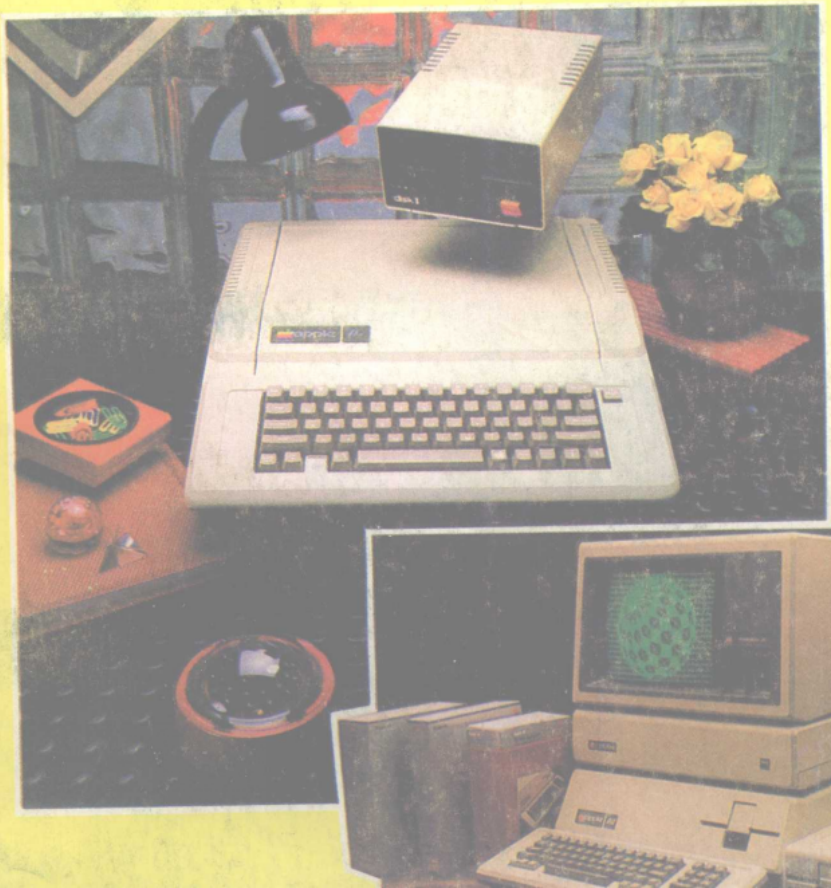
APPLE II

汇编语言程序



APPLE 組合
語言程式

北方电脑公司信息资料部

数据加载失败，请稍后重试！

序

我們常認爲用組合語言來寫程式是種既困難又難以了解而且只有專業的程式設計人員才會的技巧，然而當你對APPLE的Integer Basic或Applesoft Basic之使用已很熟悉時，那麼就沒有任何理由說是你不能利用組合語言來設計程式，因爲組合語言只有一點點不同而已。

我們往往要以舊有的知識爲基礎去學習一項新的學問，各位對培基（BASIC）語言的認識便是學習組合語言的基礎。這本書並不是傳統的方法來介紹組合語言，而是基於一個假設——各位已熟悉了培基語言，在各章節裏，我們將會把組合語言和培基語言的許多相似之處爲大家做一個比較。

爲了能有效的達到此書的目的，在開始我將介紹一些微處理機（Microprocessor）動作的細節，然後再進一步的討論更多的技巧。

這本書並沒有概括所有機器語言程式化（Machine Language Programming）的觀念，然而卻包含了必要的概念和指引以便使一個毫無經驗及認識的人能在最短的時間內了解機器語言（Machine Language），書中的內容及例子是平實而易讀的，不會因爲許多專有名詞而讓大家有壓迫的感覺。

我願在此簡單地陳述一下我的程式哲學。通常我們寫一段

程式去完成某一個工作也就等於去解決一個問題一樣，不管你的問題是什麼，你總會遇到不知如何去寫你的程式的時候，一但如此，你必須繼續下去不可放棄，終究你會得到答案的，這一點非常重要！

在寫程式的過程中我要強調的是要做一個工具的使用者，這裏所謂的工具指的是程式語言中各個指令及其特性，因爲不論用那一種電腦語言來寫程式，不外乎是將其各個指令複雜但有系統的組合在一起，所以你對這些工具不熟悉的話，你的問題將變得更困難。

我很有組織的把組合語言的各個動作及如何利用它來完成簡單的工作介紹給大家，你對其中的內容越熟，那麼程式上的問題對你越是簡單。

在各位研讀這本書的同時，我希望讀者們能用自己的話來解釋各個機器語言的指令及其特性並列成一個表，如此更能加深你的印象，加強你的操作能力。所有指令均表列在附錄 C 中。

讀者可以參考下列關於 6502 程式計劃的各項書籍，作爲本書之補充。

Randy Hyde, Using 6502 Assembly Language (Northridge, CA : Data Most, 1981) ; $19.95.

