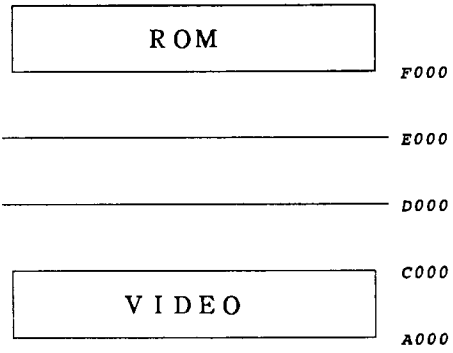


圖 5-1 頁框位址，擴充記憶體的頁框可由 C000，D000，E000 三者之一開始，但不用非常準確，只要是 16K的倍數即可。

由於 3.2版 EMS規格一些其他的限制，即使有 128K 記憶體可用，但仍必須在兩個 64K區塊之間選擇其一。供應商在提供選擇時就設了範圍。AT推出時，IBM 要求 E000h 直到 FFFFh 的部分保留給唯讀記憶體，而 PC 並沒有用到這個地方，實際上他們也沒有用到這額外的 64K區塊，這樣說只不過是用來嚇退一些供應商罷了。例如 Intel公司設計他們支援 3.2版標準的 AboveBoard 系列時，便只用了 D000h 到 DFFFh 間的單一頁框。

一些其他的製造廠，特別是 AST公司，至少給買主要或不要存取第二個頁框的選擇，視他們可能使用何種其他作業系統而訂。後來又出現了一個我們今天所用的，是一個目前被接受的標準，但在不同廠商的產品上有相當大的不同。

當我們購買一項產品時，注意且了解這些差異是很重要的，因爲在某些情況下，這些差異對性能有重要的影響，一個比較典型的例子，如前面所述，是在 3.2版規格標準時期，AST 公司上市的一系列擴充記憶體產品，在維持與 3.2版標準完全相容的同時，AST 公司在他們的 EEMS 中加入了一些額外的功能，以支援一些原來的規格標準完全不及的需要。

仍因 EMS規格標準完全是一個軟體規格，在此時期撰寫的軟體不是全部都能充全利用 EMMS 所提供的特性，那些增加的功能大部分都沒有加進今天的 LIM 4.0 EMS規格中，應用軟體對各標準採用的程度將在本書後面詳細討論。

請參考圖 5-2 ,注意這些 64K區塊 (或 "框") 更細分爲一些 "邏輯上的頁" (logical page)，每個頁有 16K，也就是一個框有四個頁。共三個框，每個框四個頁：因此這個規格現在包含了共十二個邏輯頁的配置空間。EEMS及後來的 EMS 4.0 在 640K 以下的位址時沒有什麼不同，都是盡量保持簡單，不久後我們就會看到。

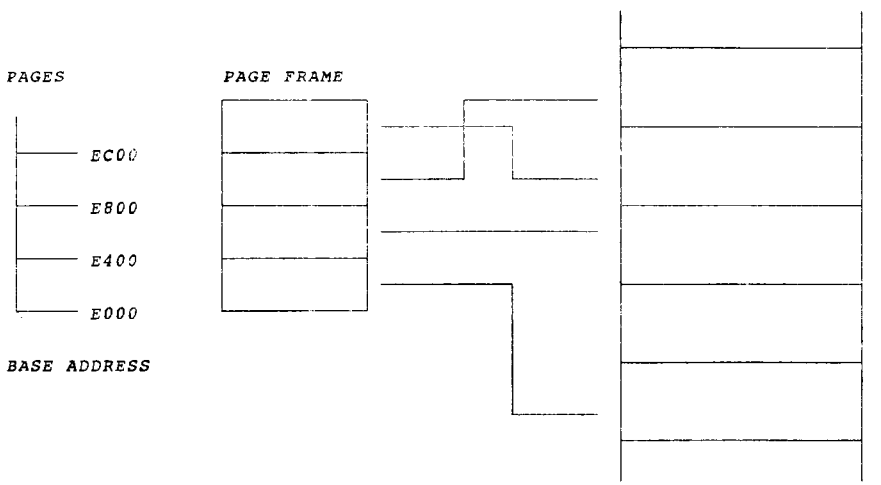


圖 5-2 頁框與邏輯頁。每個頁框分成四個 16K邏輯上的頁，本例中，頁框的基底位址爲 E000。

就如同我們所看到的，這些框都有特定的位址，每個框所含的四個 16K頁也都有特定的位址；所有的位址都在 DOS 1024K 的範圍內。這些頁框的基底位址並不一定要是 64K的整數倍，只要使用位址在 640K 以上，且未被 ROM或 RAM佔用的 16K區塊，便可以是 16K的倍數。而 DOS在這些位址上"看到"的實際記憶體區塊，其實可以在此之外的任何地方，只要裝有 RAM晶片，擴充記憶體允許最大到 32 百萬位元組。但無論在何處，DOS 只是把對擴充記憶體的位址叫用當成 C000h到 EFFFh間的位址叫用，且不將之視爲擴充記憶體或其他東西，而像對一般位址的一般叫用。

因此在操作上，所有超出範圍的擴充記憶體位址叫用，都必須轉成合法 DOS位址的區塊。也就是要旋轉 "餐桌",使得被叫用的眞實記憶體區塊，能被帶到與其位址對應的 "窗" 中，而不論其中是程式碼或一般資料。

當然，在 DOS關心的範圍內，這些事實際上都是不存在的。DOS 只有默默地注意著其 1024K內的工作。DOS 眞的是很笨，完全只是一個傳送訊息者。因此，一些其他的軟體都必須在此等待。

我們還必須有個東西把眞實的叫用攔截下來，記下被叫用的程式碼或資料放在何處，然後前去將之取出，最後帶回來放在 DOS認得的地方。這個東西通常叫做記憶體管理程式，在許多方面其重要性甚至超過眞實記憶體。由此我們可很清楚看到，我們根本不能沒有這個東西。

記憶體管理程式通常被歸於一類我們叫做 "設備驅動程式" (device driver)的軟體中。現在我們非常有可能在我們的電腦系統上裝設一個或更多的設備驅動程式，假如目前正使用擴充記憶體，那更可確定。我們可以在 CONFIG.SYS 檔案中，看到這些驅動程式的檔名，通常是以 .SYS 結尾。

週邊設備驅動程式通常提供一些週邊設備 (可能是眞實或虛擬的) 的輸入輸出服務。DOS 都會支援這些驅動程式 (或由其他作業系統支援，視你使用的系統而定) 。每個設備都必須要有一個驅動程式以做爲其與 CPU間的界面。但我們並不一定需要知道驅動程式的存在，因爲有些是 "內部的" (internal) (屬於 ROM BIOS 提供的服務的一部分) 。

其他的便是 "外部的" (external)，這表示我們必須先裝設它們，此處就要用到 CONFIG.SYS 檔。如果我們有一些週邊設備，不是如軟碟或硬碟等 ROM BIOS 或 DOS直接支援的，則每一個都必須裝設一個屬於自己的外部週邊設備驅動程式。如虛擬磁碟，特殊的印表機，或特殊的監視器，所有在正常範圍之外的，只要沒有驅動程式，週邊設備便無法動作。而簡單的說，以電腦的觀點來看，擴充記憶體也是一個 " 設備" 。

現在我們提到了眞實的設備，而到目前爲止，旋轉餐桌的比喻依然適用。對這類儲存媒介做輸入輸出確實需要旋轉一個眞實的圓碟至其能寫入資料的位置，而以後也由此處取出這些資料。這是個不錯的比喻，可以記在心中，因事實上確實有些擴充記憶體的產品用到眞實磁碟上的空間，以代替一般擴充記憶體，或當一般擴充記憶體不敷使用時，做爲補充之用。因此，我們可做一些直接的比較，或許可以使我們對擴充記憶體的世界有更清楚的了解。

5.2 擴充記憶體如何工作

BONANZA 螢幕 (R)

我們已經看到，擴充記憶體（或延伸記憶體）需要我們在我們所有傳統記憶體之外還有另一地方，可用來暫時存放資料（也就是只在某些特殊工作時需要，之後可被存入磁碟中或，如果不再需要，便將之刪除）。現在的問題是擴充記憶體如何工作？我們又如何與其溝通？

擴充記憶體依定義是透過擴充記憶體管理程式來工作。每個擴充記憶體設備，如 Intel公司的 AboveBoard 系列或 AST 公司的 RAMpage等，都需要有一個擴充記憶體管理程式以控制所有可用擴充記憶體的使用數量。請注意 "或" 這個字，我們的系統在任何情況下都不能有超過一個的 EMS擴充記憶體管理器裝置於其中。這是一個必須記住的重點，因爲不同廠商生產的擴充記憶卡需要不同的週邊設備驅動程式，而這些驅動程式不能混用。

週邊設備驅動程式這個軟體界面通常是在系統啓動時，由 CONFIG.SYS 檔裝設 .EMM驅動程式這行在 CONFIG.SYS 檔案中的位置通常沒有嚴格規定。但我們仍須小心，EMM 必須裝設在如虛擬磁碟，磁碟快取 (cache)記憶體等需要擴充記憶體之前，如圖 5-3 (這只是在擴充記憶體的情況下，若用延伸記憶體便不需要)。

```
DEVICE = EMM.SYS PC DC00 258 EXP = 01536 ND NP
DEVICE = QUIKMEM2.SYS 360
lastdrive = x
device = d:\msdos\ansi.sys
drivparm = /D:o /F:2
files = 32
buffers = 20
```

圖 5-3 一個 CONFIG.SYS 檔，用來表示一個用到擴充記憶體的虛擬磁碟驅動程式必須裝設在 EMM之後，下面以小寫表示的項目只做爲舉例之用，沒有特別的意義。

由系統觀之，擴充記憶體只有在裝設 EMM之後才存在，且沒有任何設備可以用不存在的設備。一些其他的硬體架構還可能會指定更特殊的載入順序，如 Intel的 Inboard系列之類的加速卡。若對某特別系統有問題或不解之處，可查閱該系統的使用手冊，或製造商的技術支援服務。

5.3 記憶庫切換 (Bank Switching)

BONANZA 榮團 (R)

在軟體界面裝設之後，要運用擴充記憶體，應用程式必須呼叫擴充記憶體界面，以配置，釋出，或 "記憶庫切換" 擴充記憶體的區塊。記憶庫切換技術其實不是新的東西。IBM 已經在大型電腦主機上使用了很長一段時間，而在桌上型電腦方面，蘋果電腦公司也已經成功地使用上去。

記憶庫切換動作牽涉到由一 "池" 擴充記憶體中 "映射" (mapping ,暫時指派) 記憶體到 1024K以下的一個空的位

址空間。EMS 4.0 與3.2版主要的不同便在於 3.2 版只允許在位址 640K 以上的部分，但二者的技巧是一樣的。而無論這塊記憶體實際上在什麼地方，在應用程式看來，其所處理的仍是一塊位址在1024K 以下的記憶體，也就是 DOS能接觸到的位址。

假設一個試算表需要更大的空間以存放資料，且可使用擴充記憶體，便呼叫 EMM，接下來 EMM配置這個應用所需的擴充記憶體（假設有足夠空間），且一次給 64K（四個 16K 邏輯頁）DOS 定址範圍內的頁框。圖 5-3 中，我們看到頁框由 D000h 開始。不過頁框可以由 C000h到 E000h間任何 16K 的倍數位址開始，視有無其他資料在其內而訂。

這個試算表就可以在 D000h到 DFFFh間寫入最多 64K的資料，然後做一個記憶庫切換的呼叫，EMM 就送另外四個 16K 的頁（也可能是一個或兩個或三個，視需要而訂），這四頁的 DOS位址與原來相同，而第一個四頁（及其他所有頁）的位置會被記下來，因此當需要時可以再拿回來。這個 64K 的 "窗"，只要每百萬位元組重覆做十六次切換的動作，便可以處理數百萬位元組的資料。基本上這又是我們旋轉餐桌的觀念，只是多了一些細節罷了。

5.4 誰能使用擴充記憶體

我們已經定義了擴充記憶體，現在的問題是我們如何運用？程式真能自動地使用嗎？

第二個問題的答案是否定的，完完全全的否定。程式無法自動地取用擴充記憶體，必須在其內寫入一些增加的函式

，如此幾乎所有種類的程式都能使用擴充記憶體 —— 文書處理程式、試算表、繪圖程式、TSR 等。在後面幾章中，我們也會看到一些硬體上的限制，主要是該硬體是否支援 EMS 4.0 標準的問題。除此之外，幾乎全部都能使用。

我們已經說過，要使用擴充記憶體，程式必須依此設計。這不表示一切都要回到原點，而是所有的軟體都要有一些特別給擴充記憶體使用的功能寫入其中：

- 程式必須能判斷擴充記憶是否眞的存在系統中。
- 假設程式找到了擴充記憶體，還要判斷是否有足夠的空間以符合其需求 (函式 3)
- 假設有擴充記憶體，程式必須能依需要配置擴充記憶體的頁 (函式 4或 18)
- 必須能取得可用的頁框位址 (函式 2) [亦見函式 5800h]
- 必須眞的在擴充記憶體中映射 (函式 5或 17)
- 只有在上面動作全部做完後，才能在擴充記憶體中讀、寫、執行資料，且和只有傳統記憶體中完全相同。
  如果不能完成上面五個步驟，便需載入及執行於傳統記憶體中，至此尙未全部完成，因爲：
- 必須有一個機制，使其在離開之前，能將所有的擴充記憶體頁歸還至擴充記憶體 "池" 中 (函式 6或 18)

寫出函式號碼只是要表示這些的確都是包含在規格標準所訂的正式規則中的正式函式。完整的函式列表請見附錄 A 。

在此還有一件很重要的事必須了解：這些規則並未對實際程式設限制，而與 DOS一般規則不同。程式於何處載入與

執行（或更精確地說，被允許在何處載入與執行）及如何離開，在大部分情況下是所有必須改變以運用擴充記憶體的。

不過，有一點必須提出，與上面所列只是程式設計師欲使用擴充記憶體時所必須加入函式的最小數目，當然還有更完整的工具可用來提升軟體在擴充記憶體執行的能力。想做更詳細的了解，請參考各規格標準的相關文件。

5.5 4.0 版的回填 (BACKFILL)

BONANZA 璽圖 (R)

LIM 4.0 EMS 在擴充記憶體管理增加的功能中，一個最有趣，最有力，最難以了解，也最複雜的，就是將傳統位址由 640K 向下回填 (backfill) 至通常是 256K 附近的位址。這個功能只有在擴充記憶卡提供 640K 以下記憶體時才有用。

請注意，一塊完全支援 4.0 EMS標準的擴充記憶卡，與支援 3.2版且提供 640K 以下記憶體的擴充記憶卡完全不同。3.2 EMS 不允許在 640K 以下的部分做記憶庫切換，故當擴充記憶卡可填滿所有空隙以產生一連續的 640K 區塊，這塊記憶體便被固定在某特定位址，且成爲 CPU所有。

而 256K 的下界其實是一個任意的數字，主要是基於很少電腦允許在 256K 以下的部分做記憶體管理這個事實。基本上 256K 都放在主機板上一塊固定的記憶體，通常用來載入 DOS，一些必需的系統功能，及一些 TSR (包括像 MODE 等等的 DOS公用程式)。

大部分舊的支援 3.2版的擴充記憶卡，允許我們指定其

所擁有的任意數量記憶體以將系統擴展到完整的 640K 使用者空間，然後把我們剩下的任何部分指定爲擴充記憶體。

而差異就在 "指定" 這個地方。在 3.2版下，當我們要從擴充記憶卡指定一塊記憶體給我們的系統，使其能達到我們給的全部容量，我們要用 DIP開關或軟體或其他方式，以把這塊記憶體給 CPU。由此開始，CPU 就和擁有額外記憶體，或一般擴充糟的多工卡或記憶卡一般地擁有這塊記憶體。

但在 4.0版下，當回填的記憶體被整個完成時 (如 384K或可能更多),這塊記憶體只是以輪迴的方式暫時 "借" 給系統而已 —— 就如同我們用過旋轉餐桌的比喻一樣，這塊記憶體仍爲擴充記憶體管理器所有，而非 CPU。所以，因爲只是借的，是暫時分配，故可隨時取回或交換整個 384K 或任何數量的記憶體。

因此，基本上大約有 256K 的記憶體被鎖定，且幾乎完全由 CPU所有，CPU 必須有這塊記憶體給一些需要立即且連續使用的程式或資料，例如 DOS這種不能換出，否則電腦馬上當機的程式。不過，在此位址以上的任何東西都可以自由地來去，即使是一大塊正在執行的程式碼及資料也可以來回穿梭。

這在多工處理的情況下是特別重要的，這裡有一個例子：使用回填記憶體，一些今天的擴充記憶體硬體可以支援在背景中的資料通訊 (如傳眞機或數據機) 至少達到 9600 baud ,這是一個在目前以其他方法所無法達到的數字。

即使在以前，我們也已經有了一些工作，不適合放在傳統記憶體中進行，多少需要一些額外功能。這也就是爲什有

些程式廣泛使用覆疊 (通常是以 .OVL 結尾的檔案)。 覆疊就是可用的記憶體不敷使用的情況下，程式把一部分需要用到的程式碼由磁碟拷入記憶體然後使用，當需要其他部分時，便拷入另一部分覆疊程式，並將原來的部分蓋過，被蓋過的部分也不必存起來，下次需要時，只要再從磁碟中叫出即可，如此一直到完成爲止。

覆疊需要大量的磁碟存取。磁碟動作的速度緩慢 (與 RAM 的速度比起來),在很多情況下變成一個嚴重的問題且不切實際，最後每個人可能都難以忍受。

使用輪迴式的回填記憶體以取代固定記憶體，可以使整個程式及所有額外的資料同時載入，最大可用的數量就是整個記憶體資源的數量。特大型的應用程式 (例如一千萬位元組) 現在至少在理論上可能出現在我們的桌上型電腦及 DOS 上，這都要感謝回填技術。

這並不是說像一千萬位元組這種大型的套裝應用軟體很快就會泛濫，而是在說明現在我們已經可以在 DOS下做到這種程度。

5.6 新的空間

BONANZA 堃圖 (R)

雖然回填具有很大的潛力，我們仍不能爲我們 3.2版時代的舊硬體買 4.0 EMS驅動程式而想由此得到 4.0版的能力。即使一些宣稱 "支援" 4.0 EMS 規格標準的較新硬體也做不到。

事實上，不像 DOS升級時通常都能爲舊機器增加一些新

的能力，只將擴充記憶體管理器升級往往得不到什麼新的東西。但我們仍不能把這個選擇摒除在外，這在 386機器上考慮到另外一些問題時特別有用。

剛才爲什麼又强調 DOS呢？答案很簡單：沒有任何軟體能提昇一個硬體的眞實能力，但好的軟體可使一直在硬體中卻未被利用的潛力發揮出來，而 DOS就常常能做到這點，雖然只是在用到及完成一些未用的硬體功能後，讓舊機器工作情況好一些，仍十分難得。不幸的，設計來支援原來 3.2版標準的擴充卡通常沒有支援回填的硬體能力。

但即使是舊的 8088 PC都還有希望享受到回填的好處，無論我們裝設何種附加硬體，一定要能支援回填，且要能將所有在 256K 到 640K 間找到的主機板記憶體重新對映，還必須能取得控制，才能有效地將這塊記憶體換出。

5.7 不僅僅是擴充記憶體

BONANZA 璧國 (R)

我們現在有了完整的 LIM 4.0 EMS規格標準功能 (當然需要完全的硬體支援) 之後，有的不僅是可以存取到早期 LIM 規格做不到的 32 megabytes 擴充記憶體，我們還有了一套新的 640k 以下傳統記憶體的管理工具。

就如同要使用回填的能力必須裝設完全支援 LIM 4.0 規格標準的擴充記憶體產品一般，我們要充分享受使用的樂趣，仍須系統的配合。在 3.2 EMS規格標準及使用 8088 或 80286 系統的情況下，因爲這兩個 CPU沒有如 80386支援的虛擬 8086 模式多工處理，談論 "背景" 這件事幾乎沒有意義。這個限制後來被 EEMS 的新功能打破了，這些新功能也

成爲 LIM 4.0標準的一部分。

或許了解 3.2版與 4.0版擴充記憶體之間區別的最好方式，是說 3.2版便於使用資料，而 4.0版規格對執行的程式與資料俱佳。（事實上，只要每次只存取 64K 的記憶體，3.2 版並沒有說我們不可以在擴充記憶體中執行程式。不過，其主要功能仍是暫時儲存資料。）

還有更多的事情需要考慮。4.0 版規格中仍少了一些功能，例如幫助我們找到在 640K 以上一些小塊的未用位址，或取走不需要的頁框（至少現在不需要），以用來補充 640K 以下的記憶體。至此，我們已對擴充記憶體管理做了更詳細與廣泛的認識。

第六章 擴充記憶體管理

擁有能夠使用擴充記憶體的硬體是一回事，但是想要擴充式記憶體去做它該做的事，則完全是另一回事。你必須加上一種特殊的軟體，叫做擴充式記憶體管理 (Expanded Memory Manager, EMM) 的裝置驅動程式。這適用於大部份的硬體，無論該硬體的設計是原本就限制使用擴充記憶體，或者可以改成延伸記憶體 (Extended Memry) 。

大部份擴充記憶體硬體都設計成可以適用 EMM，然而這些可能不包含在你的系統中，所以你必須探討其他軟體可能提供的功能選項，特別是 80386系統，即使它們可能也需要其它的硬體輔助，甚至於 80286。

如同名字所暗示，記憶體管理程式最主要的功能便是管理。它提供了一些 DOS未提供的基本服務，並且將 DOS連接到 DOS從未用過的地方，並且不能讓 DOS知道它去了那些地方。即使是 DOS 4.0及以後的較新版本已承認擴充記憶體，但實際上仍認爲擴充式記憶體不很重要。事實上，如果你想去呼叫該支援的話， DOS也順從的不加以阻止。

的確，DOS 4.0 加入了一些滿足擴充記憶體及 EMM驅動程式的新功能，但 DOS仍只是一個 1 megabyte的系統。從以前到現在，DOS 所能做的只是讓你能存取到 1048576個位址。 DOS不管你如何使用這些空間 (除了系統啓動時 DOS自己要使用的空間之外)。 無論新的 DOS、舊的 DOS或者是類似 DOS的軟體，它們都只能存取 1 megabyte的位址 (address)。

擴充式記憶體管理程式便利用這僅有的 1 megabyte ，或者其中一部份，使得記憶體看起來不只有 1 megabyte。但是它必須在 DOS無法區別的情況下與 DOS共同工作，這就好像小孩子偷偷地作他們不該作的事，而這些事還沒嚴重到足以激起父母的怒氣。他們返復地耳語一些小秘密，曲解規定，並試探你的耐心。但只要沒有人眞正地打破規則，就不太注意。很幸運地， DOS就像這樣。

概略地說，這描述了 DOS和擴充記憶體之間的關係: DOS 是父母，而擴充記憶體就像是三千二百萬個小孩，如果你能想像這麼令人難以想像的事。無論如何，感謝擴充記憶體管理程式及那些曲解規則的巧妙技術，擴充式記憶體已經發展成一個精巧的工具。從 DOS或是任何的類似支援擴充記憶體的角度來看，恰好相反。正是擴充式記憶必須支援 DOS且受 DOS 規則限制，否則就會當機。

因爲 DOS設定這些規定，擴充記憶體必須根據一套由這些規定擴充出來的規定動作，這些規定就是擴充式記憶體管理程式。它做任何事情，都必須依照 DOS的希望去做，擴充式記憶體管理程式必須與現有的硬體及至少某些特定的版本相容。

EMM 將 DOS規則管理不到的地方挑選出來，以提供存取那些多出的記憶體。當 4.0版的規格完全實行後，它必須同時能管理傳統的記憶體，而不只是 3.2版 EMS狹隘的 64K的頁框。就像很看重自己甚而有些自滿的小孩，現在它想要得寸進尺。

我們再次强調" 完全" 實行 4.0版的規格，因爲有多得嚇人的硬體及軟體宣稱能支援 LIM 4.0版 EMS，實際上卻只提供了很少功能。這手法似乎是只要他們不是不相容就沒問題（甚至和原來的 3.0版也相容）。或許它已不通用，但卻不是不相容。當我們探討不同的驅動程式間的差異時，這些手法就可明顯的看出。

6.1 何謂裝置驅動程式?

BONANZA 璧國 (R)

裝置驅動程式起源於當電腦還相當原始的時候。當時尙沒有類似裝置驅動程式的東西，所以程式設計師必須在每個他們寫出來的程式中加入所有有關的資料，包括輸入、輸出的管理，錯誤的處理程序，以及和週邊裝置的溝通。

如果每次寫新程式都要經過上述的所有步驟，實在是愚蠢而浪費時間。結果，這些步驟逐漸簡化並演變成標準化的模組，這些模組可加入幾乎所有的程式中。

然後這些模組逐漸演化成可以從程式中分離，加入電腦的一部份。無論任何程式進來，系統都可以使用這些驅動程式。簡單地說，它們變成了作業系統的一部份。現在它們很多都已經是 DOS內建的，例如鍵盤管理程式，磁碟機驅動程

式，及其它類似驅動程式。

最好的電腦基本上是很笨的，而且外在裝置並不很多，也不會經常改變。然而世事多變，電腦變得較聰明且較複雜，外在的裝置也變多了。必須找出一種方法使裝置驅動程式獨立於系統之外，當需要的時候，可以將它們建置(install)。大部份現代的作業系統在談到裝置驅動程式時都有某種程度的彈性，而 DOS大概是提供最多彈性的作業系統了。

實際上， DOS已經將建置驅動程式的工作簡化成以機械式的方式進行（在某些情況下）。舉例來說， "將下列這一行正確地加到你的 CONFIG.SYS 檔中" 。對其它較複雜的驅動程式，可使用安裝程式 (instllation program)此程式會分析系統另外包含什麼，並且 /或根據你對螢幕上的提示或問題的回答，來幫你作安裝驅動程式的工作。後者這種作法通常在你的 CONFIG.SYS ，中加入一行新的 DEVICE=Onnnn.SYS （如果先前沒有 CONFIG.SYS ，它會建立一個新檔），並且加入各種斜線、數字、符號等等，這些是在建置過程中用來傳參數的。

驅動程式的功能很強，上述任何方法之一在大部份的例子中都能工作，所以最好能了解此猛獸的本質。當事情不是依照預期地去進行時，就會很棘手。

6.2 擴充記憶體管理程式的內部

BONANZA 全圖 (R)

擴充記憶體管理程式可分成兩個主要部份：驅動程式及管理程式。驅動程式部份模擬許多可安裝之裝置驅動程式的功能，這些裝置驅動程式可能是用以做爲使用者和實質裝置之間的介面，例如外接的磁碟機、高解析度顯示器及其它需要使用到一些 DOS沒有提供的特殊支援的裝置。這些功能包括初始化、狀態的設定以及合法的裝置標頭 -用以警告其它裝置不可侵入範圍的認明。

提拱了這些之後，擴充式記憶體管理程式的管理程式部份提供了應用軟體與硬體之間的眞正介面。相對地，這又可分成好幾個特定功能，包括：

一> 硬體和已經依功能模組化的軟體之間的區別。
一> 擴充式記憶體的 16K個區塊 (BLOCKS) 或 "頁" 的分配。
一> 將 "頁" 映射至一個或多個 64K的頁框 (PAGE FRAME)。
一> 配置擴充式記憶體或將之釋放成不存在。
一> 支援多功能的作業系統(原先 3.0版的規格中不提供) 。
一> 診斷常式 (Diagnostic routine) 。

欲使用擴充式記憶體，應用軟體中必須包含了特定的常式及功能呼叫，以便與擴充式記憶體管理程式溝通。這種交談溝通是透過第 67h號中斷，而且有一點很重要，必須記在心中，就是 DOS不只是很消極，而且不知道系統中存有擴充記憶體。

所有有關擴充記憶體的操作必須完全不讓 DOS看到。這意思是說，我們在 DOS之下操作，程式設計者必須遵守 DOS 的規則，但是卻是完全在 DOS以外的世界操作。這也就是爲

什麼檢查擴充式記憶體並將之視爲分離的個體是那麼重要的原因了。

6.3 記憶體管理的程度

BONANZA 登國 (R)

有很多必須做的選擇，每一種擴充式記憶體板都伴隨著很多種擴充式記憶體管理程式。 DOS本身現在也配備兩組擴充式記憶體管理程式，一組專爲 IBM而設，另外一組爲較一般化的 MS-DOS 而設。接著又有第三個專爲 386的擴充式記憶體管理程式，它宣稱可以做得更好。比 DOS好？還是比那些做硬體的傢伙好？也許有可能，但是你必須下定決心，因爲你只能使用其中一種 (至少在大部份的時候) 。在某些情況下，你可以一直實驗，直到找到對你最好的一種。然而或許你會問 "到底有何不同？"

如同硬體的設計有所不同，不同的驅動程式運用擴充式記憶體的程度也有所不同。基本上它們一次只處理四個 16k頁的記憶體，這是因爲 EMS 3.2版在任何時候都只准許四個 16K頁在頁框中。這不單純是軟體的限制，除非所處的硬體 (包括擴充式記憶體板以及系統本身) 准許較大的使用，否則軟體也無能爲力。

這還是決定於硬體，因爲沒有一種像一般的裝置驅動程式一樣的東西，能將加入的記憶體本身適切地安裝在系統之中。

特別是在 LIM 4.0 EMS中不再限制一定是在隨機存取記憶體 (RAM)位址較高的 64K頁中。現在已經正式化的功能--由 640K 往低位址處理，且將一切都置換出來，包括正在

256K和 640K 之間執行的程式碼。此功能至少應符合任何的擴充式 /延伸式記憶體的能力，且同時聰明到足以辨認出它所在的電腦是否允許擴充延伸記憶體的存在，如果不允許則管理程式必須動點手腳使電腦誤認爲它們根本不存在。

這是個大原則，而且已經常規化的達成了。擴充式記憶體管理的另一些功能就不是這麼常規化，而且也超出了傳統記憶體管理的觀念。在某些系統上，有些擴充式記憶體管理程式甚至可減少系統支出且多留出一些 640K 以下的空間讓使用者執行應用程式。

6.4 系統的支出

BONANZA 螢圖 (R)

系統支出是讓每個人頭疼的問題，但不是像天氣一樣，因爲還是可以對它作很多處理，尤其是 386的系統。在 80286 及 8088 上也可以作，但很困難。

如果沒有系統支出的話，我們至少都有 640K 的記憶體可以執行應用程式，即使是在 DOS底下。終究，這是個不完美的世界， DOS甚至將自己載入也需要使用記憶體。事實上，電腦作任何事情都需要使用某種型態的的記憶體--唯讀記憶體、隨機存取記憶體或甚麼的。

費用的支出，就像死亡和稅賦一樣，是無法避免的。然而，即使它不是記憶體管理的直接或絕對必要的功能 (例如 LIM 4.0 EMS規格中定義的功能)，有些擴充式記憶體管理程式確實包含或者至少提供了選項，能顯著地降低系統的支出。這些額外的支出不僅由記憶體管理程式本身的功能負擔，同時由 DOS的許多功能及其它的常駐式程式 (TSR)共同負

擔。

這個詭計是將構成系統支出的成份收集起來，另找一個它們可以生存得很愉快的地方，並將它們移到那些地方。把它們搬離珍貴的 640K ，那個常因爲太小而無法執行應用程式的地方。當然，即使整個 640K 都能夠使用，也可能不夠用，看起來我們總能夠找到一些方法需要多一些的記憶體。然而，任何能降低系統支出的東西對我們都有幫助，有些較好的軟體驅動程式便有這些工具。

MEMORY MAP for RESIDENT PROGRAMS

Name	Hex Star	Hex End	Hex owner	Decimal Length	Text or Interrupt Numbers
DOS & drvrs	0987	1349		39,952	02 06 2F 40
COMMAND.COM	1349	140D	134A	3,120	2E
COMMAND.COM	140D	1418	134A	160	COMSPEC=c:\command.com/MEMO_DIR
	1418	141C	-Avl-	48	
CACHE-EN.COM	141C	1429	142A	192	COMSPEC=c:\command.com/MEMO_DIR
CACHE-EN.COM	1429	17EC	142A	15,392	09
VKETTE.COM	17EC	17F9	17FA	192	COMSPEC=c:\command.com/MEMO_DIR
VKETTE.COM	17F9	1C41	17FA	17,520	
COMMAND.COM	1C41	1C46	134A	64	/)////)/3/
DV.COM	1C46	1C52	1C53	176	COMSPEC=c:\command.com/MEMO_DIR
DV.COM	1C52	1C90	1C53	976	
DV.COM	1C90	7FB7	1C53	406,112	67 FA
	7FB7	7FC4	7FC5	192	COMSPEC=c:\command.com/MEMO_DIR
	7FC4	8088	7FC5	3,120	22 23
	8088	8093	7FC5	160	COMSPEC=C:\COMMAND.COM/PATH=C:\
	8093	0898	-Avl-	64	
DOSDEY.COM	0898	08A1	80A2	128	COMSPEC=C:\COMMAND.COM/PATH=C:\
DOSDEY.COM	08A1	818B	80A2	3,728	21
	818B	818F	7FC5	48	/
	818F	8198	-Cur-	128	
	8198	9FB7	-Cur-	123,360	
High DOS Mem					
Drives D-G	CA00	CB2E	CA01	4,816	
MODE.COM	CB2E	CB3B	D15E	192	COMSPEC=c:\command.com/MEMO_DIR
	CB3B	CBFF	-Avl-	3,120	
RAM or ROM	CBFF	D000	0E92	16,384	
Dev=CON	D000	D133	D001	4,896	
Drive E	D133	D15D	D134	656	
MODE.COM	D15D	D175	D15E	368	17
	D175	E000	-Avl-	59,552	

圖 6-1

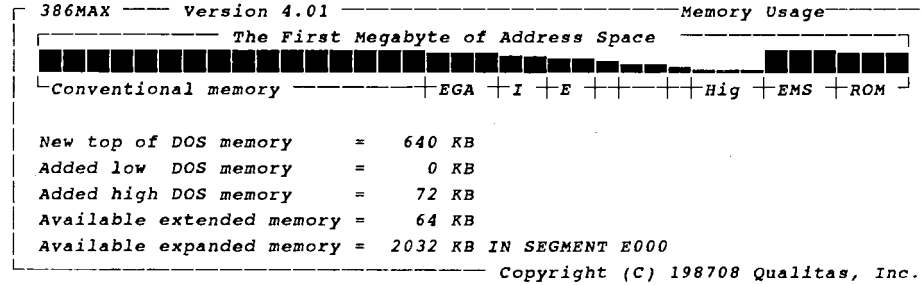
圖 6-1 圖中列出記憶體的使用，包括系統的支出（擴充式記憶體的使用並未列出）。有超過 85k的系統支出（DOS 的、常駐式程式的等等）在傳統的記憶體中。 DOS 的 MODE.COM 及一些其它的項目已被成功地載入到 640K 以上的記憶體，但仍有超過 62K的部份留著，以備使用。這一點指出我們要更努力，以將更多的項目移到 640K 以上的記憶體。本資料由 386-MAX上得來。

6.5 記憶體清道夫

BONANZA 螢圖 (R)

有少數較好的記憶體管理的程式，當你一啓動系統，在系統建置的過程中，會在電腦中看看 640K 到 1024K之間發生哪些事情。他們會去檢查看哪裏有還沒被使用到的小區塊記憶體，將它們標上記號，登記起來，並將它們的位址放入可使用的埠中 -注意我們是說位址，而不是說記憶體。此刻我們說空的，及未被使用的位址。

從這裡開始，我們會處理微小的差別，類似將綿羊從山羊群中分離出來之類的問題。在任何一部由 DOS操作之下的電腦都會有空的位址（目前我們暫且不管類似網路卡．高解析度顯示器的東西）。無論在 PC、 AT、 386 或任何 PS/2 模式的機器，這些位址均是空的。事實的關鍵在於，其中只有 384K 被影像使用（在單色顯示器所用的少至 8K），其它的都浪費掉了。




```
Extended memory usage...
  ROM mapping region   =    0 KB
  Program storage      =  136 KB
  EMS memory           = 2032 KB
  Remaining ext memory =   64 KB
  High DOS memory      =   72 KB
  Low  DOS memory      =    0 KB
Total extended memory  = 2304 KB
Total expanded memory  = 2608 KB, in use = 1600 KB, available = 1008 KB
—> Loading programs in LOW memory...
—>  62 KB available in HIGH memory, largest block is  59 KB.
The current state is ON.
```

圖 6-2

圖 6-2記憶體映射圖顯示了位址空間的使用。我特別感興趣的部份是高於 640k (A000)的區域，現在被映射至從 B000, C400 , 和 CC00 開始的區域。從 E000 開始 64K 的區域是正在使用中的頁框。被系統佔用的區域及其它唯讀記憶體，已經不能再使用，也指示在圖中。這張列表是從 386-MAX 中得來，同時指出了有 1008k的擴充式記憶體可以使用，即使此時尚未能決定它們所屬的檔案名稱。

這些不是浪費掉的記憶體，而是被浪費掉的位址空間。

在最糟的情況之下，即使你已經標示了一塊 64K的區塊給擴充式記憶體存取用仍會有 248K 之多被浪費掉。這不是相鄰的一塊 248K 的區塊，也不一定與傳統的 640K 相鄰。

當然，也許有一部份是相鄰的。例如單色顯示器的 64k 浪費掉的位址空間相鄰，若能找到一些辦法，將記憶體插入這些位址，則可使 DOS可存取的記憶體由 640K 增加至704K，若使用 CGA顯示器，則可加入 32k的相鄰空間。有些主機母板確實提供了 704k 的隨機存取記憶體，以便利用這項特性。當然，如果想使用 EGA顯示器，則必須把多出的64k 斷線，使其無法工作，因爲 EGA會使用到從 A000h開始的 64k，我們不能讓兩者同時使用同一地方。

即使使用的機器是 8088，要作這點假也是很容易的。要到達任何剩餘的位址，必須使用 80386(或者一部提供了附加硬體支援的 80286，例如提供了多功卡)，加上一點善意的宣告。我們必須重述一點，不是所有的擴充式記憶體驅動程式(甚至那些很明顯地是爲 80386系統而設計的驅動程式) 都擁有這項特殊的能力。

這個作法叫作重新映射，它相當於將存在某些地方的實際記憶體(沒有名字或未被陳述出來者，例如擴充式記憶體板) 的部份映射至這些未被使用的位址空間的一部份或全部。要找出可作用的記憶體不是件困難的工作。

這只是最初的一步，或者說兩步--找出尚未被使用而 DOS可定址的位址，並將可用的記憶體映射上去。接下來的步驟，也是最重要的一步，是使這些記憶體工作。這需要花一番功夫，因爲這些位址大部份的軟體都不使用，因爲它們總是超出限制。

```
386MAX   -- Version 4.01 -- A Memory Manager for 386 Systems
  (C) Copyright 1987-8 Qualitas, Inc.  All rights reserved.

                    EXPANDED MEMORY USAGE

Handle     Memory     Name          EMM Bersion 4.0     Frame =E000
------     ------     --------       4 Mappable Physical Pages
    0        576
    1        192
    2        256
    3         64
    5         16
    6        496
 Avail      1008
            ----
Total      2608 KB

                          EMS Page Map
      0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  1 MB
      ####▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀####▀▀##▀▀####▀▀▀▀
## Unused  -- 0000-1000  A000-B000  B800-C000  D000-E000
▀▀ DOS     -- 1000-A000
▀▀ Include -- B000-B800  C400-C800  CC00-D000
▀▀ Frame   -- E000-F000
__ ROM     -- C000-C400  C800-CC00  F000-10000
```

圖 6-3 640k 以上位址空間使用的映射圖。這張由80386-MAX 得來的圖，顯示了一個配備 2 mega 位元組擴充式記憶體的 80386系統中可用的記憶體以及位址使用的情形。在這裡，有總共 72k可用的記憶體已被定位，指出可以重新定位一個或多個程式到 640k以上的記憶體以及傳統記憶體空間空的部份。

這需要一些其它的特殊軟體，在 Desqview的QEMM 386記憶體管理中叫作 LOADHI.COM 及 LOADHI. SYS，在 Qualitas 的 386-MAX中 LOADHIGH、 LOADLOAD及 386 MAX.COM 是指令名稱。在任何一個例子中，使用這些功能會

試著將你的程式或其它軟體載入 EMM已映射好的記憶體。Qualitas的 LOADHIGH 指示在 DOS之下可與 DOS指令合用，以使用 DOS的指令 (例如 386 MAX LOADHIGH JOIN XX 等等)。你最好將之加入 AUTOEXEC.BAT 檔案中，它可幫你省下打字的時間，並避免打字錯誤 (當然你必須完全了解它的文法及含義)。

在 Desqview 的管理程式中包括了 LOADHI.COM (功能和上述的一樣) 以及 LOADHI.SYS 。你將會發現，無論在那一種管理程式中，因爲某些理由，並非所有的程式均可重新定址，但若嘗試將它重新定址也不會造成傷害。若無法載入較高位址，則會顯示適當的錯誤訊息。 Desqview 的管理程式會繼續將程式載入到較低的位址， 386-MAX也這樣作，但用較委婉且較多訊息的方式讓你知道發生了什麼事。大部份例子中，這些訊息都包括了理由。

LOADHI.SYS ，正如同它的名字所暗示，能强迫將可重新定址的裝置驅動程式載入位址較高的記憶體，畢竟裝置驅動程式只是另一種形式的常駐程式。基本上，裝置驅動程式在系統啓動時必須由 CONFIG.SYS 檔案載入，所以 LOADHI.SYS，必須在每一個所選擇的驅動程式安裝之前使用 (在 CONFIG.SYS 檔案中)。

從另一方面來看， 4.x 版以上的 MS-DOS 提供了一個裝置驅動程式 386 EMM.SYS，就功能這一點來說它與 QEMM.SYS或 386-MAX相同。這與 IBM提供的 386裝置驅動程式並不相同。

有時這些工具能夠作用，有時候不行，而有些時候，即使它們能夠作用，它們所載入高位址記憶體中的程式也不會

在那裡被執行。對於那些爲了某些理由而無法被載入高位址記憶體的程式，則自動載入低位址記憶體，但你還是試了。無論如何，這些被載入而無法在高位址記憶體執行的仍然是你的程式。你必須將它從欲載入高位址記憶體的程式名單中剔除，並重新排列。這需要花費一些試誤，在你能夠將事情做好之前。但這仍然是值得的，因爲能將越多的系統支出往高位址隨機存取記憶體填塞，意味著越多的低位址記憶體可執行應用程式。

即使在 386中 640k 以上的區域都提供使用，擴充式記憶體驅動程式 (例如 Qualitas 中的 386-MAX，以及 Quarterdeck 中的 QEMM.SYS)也會跳過高位址記憶體中的某些區域，但不一定都跳過相同的區域。例如 386 - MAX的預設值就和 QEMM 的不一樣，它保留了一整個 128k 的區塊給影像使用，而忽略了大部份 DOS機器只使用一半的事實。即使是在 EGA中，這也只用在文字狀態而非用在圖形狀態下。

視特定系統使用顯示器的情況而定，這些大部份都可以重新宣告，把它們加入可使用的記憶體中。這個作法通常是在包含了擴充式記憶體管理程式指定的 CONFIG.SYS 檔案中加入 INCLUDE=0xxxx-xxxx 的敘述。然後使用者必須將初設狀態改成適合你的特殊的建置情況。這並不會失去什麼東西，但如果你使用了軟碟開機，則會浪費一點時間，即使你以前從未嘗試過。

如果只使用了單色顯示器或 CGA，則至少可以得到保留給顯示器的記憶體中較低的 64k或 96k。這個結果比乍看之下好很多，因爲現在我們擁有從 A000h開始的一個區塊，而這個區塊正好在傳統的 640k 記憶體之上。聰明一點的驅動程式則可將這一部份加入傳統統的記憶體，而達到 704k (

若很小心處理，則可再稍微增加），這樣就可被任何的 DOS 應用程式直接使用。

若使用 EGA，代表從 A000h開始的位址已被使用，所以我們無法得到保留給影像的空間中的前 64k。也就是說 640k是傳統記憶體的限制，但可將保留給單色顯示器的空間移作他用。也就是至少可以指定 B000h -B800h， 以便讓一些可重定址的常駐程式或軟體驅動程式使用。

有一點要記住， INCLUDE這個選項並不限於指定特定的空的區塊（例如未被預設值選擇的區塊）。額外的區塊也可被拘定，但必須加在 CONFIG.SYS 檔案的同一行中。

也許某些擴充式記憶體管理程式的預先假設值會嘗試著去使用已被系統中其它硬體或軟體使用的區塊。一些有名的網路板，在這一方面是惡名昭彰的。當這種事發生時你就會知道，因爲原本行爲良好的系統開始出錯或甚至當機。萬一事情發生，你就必須決定是那些位址出狀況。在 CONFIG.SYS檔案中加入一行 EXCLUDE=0xxxx- xxxx（必須在引用驅動程式的同一行）則可將這個區塊還給需要它的人而不必介意是誰最有權利得到。

EXCLUDE選項對在系統啓動過程中待會兒會建置的其它任何裝置的軟體驅動程式特別重要。前面說過，電腦作任何事都要使用某些地方的記憶體，當擴充式記憶體管理程式安裝本身時不能知道有什麼其它的系統需要，也不會找尋任何可用的記憶體。

這只是另一處可以使用圖表或其它類似的東西來幫你節省時間或額外工作的地方。若你能將你已建置的每個新裝置

及其所需的記憶體區塊作一紀錄，則如何使用 EXCLUDE就成爲很容易的事。記住，在安裝較好的擴充式記憶體管理程式時，對於使用多少個 INCLUDE、EXCLUDE 並無實質地限制。如果所使用的硬體無論在何種情況下記憶體管理程式都會工作，則不會有害。

記住，當你在系統中使用這些不同選項作實驗時，不能對任何事造成傷害。也許最安全的作法是使用軟碟開機，直到你能確定沒有任何錯誤。

在最基本的形式中，暫時性的軟碟需要不只是 DOS的兩個隱藏系統檔案及 COMMAND.COM，它還需要一個暫時的 CONFIG.SYS檔案，其中包括舊的 CONFIG.SYS 檔案及你正要用到的安裝程式。也可加入原有的自動執行檔的拷貝，但記得第一行要將磁碟機轉換至 C。或者如下兩行的簡單形式:

```
C:
AUTOEXEC
```

這兩行在啓動時，從軟碟中讀入更改後的 CONFIG.SYS 檔案，然後切換至硬式磁碟機，以執行正常的開機自動執行檔。確認 CONFIG.SYS 檔案中的任何檔案都指定了正確的路徑，或者也可將這些檔都拷至軟碟中。

使用暫時的開機磁片，可在你時間不充裕的狀況下使用你自己的圖表，即使很久沒有接觸這件工作。在你確定它是依照你的想法在工作之前，你毋須作任何改變。在這個例子中，在你將所有浪費的位址找出之前，必須不斷地試誤。

當你更深入地探究擴充式記憶體管理程式，你會發現--如果你還不知道--並非系統中所有已建置的記憶體都以相同的速度工作。在 640k 以上的記憶體，有些差別令人驚訝。有些較聰敏的管理程式，常式化地將已經重新宣告的記憶體根據速度加以排序，並試著優先使用速度最快的記憶體，這是一個可選擇要不要的功能。

MEMORY ACCESS TIMES

Starting Address	Range Start	Range End	Length	Average Time us	Ratio to Fastest Time (1.0 = fatest)
00000000	0	640	640	570	1.0 *
000A0000	640	704	64	4231	7.4 ********
000B0000	704	736	32	570	1.0 *
000B8000	736	768	32	8514	14.9 **********
000C0000	768	784	16	4231	7.4 ********
000C4000	784	800	16	570	1.0 *
000C8000	800	808	8	4231	7.4 ********
000CA000	808	896	88	570	1.0 *
000E0000	896	960	64		Absent
000F0000	960	3120	2160	570	1.0 *
0030C000	3120	3192	72		Absent
0031E000	3192	3328	136	570	1.0 *

圖 6-4

圖 6-4由 386-MAX所得到，爲可使用的隨機存取記憶體不同區塊速度的測量。注意在這個例子中高於 640k 的部份，使用得最少的區塊其速度和傳統記憶體一樣，但有些例外。擴充式記憶體從 000F0000 開始。

