APPLE II

微电脑综合应用技术

北方电脑公司资料组

APPLE Ⅱ
微電腦綜合應用技術

汪德溥
游坤燦 編著
陳國球

編 者 序

APPLE Ⅱ微電腦在國內創造了空前的奇蹟式銷售量，這都是拜COPY 之福，姑且不論產品的品質如何及國際輿論如何，她將是引導國人進入資訊時代的主力之一。衆所週知，書籍是傳遞知識的媒介，坊間有關APPLE Ⅱ 英文書籍已不勝枚舉，但中文的專業書籍則乃嫌不足，尤其是週邊卡的應用設計，往往使人摸索費時，不得其門而入。有鑑於此本書作者參考了國外數家著名電腦期刊（BYTE、INTERFACE AGE、MICROCOMPUTING、CREATIVE COMPUTING等）擇其精華，再加以作者自己的創作，這本籌劃經年的書得以定稿。

本書共分四章，包括了硬體、軟體、遊戲設計及市面上已有的一些週邊卡線路圖。其中EPROM模擬器可使你的APPLE向發展系統邁進一大步，它不但適用於APPLE Ⅱ，只要是任何8 bit的微電腦幾乎都可使用。更有EPROM PROGRAMMER(WRITER)的製作及坊間EPROM WRITER 線路分析，使得EPROM模擬器能發揮最大功能。本書以應用爲重，但遊戲程式設計並不是它的重點，一來市面上已有不少書籍涉及此問題，二來玩GAME 畢竟不是我們學微電腦的最大課題。APPLE Ⅱ 週邊卡線路圖給「自己做迷」提供了最佳捷徑，不但有線路，還配合零件位置圖，更告訴了你如何修改BUG 。書中每篇都是精彩之作。本書除了由游坤燦、陳國球、汪德溥三位執筆外還有賴黃東芳老師的熱心指導及儒林圖書公司的鼎力支持才得以出書。筆者才疏學淺，在學業及事業兼顧之下倉促出書，遺及錯誤之處在所難免，尚祈各位先進不吝賜教，並加指正。

編者 謹識

目　錄

第一章　軟體篇

1－1　APPLE Ⅱ的整數BASIC與實數BASIC及組合語言入門…… 1－1

1－2　APPLE Ⅱ中斷信號的使用…… 1－21

1－3　POKE APPLE螢光幕…… 1－36

1－4　如何在APPLE電腦中產生動態畫面…… 1－42

1－5　讓APPLE Ⅱ教你打字…… 1－52

1－6　如何記錄卡帶中的程式…… 1－58

1－7　APPLE應用程式㈠…… 1－61

1－8　APPLE應用程式㈡…… 1－68

1－9　不加任何硬體能使APPLE講話…… 1－80

1－10　白底黑字的螢幕顯示程式…… 1－89

第二章　硬體篇

2－1　爲APPLE設計的A／D轉換器……2－ 1

2－2　EPROM模擬器……2－ 9

2－3　整數BASIC卡……2－18

2－4　爲APPLE做一個週邊界面轉接器……2－25

2－5　APPLE Ⅱ在通信上的應用……2－31

2－6　WONDERFUL EPROM PROGRAMMER……2－37

2－7　EPROM WRITER…… 2－81

第三章　遊戲程式

3－1　俯衝轟炸機……3－1
3－2　太空追逐……3－7

第四章　APPLE][I/O卡線路圖

4－1　INTEGER BASIC CARD線路圖……4－2
4－2　INTEGER BASIC CARD零件位置圖……4－2
4－3　EPROM WRITER線路圖（FOR 2716 ONLY）……4－3
4－4　EPROM WRITER線路圖（FOR 2708、2716、2732、2764）……4－4
4－5　EPROM WRITER零件位置圖（FOR 2708、2716、2732、2764）……4－4
4－6　PRINTER INTERFACE CARD線路圖……4－5
4－7　Z-80 SOFT CARD線路圖……4－7
4－8　Z-80 SOFT CARD零件位置圖……4－8
4－9　APPLE LANGUAGE CARD線路圖……4－9
4－10　APPLE LANGUAGE CARD零件位置圖……4－10
4－11　VIDEX CARD零件位置圖……4－11
4－12　RF MODULATOR線路圖……4－12

第一章
軟體篇

APPLE Ⅱ的整數BASIC與實數BASIC及組合語言入門

【前言】

本文針對APPLE的機型，分析它的優點，使讀者能對整個系統有概括的了解，並充分利用APPLE。同時還介紹組合語言程式的基本寫法，讀完本文後，讀者能夠寫一些簡單的組合語言程式。在本文的最後還有一短短的音樂程式，可使讀者自己嚐試著作曲。

APPLE Ⅱ與APPLE Ⅱ PLUS特性之比較

許多購買者對這兩種機型各自的特性並不十分了解，直覺認爲APPLE Ⅱ PLUS爲APPLE Ⅱ的改良型（目前台灣市場上皆爲APLL Ⅱ PLUS 機型），事實上這兩種型式的分別不在於實質的改進，乃在於使用範圍的不同。以下是一些特性的摘要：

APPLE Ⅱ

- 整數BASIC 語言（ Integer Basic ）
- 迷你組合程式（ MINI ASSEMBLER ）
- 數字範圍爲± 32767
- 僅能使用整數
- 速度快
- 直接進行組合語言編譯
- SWEET 16翻譯器（ Interpreter ）
- 浮點組合語言副程式
- 字串的功能受到限制