Don Inman and Kurt Inman, Apple Machine Language (Reston, VA : Reston Publishing, 1981) ; $12.95.

Lance A. Leventhal, 6502 Assembly Language (Berkely: Osborne /McGraw-Hill, 1979) ; $16.99.

Rodnay Zaks, Programming the 6502 (Berkeley : Sybex,

1981）；$ 13.95.

當然可供參考的書籍不只這些，然而在選擇時，各位務必要選擇適合自己讀的參考書籍。

這本書也包含了組合程式（Assembler）的用法（組合程式的作用類似編輯，它是用以產生機器語言程式，如果讀者不太清楚的話，在第二章裏有詳細的說明），我個人較偏愛一種叫Merlin的組合程式，它是Southwestern Data Systems發展出來的，不但是一種很好的組合程式，也包括了許多其他實用的部份，Merlin並非必要，然而有許多例子是用它來與其他的組合程式相比較，如Apple DOS Tool Kit, TED II, the S.C. Assembler 等。

另外一種從Southwestern Data Systems發展出來的一種叫Munch-A-Bug（MAB）的軟體集對我們很有幫助，它可以讓我們很容易的去追踪及偵錯一段程式，MAB也有它自己的迷你型組合程式，在本書的前段將會用到。

關於硬體方面，不論Apple Ⅱ或是Apple Ⅱ PIUS絕對能滿足大家的需求，而不需再添加其他的硬體，在某幾個章節中，我們會討論到磁碟的應用。

在讀者們進入機器語言程式化的主題之前，我必須要提醒各位，天下沒有不勞而獲的事，有許多人聆聽他人演奏了一首鋼琴曲之後讚賞不已，認爲自己也可購置一架鋼琴來練練，興沖沖地花了錢買了回來，彈了幾個鍵之後大失所望，華麗的月光小夜曲並不如想像般地從指縫間滑出，於是再也沒有勇氣繼續下去了，錢花了，東西擺著，多麼可惜！

這種情形在人類的行爲中可常見到，如果你要的是一首月光小夜曲，一卷錄音帶便可滿足你的需求。雖然人們都知道才幹＝99％練習＝99％時間 的道理，可是一但自己不能立刻成爲一個專家時都會感到失望。

想學好任何一樣的技術都必須覆按步就班，一步步來，在許多方面來說，玩 Apple 要比玩鋼琴容易的太多，但是你千萬不要想在頭一個晚上就設計出世界上最偉大的程式來。

爲自己訂定一些簡易可行的目標，例如你能將一個位元組（Byte）的資料從記憶區的某一個位址移到另一個位址嗎？如果能夠的話表示你將可以進入程式化的領域了，事實上，我認爲任何一個人都可以在這個領域當中勝過百分之八十至九十的同道，因爲這些同道們不太願意花一些必要的時間去學習其中的技巧。想要超過 99 ％的人是不太容易，但是 95％卻是非常簡單。

這本書係以循序漸近的方式來編寫的，在讀這本書的同時你也學會寫一些簡單的程式，一些有趣的程式，或是一些要用到磁碟的程式，記住，不要想在唸到第五頁時就想成爲一個專家，然而我保證你在讀完一百頁後，便會驚訝的發現，機器語言程式化是多麼的簡單。

在此我要感謝編輯 Al Tommervik 給予我的大力支持和協助，還有 Greg Voss 提供的保貴意見，及 Eric Goez 一年來在精神上給我的鼓舞。

最後我要對過去幾年裏和我共同分享電腦知識的衆多朋友獻上誠摯的謝意，不論是我的讀者或是使用我們設計的程式的

人或是經由Apple而進入我生活領域的朋友們，他們給予我的鼓勵要勝過金錢報酬對我的誘惑。

很遺憾的，有人認爲電腦將導致人性的逐漸消失，但事實上世界各地區的人類卻經由Apple來交換訊息，增進彼此間的友誼，這一點就能使那些只能表達自己的簡單工具相形見拙。