這的確是一個較無用的功能，目的只是在讓某些人爲增加的速度感到驚訝而已。然而，這已經被較好的產品提供了，總之，它不是很重要。

有件很重要的事，找出記憶體的大消耗者，並盡可能地分配 640 k以上的記憶體給它。我們經常會忘記一件事，即使是 DOS的工具，或是常駐程式，一但我們叫用過則會一直留在記憶體，直到我們重新啓動系統。

看起來無害的指令如 PRINT、ASSIGN甚至 MODE 事實上都會載入常駐程式，MOUSE.COM 也是常見的一個。與其它常會想到的常駐程式如 SIDEKICK 相較之下，它們只佔用了很少的記憶體。但這就像寓言故事中那隻可憐的老駱駝：只要背上再加一根稻草就會造成很大的差異。

```
     LOADHIGH.XY

386MAX   --Version 4.01 -- A Memory Manager of 386 System
   (C) Copyright 1987-8 Qualitas, Inc.  All rights reserved.

 w> High DOS memory opened up
 w> Loading programs in HIGH memory...
 w> High DOS memory closed off.
 w> No room in high DOS memory.
 w> 15KB available: 28KB needed for COMMAND.COM, 15KB for 386MAX.COM
```

圖 6-5

圖 6-5 因爲容量不足，無法將程式載入較高位址的隨機存取記憶體。有時這幾近浪費，但仍值得一試。

有時你會覺得好像在追逐自己的尾巴，繞成一個圓圈無處可去，當你正嘗試去動用較高位址的隨機存取記憶體時。尤其是當你知道你在高位址的隨機存取記憶體中得到一未使用的 24k區塊，但使用時得到如圖 6-5 中的訊息時。轉換失敗，訊息快速地閃過螢幕，表示你期待的事並未發生，因爲最大的相鄰區塊只有 16K，而你需要多於 28k的記憶體來載入 COMMAND.COM (實際上是重新載入，因爲系統中本來就有)。另外還要 15k的記憶體來放 386MAX.COM 。

於是你搖搖頭，將 LOADHIGH 指令從開機自動執行批次檔中去掉，並浪費掉那些空間。

如同早先所述，底線在於，擴充式記憶體管理有三項主要資源，但我們只能用一種。顯然地你可以明白到，這造成了一些差異，尤其在 80386。能作超過這些的裝置驅動程式通常不是你買記憶體機板時附贈的那個，或是最新版的 DOS。這種例子太多了，所以最安全的賭注不一定是最好的賭注。

現在我們來看看硬體，特別是記憶體擴充的產品領域。我們已經知道如果小心地作選擇，我們的能力的範圍如何。

第七章　延伸記憶體

隨著 80386的日漸普及，延伸記憶體正迅速地演變成爲在桌上型電腦之中最具潛力、且最搶手的領域。這也就是某些專家之所以預測在 1991 年之前微電腦的銷售量將會領先大型電腦約百分之六十五的原因之一。爲了對抗 80386强大的功能，且由於以 OS/2 爲賴的作業系統的失敗，許多軟體的發展者已對 80286採取新的觀點。

我們將延伸記憶體比喻成一部直達屋頂的電梯（DOS 的 1MB 記憶體的頂端），它採用一種直接且線性的存取方式，和擴充記憶旋轉式的存取方式截然不同。這兩種完全不同的記憶體各自使用不同的軟體來作存取，但也有一個共通點--都使用隨機存取記憶體 RAM。

由 DOS來作存取時，延伸記憶體是一種線性的記憶體。對其它某些未受到 DOS 1MB限制的作業系統而言，延伸記憶體則被稱爲普通的傳統記憶體。若使用眞正的三十二位元作業系統，傳統記憶體可達到 4GB（於保護模式中使用 1MB以上的部份）；若在 286上使用眞正的十六位元作業系統（也

許是一些較複雜的 OS/2)，則可達到 16MB。

這並不意味著僅將延伸記憶體用來儲存及使用資料，直到最近仍有許多使用者爲這是延伸記憶體惟一適合扮演的角色。延伸記憶體不僅可以用作虛擬磁碟 (RAM disk) 、印表機列印埠 (Print Spoolers) 或磁碟快速存取 (disk cache) 之用，而且可實際用來執行應用程式；它可執行需要用到15MB或更多記憶體的應用程式。

7.1 揭開延伸記憶體的面紗

BONANZA 螢圖 (R)

若延伸記憶體眞的那麼完美，那麼爲何有這麼多的誤解呢？這個答案很簡單，而且我們也覆述很多遍了。有一本書很清楚地這麼記述：

在 DOS環境之下，延伸記憶體有個很重要的限制：延伸記憶體無法在 DOS 2.0版或 3.0版之下執行現有的程式，只能用來作虛擬磁碟、印表機列印埠或者是磁碟快速存取。

這曾經是事實，但現在卻已經改變了。只要藉由一種我們稱爲 DOS Extender 的東西幫助，幾乎所有的程式都可由 DOS 在延伸記憶體上執行。我們毋須等待一些奇怪的、或是全新的三十二位元作業系統的問世，只要使用由十六位元作業系統 (譬如 OS/2)改進而成的三十二位元作業系統即可。同時也沒有那些有關 "一次只能在延伸記憶體上執行一件事" 的荒謬限制。只要能夠找出任何記憶體 (無論是實質記憶體或者是虛擬記憶體) 來支援，那麼可隨著自己的喜好來使

用應用程式的數量。

在延伸記憶體中，你可以在 DOS底下執行用任何一種語言所寫的應用程式，包括一些可被移轉給 DOS的特殊應用程式，但這種情況則必須藉由特殊執行環境的幫助。通常這種情況需要一些轉換程式碼的工作，至於需要轉換到什麼程度，則決定於許多因素。在許多例子中已証實了這是相常容易的，至少比重寫程式容易。

和擴充記憶體相較之下，延伸記憶體似乎有較多優點。即使從 640K 往低位址回填記憶體，並在可使用的記憶體中（高於 640K 的位址）重新映射所有可能存在的位址，運用擴充記憶體的技術仍然無法使 DOS程式能使用超過 960K 的記憶體（在實際情況中通常遠不及於 960K)。

相較之下，在保護模式中，即使是 286，亦可支援多達15MB的延伸記憶體（實際模式中則是 15+1MB)，無須重新映射，也無須切換 Bank ，更無須使用那麼多的技巧（除了由保護模式調回實際模式的問題之外，例如需要 I/O服務的時候）。

最佳的好處是，你無須爲了利用延伸記憶體，而拋棄那些優良且昂貴的擴充記憶體硬體。正如我們在前一章所提及，有些較好的擴充記憶體已經成爲延伸記憶體了。我們只須在技術上讓它像擴充記憶體般作存取即可，而且有些主要設計給擴充記憶體的硬體可以很容易地轉換給延伸記憶體。至於使用者要稱它爲擴充記憶體或是延伸記憶體，就要看使用者如何管理、如何定址而稱呼。

7.2 快速複習

BONANZA 螢圖 (R)

儘管是基於不同的理由，不論對使用者或程式設計師而言，延伸記憶體及擴充記憶體的相異之處已成爲困惑的最大來源。使用者發現難以了解此二者的區別；而程式設計師的困難則在於，長久以來它們均認爲在 DOS環境下工作比使用延伸記憶體容易得多。

如圖 7-1左邊所示，延伸記憶體 (有時稱爲線性記憶體) 在 1MB之外只是一串線性連續的位址而已，與圖右邊以 64K 頁框作存取的記憶體作比較。注意，如圖所示，在延伸記憶體之中，可使用的只有 16MB ；然而在擴充記憶體中，依照 LIM 4.0 EMS的規格，則可存取到 32MB 之多，是延伸記憶體的兩倍。這必須追溯到第一章曾討論過的有關位址接腳的限制，這同時也是限制 8088 及 DOS只有 1MB的原因。在 80286中，能被定址到的延伸記憶體最多只有 16MB； 在 80386 中，則是 4GB。這是原來的兩百五十倍。

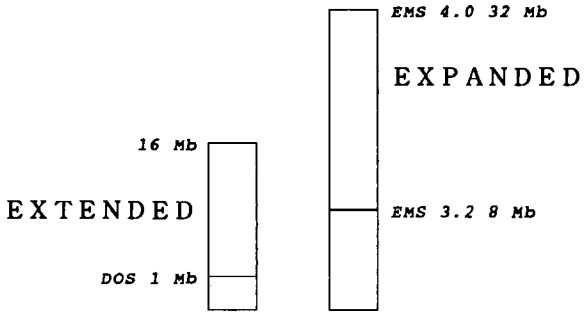


圖 7-1擴充記憶體與延伸記憶體的比較。擴充記憶體只能非直接地定址，而延伸記憶體則只是由零星位址所構成的一塊連續的線性記憶體。如圖所示者爲 16MB 的 80286系統。延

伸記憶體和擴充記憶體並不相同：擴充記憶體限制最多只有32MB，且它所能支援的應用程式必須是在正常的 DOS位址空間之下所能載入並執行的應用程式；延伸記憶體能夠支援巨大的應用程式，甚至可在 80386中支援需求高達 4GB的應用程式。

80386 的數字感覺起來較不實際，因此我們將注意力放在 80286中 16MB 延伸記憶體的限制。如同我們所知，在 DOS 下，這僅是擴充記憶體的一半。不過，注意我們在圖 7-2 中所完成的事情，我們將前一張圖中看到的擴充記憶體及延伸記憶體合併在一起，使它們在相同的 80286系統中。

這兩個截然不同的附加記憶體並不會互相排斥，現在我們將一個總共有 48MB 記憶體容量的 80286系統也放在一起。在 386中，或許不可能將這些晶片組合在一起，但我們可以想像一下那種情況。

現在我們已將任意一種附加記憶體（或者兩者同時）裝在相同的 286系統或是較 286爲佳的系統上，並且對大多數的使用者而言，16MB的記憶體已足以勝任愉快。於是便產生了一個邏輯上的問題：" 如果你尚未擁有 286、386 或加速卡，或者你正打算購買，何不乾脆買一個來裝在舊的個人電腦上？" 然而事情並非這麼容易，首先必須深入了解所謂的保護模式。

7.3 保護模式

BONANZA 螢圖 (R)

很多人喜歡有談有關保護模式的事情，但是他們所知道的，卻不比他們對其它事物的了解來得深入。

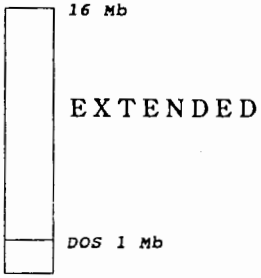


圖 7-2和擴充記憶體不一樣，延伸記憶體只是在 1024K以外的由零散位址空間組成的連續記憶體。此圖所示者爲 16MB 的限制，這只適用於 80286系統；在 80386中則可支援多達 4GB 的記憶體。

然而它的觀念實在是相當簡單的。保護模式的起源，最早是 Intel的設計師，當時他們正嘗試著實作一項保護虛擬位址（在 DOS中 1MB以外的線性位址）的項目，使得多工作業在同作時資料不會彼此破壞。

程式在實際模式或保護模式之下執行，最顯著的差別如下：

在保護模式下，段位址暫存器的內容是一個選擇器 (Selector)，而非實際位址。

這就好像使用一張替代表，利用這張表將一個可能指向任意段落或位址的呼叫攔截下來，經由一個選擇器找出程式碼或資料眞正載入的位址，重新將此呼叫定好路徑。由表中可知道程式碼或資料放在哪裏，因爲我們作的第一件事便是將相關資料放在表中。

選擇器就像是一個特務機構，譬如若是你和總統有個約會，特務機構必須先對你作檢查，之後再決是否帶你去見總統。對記憶體作存取時，選擇器提供了額外的間接等級。選擇器的內容不是記憶體中的段落的基底位址，而是表中陳述項所在的位移。

每個陳述項中，記錄了該段落的基底位址和長度，以及一些在保護模式下作記憶體保護時所需要的額外訊息。這些訊息意味著載入段位址暫存中的值，並非直接對應到實質記憶體中的位址。

當一個應用程式要存取某些特定的記憶體時（例如螢幕記憶體），首先必須將該記憶體區域的基底位址載進一個陳述項中，然後將對應至該陳述項的選擇器載入段位址暫存器。總共有兩個包含陳述項的表格可被每個作業使用，一個叫做區域陳述表（Local Descriptor Table, LPT），另一個叫作全域陳述表（Global Descriptor Table, GDT）。全域陳述表可被系統中的每一個作業使用，另外每個作業還可擁有自己的區域陳述表。全域陳述表通常映射至整個系統中的資料結構，區域陳述表則映射至各個作業各自指定的資料結構。

這當然要使用一些將此做爲內定功能的晶片及硬體結構，和 8086 及 8088 不一樣。然而它仍是相當簡單且符合邏輯的，事實上，我們必須阻止 DOS的有秩序但狹隘的範疇陷入一片全然的混沌之中。

問題在這並不產生作用，80286 就是如此。以後見之明來說，就好像設計 286之初以至於今，有人故意掉了球，於是你聽到 "界外" 的叫喊聲，並看到很多手指著不同的方向

。然而底線在於使用 286時你無法在實際模式和保護模式之間很迅速地切換，一定會使系統當掉，即使不當機，你也是走在一條瀕於毀滅的窄線上。有些專家將此歸因於 DOS本身的內在存取需求與指令使 286進出保護模式時所需的內部位址相互抵觸之故。

80386 不同的設計使每件事爲之改觀，現在程式設計師已經可以實際使用保護模式。我們甚至很難領悟到一個可能性--應用程式可以大到 15MB （在 386中還要更大），這的確很可觀。事實上，有些 OS/2 的實作可能會使用這項能力，讓實際模式的 PC/MS-DOS應用程式在 OS/2 底下（或是其它保護模式的作業系統）的保護模式中多工。

無論 OS/2 是否曾經眞的整合成一系列的作業系統，新的記憶體管理技術及 /或硬體已經整合，甚至能使很難控制的 286馴服，並給予保護模式應得的尊重。

7.4 進入: DOS EXTENDER

BONANZA 堂圃 (R)

延伸記憶體的世界 (1MB之外的部份）原本仍是一個狂野而不被馴服的領域---一種美國西部式的粗獷，導致許多有關延伸記憶體的誤解，最初印表機列印埠及磁碟快速存取使用的延伸記憶體比虛擬磁碟使用的稍多一點，然而過了不久人們便開始注意到它了。首先是 AutoCAD以及一些其它的軟體開始將資料填塞到延伸記憶體中，但仍然缺乏一套法律及秩序。若嘗試將兩組工作、兩套資料或者兩份無論是什麼的東西放在延伸記憶體中，則只有一小部份能夠生存下來。

目前最主要的差別之一，是一個稱爲 DOS Extender 的

東西。DOS Extender是一個特別的執行時期環境；這麼說吧，DOS Extender是一個迷你的作業系統，它被載入至 DOS的頂端，用以利用 DOS所放棄的地方。甚至於 Extender 的作業方式也和擴充記憶體管理程式的方式完全不同（舉例來說，Extender通常不是 "可安裝的裝置驅動程式"，而只是一個可在 DOS提示符號下載入的程式的一部份），他們各有所長，因爲它們都提供原來作業系統所缺乏的服務。

此外，DOS Extender 必須提供它本身與 DOS（一個嚴格的實際模式作業系統）之間溝通的介面。這是 DOS Extender最重要的功能之一，即使是爲 80286所寫的 Extender，也必須克服第二章中所討論的有關使處理晶片回到實際模式的困難。在這作業期間及使用多工的時候，DOS Extender會提供一個可共用的共通介面。此時有許多事必須完成。

當需要 DOS服務時，某些情況下 DOS Extender 必須處理 DOS呼叫自己。其它的情況（譬如檔案 I/O），它通常將處理器切換回 8086 的實際模式，讓 DOS處理。處理機構隨工作而不同，但觀察 DOS呼叫被處理的方式可看出目前正在作什麼工作。有時候則看不出來，因爲在保護模式之下不允許程式直接叫用 DOS。

爲避免這些錯誤情況發生，一個在保護模式下產生的典型 21h號中斷的處理如下：

1)攔截 INT 21h系統中斷呼叫。
2)將任何在延伸記憶體緩衝區中的資料移到傳統記憶體中。
3)將處理器切換成實際模式。
4)再次將中斷呼叫傳給 DOS。

5)將處理器切換回保護模式。
6)將任何資料緩衝區中的資料移回延伸記憶體。

雖然特定的特徵隨著市面上的 Extender 而不同，但大多數可從保護模式下呼叫實際模式中的常式，反之亦然。有些可直接寫在螢幕上，以省下經由 BIOS 所可能浪費掉的時間。至於支援虛擬記憶體的需求頁框，有些版本作得很好，有些則不提供。

到目前爲止，使用 DOS Extender 的主要理由之一（可能也是最主要的理由），是爲了讓應用程式能輕易地存取大量的記憶體。然而另外一個有時不予考慮的理由，是爲了有能力執行由高性能的編譯器如 MetaWare 公司出版的 High C 或其它可相提並論的軟體發展工具所開發出來的程式。

這是大規模的高性能表現，也是幾個激起人們亟欲對延伸記憶體產生興趣的推動力量之一，這些力量似乎會使 DOS 在長期之間排在一個較强而有力的地位。但 OS/2 的緩慢及漸增的不確定危機使得許多野心受到挫敗。然而對今天的軟體開發者來說，使用目前可用的延伸記憶體工具，已可以由 DOS 將 80386的三十二位元處理能力完全地加以發揮。

目前正逐漸嶄露頭角的是在許多方面可取代昔日寵兒的新一代軟體，可由 DOS在延伸記憶體中執行，不需要使用奇特的作業系統。事實上，在某些例子中，這兩套平行的新的裝置相關的軟體正逐漸地脫穎而出，儘管它們均是經由 DOS 來執行，但卻不是一定要能夠在 8088 中執行。就如同在 80386 中 DOS/ 延伸記憶體的軟體不一定要能夠在 80286中執行一樣。

透過 DOS Extender 有許多利益，撇開軟體開發者或使三者無須等待新的作業系統被開發出來的這一點不談，它是三DOS 來完成，這意味著一般任何爲 DOS環境而開發出的軟體都可以正常地運作，而不會受到 DOS Extender 的影響。這表示不會出現像 OS/2 的 "相容工具箱 (Compatibility Box)" 的花招，亦無須去等待它的出現。

7.5 對各種突發問題的套裝解答

BONANZA 璽圖 (R)

對軟體開發者而言，開發延伸記憶逐漸變成容易且廉價的事情；有些軟體開發者 (例如像 Oracle)，已經踏上自己的路，開發出屬於他們自己專利的 DOS Extender 。然而，你可以購買現成的 DOS Extender，這些 Extender中包含有顧容使用模組的函式庫，以及一些連結程式和偵錯程式，使得不論是開發新產品、或者是加强現有的產品，在突破限制時較不痛苦。

事實上，有些程式設計師宣稱這樣做比試著讓產品升級並在 OS/2 下執行來得迅速且容易多了。更進一步來說，原先因爲受到段落及大小的限制而無法在 DOS中執行的程式，現在都可以傳給 DOS執行。在很多例子中，簡化 EMS管理程式的程式碼，並將它重新編譯。這麼做是必要的，可在使用十六位元或三十二位元全部的微處理器的延伸記憶體中，提高速度以及工作效能。通常在 80386中可比 80286快二至三倍，至於比起 8086 或8088就不可以道里計了。這個套裝解答似乎大力支持這個說法。

在討論到 DOS Extender 發展的工具時，已逐漸發展出三種不同的名稱: AI Architects , Rational Systems以及Phar Lap軟體，這是分別由三家不同公司所開發出來的現成工具。在目前現有的三十二位元執行環境之中，AI Architect的 OS/386 似乎是提供了與 DOS和 BIOS 最接近的模擬工具，除非你使用了三十二位元暫存器，否則這套工具幾乎可以支援所有的 DOS呼叫，而且有其優點。

Borland 公司在他們決定加強一個叫作 Paradox 386的版本時，開始發展 Phar Lap 。順帶一提，這個版本比他們原先的十六位元版本快五倍。

Lotus 公司將他們的名字署名在 LIM EMS (過去的 3.1版並未支援) 上面之後，將他們的 3.0版本投入 DOS Extender的熱潮之中，同時提供了一個改良過的 2.X版本給現有的 PC 使用者顧客群。新的產品則配合 Rational System的 DOS Extender 一起使用，定位給 80286等級或更高級的系統。

目前 IBM也開始促銷環繞著 Phar Lap 而發展的軟體--Interleaf Publisher 。這個桌上排版軟體最吸引人的一點在於它是針對舊版的 DOS 3.3版而設計的，而不是讓此軟體在 OS/2 底下執行。這似乎是對 OS/2 的前途投下了不信任票。

Phar Lap，至少截至目前爲止，均致力於 386領域的開發。然而其它的軟體公司 (最明顯的如 AI Architect 及 Rational System)，都提供在十六位元及三十二位元的執行時期環境下均可以執行的 DOS Extender 發展工具。有許多使用者使用 CAD、CAE 以及大量科技的應用程式，這些程式

有數百個，多到足以强調 DOS Extender 對重大的工作來說是非常重要的軟體工具。

使用者在幾種較有名的 DOS Extender 中執行程式的附加檔名，和在我們所熟悉的 DOS底下可執行程式的附加檔名(.EXE 及 .COM)並不相同。 AI Architect 保護模式的程式的附加檔名是 .EXE ，Phar Lap很不幸地，選擇的附加檔名和在三十二位元模式中用它們的 Extender 所產生的檔案附加檔名相同。這沒甚麼不好，除了下列這一點: Phar Lap檔案的結構不一樣。但是 AI Architect 可藉由呼叫 .PLX 來執行 Phar Lap 的 .EXP 檔。

姑且不論在內部是如何處理，從外表看來，每件事都和以往一樣由 DOS來完成工作。每件事情都依照你的想法進行，即使你是由 Quartrdeck 公司前推出的辦公室系統中呼叫類似 Desqview 的控制程式也不例外。

7.6 何謂 VCPI (虛擬控制程式介面)？

BONANZA 璽圖 (R)

擁有能夠寫出可工作的 Extender 的技術只是上述解答的前半段而已。問題的核心不在於目前已嶄露頭角的不同的DOS Extender，亦不在於保護模式下或在控制程式中使用的DOS Extender應用程式。最終的問題包括微處理器模式的切換、硬體中斷的程序、以及延伸記憶體的共用。

衝突在於使用 386的虛擬 8086 模式的程式 (例如說 Desqview，它用自己的記憶體，包括擴充記憶體，爲每個載

入其中的應用程式--至少是爲每個視窗--建立一個分離的虛擬 8086 機器）旁邊保護模式中執行的程式，上述情形使用了其中的兩種，如果沒有某種介面來協調這些爭議，執行保護模式下的應用模式之前必須先將控制程式關掉。事情就是這麼簡單--或者說複雜，視你的觀點而定。

舉例來說，Borland 公司的 Paradox 386裏上 Phar Lap 386/DOS Extender的外衣，本來不能在 Dsgview底下執行。Quarterdck (Desqview) 卻花費了一番功夫，制定出一套彼此可接受的規則，使這兩個衝突的環境可相容。除了 Quarterdeck 以外，其它的初始發起者 (該產品後來以虛擬控制程式介面 VCPI 聞名) 尚有 Phar Lap 軟體公司、AI Architects公司、Quardram公司、Qualitas公司以及 Rational Systems公司。我們該慶幸它們並未嘗試著像 LIM 一樣組成下面這個怪名字--QOSPLASIAIQIAIQIRSI。

虛擬控制程式介面是一個重要的腳步，因爲這是第一次將所有曾經對該計劃有過腹案的公司的經驗組合起來，從而產生出來的一套合理的指導方針。此指導方針的內容如下：

任何符合虛擬控制程式介面標準的程式，均可與在同一機器下 (除了 8086 虛擬模式) 多工執行的程式共存。

結果皆大歡喜，透過這個標準，經過一些修改以配合彼此的需要，使用虛擬控制程式介面，Paradox 386 已可在 Desqview之下良好地執行。

然而，擁有一套標準和要每個人接受這套標準是兩回事。Microsoft 公司直到作者寫此書時尚未接受這個標準，導致於DOS Extender程式之間 (包括像 Lotus以及 Windows這

麼有名者) 暫時的不相容。此問題並非無去預料，但延伸記憶體對 Windows而言是那麼地重要，以至於長久以來局限了它的眼光。

最近有很多程式是在延伸記憶體中執行，而且越來越多，使得名單越來越長。對 DOS老舊的 1MB記憶體的障礙來說，名單上的每一個名字無疑地又是一記重擊。

7.7 LOTUS 將之整合--就像 1-2-3一樣

BONANZA 璽圖 (R)

LOTUS 是最早在 80286或更高級的系統上提供 DOS Extender版本的幾個應用程式之一。3.0 版是環繞著 Rational System 公司的十六位元 Extender 的技術而發展的，甚至可以如圖 7-3一般在 1MB的記憶體中執行。由此圖及下一張圖所顯示，我們可以看出哪些是延伸記憶體可以做、而擴充記憶體所不能做的事情。

注意這裡的 1MB並非單純是 8088 形式的 1MB位址空間，而是典型的 286或更高級系統上裝置的 1MB記憶體。較高位址的 384K 實際上是延伸記憶體，此處作爲工作區，用以連接較低位址的 640K 中可使用的所有資料。擴充記憶體也可以使用，但是主要是用來儲存資料。

LOTUS 3.8 ，是一個很大的程式，它無法擠在一個程式中 (至少如果考慮到 DOS時是如此)，因此它被分成幾個覆疊 (Overlay)，當需要的時候才從磁碟中載入。當我們要作某件工作時，一定要先完成載入覆疊的工作，雖然不太方便

，但是當程式太大以致於不能在有限的記憶體空間執行時，這是大家最熟悉的作法。注意在圖 7-4中做去將如何改變。

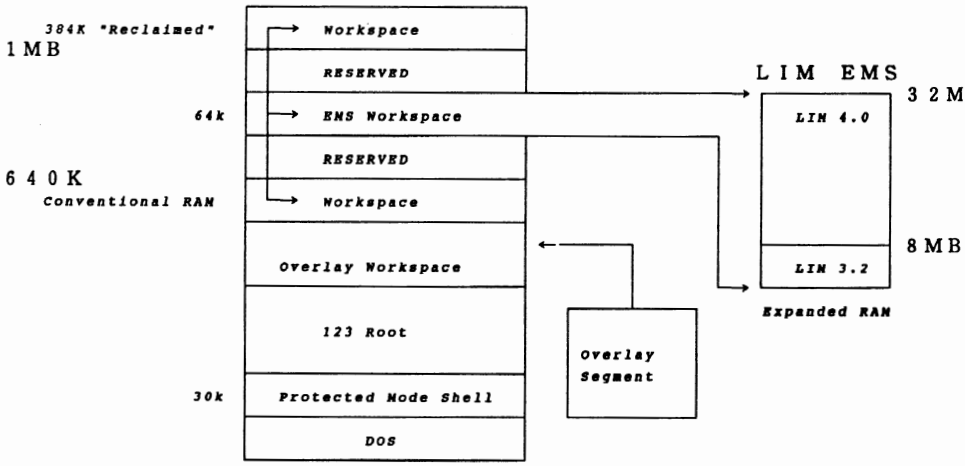


圖 7-3。DOS Extender的 LOTUS 3强調它如何作修改以配合只有 1MB記憶體以及擴充記憶體的 80286 (或更高級的) 系統。注意在需要時把覆疊載入的這個作法明顯地降低了許多動作的速度。標示著 "reclaimed"的部份是很多電腦上用以補滿 1MB記憶體之用，其中有 384K 是在 1MB以外的。

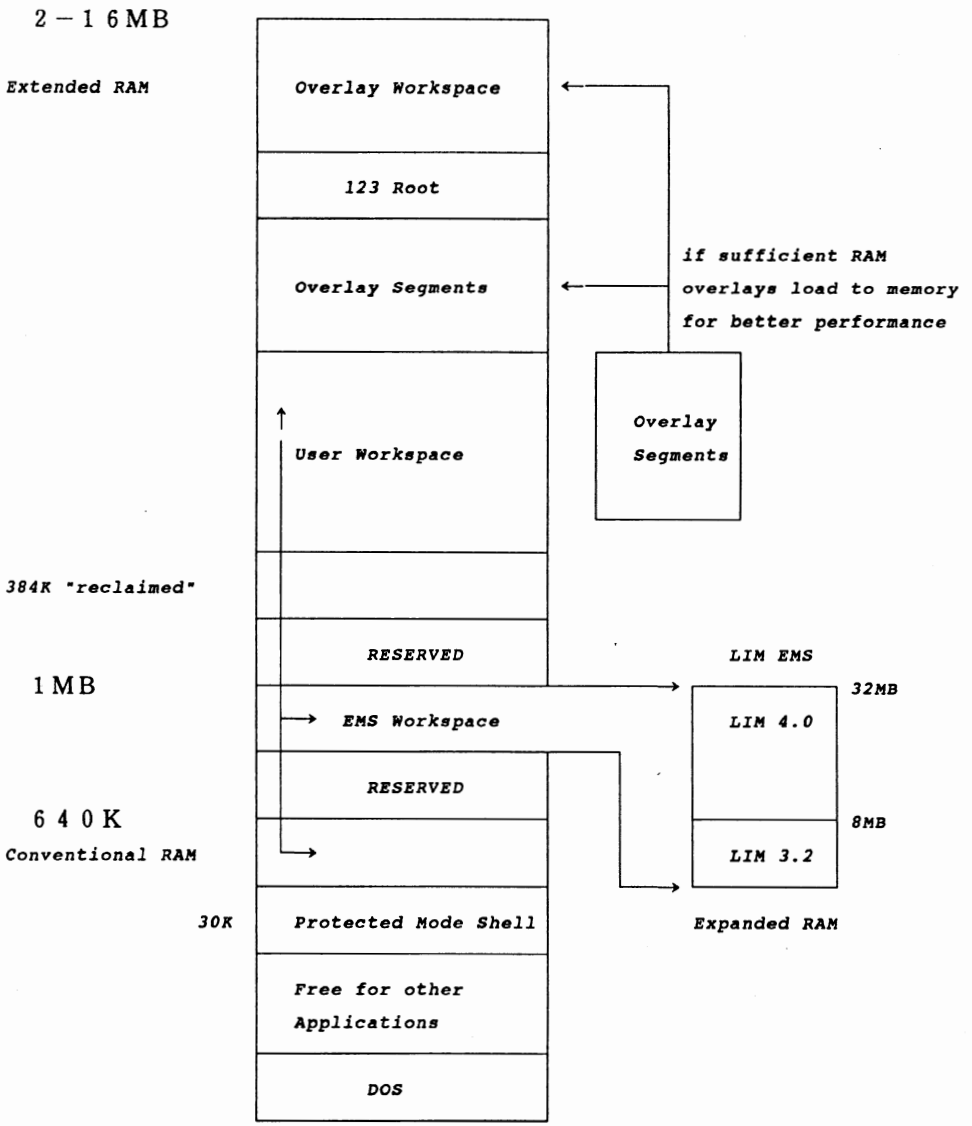


圖 7-4。LOTUS 3 在延伸記憶體環境中工作的情況。供應足夠的延伸記憶體，則所有的覆疊均將被載入隨機存取記憶體中，降低了重新載入以及反覆地磁碟存取的需求。雖然使用

了延伸記憶體，但擴充記憶體仍繼續用來作爲工作空間。

現在，決定性的因素在於你可以使用的延伸記憶體數量到底有多少，而不是你總共有多少延伸記憶體。如果可使用的延伸記憶體足夠的話，注意 1-2-3的 root 或監督程式不再放置於傳統記憶體中，它以及任何覆疊 (只要延伸記憶體的容量允許的話)，都會被載入延伸記憶體中。這意味著我們毋須時時到磁碟中尋找另一個函式的覆疊，它們都已在記憶體中，可以以隨機存取記憶體的速度加以使用。同時延伸記憶體中還有許多工作空間，提供了足夠的隨機存取記憶體。

至於資料，在 LIM 4.0中仍有高達 32MB 的擴充記憶體可供存放。在 3.0版中，仍然有完整的 32MB，但不是古老的 3.1版型式的，由擴充記憶體去存取由 2.1版支援的 LOTUS。總共可以得到的記憶體在 80286中高達 48MB。擴充記憶體仍繼續使用，但扮演支援的角色，因爲只有延伸記憶體能夠提供所需要的環境，毋須作 bank 切換，亦毋須等待覆疊的載入。事實上，在使用到大的試算表時，擴充記憶體反而會成爲瓶頸。

在此處我們以 LOTUS作爲例子，並非因爲是惟一的，而是因爲它是最典型的代表。我們看到了延伸記憶體的誘惑，這也是爲何有越來越多的發展者將未來的希望寄託在延伸記憶體身上的原因。

LOTUS 公司進入 DOS加强版的競技場，也必須注意到其它公司所發展出來的成果：在本章中稍早我們曾經討論的虛擬控制程式介面。LOTUS 使用的 DOS Extender 與虛擬控制程式介面一致，這保証了它與 Desqview 的相容性以及它在

延伸記憶體中 DOS應用程式旁處理多工的能力。

正如我們已經指出的，這需要非常謹慎的介面，因爲 80286 必須經常在實際模式與保護模式之間作切換。然而，Microsoft 的 Window 在 LOTUS公佈了 3.0版之時並未支援虛擬控制程式介面，同時拒絕爲他們的行爲做解釋，這使得未來各版本之間的相容性埋下了問題的根源，唯有等待時間來解決了。

在結束有關 LOTUS的討論之前，我們很迅速地將注意力移轉至 3.0版的另一個特性：與 OS/2 的相容能力，3.0 版並非一定要在 DOS底下才能執行。雖然 DOS是本書的主題，但一些簡短的旁白仍是需要的。在圖 7-5中可看到，在 OS/2之下，3.0 版會在一個我們稱之爲延伸記憶體的線性記憶體中執行，它使用了一個虛擬記憶體管理程式作爲置換區，以取代擴充記憶體。無論使用什麼名字，延伸記憶體仍是動作實際發生的地方。

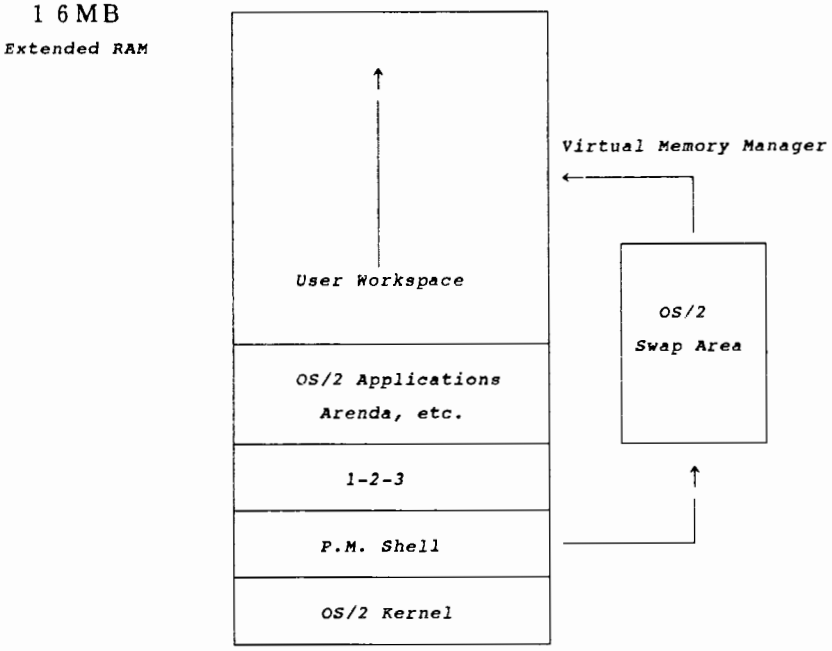


圖 7-5。LOTUS 3 在 OS/2 底下執行時將會發生此種情形。在此環境中，LOTUS 3 使用一個線性記憶體爲工作空間（最多可達 16K，80286 的限制），以得到更快的存取動作。當隨機存取記憶體不敷所需時，LOTUS 便使用磁碟作爲虛擬的記憶體置換區。

7.8 DOS 底下額外的 64K

BONANZA 發 (R)

大家都知道 DOS僅能定址 1MB，對吧？這幾乎不算是個秘密，我們以前也曾提過。然而，就如同所有的規則一般，總是會有例外：有人發現它所意味的限制僅是你不能在超過 1MB 的地方啓用一個 64K長度的 DOS程式段落。若是你的處

理晶片有足夠的定址接腳，且有延伸記憶體的話（就此二個條件而言，8088已經被判出局了），便可以不違背 DOS 1MB 的限制，啓用一段 64K長的 DOS程式段落，實際上卻讓它在 1MB 之外執行（幾乎整個 64K都在 1MB之外）。

最初是 Desqview 使這個事實曝光--Desqview寫了一個叫作 QEXT.SYS 的裝置驅動程式，用以將它們的程式碼載入 1MB 以上的記憶體--這個免費的記憶體區塊最後被列入由 Microsoft 公司公開發表的延伸記憶體規格之中。它就在那裏，合法且能起作用。你所需要的配備只是一台 286或 386，以及在 1MB以上裝置的線性記憶體；你需要延伸記憶體，但不是一般的延伸記憶體，它必須能被 DOS使用。接著 Microsoft 發表了一個和 Desqview 的 QEXT.SYS 非常相似的裝置驅動程式，Microsoft 稱之爲 HIMEM.SYS，並將其描述成一個延伸記憶體管理程式。

就功能來說，QEXT.SYS和 HIMEM.SYS看起來似乎一樣。此二者均必須被列爲 CONFIG.SYS檔案中的第一個裝置驅動程式，即使不這樣，也必須是第一個需要存取延伸記憶體的裝置驅動程式。理由很簡單：爲了被當作 DOS中啓用的程式碼段落的一部份來使用，它必須緊鄰著延伸記憶體中的部份。這個規則倒是沒有例外。

Desqview的 QEXT 驅動程式將一段 64K的程式碼段落設定在由 FFFEh- 開始的位置上，只有 16 位元組在 DOS的 1MB 之中。使得他們得到了額外的、幾乎 64K長的延伸記憶體，總共只比 1088K少一點點（1024+64），這簡直比撿到錢更棒。

當然這必須花費一番手腳，如果你還記得的話，80286

及 386有一顆叫做 A20閘的晶片，用以檢視超出 1MB的呼叫，並將其歸零--就像有一段時期通常由程式設計員所完成的事情。每個超過 1MB的合法呼叫首先開啓 A20，結束時再悄悄關閉，小心可別當機。

身爲使用者，這個意外收穫是否意味著什麼事呢？這意味著經由 QEXT 的些許幫助，Deesqview 將完全地把程式碼全部放在 286系統中的 640K 以上。所以，正當你認爲你已完全了解延伸記憶體時，又有別人找到了更新的使用延伸記憶體的方法。

讓我們面對現實吧！延伸記憶體太完美、太誘惑了，以致於我們無法坐視它閒置而不被使用、未充份發揮其功能、或是長期地被浪費掉。另外還有 4GB；爲什麼突然之間我們會覺得我們才剛起步呢？

第八章　硬體方面的關連性

既然我們已經學過了一些基本概念而且也知道將要繼續探討的內容，此刻正是設法深入研究的時候。很明顯的，若只是帶著你的支票簿到電腦商店去閒逛一下，比起這本書所能給你的東西要少得多了。（如果你不想只是隨隨便便地閱讀這本書的話。）

我們首先假設你想要把延伸 (extended) 或者擴充 (expanded)記憶體加入某一個系統內。因爲我們知道大部份的情形下，這兩種板子都能提供你相同的需求。

LIM 3.2 和 4.0 EMS 的軟體設計規格 (specifications)之間的差別以及它們的內容都已經在前幾章討論過，所以在此也不必再重覆說明。然而，要隨時記得許多廠商仍會賣給你能夠 "支援" 4.0 設計規格的板子，而這些板子實在是只能說是不會 "不相容" 而已。

本章主要是從記憶體擴充板的觀點來討論硬體架構，但是每當有新的電腦系統產生，此爭論將會更激烈。擴充產品

只能增加基礎系統的能力，卻不能免除設計上的缺陷。

很不幸的，並沒有 4.0 EMS的硬體設計規格存在。就好像不是所有的電腦製造商都允許你把傳統的記憶體當作擴充記憶體堆 (expanded memory pool) 的一部分而任意地移入移出一樣，並不是所有的記憶體電路板都提供你所須的硬體支援。你不能更改已在工廠接好的電腦線路，而且也不能更改記憶體母板連接在電腦內的方式。這尤其適合於對 640K 以下的傳統記憶體作記憶庫交換 (bank switching)。

依估計而言，今日大約有 90%的電腦至少可做記憶庫交換到 256K 。 其餘的一，二種可到 128K，另有一些可允許記憶庫交換到 0K。 然而，我們需要達到 0K 的地步嗎 ?答案是否定的。很明顯的，這樣雖不會造成任何損失，但是即使是製造機器或者安裝母板的人能夠作記憶庫交換達到 0K，這些人也不能給你充分的理由來說明爲何要做到這麼低的地步。

當 AST首先提出他們的 EEMS 架構時，沒有人 (甚至連 AST 的人) 知道他們可以處理記憶體擴充的問題。就像許多偉大的發明一樣，由於太早問世而必須等待某些東西的來臨才能指引出它的用處。實際上仍存在著一些限制。例如，你必須把作業系統 (operation system) 載入到 CPU便於操縱的地方。也許將來有一天 DOS會把自己載入而超過目前640K 的記憶體，而把你的應用程式載入到 640K 以下的部分。

很幸運的，當你到外面買記憶體母板時，不須要指定可以達到某一程度。通常會有一個底部起始位址 (bottom starting address)，可依照你的情況來設定你的記憶體母板。你所要確定的只是某一個板子的最低基底位址正是你想

要的就可以了。然後你只要把板子設成你的系統可取得的最低位址即可，如圖 8-1 所示。

基底位址:

KILOBYTES	HEXADECIMAL
00K	0000H
64K	1000H
128K	2000H
192K	3000H
256K	4000H
320K	5000H
384K	6000H
448K	7000H
512K	8000H
576K	9000H
640K	A000H
704K	B000H
768K	C000H
832K	D000H
896K	E000H
960K	F000H

圖8-1 基底位址一般都設爲 64K的整數倍數。這個表把基底位址以 16 進位的形式印出，同時也列出相對應的千位元數 (KILOBYTES)。

在任一情形下，擴充板的基底位址必須使得 CPU所看到的記憶體是一個連續的區塊。舉例來說，如果你要往回填寫到 256K 的資料，則系統記憶體就不能只到 128K 。當然僅有少數的電腦可允許填寫那麼多的資料。如此一來你當然希望有一個至少可以到 256K 的母板 — 如果你的系統允許的

話。如果不會花費你太多的錢是無關緊要，但是如果比別的產品高得太多，也就不需要了。

不管怎樣，要買一個記憶體母板來作爲擴充記憶體，要考慮的最重要因素之一就是有多少和什麼種類的暫存器(registers)。 以下就來討論。

8.1 暫存器 (registers)

隨著 LIM 4.0的來臨，以前被認爲理所當然的某個東西突然間變得新奇而又重要。它們被稱爲對應暫存器(mapping registers) 暫存器的功能是指向記憶體堆(memory pool) 中的某一 16K 的區塊。通常此記憶體堆可包含 32 megabytes 的擴充記憶體。記憶體就是以此對應方式來截取呼叫到 DOS可接受的位址 (低於 1 megabyte)然後再指向記憶體堆中的某個已被暫時指定來回應這些呼叫的區塊。

擴充記憶體的關鍵就是：截取所需要的位址然後重新指向 DOS所無法達到的地方。否則就不接受此位址需求(address request)。 這就好像打電話到你家卻沒有人在家，但是還是可以在別的地方找到你一樣。

所以擴充記憶體管理器 (expanded memory manager，簡稱 EMM) 要作的第一件事是先要找到一個可用的記憶體區塊。做完後，EMM 還必須知道此區塊的位置以便於下次要使用該區塊時能夠馬上找到。於是暫存器就用上了：每一暫存器對應到某一 16K區塊的基底位址，然後再把要呼叫這些區塊的對應位址對應到只有它自己才知道的擴充記憶體中的某

處區塊。

在早期的 3.2 EMS計畫書內只有一個 64K的頁框 (page frame)可使用，其中包含了四個 16K的分頁 (page)。 如圖 8-2 所描繪的。如果每一個分頁需要一個指標或暫存器，那麼在此便需要 4個。

一組暫存器 (在原來的 3.2 EMS版本爲 4個) 形成一個暫存器集 (register set)。 一個暫存器集可以指向擴充記憶體內的**某個** 16K的區塊。如果你想在不同的程式中讀取擴充記憶體，**甚至**在相同的程式內讀取不同區塊的資料，則你必須至少有一個如圖8-3 所示的交替暫存器集 (alternate register set)。

請注意本例中的各暫存器彼此間互相交錯。這是因爲即使在軟體上有一組規則來決定這些區塊如何來選擇，但是如果你詳看其內部，仍會顯得相當凌亂。只要暫存器集能夠保持它的型別、位置和用途，即使彼此交錯也沒有關係。

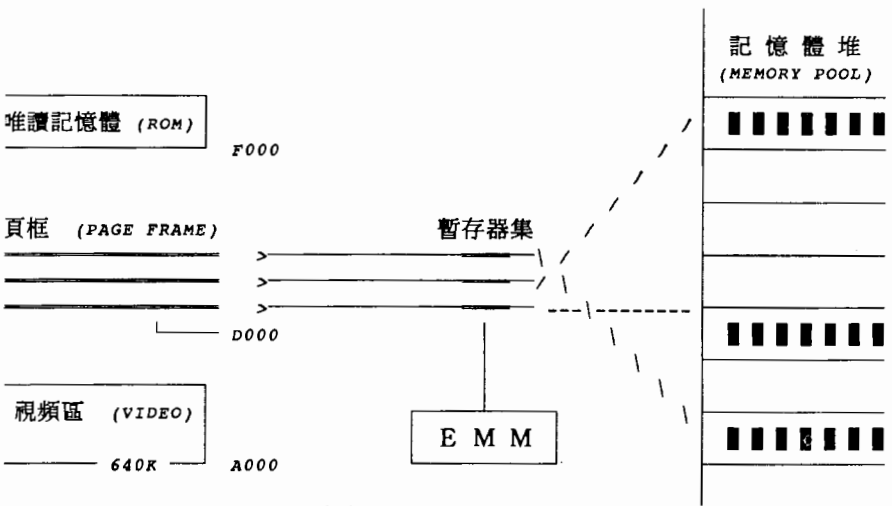


圖 8-2

圖8-2 我們可以看到頁框 (被分割成四個 16K的分頁)，由擴充記憶體管理器 (EMM)所控制的暫存器集和記憶體堆中由各個暫存器所指的 16K的區塊三者間的關係。特別注意被定址的區塊並非連續的。爲了簡單起見，這裡只使用 4個暫存器來作說明 — 此正是舊版 3.2中所需要的。而大部分支援 4.0 版計畫書的母板把每一暫存器集的暫存器數目增加到64個以便對應到傳統記憶體位址空間。

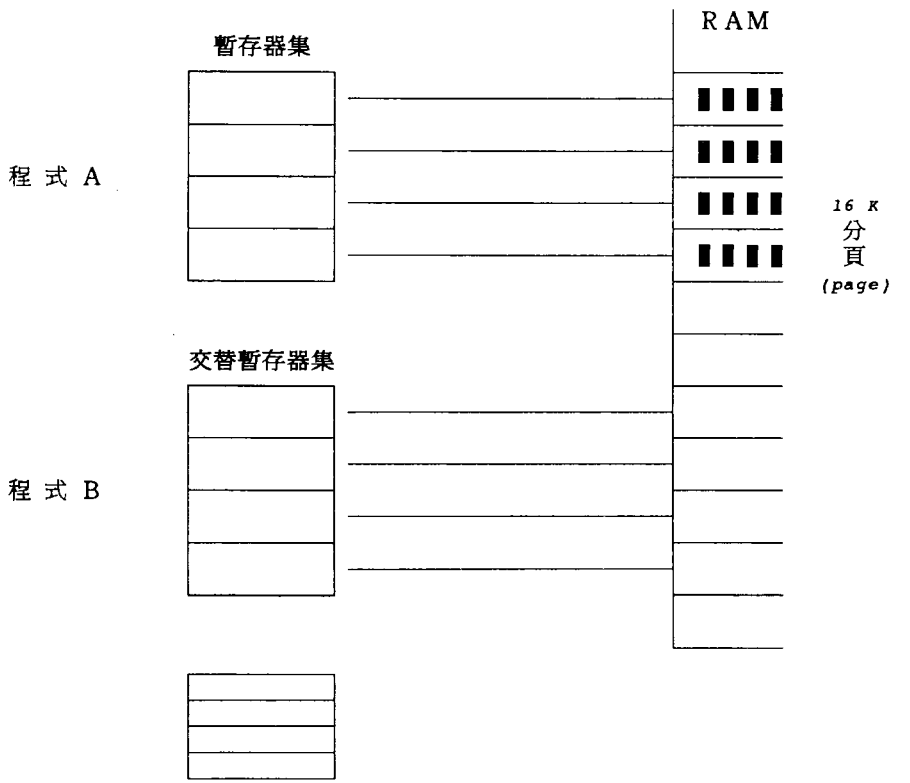


圖 8-3

圖8-3 這裡說明交替暫存器集的用法。程式 A所用的暫存器集指向四個 16K的分頁，但是程式 B使用一個已經載入的交替暫存器集而且可以立即對應到另外一組分頁。同時還準備了第三個暫存器集以供所需。同樣的爲了說明方便，每個暫存器集只用 4個暫存器。然而在實際的運作中，4.0 EMS 使用 64 個暫存器。

到目前爲止，我們仍在討論過時的 LIM 4.0 EMS設計規格。新版的設計提供許多早期版本中所沒有的新的記憶體管理功能。然而，市面上大約有 50%支援 LIM 4.0 EMS 設計的擴充記憶體母板，在這本書寫作期間還未支援我們在早期版本中所提過的特性。許多的擴充記憶體管理器 (EMM)的設備驅動程式 (device driver)也以同樣方式分佈。

注意：　"支援" 4.0 EMS 設計規格並不一定是可以提供這份設計規格中所有的特性和功能。它所指的是廠商並沒有做出無法配合該設計的怪異的東西來。舉例來說，IBM 公司就做出了能夠 "支援" 4.0 EMS 的某些怪異東西 — 使得完全無法與其他人的解釋相容。

不論如何，只要有擴充記憶體就會有暫存器和暫存器集(register sets)。 它們的重要性與 4.0 EMS有很大的關係。在 3.2版本中你無法作多元工作 (multitasking)。 如今藉由 EEMS 或者 4.0版 EMS就可達到。主要關鍵在於暫存器。

3.2 版和 4.0版的差別在於一個暫存器集中的暫存器數目。在 3.2版中只用了四個 16K的分頁 (page)，相當於一個 64K的頁框 (page frame)。但實際上四個暫存器顯然是不大夠。

AST 是首先把記憶體位址延伸到 DOS的 1 megabyte 位址空間內 (AST 是支持他們自己激進的 EEMS 架構, 此架構是採行基本 EMS的觀念並且加以擴展)。 爲了能以 16K的區塊來達到這個目的, 必須使用 64 個指標或暫存器 (1024/16=64)。然而在實際上有一些限制, 最重要的論點是置換 (swap in or out) 傳統的記憶體。這表示不只是一個頁框中的 4個分頁, 而且還有從 640K 以下的許多額外的分頁的問題。如果我們像其他人一樣僅僅從 640K 置換到 256K, 則每個暫存器集還需要 24 個暫存器 (384/16=24)。則在高位元記憶體中每個頁框仍需要 4個暫存器。但是在 4.0 版中, 頁框可以在 C000H 和 E000H 之間有一個基底位址--實際上有 12 個可能的基底位址 (並非 4個) 或者共需 36 個暫器。

如果你只有單色 (monochrome) 或 CGA顯示器, 你可以發現另有 64K可置換 (swappable)的傳統記憶體可使用 (從 A000H 到 AFFFH)。 如果你現在擁有 IBM PS/2 機型 50 或 60, 你就必須置換到零。確實的數字可能有變, 但是如果要實作 4.0 EMS而且把 4.0版中的結果作出來的話, 可能須要一群相當數目的暫存器。而這些數目只能用來做成一個暫存器集。因爲大部分的電腦事物都傾向於 2的次方, 而 64 似乎是一個 "好" 的數目字--是我們在 3.2版中所需要暫存器數目的 16 倍。

每個暫存器集有 64 個暫存器的問題已經解決了。那麼要多少個暫存器集呢 ?這個問題很難估算。看看工業界的情形, AST的 RAMpage 提供32 個暫存器集。而 ALL ChargeCard ——種記憶體管理卡可使 286 機型如同 386的機器 — 並不如入本身的記憶體, 但是提供 16 組暫存器集

來管理。至於你實際需要多少數目，如果你使用 Desqview 或 Windows甚至使用 Carousel 來作背景交換 (context switching)，很明顯的你每開一個視窗就至少需要一組暫存器集。當然我們不可能一次就開了 64 個視窗吧 !然而 Desqview 的預設值是 8，也可以設定得更多。

而交替暫存器集 (alternate register sets)的確實數目並不像每一組暫存器集內的暫存器數目要求的來得嚴格。並不像個別的暫存器一樣，多餘的暫存器集可以當作資料存入 RAM中。然而在其性能評估上會有很大的差異。

一個確實的硬體暫存器 (actual hardware register)是一個如 RAM晶片般實質的記憶體裝置，它是用在特殊用途上而且通常只有 EMM能隨時用它。如果每個暫存器集被存放在指定的裝置內，則在視窗間的移動 (舉例來說) 就只是選擇一個使用中的交替暫存器集同時告訴它去做該做的事。所有的指標都已在定位上，所需要的只是加以驅動即可。此步驟在表面上看來是瞬間發生的。

然而實際的指標必須以硬體裝置來完成。如果只有一組硬體暫存器集 (hardware register set)可用，則每當你要做移動時，有兩件事情會發生：你必須把眞正的暫存器內容拷貝到記憶體中存起來 (每一暫存器要用到幾個輸出指令和一些資料)，然後把另一個模擬 (emulated) 的暫存器拷到眞正的暫存器內，如此每次你就必須把另一個 16K的區塊置換出來。

在 3.2版中，每個暫存器集只要 4個暫存器來處理 4個分頁。而花在拷貝暫存器資料的時間是微不足道的，同時你只有在置換資料而非在執行程式。然而使用 64 個暫存器就

相對的更慢了。每次你必須載入和載出資料就比使它們先歸於定位要慢多了。它們之間的差別往往在於微秒 (百萬分之一) 和毫秒 (千分之一) 的差別，而其效率也相差在 10 倍左右。

這些說明並不是要求你一定要有 32 或 16 組暫存器集。而是說使用一些交替硬體暫存器較爲可行。而且每個暫存器集有 64 個暫存器也能吻合 4.0 EMS設計的核心和精神。在 DOS中這也是你能使用的數目。

8.2 匯流排速度 (bus speed)

另一個爭議的重點是匯流排速度 (bus speed)。記憶體母板上的記憶體並不以處理機的速度來執行。而是以匯流排速度來執行，通常只有 "時鐘速度" (clock speed) 的一半。實際上，板子上安裝了許多 "加速器" (accelator) 才會有此速度，否則其本身速度要慢得多。一般而言，就一個未修改過的系統來說，一個 25MHz的電腦可能有 12.5MHz的匯流排速度。有一個例外是 Wells American 的 14MHz 286機器的處理機和匯流排速度皆爲 14MHz。少數的擴充記憶體母板實際上可以趕上這些速度，但是有更多的母板卻失敗了。

有件事會讓我們感到懷疑，那就是在應用了最高技術 (state-of-the-art)所建造而成的系統內的匯流排速度會比廣告中所說的時鐘速度還要低一些。因爲低速度 — 能與硬體相容的速度 — 其實指的就是性能的低落。

於是，製造廠商就到了進退兩難的困境了。舉例來說，像是 Newer Technology 母板的速度就作了很大的改進。他

們曾自誇說他們的記憶體母板可與 12.5MHz甚至更高的匯流排速度相容。連他們的聚合板 (Concentration board)也可以在單獨的一塊板子上支援 32 megabytes 的晶片(chip)。

■註: 大部分的電腦的匯流排速度 (bus speed)遠低於時鐘速度 (clock speed), 而此聚合板可達到 12.5MHz 匯流排速度, 算是極快了。

時鐘速度一直在逐步的增加中, 而匯流排速度卻存在著幾個實際上的限制。例如 PC/AT匯流排 (bus)就有設計上的問題存在。早期的 PC 很少有什麼值得提出來討論的東西, 但是在現在的 PC 世界中, 自從匯流排的開放性結構被認爲是最重要的因素之一以後, 便點燃了一片希望。已經有一些人嘗試去修改 AT 匯流排 — 例如 AST就想藉由母板背面中以前未曾使用過的匯流排接點來增加一些功能。只是根本上的缺陷還是存在。

一旦其他人失敗後, 就會有另外一個有技術的設計師取而代之, 根據前人的經驗來保持和時鐘速度以及匯流排速度在功能上的相容性。他們常用的一種方法就是使用等待狀態(wait states) 來解決。還有一種 Newer Technology 母板, 通常被稱爲 attention! , 可以趕上 12.5MHz的線路速度。

■註: attention! 最引人注意的特徵之一是可以用軟體來選擇適合的時間安排 (timing) 以配合各種情形。而且 attention! 可以在線路之間加入數個等待狀態(wait states) , 如果需要的話, 也可以擴充達到 16 Mb記憶體。

attention!使用軟體來選擇時間安排的方法是: 它擁有8組軟體選擇時間安排器 (selectable timing sets), 範圍由 0到幾個等待狀態 (wait states)。事實上, 由 SCO使用 XENIX 以及 Novell 和其他人的證明可得知, 此母板的效果很好。

也許改變匯流排的設計可以解決一些問題。至少這個觀念在 IBM PS/2 電腦的 Microchannel 架構中已經實現了。Microchannel值得誇耀的一點是使用之後, 匯流排可以時鐘速度來執行。事實上 Microchannel 已經提出了幾個影響效率極大的多項論點。

但是對整個工業界而言, 既無法推廣也沒有得到什麼好處。因爲 IBM是一個具有許多獨占性特質的公司, 如此的設計才能使他們可以合理的控制一大群使用者。所以也不必期望 AT 匯流排 (AT Bus) 能很快的被取代。

還有另外一個辦法。除了 PC/AT 之外, 大部分的 386 匯流排都有一個特別專用的 32 位元的插孔 (socket) 。此插孔可以讓製造廠商設計一個 32 位元的記憶體擴充板而不受匯流排速度 (bus speed)的拘限。但是很不幸的, 到目前爲止, 這些 32 位元擴充槽 (slots)並沒有一個工業界認定的標準, 而且也因爲缺少這種標準使得沒有第三家製造商願意發展 32 位元的擴充板來配合這些擴充槽。

其實沒有第三種擴充板也不能說是什麼大問題。實際上你只會從製造電腦的人買回來 32 位元的擴充板, 而這些製造商也會設計他們自己的 32 位元匯流排。這聽起來似乎不錯而且很合乎邏輯。但是你常會發現買回來後常常不能用。這是因爲一個製造商提供一個他們自己的 32 位元擴充槽並

不保證他們也能夠提供一個能夠配合的記憶體擴充板。比如Delta Gold公司就花了一整年的時間來爲他們的第一批 386電腦製造 32 位元擴充板。如果沒有那塊板子，你還可以說你有一台 80386電腦，但是如果一提到效率的話，這台電腦就不是那麼好用了。

我們在此提到這個故事的目的並不是要讀者拿著石頭去丟那家公司，而是要說明當你買了某種電腦硬體裝置時，你所希望的是這個裝置能 "立刻" 有合理的效率產生，而不是要等到明年或任何時候才有效。

除此以外，還有一些其他的匯流排架構也被使用著。只是到目前爲止它們在市場上並沒有任何重大的影響力。所以奉勸諸位細心的購買者，還有許多因素要好好地考慮清楚。

8.3 晶片與 SIMM: 焊接式與插孔式

BONANZA 榮團 (R)

至於次要的爭論 — 但是也不能認爲是理所當然的 — 也許是記憶體本身的問題。因爲並不是所有的記憶體都做得一模一樣，比如說我們所討論過的回復記憶的速度(refresh speed)就是其中一項。擴充板上記憶體的種類也是另一個不可忽略的考慮因素之一。

由於幾個原因 — 其中之一就是錢，使得單式線上記憶體模組 (Single inline memory modules, 簡稱 SIMM)要比用個別的晶片 (chips)所做成的裝置要來得好。因爲如此，SIMM製造的費用經常低於一堆個別的晶片。

同時 SIMM 也毫無疑問地更安全而且易於處理。尤其是

對一個可能只買部分安裝好的擴充板的使用者而言，也許將來他需要更大的記憶體來補滿這個擴充板的不足。由於它們所佔的體積較大，所以也就不須要像小晶片般的靈巧。而且因爲你不必用手去接觸那些晶片纖細的 "腳" (pins)，所以也可以減少一些傷害。

當然也有人提到如果有一個焊接式 (soldered)的 SIMM 包含了某個壞的晶片，不但無法運作，而且使用者也沒有辦法修理。其實這也不是什麼大問題。如果你沒有仔細地留意的話，其實晶片損壞的頻率並不高。況且有許多的製造商有許多的修理設備來支援他們產品的使用者。

市面上有兩種 SIMM: 一種是插孔式的 (socketed)，另一種是焊接式的 (soldered)。 其實在應用上，這兩種都是可以交替使用的。但是兩者比較起來，焊接式的似乎比較值得介紹。因爲體積佔得較小，所以使用焊接式的 SIMM 可以使得許多擴充板安裝記憶體的容量加倍。

另外一個考慮因素是 :最好的選擇就是買一個能和你的硬體相容的最大容量。因爲 SIMM 的容量愈大通常也就是記憶體母板的容量愈大。而且高容量的晶片或 SIMM 通常是耗費更少的電力 (以每個 byte 來記算) 。同樣的如果以每個 byte來計算的話，1-meg SIMM的費用要比 256K 來得少。如此一來，更低的費用和電力消耗，也就產生更少的熱。既然如此，爲了更節省，爲何不使用以前舊的母板呢 ?以下要說明。

8.4　千萬不要使用舊的記憶體母板

BONANZA 螢團 (R)

我們一定會提出一個問題來 — 如果在 286或 386系統內使用舊的 8位元 PC 擴充記憶體母板會如何呢? 8 位元母板幾乎可以適合所有的系統, 而把 16 位元母板放到 IBM AT 機型上去也能夠適應得很好。

但是賣給你新的事物而以前的舊東西也能有一樣的效果並不會帶給製造商任何的好處。擴充板 (expansion board) 是設計在特殊系統內使用的 — 包括以前的 8位元和 16 位元以及最近 IBM PS/2 的 Microchannel 都是。製造這些新機器主要是要把許多舊的記憶體架構廢除掉。不僅使得 CPU 的速度提高, 也使得匯流排速度 (bus speed) 也提高。

也許你會說: "CPU 速度愈快, 當然匯流排速度也會跟著變快" 。其實事情沒有這麼簡單。在今日的電腦中, 匯流排速度極少會跟 CPU的 "時鐘速度 (clock speed)" 一樣快的。絕大部分都低於 12.5MHz, 極少有超過的。其中一個例外就是 Wells American 的 14HMz 286 AT 型電腦, 其匯流排可以 14HMz的極速執行。瞭解這些之後你應該很清楚不要爲了節省幾塊錢而在你的電腦上使用這些舊的母板。

現在你又會問了: "爲何幾乎所有舊的記憶體母板都會和加速卡 (accelerator cards)一起使用呢? " 這是一個值得回答的問題。加速卡 — 如以後要討論的 Intel Inboard 386 — 比如說可以 16MHz的速度執行微處理機 (microporcessor), 但是必須加入一連串的等待狀態

(wait states) 以把速度降低到匯流排 (bus)能處理的程度。這樣一來如果有 I/O佔用到此匯流排時將會嚴重影響其效率。這就是爲什麼使用這類的母板時比較喜歡用直接存取 (direct access) 方式的背負記憶卡 (piggyback memory card)。

等待狀態 (wait state) 簡單的說是一種使系統的某部分有喘息的機會的一種方式。它是一個非常常見的設計，只是--除了加速卡是要用來替其他硬體作決定的之外--大部分都很慢。

正如你所看到的，增加延伸 (extended) 或擴充 (expanded)記憶體要比買記憶體擴充板重要得多了。當我們要討論硬體時也存在著許多相異之處。在 LIM 4.0 EMS計畫書中就從軟體的觀點指出那些必須實現的軟體功能，和硬體部分。軟體方面也正是 LIM 4.0 EMS 所强調的重點。

8.5 選擇延伸(extended)還是擴充(expanded)記憶體

BONANZA 螢圖 (R)

有些人並不想要也不需要延伸記憶體 (extended memory)，而另有些人卻只用到擴充記憶體。時間是一直在變的，也許擴充記憶體忽然間會變成工業界的軟體發展者的最愛 — 其原因不只是新產品的開發成功，而且是功能更强大的 DOS延伸版本的產生 (特別是 386市場)。

很幸運的是，這兩種記憶體架構並不互相排斥，只是在管理記憶體的方式稍有不同。至於流行的趨勢是傾向於能把安裝好的記憶體當作延伸記憶體或者是當作擴充記憶體，甚至是二者的混合的記憶體擴充板，在大部分情形下我們並不在乎。

前面所提到過的，產品太早問世並不是一件好事。但是像 AST提出他們的 EEMS 時，要不是有人指點迷津使得他們的產品眞能做些事情的話，否則他們可能會被認爲有點精神錯亂。另外一個大膽的暴發戶，Compaq，就更加的推廣它。但是我們如果看看以前的電腦雜誌可能會發現有許多類似精神錯亂的論點被刊登出來，而現在它們多已被遺忘而且也不再受支持。

實際上，產品支持度的問題卻是許多購買者最可能忽略的論點之一。因爲再怎麼好的產品如果沒有賣方的支持也是不算什麼的。然而很不幸的就是並沒有一個公式或基準可以來衡量支持的程度，而且有時候小而沒有名氣的人可能給予最高的支持，而有時候卻又毫不支持。所以如果你看到某一產品完全合乎你的需求卻又是一間小而無名的公司所製造的，給你一個良心的建議--你最好確定賣給你東西的人是能信任的。

8.6 系統安裝

記憶體擴充板是非常容易安裝的，頂多是用到螺絲起子和一些小技巧，也不必用什麼特殊的工具。比較好的廠商都會幫你安裝任何的母板，其他硬體甚至軟體。如果你是爲了商業用途而購買的，不論是小型辦公室或者大型公司，在買

賣合約上都應該要求廠商作安裝的工作。

但是如果你是爲自己而買的，自己動手做是有很多好處的。有一天如果你的機器故障而必須做同樣裝卸的動作時，你就好像是做了一次成功的外科手術而會爲你的雙手感到驕傲。如果你動手摸過現今大部分的記憶體介面卡的話，你慢慢的也會變成專家了。

大部分的延伸/ 擴充記憶體母板能夠用在配合它們的擴充槽內。但是爲了顧及線路的問題，必須先移去幾個其他的母板 — 特別是磁碟控制器 (disk controller) 和加速板 (accelerator boards)。有些母板，例如 Intel InBoard的加速卡 (accelerator cards)就使用了相當厚平的接頭 (cable)。 由於這種接頭的限制，使得擴充槽的位置也必須事先決定好位置。

相對的，絲帶接頭 (ribbon cable) 可以讓我們非常容易地重新安排線路，因爲把連接在它們身上的母板移到其他位置似乎並不太明智。藉由這種接頭而想在線路的方向做90度的變化也是非常容易做到的。你可以慢慢的折曲這種接頭的帶狀線路 (ribbon) 到很緊的程度 (就像纏繞在你的小指頭周圍一樣，也許還要更緊) 。你可以很容易的在相鄰兩個擴充槽內的兩個電路板之間做到這件事。

但是仍然有一些記憶體擴充板使用 DIP轉切 (switches)來設定基底位址等等，而且有越來越傾向於所謂的軟體建構 (software setup)。 在最近幾年中，系統安裝簡易已經吸引更多製造商的注意力了,而且使用 DIP轉切也容易使得硬體歸於定位。舉例來說，前面所提的 Newer Technology 聚合板 (Concentration board) 中，其 DIP轉

切開關就在這塊板子的背面上方，如此一來我們就可以在電腦外殼上做設定的工作。

一般說來，一個不具特殊技術的使用者大約可在半個小時之內完成這個工作。至於相當熟悉其內部細節的人可以在10到 15 分鐘之內完成。如果使用的是 IBM的Microchannel的話，硬體安裝的操作方式基本上和 PC/AT 相同，但是其建構 (setup)則完全交由軟體而且幾乎是自動完成的。你只需要把建構磁片 (setup disk) 放入磁碟機中，然後再鍵入某個指令來啓動就可以了。

以上所講的就是讓你在前往 640K 以外的領域中走得較爲踏實；這也足以讓你應付延伸或擴充記憶體了。在下一章中我們將探討一些其他能增加效率的改進方法。如果你現在使用的是 8088 或者是 80286，這些方法也正是你想要知道的。無論如何，關鍵還是在記憶體本身。如果沒有了它，其餘的東西都會變得不切實際。

第九章　品質升級的抉擇

將來有一天，大部分電腦的使用者會面臨到提升他們整個系統品質的問題。以電腦的需求，技術和心態來說，爲了表現已達到相當的成熟度，這是一個非常正常而且必須的階段。所以現在要講的，不是提升品質的需要，而是當增加記憶體仍不足以解決問題時，要實行三種可能途徑中的那一個？以下三者我們要選擇其中一個：

=> 藉由加速卡 (accelerator card) 來做部分的系統移植；使用相同的母板 (mother board)，相同的硬體，或者是

=> 換掉整個母板，否則的話就保留所有或部分原來的硬體，或者是

=> 從頭重新做起，把你現在的 CPU(S) 換成一個完全嶄新的 80286或 80386 系統。

第一個和第三個選擇分別是最便宜的和最貴的方法，而

且也是最簡單的但不見得是最好的。第二個選擇是比較中庸的方法，這比單純的買一個全新的設備或只是增加記憶體的方法要考慮得週到一點。

然而以上三者都未必是最好的方法。暫時把錢拋在一邊不予考慮，買一個全新的硬體也不一定能得到最佳的回饋。這完全要看你的需求和這個新設備能滿足你需求的程度而定—不僅今年如此，往後的幾年也是如此。

爲了達到我們的目的，把購買新設備的方法擺在最後。這麼做是有幾個理由的。如果你對硬體能抱持審慎的態度，那麼在你還未考量好所有的可行途徑之前，千萬不要急著去買別的東西。所以我們就從最不昂貴的方法開始說明 — 加速卡。

9.1 加速卡

BONANZA 螢圖 (R)

(accelerator cards)

市面上有許多的加速卡，其價格約在一千美金左右。它幾乎能夠給所有的 8088 或 80286注入一個新的生命。這些加速卡中有的可以把 8088 變成 286或 386 ，還有一些則專門用在提升 80286的品質上。每一種都有它們各自的優點和缺點，只是完全依你的工作性質而定。此加速卡可能利用你所需要的新軟體和特徵而成爲代價高卻有效的方法。

典型的加速卡是插入你機器的擴充槽內，而且只是取代微處理機晶片 (microprocessor chip)而已。如此一來便會有一個速度更快的時鐘和全新的記憶體。

有件事值得一提的是 Intel公司製造了最受歡迎的加速卡。而且當他們在製作 286轉換成 386的加速卡時，他們的 8088 轉成 386的加速卡也能以更低的價格產生更明顯的改進來。只要有一個 16MHz的時鐘，則可以使一部舊 PC 的執行速度平均大概快四到五倍，大約是 AT 速度的 2 倍左右。如果你需要的話，其 Inboard 也有供 80287 或 80387輔助處理器使用的插孔 (socket)。

即使你想要加裝 Intel的背負記憶體擴充模組(piggyback memory expansion modules)的話，也只不過佔用了你一個寶貴的擴充槽而已。但是要提醒你的是在你的 PC 上必須有至少 125 瓦特 (WATT)的電源供應才行。

現在突然間以前的 8088 PC似乎變成了 386的機器了。其實也不盡然。因爲和典型的 80386機型比較起來，它們欠缺了 16 位元的匯流排 (16-bit bus)。 爲了解決這個問題， Intel也製造了能達到 4 megabytes、32位元延展記憶體的背負記憶體擴充板。此記憶體也能適用於 LIM 4.0的擴充記憶體，如此一來便能完全避開匯流排的限制了。

所有 I/O的動作都要利用 8位元匯流排來完成。而大部分安裝在 8088 電腦中的硬碟的存取速度要比那些專門用在更快速機器上較昂貴的硬碟要慢得多。其實除了存取速度之外還有更多的論點未加以說明。一個速度快的硬碟如果沒有安裝妥當還是會變得很慢。稍後在本章中將會介紹如何使一個舊的硬碟的存取速度變快。

如果你使用的是 PC ，那麼不論你怎麼努力，還是有一個 8位元匯流排的瓶頸存在。但是考慮一下要利用 DOS延伸

版本的 32 位元延伸記憶體來執行的程式的數目的話，情形可能不會那麼糟了。這些程式都試著要儘快地離開 DOS和這些問題，並且儘量把 I/O的動作降到最少的程度。如果它們能夠愈有效地解決這些問題，其他的因素也就愈無關緊要。

用來提高 8088 系統品質的加速卡會受到能被定址的記憶體總數量的限制。能被定址的記憶體總數部分是因爲系統內晶片的性質的影響，部分是爲了某個特定的電路板的關係。例如 Intel 的 inBoard 386 個人電腦只能夠定址 4 megabytes 的記憶體空間，所以爲了能充分運用它的 386，就必須使用延伸式記憶體 (extended memory)。然而，8088並沒有提供位址線路給延伸式記憶體使用，同樣的 8位元匯流排也沒有。所以實際上這種提高品質的方法還是有 5 megabytes 的上限 1除了 4 megabytes之外，在 InBoard中還有 1 megabyte)。

其實對許多使用者而言並不太可能把個人電腦擴充到這種程度。眞正能擴充到 4, 5 megabytes 的電腦數目並不多，至於 16 或 32 megabytes 更不用提了。

你也許會懷疑，爲什麼像 Intel Inboard之類的加速卡只使用他們自己的記憶體 (在 Inboard中佔 1 megabyte)而且取代了你自己系統內電路板上原來的記憶體。有一個理所當然的理由可以說明這件事：如果你記得我們在前幾章中曾提過記憶體能以微秒 (narosecond) 的速度來恢復記憶 (refresh) 的話，這個理由就變得很明顯了。個人電腦上的晶片或者是你在未來加裝上去的晶片絕大部分都是速度慢的 — 慢到無法在 16MHz 系統中使用。

286 的加速板在概念上與 8088 的不同。所不同的不只

是 16 位元和 8位元匯流排的差別，其中最主要的是 80286 的母板是設計成可以使用延伸式記憶體而 8088 就不是了。雖然 Intel提供了能增加 2 megabytes的 32 位元擴充板，它仍可以忍受任何在未改進的機器中所加入的記憶體擴充板。如此一來，便可在增加記憶體方面提供更多的選擇性。

至於 16MHz的時鐘速度 (clock speed)也沒有需要爭論的。把 286轉成 386的 InBoard的特徵之一就是它是設計來和 120微秒 (narosecond) 的主記憶體共同使用的，而並非是和比較貴的 80 微秒的主記憶體。當要存取它的背負記憶體時就加入二個等待狀態 (wait states)；而要傳送資料到匯流排時就加入 8到 10 個等待狀態 — 如此便可把速度降低到任何新加的電路板所能處理的程度。

安裝一個典型的加速板所須的技術和安裝一般的擴充板是差不多一樣的。這個意思是說如果你曾經把電腦的外殼拆開然後看看裡面的樣子的話，你就差不多已經成功了一半。當然如果你曾經安裝過擴充板的話，你大概是不會遇到什麼麻煩的。

你通常必須把原來的微處理機晶片從 IC 插孔上拔起來，但是要準備一個特殊的工具才會比較容易。而且你爲了要把加速板放到某個特別的擴充槽 (例如插到 8088 或 80286 IC插座的帶狀線路能夠和它相連) 時，必須把某些已經有的擴充板移到別的擴充槽去。

你可能會爲了花在啓動 (boot)你的新系統所需的時間而煩惱 — 特別是對 PC 而言。其實，就母板而言，加速板要一直到系統幾乎要完成啓動工作而且在 CONFIG.SYS 檔把設備驅動程式載入時才會被確認存在的。而原來的 ROM BIOS

啓動系統的意思是指它必須做整個記憶體的檢查工作等等。就連 InBoard也是如此。在它執行 AUTOEXEC 檔之前，大約要花兩分鐘的時間來做準備工作。畢竟花這些時間還是值得的。

9.2 採用新的母板

BONANZA 堂圖 (R)

走取代整個母板這條路要比僅僅加入一個加速卡到舊系統多花一些錢 — 尤其你是從 80286開始做品質升級的情形下。但是如果你是爲了 8088 而做品質升級的話，這個方法就要比加速卡更爲有效。

一個新的 80386母板一般都會擁有 8/16 位元 AT 機型匯流排的優點。但是有某些規定要利用某種專用的電路板來直接做 32 位元記憶體的存取。即使有 32 位元的 IC 插孔存在，也不要把這件事認爲是理所當然的。因爲就是有些具有完整電腦系統的製造商，雖然他們有製造 32 位元的 IC 插孔，也是會把速度變慢使得能夠提供給所有能與之配合的記憶體電路板來使用。

取代系統電路板所須的專業知識要比安裝一個加速板所需的多得多 — 但是在大部分情況下也不盡然。除非你做了不該做的事，而且只要你能夠確定你買的板子是能夠配合你的電腦的話，稍微花一點時間和耐性就可以解決了。

當然你可能會遇到接線鬆脫的情形，例如從電源到電路板的接線、從電源到磁碟機的接線等等。最好能夠對每個接線作標記以表示這條線是從那裡接到那裡，以後要觀察也比較方便。

其實並不是所有的接線都必須栓得很緊，而且每一條線有二個方向要找起來也是很煩人的事。即使眞的要這麼做，你也不會有什麼大麻煩的。因爲經驗告訴我們，電路板不能用主要是因爲你可能在底部把兩條線接在一起，而不可能是燒壞了。所以你試著在接頭的周圍找找看錯誤所在。一點點的疏忽是在所難免，但是只要多加小心就可避免。實際的情形並不像你想像中的那麼糟，只要你多留意一些，例如該留空隙的就要留空隙，而且確定所有的接點都已經緊接好了。

但是並非所有加速卡方面的爭論都消失不見了。如果你要把母板移到 8088 電腦上，8 位元匯流排的問題已經不在了，但是匯流排速度及你曾經用過的擴充板相容性的問題(在最後一章討論)就變得非常實際。

除非你也把硬碟換掉，否則你仍然會遇到 I/O的瓶頸。但是通常有一個補救的辦法，使你不必新買一個存取速度快的磁碟。這個辦法就像當你在安裝一個加速卡的時候一樣：改變現在磁碟內資料的交錯 (interleave)。

9.3 交錯率 (interleave factor)

BONANZA 螢圖 (R)

當在處理硬碟的問題時，最不讓人瞭解的觀念就是所謂的交錯率 (interleaving factor)。因爲即使是慢的硬碟通常也快到讓電腦來不及讀取相鄰兩個磁區 (sector) 的資料。所以我們必須設法使硬碟慢下來才能讓電腦在讀/寫下一個磁區之前把剛才的資料 "消化" 完。但是有件事很不幸的

就是你不能眞的讓硬碟慢下來還期望它能運轉得很順利。所以我們就用所謂交錯 (interleaving) 的方式來達成。以下介紹這個方法。

顧名思義，這個方法就是要你把系統所要連續讀取的資料分割成一個個的磁區，在每一個磁區間插入其他別的資料磁區，至於要插入多少個，只要能夠讓系統在其他的資料進來之前有時間把它 "吞" 進去就可以了。很明顯的，你不可能確實地在磁碟上作交錯的工作。所以技巧是在寫資料的時候依著: 寫一個，跳過一個，寫一個，....的方式進行。同樣的讀資料時也是依同樣的方式: 讀，跳過，讀，....。這樣其實就是一個機械式的等待狀態 (mechanical wait state)。

如果系統在下一個要讀取的磁區進來之前還沒有把最後的一個字元 "吞" 進來的話，它便會省略新讀的資料然後等到磁碟作一個完全旋轉把該未讀完的磁區再度移到磁頭的下方。這樣便浪費了空轉一周的時間。所以要讀的資料愈多，浪費在空轉的時間也愈多。

相反地，如果交錯的個數設得太多，經常會讓磁頭閒著沒事做，而且也浪費時間。雖然不像繞了一圈所浪費的多，可是也足以使效率大大的降低。

所以有一個交錯率來控制這些情形發生。此交錯率是設定在你要建構 (set up) 你的硬碟或者是你要格式化 (format)硬碟時。典型的交錯率一般是:

=> 4:1 對 4.77MHz 的 8088
=> 2:1 對 8-10MHz 的電腦

=> 1:1 對 80386

時鐘速度對交錯率扮演一個很重要的角色。所以當我們安裝一個具有更快速時鐘的裝置（如線路板或加速板）時，其時間安排（timing）也會改變。雖然時間安排變了，但是磁碟的執行速度卻未曾改變。這就是說以前正確的交錯率在你安裝完畢之後就不再是對的了。（請參閱表9-1）

表9-1 交錯率和有效的資料傳輸速率對照表。是從加裝了Intel Inboard 386-PC的個人電腦上所做的試驗而得。我們可以看到交錯率爲 5的情形最佳。而交錯率爲 1或 2的情形是用在 80386系統上。

交錯率	有效傳輸速率	時間	相對比較
1	28.3	75.0	
2	26.8	29.1	
3	25.5	83.3	
4	26.8	79.2	
5	85.0	25.0	
6	85.0	25.0	
7	72.7	29.2	
8	63.7	33.3	
9	56.6	37.5	
10	51.0	41.7	
11	46.4	45.8	
12	42.5	50.0	
13	39.2	54.2	
14	36.4	58.3	
15	34.0	62.5	
16	31.9	66.7	

在這個試驗中使用了一個非常特殊的軟體 — Hoptimum

，用來檢驗某交錯率的效果如何，同時可改進該交錯率。Hoptimum是由 Paul Mace公司所完成，程式寫得非常好，而且非常有技巧。它要求你的硬碟必須一個磁區 (sector) 接著一個磁區地重新格式化。資料先從要被重新格式化的磁區內讀到記憶體中，然後再寫回去。一般說來會進行得很順利，但爲了以防萬一，最好在你進行之前先把你不想失去的資料做個備份。

當然你也可以買個速度快的硬碟就算了，但是如此一來你花在提升品質的代價可能要比安裝加速板的代價還要高出一半以上。所以要好好的考慮一下。同時要記著某個硬碟只能達到某個程度的極限，你也不能要求得太高。所以Hoptimum仍是值得嘗試的。

9.4 超速 "磁碟" 以及磁碟高速緩衝記憶體 (disk caching)

BONANZA 瑩圖 (R)

你可能曾經用過所謂的記憶體磁碟 (RAM disk)。每次啓動你的系統時，你可以用 AUTOEXEC 來爲你的某些暫時檔案建立一個快速 "磁碟" 以便作快速的存取動作。比較起來，任何一種機械式的磁碟動作都比較慢。

如果今天我們要使用更大的資料庫，更大的檔案和更大的程式，磁碟的 I/O動作將會是個很大的瓶頸。在你經常使用 DBASE之後就會知道浪費在磁碟 I/O的時間是多麼的多了。磁碟高速緩衝記憶體 (disk caching)便是針對這個問題

而來的，它能夠稍微使資料的搬進搬出更有效率地完成。但是有一個限制存在--有多少的記憶體能夠用來作磁碟高速緩衝記憶體？如果是 512K 就很有幫助，如果是 1 mega 就更好。不過這些數目一旦和所有檔案和資料大小的總數目比較起來，只不過是一大桶水裡面的一滴水而已。

如果你有記憶體的話當然能使你的磁碟高速緩衝記憶體的大小變大。尤其在你只偶爾會用到一個比較大的磁碟高速緩衝記憶體時，這樣便可以把記憶體讓給其他的使用者。如果你所需的磁碟高速緩衝記憶體是一個定值的話，你最好考慮用硬體來作你的磁碟高速緩衝記憶體。Distributed Processing Technology 公司就曾經製造了一個典型的例子，而它的容量大約是 10 megabytes。

另外一種方法就是使用記憶體磁碟 (RAM disk)。提醒你，在市面上，一般性的記憶體磁碟是不存在的。記憶體磁碟的容量是 10, 100, 1000 megabytes, 甚至更多。不妨把記憶體磁碟想像成大到可以載入你的整個硬碟而且提供你和記憶體一樣快的存取速度。在你家附近的電腦商店可能不會找到它，可是它還是有存在的。

Newer Technology是可擴充記憶體磁碟 (expandable RAM disks)的主要供應商。這間公司在我們討論快速記憶體的地方有提到過。他們的 DartCard 可以增加 4,16 或 64 megabytes。

雖然它在市面上主要是記憶體磁碟系統，但是如果需要的話，在 DartCard 上的記憶體也可以當作延伸式記憶體 (extended memory)。有 8 megabytes 能夠當作 LIM 3.2 EMS 擴充式記憶體。這在 PC (8 位元), AT, SCSI 和 ESDI

的介面上都可以看得到，而且也可以當作是與 BIOS 相容，可啓動系統的磁碟。

9.5 有關 ALL ChargeCard 286

BONANZA 曼圖 (R)

在市面上提供給使用者用來提升 80286品質的硬體中，最有趣的部分不是加速卡，而是記憶體管理程式 (memory manager)。直到目前爲止我們所討論的記憶體管理程式只是一種軟體的功能 — 一個可安裝的設備驅動程式，經常提供除了 LIM 4.0 EMS設計規格內所須的其他功能。有一間加拿大的電腦公司更進一步提供了支援微處理機晶片的硬體 (此公司爲 All Computers Inc.)。

正如第二章中所討論以及第六章對擴充記憶體驅動程式的進一步說明，單單靠軟體是不能夠表現 8088 和 80286晶片的潛在能力。這主要是因爲太少的潛在能力可以發展。然而正因爲兩者都可以支援運算輔助處理器 (即 8087 或 80287)，所以 (特別是 80286) 可以加裝一些設備來藉由輔助的硬體支援其他的功能。

All ChargeCard 286就是這類產品的代表 (它是第一個引起廣泛的注意，而且也是最成功的其中一種) 。它增加了多元工作 (multitasking) 的功能和記憶體重覆映射 (memory remapping)的能力。

ChargeCard 286藉著不斷的設計 80286的結果，使得能眞正地把記憶體位址映射到從 640K 到 960K 的範圍 — 最底下的 64K部分是保留給 ROM使用。這也在一般的視頻位址 (video address)的範圍內，只是提供了有 960K連續位址的

記憶體。

至於視頻位址區該怎麼辦呢？ChargeCard 重覆映射的缺點就是：爲了讓程式能夠寫資料到視頻位址區，必須修改這些程式。All Computers 公司提供了一個不難的步驟，此步驟須藉助一個稱爲 ALLPREP的工具。只要你不要更改到原來的版本，先準備一個備份，就不會把你有價值的軟體給毀掉。如果你在安裝上有任何的問題，最好由專業的人來做。

安裝好之後，經由 ChargeCard 所提供的一個工具將會把 .COM 和 .EXE 的檔案以及 TSR載入到記憶體 640K 以上的部分。任何一個正在用一個複雜的設備驅動程式來把記憶體重覆映射到未曾用過的位址空間的 386使用者此時可能會說 "那有什麼了不起呢？" 但是對 286而言，那實在是很了不得的事情。實際上，ChargeCard 不僅使 286 的效率就像 386 一樣，而且也可以在其上執行 386的軟體，例如 PC-MOS/386 (一種作業系統，我們將在以後說明)。

至於 ChargeCard 所提供的其他用具提供了系統映射 (system mapping) 和分析工具箱 (analysis tools)，這些功能類似於我們在第六章所討論的 386 MAX設備驅動程式和 QEMM。

假設我們有充分的支持 LIM 4.0 EMS的設計規格。ChargeCard對 640K 以上的部分有重覆映射的能力。ChargeCard允許原來佔據位於 640K 以下珍貴的記憶體的設備驅動程式和其他東西重新定址 (relocation) 到 640K 以上的部分。這就是說能夠在 640K 以下的部分騰出更多的空間給你的應用程式使用。這也正是我們在討論 80386設備驅動程式時的重要特性之一。

這種品質提升的方法並沒有改變時鐘速度。它並不像加速板般的提高微處理機晶片的品質。但是其所花的代價和一些比較少利處的加速板比較起來就低得多了。它不僅在今天提供了一個解決之道，同時也爲將來搭起了一座橋樑 — 如OS/2和 XENIX等等。這是因爲 ChargeCard 就是藉著把延伸式記憶體轉變成擴充式記憶體來運作的，如果需要的話也可以由軟體轉切到延伸式記憶體。

就像你的雙手手指交叉的形式一樣的容易，ChargeCard 能夠直接插入原來在 8088 或 80286上的 IC 插座(socket)上。直接插入系統內部的處理機晶片上而不藉由擴充槽的原因是很重要的。直接插入可以讓 ChargeCard 來直接存取資料，而且不論是那一種記憶體安裝在母板上，都能把它的管理架構 (management scheme)包含在內。

這主要是因爲許多的加速板或記憶體擴充板要求系統的記憶體必須失去運作能力，而由它們自己本身的記憶體去取代之，任由系統記憶體空著不用。例如 Intel的 InBoard系列除了以 80286或 80386取代原來的微處理機晶片之外，還用它自己的記憶體來取代系統的記憶體。在舊版的 LIM 3.2 EMS 設計規格中還不是這樣子。但是在 4.0 EMS中，有許多的硬體廠商發覺只要取代原來的記憶體將會變得更容易而且值得。

除了你原來系統天生速度上的限制之外，還有其他的限制也要注意。在 80286系統中 ChargeCard 能夠管理的最大記憶體是 16 megabytes 。這和使用擴充記憶體時的 32 megabytes 不同是因爲 80286藉由延伸式記憶體所能處理的只能到 16 megabytes ，而且即使我們可以把延伸式記憶體

當作擴充式記憶體來使用，我們眞正處理的還是延伸式記憶體。對 8088 系統而言，ChargeCard所能管理的記憶體最大容量是 10 megabytes。

其實這樣的記憶體還是很大的。當你談論到這類的數字時你可能會聯想到某個具有快速的時鐘，快速的磁碟存取速度和快速的記憶體存取速度的東西。其實不然，在許多應用上，All ChargeCard和其他可能出現的類似東西只是代表著到現在爲止能以低費用達到最高的效率而已。

All 的 ChargeCard 能夠和所有能提供延伸式記憶體的擴充式記憶體一起使用。今日大部分比較好的記憶體電路板提供使用者選擇使用延伸式或擴充式記憶體的功能，而且如果需求改變的話也可以隨時轉換。在使用 ChargeCard 時你只需要把它們設爲延伸式記憶體而非擴充式記憶體即可。

這種方法應該對那些還有像3.2 EMS Intel AboveBoard 和 QuadEMS+ 的舊電路板的人比較有興趣。這些舊的電路板只靠安裝一些設備驅動程式是不足以提升到能達到 4.0 EMS 的效能。藉由把記憶體視爲延伸式記憶體然後再轉換成擴充式記憶體的方法，ChargeCard實際上已把記憶體變成 4.0的支持者了。

要注意 ChargeCard 是一個設備 (device)，所以像其他的設備一樣需要設備驅動程式 (device driver)。因此系統不僅要辨認此設備，而且也要提供必須的軟體介面。ChargeCard本身就包含了所需要的專用軟體，這是因爲它本身硬體支援的獨持性質以及它的軟體介面並不適合於一般用途。

不僅如此，因爲它重覆映射的能力以及爲了要替你的應用程式提供一塊連續的記憶體 (理論上會高到 960K 的位址)，它必須修改視頻驅動程式 (video driver) 來把視頻位址區移出原來 A000h～BFFFh 的範圍。所以要完成這些工作的軟體也必須包含在內。

All Computers 公司 (ChargeCard的製造者) 附帶地爲386系統提供了 Charge386 介面卡以提高記憶體的管理。但是他們在這方面的改進似乎比不上其他如 8088 或 80286生產線上的方法，所以在此就不詳細解釋。不過他們也進一步說明了認爲今日電腦中所有的晶片都已經充分地利用了是一種謬論。同時，他們也說明了只有更小心的製作和設計才是可行之道。

所以，在你捨棄現有的東西並且到外面重新買一個新的之前，還有幾個選擇要好好考慮。提升你系統元件的品質可能會給你目前所有的計算能力，而且可以期待著將來的來臨。

你會發現 IBM是給我們一個奇怪玩具 (它的名字叫 PC) 的人 (PC 在剛出現時並沒有人知道它能做什麼)，同時又給我們開放性結構 (open architecture)使得我們現在能夠使一部舊的 PC 返老還童，就像喝了青春之泉一般。你可以確定沒有人能夠想通這個自相矛盾的問題。

第十章　最終的改進

在某個階段時任何的補救方法都不夠用，例如加速卡或加裝另一個記憶卡。而現在已出現了一種新的晶片，一種比 80386 更好的晶片。

80486 已成功地發展出來了。雖然現在一般使用者還未用到，店面也尚未出售 —— 而假設現在已經出售，價錢也將貴得驚人。雖然我們現在才剛對 80386能替我們做什麼勾劃出一個大略的架構，這種是已將 80386 淘汰的新晶片就已出來了。這種晶片擁有相當於一百萬個電晶體，其速度爲 80386 的三到四倍，但卻是屬於 8086 家族中成熟且反向相容 (backward-compatible)的一員。

這是什麼意思呢? 這是說大約要在幾年後一般使用者才有機會用到。80486 是一種高用途 (high-end) 的晶片，專門銷售於高用途的市場。它在科學和技術應用上的處理能力會使其他問市的產品望塵莫及。對大多數的人來說，這也是一種好奇心。

但是一般看來，個人電腦上最受歡迎、最廣被使用的仍將是 80386 系列。在本書中不只一次提到 80386 ，且將許多問題的解答都設爲 80386。在這種時候，幾乎沒有人能否定 8088 在其最高的限度下，仍是可用的；當然，除了一些特定的情況。於是，386 是否眞的需要嗎？或是其實只要用 286 就夠了且很便宜 ？

事實上答案並不是這麼簡單。如同 8088 會持續一段相當長的時間來滿足許多使用者的需要 (例如普通的文書處理)，80286 可利用其較强的能力來增加些許速度，而解救其他受限於速度的使用者，和缺少 8088 增加的記憶體支援的使用者。

至少目前這種情形是適用的。80286 的問題已有辦法解決，例如由保護狀態改回正常狀態，或是回到 DOS。而又因有了硬體加强的幫助，286 擁有者可由多用途卡 (All Charge Card) 將此二者 (80286 和 80386) 間的差距大大的縮減。

就一個最單純的例子來說，有多少人會眞的想要或需要使用到比 80286 所提供的 16 megabyte 增加記憶體容量限制更多？事實上顯而易見的，這並不是眞正的問題。眞正的問題是 16 bit 系統對抗 32 bit 系統以及 16 bit 軟體對抗 32 bit 軟體 —— 甚至於以 8 bit 套裝軟體起家者也跳過了 16 bit 286 市場而充份利用 32 bit 的優異速度 —— 感謝一些新的擴充式記憶體發展工具。

現在對任何考慮購買一個新的 286的人的最好忠告應該是：若能避免則儘量不要買。在 286設計中所存在的問題並不會消失。正如擴充式記憶體變得越來越重要，以及 DOS

延伸 32 bit 軟體變得越來越普遍 (DOS 延伸 32 bit 軟體能在 286上執行), 286機器的有效性則變得越來越值得懷疑。

現在有另外一種 80386晶片, 即 80386SX, 可提供製造商一較低成本的選擇來提供完全 32 bit 的處理能力。許多人預測 80386SX會在不久的將來以中等價位全面地取代 80286, 且可作爲 8088和 80386之間差距的橋樑, 以及填補 286 所不及的部分。