親愛的讀者們，願您和我共享Apple及程式化爲我們人類所帶來的快樂。

Roger Wanger

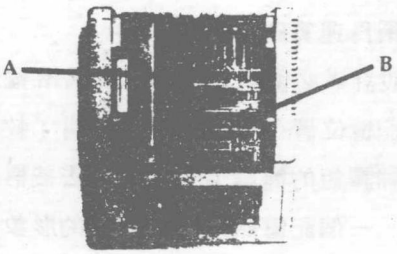
Santee, California

December 1, 1981

目　錄

第1章 Apple的結構

第一個要考慮的是 Apple 自己本身的結構，瞭解這點可幫助我們想像機器的運作是如何進行的。然而我們並不需要徹底瞭解它的結構，讓我們先打開上面的蓋子看看，不要膽小，注視著呈現在你眼前的機器構造，看看它內部是由什麼組合而成。當蓋子打開後，你即可看到如下面照片中所顯示的圖形。



其中我們主要有興趣的是 6502 微處理機，如圖中A所指示的部份；以及記憶體晶片排，如圖中B所指示的區域。如果讀者並非電子行家也無所謂，你可以將它的運作當做是一種神奇的力量。在機器中，記憶體晶片用以儲存數以千計的數值，而 6502 微處理機則用以監督管理其間所發生的動作，其餘的

電子元件祇是用來供應給記憶體和6502 微處理機使用的。這些線路可顯示資料於螢光幕上，允許計算機監督控制鍵盤，因此才使得你能從鍵盤上輸入資料。

螢光幕和鍵盤在計算機中是次要的配件，它們能使你交付工作給計算機，如果不接螢光幕或是鍵盤的話，Apple 自己也能對自己交談的很好，祇是如此的話就失去了和人交談的依據。

6502運作

6502 微處理機是如何工作的呢？系統的心臟部份就是6502 微處理機，這個裝置在某一範圍位址的記憶體內尋找資料而運作。在每一個位置裏，它會找到一些特殊的數值，依據所找到的數值而執行所已知的動作。這個動作可能是做加減某一個數，儲存一個數至某處，或是其它不同的工作，而這些被解釋的數值通常稱爲運算碼（ opcodes ）。

在以往程式設計者必須將每一個運算碼給輸入，一次輸入一個碼至連續的記憶位置中。現在的人發明了較簡單的方法，使用軟體裝置去翻譯短的簡縮字，這種方法被稱爲記憶語法（ mnemonics ）。一個記憶語是依據聲音的形象簡縮而形成的簡縮命令或碼字，例如：STX是儲存字STore X類似的發音而簡縮的。經軟體翻譯後計算機才會理解那一個數值是可用的，而且會監督管理這些數值將其連續的儲存在記憶位置中。這種軟體裝置一般被稱爲組合程式，它使我們更得以自然地與計算機連繫。事實上，培基語言可被視爲極端的組合程式。我們僅使用 PRINT 或 INPUT 來描述所需完成運作的整體集合命

令。

在某些方面，組合語言比培基語言容易使用。組合語言有55個命令需要去學習，而培基語言則需要學習一百多個命令。機器碼的運作速度非常快而且一般都很精簡的使用記憶體的空間就能完成已知的運作。在培基語言中則是速度慢且佔用大量的記憶空間。

記憶位置

在處理 Apple 機器的運作層次中，你所面臨可能最不熟悉的是定址和一般數目的習慣用法。除非你生活在特殊而迷人的環境中，在處理 Apple 的觀點上，你應該絕對中止像下面所示動作的產生：

8BF2- A=03 X=9D Y=00 P=36 S=F2

在某些機器層次的過程中，有時會突然碰到上面所示的狀況而中止了運作，這通常是因爲記憶體被不需要的動作修改了所造成的。

你相信嗎？這種情況下 Apple 實在是想告訴我們有些什麼事情在這裏發生了。究竟 Apple 遭遇到什麼而促使它產生中止的動作，而且在恢復的過程中，它將中止處的位址和計算機的狀態都在終端機上顯示出來，正如前面所示的黑體字一樣。

資料最左邊是較重要的部份，它是動作中止發生時的位址。我們爲何要知道中止時的地址呢？以及你記得自己所擁有的是 16K，32K，或 48K的 Apple 中的那一種？這是以機器所能