386SX 從最初的支援時鐘速度到至少爲 16 MHZ , 逐漸發展至超越最近 286 所及的速度。當然 386SX 可執行任何 8086/8088 的軟體。更重要的是身爲一個 32 bit 的處理機, 它可執行在市場上以及 DOS 未來發展上極具影響力的 32 bit DOS 延伸軟體。

386SX 並不是 Intel 公司因 80386 日益普遍而想出的同一種商品賣兩種價錢的噱頭。它是一個完全不同的晶片, 不論在操作特色及其他方面皆十分的不同。舉例來說, 386SX 所提供的最大實際記憶體 (RAM)被限制爲 16 mega bytes , 與 80286 相同, 而不是如 80386 般爲 4 gigabytes。但是它可以使用高達 64 terabytes 的虛擬記憶體, 及提供最大爲 4 gigabytes 的段落大小, 這些數字又與最原始的 80386 相同。

386SX 並沒有提供 32 bit 匯流排結構。這個一如實際記憶體定址的數量, 正是 SX 晶片上接腳數目的功能 —— 比 80386少了 32 隻接腳。然而, 16 bit匯流排並不會對它的處理能力作任何的限制, 只會對其排線存取作限制。連同其他的因素, 這使得系統主機板在設計時加入一 32 bit 的

排線來支援 80386 的成本能大幅降低。

如同一臺 DOS機器，SX 和 386 一樣擁有同樣的重新定址的功能。另外，還有一種 80387SX的同作處理機，專門用來處理責任重大的數字壓縮。實際上在許多的應用中，386 和 386SX機器之間的差異應被忽略；而在某些情形下，甚至不應注意到任何差異。這種情形下的成本比較起來也較一完整的 386 來得低。

雖然它比 386不足、但仍然較一 286高等且貴了好幾百美元。要將一擁有 All Charge Card的 286升等，則將比前所述花費更多、更貴。然而，這並不意味著 286將完全被淘汰。至少一些市場分析家仍認爲它可維持一陣子。雖然過氣、但仍未被淘汰。但是由於 386SX的緣故，286 將被迫降價，而又連帶地使 8088 價錢降得更低。

10.1 別忘了匯流排

BONANZA 螢圖 (R)

正當 CPU (中央處理單元) 的速度越來越快、匯流排的速度和設計很明顯地成爲一重要瓶頸及主題。在前面有關擴充式記憶體板的章節中也曾提到這個主題。

今日的科技已使得舊的匯流排的結構被淘汰。甚至於現在許多記憶板仍無法與有些機器上的匯流排速相容；這也不是由換用速度較快的晶片就能完成的。問題是出在匯流排本身。所以最好的改進方法就是徹底的更改匯流排。

這也已完成了。也就是 IBM推出 PS/2 的目的。然而 IBM 目前卻嚴格地守著此項設計—微通道 (Microchannel)

結構的版權，以防止他人濫用。

微通道究竟有什麼特性？它可以 CPU的速度執行，且沒有延遲、不需等待。其中一最重要的特性是：較進步的匯流排調停 (advanced bus arbitration) 。這個特性的意思是指許多信號能同時分用同一個匯流排。傳統的個人的電腦一次只能處理一個動作，例如將一個區塊的資料搬到硬碟或由硬碟搬出。微通道可處理此動作且在同時處理其他別的資料轉移的動作。這是一個多功的匯流排。

實際上這隱含了整個直接記憶體存取 (DMA)設計的改變的理念。有了微通道後，八個直接記憶體存取可同時移動資料而不需使用微處理機的運作。直接記憶體存取其實是由另外一個比微處理機小的晶片 —— 副處理機 (coprocessor) 來控制。副處理是專門設計來作運輸管理的工作。

有了微通道後，主電腦只需要告訴直接記憶體存取控制晶片它需要做何種計算、何時做、在哪做，而在傳達此訊息時，直接記憶體存取甚至可正在處理別的事情。這一點與傳統的電腦匯流排有很大的不同。在傳統的匯流排上。當產生一個直接記憶體存取的傳遞時，幾乎所有其他的處理都必須停下來。

微通道是一種非常聰明的結構。它可處理在軟體裝設時產生的錯誤 (如某一記憶體槽 (bank) 損壞)，且可以跳過這塊損壞的記憶體並繼續運作。

以上這些並不代表 PC/AT的開放結構已走到盡頭了。這些只是用來說明事實上存在某種比較好的結構。系統設計師可使用許多其他的技巧，如增加 I/O的速度。

同樣的，這也不表示說利用一些技巧並不能增進微通道的速度。市場上已出現許多種微通道，且還會出現更多。只能讓時間來證明微通道的功效。

有些尖端的第三組織廠商正在生產不同型式的產品。然而，在有執照及其他因素（如在此情形下所需的一些 VLSI 晶片）之間，使用者不應期盼微通道產品在市場上將有充塞現象、如早期的匯流排一般。

至此，我們已介紹了不少的硬體與軟體。其中有電腦使用別人的記憶體，而在另一個極端則有電腦使用自己的記憶體，且在使用後丟棄，而不讓別人來用。在工業中所製造的產品則介於這兩者之間。然而這種型態卻早已不適用了。

這是否表示使用者應購買沒有記憶體的電腦呢？也不盡然。然而，當我們研究計算機的發展史後，這可能是一個不錯的主意。但是甚至與超音速的時鐘速度比起來，記憶體管理今後都將是一最重要的主題。這其中也隱含了一個十分有趣的故事。

先前我們曾提過 EMS的本質，卻絲毫未曾提到 EEMS 的本質。EEMS是擴充式記憶體的基石。EEMS强調記憶體並不被 CPU 所擁有，卻是如個人電腦的本質一般，能獨立控制；這只是一種觀念，而沒有實際的應用。然而，在 AST中有一些聰明的人，被某些存在的邏輯所激發，認爲一定有其他更好的方法。

所以 AST創造了他們自己的一套標準，與當時所被接受的 LIM 3.2 EMS相容，但卻更深入一些 LIM 3.2 EMS所不及

的領域。這並不是爲了滿足某種需要而設計的，實際上並沒有任何需要。AST 如此設計只是因爲他們認爲這樣設計才是正確的。身爲工程師，他們創建了一個較好的叫做 "whatzit" 的陷阱。然而，沒有人知道 "whatzit"可被賣出去且抓住一隻老鼠則就沒問題了。

而在某日有一群人帶著 Desqview 來說：「喂！我們可用這個來做多功！」之前，AST 與多功完全扯不上關係。

這個故事的寓意十分簡單。在這個行業中新的構想是十分有限的。雖然一時之間這些構想是十分支離破碎的。這當然也包括一個完全錯誤的開頭，一如哥倫布當年被嘲笑一般。

只要出現更好的發展，軟體總是可被改進，甚至與硬體相容。然而硬體卻完全不是這麼一回事。假設在設計時硬體並未曾預期到將有任何的改變，則在日後很難對其做什麼改進。

這對使用者代表了什麼意義呢？當然在不同的情形下各有不同，但是在九０年代，不管由任何方面出發，80386 已逐漸成爲任何架構的基本結構了。

這並不是說每一個人都應該趕快去買 4.0 LIM EMS下32 mega的電腦。雖然與幾年前比起來，很大區域的記憶體價格已降低了許多，但是與中央處理單元（CPU）相比較，仍然嫌太貴。所以正如其他物品、應該定出一平均價格。而那又是多少錢呢？

根據本書所作的調查，一般使用者在建置任何系統時所

使用到最大的記憶體空間爲 28 megabytes。

好吧! 在用 Microsoft's Windows, 或是 Softlogic Solution's Carousel , 或是 Quarterdeck's Desqview (三種最普遍的視窗環境, 可同時載入不只一個應用程式) 時再多加一 mega 左右好了。這三個軟體可讓使用者同時載入 Word Processor, Spreadsheet, Database 和其他幾種應用軟體。當然, 那要看那些應用軟體及其伴隨檔案的大小而定。然而, 在一般公司中, 這樣的記憶體容量則足夠了。

其中的問題是, 每次一有多餘的記憶體空間, 使用者就會想再多有一點、再多一點有多好。當使用正確的硬體時, 多加一些容量並不會造成任何困擾。然而, 當有了加增記憶體產品後, 有些困難的選擇必須要決定。

軟體很容易會被更好的軟體所取代。這是不可避免的。但是從另一方面來說, 有多少新發展出來的軟體能在今日的硬體上使用。或是假若能使用, 使用出來的效果又能有多好? 於是, 這些問題將繫於今日使用者如何小心地來選擇硬體設備。

10.2 重新循環的舊產品

BONANZA 螢圖 (R)

於是使用者下了一個決定去買一臺新的電腦。實際上, 舊的個人電腦、 XT 、甚至於 AT 已沒什麼人要買了。買的人可能比使用者插在插座上的舊卡都還少。高科技結晶的舊貨拍賣市場的時代尚未來臨, 而許多經銷商也不希望使用者能自行處理舊產品。

當然，還是有一些這樣的舊貨市場，只是使用者必須自己去找。在許多時候，這些舊產品仍有許多剩餘價值，且若就這樣丟棄了是十分可惜的（與當初購買的價格比較時更是如此）。今天許多公司中的員工所能從電腦上獲得的好處比從預算上獲得的多得多。所以在員工的桌上放置一臺電腦是一件十分合理的事。

然而較不能讓員工明瞭的，是將此電腦與其他電腦連線以形成一網路系統。在 DOS的選擇那章中，我們曾提到利用作業系統，多使用者系統如 PC MOS/386 可讓 DOS和 XENIX 同時存在，來節省新的網路系統的費用。但那些並不是僅有的辦法。區域性網路 (LAN, Local Area Networking) 帶來了各種不同的選擇。所以，不要太快淘汰舊機器，甚至於 8088。

最後一個注意事項：同樣請記得在前幾章我們曾提到曾有一款 80386的機器，推出後一直讓顧客等了好久才等到它的記憶卡。所以，記住在購買任何獨佔性產品之前，先確定此產品已上市了，而不是尚在計劃中、準備推出之類的。

第十一章　幾乎是多功

多功 (multitasking) 的定義爲在電腦上同時處理一件以上的工作。多功已成爲現在桌上電腦的最大突破之一，以及關係產能最大的因素之一。雖然眞正的多功至今仍是錯覺，以上所述仍然成立。而這個錯覺的產生尙分爲許多不同的階段。

11.1　程式轉換 (context switching) — Carousel的旋轉門方法

BONANZA 螢圃 (R)

SoftLogic Solution的 Software Carousel對許多使用者而言是最好且最便宜的方法來達成幾乎是多功的工作。他們的 5.0版可同時開兩個視窗呼叫兩個程式，例如使用者可利用一個螢幕來顯示 Spreadsheets 作爲參考，而利用另一個螢幕以文字處理系統來寫報告。這個幾乎多功已足夠滿足許多使用者的需求，且可增加產能。尤其當仍有多餘的記憶

體時，更是如此。

Carousel是最早使用擴充/ 延伸式記憶體的程式之一，且雖然它無法執行 EMS 4.0版上的函數，它仍是目前最持久的一個程式。

有了 Carousels 5.0版之後，使用者可同時載入十二個應用程式，且維持本身的檔案開啓；而在切換時可經由使用者自定某鍵而按下則完成。甚至到現在 Carousel 5.0 仍是一個最經濟的方法。Carousel本身也提供了虛擬記憶體，以便在隨機存取記憶體 (RAM)不夠時隨時置換到磁碟上。

Carousel所利用的技巧即是將整個區塊的記憶體來作交換。而此記憶體區塊的大小，隨系統的負擔而定，最多可達544K。這裡所謂的置換不只包括一般套裝程式和任何使用者載入的 TSR，更包含了任何已開啓檔案的置換。當然，此置換也包括影像記憶體，故在置換後的螢幕與置換前相同。

Carousel首先使用傳統式記憶體，而後改爲擴充式記憶體。它只有在已用盡所有的記憶體後才會用到虛擬記憶體。在作何情形下，Carousel只會將各種資源用到使用者所設的限度之下；當然，這個限度不能超過所有的總數。然而，在Carousel使用所有資源的方式，也使其成爲一種如同本書中所提及的其他軟體一般的混合式。

這裡必須注意的是，Carousel並不像 Desqview 或是 Microsoft Windows 一樣，爲 LIM 4.0 EMS的規格。甚至在5.0 版中，Carousel並不能眞正地置換進或出眞正的記憶體區塊。

事實上，Carousel使用一全然不同的方法：複置。它並不像 EMS 4.0版一般保留記憶體及其內容在原來的位置，而只置換位址。Carousel小心地一區塊、一區塊的複置傳統和影像記憶體中的內容，至所被指定的地方。然後，一區塊、一區塊地讀進使用者下一次所呼叫的應用程式，或假設下一個程式尚未被載入，則載入該程式。

若是在複置時，使用者利用硬碟來模擬擴充式記憶體而不是使用隨機存取記憶體，則使用者可在每次硬碟「吞」下一區塊記憶體時，看到燈光閃爍不斷。就算有足夠的隨機存取記憶體來支援實際記憶體，Carousel仍然必須處理此複置的步驟。假設使用者使用隨機存取記憶體，而不是用複置/置換到磁碟機上，會快得多；但這樣子卻不是同時的了。一般來說，視所置換的區塊成視窗而定，一個 4.77MHz的個人電腦需花費一到三秒；而在同樣的機器上，需花費四十五秒來作一個大的視窗的雙重置換到磁碟機上。

這非常的慢，但是這也是看從什麼角度來看而定。假設隨機存取記憶體不夠，或是當 Carousel 正在從磁碟機上複置下來，或是要複置到磁碟機上時需要等待，則這是很慢。然而慢是在比較下產生出來的。和關閉檔案，結束一程式然後載入另一個程式，再開啓檔案這一連串的動作比起來，卻又快了許多。

在現有的一些虛擬記憶體策略之中，複置不愧爲其中最慢的，但這也是唯一能達到 Carousel 的需求的方法。然而，使用者可自己使用一些策略來增進記憶體使用，如將最快的隨機存取記憶體資源分配給最重要的程式（當然此程式將比其他程式先載入）。事實上，假設使用者同時載入兩個程式，且又正好有足夠的記憶體來支持，Carousel只有在使用

者欲載入第三或第四個程式時，才會執行 「複置至磁碟機」的技巧。

Carousel其實是一個非常聰明的程式。使用者可對著Carousel指定任何欲配置給它的記憶體，如全部或部分的隨機存取記憶體，或是任意的磁碟機容量。它會根據這些資料判斷出可開之最大視窗。這也可說是使用者告訴此程式他想要做什麼，而本程式就會竭盡所能地去做。假設使用者在載入 Carousel 之前先載入另一 TSR，且並未將此情形告訴 Carousel，Carousel會自己判斷且利用所剩的記憶體開一個較小的視窗。

就算使用者移開部分 Carousel 將要用到的隨機存取記憶體，例如拆掉一塊擴充板，Carousel也會自己去適應這種情形。當記憶體不夠時，他也絕不容許除了已開啓的視窗外，再去開別的視窗。它是一個令使用者非常方便的一個工具。

Carousel，或是 Microsoft Windows，或是 Desqview，何其他的視窗環境所需要的記憶體數量可由簡單的數學及周密的管理而得。現假設正有??的視窗正在執行，且每個視窗的大小爲各 544K 。其中有五個視窗必須被儲存起來，這差不多需要 3 mega 左右。許多程式都必須在小視窗下執行，所以最好即是給這些程式所能執行的最小的視窗。這是盡可能地將空間留給其他等待中的程式。

使用者可不經由重新啓動系統而離開 Carousel ，但是如同 Desqview ，使用者不能分別將視窗關掉以求取更多的記憶體供應設置 Carousel 時並未讓 Carousel 知道的程式來用。然而，使用者可跳出在某一視窗下的程式而使用此處

之記憶體。另一種選擇即是將一視窗當作一開放式 DOS視窗以便使用。

使用者可利用 Carousel 來做非常多的事情，且可利用許多技巧使其幾乎爲多功。事實上，事先載入某些背景通訊軟體或硬體/ 軟體組合可眞的達到多功。

綜合來說，Carousel的幾乎多功可成爲一贏家，甚至於對某些公司中有著極强而欲使用一些較複雜的系統的使用者亦是如此。

11.2 背景中的工作置換

BONANZA 螢圖 (R)

有另一種途徑可在 640K 的個人電腦階段上使得一些重要元素達到多功。這是 TSR中的一種，平時安靜地在背景中不被注意，直到被一些外在的信號指令趨動使其活動。在一個不具有眞正多功的機器上，它們會接受指令，而將使用者正在前景執行的程式暫時放在一邊，當它做完後才會回到剛剛離開的地方。

通訊 (無論是經由傳眞板或是 modem) 是這種背景科技最好且最普遍的例子。這個在簡單的兩臺鄰近的電腦相連，基本的網路時也適用。背景程式會視有多少東西必須輸送而決定將控制權轉移的時間。這是一個非常方便的工具。

背景程式最大的好處在於，當使用者決定將系統升等時，除了少數的網路設計外，它可直接在多功環境下執行。於是，以下我們將來討論這一類中的幾種普遍應用。

11.3 傳眞板 (fax board)

傳眞機不僅只是通訊的一個方法而已。它已成爲生活的一部分，尤其是在分工的世界中。似乎在每一個地方都有傳眞機的存在。然而，仍有一些事是電腦能做而傳眞機所不能的，所以此二者的結合是十分的自然。就算使用者並不需要電腦化傳眞的所有功能，但加裝一傳眞板的價錢大概只要一個傳眞電話價格的一半左右。

利用電腦作爲傳眞輸送機，使用者可直接由磁碟機來送檔案。送進來的資料可先暫時或永久的存在磁碟機中，或是直接送到印表機。

JTFAX ，一種由 Quadvam推出的傳眞設備，爲一種可由硬體和專利軟體將傳眞與電腦結合的例子。雖然它們提供了許多不同的設備，在此我們僅討論如何利用最低階的傳眞板來傳送。

JTFAX 的半口 (half-slot)板可裝置在任何開放式八或十六位元 (個人電腦類)的 PC/AT 或其它相容的匯流排上。十分有趣的是，當我們正在處理硬體上的設置時，並不需要在其 CONFIG.SYS 檔中設置趨動裝置。軟體可由 AUTOEXEC.BAT檔或由 DOS的提示下來裝置。通常將其設爲在啓動時可自動設置較好。如此可使使用者在需要時一按鍵盤即可。

使用者可將 JTFAX設爲自動管理電話連線，應用在有電話進來或是由人工操作時；假設使用者有一條另外的資料線則設爲「自動接收」應是最好。軟體將會如任何 TSR般運作

，但輸入的資料是來自電話線，而不是由鍵盤輸入。若是在一項工作之中出現這種情形，則螢幕上會顯示出已收到一個 FAX 訊息。如同其他任何 TSR，當 FAX在處理時，其他的動作皆暫停。

進來的訊息可自動被存在磁碟機或送至印表機。一旦此訊息被存入或已送出至印表機。控制權馬上就轉回使用者，使用者可回到剛剛控制權轉移處，直到下一通電話出現。

人工操作則完全不同，主要是利用同一條線來傳送一般的電話及傳眞。在這種情形下，使用者必須接電話，而若對方說他欲送一傳眞時，按下 F8 鍵切換。

此軟體利用一類似 DOS或 OS/2 下的拍紙簿(scratch book)中叫環境 (Environment)的東西，且十分普遍地爲TSR所利用。使用者可利用它來告訴系統希望將傳進來的資料存在哪個檔中，哪個目錄下，或是送到哪。這和 PATH 的運作有些類似，但必須用到一 SET指令，如下:

```
SET FAX=C:\FAX\FILES
```

這與 FAX的指令不同，FAX 會直接趨動軟體。在上述情形下軟體是由設置的程式來趨動。

送出的傳眞也可自動或人工地由許多傳送選項來做。這是否表示多功? 不是。雖然這不是多功，但仍被許多公司普遍應用。而當它在 Carousel 之前被載入，則十分接近多功。而在往後欲將系統升等來達到多功時，仍然可以使用

JTFAX。

11.4 近似網路

BONANZA 螢圖 (R)

仔細地來想一想，網路其實是一種多功。網路將兩個或兩個以上的電腦連線，使得許多使用者由一個供應站共同使用一資料或程式。除非此供應器只負責供應資料，否則其爲多功。一般來說供應站必須讓一操作者在前景，而其本身在背景提供眾使用者的需要。但這如何達成？切割時間。甚至於在一 8088 個人電腦上亦是如此。

DeskLink，另一種利用串聯至串聯埠(serial-port-to-serial-port)多功地來做網路。但 DeskLink 只能使用兩台電腦，即將其中一台作供應站而將所有檔案提供另一台用。當然這並不只限於檔案。DeskLink讓使用者也可共同使用一印表機，且有一內設線上同時周邊處理來達成此一功能。

假設有人認爲串聯埠用來作處理太慢了，那就錯了。事實上串聯埠的速度是很快的。是接在串聯埠上的裝置使它慢到 300 baud，1200 baud 或是 9600 baud。事實上，串聯埠之最快速度可達 115,000 baud，而這是大約許多 1200 baud的 modem 的一百倍。

比較起來，DeskLink操作的速度：115K，遠比 DOS將檔案複置到磁碟片上的速度快多了。

這有點像 DOS的指令：DOS ASSIGN。遠處的單位由供應站存取磁碟，就如同磁碟機在本身內部一樣，只不過代表磁碟機的字母與一般機器不太一樣。舉例來說，供應站上的 C

磁碟機可能在遠端機器上以 D或 E來代表。然而在主機器上的使用者欲存取時仍將以 C磁碟機來表示。

如同前面所提，傳眞機在背景運作，然而 DeskLink 則是--TSR 。當遠端欲從主機上存取時，如同任何 TSR一般，主機上的使用者的前景會暫停。

所以，有效地來使用　必須要由使用者盡量降低存取的時間到最小來完成。然而，假設使用者欲將系統升等來提供眞正的多功，如由 386上執行 Desqview，DeskLink 並不能如此被使用。因爲 DeskLink 和多功皆需要同時直接存取，所以兩者不能並存。這是非常可惜的一件事，不然兩者一起會對一些公司非常方便。

11.5　線上同時周邊處理 (spooler) 和緩衝輸出

BONANZA 榮團 (R)

另一個可增加許多生產力，而又便宜且簡單的升等爲列印線上同時周邊處理的某種形式。這其實是一種多功，因爲使用者可繼續在電腦上工作，而軟體則在背景工作，讓印表機不停地列印、列印....。線上同時周邊處理所附加的另一個好處則是可在任何電腦上運作。

列印線上同時周邊處理的最基本型態是一個隨機存取記憶體區塊 —— 傳統記憶體，用來當作輸出至印表機的儲存空間或緩衝區。 DOS中提供了一個列印線上同時周邊處理——— PRINT.COM，而市面上也有出售其他種類。另外，許多文字處理機或其他應用程式有它們自己的內部線上周邊處理。

注意這裡所說的是「傳統」記憶體。假設這其中有任何最近不會用到的東西，這即是浪費珍貴的傳統記憶體空間。

然而，有些線上同時周邊處理（特別是那些在某應用程式中運作者）成功地隱藏起來，它們所造成的開支則爲可忽略的。而另一方面，有些外部的線上同時周邊處理必須由使用者找出一特別的區塊；而當然有些並不用如此。然而，總是要找出記憶體來用的。

爲了降低這種支出，一些較複雜的線上同時周邊處理如 PrintQ 和 Printer Genius，利用硬碟的空間來代替隨機存取記憶體的儲存空間。

這並不是像 DOS 的 PRINT 指令，或是其他類似的必須添加各種擴展板的線上同時周邊處理，一般較好的線上同時周邊處理可讓使用者設定優先權，使得資料不一定要是先進先出，且此種順序可在執行中間改變。有些甚至提供如編修,格式化等功能。

另外一較貴的設備提供額外的記憶體 —— 在使用者的電腦和印表機之間提供一個緩衝區。雖然此緩衝區除此功能外不會再被用到，但這卻也不會添加支出。從電腦的觀點來看這是隱形的，使用者送一個檔案至印表機，而緩衝區先容納其最大容量，當印表機的內部緩衝區空出來時，此緩衝區再以小包裝將資料送出。

這種緩衝區的大小由 256 K至好幾 megabytes的檔案（其中許多在設置時爲其最小容量，依需要再擴充）。外接板的緩衝區並不會去判斷說它所接收的輸入是直接由原始檔而來，或是中間經由別人修改過了。

外接板 (outboard) 印表機緩衝區的最可能敗筆在許多情況下，因爲電腦和印表機之間並沒有直接的溝通，使用者並不會接收到一般所收到的錯誤訊息，如印表機沒開等。緩衝區通常蠻笨的。它只會一直接收資料，一直收、一直收，直到有人發現有問題發生了。但最糟的是通常總是很久以後才有人發現到有問題。

然而，外接板是一個便宜又有效的方法來增進生產力，且在多情形下可加上多使用者網路。

11.6　週邊設備共用

BONANZA 堡國 (R)

通常當緩衝區擴充到可包含一整套設備時，稱爲週邊設備共用。

許多較複雜的設備，價錢最高可達幾千美元，可讓使用者任意連接電腦和印表機，只要 I/O站可容納得下。這些 I/O 站的差別很大。例如一個有六個站的單位可供應兩臺電腦，共用四台印表機，或是四台電腦共用兩台印表機，或是兩台電腦共用三台印表機和一台繪圖機。

同樣的站不但可同時作輸入和輸出，有些 (如某些 modem)可依需要來建構爲串聯或並聯。並聯輸入串聯輸出，或是串聯輸入並聯輸出並不會少見，這使得整個系統建構上更富彈性。

外部緩衝器或週邊設備共用，不管是串聯或並聯，其中最大的好處即是並不會佔用擴充槽，或增加電腦的負擔，且

在多使用者系統中，甚至可連接不相容的電腦的共同印表機上。通常也包括了差不多好幾 megabyte 的緩衝區容量，但在初設時也是先設較少的容量。優先順序也可設定，而內部緩衝器則使所有的印表機保持可執行的狀態。

11.7 結論：近似是否夠好？

本章只討論了如果增進生產力而又不需做太多升等工作時的一小部分而已。在許多情況下這一小部分已足夠滿足使用者的所有需要了，且不僅指現在而言，未來亦是。然而所有我們所討論的，都不是眞正的多功。但在差不多八十美元的批發價格下可買到的 Carousel ，使用者已可獲得一相當不錯的線上同時周邊處理。這第一步並未太差。

第十二章　多工系統的實際運作

在往 386發展的過程中，發生了一件有趣的事，就是出現了貨眞價實的多工 (REAL HONEST-TO-GODNESS multitasking) 。這種多工是建立在 8088 系統之上，可執行兩個、三個、四個、六個程式，連大程式也不例外。這並不只是本文的切換 (context switching)不像 Windows。Windows 在當時能夠執行多工，但是這些多工程式連同 DOS 以及位址高於 DOS的程式，所佔用的記憶體仍然不能超越 640K 的限制。

這是眞實的多工作業 (multitasking) ---一部低速的 PC能在前景 (foreground) 或背景 (background) 的模式之下執行程式碼。當然， 286系列機種也能如此，而且由於 286 的速度較快，故執行的時候也比較順暢。無論如何，使用者所需的配備只不過是一部 PC 以及額外的記憶體，另外再加上一套 Quarterdeck公司所出版的軟體-- Desqview 即可。值得一提的是， Quarterdeck是最近才崛起的軟體公司。

Quarterdeck 公司的 Desqview 挾其多工能力而來，再與 AST的 EEMS (Enhanced Expanded Memory Specification --加强型擴充記憶體規格) 配合之後，記憶體的藩籬就隨之而瓦解了。多工作業的時代已經來臨-- 說得過份一點，這個時代是由激進派的軟體公司 Quarterdeck 及硬體銷售商所帶來，剛開始的時候，並不是每個人都能瞭解這是怎麼回事，因爲 multitask這個新名詞才出現不久。

AST 在當時推出了一次聰明的產品宣傳，他們開始在 EEMS擴充板上附贈 (bundling) Desqview這套軟體，這種作法的確很高明。因爲 Desqview 配合 EEMS 擴充板執行時，若要執行更多的程式，就必須添加更多的記憶體，而更多的記憶體則表示更多的 EEMS 擴充板。當然，使用者可以單獨購買 Desqview 執行多工，而不必使用到 EEMS 擴充板所提供的記憶體，(Desqview可以在普通的 3.2版 EMS擴充板上執行。) 然而， Desqview 仍有其極限，能夠同時執行的多工程式，其大小總和不能超過 640K 。也就是只有在同一時間內能放入傳統 640K 之內的程式才能執行多工。當然，上述的極限是由於沒有 AST的 EEMS 擴充板所造成；若配合一片或多片的 EEMS 擴充板執行，就能克服此極限。

如果讀者還記得第五章所提到的內容的話，就可知道 EEMS是當時 LIM 3.2版 EMS規格的變種。 EEMS 完全遵守 LIM 3.2 的規格，而且所做的動作都是 LIM 3.2所允許的，然而，在此之外， EEMS 還加上了在傳統記憶體上整排切換的功能 (ability to bank switch conventional memory)。加上這項功能的 EEMS 在當時已經是非常先進了，就連它的設計者也無法確切地說出此功能的眞正用途。

然而，此功能即是關鍵所在，而且眞的是一個轉捩點。就是這一點使得整個電腦界對此功能耳熟能詳。很快地，這種把 4個 16K的整批資料經高位址區域的頁框 (page frame)整排切換進來或切換出去的功能，其重要性僅次於 RAM disks （記憶體虛擬磁碟），printer spooler (印表緩衝區)，以及如足球場那麼大的試算表之類的功能。把傳統記憶體的內容移除，在必要的時候才移回去的作法，使整個電腦界完全改觀。

這裏面有些說法就像哥白尼當時提出地球並非宇宙中心的情形相似。雖然在當時被視爲異教邪說，但後來仍證明是眞的。曾經有人敢這麼說，並且證明 CPU並非電腦的中心，它只不過是一個大架構中的服務者 (servant)罷了，此時記憶體管理很快地就突顯出其重要性。

這種啓發所隱含的意義讓許多相信此說法的人感到徬徨，而有些人仍然不相信上述的說法，就好像哥白尼時代那些堅信地球是平坦的人一樣。因此從那個時候，電腦界開始分成兩個陣營：有一群是已經開化而接受此說法；而有另外一群則是墨守成規，仍然與哥白尼時代相信地球是平的人一樣。甚至到目前爲止，在相信 "地球是平的" 這一群電腦界當中，其成員數目及倡導者都令人驚愕不已。當然，IBM PS/2 系列 model 50 及 60 的設計者是屬於其中的特許成員 (charter member)。

要瞭解這些問題的關鍵在於：那些事情並不像軟體在無形中替我們所做的事情，這也正是我們所購買及使用的電腦之設計哲學。此外，實行 (implement)這些新發展功能之簡易性是機器在設計時的直接考慮。在此說些題外話，以先前所提的爲例，使用者可以在 PS/2 model 50或 model 60 的

電腦上把系統記憶體切換出去 (swap out) , 以配合 Desqview及 EEMS (以及現在的 4.0版 EMS)執行多工功能。但只能藉由抑制 (disable)系統記憶體才能達成目的, 所以系統記憶體就完全浪費掉而無法把它加到管理記憶體的緩衝區 (pool of managed memory) 上。在設計上, CPU或其他人都無法在此情況下動到它。

我們拿 AST的 Premium 286爲例, 這跟上述的機型是個完全相反的對照, AST的 Premium 286在主機板上竟然不提供記憶體插座, 眞的建一個都沒有, 從 0k 開始的每一 bit 記憶體都是裝置在擴充板上的管理記憶體。根據他們的說法: 有些基本的硬體問題 (hardware issue) 是無法由軟體單獨解決的。

Intel 公司當然已擁有相當大的既得利益。身爲 LIM 3.2 版的規格制定者及記憶體擴充板製造者之一, 卻沒有在 LIM 3.2 當中包含了 EEMS 的先進功能, 很明顯地, 他們已處於進退兩難的狀態。若承認 EEMS 的規格優於 LIM 3.2, 就好像逼他們吃下毒藥一樣--很少人能承認別人比自己好。然而, 仔細觀察他們的新產品 80386晶片, 他們讓此晶片具有多工處理的能力, 而此晶片在經過適當的利用後, 表現出來的竟像是 EEMS 所提供的硬體支援。在同時, 有一些 Lotus 的客戶對舊版本的 EMS 3.2規格的限制已經開始感到不滿意, 因爲此規格對擴充記憶體的存取極限在實際應用上只能到它所承諾的 8 mega 位元組的一半-- 4 mega 位元組。

Intel 公司挑起了這個擔子。 LIM聯盟制定了一份新的文件: LIM 4.0 EMS規格文件, 其中 LIM仍代表 Lotus/Intel/Microsoft , 但事實上, 這份文件在基本上只

是 AST的 EEMS 套上了 LIM的封面罷了。（註一：根據內幕消息， AST當時並沒有收到這些公司的諮詢要求。）新版的 4.0 EMS 並未包含所有 EEMS 的特性。（然而，有趣的是：它的確指示硬體製造商，並且建議們能與一項未包含於 LIM 4.0 的 EEMS 特性保持相容。）基本上， LIM 4.0的確包含了某些特性，而這些特性都被證明是使得多工能力實現的關鍵。

瞭解 LIM 4.0 EMS規格制定的來龍去脈之後，要瞭解我們現在所處的環境就非常容易了。 4.0版規格並未修改任何規則。合乎舊版 3.2 EMS規格的硬體，在 4.0 EMS的規格下，仍可適用。換句話說， 4.0版的 EMS只不過新增了一些新的特性。因此，任何與 3.2版規格一致的硬體，在 4.0版上一樣可以合乎要求。所以，只要他們不發表他們未支援的特殊功能，我們就不能對 4.0版所發表的支援產生爭辯。總之，支援 EMS 4.0版規格的產品並不一定能支援多工作業。

這樣子是不是有點像欺騙顧客的黑店？的確是的。但有誰規定電腦界不能像其他自動工業，藥品廣告，化粧品，或其他廣告商一樣，不以整個事實來做廣告，只取其有利之處大做宣傳。反正，只要不做誇大不實的宣傳就好了。

12.1　多工作業的眞相

BONANZA 登 (R)

曾幾何時，除了看看標題及大號的粗體字印刷之外，更重要的事情就是仔細看看那些印得細細小小的特殊規格。

幾乎沒有東西能夠因爲理所當然或只因爲門市小姐的微笑而購買，然而 4.0版的 EMS的確可歸於此類。

爲了澄清一個觀念，以免讀者造成誤解，我們在此聲明：多工作業並不是 Deskqview所首創的。多工作業在早期的型電腦時期就已經出現了。而且 Microsoft的 Windowns 也已經達到這種功能，但是都受制於 DOS 640K 的障礙。Desqview所做的只不過是利用了 "使 AST的 EEMS 特別" 的特性：從 640K 至 256K 的傳統記憶體做整排切換 (bank switch) 的能力--在 LIM EMS規格當中所需求的 64K頁框只是一點紅利 (bonus)罷了。由於能夠映進及映出 (map in and out)這 384K 的傳統記憶體，所以事情大致都已經安排妥當了。

很有趣地，舊型的 PC 比現在某些機器更適合執行多工。至少舊式的 PC 能在主機板上設定 DIP開關，以顯示目前系統的記憶體大小爲 256K (註二：在開機的時候，前使者也可以設定爲只有 64K，然後再利用軟體命令指定給系統更多的記憶體--包括眞正裝置在主機板上的附加記憶體。這有時用來當做快速開機的方法，因爲這樣可忽略平常 BIOS 對 64K 以上的 RAM做檢查的工作。) 其實， 256K 也正是舊型 PC 主機板上所能容納記憶體的極限。

設定之後，其他記憶體就能映對到這些由 DIP開關抑制 (disable) 的位址。而且如果加上了 AST的 EEMS 擴充板，這 384K 的擴充記憶體就能從傳統的 256K 記憶 "位址" 開始映進映出。 (註三：雖然，如同前幾章所討論， 256K 的限制並非神聖而不可侵犯，除了最早的 PC 之外， PC 的設計規定：主機板上插滿 256K 記憶體以後，新加入的記憶體擴充板才能定址；更重要的是，欲加下擴充記憶體之前必須填滿 640K 的傳統記憶體才行。 (譯者按：請注意上述兩個敘述的差異。)

容量到達 384K 的程式碼或資資料可以從傳統的記憶位址空間移除，並且在原處把其他 384K 的整個區塊切換進來。那 384K 對大部份的 DOS程式來說已經是綽綽有餘，而 Deskqview 所與眾不同的任務就是：使眞正該執行但是被切換出去的區塊再生 (alive)，把暫時不執行的區塊切換出去，直到這些區塊眞正要執行時才切換進來。

Desqview這種作法使得其他程式能分享 CPU時間，這就是所謂的時間分割 (time slicing)。也許我們看不到這種現象，但無論這種現象在任何處發生，我們仍可從下列的例子看出其作用：假設使用者的試算表正重新計算，而且在此時把此工作切換出去，然後換進另一個工作當作前景程式，那麼此試算表的計算工作仍然在背景之下繼續進行。等使用者把此試算表再切換回前景時，其計算工作大概已經完成了，而在等待的同時，使用者可能利用此空檔執行其他程式而完成其他工作。幾乎所有不需持續觀察其變化或是不需要鍵盤輸入但很耗費時間的工作，都很適合放入背景模式執行。例如在執行通訊軟體時，這種功能更顯示出其價值。(譯者按：對喜歡玩 BBS的讀者而言，都希望在 Upload 或 Download檔案的時候能夠做其他的工作，以免在終端機前等候，此時這種工作即可放入背景執行。)

有很多人會同意下列這種說法，一部 4.77 MHZ 的 PC 若試著把 CPU時間分給許多不同的工作，那麼在比較執行速度時無疑略遜一籌。然而，由於在那個時代處理每一件工作所花的時間都比現在要多，因此對那個時代的人來說，能在背景模式執行度較慢而不必注意執行過程的程式，其代表的意義比現代桌面上的速度較慢競爭更深遠。顯而易見地，這很快就引發了一場席捲市場的革命。當 80386上市的時候，

Desqview這份炙手可熱的軟體早就爲這新一代電腦準備好了，而且配上了 80386的 Desqview 變成了更快速、更順暢，並且更具威力的工具。

12.2 建立更佳的捕鼠器

BONANZA 螢圖 (R)

-- Desqview

現在 Desqview 仍然能執行那些令它名聲大振的任務，即使在 AST或 Quarterdeck尚未嶄露頭角之前即推出的電腦上， Desqview 一樣能執行多工。然而， Desqview 繼續保持其領先的腳步，成爲首先利用 80386晶片重新對映 (remap) 功能的軟體之一，而且首先成爲上市的商業產品，讓終端使用者 (end-user) 得以享用其功能。藉著把 DOS位址之上的某些程式及 TSR程式重新定址 (relocate) 到 64K 以上，使得我們能夠在 640K 以下能夠擁有較多的可用空間，以執行應用程式。

如同以前所述，在 640K 至 1024K之間的位址並沒有裝設任何記憶體。在那裏只有未用的位址。少數被 ROM晶片所占用以及開機時保留給視訊 (video)的一塊除外。 (註四: 此位址對 CPU來說仍是可用的，但這些視訊記憶體是由視訊界面卡或匹配器所提供。在考慮可用的記憶體資源時，視訊記憶體並不視爲系統記憶體的一部份。) 事實上，使得 80386 晶片與其他晶片不同的地方在於: 記憶體能對映到這些位於 640K 以上的未用位址，然後利用以前曾經詳述過的技巧，使得在傳統記憶體中已重新定址過 (relocated)的程式碼能使用這些位址。

然而，Desqview 並不因此而滿足，它在上述這些 "無人之境" 挖出了更多的東西。也許使用者早就聽過有關 64K額外 (extra) 記憶體的事情，這些記憶體在稍早就已有人發現，而且也可以從 Desqview 呼叫。(註五：請讀者參閱第七章當中有關 XMS部份的討論。) 假定延伸記憶體已經裝設完成的話，那麼由於 DOS及整個 8086 家族所使用的 64K 分段記憶架構，以及由此所產生的特殊情形，造成這額外的 64K記憶體在事實上是存在的。(註六：在 80386的 32位元保護模式下是例外情形，因爲此模式之分段可以不是 64k，在實體模式或虛擬 8086 的模式下，80386仍然使用 64K 的記憶分段。)

這個技巧是由於分段 (segment)的起點並不一定要落在 64k 的整數倍位址之上。在 DOS可定址的範圍之內，一個 64K 分段的起點可以從任何位址是 16 倍數的地方開始。至少在理論上位址達到 1024K-1的地方都還能當做分段的起點。只要 DOS能處理這些基底位址 (base address)，我們就不用操心眞正的程式碼到底在何處執行。Desqview 既然發現了這塊 64K新生地，接下來的動作就是把自己的一部份藏身在這塊從來沒有人考慮動用的空間內。

Desqview之所以能在 286系統上使用上述這塊空間，是藉由一個具專利的驅動程式 (device driver)-- QEXT.SYS 所達成的，(註七：Microsoft很明顯地對這個驅動程式心動不已，大約在一年之後也出版了一個相當類似的產品，在當時 MS-DOS 4.x 版的散布磁片上，以未說明的檔案 (undocumented file)爲面目而問世，此檔案剛開始命名爲 HIMEM.SYS，而且在 PC-DOS 的版本上並無此程式。這個檔案在延伸記憶體規格 (XMS -eXtended Memory Specification)當中曾經提到，但只是一筆帶過而無詳細說

明。這份在 1988 年 8月 23 日所發表的文件裏，有關此部份的敘述僅止於： "這個延伸記憶體管理器現由 Microsoft 公司銷售 -- "The Extended Memory Manager currently being distributed by Microsoft" 。) QEXT.SYS爲 Desqview 出貨時的標準配備。 QEXT 以FFFEh爲基底位址(也就是 1024K-32)，此外 QEXT在裝置驅動程式時必須具有優先權，也就是有許多處理延伸記憶體的驅動程式在一起時， QEXT.SYS 一定要先安裝。此關鍵在於上述的那 64K記憶體. 其基底位址恰好位於 DOS 1 mega 位元組的邊緣，所以此驅動程式必須要占有從 1024K開始的延伸記憶體才能動作。而且在那 64K當中不能有任何間隙(gap)，以免造成此程式無法存取連續 64k記憶體。總之，要使這種技巧能發生作用，其先決條件就是 64k記憶體必須連續。

若配合 386系統使用 Desqview ，則 Desqview 提供了另一個截然不同的驅動程式 QEMM.SYS ，然而這個驅動程式必須單獨購買。雖然 QEMM 能夠把記憶體對映到位址間隙內(事實上，此間隙可以小到 4K)，但是可能的話，最好能夠使用較大的連續區塊，使得 ROM或其他可移動的驅動程式能夠放入這些較大的連續區塊內。

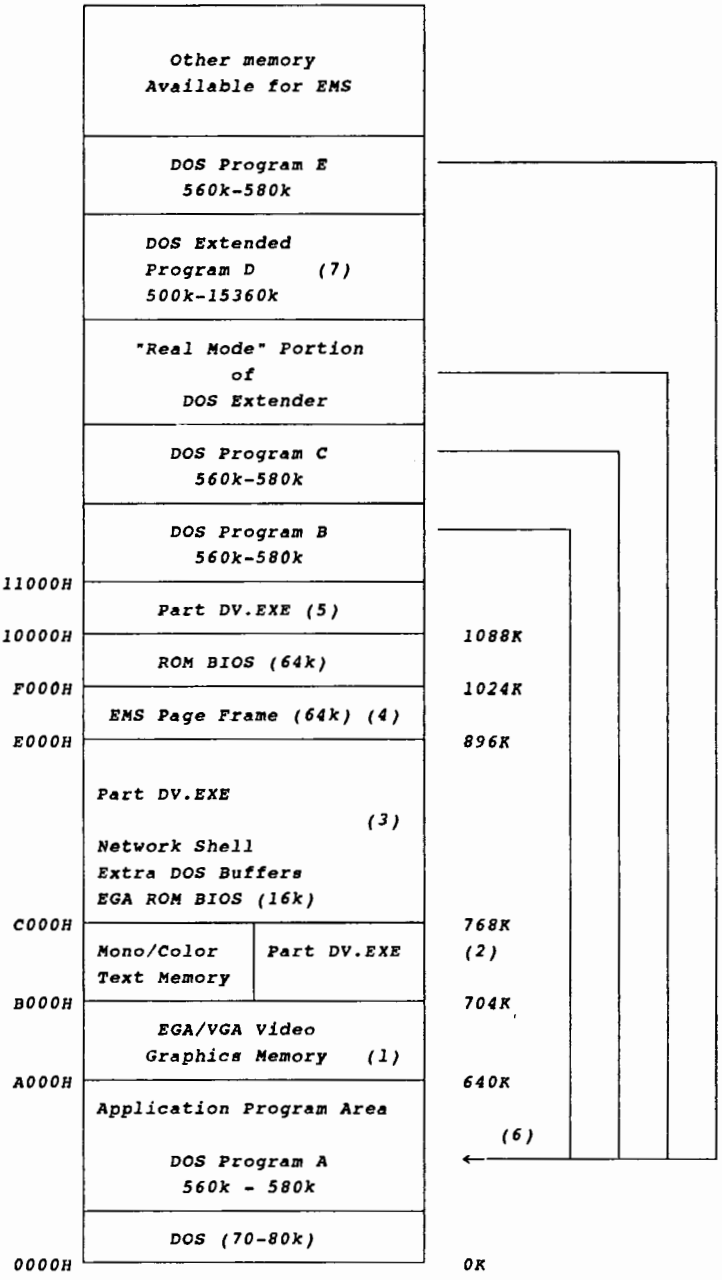

Other memory
Available for EMS
DOS Program E
560k-580k
DOS Extended
Program D (7)
500k-15360k
"Real Mode" Portion
of
DOS Extender
DOS Program C
560k-580k
DOS Program B
560k-580k
11000H
Part DV.EXE (5)
10000H
1088K
ROM BIOS (64k)
F000H
1024K
EMS Page Frame (64k) (4)
E000H
896K
Part DV.EXE
(3)
Network Shell
Extra DOS Buffers
EGA ROM BIOS (16k)
C000H
768K
Mono/Color
Text Memory
Part DV.EXE
(2)
B000H
704K
EGA/VGA Video
Graphics Memory (1)
A000H
640K
Application Program Area
(6)
DOS Program A
560k - 580k
DOS (70-80k)
0000H
0K

圖 12-1 圖示 80386記憶體的使用情形，此爲 Desqview 對 1024K 以上及以下記憶體的典型使用情形。特別注意那些位於視訊記憶體以上、記憶體頁框以下的程式－－ DV.EXE 的一部份，以及其他公用程式。還有那些未使用的視訊記憶體位址 (如果可用的話) 以及位於1024K～1088K的 XMS區段也值得注意。在此圖表的頂端，DOS 延伸程式可以是 Lotus 3或是其他 VCPI 相容的程式，連用在擴充記憶體內執行 DOS 程式。

由於 QEMM 可以隨心所欲地使各種不同的技巧，所以在 386 上執行的 Desqview 能眞正地節省傳統記憶體的使用,而且所節省的數量等於在其之上的程式大小，當然，這也依系統組態裏的內含而定。轉換成有意義的項目之後，在此情形下 Desqview 就能提供使用者多重的多工視窗--而且每一個工作所能佔用的空間就跟使用者未執行 QEMM 之前，單獨執行此工作所佔用的空間一樣。這的確是零開支 (zero overhead)，一點也不耗費記憶體。嚴格說來，這也不算什麼，只要有足夠的記憶體支援這些視窗即可。

QEMM也表現了其他重要的功能，包括了 EMS的模擬--假的 LIM 4.0記憶體 (fake LIM 4.0)。(請參閱圖12-2)。其實也不全然是假的，這些記憶體夠眞實了。只是這些記憶體在開機時並不是擴充記憶體，而是藉由 80386晶片的硬體對映 (hardware mapping) 功能，使得這些在開機時爲延伸性質的記憶體，經由 QEMM 的攔截而讓應用軟體感覺不到兩者的差異。然而延伸記憶體仍是延伸記憶體，而應用軟體也不管它用的是那一種記憶體，只要它 "看" 到的是一整塊位於 640K 以下的記憶體即可。這是利用 "障眼法" 的技巧所達成。

```
Current Mode                  = ON
Expanded Memory Available     = 1552K
Page Frame Address            = E000H

Expanded memory is being used
```

Area	Size	Status
0000 - 9FFF	640K	Conventional
A000 - AFFF	64K	Video
B000 - B7FF	32k	Hign RAM
B800 - BFFF	32K	Video
C000 - C3FF	16K	ROM
C400 - C7FF	16K	High RAM
C800 - C9FF	8K	ROM
CA00 - DFFF	88K	High RAM
E000 - EFFF	64K	Page Frame
F000 - FFFF	64K	ROM

圖 12-2　Quarterdeck爲 386所發展的 QEMM 能察覺386 系統內尚未使用的位址空間，而且有足夠的記憶體可供使用時， QEMM 會把這些記憶體對映到這些位於 1 mega 之下而且可使用的區域內。

12.3　多工作業能做什麼?

BONANZA 螢圖 (R)

我們持續討論的著眼點當然在於多工作業有什麼好處。然而 Desqview 及 Microsoft對多工的作法完全不同，（由於 386版 Window 的出現， Microsoft終於解脫了以前640K之外無法多工的束縛。）所以有必要多做說明。由於 Desqview比 Windows還早達成多工的設計，所以在看看 Windows 如何動作之前，在此多探討一些 Desqview 的內容

。

Desqview處理視窗的方式跟 Windows有點不同，但它的確可開啓許多視窗。雖然預設的個數只有 8個，但是Desqview能同時處理的程式卻高達 50 個。

然而，更重要的是 Desqview 的多工特性，這是Desqview處理最好的部份。例如一片傳眞卡 (FAX card) 或一般的數據機 (modem)配合通訊軟體執行時，可以在背景模式下執行，以下載 (download) 最新的股市行情。事實上，像排序，搜尋，冗長的計算等耗費 CPU時間的工作，由於使用者不必一直觀察其過程，所以很適合在背景執行。然而，別忘了 Desqview 的執行環境是以 DOS及 8086 家族爲基礎。

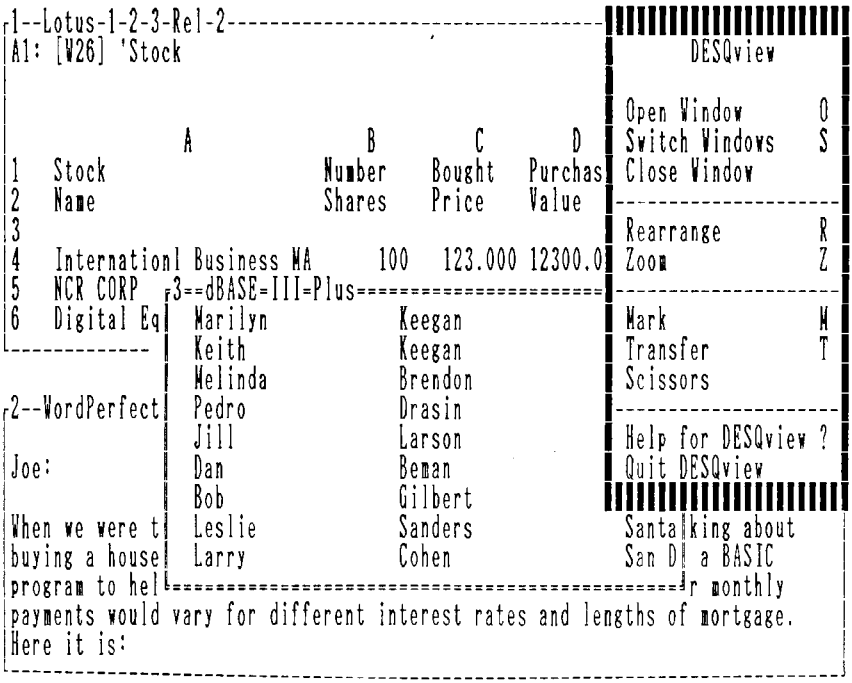


圖 12-3 圖中爲 Desqview 的螢幕展示，以說明其多工的能力。甚至連同時執行的程式，其螢幕輸出也會同時在個別的視窗內顯現出來。對許多應用程式來說，可能會要求使用整個螢幕，而我們可從 Desqview 的選單中選擇此功能。右上角的選單並非下拉式的，但此選單會很快（幾乎是立即）而且自動地出現，視窗內所示爲那一階層下所能選擇的項目。

不過，由於延伸記憶體已經扮演其正確的角色，所以Desqview當中另一項極爲特殊的多工特性很快地就成爲眾所矚目的焦點。

使用 80286或 80386的機器時， Desqview 所能處理的就不只是平常 DOS程式的多工了，它甚至能夠經由 DOS延伸程式 (DOS Extender) 來達成巨大的 386規格程式之多工（這種 386規格的應用程式可深入延伸記憶體內）。這些工作都能在 Desqview 的視窗內或經由其視窗執行——而Microsoft 的 Windows就不敢說它具此功能。

這是一個關鍵的領域，而且很明顯地會愈來愈重要。Desqview不僅做到了，而且對 VCPI (Virtual Control Program interfacr)這個必要界面的發展頁獻極大，因爲有了這個界面，上述的功能才能實現。處理一些平常 DOS程式的多工是一回事；而能夠混合各種操作模式且執行多工又是另一回事。（試想：每一種操作模式都有一套完全不同的操作規則時，其動作之複雜度就可想而知了。

Desqview於此時的任務並不只是 80386時間分割已，它

還要在各種操作模式間切換。例如以 DOS延伸程式爲基礎的程或就處於保護模式; 而所有在 Desqview 控制之下的一般 DOS 程式則處於虛擬 8086 模式 (Virtual 8086 mode)。所以 Desqview 帶來了 DOS延伸程式事業的領導者以及一些延伸記憶體的使用者。這有點 "施與受" 的意味, 但它們也儘量完成一套規則, 使得它們能並存。

在你說: "嘿, 但那對我並沒有影響 "之前, 最好再考慮幾分鐘, 我們所討論的並不只是那些特殊的新程式。我們要討論的還包括了現在你可能會用到的程式--如試算表 (spread sheet), 資料庫 (database) 一桌上出版系統 (desktop publishing), 說得更明白一點, 就是像 Borland 的 Paradox及 Oracle 。 (譯者按: 國內的使用者大概會更熟悉 dBase, Lotus 1-2-3, Excel, 或是雅墨、柏泰等排版系統吧!)

Borland 的產品在這裏恰好可以用來當範例, 因爲 Borland 的 DOS延伸 386版本 (DOS Extended 386 version 在剛上市的時候並不能在 Desqview 上執行。你可以選擇使用 Paradox 386或是在 Desqview 的多工環境下執行其他應用軟體; 但是在同一時間內只能選擇其中之一。不過, 這是因爲當時還沒出現 VCPI 或其他標準介面所致。事實上, 也是由於 Paradox 386這類程式發生了這種問題才引起大家對軟體發展介面的重視。在軟體發展者開始開發80386 的眞正威力及發現新問題的同時, 這種介面的需求也愈來愈受注意。

VCPI的出現就是用來解決這個問題。在此之後不久, Oracle的新版本就開始發售, 以配合 DOS延伸程式讓 32 位元處理器能淋漓盡致地發揮其功能。 IBM的 Interleaf

Publisher 也使用了 DOS延伸程式，而且在市場上已上市或即將出版的軟體也大部份具有 DOS-Extended 的 386版本。這股風潮以迅雷不及掩耳的速度就成爲市場上主要的考慮因素。

事實上， Desqview 非常有彈性，連特別爲 Windows設計的應用軟體--如 Microsoft的 Excel--也能在Desqview上執行。以 Excel爲例，甚至推出了規模較小的 Window 版本，這樣才能載入圖形環境下執行。雖然Desqview不能使 Excel 執行得快一點，（因爲 Window 專用的程式必需在圖形模式下執行。）但是在 Desqview 的視窗內， Excel一樣可正常執行，甚至在 8088 上一樣可以執行多工。

如同使用者所見，自從 Desqview 崛起之後，雖然它成爲，一個愈來愈成熟的軟體，但它卻從未拋棄那些曾經造就它名聲的 " PC族" 。 Quarterdeck 不僅繼續發售產品供 8088系列等低階機種使用；在 80386的機種上，使用者也不必擁有專門的知識即可使用 Desqview 。事實上，就算是完全以 386爲導向的版本，使用者也有選擇的權利。使用者可選擇較簡易或較進階的方式來安裝這套軟體。實際上，大多數的使用者用簡易的方式來安裝會比較好，至少要等到安裝完成並使用一段日子以後才能對此軟體運用自如。使用者隨時可以改變心意而變更 Desqview 先前所假定的預設值。

在最基礎的等級當中， Desqview 假定了一個智慧型的階級，它並不會用滑鼠去迎合那些外表裝飾精美的圖像 (icon)。它只不過是一個命令導向 (command oriented) 系統，雖然它提供了躍出式功能表 (pop-up menu)來提示使用者，但從未降低使用者的效率。就是因爲 Desqview 沒有拉低使用者的效率，故它就大量地使用功能表，幾乎每一個命

令鍵都會使螢幕躍出一個功能表。只要你認爲它有多少親和性它大概就有多少，如果你試了許多命令都不成功，只要按一個問號 (?)就會立即顯示一個輔助功能表。

"立即性" 是 Desqview 的特點之一，這也使得它和 Windows 有高下之分。若要達到 Windows那種看起來很漂亮的畫面，就必須在圖形模或下運作，而這會很花時間。在 Windows 當中所展示給使用者的每一個提示、以及每一個必要的選單，都必需在圖形模式之下一個畫點接著一個畫點地產生。而 Desqview 的文字模式功能表就不必讓使用者有等待的感覺，一旦你想執行某一命令，在你按下按鍵的同時,畫面幾乎是同一時間顯現的，當然，載入程式到視窗的動作除外。在載入及開啟檔案過程中所花的時間，跟 Windows比起來是一樣的。

在 386上，若使用較早期的 Desqview 可能會出現問題，當程式直接把資料寫到記憶體而不經過 BIOS 時就會出現麻煩，而且這類程式爲數不少。這種程式在背景模式下執行的時候並不知道 "螢幕不是它的" ，所以它就直接在影響記憶體下寫入資料，並且蓋掉了前景程式的畫面。說得客氣一點，這種程式眞是一個討厭鬼。然而在 2.2版當中，這個問題就解決了，在 xx-PIF.DVP 的檔案當中加上適當的設定，前景程式就可完全擁有整個螢幕而不受背景程式的干擾。

從外觀來看， Desqview 在某一方面與 Windows有點相似， Desqview 用一種 PIF格式的檔案 (xx-PIF.DVP) 來執行一些常用的程式。不過它跟 Microsoft的格式有很大的不同，在 Desqview 當中的 PIF通常很容易建立，對剛入門的 Desqview使用者來說，它還提供了許多起動檔案以配合許多常用的應用程式。其他的 xx-PIF.DVP檔可以利用 CP

(Change Program)這個功能來設定。 CP 這個功能非常快速，又有易懂的選單提示幫助使用者來設定檔案。此外，也可利用現成的呼叫程式 (call-up program)以同樣的方式修改。

如同 Desqview 的一貫作風，連 CP 功能也分爲兩個等級，以配合使用者的熟練程度。預設的 CP 畫面只包含了非常基本的資訊，有程式名稱、路徑、最小空間需求，使用者自行定義的雙字元呼叫命令 (two-character call-up command)等。在某些空格內，Desqview還預先填入了預設值以配合多數應用程式。若使者願意的話，也可使用批次檔案的名稱代替眞正的程式名稱。

上述只是第一個 CP 畫面，功能鍵 F1 還可叫出一整個螢幕的進階功能。如果使用者想修改的話，可以在空格之內填入你想要的預設值，例如視窗的大小、外觀及位置，最大記憶體，程式是否切換到磁碟上，是否在背景模式下執行等等。如果發生了鍵盤衝突或是類似的事件，還有許多值可以調整。在任何狀況下，一定有足夠的預設值讓使用者起動程式，也就是說，只要用預設值大概就可起動大部份的程式了。

在這些進階的設定當中，對多工執行最具關鍵的就是指定致能 (enable) 或抑制 (disable)背景操作，以及切換至磁碟的 Yes/No 選項。 enable/disable 選項的意義，可以從字面上直接看出；對一些使用者來說， swap to disk (切換至磁碟) 選項所隱含的意義可能比較不明顯。

有一種使用虛擬記憶體的形式，通常是以頁爲要求單位 (demand-paged)，此方式可以令未使用的部份程式碼切換至

磁碟，而把眞正執行的程式碼留在 RAM當中。當 Desqview 做磁碟切換的時候並不具選擇性，只是直接切換。這些被切換至磁碟的程式就被 "冰凍" 起來，直到必要的時候才會從磁碟切回 RAM裡面。

頁需求的虛擬化 (Demand-paged virtualization)這個名詞大概只有在你需要執行很大的程式時，才會有興趣去研究它。這種大程式就像 Phar Lap's 一樣，需要 DOS延伸程式的幫助而在 32 位元保護模式執行的程式。然而，程式本身可能早已進行必要的記憶體管理，即使你已經這一類程式,你還是要告訴 Desqview: "它沒有被切換至磁碟" ，因爲從Desqview的觀點來看，除非使用者眞的要讓程式凍結起來或是程式本身早就提供了 "頁需求虛擬化" 的功能，否則 Desqview不會切換這些程式。

除非程式眞的需要在背景模式下執行，否則使用者應該不會把 "Runs in background" 選項設成 Yes (Y)。因爲如果設成 Y的話，當程式載入之後就會自動地共用 CPU時間,即使程式在背景模式下什麼也沒做。由於時鐘週期 (clock tick) 是有限的資源，所以使用者最好讓程式在眞的需要 CPU 時間的時候才分配給它，以節約這些資源。在背景模式下眞的會繼續動作的程式才需要在背景模式下共用 CPU時間。

Desqview有一個內建的電話撥號器，這個撥號器使人聯想起舊式的 Sidekick 撥號器。若使用者裝有數據機的話,可以從螢幕上選一個電話號碼，使此電話號碼成爲高亮度顯示，然後就可撥出去了。 (當然，撥號的動作是由電腦代勞。) 這種功能在配合某些自由格式的資料庫程式使用時會特別順手，像 Broderbund 公司的 Memory Mate，一個只需

要很小視窗的 TSR (常駐程式)，使得它能夠隨時叫用而不會從主要的應用程式中奪取太多記憶體。

事實上，在使用 Desqview 或 Windows這種撥號功能時，引發了一件有趣的事。只因爲：使用者在多工環境下並不表示一定要捨棄某些利益--像是在視窗之間快速切換，無論視窗大小。有許多使用者因爲在有限的記憶體當中擠進太多程式而使得記憶體顯得擁擠不堪，因而捨棄了大部份有用的常駐程式。在這兩者之間實在很難抉擇，因爲我們要享受這些常駐程式帶來的便利時，也要付出某種程度的犧牲--一些可貴的記憶體。

我們把上述的問題及前幾章提過的 Software Carousel一起提出來。但是爲了那些跳過那章而直接閱讀本章的讀者，有必要在此重述。不只是許多應用程式愈來愈大，就連許多常駐程式也一樣。以 Sidekick 爲例，剛開始對記憶體的需求是很適度的，在 256K 記憶體就算蠻大的時代裏，Sidekick還必需節制記憶體的使用。但是現在的Sidekick Plus就不一樣了，現在的常駐程式竟然大於以前的應用程式，即使使用者能把 Sidekick Plus的大部份放到擴充記憶體，它還是在任何視窗內或多或少佔用了一點記憶體，不論是Sidekick Plus 本身或是在其他程式之下。

事實上，如果你喜歡的的話，在 Desqview 或Carousel之下，可以一個接一個地叫用常駐程式，讓全部的視窗都由常駐程式組成。 (註八： Windows的操作模式有點不同，而且不允許從 "修正記憶體 (modify memory)的程式" 中切換出去。) 也許使用者可以得到所有想要的常駐程式，而不必從主要的應用軟體視窗內拿走任何記憶體。

接下來要討論的就是: 在 Desqview 當中, 一個應用程式的視窗可以多大呢? 這跟使用者所使用的 DOS版本有關, 當然, 系統的組態也有影響。一般說來, 以 80386電腦配合 MS-DOS 4.0版而無任何常駐程式時, 在理想狀況下應該可以到達 530K , 若再加上技巧性地裁剪, 大概可以多 "壓榨" 一點。

還有, Desqview 有內建的 cut-and-paste (剪貼板) 功能--但現今的軟體不是如此嗎? 可是 Desqview 在這方面顯然落後 Windows, 因爲 Desqview 的使用者可能比較不喜歡使用滑鼠來佔用桌面空間。除此之外, Desqview 的功能就相當完備了。

另外讓許多使用者感到便利的就是 Desqview 的巨集功能 (Macro facility) 。不過 Desqview 把它稱爲 script (腳本) , 這種巨集可以利用 learn (學習) 的功能建立, 這種功能就是: 當使用者執行某項操作時, 把這個操作的按鍵順序 (series of keys trokes)記錄並儲存下來, 然後把此操作指定到使用者自定的按鍵上。這些並不是以 ASCII的型式儲存下來。有一個附在 Desqview 上的程式可以把這些按鍵順序轉換成 ASCII格式, 以便使用者能利用文書處理器或編輯程式來編輯這些內容, 這一個公用程式可以把這些 ASCII 格式的檔案轉換爲 Desqview 所用的格式, 以便使用者利用這些檔案。

對許多具有程式設計經驗的使用者來說, Desqview 在說明文件內容記載了許多功能强大的使用者自定特性。在這些最具意義的特性當中, 有一個 DV_PAUSE 可以使多工的效率達到最大。例如: 如果一個在背景模式下執行的程式因爲無事可做而虛耗 CPU時間 (idle), DV_PAUSE 會捨棄

(relinquish)它自己所剩的 CPU時間 (time slice) 。這樣可以讓原本被浪費掉的週期釋出來，供其他在背景模式下執行的程式使用。 (請讀者注意: 這種機制 (mechanism)跟 Desqview在處理一個程式等待按鍵時的 "跳過" 方式 (skip over) 不同。)

對一個含有關鍵程式碼的程式來說，可以在此段關鍵碼之前加上 DV_BEGIN_CRITICAL，這樣 Desqview 就不會讓它失去時鐘週期 (slice out)。當然，在此段關鍵碼的結尾也有一個 DV_END_CRITICAL的常式相對應。

上述及其他兩個常式的組合語言列表已包含在該手冊當中，故 Desqview 可以說是一個高度開發而且精緻的高水準工具。然而，就算你不是電腦高手，也沒有 386電腦，仍然可以使 Desqview 發揮其功能。

第十三章　眞的需要 DOS 4.x嗎?

有一個譏評就是: 隨著案頭 (desktop)科技的發展, 其幅度不再以仟位元組 (K)或是佰萬位元組 (M)來量測, 而是以十億位元組 (gigabytes)來測量以後, 使 DOS這隻眞正的 "恐龍" 必須設法殘存。然而 DOS不僅存活了, 而且以其與生俱來的低速 8位元, 1 mega 位元組作業系統的身份, 在市場上日漸茁壯。那時候所配備的記憶體大小, 以及電腦的執行速度, 在現代的眼光看來都有點不太合理, 尤其是記憶體的限制。既然 DOS已經愈來愈茁壯, 甚至獲得更多的力量, 在事實上它不僅是一個作業系統, 還是一個投擲墊、跳板。

的確, 以作業系統角度來看, DOS的功能並不完全, 而且有些笨拙, 有點遲鈍, 最重要的是其限制--限制得非常嚴格。然而, 當我們突破了 DOS所控制的 1 mega 限制時, 這些限制已經解決了一些。 DOS無法控制的部份就留給其他的程式--這類程式通常是一組設計好的新規則, 以合乎新技術的需求, 甚至是那些我們還無法預期的新技術。隨著我們超越這些限制, 這些限制就被拋得遠遠地, 就像太空梭脫離地

心引力一樣。

DOS 4.0 是首次正式承認擴充記憶體存在性及合法性的版本。同時它也爲了解決使用擴充記憶體所產生的特殊問題，首次引進了一些特殊功能。但使用者需要嗎? 或者說，使用者會採用嗎? 或者爲了堅持 "舊的就是好的" 而繼續採用 Microsoft 的產品?

對第一個問題的回答，可以斬釘截鐵地說 "不! "，除非使用者的電腦是一部配有原廠 IBM擴充記憶體界面卡的 IBM PS/2。至少在剛開始時， IBM以相當不同的方式來設計 LIM 4.0 的 EMS規格，這跟其他廠商的產品有很大的差異，並且產生了初始化的相容問題。而爲了配合這些 IBM擴充板所寫的驅動程式是附加在 IBM 4.x版的 DOS之上。

除了上述情形之外，使用者並不需要 4.x版的 DOS，或其他任何特殊版本的 DOS。 (註一: 有些軟體特別需要較高版本DOS 的原因是因爲它會存取到舊版本 DOS未具有的特性。使用者必須注意並觀察任何有關此類的規格。然而，這些較特殊的DOS 版本是爲了某些軟體的需要，而非爲了一般性的擴充記憶體存取及使用。) 雖然在版本升級時，還包括其他考慮因素，但大致上說來，使用者可以利用 DOS 2.0以上的版本來存取並使用擴充記憶體， (註二: 像 VDISK這類特殊的軟體可能需要較高版本的 DOS，否則無法存取或使用延伸記憶體; 但是能自行處理延伸記憶體之程式則例外，因爲所有在延伸記憶體內的事件， DOS都管不著，而且也只有脫離 DOS管轄才可存取延伸記憶體。此時使用者就可以不必考慮所用的DOS 版本，只要軟體本身有辦法在必要時回到 DOS 處理輸出入(I/O)即可。)至少到 DOS 3.x系列版本爲止，許多新加入的內部程式及外部程式 (如 XCOPY, REPLACE 等等

），連同舊有程式的改進及增强，使得 DOS的功能愈來愈强，這也是不一定要更新版本的原因之一。

除非你的電腦是 386系統，而且能把記憶體對應到640K以上的未用位址空間，否則從 3.3升級到 4.0的代價就是多付出 18K (17,480位元組) 的可用工作空間，這樣使得應用程式的空間相對地減少。如圖 13-1 及 13-2 所示。(註三：這個比較是在 IBM PS/2 MODEL 30/286 上所做的,這種數據會隨著 PC-DOS 及 MS-DOS 之間的不同，以及各種特殊版本的不同而有些微的差異。這些只不過是代表性的數據。此外，也請注意觀察 4.0版的 CHKDSK 提供了額外的資訊，在 3.3 版之前並無此資訊。) 即使你不使用新的 DOS shell功能-- "Presentation Manager" (畫面管理器)，光是一個 shell(殼層，請參閱第一章之註解) 就會再佔用9800位元組，將近 10K的記憶體。

```
Volume DOS_33      created Oct 25, 1990 1:17p

 33462272 bytes total disk space
    79872 bytes in 3 hidden files
    73728 bytes in 34 directories
 19458048 bytes in 719 user files
 13850624 bytes available on disk

   655360 bytes total memory
   551440 bytes free
```

圖13-1　DOS 4.x 版之前典型的 CHKDSK 報告，找不到有什麼地方出問題。此時所做的事較少。

```
Volume Serial Number is 241E-10D0

 21204992 bytes total disk space
    71680 bytes in 2 hidden files
   147456 bytes in 65 directories
 17014784 bytes in 1407 user files
    12288 bytes in bad sectors
  3958784 bytes available on disk

     2048 bytes in each allocation unit
    10354 total allocation units on disk
     1933 available allocation units on disk

   655360 bytes total memory
   551440 bytes free
```

圖13-2 DOS 4.x 版提供了較多的資料，但也顯得較不具親和性。新的選項 /F ，如果在所有拯救磁簇 (cluster) 及串鏈 (chains) 的方法皆用盡之前，就不分青紅皂白地使用它，常會使你欲哭無淚。圖中的編號是 DOS 4.x所新增的，在 format 時會自動替磁片加上一個編號。 (Serial number)。

無論系統的微處理器形式爲何， DOS 4.x在啓動時，不僅會認得擴充記憶體的存在，同時也會利用其中的一些記憶體，以降低系統的負荷。無論如何，它會加入某些擴充記憶體的功能及支援，包括前幾版未曾提供的驅動程式。整個事實並不只是擴充記憶體，即使沒有 shell或 Presentation Manager ，它仍然比舊版本更具親和力，如果讀者願意原諒他們的陳腔爛調的話。 (關於其所宣稱的 "親和力" ，讓我們看看圖 13-3)。

```
C>mode
```

```
Status for device LPT1:
-----------------------

LPT1: rerouted to COM1:

Status for device LPT2:
-----------------------

LPT2: not rerouted

Status for device LPT3:
-----------------------

LPT3: not rerouted

Status for device CON:
----------------------

COLUMNS=80
LINES=25
Code page operation not supported on this device
```

圖13-3　在提示符號下執行無參數的 MODE 所傳回的結果。報告中包括現在的埠 (port)，顯示的設定模式。在 4.x之下，這些資訊是非常易懂而且很有幫助。DOS 大概不想在親和力方面跟其他程式一較長短，所以只提供了少得可憐的利益給一般使用者。雖然使用者有時要對此多花腦筋，但 4.x版看起來眞的加入了更多現有的資訊。

其他有關親和力的證據可在 FORMAT 的命令中發現，使用者看到的是格式化的進度百分比，而非原來所報告的磁柱(cylinder)等等。此外，使用萬用字元刪除檔案時，會出現相對應的目錄名稱，確認使用者欲刪除的檔案之所在目錄。

此外，在其他方面也做了許多改進，這大大地改進了 DOS在傳統上無親和力的形象，但是還有一些尙未改進。

DOS 4.x 的大小比起前幾版又增大了不少，而且在理論上會佔用更多的記憶體空間。(註四: 最近有些軟體對 4.x版的 DOS 允許特別的寬容。以 Software Carousel爲例，當偵測到 DOS版本爲 4.x版時，會自動地把最小的分割大小 (minimum partition size) 從 32K增加到 48K，而在較早版本的 DOS上自動使用較小值－－ 32K。) 然而在某些狀況下， DOS 4.x可能出人意料地給予使用者的應用程式較大的空間，舊版本所能提供的空間反而較小。但使用者不要完全指望這些特殊狀況，因爲事實並不總是如此，我們只要記得 DOS 4.x具有這個潛能即可。新版本的 4.x的設計目的之一就是要比舊版本靈巧。

在理論上 DOS 4.x應該可以深入電腦機殼內，並且偵察使用者的主機上到底裝了哪些設備，然後決定是否依據其發現提供內部的驅動程式。這是一個非常聰明的主意，而且這種時代早該來來臨了，因爲不論是內部或是外部驅動程式都會 "吃掉" 記憶體。在以往任何事物都很單純的時代裏，只有少數週邊裝置，而且只要一些記憶體就可執行程式，所以一點也沒有影響。但是在現在影響就很大了，所以只有我們能多節約一些無謂的記憶體浪費，才能多留一些記憶體空間來執行我們的應用程式。 DOS似乎正嘗試這種作法，但是在朝著這個方向前進的途中，卻又做出了一些相當反常的動作。

讓我們在本章後半部再深入討論 4.x版上極具噱頭的 "殼"--感覺上跟畫面管理器。現在我們把注意力移到4.x 版本身; 它所提供的功能我們以前還沒見過，而且似乎很少有

人預料得到。

13.1　DOS 4.x版所提供的功能

除去擴充記憶體驅動程式以及那個畫面管理器不談，對一般使用者來說，相對於 3.2或 3.3版的 DOS， 4.0版新提供的功能簡直寥寥無幾。我們强調 "一般" 這個字眼，並不是藐視 4.0版所新增的那些有用功能，而是因爲我們發現一個簡單的事實， "一般" 的使用者也許用不到某些在舊版 DOS 內即具有的功能，特別是那些外部功能。

談到 Microsoft的驅動程式 (我們不管那些由 PC-DOS 所提供的驅動程式，特別是那些爲了解決 IBM原廠記憶體擴充板問題而提供的驅動程式。)，他們所提供的功能早就可經由其他來源的驅動程式獲得。至少在前幾版的 IBM PC-DOS或MS-DOS上是如此。事實上 Microsoft的驅動程式並不提供 80386的重新對映 (remap)支援 (因爲在驅動程式內並無設計此支援，而且文件上也未說明。)，然而這種功能早就有其他公司的驅動程式支援了，像十二章所提的 Qrarterdeck 公司 (QEMM)，以及 Qualitas 公司 (386 MAX)，這些公司的驅動程式比 Microsoft所提供的要好一點。

不過以一個嚴謹的使用者來說，除去這驅動程式以後，DOS 4.x 版看起來就不夠份量了。但是 DOS 4.x版的確提供了一些處理擴充記憶體時的工具程式，而且毫不含糊。

MEM.COM 是這些工具程式中最明顯的一個，事實上這也是在 DOS 4.x版內唯一新提供的公用程式。如果只鍵入

MEM 而不加任何參數，則執行結果如圖 13-4 ，提供了系統記憶體的資訊，無論如何， MEM並不因擴充記憶體的內容爲何而有所限制，事實上， MEM所報告的大部份內容多是跟傳統記憶體有直接關連。

```
C>mem
```

```
  285696 bytes total memory
  285696 bytes available
  120752 largest executable program size

 2048000 bytes total EMS memory
  425984 bytes free EMS memory
```

圖13-4 MEM.EXE ，在 DOS 4.x系列當中，唯一新提供的外部公用程式，乍看之下只提供了一些功能，但是在必要時會提供更多資訊。

然而，當 MEM配上了 /PROGRAM 或 /DEBUG 兩個開關之一時， MEM所顯示的資訊則變得有用而不容忽視了。例如，/PROGRAM選項會把載入記憶體當中的程式做一列表 (請參考圖 13-5)，而 /DEBUG 選項會顯示程式、內部驅動程式，以及其他程式上的資訊 (請參考圖 13-6)。請注意，在此的 /DEBUG選項是附屬於 MEM.COM; 而跟 DEBUG.COM無關連。還有就是: 擴充記憶體的資訊只有在系統已安裝 LIM 4.0 EMS 驅動程式時才會顯示出來。

```
C>mem /program
```

Address	Name	Size	Type
000000		000400	Interrupt Vector
000400		000100	ROM Communication Area
000500		000200	DOS Communication Area
000700	IO	0020D0	System Program
0027D0	MSDOS	008E20	System Program
00B5F0	IO	00C9A0	System Data
	INBRDPC	004F00	DEVICE=
	QEMM	000620	DEVICE=
	KSOFT	000C10	DEVICE=
	CLK456	0002D0	DEVICE=
	ANSI	001180	DEVICE=
	QUIKMEM2	000300	DEVICE=
		000820	FILES=
		000100	FCBS=
		003E70	BUFFERS=
		0008F0	LASTDRIVE=
017FA0	COMMAND	000040	Data
017FF0		000070	Data
018070	FASTOPEN	002770	Program
01A7F0	COMMAND	001640	Program
01BE40	COMMAND	000100	Environment
01BF50	DV	0000D0	Environment
01C030	APPEND	001E20	Program
01DE60	VRETTE	0000E0	Environment
01DF50	VRETTE	0044B0	Program
022410		0000E0	Data
022500	MODE	0001E0	Program
0226F0	DV	0001F0	Program
0228F0	DV	006200	Data
028B00	COMMAND	000110	Data
028C20	COMMAND	001640	Program
02A270	COMMAND	000100	Environment
02A380	MSDOS	000040	-- Free --
02A3D0	DOSKEY	0000E0	Environment
02A4C0	DOSKEY	000E90	Program
02B360	COMMAND	000040	Data
02B3B0	COMMAND	0000D0	Data
02B490	SHELLB	000E80	Program
02C320	COMMAND	001640	Program
02D970	COMMAND	0000C0	Environment
02DA40	MEM	0000D0	Environment
02DB20	MEM	012F00	Program
040A30	MSDOS	05F5A0	-- Free --

```
654336 bytes total memory
654336 bytes available
468144 largest executable program size

262144 bytes total EMS memory
262144 bytes free EMS memory
```

圖13-5 MEM 的兩個選項之一，可以較詳細地報告・・的狀態。 /PROGRAM 顯示了傳統記憶體體的使用情形。但是在擴充記憶體方面，只提供了剩餘擴充記憶體及所有擴充記憶體大小的資訊，而不包括使用情形。

```
C>mem /debug
```

```
Address     Name          Size      Type
------------------------------------------------------------------------
000000                    000400    Interrupt Vector
000400                    000100    ROM Communication Area
000500                    000200    DOS Communication Area

000700      IO            0020D0    System Program
                CON                   System Device Driver
                AUX                   System Device Driver
                PRN                   System Device Driver
                CLOCK$                System Device Driver
                A: - C:               System Device Driver
                COM1                  System Device Driver
                LPT1                  System Device Driver
                LPT2                  System Device Driver
                LPT3                  System Device Driver
                COM2                  System Device Driver
                COM3                  System Device Driver
                COM4                  System Device Driver

0027D0      MSDOS         008E20    System Program
```

```
00B5F0      IO            00C9A0     System Data
              INBRDPC     004F00      DEVICE=
              QEMM        000620      DEVICE=
              KSOFT       000C10      DEVICE=
              CLK456      0002D0      DEVICE=
              ANSI        001180      DEVICE=
              QUIKMEM2    000300      DEVICE=
                          000820      FILES=
                          000100      FCBS=
                          003E70      BUFFERS=
                          0008F0      LASTDRIVE=
017FA0      COMMAND       000040     Data
017FF0                    000070     Data
018070      FASTOPEN      002770     Program
01A7F0      COMMAND       001640     Program
01BE40      COMMAND       000100     Environment
01BF50      DV            0000D0     Environment
01C030      APPEND        001E20     Program
01DE60      VKETTE        0000E0     Environment
01DF50      VKETTE        0044B0     Program
022410                    0000E0     Data
022500      MODE          0001E0     Program
0226F0      DV            0001F0     Program
0228F0      DV            006200     Data
028B00      COMMAND       000110     Data
028C20      COMMAND       001640     Program
02A270      COMMAND       000100     Environment
02A380      MSDOS         000040     -- Free --
02A3D0      DOSKEY        0000E0     Environment
02A4C0      DOSKEY        000E90     Program
02B360      COMMAND       000040     Data
02B3B0      COMMAND       0000D0     Data
02B490      SHELLB        000E80     Program
02C320      COMMAND       001640     Program
02D970      COMMAND       0000C0     Environment
02DA40      MEM           0000D0     Environment
02DB20      MEM           012F00     Program
040A30      MSDOS         05F5A0     -- Free --
```

```
654336 bytes total memory
654336 bytes available
468144 largest executable program size
```

```
Handle      EMS Name      Size
------      --------      ------
  0                       000000
  1                       030000
  2                       010000
  3                       008000
  4                       024000
  5                       038000
  6                       01C000
  7                       07C000
  8                       004000

262144 bytes total EMS memory
262144 bytes free EMS memory
```

圖13-6 MEM/DEBUG 能更做得深入，它能顯示/PROGRAM選項所顯示的全部資訊，此外，對裝設的內部驅動程式都加上了特定的名稱（在 000700 處）。更重要的是，它顯示了現在已配置空間的權杖（handle）及其配置空間大小，還有未配置的擴充記憶體大小，這樣能詳細地顯示擴充記憶體之使用情形。

由上述的範例說明，使用者可以看出 MEM提供許多有用的資訊，特別是對那些熱中於從系統中壓榨出更多記憶體的人們。事實上，像 Desqview 的 QEMM 或是 Qualitas 的 386 MAX 等擴充記憶體驅動程式，都附有類似的公用程式以獲取這類資訊，而在早期的 DOS版本下，那些人就是利用這些程式來幫助他們向系統壓榨更多空間。

然而值得注意的一點是：在這類軟體當中有些雖能在舊版本的 DOS上執行，但到了 DOS 4.x版以後就無法正常地執行其應有的功能。請看圖 13-7 ，此爲公用程式 PCMAP在

MS-DOS 4.01 之下執行所得的螢幕傾印結果。在圖中的 unknown 顯示此程式無法從環境變數取得傳統記憶體中各記憶區塊的使用情形。（譯者按：此處所指的環境變數跟 DOS 提示符號下，由 SET顯示出的環境變數有點不一樣。）

```
C>pcmap
```

```
PCMAP 1.0 (c) 1987, Ziff-Davis Publishing Corp.
Segment Paragraphs   Bytes    Program
1808H      0277H      10096      COMMAND.COM
1A80H      0174H       5952      (Unknown)
2251H      001EH        480      (Unknown)
1DF6H      0459H      17808      (Unknown)
1C04H      01E2H       7712      (Unknown)
2270H      064CH      25792      (Unknown)
2290H      000EH        224      (Unknown)
28C3H      0189H       6288      (Unknown)
2A39H      0004H         64      (Free   )
2A4DH      00F7H       3952      (Unknown)
2B4AH      00E8H       3712      (Unknown)
2C33H      017DH       6096      (Unknown)
2DB4H      7258H     468352      (Unknown)  (Free space)
```

圖13-7　原來用於舊版本 DOS的 PCMAP可用來指出程式當掉的原因，在 DOS 4.x版之下，卻無法發揮其作用，以指出哪些位址是由哪些程式佔用。這說明了在舊版本 DOS所寫的程式，有時無法在 4.x的版本下正常執行。

在擴充記憶體方面所能提供的資訊，除了可經由其他公用程式獲得以外，事實上， 4.x版的 DEBUG也加強了許多新的功能以處理擴充記憶體，而且這些功能可能是其他同類程

式所缺乏的。在 DEBUG的提示符號下，有 4個新的雙字元命令可供輸入，在圖 13-8 當中即展示了 XA 命令的使用情形。(XA的用途在於配置記憶頁 (pages allocation) 。)

```
C>debug
-XA nn

Handle created=xxxx
```

圖13-8 利用 DEBUG的 XA 命令，從擴充記憶體內配置記憶頁，圖中的 nn 爲所需的記憶體頁數，若配置成功的話，"Handle created" 的訊息會出現，其中 xxxx爲 handle 的代碼。

其它三個有關擴充記憶體的 DEBUG命令如下 (註: 括號內爲所需參數) :

→ XD (handle by number) 歸還 handle 所配置的記憶體。如果動作成功則會顯示 "Handle xxxx deallocated"的訊息。其中 xxxx是 handle的代碼。

→ XM (logical page physical page, handle) 把特定的 EMS HANDLE 所屬的邏輯頁 (logical page) 對映到實體頁 (physical page)上。如果動作成功的話，會顯示"Logical page nn mapped to physical page nn" 。

→ XS (不需任何參數) 顯示 EMS的使用情形，包括已開啓之 handle 及 handle 所配置之記憶頁。同時也列出了已配置的記憶頁數和可配置的記憶頁數。另外也有實體頁的頁框分段 (frame segment)。

在這四個命令當中，對大多數使用者來說，XS命令也許最有用處，因爲它只傳回使用者系統狀態的資訊。（如圖13-9所示）其他的命令都有特定的用途，除非你認爲你是一個喜歡實地演練的程式設計師，否則這些都會由你所使用的軟體來代勞，而不用你的協助或干涉。

```
C>debug
-xs

Handle 0000 has 0000 pages allocated
Handle 0001 has 000C pages allocated
Handle 0002 has 0004 pages allocated
Handle 0003 has 0002 pages allocated
Handle 0004 has 0007 pages allocated
Handle 0005 has 000E pages allocated
Handle 0006 has 001F pages allocated
Handle 0007 has 0007 pages allocated
Handle 0008 has 0009 pages allocated
Handle 0009 has 0011 pages allocated
Handle 000A has 0001 pages allocated
Handle 000B has 0016 pages allocated

Physical page 00 = Frame segment E000
Physical page 01 = Frame segment E400
Physical page 02 = Frame segment E800
Physical page 03 = Frame segment EC00

7E of a total 7E EMS pages have been allocated
 C of a total FF EMS handles have been allocated
-
```

圖13-9　利用 DOS 4.x版 DEBUG的 XS 命令來觀察EMS 的使用狀況。在此範例中的系統，雖然還有很多handle尚未使用，但是已使用的handle已經配置完所有可

用的 EMS記憶體。

整體看來，不管它體積較大的缺點， 4.x版還是執行得較快。 Microsoft已使用了示同的緩衝區處理法，以加速磁碟存取，而且這次也在其他方面做了一些改變。

除了上述所提的內容以外，對一般使用者來說，舊版本跟新舊本的 DOS之間就幾乎沒什麼差異了。然而，有一點值得特別注意： DRIVPARM 這個選項在 3.2版被引進，但 3.3版又被廢除，而 4.x 版再度恢復此選項。 (DRIVPRAM選項是內部驅動程式所用，以支援實際的磁碟。) 這允許 DOS對高密度磁碟提供支援，以配合那些 ROM BIOS無法提供這類支援的舊機種。

接下來談的是較具技術性的話題，早期的版本包含了一個有趣的驅動程式 HIMEM.SYS，由於這個驅動程式太晚加入版本內，所以文件內並不包含 HIMEM.SYS的說明，只好把說明放在 README.TXT 之內。這個驅動程式有趣之處在於它允許延伸記憶體最底端的 64K能被 DOS使用，這樣就比我們掛在嘴邊的 1 mega 限制還多出了 64K。不過這只適用於 286及 386 系統，這跟 Desqview 擁有的 QEXT.SYS 相當類似。 (註五： Desqview 早在 Microsoft的版本之前就使用了這額外的 64K記憶體 (從 FFFEh開始)，而且已經用了一段不短的日子。)

另一個新的可安裝驅動程式爲 EMM386.SYS ，允許在 80386 系統上的 MS-DOS 程式以延伸記憶體模擬擴充記憶體。此驅動程式能支援 LIM EMS 4.0 (Lotus, Intel, Microsoft Expanded Memory Specification 4.0)。很明顯地，這個驅動程式也是後來才追加的，所以在早期的

MS-DOS版本上，只有 README.TXT 才有說明。這個驅動程式的功能有點類似 Desqview 的 QEMM.SYS 及 Qualitas 的 386 MAX ，這兩個軟體在本書的其他部份都有詳細的討論。然而 EMM386.SYS 缺乏後兩者所提供的對映支援 (mapping support)，此支援是用來顯示其他位址的使用情形，特別是 640K至 1024K。

IBM 所擁有的 4.x版有他們自己的 286及 386驅動程式，稱爲 XMA2EMS.SYS及 XMAEM，而非上一段所述的名稱。當然，這個版本上的傳統 IBM風格的 BASIC及 BASICA 只能在原廠的 IBM機器上執行，因爲只有原廠機器上的 ROM才有 BASIC 的核心 (kernel) 。

MICROSOFT 宣稱 4.x版系列將在往後的版本上逐漸增進其執行速度。 (譯者按: 然而 MS DOS 5.0 版的迅速出現似乎代表著另一層意義。) 他們們雖然一直強調這些尚未完成的功能，但事實上，至少剛開始時， MS-DOS 4.x 版的散布磁片(distribution disk) 上明顯地包含了某些未說明的功能，所以他們已經對即將到來的事物留了伏筆。

13.2 畫面管理程式

BONANZA 螢圖 (R)

(Presentation Manager)

許多使用者對 DOS所提供的使用介面早就抱怨不已。然而，不可否認地，比起 CP/M 時代那些笨拙的語法規則，DOS還是有其值得讚賞之處，至少對那些較不在意的使用者來說是如此。當然，還有 Microsoft Windows，它提供了一個視窗，圖像導向的介面 (icon-oriented interface)，我

們會在本章後面做更深入地討論。

OS/2承諾了一個 "Presentation Manager (畫面管理程式)" 的視覺介面。但你不必等候了, 因爲 DOS從 4.x版開始也有一個畫面管理程式。不過這是可選擇的, 讓使用者仍然能選擇他們以前慣用的執行環境－－開機後是一個傳統且毫無人性的畫面, 外加一個閃爍的游標, 而且在鍵入命令時, 無論是語法或拼字都必須正確無誤。 DOS的 shell或者說是畫面管理程式的出現, 對此問題在貧乏的畫面處理及 Windows 型式的作法之間, 提供了一個折衷的辦法。

然而, 這還是存在一個問題。如同先前所提, 這個 shell 佔用了超過 9800 位元組的寶貴記憶體空間, 而且它佔用的是傳統記憶體, 這些傳統記憶體原本是用來執行我們的應用程式。你可以退出這個 shell, 但是這個 shell大概只會歸還一半的記憶體, 當再度呼叫這個 shell時, 它又會偷走一些記憶體, 而且再也不歸還了。 (註六: 這個結果的根據是原始 4.0版上市後一年左右所發行的版本。)

在專注於如何替應用軟體爭取更多可用記憶體同時,DOS竟然再一次提供了一種程式, 讓使用者的可用記憶體變得更小。不像 DOS 4.x本身, 畫面管理程式無法縮小它那龐大的體積就 "噗通" 一聲, 掉進使用者可用的記憶體空間內, 賴著不走。

事實上, 這個 shell所有的功能都能利用鍵盤來執行,就像 Window 一樣, 但是用滑鼠會更簡單、更快速。 (註七: 在此所指的鍵盤是指 IBM加强型鍵盤, 有 12 個功能鍵的那種, 因爲有些功能所對應的功能鍵, 在 84 鍵的鍵盤上並沒有。) 然而使用者要注意的是: 不論你的滑鼠按鈕是單

鍵、雙鍵，甚至三鍵，這個 shell只會用到一個按鈕。在桌面上移動滑鼠以控制螢幕上的指標，以代替方向鍵的控制；當然，沒有滑鼠裝置的使用者仍然可以使用方向鍵來操縱。要選擇某項功能時，只需按一下滑鼠鈕，或是按下鍵盤上的 Enter 鍵即可。

爲了讓讀者更明瞭，在此我們必須指出：雖然許多使用者在 AUTOEXEC.BAT 的檔案中加上了 DOSSHELL 的命令以呼叫畫面管理器，但是它一樣能在 DOS的提示符號下呼叫出來。如果你想用它的話就可以呼叫它，而不必放棄任何東西。然而，在某些狀況下不應該去呼叫畫面管理器，例如從應用軟體內利用 dos shell功能暫時回到 DOS時，在此狀況下，這個應用軟體本身就已經是一個 shell，所以最好不要在此時呼叫畫面管理器。

13.3 樂園內的問題

BONANZA 螢圖 (R)

(Problems in paradise)

在使用 4.0版的時候，就已經發現了一些明顯的問題，特別是配合某些 DOS的公用程式使用擴充記憶體的時候。只有在 4.x版上使用 BUFFERS= ，以及 VDISK等程式，並加上 /x選項時，才會把這些程式載入擴充記憶體內，而非傳統記憶體。

至少在早期的 4.x版上， /x 選項會奪取 2個 EMS可對映的實體頁 (mappable physical pages)以讓 DOS獨享，但是它卻無法適當地認出它所奪取的記憶頁。另外， /x 選項在某些情形下，會不明不白地盜用了主機板記憶項端的

32K, 這樣就會發生問題, 也製造了系統當掉的可能性。

不過, 這個問題並不嚴重, 因爲使用這些命令時不要加上 /x 選項 (或是其他選項) 即可避免。 (註八: SETUP的過程中, 如果選用了 max DOS functions的選項, 在新的CONFIG檔案內會自動加上一行 ANSI.SYS/x , 而 FASTOPEN雖然也加入了, 但是並無 /x 選項。) 然而, 有趣的是 DOS竟然製造了這種問題。

4.0 版在剛開始的時候也存在著不相容的問題--幾乎在任何新的主要版本出版時都會發生這種問題。 (譯者按: 在此的主要版本是指 2.x, 3.x, 4.x這種大幅度改進的版本。) 即使當時最新版的 Lotus 1-2-3在某些機器上, 對這個突然竄出的 DOS版本也是毫無辦法。事實上, 在某些使用MS-DOS 4.0的機器上, 連載入 Lotus 1-2-3都不行。於是有許多軟體發展者匆匆忙忙地出來解決這些問題, 使得它們的軟體能跟 4.x版相容, 同時也能繼續保有舊版本 DOS的相容性。

雖然這種層出不窮的不相容的問題大部份都出現其他公司所發展的各式各樣軟體上, 而且很容易就解決, 但是還有一個必須注意的問題就是: 這樣會妨礙檔案的存取。

資料壓縮技術可說是一個非常有用的工具, 它可把更多的資料壓至同一空間。有一個相當受歡迎的壓縮程式叫Squish (壓榨), 這個程式能不著痕跡地壓縮並解壓縮資料, 也就是說使用者看不到這個過程。經過壓縮的檔案看起來就不像原來未經壓縮的檔案了。 ASCII格式的文字檔壓縮後就像試算表或資料庫檔案--看起來就像一堆毫無頭緒的符號。很不幸地, Squish 就跟 4.x版 DOS不相容。因此, 當

你試圖在載入壓縮過的應用程式前使用 Squish，DOS會顯示一個錯誤訊息，在你隨著螢幕的指示往回追蹤之後，就可得知 Squish 跟 DOS 4.0 (或者是其他 Squish 不喜歡的任何版本) 不相容。

這使用使用者及那些壓縮過的檔案，還有這個用來壓縮檔案的軟體都束手無策。至少在你堅持以任何代價使用 DOS 4.0 的時候是如此。很幸運地，有一個簡單的解決辦法，這個辦法不是叫你把 Squish 丟掉，而是利用下列步驟來解決問題:

(1) 從 2.x或 3.x版的 DOS磁片起動系統。
(2) 把資料檔案解開壓縮。
(3) 把這些完全解開的資料檔案儲存起來。(至於完全解開的狀況如何，在 Squish 的手冊中有說明--事實上，這本手冊在任何人聽過 4.x版 DOS之前就已經完成很久了。

只要解開壓縮，並且把這些不含 Squish 資料的檔案存回目錄之後，這些檔案就成爲正常而且可用的檔案了。

當然，這表示檔案會比較大，而且如果你的硬碟在檔案解開壓縮前就已經很擁擠的話，你可能要把舊有的壓縮檔存到軟碟，以騰出更多的硬碟空間。爲了讓使用者瞭解得更清楚一點，我們必須指出: 原來發行 Squish 的 Sundog 公司已經出版了 Squish Plus，這跟原來的 Squish 不同，而且其壓縮檔案不能直接通用。無論如何，要點在於使用者可以從軟碟起動系統，以回到舊版本的 DOS環境。即使這樣做只是爲了避免某些特殊的不相容性，這也是一個值得牢記在心的技巧，特別是在任何昇級的狀況下。

13.4 該何去何從呢? (Where do we go from here?)

當然, 在 4.x及舊版本的 DOS之間具有許多不同之處。以客觀的角度來看 DOS 4.0, 它造成的問題似乎比它能解決的問題還多。很明顯地, 這個版本的 DOS是到目前爲止改寫幅度最大, 也最令人猜疑的版本。究竟 DOS有什麼新任務值得現在花費心力來做大幅度的改寫呢?

從許多方面來看, DOS 4.x是一隻奇特的怪獸, 在某些方面, 對舊版 DOS的掌握都顯得有點挫折感。例如在安裝 4.x 版的時候, 有時要用點不按牌理出牌的技巧。特別是早期 IBM的 PC-DOS 4.0 , 例如在許多狀況下, 甚至無法認得以前裝在其他 IBM原廠機器上的硬碟。很不幸地, 這種情形並非很少發生, 第一版的 4.x系列 DOS上有許多明顯的錯誤, 至少其中有些錯誤已經好幾個月都沒有獲得解決。 (譯者按: 若從軟碟起動 DOS 4.0, 尙無法認得以 DOS 3.3爲系統的硬碟。)

還有特別是你要把舊機種電腦上的 DOS版本昇級爲 4.x 時,你需要有原版的 IBM 或 Microsoft 4.x的散布磁片 (distribution disk) , 當然, 磁片的規格必需是你的機器所能接受, 你總不能拿 1.2M 的磁片給 360K 的磁碟機用吧! 但是有了這些原版磁片之後, 並不是直接把內容拷貝到另外一片磁碟上就算了, 4.x的作法跟以前的版本有點不同。當然有方法可以規避這些步驟, 但是並不一定很容易就能找到。

最容易遇到的問題似乎來自使 DOS更 "靈巧" 所花的心力。 DOS更靈巧以後，就可以斟酌你的系統組態，找出最佳的搭配，把不必要的服務及內部驅動程式刪除，讓它的大小能在可接受的範圍之內。事實上，在你眞的要安裝之前，SELECT會提供三個選擇：

→最少的 DOS函數：最大的程式工作空間。

→在 DOS函數及程式工作空間當中取得平衡。

→最多的 DOS函數：最小的程式工作空間。

然而，根據這些選擇所做出來的裝結果，似乎很難看出這三個選擇有什麼差別。在實際測試後，這三個選擇的結果沒有任何差異。 (註九：使用的機器爲 IBM PS/2 model 30/286, DOS 版本爲 IBM PC DOS Ver 4.01) 事實上，選擇第一個項目且不裝設印表機驅動器，或者選擇第三個項目而指定了許多種印表機，所得的結果是一樣的，讀者可以從磁片的剩餘空間完全相同的事實看出來。雖然第三個選項的結果當中， CONFIG.SYS 檔案內所包含的內容，有些是第一個選項的結果當中所沒有的。但是所剩餘的磁碟空間來看，是完完全全地相等，一個位元組都不差。 (譯者按：雖然兩個 CONFIG.SYS 的大小大同，但由於 DOS配置給檔案的磁碟空間是以磁簇 (cluster)爲單位，所以如果兩個 CONFIG.SYS 檔需要相等數目的 cluster時，兩者所占的磁碟空間是一樣的。經過解釋以後，讀者應該對這段內容所產生的疑點有所澄清吧！)

我們再拿 MS-DOS 4.x 版於另外一臺機器上測試，有了

386 配合以後，結果就全然不同了 (註十: 測試的機器爲原廠的 IBM PC ，但是已經加上了 16MHZ的 Intel In Board386, 以及附加的記憶體。) －－事實上，這個結果可以獲得較大的可用工作空間，這比 MS-DOS 3.2 版所剩餘的工作空間還大， (這是選擇第二個選項而不指定印表機的結果)，這樣的結果連 Microsoft都無法解釋，只好讓讀者去猜想!

最佳的建議似乎就是使用原版磁片中，標有 Install的那片磁片，並且讓它去做它的事情。這片 Install磁片可以開機，並且它包含許多檔案，包括安裝程式， SELECT.COM ，在 AUTOEXEC.BAT 檔案中會自動呼叫此檔案。 (註十一: 在 5.25吋的磁片上， SELECT事實上放在第二片－－標有 SELECT那一片，當安裝程式要求時，必須把它放入 A磁碟機。) 在回答問題以後，就可以讓 4.x自行安裝，把原來硬碟上任何版本的 DOS都昇級爲 4.x。 (註十二: 4.0版也可以安裝到軟碟上，事實上，安裝程式在安裝的過程中，會眞正地產生一片可開機的軟碟，這可當成系統的備份。)

只要你的硬碟安裝好 DOS 4.x (或者從軟碟用先前所做的 4.x版開機磁片起動電腦) 以後， DOS 4.x的動作在基本上和前幾版的 DOS是相同的。在你正確地安裝 4.x版以後，你就可以利用 SYS d: 的命令把系統檔案到任何密度的格式化磁片上，以建立開機磁片。 (註十三: 這種作法在早期的版本並沒有說明，但是如果你想從 SELECT 程式的開頭畫面或是稍後的畫面中跳出 (用 ESCape)的話，也沒有關係。只是你必需親自在 A> 的提示符號下鍵入 sys d: 的命令，以安裝系統檔案，安裝完成後，系傳會傳回 System transferred 的訊息。)

在安裝 4.0版的時候，使用者的硬碟上至少要有 1 mega位元組以上的可用空間，這比以前的任何版本都要大很多。很不幸地，安裝程式似乎不是很有彈性－－而且令人驚訝地不完全。例如，要在 4.0版下使用畫面管理器時，AUTOEXEC.BAT及 CONFIG.SYS 檔案內需要有某些特殊的資料。如果這些檔案不存在，則安裝程式會在你的系統上建立它們；若系統上早就有這些檔案的話，安裝程式會建立兩個檔案，一個是 AUTOEXEC.400 ，另一個是 CONFIG.400 ，這些檔案內含有一些命令及參數，必須加至 AUTOEXEC.BAT 及 CONFIG.SYS檔案內，不過這個動作就要讓使用者親自去合併，或者是把這些 .400 檔案的內容抄至 AUTOEXEC.BAT 及 CONFIG.SYS檔案內了。

畫面管理器只能利用 DOSSHELL 的命令呼叫 (不是加至 AUTOEXEC.BAT檔當中，就是在提示符號下鍵入命令。)，這個命令會呼叫 DOSSHELL.BAT批次檔。這個批次檔會依序呼叫 shell，並且傳入必要的參數，如圖 13-10所示。

```
C:
CD C:\DOS
SHELLB DOSSHELL
IF ERRORLEVEL 255 GOTO END
COMMON
BREAK=OFF
SHELLC/MOS:PCMSDRV.MOS/TRAN/COLOR/DOS/MENU/MUL
      /SND/MEU:SHELL.MEU/CLR:SHELL.CLR/PROMPT
      /MAINT/EXIT/SWAP/DATE
END
BREAK:ON
```

圖13-10 以實用的觀點來看，DOS 4.x 的畫面管理器最好用批次檔來呼叫。 SELECT 這個安裝程式會自動建立

AUTOEXEC.BAT (或是上述的 AUTOEXEC.400 及 CONFIG.400等檔案), 內含許多必要的參數, 以符合 4.x版安裝時,使用者所回答的系統配置。

畫面管理器共能由安裝程式所建立的批次檔來呼叫使用, 因爲安裝時已設定某些參數。 (註十四: 因爲畫面管理器是一項可選擇的功能, 除非在安裝過程中選擇畫面管理器的安裝, 否則不會建立這個批次檔。) 這個批次檔的名稱可以改變, 但是一旦改變了檔名, 在圖 13-10當中的第三行, @SHELLB DOSSHELL的 DOSSHELL 必需改爲相對應的檔名。

大部份的使用者會發現使用畫面管理器時, 只需憑直覺即可, 而且不需隨時翻閱手冊就能運用自如。特別是使用滑鼠時, 此時 shell裏面所有功能呼叫都可簡化爲 "point and shoot"的動作, 也就是把滑鼠移到定位, 按下按鈕即可。當需要使用者輸入資料時, 就會出現一個 pop-up (躍出式) 視窗, 以提示使用者輸入資料。

這個 shell (註: 讀者可以把此處所提的 shell, 都看成先前所提時畫面管理程式。) 所提供的服務可分爲 4個不同的部份, 但是以實用的觀點來看, 只有二個部份才有實用價值, 因爲其中有一部份只是返回 DOS (這是你原來急欲脫離的 DOS提示符號。), 還有另一部份只是改變畫面管理器的顏色設定而已。

那麼剩下的兩個部份--檔案系統及 DOS公用程式。檔案系統的功能也許是最具意義的, 然而它缺乏了彈性, 使用者親和力, 或是其他公司發展的檔案管理器之功能, 像是 XTREE PRO 或 Borland的 Sidekick Plus等軟體, 它們的功能更多, 而且可以在 4.x版或以前的 DOS版本下執行。 (譯

者按: 國內的讀者應該更熟悉 PCTOOLS, PCSHELL 或Norton Utility 等軟體, 而不只是 XTREE PRO吧!)

檔案系統 (File System)提示了一個花俏的特性, 就是每一個檔名旁邊都有一個圖像, 讓使用者一眼就可看出檔案是否爲文字檔。這個功能的確很花俏, 但是對一個已經踏入延伸記憶體及擴充記憶體領域的使用者來說, 大概不需要一個特別的圖像來分辨檔案是否爲文字檔。

DOS 公用程式 (DOS Utility)部份的功能則相當有限, 只包含了時間、日期的設定, 磁碟拷貝 (disk copy), 磁碟比較 (disk compare) , 格式化 (format) , 以及硬碟的備份及存回 (backup and restore) 。除了這些功能以外, 其他的仍然要偏勞使用者。考慮各方的表現, 這個 shell的整體似乎相當有限, 至少以它目前所能完成的功能來看是如此。

13.5 The Bottom Line

BONANZA 榮獲 (R)

在此我們把注意力放在 DOS 4.x與擴充記憶體及延伸記憶體的使用關係上。我們已經大致討論過 4.x版佔用了多少寶貴的記憶體空間, 以及它加入了哪些新的功能來解決擴充記憶體的特殊問題。

4.x 版跟其他版本對延伸記憶體的處理都一樣, 什麼事也沒做。根據定義, DOS 是無法認得延伸記憶體的。因些 DOS 無助於應用程式在保護模式下執行。

然而整體看來, 4.x版跟往常一樣, 似乎試著對各種使

用者的要求都能面面俱到。而且按照慣例，它還是無法合乎每個使用者的需求及期望。不過它還是爲眾人帶來了許多有趣而且有用的功能，特別是對兩種極端的使用者：一種是剛入門的新手，另一種則是熟練的使用者。對新手這種看到 DOS 就避之唯恐不及的人， shell (或者說是畫面管理程式) 可提供入門的訓練，而在以往這個部份完全被忽略了。在 4.x 版當中，於螢幕上提供了更有用而且更有幫助的資訊，特別是對處理事務的人。還有，對那些偏重擴充記憶體的 DOS 使用者來說， DOS 4.x也加上了許多新的服務。

的確，除了某些特別的狀況以外， DOS 4.x版會大一點，而且貪婪地要求更多記憶體。然而從均衡的角度來看，除去某些因素以後，新的 4.x版本讓 DOS看起來比以前好得多--不僅現在如此，甚至未來有一段時間也是如此，不過這是由於 DOS 4.x藉擴充記憶體之助所致。(最近也開始利用到延伸記憶體。) 很明顯地， DOS在許多 OS (作業系統) 環伺之下仍然不會沒落，至少不會那麼快。

至於你是否眞的需要或眞的想要新版本的 DOS--或是你是否想要成爲 DOS的忠實使用者，這些都要根據你自己的狀況來決定，下一章我們將討論其他的作業系統。

第十四章
要選擇什麼呢?

目前有許多相容的硬體可選擇，而且有一個全新的行業伴隨它們成長，也許這是一個適當的時機介紹讀者看有哪些作業系統可供使用。

讀者現在大概知道 DOS並非舊式 PC 上的第一個作業系統。不過它目前仍然是整個 8086 家族當中最普遍的作業系統。然而，除了 IBM以外，大概沒有任何人會把 DOS看得如此神聖而不可侵犯。它們是一體兩面的：第一部 PC 由 IBM 製造，而第一個給 PC 使用的作業系統之一就是 DOS。它是首先出現的 PC 作業系統之，但不一定是最好的。還有一個 DOS 的代用品 CP/M-86，雖然它從未消失（不論你是否注意過），但是很快地就被 DOS搶盡光彩，而不再成爲眾所矚目的焦點。（註一：事實上在當時 CP/M 有很多成熟的軟體產品，而且數量比 DOS的還多。因爲 CP/M 的軟體支援很多，所以就有其他公司發展擴充板，把 Z-80 或類似的微處理機放到擴充板上，當成輔助處理器，以允許系統能交互使用任何一種軟體。隨著更多 DOS應用軟體的開發，以及原有的 CP/M版本改編成 DOS版本後，這些擴充板就大量消失了。）

在我們深入其它選擇之前，能對作業系統的功能做更進一步瞭解也許是最佳作法。在輸出及輸入服務，檔案管理之外，作業系統還有更多功能。我們經常把 "相容" 這個名詞掛在嘴邊，而且不是很嚴謹地使用這個名詞。這件事的實情是：許多東西的相容性並不是那麼一回事，特別是說到硬體的相容時，更是如此。造成這些差別的幕後原因就是引起鉅額訴訟的專利權。

也許大多數使用者知道 IBM所銷售的 DOS有很大的差別，至少在某些方面跟 MS-DOS 不太相同-- MS-DOS 較具一般性。而 IBM在他們的機器內裝設了獨有的特性，這正是其他製造商所不敢拷貝的原因，雖然他們曾試圖這麼做，但是代價卻很驚人。不過，有趣的是，從作業系統的觀點來看，較具一般性的 MS-DOS 在 IBM的機器上通常能執行得很好，雖然在其他方面並不一定如此。以 IBM BASIC爲例，只能在裝有 IBM ROM BIOS 的原廠機器上執行，因爲在 IBM ROM BIOS上含有大部份的 BASIC程式碼。

有時 BIOS 就是問題發生的根源。爲了避免不必要的麻煩，有些早期從韓國製電腦所使用的 BIOS ，可以說是在刀口邊緣徘徊，既不能侵犯專利權，也不能失其相容性。有許多硬體上的不相容都發生在較不重要的小地方，然而，問題仍然存在。因爲 IBM的 DOS版本當中存有硬體相依 (device dependency) 的問題，所以 DOS或是其他在 PC 上的作業系統必須設法隱藏這些差異性，讓使用者及應用軟體感覺不出這些差異性。這雖然是件麻煩的事，但是仍要有人去做。

所以，在著手設計一個作業系統之前還有許多事情要考慮－－即使你認爲你已經準備就緒。還記得這話嗎？無限多

的猴子在無限多的鍵盤旁邊。我們已經讓這些東西恢復正常了，不論是修正或是事後補救。（我們所指的這些東西是：無限多的指頭，無限多的鍵盤，無限多的擴充板，無限多的電腦，無限多的應用軟體，以及無限多的版本。）在這麼多的考慮當中，大概可以由此寫出一本鉅著。當然，大多數的人若面對這些問題時大概會受不了。

對於這些問題我們不必再 "你看看我，我看看你" 而互相推辭。 Microsoft已經共同承擔這些問題，每當有新版本出現，它就必須分擔這些頭痛的問題。幾乎每一個版本的 DOS 剛出版時，都會有不相容的情形出現。通常是某人的軟體，或是軟體的組合，硬體的搭配等方面出問題，直到有人出來修正這個問題或是採取補救措施才能解決，像 DOS 4.0 就是這樣。雖然 DOS想要兼顧各種機型、各種軟體、以及各種使用者，但它還是可能擊出 "界外球" ——至少在某些機器上，甚至連 Lotus 1-2-3都無法執行。

然而，在 Microsoft的優勢方面，它們擁有其他競爭者所欠缺的資源，在電腦界裏，擁有更多可用的資源就表示擁有較佳的機會來快速解決某些問題。

使用者千萬不要指望 Microsoft或其他公司會因爲某些默默無聞的軟體無法在特定的 DOS版本執行而顯示出高度的關心，更不用提其他的作業系統了。如果這些毫無名氣的軟體對你來說非常重要，而它卻能在 X牌的機器上以 Y版本執行，這都與他們無關。當你向他們反映時，你所得的答案大概就是：你可以用 X牌電腦搭配 Y版本來執行；再不然就是建議你去找那些軟體的原作者。

在主題完全離開 DOS之前我們至少應該做一次 DOS的回

顧。雖然這句話在以前就說過了，但的確有重述的必要：使用者並不一定要使用 DOS 4.0，以便從所有新的延伸及擴充記憶體發展中，得到完全的利益。幾乎任何版本都能做得很好（通常是指 2.0～3.3 這些版本）。如果你仍然使用 2.1 版，而且也很滿意，那麼就繼續使用，除非這樣會發生問題，否則也不用修正它。

若沒有 4.0，那麼 3.2版也許是一般使用者的最佳選擇。的確，雖然 3.2版缺乏 3.3版的精簡，但是對較舊型的機種來說，3.2 版能提供 DRIVPARM 以支援逐漸普及的三吋半磁碟，這是 3.3版所缺乏的（但 4.0版又恢復這項參數）。

然而，根據本書的文意，不論是延伸記憶體，擴充記憶體，DOS ，或是任何你想要使用的東西--只不過是跳板罷了。所有關於這些記憶體的動作，都是在 DOS以外進行。請讀者牢記在心：延伸記憶體及擴充記憶體這兩種差異極大的技術，雖然都根據很久以前的基礎而建立，但都是爲 DOS而發展的。在發展這些技術的過程中，通常不會考慮到是否能在其他的作業系統下動作。

當然，有些事情是 DOS辦不到的，因此其他的作業系統就顯得很吸引人，特別是某些商用軟體。 OS/2 就是一個例子雖然它已經上市很久了，但是很少人會注意它。它是否能在市場上生存就要看某些問題而定--特別是當延伸及擴充記憶體所能處理的事情愈來愈多時， OS/2 就愈受威脅了。甚至在某些情形，延伸及擴充記憶能做得比 OS/2 好，而且更快。

我們甚至可以說，由於 DOS延伸程式 (DOS Extender) OS/2。從我們開始注意到 OS/2 的時候開始，它的本質根本

就是 80286作業系統，並預留 80386的發展空間。這樣子並沒有什麼不對勁，但是在 DOS的環境下，還有許多 80286的功能未能發揮。

然而，不論是 DOS或是 80286方面，都還面臨太多其他的問題。直到 80386挾其三種操作模式出現以後，這些問題才顯得不太嚴重。 386這三種不同的操作模式爲：實體(real)模式，完全的保護 (protected)，以及模式 8086 家族首次出現的虛擬 (Virtual)模式。

後兩個模式的注意力都集中在延伸記憶體上。然而，在 80386 的架構上有其他部分的差異，這種差異使得桌上型電腦能具有某些接近大型電腦的威力，因此從這個角度來看，的確很具意義。不同於此家族當中的早期晶片， 80386的工作空間並不侷限於 64K的分段內。分段 (segment)可以爲任意大小，直到 4giga位元組 (40億位元組)， 所以分段就不是問題了。它能以 64K的分段來執行 DOS程式，但並不需永遠如此。

而 80286在各種方面都還受 PC 的束縛， 80386則像脫韁野馬般的自由。既然有 32 位元的處理能力，那麼有許多作業系統因而出現也就不足爲奇了。這些作業系統都保留了足夠的 DOS型式，以便與所有在 DOS環境下執行的軟體相容 (即使不是全部相容，至少也是大部份相容) 。但是這些作業系統利用 80386特性的方式及程度都不太一樣。

嘿，你可能會說：由於 DOS延伸程式及擴充記憶體的出現，使得 DOS不再受 1 mega 記憶體的限制，所以 OS/2 不就可以利用同樣的技術，以發揮 32 位元處理的威力了嗎？不錯，這樣的確可以完成許多事情，而且專門爲 32 位元而

寫的 OS/2 無疑地能做得更多。然而，這需要更多時間。

同時， OS/2 的 DOS相容性仍然有許問題尚未解決。當我們觀察到有大量的軟體無法在 OS/2 的相容盒 (compatibilty box) ，內正常執行時，問題就明顯了。相容盒的目的就是讓我們回到 DOS; 這並不一定是 Microsoft的傑作，而是某些東西能夠模擬 DOS那種 "感覺"， 更重要的一點是它能模擬得近乎眞實，使得我們的應用軟體以爲它眞的在 DOS裏面。

請記得，由於擴充記憶體或延伸記憶體是爲 DOS執行環境而建立的，所以我們無法確定這些記憶體是否能在其它作業系統發揮作用。因此，在本書的最後幾個章節裏，我們將把問題的重心放在這些作業系統與記憶體的搭配上，而不只是對這些作業系統做一次瀏覽。

14.1 PC-MOS/386

BONANZA 金牌 (R)

PC-MOS/386 (MOS--Modular Operating System)看起來跟 DOS差不多，而且是以模組 (moudle) 爲單位銷售的的，使用者可一次購足，或是只購買一部份。由於 PC-MOS 386 的外觀很像 DOS，所以乍看之下很難想像其中有什麼與眾不同的功能，以及它的內部跟 Microsoft的產品有什麼不同。

此產品的特點就是它的模組，全套的 PC-MOS/386 是一個完全的多使用者，多工的作業系統。 (此產品最多可支援25個使用者，而且出售時有單一使用者， 5位使用者，以及25位使用者三種包裝，其中單一使用者及 5位使用者的包裝可昇級爲 25 位使用者的版本。) 此系統的設計者也提供了

以軟體爲基礎的埠對埠網路公用模組 (port-to-port networking facility module) 。

雖然 PC-MOS/386 主要以 386系列爲市場，但也可以用在80286 的機器上，特別是那些裝有 AllChargeCard或其它記憶體管理擴充卡的電腦，因爲加裝這些界面卡之後，可以讓受限制的 80286具有重新對映 (remap)的功能。即使在 80286 上沒有加强型的記憶體管理， PC-MOS 仍然可藉某些特殊驅動程式之助而把它的部份核心 (kernel) 重新定位 (relocate)到 1 mega 以上的延伸記憶體。

還有一項值得一提的特點， PC-MOS/386 可以在 8086 或8088的系統上執行，不像 OS/2 或其它同級的作業系統。乍看之下似乎沒什麼重要性，但是因爲 8088 機種可以用在 PC-MOS/386的埠對埠多使用者組態 (port-to-port multiuser configuration)裏，所以這有助於所有使用者在同一個作業系統下執行程式。 (雖然這並不是硬性規定。)

在安裝 PC-MOS/386 時，比起安裝任何新版本的 Microsoft 產品還要簡單--特別是跟 DOS 4.X的安裝過程相比。 PC-MOS/386 的安裝不需複雜的安裝程式，也不需特殊的步驟。基本上，如果一切正常的話，只需把PC-MOS/386的檔案直接拷貝到硬碟即可。即使利用 DOS原來的 AUTOEXEC.BAT 及 CONFIG.SYS也都能正常動作--至少能讓你開機並且執行程式。

如果要仔細調整，以獲得最多 386功能的話，則需要多一點時間，這也是獲得最佳化的 386系統所必須付出的代價。有些 DOS CONFIG.SYS 內的命令必須刪除，例如FASTOPEN 就是一個例子；還有其他 PC-MOS 的特殊命令必須加入。當

然, 386的主要優點之一就是它能重新對映記憶體, 使你能把記憶體放入 640K 以上的未用位址空間。PC-MOS/386支援此重新對映的功能, 讓 PC-MOS 能使用這些空間, 其作法是在 CONFIG.SYS 內加上 FREEMEM命令, 而且最多可指定 5個參數, 以指出 640K 以上不連續的可用記憶區塊。

對一般人來說, 這筆交易並不算太大, 但如果你的個性比較審慎的話, 你可以把你原有的 CONFIG.SYS 及 AUTOEXEC.BAT拷貝到 PC-MOS 的試用磁片 (evaluation disk) 上, 以試用 PC-MOS 的功能, 避免你去做那些要命的心臟移植--把 DOS換成 PC-MOS 的動作。不過這也帶給我們一個有趣的話題。

PC-MOS/386在商場上具有一項特殊的退費保證。每一套PC-MOS的包裝都附有一片試用磁片, 以及一套完整的PC-MOS/386工作檔案, 再加上一片公用程式磁片。這不只含作業系統的核心, 還包括附加的軟體模組, 此模組有點像他們的 LANLink網路套裝軟體。這種作法可以讓使用者從試用磁片執行 PC-MOS , 並且可以在 PC-MOS 上執行使用者的應用軟體, 以獲得較眞實的臨場感。如同上一段所述, 只要把使用者原有的 CONFIG.SYS 及 AUTOEXEC.BAT 檔案拷貝到試用磁片即可。

從試用磁片起動電腦之後就進入 PC-MOS 了, 至少PC-MOS已經對你的系統有了大略的規劃。只要你不打開密封的散布磁片 (distribution disk), 若是不滿意的話, 可以在 30 天內退回整套產品, 以全額退款。不過, 大部份的軟體並不像 PC-MOS 這樣銷售, 是美中不足的事。

很自然地, 試用磁片或多或少都會有保留, 但是PC-MOS

的保留實在微不足道。試用磁片在開機後 30 分鐘左右會自動讓系統當掉。不過你可以重新開機，而且次數不限，但是一旦 30 分鐘的限制到了，你的工作還是會再一次停在那裏。你總不能長期利用試用磁片工作，不過試用磁片的確足以讓你體會 PC-MOS 的功能了。

這種作法非常聰明，在試用磁片上的檔案大部份是貨眞價實的。在試用磁片上的 231K 檔案裏，其中一部份是從密封的封套內拷貝出來的，包括三個作業系統檔案 (註二: $MOS.SYS, $$SHELL.SYS 及 COMMAND.COM) ；以及其他相關檔案，跟其他試用磁片不同的是， PC-MOS 的試用磁片可以建立、修改，並儲存眞正的檔案，無論在任何 DOS的應用軟體下，都可這麼做。

毫無疑問地， PC-MOS 也有不相容的情形發生。當時有一個很著名的不相容性就是 PC-MOS 跟 Intel InBoard 386 加速卡的驅動程式不相容。然而從整體來看， PC-MOS 對 DOS環境的模擬已相當接近。

在 PC-MOS 的特有命令之前都有一個前置的 "."，這是讓人第一眼感覺到 PC-MOS 與 DOS語法不同的地方。這個前置的圓點如果在必要時，可區別某些混淆不清的狀況，例如當 PC-MOS 的命令若不加上前置圓點時，可能會跟某些軟體的起動命令雷同。不過這也不算什麼問題，事實上 PC-MOS 的預設值是 DOT OFF，除非你特別指定才會設定爲 DOT ON 。

有一些 DOS的外部命令對 PC-MOS/386 而言卻是內部命令。事實上 PC-MOS 的建構方式跟 Microsoft DOS有很大的不同，因爲 PC-MOS 的主要內容都放在 $$MOS.SYS內 (在

3.0 版爲 112032 位元組當中，很難看出它所扮演的角色。

LANLink 則跟同一時期的多使用者系統不同，雖然完全以軟體爲基礎，但是並不按照下列的哲學設計--靈巧的伺服器及毫無功能的終端機 (smart server/dumb terminal)。當然，至少要有一部機器充當伺服器，並且載入衛星軟體 (satellite software)。然而， PC-MOS 上的終端機並非毫無功能，而且只要載入適當的軟體及按照情況設定組態就可擁有一部以上的伺服器。伺服器所佔的空間大概是 32K，而每一部份終端機的衛星軟體則佔用 24K記憶體。

雖然 Software Link要求使用者在每一部終端機上執行 PC-MOS，但有趣的是，這並不是絕對必要的。事實上有些終端機可以執行 DOS，只是有些地方要稍微注意一下。 (其實，就算全部的終端機都執行 DOS也沒有關係。) 在終端機之間的傳輸速度非常快速；利用三條電線聯結兩個串列埠時，可達 115000 baud (譯者按: baud 之單位爲 bit/sec)，根據發表的數據，這比平行埠對平行埠 (parallel-to-parallel)的連結方式還快。 (這種平行埠纜線比較特別，跟一般的平行印表機纜線不同) 除了平行埠的速接纜線比較特殊以外，就沒有其他特別的硬體需求了。如果需要較多的外接埠， Software Link也可以提供適合的哽體支援，但是其他的硬體埠 (port hardware)也可代替。

PC-MOS跟 DOS的搭配並不是最完美的，所以不是最佳解法。然而，以多使用者系統的觀察點來看，它也許值得多看幾眼。

14.2 CONCURRENT DOS

Digital Research已經擴展了他們 "DOS 相容的作業系統" 的生產線。事實上， Digital Research 是早期發展 DOS 相容品的公司之一，不過大多數的 PC 使用者對它們的 GEM 產品 (譯者按: 類似 Paint Brush的繪圖軟體) 以及圖形出版軟體等產品會更熟悉。他們把心力投注於改進多人多工的作業系統上，特別是針對 2-5個使用者的辦公室環境。雖然他們的主力產品是針對 286及 386市場，但是最近也增加了一個體積稍小的 8位元產品，以便在任何 8086 家族上執行，從 8088 開始，都可執行這個版本。

因此 Digital現在可以爲 8086 家族的微處理機提供全系列的 DOS相容作業系統。從最低階的產品－－以 8088 爲導向的 DR DOS 開始; 然後是 16 位元的 8086 , 80286 系統上的多人多工系統－－Concurrent DOS XM ; 最高階的產品則爲了針對不同的 386市場而推出了兩種 DOS相容的多人多工系統，一個是 Conurrent DOS 386，另一個是 Concurrent DOS 386/MGE (Multiuser Graphic Edition--多人圖形版本) 。 Digital甚至提供一個 OS/2 的代用品，但是不提供 DOS的相容性。

在他們所有的 DOS型式系統中，都支援擴充記憶體及延伸記憶體。在本書完稿前， XM 版本並不完全支援 LIM 4.0 EMS 規格內的全部 32 mega位元組記憶體以及某些功能。這種限制是由於使用內部驅動程式而非外部驅動程式所致。然而，更嚴重的是，它還存在某些重大的限制，例如處理程式的類別爲何，以及能在何處執行等。

除了他們所有作業系統都跟標準的記憶體擴充板相容以

外，他們的 386作業系統產品線特別支援使用準延伸 /擴充記憶板的 LIM 4.0 EMS記憶體，而且根據規格可支援至 32 mega位元組的上限。他們也支援到 4 giga 的位元組的線性 (linar, extended) 記憶體，這簡直表示這些系統不會妨礙其他的記憶體存取，即使這些作業系統無法存取這些記憶體。(註三: 如同先前所述，即使原始 AT 的晶片 80286可以定址到 16 mega位元組，但是原始 AT 的定址範圍只有 4 mega。這種情況是由於硬體限制，但是這種事不應把硬體限制視爲理所當然的。) 當然，這些數字是非常荒謬的，但是在有些使用的系統中，有人眞的安裝了大約 30 mega位元組記憶體。(請注意，是 30 mega的 RAM !)

然而，這並不能類推到我們用來討論 DOS型式環境下的記憶體使用情形。在 Digital的多人系統下，事實上所有的遠方終端機 (remotes)都只是中央電腦，或者說是主電腦 (host computer) 的一個 process (處理程序) 而已。因此外接愈多遠方終端機，必然會執行更多的應用程式，當然，主電腦的記憶體需求就愈大。

以此基礎來設計的多人作業系統 Concurrent DOS 386 能在主要的終端機上 (main console) 執行最多 4個作業的多工，而在每一部衛星終端機上能支援最多 2個作業的多工 (/MGE 版本則可支援至 4個) 。理論上，在這種系統下，最多有 255個不同的任務 (task) 同時執行。不過眞正的數目要看主電腦的記憶體及特定應用軟體的記憶體需求而定。

有趣的是，即使是一個多人的作業系統，對使用者來說仍然察覺不出它的存在。不過這並不表示 Concurrent DOS 允許所有使用者對所用檔案進行無限制的存取。它所包含的密碼保護部份幾乎跟目前已知的網路系統相比美，而且可以

視需要而起動此功能。

在此系統下，通常不需網路管理者 (network manager)來處理網路的相關事務。事實上使用者可以在任何終端機前 (這包括主電腦的終端機) 執行自己的工作。不過，如果有使用者想處理圖形的話，就必須在主電腦的終端機上處理，因爲系統並不支援遠方終端機圖形 (但是特別的圖形版本， Concurrent DOS 386/MGE 例外。

不要忘了，它只支援文字模式，而且所有的處理都在主電腦上，不論遠方終端機是舊型 PC或 AT，甚至更高級的機種，其工作只不過是把鍵盤輸入傳到主電腦，並且從主電腦接收螢幕輸出顯示出來而已，除此之外，當成終端機使用的電腦並不做任何事情。

這樣子在 Concurrent DOS XM上就有一個嚴重的問題必須考慮，由於它無法在遠方終端機上處理圖形，這表示 XM 版本無法在遠方終端機上執行動作不正常的程式--那些直接寫入螢幕、 BIOS 、或其他硬體內的程式，光是這種缺陷就使得許多極受歡迎的軟體如 Lotus, dBase, WordStar 等等無法執行，實在很可惜!

除了原來 DOS已有的軟體支援以外， XM 還額外提供了什麼呢? 事實上比你預期的還多，連同 Concurrent DOA XM (比 DR DOS 還要早的產品)，還有一整個動作正常的軟體支援庫。大概有 700多種英文版的應用軟體，而且有將近 1400多種應用軟體支援幾乎能滿足任何可能面臨的需求，但也並非多數軟體都能成爲眾人所喜愛的產品。

許多適合 Concurrent DOS 使用的軟體都是 Digital特

別爲藥房，零售商，會計公司，批發商等散布各處的行業所發展的專有垂直行銷軟體 (proprietary vertical market applications) 。

然而，這就是 Concurrent DOS 開始嶄露頭角的基礎,因爲有許多工作的軟體基礎並非 DOS軟體。有很多程式是爲CP/M而寫，還有一些則是爲它的衍生產品: CP/M-86, MP/M-86 及 Concurrent CP/M所發展的。當 DOS成爲市面上作業系統的選擇時，以 CP/M 爲主的 Digital就成爲當時仍爲新秀的 Microsoft之手下敗將了。在原始 PC 的時代裏，IBM 提供了 DOS及 CP/M 兩種選擇，讓那些想要增加一部軟碟的人擇一使用。這些並不是提供給堅持使用卡帶 BASIC及錄音帶做爲大量儲存工具的使用者使用的。

雖然 Digital落敗了，但很明顯地， Digital看出了未來的潮流，以及 CP/M 的市場愈來愈狹小的趨勢因而促成了Concurrent DOS的出現，這樣子有點碰運氣，而且這種配關係也不見得很妥當。

DOS 及 CP/M 在建立及操作檔案的方式有很大的不同。但是從外表來看，這種差別並不明顯，特別是有些內部命令是相同的，如 DIR, DEL, TYPE 等，這更不容易立即察覺其不同。然而，這些外表的相似性都只是表面上的，在表面下的發展則相當不同。在一般狀況下， CP/M 及 DOS都能在同一個微處理機下執行，但是試圖讓這兩個相近的產品結合通常會造成難以接受的結果。 (就像近親通婚通常會產生白痴的下一代一樣。)

然而， Digital做得非常成功--成功得足以讓使用者的注意力從它的缺點轉移到 Concurrent DOS 的其他功能之上

。當然，這些功能當中，最主要的功能就是 Concurrent DOS XM及 386版本所提供的多使用者服務，而且此服務不需使用到 LAN (Local Area Network--小型區域網路) 操作，這樣可以省下一些硬體設備。雖然對 Digital及Concurrent DOS來說並不是很特別，但這些都可經由串列埠(serial port) 做到，不需特別的 LAN界面卡或是複雜的通訊協定(protocal)，只要使用任何 Wyse 60或是 PC/AT相容的硬體，並且可充當終端機使用即可。這包括一部執行Concurrent DOS 386的 386機器當成主電腦使用。

Concurrent DOS的 386設計的確是市場上的明星。它能妥善處理某些動作不正常的程式，以避免 XM 所遇上的問題，而且許多工作的執行表現也較爲順暢。當然，這跟它們的32位元系統設計，不無關係。

除了極少數的命令外， Concurrent DOS 386 的命令結構跟 DOS完全相同，這對許多多人系統的 "學習曲線" 來說，的確有正面的助益。在某些命令上雖然有點出入，而且有些命令也是傳統 DOS所沒有的，但是 Concurrent DOS 當中有一項很好的功能，就是它處理求助呼叫的方法。

在螢幕上任何 Concurrent DOS 命令的特定輔助資料可以用下列的方法呼叫出來: 在命令之後加上 /H 即可。例如想要得知格式化的輔助訊息，只要鍵入 FORMAT/H ，很聰明吧! 如果你連命令的拼法都不清楚，只需鍵入 HELP ，就會有輔助訊息出現，雖然是 DOS型式的作法，但比較具有親和力了。

Concurrent DOS內部有許多有趣的特性。因爲主電腦也必須是伺服器，在 Concurrent DOS 之下的硬碟容量可以高

達 512M 位元組。單一檔案的長度最大可達到 128M 。

讀者也許會問，主機的單一微處理器如何服務 Concurrent DOS下的所有終端機，以滿足使用者的需求呢? 我們已經討論過時間分割 (time slicing) 的隱含意義了，就是以某種循環方式，分配給每一個任務 (task) 瞬間的微處理器使用權。這樣也是處理器的極限了。然而，Concurrent DOS管理時間分割的方式有點不同。這種方式稱爲閒置偵測 (idle detection) ，在這種情形之下的時間片段 (time slice) 不是固定長度，如此不需要此時間片段的工作才不會任意浪費時間片段。

任何一個使用者所能分配到的最長時間片段大約是六十分之一秒－－這樣在 5個使用者的組態裏，每一個使用者在每一秒內大概可接受主電腦的 12 次服務，假設每一個使用者的工作都在執行的話。然而，如果主電腦偵測到任何一部終端機沒有工作執行時，它不僅立刻跳至下一個工作，而且讓此閒置的終端機在接下來的兩次輪迴都不能分配到時間片段。這,不會發生什麼問題，因爲在這麼短暫的時間裏，所有的按鍵都會被接收到緩衝區內。只要偵測到有動作發生，這個進度落後的工作又會放回循環中，螢幕也會隨鍵盤輸入而更新，使用者根本察覺不出有何異樣。

在 RS-232 串列纜線上的傳輸速率並不算很快; 雖然跟平常的數據機比起來， 38400 baud 已經夠快了，但是跟其他軟體以點對點 (point-to-point) 的方式所達成的傳輸速率比起來，還是顯得很慢。幸好除了 /MGE 版本以外，這些以文字導向的系統裏，終端機只會送出鍵盤掃瞄碼 (keyboard scancode) 至主電腦，而接收幾 K的螢幕資料。

Concurrent DOS 386/MGE是一個多人多工的圖形導向產品，主要是爲了桌上出版系統 (desktop publishing) 及其他方面的應用，而且在系統內所有使用者的終端機都可獲得圖形支援。對使用者而言，不同之處只不過是文字模式的支援再加上圖形的支援，但是系統卻需要某些特殊的硬體裝置。一般的串列通訊已經不敷使用，因此 /MGE 系統上的主機必須以光纖 (fiberoptics)與遠程終端機相連。這樣一來，一般的 PC 或相容產品就無法當成工作站使用，而需要以 Sun River 光纖工作站代替。

除去上述的特殊情形以外， Concurrent DOS 通常只需一部 PC 就可充當終端機使用 (Wyse 60 系列也可以當成終端機使用。) 386版本也允許某些特別組態的硬體設備當成終端機，不過這在 XM 版本是不允許的。

不管 Concurrent DOS 是針對多人多工系統的市場而推出的事實，在包裝拆封後，它並不能直接執行多工，而必須設定某些東西。事實上，剛開始一般使用者大概只會操作本文切換 (context switch) 。然而經過一些挫折及學習新系統的折磨後，我們就能讓它辦到我們想要的事－－多人、多工的環境。

整體看來，我們在 Concurrent DOS 的身上又看到了跟 Microsoft 一樣的情形；一個作業系統的發展者試圖在各方面滿足各種使用者的需求，但是跟 Microsoft一樣，在某些方面非常成功，而其他則乏善可陳。因此，如果使用者所著重的是網路的功能，那麼 Concurrent DOS 所提供了多個可行的解決方案。也爲那些死硬派的 PC/M 使用者提供了一個漸進的方向進入 DOS，而不必抱怨爲什麼他們需要在短時間內適應全新的軟體。

14.3 DOS及 UNIX : 一個不簡單的聯盟

BONANZA 金牌 (R)

曾經有一段時間，使用者不願把 DOS及 UNIX 相提並論。但是現在改變了，也許使用者在陌生人面前仍然不想把這兩者相提並論，但是在較開明的朋友之間，有時就沒什麼關係了。

事實在於很少有軟體能讓兩者完全共存，同時也要讓同一部 386的機器上的 UNIX (XENIX) 檔案及 DOS檔案能夠通用，就像一個快樂的大家庭一樣。在此我們再一次提出目前 80386 環境專用的系統上，所發生的一些因硬體而異 (device specific) 的問題。然而，不論你喜不喜歡，最好能習慣這這種想法，因爲是否要把 DOS及 UNIX 混合在一起，或者是繼續停留在 DOS階段，使用者必須要多瞭解這些相依關係。

在 DOS及 UNIX-- 特別是合乎我們目的的 XENIX--當中，我們選擇 XENIX--一個 Microsoft授權的 UNIX 多人多工作業系統之 SCO (Santa Cruz Operation) 版本，原因如下：

(1) 根據統計，它在 PC 的 UNIX 的市場上有 90%的佔有率。

(2) XENIX 提供了他們特別的多人多工之 DOS-under5 XENIX shell-SCO VP/ix ，以搭起 DOS 及 XENIX之間

的橋樑。

這是兩個相當明顯的理由，特別是因爲本書的主題在於討論 DOS及其延伸部份的使用。而矛盾之處就在於：使用 UNIX作業環境也許超出本書的範圍。不過，市面上有許多特別爲 UNIX 所寫的書籍，有興趣的讀者可以看看。（譯者按：瑩圃最近也出版了有關 UNIX 的自學手冊，這個資訊對讀者應該有幫助。）

我們先把主題限定於 DOS-to-Xenix-to-DOS的可攜性，然後再回到正題；所著重的是附屬於 XENIX之下的公用程式 SCO VP/ix ，而非 XENIX系統本身。特別是因爲我們最常聽到及爭論最多的 UNIX 話題之一竟然是它缺乏相容性，以及昇級以後改寫整個軟體支援庫所需的成本會轉嫁給使用者。

也許對 SCO VP/ix的最佳工作評價爲：在系統內並不易察覺它的存在。事實上，它隱藏的功夫甚至讓使用者不需在硬碟上劃出另一個區域即可容納它。檔案可混在一起， SCO VP/ix 管理了 DOS及 XENIX之間的所有檔案轉轉向 (rederiction) 及管理 (piping) ，必要的話還會處理一些程序。在 DOS或 XENIX下都可起動一個新的工作 (session)，這沒什麼關係。

本書的主要目的之一即是討論擴充及延伸記憶體的使用，所以我們也要討論一下 XENIX如何使用及處理那些擴充及延伸記憶體，以及如何處理那些動用這兩種記憶體的 DOS應用程式。

SCO VP/ix 的確可 支援 LIM 4.0 EMS 規則，但是這並非沒有限制，有些地方還是要特別注意。最重要的就是它並

不支援 LIM硬體，因此這些擴充板就不受支援、也不再需要了。 SCO VP/ix是利用標準的 XENIX記憶體來模擬 LIN指定的硬體動作。

當使用 SCO VP/ix模擬的 LIM擴充記憶體時，最大的可用記憶體是由 SCO VP/ix所設定，而非 LIM規格的設定。在目前的版本當中的上限爲 2 mega ，而不是 LIM 4.0所制定的 32 mega位元組。這對廣大的擴充記憶體用戶來說是件壞消息，因爲這樣的記憶體空間只有舊版 LIM 3.2規格所容許的四分之一。據 Santa Cruz 公司宣稱，在往後的版本當中將提供較大的模擬擴充記憶體，但使用者最好能檢查一下你所使用的版本，以免因爲這種限制而造成極不愉快的經驗。

事實上，所謂的 XENIX標準記憶體只是很基本的東西，就是我們在 DOS環境下所稱的延伸記憶體。此線性記憶體位址從 0h 開始遞增。要執行 XENIX系統，建議至少要有 2 mega記憶體；而建議最少要有 4 mega 記憶體才能執行得順暢，而且每增加一位 SCO VP/ix系統的使用者，最好能再增加 1 mega 記憶體。

除了一些標準及非常基本的硬體配備--如鍵盤、標準顯示介面卡、軟式磁碟機、以及串列埠、平行埠等裝置以外，DOS 及 XENIX的整體相容性僅止於檔案及資料的交換，而且是在混合且匹配 (mix-and-match)的環境下。 SCO VP/ix 並不支援以 DOS爲導向的特殊顯示卡，網路及通訊界面卡，但是特別替 SCO XENIX 386作業系統所設計的硬體裝置則可支援。人總要入境隨俗，如果你不願意的話，雖然對XENIX無損，但是在這個簡單的電腦世界裏，你可能會處處碰壁。

因此，在 XENIX下無法使用 DOS機器所使用的通訊界面

卡。那麼 DOS的通訊程式呢? 通常是沒有問題的。 SCO VP/ix 可利用軟體方式來使用平常的串列埠，反而無法處理那些它看起來不相容的裝置。

說了這麼多的用意就是提醒使用者: 如果計劃使用 UNIX的話，你可以現在就開始計劃，而且大部份的軟體及硬體都還能派上用場。有了這種認知以後，只要在增加新配備或新軟體時確認它們的相容性，即可放心地投資。

由於篇幅有限，我們只能對此做蜻蜓點水式地介紹，目前還有其他作業系統未能介紹，而且確信還有其他作業系統出現。然而，因爲我們把焦點放在 DOS及其類似產品上，所以甚至連 OS/2 都很少提到，不過它至少跟 DOS軟體之間存在某種程度的相容性。事實上，在 OS/2 的相容盒內，作用就相等於 XENIX或是 Concurrent DOS 的 XM 版本，不過，那要看看你所使用的應用體軟體而定。

無論如何，即使本書已指出許多實際使用上的問題，但是這些問題在基本都未解決。而許多程式爲了解決這些問題而大量出現，但是，這些程式並非作業系統，這是很重要的。它們只不過是一種傳遞媒介，一種介面。眞正重要的是你的應用軟體; 那些應用軟體所完成的工作才讓這些介面程式產生價值。

明顯地，還有其他選擇。 DOS只不過是其中之一，我們所介紹的一些作業系統也只是其中的少數。當然，在某些情況下，有許多無可奈何的原因使用得我們必須選擇其他系統; 如果有些特殊需求但 DOS無法辦到時，這些系統就能找到它們的容身之處 (niche)，這種特殊需求可能就是多人使用的環境; 或者說是爲了遷就某些特殊的軟體，那些商業應用

或科學上的程式支援也是不可忽視的。

然而，不論你是否喜歡， DOS仍舊是 DOS，而且是一種標準。 DOS仍然是一種競技場，軟體發展者在此比較軟體的表現，廠商也以 DOS爲環境來測試硬體，只要軟體能在 DOS下執行就行了，至於是否能在另外一套規則下執行，則要看看廠商支援的程度而定，不過這種支援有時是有限的。這並不是說廠商不想試著解決這種不相容性，而是 DOS才是他們主要的市場目標。

第十五章 崩潰(CRASH)過程

請注意下面這一個電腦專用術語 (Workaround) 。你也許不熟悉這個字的意義，但它的確已是一個長久存在的問題了。事實上，它比擴充記憶體的種種問題還重要。而且對具有記憶體映射能力的 80386而言，它更是一個急待解決的問題。

長久以來，IBM 及 Miscrosoft 皆不允許使用 640K 以上的記憶體空間。而 LIM (及 EEMS)則首次突破這項限制，提供 640K 以上的記憶體。有些人認爲這是 "多此一舉" --因爲沒有人會使用到這些多出來的記憶體! 但是，已有許多頗具規模的電腦公司正在逐步實現這項計畫，由此可見，增大記憶體空間絕不是沒有意義的。最大的原因是我們必須配合網路介面以及其他新發展出來的產品。

例如，以過去和現在的標準來看皆合法的一起始位址爲 E000h 的一個 64K的 "頁框" (page frame)。IBM 早已宣布從 AT 之後該空間便供 ROM使用。但實際上，大部分的機種--例如 PS/2 機種 30/286 --所提供給 ROM的空間卻被限制

在 F000h到 FFFFh之間。從圖 15-1 中的 ROM ID 便可看出。

```
ROM date at address F000: FFF5 is 08/25/88.
(C) notice at address F000: E005 is Copr. IBM 1981.
ID Byte at address F000: FFFE IS FC
```

圖 15-1 IBM PS/2 機種 30/286 的系統 ROM的資料

儘管 Intel的 AboveBoard 沒有從 E000h開始的那個"頁"，但仍有許多的擴充記憶板允許 EMS何使用 E000h作爲一頁的上端起始位址。在 EMS中，一頁的基底位址可以低至 C000h (當沒有 EGA ROM的時候)，或者當你的電腦沒有任何 ROM佔據這些位址時，基底位址爲 E000h也是有效的。

以下有二種方去可供你檢查你的電腦內部狀況:

1)首先，你可以在你的電腦裏安裝些新的軟體，使之崩潰 (crash)後將之移去。然後將工作區映射至 640K 以上，找出發生位址衝突之處，並檢視是否有什麼可移至另一位置的。如此繼續，或者，

2)直接將存在於 640K 以上的記憶體的使用狀況映射出來，即可在開始之前找出可能發生的位址衝突。

而這二者中，我們認爲第二種方法較適用。但是它仍然無法發現發生在 640K 以下的記憶體位址衝突之問題。而在那 (640K以下) 的底部正是許多不同的設備取用你的系統的

入口。故爲了能了解 640K 以下的記憶體的情況，你可以採用以下二種方法:

1)同樣地，在電腦中安裝新的軟體，使之崩潰後移去。這次，則是將使用的記憶體映射至 640K 以下，找出發生位址衝突之處，並檢視有什麼可被移到另一位址的，如此繼續。或者，

2)直接將 640K 以下的記憶體使用狀況映射出來，即可在開始之前找出可能發生的位址衝突。

換言之，由以上提供的方法可看出檢視電腦記憶體內容的工作是非做不可的！否則當你在使用記憶體時可是後患無窮！

以上的工作皆可以利用 4.X DOS來完成。如果沒有 4.X DOS ，還有其他可提供大致相同功能的軟體。若能善用，可以得到很大的方便。尤其是用於 80386系列 (或者是裝備有類似 ALL CHARGECARD 的記憶體管理系統的 286系列)，這類允許將記憶體映射至 1024K以下任何位址的機種。

在第六章已說明了儘可能地將記憶體映射至 640K 以上的重要性。藉著如此映射，可以把 640K 以下的記憶體中被重覆使用 (overhead) 的部分重新配置，以期能釋放出更多自由記憶體，供自己的應用程式利用。

15.1　高位址記憶體

BONANZA 螢圖 (R)

您應該儘可能地使用高位址記憶體，但是，在開始著手

之前還有更重要的一件事是：弄清楚記憶體內有些什麼東西。因爲，一旦你放進了新的東西，舊的內容馬上被蓋掉。然後，再利用重新映射便可以將資料置入適當的高位址記憶體中。

假如你從來使用過擴充記憶體，那麼，現在正是時候。

帶塊擴充記憶體回家，打開包裝，掀開你的電腦主機上蓋，然後站在它前面。EMS (Expanded Memory Specification)允許一頁的起始位置爲 C000h，或者在 C000h 以上距離爲 16k的倍數之處，只要 "頁" 的上端不超過 F000h即可，總共有 192k可供工作使用。"頁" 的起始位地最好定在 C000h，因爲有一些機種用 E000h當作 ROM的基底位址，爲了安全起見，作此決定。由圖 15-2 可見 C000h 緊接在 BFFFh上方，而 A000h到 BFFFh間的空間是預留給螢幕用，但 EGA只用了較低的一手，所以 C000h看起來很合適，但卻有可能會崩潰！

```
        BASE ADDRESS
        ------------

             FC00
             F800
             F400    SYSTEM ROM
             F000
        --> EC00
        --> E800
        --> E400
E000 ----> E000
        --> DC00
        --> D800
        --> D400
D000 ----> D000
        --> CC00
        --> C800
        --> C400
C000 ----> C000
        --> BC00
        --> B800
        --> B400
B000 ----> B000
        --> AC00
        --> A800
        --> A400    VIDEO
A000 ----> A000
```

圖 15-2 從 A000h到 FFFFh的 ROM及顯示區

你該作的是馬上去檢視那塊記憶體，看看究竟有些什麼"存貨"。如果你的機種是 386, Qualitas' 386MAX.sys 能夠協助你輕易地查出 640k 以上的記憶體內的情形。如圖 15-3所示。當使用 386MAX 時，每次系統開啓 (boot) 螢幕上便出現類似圖 15-3 的畫面 (圖 15-3 是一個已載入的系

統）。

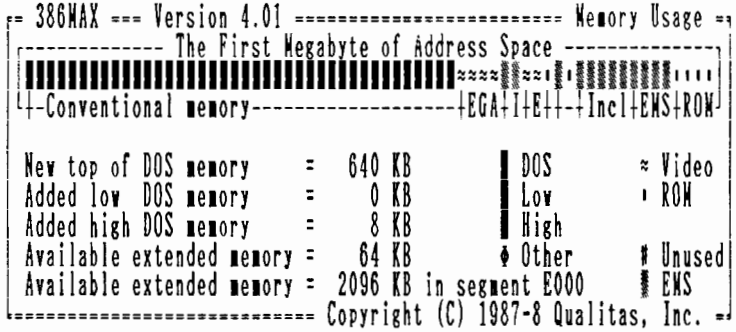


圖 15-3 在使用 386MAX 作爲擴充記憶體處理程式時，以上圖表會在螢幕閃過，時間卻很短暫，幾乎來不及細看，但在圖 15-4 中可見 386MAX 能夠找出所有在 ROM中可能發生的衝突。

利用 386MAX 所提供的功能選擇可以得到類似圖 15-4 的結果，更清楚地印出關於 ROM的情況。以圖 15-4 爲例，可以看出在 C000h開始的那個 "頁" 中有二組 ROM，原先還以爲是安全的呢！接下來，又有了新的疑問。這二組 ROM究竟是何來歷？DOS 的 DEBUG能解答這個問題。

ROM In High Memory				
Starting Address	Range Start	End	Length	ROM Option
000C0000	768	784	16	C000-C400
000C8000	800	808	8	C800-CA00
000F0000	960	1024	64	F000-10000

圖 15-4 利用 386MAX 來搜尋 640K 以上的記憶體，結果發現有三塊區域有 ROM。雖然它並沒有標示出使用者爲何，但並不難決定。這三塊區域正是當在映射記憶體時應該避免使用到的。

任何版本的 DEBUG皆可使用。在載入 DEBUG後輸入基底位址，此外另須 ":"及 0緊接於後，當作相對補償 (offset)。然後螢幕便會出現類似圖 15-5 的畫面。

ROM In High Memory				
Starting Address	Range Start	End	Length	ROM Option
000C0000	768	784	16	C000-C400
000C8000	800	808	8	C800-CA00
000F0000	960	1024	64	F000-10000

圖 15-5 在找到 ROM基底位址後便可利用 DOS的 DEBUG dump(d) 功能一探使用這些 ROM的裝置究竟爲何。上圖可見的二塊 ROM 分別是供 EGA card及硬碟控制器所用。 (註: 在引號 ":"之右至少要有一個字元。)

檢視二組 ROM的位址得圖 15-5 , 可見其中一組爲硬碟控制器 (XEBEC), 而另一組則屬於 EGA Video card (PARADISE SYSTEM) 。EGA 卡有 ROM, 硬碟有 ROM, 當然其他很多裝備也都有 。這二項裝備的 ROM 佔用了在 C000h的 "頁" 的中間部分, 其他諸如連接網路用卡 (networking card) 等裝備也有時會佔用 640k 以上的記憶體空間。剩餘的記憶體, 在 CA00h以上就沒有什麼東西了 (從 F000h到 10000h所放的是系統本身) 。

爲了方便起見，以下的討論以 ROM爲主。但是一個裝備較多的 386系統在高位址記憶區是非常擁擠的，如圖 15-6 的情況。圖中除了剛剛檢視出來的二組 ROM以外，還有高位址 RAM映射到這塊記憶體內。這組 RAM共佔 136K,且分佔在三塊大小分別爲 32K, 16K 及 88K的記憶體。此外尙有一個 "頁框"，起始位置在 E000h，可供進入擴充記憶之用。

```
Current Mode                = ON
Expanded Memory Available = 1552K
Page Frame Address          = E000H

Expanded memory is being used
```

Area	Size	Status
0000 - 9FFF	640K	Conventional
A000 - AFFF	64K	Video
B000 - B7FF	32k	Hign RAM
B800 - BFFF	32K	Video
C000 - C3FF	16K	ROM
C400 - C7FF	16K	High RAM
C800 - C9FF	8K	ROM
CA00 - DFFF	88K	High RAM
E000 - EFFF	64K	Page Frame
F000 - FFFF	64K	ROM

圖 15-6 在這次的 386系統的高位址 RAM中，分別爲 ROM，顯示區及 EMS頁框所佔用。而且正好有 136K 的RAM 可以置入，這使得一切努力都變得很值得。

如上述把記憶體內各種的映射結果完整地記錄下來將有

很大的用處。例如，在你希望搬動記憶體內容或設置一個新的硬體設備等這種會改變或破壞系統本身的工作前，若能確定情形，便能安全地工作。

較佳的 386記憶體管理軟體 QEMM 及 386MAX 等都相當方便使用。每當你要設置一個新的系統時，它們會自動地檢視 640K 以上的記憶體空間的內容。然後，再依預先設定好的方式將你的系統映射至找到的可用記憶體上--但不必要佔用所有的可用空間。以 386MAX 爲例，它會跳過一個大小爲128K的影像區 (Video area) 不用，此處正好可以供你使用。做法很簡單，手冊會說明做法。但是，會因爲監視器 (monitor) 不同而有異。在 CONFIG.SYS 檔中 (如圖 15-7 所示) 必須加上 INCLUDE敍述，使得未被 EGA佔用的 32K記憶體 (B000h～B7FFh) 可資使用。

```
DEVICE=C:\386MAX.SYS INCLUDE=B000-B800, 64
device=c:\386disk.sys  360
device=c:\dos32\ansi.sys
lastdrive=z
drivparm=/d:0 /f:2
files=32
buffers=20
```

圖 15-7 CONFIG.SYS檔中有 386系統的擴充記憶體驅動程式，顯示了 INCLUDE敍述的用途，它能使得映射記憶體至某特定區塊，或避開某特定區塊，爲的是驅動器的某些特質。此外，有些驅動器則允許使用者特別設定某些記憶區塊而自行調整以適應之。

多出額外的 32K記憶可用，386MAX當然不會放過。如果

不需要再設置別的系統，你便可告知 386MAX 這 32K的可用記憶體，它便會使用。

有時，在設置過程 (boot cycle) 的早期，某些位址是否爲空白可用或另有他用，對 EMM來說並不清楚。這種情況下必須使用 EXCLUDE敍述來特別標記某些界限不明的區域。

以上的討論，對操作調整 386系統有很大的幫助，但在完成後，須特別注意不可在頂端加入新的內容以免破壞其他的內容。

雖然看起來工作不少，但若照圖 15-8 所示，有系統地完成，也並不如想像中的困難。從 640K 以上的空白位址開始，先把本章已述及的 ROM、 "頁框" 等必須佔用的記憶體保留起來。然而，空白的記憶體不必要標記，EMM 會自行找到並映射於斯。但是，在 CONFIG.SYS 中已經標記了的 INCLUDE 區域 (B000h～B800h) 卻像永久裝置，只要 CONFIG.SYS有 INCLUDE，它便存在。

圖 15-8 很簡單，它標示著 16K一頁的間隔，這是一般常用的大小，儘管有的特殊情況時可能降到 4K 。然而 16K 已經夠精細了。這些資料可以被存在電腦中或列印出來。但當你想要處理你的系統時，手邊有著即時 (current)的資料將會省下一大段時間。但是，問題不只發生在 640k 以上的記憶體中，640k以下也有可能！

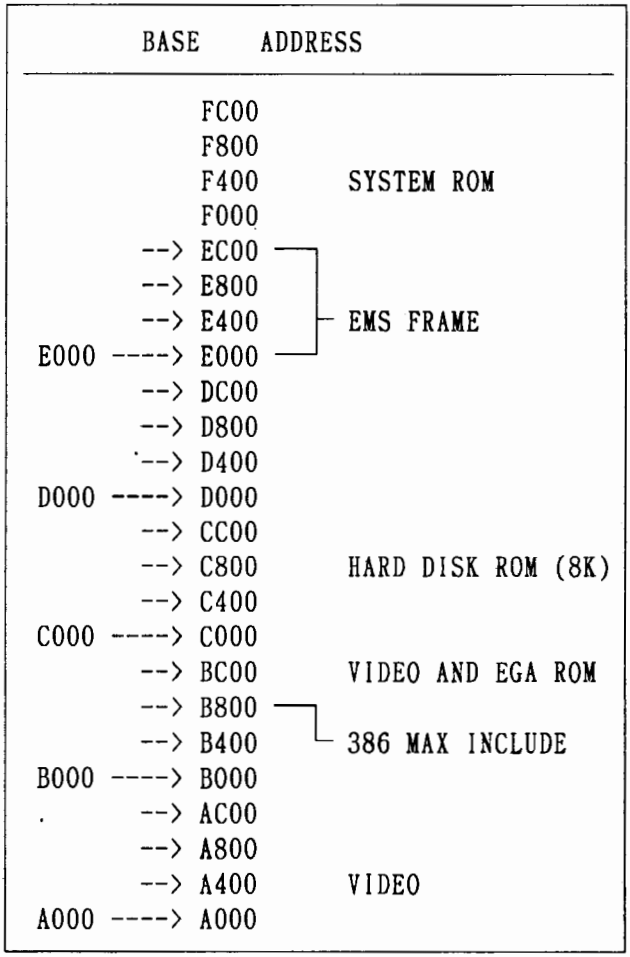


圖 15-8 簡單地顯示出 640K 以上的情況。類似的功能在設置新硬體 (或軟體)時對防止崩潰及增加效率有莫大的助益。

15.2　當埠與埠發生衝突時....

" 埠 "(PORT)是另一個不容忽視的重要課題。

我們習慣於使用並列 (parallel) 或序列 (serial) 的埠，而且皆已設定好固定的位址。然而，我們常須面對另一種情況，即是必須自己爲新的設備來安排位址。不像先前所討論的高位址 RAM須佔用 16K大小的頁，埠只需要佔用比較少量的記憶體--通常只有數個位元組。但是，它們卻不完全如此整齊--不同的埠可能會用到不同大小的記憶體！但這不正是你所需要的嗎！

實際上，藉著系統化方法可以順利地解決這個問題--即是先前所討論的在高位址記憶體時所採用的方法，而不必去一部分一部分地嘗試、錯誤。

圖 15-9 中列出了供埠使用的記憶體空間及其內容。這些位址皆是用來供與實際或虛擬設備互相溝通用。或許，曾經設定的裝備皆需要一個埠。

```
02A0h unused
02B0h unused
02C0h unused
02D0h unused
02E0h reserved
0200h games or clock
0210h PDI expansion board
0220h reserved
0230h reserved
0240h reserved
0250h unused
0260h unused
0270h unused
0278h LPT3
0280h unused
0290h unused
02F0h reserved
02F8h COM2
02H8h COM4
0300h 0303h  JT FAX
0304h 0307h  JT FAX
0308h 030Bh  JT FAX
0310h 0313h  JT FAX
0310h TAKE TEN
0314h 0317h  JT FAX
0318h 031Bh  JT FAX
0320h TAKE TEN
0330h PAN FAX
0330h TAKE TEN
0340h PAN FAX
0378h LPT2
03E8h COM3
03F8h COM1
030Ch 030Fh   JT FAX
031Ch 031Fh   JT FAX
0440h TAKE TEN
```

圖 15-9 裝置位址表

DOS 設定一些埠可自 ROM或 DIP的開關裝置讀取資料。有些埠是由裝置的驅動器自動設定--某些功能較强的驅動器甚至能夠自行尋找足夠的空間，然後把自己安裝完畢。

埠有時會發生衝突的現象，但它並不會使整個系統崩潰，因爲埠的設定通常矛生在電腦將控制權交出給使用者之前。發生衝撞時，會有其中之一或更多的設備無去與電腦完成連線。此時，也許會有些錯誤訊息出現在螢幕上。無論什麼原因，也許看起來很嚴重，只需要把它看成類似於某塊記憶體的衝突即可。

在圖 15-9 中把一列可供埠使用的記憶體分配給四種裝置--包括 Data Control Corp "Take Ten" removable media drive,一些典型的 FAX BOARD (包括 Panasonic及 JT FAX),以及一片附有時鐘及日曆的多功能擴充板--由 Pure Data Limited 所製造的 Canadian Board。

在這四種設備及 0300h到 0400h的記憶體之間已經有二處可能發生衝突 (以粗體字標示) 以及一些擁擠的情況。注意 JT FAX board 的間隔爲 0004h，然而有的裝置取間隔爲 0010h ，所以，絕不能只注意到起始位置沒有衝突發生就以爲可以高枕無憂了。

有二種方法來追蹤記憶體內容。可以利用圖解或者，若有功能不錯的文字 (WORD) 處理器，有排序的能力，可以保存一組態串列 (configuration list) 在電腦內，如圖15-9所示者。

15.3 軟體崩潰可能很麻煩

BONANZA 曼圖 (R)

儘管幾乎不可能只使用擴充記憶體，然而發生系統崩潰的機會也會因爲有愈來愈多的東西需要同時被放在額外記憶體中而增加。此外，當須要去把問題挑出來時，更是一件勞煩的工作，尤其是沒有使用系統化的處理方法，則問題更形複雜。而一般立即的結論通常是假設問題是出在正在執行的應用軟體中。

然而，問題有可能是一個 TSR等得不耐煩了，也許它對應用軟體所產生的結果不盡滿意，或者，是 DOS的版本出了問題。而你將會發現，問題並不那麼容易；但其實也並不如想像中的困難--如果能妥善處理的話。

實際上，並非每次系統發生小問題便須要勞師動眾地去找出癥結所在，在這數以百萬計的位元組中難免會有些意外發生。只須在問題頻頻或有某種固定模式的問題發生時，才開始去挑出毛病。

通常系統本身便可以有很大線索，儘管它是有問題的。例如，當下列的錯誤訊息出現：

```
Execption Error #13 at 1607: 0000
Error code 0000
Do you want to Terminate the program ,R)eboot
, or to C)ontinue?
```

當看到上面的訊息時，別急著重設系統 (reboot);至少

注意一下其中的數字--尤其是位址，是最重要的。此外上例中還有 Exceeption 13及 Error Code 0000等字樣。除非已知道了發生問題的軟體爲何，否則對我們來說是沒有用的。把這些訊息記錄下來，它們在維修系統時會有很大的幫助。

在本例中，我們可以從位址 1607(h)下手--一個當在搜尋問題所在時，重複出現的位址。從 1607(h)，可以知道問題出在記憶體較低的部分。現在的問題是--在那發生了什麼事？

至此，問題已被相當地簡化了，而且只要系統在癱瘓之前能夠留下類似的線索，你就可以故技重施。現在需要一個軟體，必須能夠讀出任何被載入的軟體的位置及形式--當然，它本身不能先當掉。這也就是說，這個軟體必須能允許使用者在使用時仍能與 DOS溝通工作，如此一來才可以把包括它本身在內的所有軟體的情況盡數囊括。並非所有的軟體皆有此能力，不過，大部分的軟體都具備。

有些比較複雜的記憶體管理軟體包括有映射功能 (mapping utilities) 。也有一些映射功能是獨立出來的，可以在 Microsoft或 Desqview 下執行，或與其他的視窗環境及軟體配合使用。圖 15-10便是一例。它便是從某種這類獨立的程式所得到的結果--事實上，個軟體叫做 PCMAP.COM 。這個程式可以在 PC maganize中找到。它只相容于 DOS 3.3 。此外尙有許多與 DOS 4.0相容的軟體也有這個功能。

在圖 15-10中雖然沒有任何程式 "正好" 落在 1607(h)止，但是 1607(h)卻是在 CULPRIT.COM所佔用的記憶體塊之中。利用一點簡單的數學 (如果你不善於 16 進位的運算，

可以借用 Sidekick 的 16 進位計算機的幫助) 可以發現:

1607(h)-15F7(h) =10(h)

```
PCMAP 1.0 (c) 1987, Ziff-Davis Publishing Corp.
Segment Paragraphs      Bytes       Program
1517H       00D1H        3344       COMMAND.COM
1E58H       0375H       14160       DV.COM
1E34H       001FH         496       UNCRASH.COM
1E0FH       0023H         560       MODE.COM
19C7H       0446H       17504       VKETTE.COM
15F7H       03CEH       15584       CULPRIT.COM
1E78H       0003H          48       COMMAND.COM
21D3H       00DAH        3488       COMMAND.COM
22A2H       0004H          64       (FREE)
22B4H       0A2BH       41648       SQLOAD.EXE
2CE1H       0065H        1616       CHSTACK.COM
2D48H       335EH      210400       EDITOR.EXE
60A7H       00D8H        3456       COMMAND.COM
6183H       3E87H      256112       PCMAP.COM (FREE SPACE)
```

圖 15-10 有 Culprit的位址映射表

這使得問題落在 CULPRIT--一個用作 disk caching 的程式--所佔用的 15584個 bytes之中。利用 AUTOEXEC.BAT 將之載入, 如果每次問題都出現在相同的位置 1607(h), 那就是了!

還可以更進一步來証明, 即把 CULPRIT排除在載入程序之外, 然後看看是否眞的不再發生問題了。但是把 culprit 移走會產生新的問題。例如在本例中若有別的程式會時叫 Culprit 的 creator, 若此, 則利用類似回信的方式來答--別的程式已經有了相同的問題。

除了在記憶體低位址的部分會發生問題之外，高位址的記憶體當然也可能發生問題。特別是當想要從那 "擠" 出一塊可用的記憶體時尤其可能。然而，就像低位址記憶體有可使用的的映射功能軟體一樣，高位址記憶體也有適用于A000(h) 到 FFFF(h)之間的軟體映射功能。

有些裝置，尤其是繪圖調整介面 (graphics adapter，例如: Sigma Color 400 Graphics Adapter 及某些Vermont Microsystem 等。) 會使用到合法的高位址 RAM以作爲所謂的 "Scratch Pad"空間。這塊記憶體將沒有標記而且不能再被其他軟體叫用。但是，如果不小心去叫用了些看起來像是未被叫用過的記憶體，那麼崩潰將要發生！本例中如果一個繪圖板佔用了 scratch pad的空間，你將會發現螢幕不會顯示。

去找尋問題之所在有時會關係到在擴充記憶體內的視窗(Window)下所執行的程式。這些輸出的位址都是 DOS的位址。而隨著視窗的切換，程式群的實際位址每次都不一樣--當然，在同一個工作期間，同一個程式的位址是不會變的。

儘管此處所列的例子在任何的 Desqview 視窗的基底位址之下皆有些其他的程式，但通常就是如此，而只有在發生崩潰那一刻所映射得到關於視窗內所開啓的軟體的結果能提供些線索。如果你必須改變視窗才能與 DOS溝通，那這切換的動作將會把應用軟體藏在擴充記憶體中的某個位置，使得無法映射成功。

最後的一點忠告: 絕不要同時改變二種以上的結構 (軟體或硬體) --即使有時候好像不錯--能夠掌握住一個或二個條件變數將會使事情進行得順利多了。

第十六章 視窗及水晶球

"視窗系統" (Windows) 使得多工 (Multitasking) 成爲可能。Microsoft 便藉著利於使用的，功能表式的 (Menu-driven) 繪圖介面成功地將之呈現在世人的眼前。但不可否認地，多工的範圍卻受著記憶體大小的限制--只能對容許被放在 640k 記憶體內的所有軟體進行多工處理，其中還必須包括 DOS、視窗系統等必定會佔用空間的超支使用。但是在 Desqview 及 EEMS 出現以前的情況的確就是如此。有人利用滑鼠及下拉式功能表，而也吸引了一些追隨者。

視窗系統的發展由來已久。繪圖介面仍然是視窗系統有名的特色。不只是與其他軟體之間可作爲介面，有許多的應用軟體可以在視窗系統下執行，而且有些特殊軟體是專門爲它而寫的，例如 Aldus Pagemaker及 Microsoft自己完成的 Excel spreadsheet 等。

一個視窗系統的作業環境對許多應用軟體來說都是必備的，有些則是選擇性的。以下便有一簡短的對視窗的說明。

視窗系統最明顯的一項特色就是它以圖形爲主 (graphics oriented)，這和與 Desqview 的以文字爲主 (text oriented) 大相逕庭。但這並不表示不能在視窗系統中使用文字或者不能在 Desqview 中繪圖，而是用以定義各程式的執行方式，因爲這正是視窗系統及 Digital的 GEM--此二者在某方面很相像--的優點及缺點所在。

在文字模式下，把資料表示在螢幕上的方法相當簡單而直接。只需敲下鍵盤上的某個鍵--例如 "a"，然後會有一個很簡單的碼傳送給影像系統。八個位元，即一個位元組即足夠以用來表示 256個不同的單獨字元。而爲了表現顏色還需要便多的資訊，但一個位元組能夠最多代表 256個字元。在一個 25 行，每行 80 個字元的螢幕上最多有 2000 個字元，甚至不必移動多少資料。

在另一端，這個代表 "a" 字元的碼會驅動一個字元陣列，在其中，控制了螢幕上一個字元所佔用的一個長方形空間。在那黑盒子裏包括了一些點或圖元 (pixel)及字元產生器 (character generator)，它曉得在諸多圖元中，那些該亮，那些該暗--就像會閃動的廣告招牌上的燈泡--來顯示出一個能被辨視出是 "a"的圖形。只需一些資料便能達到目的。當然，只有一種字形，但是它的確完成了工作而且相當快捷及容易，這就是文字模式。

視窗系統 (Windows)則在繪圖模式下工作，它並不用字元產生器，而是在螢幕上繪圖，一次一個點。在一個解係度爲 600*350的螢幕上則必須分別控制 210,000個點。首先，視窗系統必須指明對每個點的處理方式，然後必須傳送所有的資料--相當耗時的一種方式。

也許你要問：顯然有比較容易的方法，爲什麼要自討苦吃？答案是：不能在文字模式下繪圖呀！讓我們來看看桌上列印系統。爲了表現某一頁的內容，也許需要畫出不同大小及字型的字元，也許有相片或者其他需要一個位元一個位元映射的圖形，也許有些希望要放大或縮小的圖案；以上種種皆必須在螢幕上一點一點地去描出來。

於是，有許多應用軟體非要有繪圖模式的工作環境不可！使用者也許沒有察覺這點，因爲許多產品都有其自己的工作環境。當在使用一種桌上列印系統來查看磁碟中的目錄時，有可能可以找到某個數字旁標記有 WIN或 GEM等字樣。則可以在文字模式下將之從 DOS中載入，但一旦載入後，便會轉成在繪圖模式中。

即使像是 Excel, Microsoft 的 Spreadsheet程式也是在繪圖模式下執行。如果並不是在一般狀況下使用視窗系統，則設置程式 (installation program) 會依使用需要設置適當的內容。

視窗系統還提供了方便使用者與 DOS溝通的一般以文字爲主的應用程式。當然，利用 4.0擴充記憶體它也提供了多工作用，足以與 Desqview 在許多方面相提並論。二者雖在很多方面互相呼應但卻不完全相同，特別是在虛擬控制程式介面上 (Virtual Control Program Interface (VCPI)) 的問題，Microsoft 的產品仍然尚未解決。至少就 VCPI 的不相容性來看，希望在擴充記憶體中執行多工系統的可行性大減。

然而視窗系統已經成功地發展成一個相當複雜的產品

--實際上是二種產品，一種專爲 80386的特性設計的，另一種則主要是爲了 80286。滑鼠也包括在功能之內。如果你不喜歡滑鼠或者你的桌子擺不下，那像 MicroSpeed 的 FastTRAP之屬的追蹤球 (track ball) 也可以提供所有相同的功能，甚至還有處理三維空間的能力。

更進步的加强視窗系統，在 2.1版中--可以視需要選擇 Windows/286 及 Windows/386，分別適用于 80286及 80386 獨特的微處理機晶片--Microsoft 加上了一個額外的記憶體區段，使得 DOS以百萬計的位元組又增加了 64K (其基底位址須維特在 1024K之內) 。這也是在前面所提到的 Extended Memory Specification (XMS) 中所描述的那額外的 64K。

上面所討論的，Microsoft 將之定名爲 High Memory Area(HMA) 。當然，當把 Windows/286用在 8088 或 8086 時 HMA是不管用的，因爲 8088 及 8086 根本無法表示 1024k 以上的位址。但加强型的 286版則可向下相容。

16.1 相容性的問題

BONANZA 螢圖 (R)

在這裏嘗試著把許多關於擴充記憶體的話題提出來討論是相當適合的，我們可以透過視窗系統來看。視窗系統的歷史及其演進的過程可以代表許多軟體的演進過程。其中第一項就是突破 640k 的限制。

在最後，視窗系統面除了其他軟體也同樣面臨的一個事實--老型的個人電腦已經過時，例如模型 T。但有一些模型 T 的機種仍然繼續在工作，而且很有可能繼續下去。

向下的相容性已被發展到一定程展，而相容的深度太長有時可能也會有一些缺點存在。長久以來所有在 DOS上執行的機器皆被視爲同種--好像把用汽油當燃料的車子歸爲同種一樣，即使是 AT ，被認爲只是速度較快的 PC。

但是以上的想法已不再是正確的了。

無論從機器本身或者從機器背後所發掘出來差異甚多的各自的特性來看，大家的目的不再是希望達到一個共同的基礎點了。80486 的發展則更加深了這個現象，它能與所有8086家族系列相容，已經遠遠地超過 386所開發出來的功能。

在電腦業間的聯盟是非常薄弱、易變的。有時候不能只看外表就了解內部的實際狀況。在這有二個問題，比較簡單的是延伸 (extended) 及擴充記憶體 (expanded memory) 之間的問題。擴充記憶體對許多應用軟體來說是適用的，但是它有所限制。當把一株植物栽在較小的盆中，則它的根的發展因之而受到阻礙。應用軟體也常受苦於此類似的情況。如果只有一人將植物轉植到較大的盆--即現在有人有了 4giga 的位元組的固態 RAM，而且通通在一個區段內，而虛擬記憶體則變爲原來的 16 倍。如此一來，必定會有一些應用軟體會面對這偌大的記憶體而不知所措。

另一個較複雜的問題是：究竟要維持相容性的還是提供二種不同的產品以吸引不同的顧客好呢？一個可行的選擇是：儘可能地使得二者之間的距離縮小而不致於太過明顯。

本書的重點是記憶體本身，所以以下將著重在實際問題

，並試著看看一些先進的觀念。

16.2 燈光及陰影

BONANZA 堂圖 (R)

目前爲止，本書對擴充記憶體的討論皆在比較特殊而令人印象深刻的範圍。因爲令人印象深刻的較好閱讀。DOS Extenders 在 real mode及 protected mode 之間跳來跳去；多工也是令人印象深刻的--儘管每個應用軟體個別看起來皆相當平常。在令人印象深刻的範圍裏，我們使用電腦的一些通常因素有時會被遺忘。但是，我們平時利用電腦來處理生活中的一些事務--這正是它顯得有價值之處。

如果 expanded memory算是令人印象深刻的活，那 extended memory 就應該是 "驚人" 了。然而對發明這些東西的工程師來說。這樣就夠了。但是，它並不能只被看成是工業技術上的成就，眞正的價值是在我們與這些設備之間的互相作用。

這些設備的意義在於我們的軟體如何能從之獲益，及在於它使我們不得不仔細地審視這個領域的發展過程。有一些相當重要的軟體上的發展，當它們第一次宣稱有一新版軟體能使用 DOS extender 時，被評估爲相當有價值。

使用 DOS Extender 的結果是立刻對某些機器失去相容性，很明顯地 8088 等立刻被排除在外，因爲它根本無法承受擴充記憶體。接下來，如果繼續要求完整的 32 位元處理及虛擬無限制的區段大小，則 286將也會被除名。這相當引人注目，特別是這些名詞已經成爲 "家庭常用名詞" 了。

能從 expanded memory中獲益者不太多，因爲這項產品在沒有 EMS的機器中仍然繼續工作。

是否聽說過 XYWrite？它是許多專業作家最愛用的文字處理程式 (Word processor)，它的好處在使得可利用空間增加了一倍。XYWrite 第一次使用擴充記憶體，把它的字典，辭典拼字機等共 112k，移出傳統記憶區而放在擴充記憶內。並沒有過分誇張，但它的確發生了，這對知道如何使用它的使用者來說有著很大的改變。

其他軟體也有相同的情況。通常這種特色甚至不會在使用說明書上出現。一般皆認爲，如果軟體中有任何改變，就會有一個新的版本出來。其實並不盡然--重要的改變是如此，小的改變就不一定了。這些小改變通常會記載在 README 檔中。

以 DOS爲例。稍早我們提到 DRIVEPARM磁碟機，在 DOS 3.2 中曾出現，在 3.3版中消失了，但在 4.0版中卻又再度出現。

這類改變並不會出現在廣告或文件中，一種產品將會有不同的方式來處理 3.2及 4.0的擴充記憶體。例如 Borland 公司的 Sidekick Plus會使用到擴充記憶體，但若沒有，也照樣可以工作，但它本身是一個大程式，且尙須 384k 額外的記憶體及 1.5 megabytes的硬碟空間來供設置時用。

在設置完成後的需求就不再這麼大。如果有 4.0的擴充記憶體，它會在適應之後開始發揮功能。它也可以使用 3.2 擴充記憶體，但會受到相當的限制，事實上，它會願意使用磁碟而不用 3.2EMS 。這些都不會出現在手冊上，你只有在

不同的環境下去執行程式才會有機會了解眞相。

在 4.0中並沒有任何新名字出現，或就簡單地稱之爲擴充記憶體。也許我們該爲 AST的 "EEMS" 感到迷惑；因爲 AST 並沒有明確地說明，甚至沒有給任何線索來表明 EEMS 究竟是什麼東西--它有什麼意義。但至少它與 EMS區分開了。就一個純實用的觀點來看，我們所擁有擴充記憶體有三種，而非二種。

首先是 3.X expanded memory，它是被 Lotus及 Intel 等公司設計來供儲存資料用，並沒有別的特色。

接下來是 4.X expanded memory。當然 4.x能作 3.x所有能做的事，此外 4.x還可以在擴充記憶體內執行眞正的完全大小的程式。4.x 又能提供另一種 (或數種) 作業環境。例如多工系統。

最後是 extended memory。而 extended 又可以再分成二種程度，依機器而定。

然而，在用 expanded memory時常是將之全部堆在一起。這樣不好，因爲使用者將無法辨視其中的差異及各別的特性--如果能清楚這些，將會有很大的助益。

有一點必須澄清：即 expandeed memory 並不一定比 extended memory 的功用少，功能差。此二者不相同的，所以適用于不同的需要。若給定一組需求，則應用軟體也許會從 expanded memory中獲得較大的利益。同時，也可想像有某一種應用軟體會選用 extended memory是因爲 expanded memory 無法滿足它的特別需求之故。所有事都是相對的。

每件事皆是相對的--即使是我們所需要的記憶體的大小也是一樣。在某特定的數值以下將無法工作，但在其之上就可以了。

16.3　到底要多少記憶體

大多數人都曾使用過相當多的記憶體--無論數量多少。記憶體是加成性的。首先加 1 meg--甚至超過你認爲你所需要的。但是沒多久之後，你會發現用完了！系統又缺乏 RAM 了，而正在把東西放到磁碟上去。又須要多加 1 meg，如此繼續。

暫且不管執行更大的軟體的可能性，如果你常要執行很多程式你很可能在 Windows、Desqview或其他的視窗環境下執行工作。甚至有時不需眞的多工系統，只要用軟體做程式的切換工作就足夠了。使用者會希望載入一個以上的應用軟體及資料檔，並且每個都是開啓的，而且隨時能夠彼此互相切換--二個、三個甚至更多個不需要關閉的檔案而不必要經常載入、換出的軟體。

然而，眞正的記憶體並不便宜。無論是 Expanded 或是 Extended的價格基本上是相同的，但當用到 DOS-Extender-based software 時也許需要更多一點記憶體。至於 OS/2 則在載入 OS/2 的作業系統時將須要更多的記憶體，比大多數的使用者曾經用過的都多。

事實上，我們將會面臨一個本末倒置的狀況，就是：電腦的價格不再是考慮的重點，所必須注意的都是究竟需要多

少記憶體？價格到底多少？

尚有更甚者。當談論到 80386時便會提及它所能支配的 10^9(giga)byte。 但以今日的市價來計算，要購買這麼多記憶體使須的錢約與天文數字相當。即使價格不再是問題了，卻要到那去放置 36000個 1-megabyte 的晶片呢？還有，要到那去找到這數千瓦功率的電源？非此，不足以驅動整個系統。

但並不是這些問題將永遠困擾桌上軟體的使用者。事實上從今天的觀點來看，那些目標的達成也是指日可待的。

新的工業技術已完成了一組模組化的 RAM磁碟記憶擴充系統，足以擴充到 704 begabytes (0.7 gigabytes)，即所謂的 DartCard system。它不能插在一單槽內。而必須設置在磁碟機空間中，經由數種介面模組 (PC/AT bus, SCSI 或 ESDI) 來與系統連結。將四個這種模組連在一起 (可以作到) 。就可以突破 gigabyte 的籓籬而一舉得到 2.8 gigabytes 了。

此外，記憶體尚有偵錯及改正的功能(Error Detection and Correction, (EDC))。直到現今，RAM 的主要用途是利用它的快速度來代替磁碟機做存取的工作 (最快可以高達 500 倍)。

若要大量的買晶片，用 1-MEG/SIMM 所組成的 1 gigabyte記憶體，約需要花費$250,000。雖然對大多數用途來說仍嫌太貴，但與天文數字相較已有相當大的改進了。

至於數千瓦的動力來源--想想看如何從 65 Watt的電源

提高至可供一般 PC 使用。在今天，1000 megabytes只須 7 瓦的電源即可--那是用 1-meg的晶片組成的，而現在有了更大容量的晶片以及更有效率的運作。

從 Gigabytes到 terabytes。這些數字的確令人驚訝，但我們今天視之尋常的，在過去也曾被認爲不可思議。在 DOS 超越了 640K 以後，記憶體容量的極限都是無遠弗屆的了。

附錄A LIM 4.0 EMM 所提供的高階函數

下面所描述的是在 LIM 4.0 EMM中可供使用的一些高階函數:

編號	函數說明
1	Get Status測試擴充記憶體之硬體是否正常工作。
2	Get Page Frame Address可獲得 Expanded Memory Manager 所在的頁框的區段位置。
3	Get Unallocated Page Cont 可得知系統中擴充記憶體所提供的總邏輯頁數及尚未分配出去的頁數。
4	Allocate Pages通知系統 EMM程式將要使用擴充記憶體, 並分配所要求數目的邏輯頁給它, 然後掌握操作權。
5	Map/Unmap Handle Page 把設定給某一個操作者的邏輯頁中的一頁映射至 Expanded Memory Manager的頁框中四個實際頁中的一頁。

編號	函數說明
6	Deallocate Pages把某個操作者所擁有的邏輯頁全部釋出並交出操作權。
7	Get Version 傳回 Expanded Memory Manager軟體的版本號碼。
8	Save Page Map 把所有擴充記憶體裏的頁映射暫存器(page mapping register) 的內容存入一內部儲存區。
9	Restore Page Map爲某個特定的操作者把頁映射暫存器的內容從內部儲存區存回原處。
10	留供以後擴充用。
11	留供以後擴充用。
12	Get Handle Count傳回系統中開啓 EMM操作權的數目
13	Get Handle Page 傳回某個 EMM操作者所擁有的總頁數。
14	Get ALL Handle Page 傳回一個陣列, 其中包括每個活動中的操作者其所擁有的頁數。
15	Get/Set Page Map把所有可映射的記憶體區域 (包括原本的及擴充的) 的映射內容存入由應用軟體所提供上陣列之後再將寫回原處。
16	Get/Set Partial Page Map可以替某特定的記憶區存入部分映射的內容。
17	Map/Unmap Mulltiple Handle Pages可以映射 (或反映射 (Unmap)) 邏輯頁至實際頁, 只要系統提供的實際頁皆可。
18	Reallocate Pages可以增減分配給某個操作者的擴充記憶體的數量。

編號	函數說明
19	Get/Set Handle Attribute允許應用軟體來決定及設定關於操作權的一些屬性 (Attribute)。
20	Get/Set Handle Name 能得知某個操作者現在的名字(8 個字元)並給定一個新名。
21	Get Handle Directory傳回關於活動中操作者及其名字的資訊。
22	Alter Page Map & Jump 可以改變記憶體映射內容並把控制權轉移至某特定位址。
23	Alter Page Map & Call 改變映射內容及把控制權轉移至某特定位置。此外還可以回復原始狀況，即把原本的內容存回並把控制權交回給呼叫者。
24	Move/Exchange Memory Region 可以把某一塊記憶體的內容拷貝到另一塊記憶體上，或二者互相交換。可以是原本到擴充的，原本到原本的，擴充到原本的或擴充到擴充的記憶體。
25	Get Mappable Physical Address Array 將系統中每一個可映射的實際頁的實際頁號碼及區段位址放入一個陣列，並將之傳回。
26	Get Expanded Memory Hardware Information傳回有關硬體裝置擴充記憶體的能力的資訊。
27	Allocate Standard/Raw Pages 分配給作業系統標準的或非標準數目的頁，並從 EMM之中給一個操作者來負責這些記憶體。
28	Alternate Map Register Set允許應用軟體模擬不同組的硬體映射暫存器。
29	Prepare Expanded Memory Hardware for Warm Boot 爲溫開機準備好擴充記憶體的硬體設備。

編號	函數說明
30	Enable/Disable OS/E 使作業系統設計者能控制爲作業系統設計的功能--使致能 (enable) 或使無效 (disable)。

附錄B 16進位的基本運算

以下所討論的是關於位址 (address)--電腦內的位址。電腦所做的每一件事都與位址有關。設備--例如印表機、鍵盤、螢幕等--等是利用位址來沒置的。軟體則自己設置在某位址上。準確地說，一共有 1048567個唯一的且可辨視的絕對位址--或可以用另一種較簡單的表示法 FFFFh。

很明顯地，16進位要比 2進位表示法簡單多了。在這裏將只討論一些比較基本的觀念運算--已足夠簡化不少事情--，而不深究。表 B-1中，1,048.576 轉成 16 進位表示法後有20個區塊 (block)。

十進位是以十爲一數，它的好處是在方便--至少對人來說是的。而電腦則是以二爲一數，原因是每個電晶體的輸出都只有二種，0 及 1。但是電腦以二進位運算時速度卻是比人用十進位運算要快得多。有時候，必須要配合電腦的數字系統，但卻不必直接去面對一長串的 0與 1。

16 (2^4) 是一個很好折衷點；前十個區塊(block)

(0-9) 就包含了整個 640k 的傳統記憶區，可供一般應用程式使用。一串不可能記得住的十進位數字將被轉換成簡潔、整齊的 16 進位數字。

到目前爲止一切看起來都不錯，但我們還需要六個新的字元來滿足以 16 爲一數的表示法。我們用字母 A、B、C、D、E及 F。現在，各位的手指不夠用了吧，不妨試試腳趾。總而言之，現在共有 0、1、2、3、4、5、6、7、8、9、A、B、C、D、E、F 等 16 個數字。

在圖 A-1中可見到計數法已進步到用一連串的 64K的區塊來表示大小--實際上並非是完整的 64000，而只有 65536。但是以 16 進位的表示法來看確實是相當漂亮的一組數字10000h。這正是之所以要了解 16 近位的目的了。然而，在近來的書裏頭常可見到的有二種位址表示法。二者都是用16進位的表示法，數字與字母混合使用，但有的用四位，有的用五位。

事實上，即使使用 16 進位仍然須要 5位數才夠用以代表 1024K位元組的位址。更準確地說，其須 20 個位元才能指向任何位址。但是 8088 只有 16 位元的位址暫存器可用，所以只得把位址分成二個部分，以配合暫存器來使用。結果，位址被分成一組 4個字元的區段位址 Segment (CCCC:) 加上另一組四個字元的補償 (offset)，使得位址看起來如下:

0000:0000

然而，爲了方便的目的儘量減少數字，所以此處的區段位址就包括了一切需要的資料在內。

爲了簡單起見，本書將用十近位的近似表示法來表示記憶體的區塊--例如 64K--。但是 DOS裏重要的中斷點 (break point) 皆位在 65536的倍數的位址上--16進位的 10000 的倍數。所以，65536 的三倍是多少？我們可以很快地知道是 16 進位的 30000，或者是去掉一個 0則可以得到一個 4位數的區段--3000。從上可知，16進位的確在處理電腦上爲我們省了不少事。

表 B-1 十六進位及二的乘冪

位址接頭	可能的位址
1	2
2	4
3	8
4	16
5	32
6	64
7	128
8	256
9	512
10	1024
11	2048
12	4096
13	8192
14	16384
15	32768
16	65536
17	131,072
18	262,144
19	524,288
20	1,048,576
21	2,097,152
22	4,194,304
23	8,388,608
24	16,777,216
25	33,554,432
26	67,108,864
27	134,217,728
28	268,435,456
29	536,870,912
30	1,073,741,824
31	2,147,483,648
32	4,294,967,296

附錄C　名詞解釋

alternate register set

是一組額外的指標，可供超過 64k以外的資料讀取（在 3.2 EMS 之下）或者提供一組 16K的區塊來填原本的記憶體或讀存擴充記憶體（在 4.0 EMS之下）。備用暫存器組是必須的，因爲單有一組暫存器只夠指向一組 16K 的區塊（邏輯頁）。

ASCII

是 American Standard Code for information

Interchange 的縮寫。是用在不同機器間傳輸使用者可辨視字元組的一種工具。

beta test

在一般分配前，對一種產品實際運作試的最後狀態。

BIOS

是 basic in/out Services的縮寫，是一組用以在低階管理電腦動作的指令。

boot Sector

一個特別的磁區 (disk sector)供電腦啓動時載入作業系統之用。

byte

計算電腦記憶體的基本單位。一個字元--例如：一個字

母，數字或驚嘆號等--作用一個 byte 。一個位元組(byte)包括 8個位元 (bit)。

Conventional memory

指的是低於 640k 的記憶體，有時會增加到 704k 或更多，依影像顯示的種類而定。

Configuration

包括一群組成一個電腦系統的所有硬體及軟體。通常會有鍵盤、顯示器、任何印表機，作業系統，任何應用軟體及硬體。

data compression

壓縮資料使得所需佔用的儲存空間減少。

device driver

一組軟體包括執行某一裝置的規格表 (specification)。當叫用之後便能使裝置運作，並可與之溝通。

DOS

是 Disk Operation System的縮寫。通常是一般用法，但在本書中是專指 Microsoft作業系統，標記爲 IBM DOS (有時爲 PC DOS) 及 MS-DOS(經常被電腦商改名，例如 COMPAQ DOS 或 TANDY DOS等)。

EEMS

是 Enhanced Expanded Meemory Specification的縮寫。是早期的 LIM 3.2 EMS的加强版。包含 EMS的所有功能，有許多特色與 LIM 4.0 EMS相時應。

executable file

是不需要再編譯即可執行的檔案。其內容是機器語言。

expanded memory

在 1百萬位元組 (megabyte) 以上的非線性記憶體。是由 Expanded Memory Manager (EMM)所設置完成的以區塊爲單位的記憶體。EMM 必須在 DOS的 1 megabyte 限制之內。

extended memory

在 1 megabyte 以上的線性記憶體。可以由 80286及更高的微處理機直接使用。也許可以用作模仿 expanded memory。

gigabyte

10^9 個位元組。

hexadecimal

以 16 爲基底的數字系統。其來自電腦的二元性(2^4)而無法直接處理 10 進位的數值。

high RAM

是指映射至 640K 以上的未使用的記憶體的 RAM。

kilobyte

10^3 位元組。

LIM

是 Lotus/Intel/Microsoft的縮寫，用以描述 EMS的發展過程包含了多家公司的共同努力。

linear memory

指直接可讀寫的記憶體。由傳統記憶體 (從 640k 到 1MB)擴充記憶體 (1MB 以上的) 組成。以供 80286及 80386 爲主的機器使用。

logical page

一個大小爲 16k的記憶體區塊，在 3.2 EMS下只能用在640K以上的區塊。在 4.0 EMS下則是 1024K以下的區塊皆可以。

low RAM

見 "conventional RAM"。

megabyte

10^6 位元組。

multitasking

同時執行多個應用軟體。在 DOS下可以利用一個多工或視窗環境，例如 Microsoft Windows或 Desqview 等來達成。而保護模式的作業系統，例如 Xenix及 OS/2 則本身具備這功能。

nonosecond

10^{-9} second。

Overhead

任何被載入的軟體所佔用的記憶體。作業系統需要相當大量的記憶體--也是 overhead。

Page

指一塊 16k的資訊可以從擴充記憶體儲存至傳統記憶體，但必須藉由整排交換程序 (bank switch process)。

page frame

是 page 在被交換時所用到的一塊區域。

page register

一塊記憶位置，像是一個點一樣，指向記憶體中的某個位址。

paged memory

記憶體被分成 16k的區塊之謂。

perfect superset

用以描述一組函數功能及命令超過了系統原本提供的服務，但仍維持相同的相容性。例如，EEMS就是一組 LIM EMS 的 perfeect superset。

port

指的是實際位址或是週邊設備與 CPU的溝通管道。

print spooler

在 RAM中一塊區域，用以存放即將送到印表的資料，然後可以讓處理機去做別的工作。雖然這不算是多工(multitasking)，但也能讓使用者在印列尚未完成時，即可去做別的工作。

protected mode

是一種運作的特別模式，允許定址在 16MB 以上的擴充記憶體。現在的 Xenix及 OS/2 等不同類型的作業系統皆有。80286 也可以使用。

RAM

是 Random-Access Memory 的縮寫。是可變內容的記憶體；在處理運作時可用作儲存程式及資料的區域。

RAM Disk

把 RAM中的一塊區域設定資料的快速讀寫用。當資料被

置於 RAM Disk 中時，若有某程式要讀寫它則可以很快地動作，比從軟碟或硬碟中去讀都要快。

register

用作指標指向擴充記憶體中某個 16k的資料區塊。

register set

某個程式或應用軟體所用到的所有的暫存器的集合。

remapping

指把未用過的位址 (通常在 640k 以上) 設給實際 RAM 中的區塊，使得它們就像在這些位址上一樣。一般限用在 80386系統，是由微處理機晶片自行提供的。

ROM BIOS

低階的 BIOS ，是當系統開啓時從 Read Only Memory

中載入。

SIMM

是 Single Inline Memory Module的縮寫，爲一個迷你電路板通常包括有一完整的--通常有九個--記憶晶片。這種記憶體單元比 DRAM 佔用較小的空間，但卻需要特殊的接合技術。

Split Memory Addressing

是 AST所用的術語，指的是他們的電路板能夠把記憶體中的一部分用作傳統記憶體 (最多到 640k)，並把剩下的用作擴充記憶體。

superset

用以表示某組函數及命令超過了系統原本提供的，而有額外的服務。

terabyte

10^{12}位元組

time slicing

通常用在交換的技術上。目的在把單一的微處理機分配給二個以上的應用軟體，如果交換得夠快便可以藉之產生多工的假象。

thrashing

當分配給應用軟體的一個 time slice 不夠來完成讀寫動作時，則在每次時間快到時再重新來一次直到完成爲止。

upper memory

有時亦稱爲 "reserved memory"，是指位於 640k 到

1MB 之間的記憶體而言。它原本是供系統功能，例如顯示區，ROM BIOS 及其他附屬功能。現在，則包括了供擴充記憶體用的頁框及從傳統記憶區 (640K)裏 remapping 後的諸多功能。

Vdisk

是一個 IBM的軟體，能在 RAM中產生一個虛擬的磁碟機。

volatile

用作描述記憶體，只要接有電力就可以保有其內容。一般的 RAM即是如此。

凡書號第四碼爲英文字母之書籍
A: 書本及可執行程式 (不含 SOURCE)
B: 只有書本不含程式
S: 書本及可執行程式 (含 SOURCE)

本公司書背上A 1～D 5代碼係依書本內容深淺而定，供讀者選書之參考。A到E代表5級，每一級再細分爲1～5階，如: C比A深，而C 5又比C 2深。

dBASE (1)

等級	書號	書名	定價	頁數
A3	36210	學習dBASE III +資料庫管理(程式設計篇)	$150	P. 350
C3	39410	CLIPPER 技術手冊	$200	P. 656
C2	39510	CLIPPER 指令與函數參考手冊	$220	P. 736
C1	44810	區域網路 (以 dBASE 應用爲例)	$200	P. 157
C4	48210	如何設計會計系統	$290	P. 576
A2	50910	dBASE 實習入門	$160	P. 335
B2	52010	CLIPPER 字典 (指令與函數)	$350	P. 890
A2	52210	dBASE 資料處理實務入門	$180	P. 376
C4	54010	dBASE IV 程式設計	$220	P. 464
B2	55110	dBASE 程式技巧與陷阱	$220	P. 496
B1	57510	dBASE 資料庫管理	$200	P. 352
B1	58610	dBASE 資料處理入門	$150	P. 320
A2	62910	"諾頓 (NORTON)" dBASE、LOTUS、PCX 圖形檔, 工具箱	$150	P. 240
B3	64210	DR.COMPILER 804 使用手冊	$220	P. 416
C1	64310	DR.COMPILER 804 使用技巧	$200	P. 536
B1	65910	dBASE 程式設計實務	$260	P. 400
B1	63710	dBASE III 指令函數字典	$280	P. 832
B1	63719	dBASE III 字典 [小字典]	$280	P. 624
A2	67310	dBASE (基礎使用)	$150	P. 256
C1	68210	CLIPPER 5.0 使用手冊 (指令函數)	$280	P. 912
C1	68219	CLIPPER 5.0 使用手冊 (指令函數)[小字典]	$280	P. 432
C1	68310	CLIPPER 5.0 使用手冊 (程式設計入門)	$220	P. 496
C1	68410	Clipper 5.0 技術手冊	$220	P. 544
C1	70010	Dr.Compiler 804 入門	$260	P. 416
C1	70410	以 400個實例探討 CLIPPER (5.0)	$240	P. 480
C1	70419	以 400個實例探討 CLIPPER (5.0) [小字典]	$240	P. 512
A2	70610	dBASE 學習手冊	$120	p. 224
B3	70710	dBASE III PLUS程式設計篇	$280	P. 528
A1	70910	dBASE 快速入門	$180	P. 296
B1	72310	Clipper 入門手冊	$280	P. 376
B2	72410	Clipper 技術研究	$280	P. 416
B2	72710	dBASE 程式設計技巧與應用	$250	P. 520
C2	73110	資料庫的基本觀念—以 dBASE爲例	$220	P. 400
A3	74010	dBASE 教室	$280	P. 448
A3	74310	一學學成 dBASE	$280	P. 960

C (2)

等級	書號	書名	定價	頁數
C4	35720	POPUP 1.0 C 擴充程式館	$300	P. 528
C4	35720	POPUP 1.0 C 擴充程式館 (含軟體)	$999	P. 528
C4	36520	C 與資料庫管理系統	$260	P. 500
C3	40820	ASSEMBLER 或 C與各種語言之聯結	$90	P. 191
C3	41220	MICROSOFT C 5.1 進階與應用	$340	P. 820
C1	42420	MICROSOFT C 5.1 程式館手冊	$240	P.1090
C1	42520	C 語言教室	$220	P. 450
C1	42620	MICROSOFT C 使用手冊(適用5.1版)	$160	P. 424
C1	43920	TURBO C程式設計備忘手冊 1.5 2.0	$220	P. 460
D2	45120	TURBO DEBUGGER 使用手冊	$280	P. 530
C1	51320	圖解 TURBO C 程式設計入門	$160	P. 500
D3	54220	C 語言繪圖技巧	$280	P. 488
C1	59620	C 程式設計入門	$280	P. 992
B1	61820	資料結構-以C實作(上冊)	$290	P. 674
B2	61920	資料結構—以C實作(下冊)	$260	P. 560
C1	62120	TURBO C 指令函數字典	$240	P. 752
C1	62129	TURBO C 指令函數字典 [小字典]	$180	P. 304
C1	62820	TURBO C 指令函數速查手冊	$140	P. 384
D5	66220	以 C設計 PC TOOLS 之祕訣	$220	P. 352
C1	67520	TURBO C ++字典 (指令函數)	$280	P. 880
C1	67620	TURBO C ++手冊 (指令函數)	$280	P. 880
C1	67820	深入了解 TURBO C	$250	P. 464
C3	68020	以 100個實例探討 C語言	$280	P. 592
C1	68120	TURBO C++ 使用手冊	$210	P. 432
C1	69020	TURBO C 2.0 使用手冊	$210	P. 784
C1	69320	TURBO C++ 入門指引	$220	P. 328
C1	69420	TURBO C++ 程式設計師手冊	$300	P. 592
D5	69620	TURBO C繪圖之技巧	$300	P. 368
C1	70120	認識 C語言	$250	P. 496
C1	72820	C 語言ㄅㄆㄇ	$250	P. 520
D2	73220	常駐記憶體程式技巧及應用	$200	P. 432
C1	73820	MICROSOFT C X.x程式設計學習手冊	$280	P. 624
C1	74420	一學學成 TURBO C (1990夏第十四更新版)	$260	P. 816
C1	74620	C 語言武功祕笈	$280	P. 752
C1	74729	以 400個實例探討 Turbo C [小字典]	$180	P. 400
C1	75020	TURBO C 2.0 程式館參考手冊	$260	P. 896
D4	75320	高等 C程式庫設計範例	$290	P. 544
D4	75420	C 語言除錯技巧—CodeView	$280	P. 416

DOS (3)

等級	書號	書名	定價	頁數
A1	38430	DOS 4.01 學習與使用手冊	$200	P. 992
A1	44230	MS-DOS 4.01 使用手冊及範例詳解	$200	P. 992
A1	49130	精編 DOS 3.3 使用手冊	$100	P. 320
A2	51230	學習與活用 DOS	$240	P. 680
A1	51930	PC 檔案管理與維護	$160	P. 320
A1	56330	DOS 字典	$250	P. 488
C1	57330	DOS 與組合語言技術手冊	$350	P. 840
A1	64130	DOS 使用入門與進階	$150	P. 320
A2	66130	MS-DOS 應用手冊	$280	P. 496
A1	68530	DOS 5.0指令使用手冊	$280	P. 560
A1	68539	DOS 5.0指令使用手冊 [小字典]	$250	P. 448
A1	69230	DOS 3.3 使用與實例學習手冊	$150	P. 736
A1	69730	DOS 5.0 學習使用手冊	$220	P. 384
A2	70330	DOS 與批次檔之技巧	$150	P. 304
D5	71130	DOS 系統函數呼叫手冊	$220	P. 376
C2	73330	突破 640K 程式設計技巧	$240	P. 336
A1	75230	IBM PC DOS 基本教材	$220	P. 496

BASIC (4)

等級	書號	書名	定價	頁數
A1	40640	BASIC 入門教室	$120	P. 285
A1	54140	圖解 BASIC 教材	$150	P. 320
A1	57840	中學生的IBM PC BASIC 入門	$100	P. 264
A1	71040	BASIC 程式範例及解說	$220	P. 360

文書處理與中文 (5)

等級	書號	書名	定價	頁數

A1	46750	零壹中文學習與使用	$170	P. 500
A1	59350	PE3(DW3) Profile 設計技巧	$150	P. 240
A2	59450	PE3(DW3) 中文編輯學習手冊	$160	P. 288
A1	61350	倚天中文學習與使用	$160	P. 312
A1	61450	倚天中文學習與使用 (附學習磁片)	$200	P. 312
A1	63250	文書處理—計算機應用 (上冊) 依 "國際貿易" 科系教授規劃	$140	P. 352
A1	63350	文書處理—計算機應用 (下冊) 依 "國際貿易" 科系教授規劃	$140	P. 256
A1	63450	文書處理—計算機應用 依 "商學" 教授所訂章節內容	$140	P. 500
A1	67250	PE3(DW3)快速入門 (1990)	$180	P. 320
A1	70550	文書處理實務 PE3(DW3)計算機應用	$220	P. 512
A1	71450	PE3(DW3)專業版六大擴充功能	$260	P. 280
A1	71750	PE3 (DW3) 與各種中文系統使用手冊	$260	P. 472
A1	71850	倚天中文及 PE3 (DW3) 之應用	$260	P. 472
A1	73550	PE3(DW3) USER'S MANUAL 使用手冊	$260	P. 336
A1	73950	PE2 進階 PE3	$280	P. 448
A1	74250	文書處理實典	$260	P. 544
A1	74550	PE3(DW3)中文/英文 文書處理	$280	P. 816
A1	74850	PE3(DW3) PROGRAMMER'S GUIDE指令應用手冊	$260	P. 528
A1	74859	PE3(DW3) PROGRAMMER'S GUIDE應用手冊-字典	$260	P. 480
A1	75550	PE3(DW3)中文/英文 文書處理 (上冊)	$140	P. 416
A1	75650	PE3(DW3)中文/英文 文書處理 (下冊)	$140	P. 416

組合語言 (6)

等級	書號	書名	定價	頁數
D4	42860	"組合語言" 在週邊介面上之活用	$230	P. 400
C1	44460	組合語言程式設計範例及解剖	$220	P. 380
C2	50860	組合語言學習與應用	$280	P. 656
C1	54460	PC 硬體架構及系統軟體原理	$280	P. 704
C1	57260	組合語言根基	$320	P. 784
B1	59560	組合語言程式設計手冊 (含 80386)	$300	P. 752
C3	60260	組合程式範—週邊控制	$280	P. 528
C3	60360	組合程式範—數學運算	$280	P. 624
C3	60460	組合程式範—實務 (含繪圖)	$280	P. 464
C3	61160	組合語言在硬體上之應用 (入門)	$240	P. 340
C1	62060	PE3 & Assembler	$220	P. 304
C1	63060	SoftTip 組合語言字典 (1990 夏)	$280	P. 720
D4	63560	DOS & BIOS 實例學習教材	$140	P. 330
D2	64460	TURBO C/ASSEMBLER 與其它語言聯結之技巧	$240	P. 448
C1	66360	IBM PC 組合語言詳解 (附贈 PA 軟體及範例程式)	$170	P. 352
C1	66460	IBM PC 組合語言詳解	$120	P. 352
D4	67760	" 組合語言"在週邊介面之活用	$180	P. 320
D5	69560	如何以 ASM設計 PC TOOLS之祕訣	$220	P. 400
B1	71960	BIOS技術手冊	$290	P. 520
C1	72260	MASM 使用手冊 (MACRO ASSEMBLER)	$290	P. 582
D2	74160	組合語言實例解析	$250	P. 464
D1	75160	BIOS & DOS 的應用實例	$280	P. 432

電腦應用軟體 (7)

等級	書號	書名	定價	頁數
A2	35170	電腦化財務會計系統(計算機應用)	$280	P. 584
A2	56070	LOTUS 速成	$300	P. 464
A2	56570	LOTUS 1-2-3 使用手冊 (3.0 版)	$230	P. 496
A1	56770	PC TOOLS 5.5 大全手冊	$380	P. 928
A2	56870	PC TOOLS DESKTOP 5.5 使用手冊	$220	P. 496
A1	56970	PC TOOLS 5.5 公用程式使用手冊	$220	P. 496
A1	62470	國際貿易電腦化作業(上冊)—計算機概論 (軟體贈送)	$160	P. 480
A2	62570	國際貿易電腦化作業(下冊)—計算機概論	$160	P. 384
A1	62670	國際貿易電腦化作業(合訂本)—計算機概論 (軟體贈送)	$300	P. 832
A2	65570	PC TOOLS 6.0 大全	$380	P.1080
A2	65670	PC TOOLS 6.0 DESKTOP 及 BACKUP 使用手冊	$250	P. 640
A2	65770	PC TOOLS 6.0 PC SHELL 及公用程式使用手冊	$200	P. 464
A1	68670	最新 PC TOOLS 使用手冊	$140	P. 240
A1	71270	以 300個實例探討 LOTUS	$290	P. 704
A1	71279	LOTUS 指令函數字典 [小字典]	$260	P. 608
A2	72070	LOTUS 套裝軟體應用入門	$240	P. 464
A2	72170	LOTUS 1-2-3 學習與速查手冊	$240	P. 448
A2	72670	LOTUS 入門與應用	$300	P. 608
A3	74970	WINDOWS 3.0 使用入門	$260	P. 496

電腦基礎入門 (8)

等級	書號	書名	定價	頁數
A1	63680	中英電腦字典 (1990夏第五版)	$380	P.1120
A1	64080	電腦化大全 — 適合學校老師進修	$280	P. 660
A1	67980	IBM PC電腦入門	$210	P. 570
A1	73080	電腦化大全 — 適進修培訓	$280	P. 608

其他 (9)

等級	書號	書名	定價	頁數
D3	47990	EGA 與 VGA 程式設計實務	$400	P. 512
C1	55690	TURBO PASCAL 5.5 使用手冊	$250	P. 632
C2	55790	TURBO PASCAL 5.5 程式設計	$280	P. 704
D5	57190	80386 全貌及奧祕	$180	P. 280
C3	59190	"諾威爾" 網路使用指引	$280	P. 432
C3	60590	"諾威爾" 網路指令應用手冊	$220	P. 464
C1	64590	檔案在硬式磁碟之規劃實務	$200	P. 392
C3	66590	系統分析 (附贈程式範例磁片)	$220	P. 480
D2	71390	EGA 與 VGA設計剖析 (1991年第四版)	$280	P. 376
C1	72590	諾威爾 (Novell) 網路系統函數技術手冊	$340	P. 432
A1	73690	尖端電腦新知第 26 期	$90	P. 256

本公司出版套裝軟體

等級	書號	書名	定價	頁數
A1	47270	PE3 文書博士(DW3)	$4999	P.728
A2	51470	電腦化財務會計系統 (軟碟版)	$999	
A2	51470	電腦化財務會計系統 (硬碟版)	$9999	
A1	624A0	國際貿易電腦化作業(上下冊)—計算機應用	$999	P.832
A1	624B0	國際貿易電腦化作業(上下冊)—計算機應用	$300	P.832
A1	624S0	國際貿易電腦化作業(上下冊)—計算機應用	$9999	P.832
A1	626A0	國際貿易電腦化作業(合訂本)—計算機應用	$999	P.832
A1	626B0	國際貿易電腦化作業(合訂本)—計算機應用	$300	P.832
A1	626S0	國際貿易電腦化作業(合訂本)—計算機應用	$9999	P.832
A1	548B0	國際貿易押匯文件系統	$300	P.304
A1	548A0	國際貿易押匯文件系統	$999	P.304
A1	548S0	國際貿易押匯文件系統	$4999	P.304
C3	550B0	Dr.GRAPHIC (繪圖博士) VER.HERCULUS	$310	P.712
C3	550S0	Dr.GRAPHIC (繪圖博士) VER.HERCULUS	$999	P.712
A2	555B0	國際貿易電腦化系統	$300	P.760
A2	555A0	國際貿易電腦化系統 (軟碟版)	$999	P.760
A2	555S0	國際貿易電腦化系統 (硬碟版)	$9999	P.760
B1	637A0	dBASE III 指令函數字典	$350	P.832
A2	708B0	會計博士使用手冊	$280	P.272
A2	737B0	會計博士第三代使用手冊	$280	P.384

瑩圃讀者獎學金申請辦法

【條件:】

1.自認不論在那一水平基礎上，學習的態度，用心程度都優於別人者。
2.願意將自己學習過程之心得分享給其他後進之讀者者。
3.不論是初學者或已是電腦高手，只要對於本書之後進讀者有任何貢獻者。

【辦法及獎學金計算原則:】

1.在研讀過程中之筆記，足以幫助後進讀者提升學習效率者。 (每字 2元)
2.與其他同類書籍之比較，那家出版之那本書有何優點，是本書所沒有的。 (每字1 元)
3.那一頁範例或說明，雖正確，但您覺得還有更好的寫法 (如重寫簡潔易懂之範例，以分析圖解，製表輔助說明等) 每一範例 100元。
4.重要資料、技巧、說明，其它書 (指出書名幷請影印附上) 有提到本書遺漏者，每字 1元。
5.本書那一頁那一行說明，語焉不詳，每字 1元。
6.本書那一頁那一行發現一個錯別字每字 5元。

【說明:】

雖然瑩圃 (不論內容、印刷、紙張、封面設計) 不斷的在改進書本品質，或者可以做得更好但一定有美中不足的地方。依過去記錄，促使書本出版之著作、編輯、校對....等人員，當然功不可沒，依過去記錄，但更重要的是廣大讀者群以各種的角度對本書提出改進的意見，或提供其寶貴的心得、更新的技巧，甚至校稿時未發現之錯別字，使本書每一版都有大幅的進步。

這種讓每一位優秀的讀者都能成爲下一版著作者之一，這是一件多麼有意義的事，故瑩圃爲擴大獎勵更多的讀者參與，特訂獎學金辦法。例如:

若原書: 這些函數含有各種不同數值資料型態的絕對值，爲了使程式最小，可用上述 abs的巨集指令。
將其修改: 這些函數主要用來計算各種不同數值型態的絕對值，爲了使程式碼減至最小,可使用上述的函數呼叫方式加在程式中而取代 abs的巨集指令集。

瑩圃即以 63 個字數計算獎學金。 (註: 本辦法限出版一年內有效)。

來信請按下列格式書寫:

姓　　名:				民國____年___月___日生	
永久住址:				TEL: -	
目前住址:				TEL: -	
學　　歷:				科系:	
善長程式語言: ☐組合語言 ☐C ☐PASCAL ☐ dBASE ☐ BASIC ☐其它					
過去的特殊表現:					
書名		書號 (BALA NO)		出版日期	

【讀者獎學金之支付辦法:】

1.500 元以下以贈書禮券支付。　　2.500 元以上之部份，一半現金，一半贈書禮券。

贈書禮券: 向瑩圃經銷商郵購買書可以抵扣書價。

●你購買之 ＰＥ３ "文書處理" 軟體是 3.0以上新版？或是已被淘汰的舊版？

ＰＥ３ 3.45 版已於 1990 年二月發行，新增功能除原 PE2、PE3 1.0 版、WordStar、PCTOOLS 原有功能外，

1.中文排版功能。

2.利用游標劃線：

在螢幕上我們除可以游標劃線外，並具智慧性的處理線條交叉：如當兩條線成交叉時會自然呈十字線條。

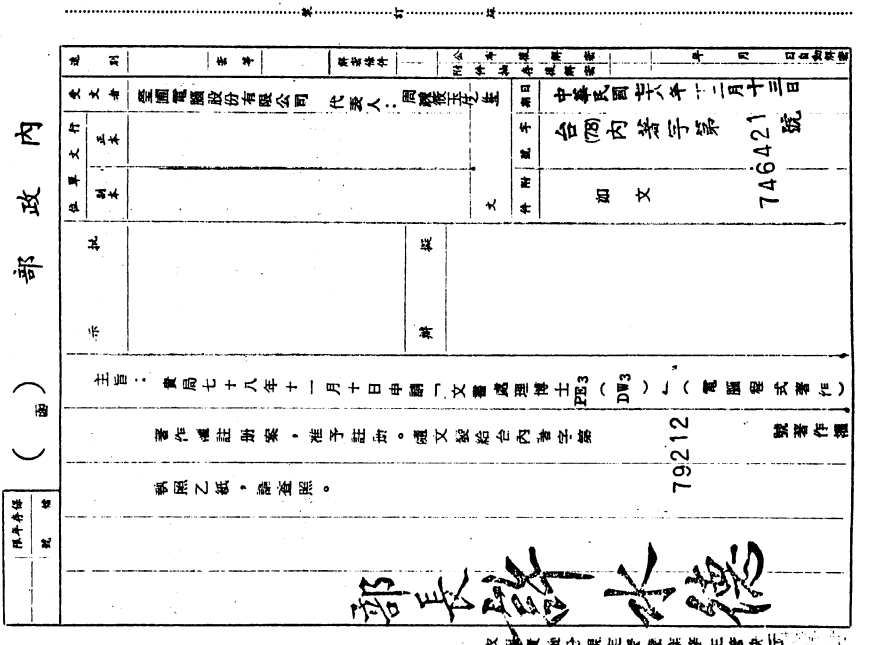
3.計算功能：

傳統計算功能是利用計算機，但常發生按鍵錯誤。在 ＰＥ３ 只抓取檔案內資料計算，使計算更能精確可靠。

4.大量郵件處理 (MailMerge) 及印製郵寄標籤

●中文命名為 "DW3" 功能與 "ＰＥ３" 完全相同

請向各地經銷商接洽或 FAX:(02)321-6584、705-4472 以書面詢問



瑩圃榮譽會員———您讀書，瑩圃出錢！

親愛的讀者：

您是本公司之忠實讀者或會員，因而告訴您一個珍貴的消息--瑩圃正舉辦"評選榮譽會員"活動，而今您可以不必再似往昔一般須排隊擠公車購買電腦書籍，甚或錯失了好書而有遺珠之恨，既然您對電腦資訊持有一份執著、熱愛，何以不把握此一機會，申請成為瑩圃的榮譽會員，即可免費獲取目前市面尚未發行或即將發行的好書或軟體。

■成立宗旨：

1.免費以最新的電腦書籍及軟體，回饋於忠實讀者。
2.促進並鼓勵電腦知識的研究風氣。
3.更加提升瑩圃出版之書籍的品質水準。

■資格要求：

1.自認對電腦有"狂熱"者。
2.曾精讀本公司出版之書籍並能提出閱讀心得,筆記(自行撰寫之程式)供參考者。
3.經由雙方協議，榮譽會員於同意接下 BIC的程式或書籍後，應在約定的時間內，（最長不超過 30天）將其內容精讀及測試完畢後，指出其中有差異、不易了解或錯誤之處，並依能力所及加以修改。並不得將資料轉交他人。

■榮譽會員享有之特權：

1.得以優先測試國外最新且珍貴的原版程式磁片，例如目前於全世界方發行等軟體。
2.免費優先閱讀到最新的電腦資料及書籍，先享尖端電腦資訊的奧妙。

●榮譽會員申請辦法

1.請填寫榮譽會員--個人基本資料。（影印即可）
2.請立即寄至郵政信箱 44---89號Ｒ＆Ｄ榮譽會員服務組。

****

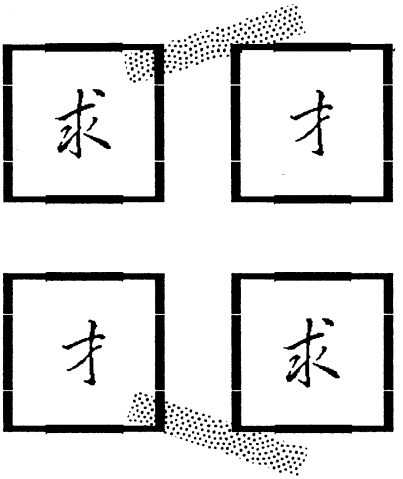
榮譽會員 ----- 申請表個人基本資料（可影印或自行謄打）

****

1.姓名：----------　　2.職業：----------
3.電話:目前（---）----------　永久（---）----------
（請填寫方便連絡）
4.住址:目前 ----------
永久 ----------
5.最高學歷：----------學校--------科(系) ----(年級)
6.精通的語言：A.C 語言　B.dBASE Ⅲ PLUS
C.MS-DOS　D.組合語言　E.其他
請依精通程度填寫 1:------ 2:------ 3:------ 4:------
7.曾經購買本公司的那些書籍（請以書號填寫即可）

8.對本公司之寶貴意見及建議。

9.家中是否有電腦？ A.有 ----- B.無 ----- C.硬碟 ---MB
10. 目前工作上常用到的語言
□ C語言 □ dBASE Ⅲ PLUS □ DOS
□ 組合語言 □ 其他（----------）
11.若 BIC 提供資料太多，你是否可另找尋 1～3 個夥伴一起研究。
12.隨函附上磁片--------片,閱讀心得--------頁供參考。



除了"錢"途以外，
您還需"前"途

朋友！除了追求前（錢）途之外，您每天享受挑戰、成長、奮鬥，以及成就的滋味嗎？在"營團"獲得的要比別人來的多，營團快速發展正代表著我們每一位的"成長"與"成就"。

朋友！假若您不想過一個平淡（無味）或喧囂卻不"實在"的人生。而羨慕我們每天享受挑戰、成長、奮鬥及成就的滋味，對前（錢）途充滿了信心與歡笑！

那麼不論您對於：
管理、財務、程式語言（C、dBASE、Dr.Compiler、ASM、Lotus、..）及業務推廣、企劃、會計、應用軟體、教材製作、.....等有特別的喜好及專長。

不論您擬全職的加入或目前僅能以兼職方式，都受歡迎！請立即附專長、自傳及簡歷寄：

台北郵政信箱 44-89　　周總經理親展

意見調查表

有許多讀者來函表示他們對瑩圃電腦書籍的看法。讚美之詞帶給R＆D工作人員莫大的鼓勵；責備及要求則促使我們更加努力。套句常用的話：

ㄐㄧㄣ ㄉㄨㄛ ㄒㄧㄚ！

從創業至今瑩圃一直抱著服務大眾的態度來經營，由於主管的要求：『必須以最快的速度，是供讀者最好的資訊』，所以R＆D的人員幾乎個個都是以公司為家。當然，在求新求快的前題之下，書本的一些小瑕疵也隨之而生。

例如：『損失』打成『捐失』，『問題』打成『門題』。會不會造成讀者在研究的過程中困擾，但是在求新求快的前題而的瑕疵，究竟孰輕？孰重？

所以特地製作了一個問卷調查，詢問大家的意見。

您覺得瑩圃的電腦書籍：

1. 錯字方面：□多得無法忍受　□是有一些錯字　□比例已經很小

1a. 錯字是否會影響讀者對於內容的吸收

□會　□不會　□可以接受

2. 文筆方面：□生硬　□普通　□不錯

2a. 文筆是否會影響讀者對於內容的了解

□會　□不會　□可以接受

3. 價錢方面：□貴得離譜　□合理　□很便宜

4. 內容方面：□有獨到之處　□稀鬆平常　□不夠

5. 封面設計：□生硬　□普通　□很好

6. 出書速度：□太慢　□普通　□快速

7. 版面編排：□滿意　□普通　□生硬

為了減少錯字，讓文筆更加流暢，版面更具變化...等。

1. 增加書本的售價，您覺得：

□可增加 20--30%　□可增加 10--20%　□不加價

2. 降低出書的速度，您覺得：

□可延 2--3 月　□可延 3--4 月　□不可延期

3. 改變排版方式，您覺得：

□可延 2--3 星期，加價 5--10%　□可延 3-4星期，加價10--15%

□不可延期，亦不可加價

希望讀者能把優點告訴大家，把缺點告訴我們，讓瑩圃與讀者更密切合作，共同成長。

學電腦

求新知

**** 在電腦的領域裡

到底有那些真正的新知 *****

請看『電腦新知』

它包含有：

- 1.暢銷軟體新知
- 2.專業技術新知
- 3.熱門資訊新知

----只有真正的新知來充實內容，不以廣告來擴充篇幅。

『電腦新知』已獲得廣大讀者的肯定，只要一年12期 900元的訂費，您就有掌握新知的機會！

此訂閱單請剪下貼在劃撥單上

"電腦新知" 雜誌訂閱單

■訂　　戶：□新訂戶　□續訂戶　電腦編號----------

■訂閱期限：自----年----月起，至----年----月；共----期

■訂 閱 人：姓名　　先生／小姐

電　　話：(公) ---------- (宅) ----------

寄書地址----------□□□□□

■發票抬頭:----------

統一編號:----------

發票地址:----------

■郵寄方式：國內□平寄　□掛號（每期另加20元）　國外□

■年　　齡：□20歲以下　□20～30歲　□30～40歲
□40～50歲　□50～60歲　□60歲以上

■教育程度：□高中(工)　□大專　□研究所以上

■職　　業：□資訊　□軍公教　□電子業　□機電業　□買賣業
□學生　□出版業　□一般服務業　□其他---------

請存款人注意

一、如須限時存款請於存款單上貼足『限時專送』資費郵票。

二、每筆存款至少須在新臺幣十元以上但存款尾數不在此限。

三、本存款單不得附寄其它文件。

● 訂購辦法

地區 / 價格	台灣(平郵)	國外(航空，含郵費)	
		港澳地區	歐美非地區
全年(12期)	NT 900元	NT 1800元	NT 2600元
二年(24期)	NT 1700元	———	———

郵政劃撥儲金存款單

收款人	帳號	帳號未滿八位數者，帳號前空格請填0。
	07923480	
戶名	魏筱玉	

新臺幣：

（請用壹、貳、叁、肆、伍、陸、柒、捌、玖、零等大寫並於數末加一整字）

寄款人	姓名	
	住址	□□□□□
	電話	

局號：　　收據號碼：

主管：　　經辦員：

手續費　次　元

本聯由劃撥中心存查

存款後由郵局製給正式收據為憑，本單不作收據用。帳戶本人存款此聯不必填寫，但請勿撕開。

郵政劃撥儲金存款單

收款人	帳號	
	07923480	
戶名	魏筱玉	

新臺幣：

（請用壹、貳、叁、肆、伍、陸、柒、捌、玖、零等大寫並於數末加一整字）

寄款人	姓名	
	住址	□□□□□
	電話	

主管：　　經辦員：

經辦局號	帳號	日期	存款金額
登帳編號		工作站號	

手續費　次　元

本聯經劃撥中心登帳後寄交帳戶

虛線內備機器印證用請勿填寫。

請注意：一、帳號、戶名及寄款人姓名住址請詳細填明，以免誤寄。二、託付交換票據之存款，務請於交換前一、二天存入，必要時，可請存款局先以電話通知劃撥中心局，唯長途電話費由存款人負擔。如因電話故障等原因無法及時通知者，應由存款人自行負責。

讀者請注意：

由於瑩圃出版甚速且種類繁多，部份書局空間不敷陳列，往往造成讀者選購時的不便，鑑此，"瑩圃"與"下列書局"爲加强服務忠實讀者特立電腦書籍專櫃，只要"瑩圃"目錄上有的書，絕不讓讀者空跑。

假若你在原常購買之書局或電腦書店無法購得你所須之目前瑩圃叢書，可至下列書局購書。而該處又無存貨時，請向該店取得證明再以郵政劃撥

帳號：　07923480　　戶名：魏筱玉　　郵購，本公司免收郵費

店　名	位　　置	電　話
萬得福實業	台北市光華商場地下一樓28號	(02)321-8513
何嘉仁書店(信義店)	台北市瑞安街276巷5-1號	(02)325-6471
何嘉仁實業	台北市瑞安街276巷5-1號1F	(02)325-6471
台北建興	台北市重慶南路一段 45號	(02)361-5890
台北三民	台北市重慶南路一段 61號	(02)361-7511
公館景儷	台北市羅斯福路三段277號	(02)362-3963
公館歐亞	台北市新生南路三段102號	(02)363-6141
台北建宏	台北市重慶南路一段63號	(02)381-8884
立眾企業	台北市中華商場忠段 2F 17-21號	(02)382-1078
公館桂冠	台北市新生南路三段96-4號	(02)391-1407
景佳實業	台北市八德路一段 51號　2F (光華商場)	(02)394-1452
天下電腦	台北市八德路一段 51號 (光華商場)	(02)394-3361
安政電腦	台北市八德路一段 51號 (光華商場)	(02)394-4916
星聚電腦	光華商場地下室68號	(02)394-9011
正揚電腦	台北市八德路1段51號 (光華商場)	(02)397-1461
新學友書局(重慶店)	台北市重慶北路2段235-4號	(02)563-2221
淡友電腦	淡水鎭學府路 51巷　2弄　7號	(02)621-3353
淡友書局	淡水鎭英專路 71號	(02)621-7840
敦南松崗	台北市敦化南路591號　3F	(02)708-2125
新學友書局(民生店)	台北市民生東路757號	(02)712-4747
紀伊國屋書店	台北市忠孝東路四段45號 (太平洋崇光百貨)	(02)721-2304
新學友(寶慶店)	台北市寶慶路32號五樓	(02)755-7271
久大書香	台北市敦化南路385號506室	(02)776-5040
新學友(天母店)	台北市天母東路36號	(02)873-5566
新學友書局(士林店)	台北市士林區文林路281號4F	(02)882-2002
漢鼎電腦	新莊市中正路500號	(02)901-5238
千木科技電腦	新店市北新路二段 27號	(02)913-1966
漢鼎電腦(新店分店)	新店市北新路三段224號	(02)915-0486
中和天才書坊	中和市景新街347號	(02)941-6204
新學友書局(板橋店)	板橋市中山路一段67號	(02)954-0646
大有爲文化事業	板橋市館前東路 47號地下室	(02)957-1958
三重文海	三重市重新路二段 69號	(02)986-9436
諾貝爾書城(中山店)	桃園市中山路105號	(03)334-2932
諾貝爾書局	桃園市中正路 56號	(03)334-8908
桃園文化	桃園市中正路 37號	(03)334-8908
奇奇書局	桃園市中山路156號	(03)338-8039
諾貝爾(中壢一店)	中壢市中正路145號1樓	(03)422-6240
中壢文明	中壢市中和路118號	(03)425-7145
古今集成文化事業	新竹市中正路 16 號	(035)244-627
展書堂文化事業	新竹市中正路 38號	(035)244-683
新竹六藝	新竹市光復路二段460號	(035)724-037
全國書局	苗栗市爲公路 456 號	(037)263-625
正道書局	苗栗市恭敬里華崗路18號 (聯合工專附近)	(037)355-280
金玉堂文化廣場	花蓮市博愛街195號	(038)326-365
新風電腦	花蓮市中山路406之1號	(038)336-007
瓊林圖書專業	花蓮市中華路141號	(038)351-093
信興資訊社	花蓮市日禮路103號	(038)356-478
總領電腦資訊	宜蘭市渭水路23之15號	(039)333-783
大雅資訊廣場	宜蘭市復興路二段73號	(039)353-008
羅東聯合	宜蘭縣羅東鎭和平路 40號	(039)571-936
光南書局	台中市三民路三段 92號 (來來百貨對面)	(04)221-5646
諾貝爾書城	台中市中正路18號 (火車站附近)	(04)222-7163

店　名	位　置	電　話
培基電腦	台中市綠川西街93巷15號 (電子街)	(04)224-5221
新學友書局	台中市太平路28-3號 (來來百貨附近)	(04)225-7388
逢甲書局	台中市逢甲路4號 (逢甲大學附近)	(04)251-1039
揚業圖書	台中市福星路381號	(04)254-0171
三民書局	豐原市三民路21號 (火車站附近)	(04)524-4752
文星書局	台中縣大甲鎮鎮政路24號	(04)687-2525
諾貝爾書城	豐原市中正路34號 (火車站附近)	(045)285-196
墊腳石書店	彰化市中山路二段332號	(047)220-224
建彥書局	彰化市光復路177號 (近彰化客運)	(047)252-792
文明書局	彰化縣員林鎮中山路二段74號 (近員林客運)	(048)344-935
墊腳石書店	南投縣草屯鎮碧山路150號	(049)356-908
超群電腦公司	嘉義市西門街 130 號	(05)225-1447
宏圖書局	嘉義市中山路370號	(05)233-2080
實用電腦資訊	台南市西門路二段433號	(06)220-1122
我的資訊	台南市北門路一段26號	(06)220-5911
台南南台圖書	台南市北門路一段61號	(06)223-6080
船富資訊	台南市北門路一段74號	(06)224-1795
精優電腦資訊	台南市北門路一段117號	(06)228-6660
敦煌書局	台南市中山路163號	(06)229-6347
成大書城	台南市大學路6號	(06)235-4180
金寶書局(推廣部)	台南市青年路232巷10號	(06)235-9031
文山書局	台南市育樂街25號	(06)236-9278
今科電腦	台南市東光路二段28號	(06)237-4525
成大圖書館	台南市大學路1號	(06)237-6362
金典書局	台南市林森路一段192號	(06)268-1049
康特電腦	台南市大學路10號2樓	(06)274-6365
ㄅㄆㄇ資訊	高雄市後昌路614號	(06)363-6361
聖佳資訊	新營市新進路5號	(06)632-3962
長信電腦	高雄市路竹鄉中華路126號	(06)696-6278
優佳電腦科技	台南縣佳里鎮延平路286號	(06)723-2690～1
瓊林書局	高雄市六合二路133號	(07)201-2438
展誠書局	高雄市中山一路215號	(07)201-5658
環球資訊	高雄市七賢一路500號	(07)211-7149
資訊與生活	高雄市中正四路70號	(07)221-2023
光統圖書公司	高雄市建國三路29號	(07)221-6008
復文書局(本店)	高雄市泉州街5號	(07)241-5267
三民書局	高雄市建國三路67號	(07)261-5320
站前大書城	高雄市建國三路 15 號地下樓	(07)271-9461
登文書局	高雄市中山一路276號	(07)282-1336
亞洲資訊	高雄市七賢二路233號	(07)282-3146
天下資訊	高雄市中山一路189號	(07)282-8175
青年書局	高雄市青年一路141號	(07)332-4910
青聯圖書公司	高雄市青年路210號	(07)332-5350
愛智圖書公司	高雄市林森二路69號	(07)334-3200
天祥資訊	高雄市天祥一路142巷8號	(07)342-5436
合成.華興書局	高雄市楠梓區後昌路718巷9號	(07)363-6467
建工書局	高雄市建工路472號	(07)383-7708
復文書局(工專店)	高雄市建工路415號	(07)389-5457
黎明文化公司	高雄市五福四路95號	(07)521-0416
中山大學	高雄市蓮海路70號	(07)531-6171
敦煌書局	高雄市五福四路76,78號	(07)531-8700
華友資訊	高雄市左營區左營大路	(07)581-3575
卓越書局	高雄市左營區實踐路31號	(07)582-2478
上品大書城	高雄市左營區左營大路51號	(07)583-2644
劍橋資訊	岡山鎮柳橋西路102號	(07)621-2280
聯訊電腦中心	高雄縣岡山鎮民族路71號	(07)621-4463
祥翌電腦	高雄市中正三路5號10F之2	(07)662-2903
華資電腦	高雄縣路竹鄉中山路953號	(07)696-4611
龍傳電腦	高雄市大順三路296號	(07)711-8166
進特電腦公司	高雄市四維二路257號	(07)722-0057
復文書局(中山店)	高雄市蓮海路70號	(07)751-9450
光統(文化)	高雄市廣州街150號	(07)761-3227
雄大書局	高雄市武昌街57號	(07)761-7359
清大資訊 五甲店	鳳山市五甲二路230號	(07)822-8835
屏東復文書局	屏東市林森路24-20號	(08)722-2286
屏東書城	屏東市復興北路6號	(08)732-0174
百科書店(屏東)	屏東市中華路27號	(08)732-3447
安特買資訊	屏東市仁愛路172號	(08)736-3437
慈暉電腦行	台東市中華路一段376巷 20號	(089)323-211
北淞汎昱	台東市中正路336號	(089)335-398

讀者來函答覆

曾有不少讀者來函詢問爲何本公司出版之書籍中，有許多雷同的地方。故在此做一說明。

每一學校各科之課程標準及需要之深淺皆不一樣，而教育部在訂定每科之教學標準上亦有所差別。每一個人對每一本書皆有不同的看法，而我們也一向非常尊重各科老師和讀者意見，我們也認爲他們的看法都是正確的。所以爲了配合課程標準及各校各科老師的意見，我們才會出版內容雷同的書籍。

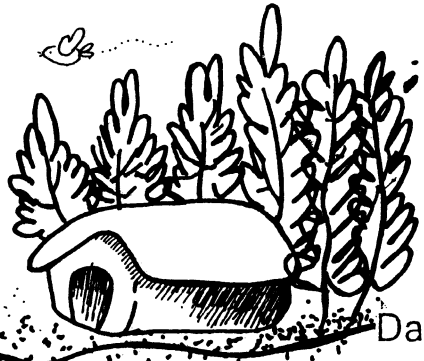
就像一件襯衫，它也許會有各種樣式的設計，然它還是被歸於襯衫類。而我們也會在每本書的序言部份，提醒讀者其相類似的書，給老師及讀者多一個選擇的權利，更避免讀者重覆選到相同的書籍。

在此特別說明，期使讀者能夠了解本公司的用心良苦。

爲提升書本印刷品質，同時瑩圃每本書之銷售量已達到自動機器之最低量，故率先改採最新式之自動機器印刷，過去原用人工貼上去之補書條，也改在書本最後一頁增闢本頁以方便機器作業。故請書局在出售書時，改用刀片或剪刀小心取下補書條。

瑩圃電腦 R&D 敬上

M • E • M • O



BIC 73330

製　　作：瑩圃電腦
發 行 者：周魏筱玉
發 行 所：台北市和平西路一段84號10樓(瑩圃大廈)
信件請寄：台北郵政 6-52 號信箱
郵政劃撥帳號：07923480　　戶名：魏筱玉
FAX　　：(02)365-6584

中華民國八十年八月出版
出版登記證：局版臺業字第四二八九號

● 本書資料曾參考下列書籍 ●

ADVANCED MS-DOS
ASSEMBLY LANGUAGE TECHNIQUES FOR THE IBM PC
DOS POWER USER'S GUIDE
RUNNING PC DOS
SUPERCHARGE MS DOS
MS-DOS BEYOND 640K

M • E • M • O