APPLE Ⅱ PLUS

- 實數BASIC（APPLESOFT BASIC ）
- 具有自動起始的ROM
- 用十進位浮點方式表示數字
- 數字範圍± 9.999999999E37
- 增進字串的功能
- 擴充程式設計的指令

兩者明顯的不同是各自的BASIC 程式不能互用，若要將適合APPLE Ⅱ的BASIC 程式改寫爲適合APPLE Ⅱ PLUS的程式也不是一件簡單的工作，稍後我們將更深入來討論。

若我們買一個價值台幣千元左右的語言卡（ LANGUAGE CARD）與我們所使用的機型配合，就能具備所有的優點了。一般來說最普遍且實用的軟體是用INTEGER BASIC 寫的，因此若僅是個人的嗜好，且爲了節省語言卡的花費，最好的搭配是APPLE Ⅱ 加上存於RAM 中的APPLESOFT（原始程式可放於磁帶或磁碟）。至於在科技及商業的應用時，需要擴充APPLESOFT的功能，因此APPLE Ⅱ PLUS 爲較佳的選擇。

若我們欲深入了解組合語言（ASSEMBLY LANGUAGE）及其程式設計，就需購買APPLE Ⅱ，因爲APPLE Ⅱ PLUS有自動開始（AUTOSTART）的設計，使得組合語言許多有用的性能被消除。對初學者來說，許多人對組合語言使用的範圍仍然不了解，組合語言程式常是其它程式中的部分程式，可視爲完整的作業系統（OPERAT ING SYSTEMS），在APPLE 微電腦就是用此方式。我們若能充分了解APPLE的性能，這些問題皆會消失，不要限定我們的求知慾。請記住，任何事情只要能了解，都成爲容易的事了。

還有一點，許多電腦銷售店對這兩種機型售價皆相同，因爲在硬體上是沒有分別的，只是軟體不同而已，若再進一步建議，購APPLE Ⅱ PLUS 對一般使用者是較佳的。

整數BASIC卡（Integer BASIC Card）

若我們選擇APPLE Ⅱ PLUS配上整數Basic卡，我們就具備了兩種機型的所有優點，除了具有所有組合語言的功用外，尚有程式師輔助（PROGRAMMER'S AID）ROM 的好處，此ROM 是放在卡上，它的好處如下：

- 行號自動增加
- 磁帶鑑定Tape verify（basic）
- 磁帶鑑定Tape verify(binary)
- 重新組合Relocate（binary）
- Ram 測試程式
- 音樂副程式
- 高解析度繪圖副程式 High Res. Graphics Routines

有了這些特性，我們可做APPLE 能做的任何工作。而且語言系統的互換很簡單，我們就不會當手中有APPLE的程式，卻因語言系統不合而感到頭痛了。

將Integer轉換成 Applesoft

有許多程式嘗試將 INTEGER BASIC 轉換成APPLESOFT，如果是不複雜的程式這工作可能較容易，但是若程式很複雜，這項工作可就艱難了，往往要花費許多時間將指令及程式的語法重新更改，這就要十分熟悉兩者的不同點才會更改了。另一種方法是用 INTEGER BASIC 產生一個程式檔案，再用APPLE SOFT 產生同樣程式的一個檔案。此方式在APPLE的CONTACT #5 中有描述。下面表示它如何的工作（注意@表示CONTROL D）：

```
0  PRINT"@ OPEN X"
1  POKE 33,33
2  PRINT"@ WRITE X"
3  LIST
4  PRINT"@ CLOSE"
5  END
```

此程式亦可寫在同一行上，並插入你所寫的 INTEGER BASIC 程式的任何部份。請別忘記。這樣用法要和DISK Ⅱ 系統連用。一但 RUN 這新程式時，一個名字爲“X”的檔案將被OPEN，同時我們要工作的程式將會寫入此檔案中。在磁碟停止後，按 FP鍵使此檔案轉爲APPLESOFT並執行此檔案，此時我們的程式爲APPLESOFT。固然我們可以寫一程式，使它具有將某種語言翻譯成另一種語言的能力，但我懷疑是否有人會爲了這效果而花許多錢去設計，這或許在

APPLE中是根本不適合的。

若我們有一個用 INTEGER BASIC 寫的程式，想要在APPLE SOFT 語言系統上執行時，眞正有趣的事開始了，因爲你需要找出並更改所有不同的指令和語法，先讓我們看看這些不同點。

- 輸入指令

  IB－ INPUT"APPLE",A 用一逗號

  AS－ INPUT"APPLE";A 用一分號

- 字串的指令

  IB－ PRINT A$(I,I)

  AS－ PRINT MID$(A$,I,I)

  在INTEGER 中，字串指令只有一種，但APPLESOFT中可包含LEFT$,RIGHT$,VAL和STR$ 指令。

- MOD 函數（除法）

  IB－ POKE1, TRY MOD 256

  AS－ POKE1, TRY－INT(TRY／256)＊256

  或是

  IB－Z=X MOD Y

  AS－Z=X－INT(X／Y)＊Y

- IF 指令

  IB－IF X THEN200:GOTO 500

  AS－IF X THEN200:

  GOTO 500

在INTEGER 中，若X爲假（0）此程式讓冒號後的下一指令，在APPLESOFT中，若X爲假則程式跳至下一行執行，冒號後的指令不執行。

IB－IF X#Y THEN500

AS－IF　X<>Y　THEN500

在INTEGER 中用＃號來表示不等式

• 變數名稱

IB－TRY1=TRY2+TRY3

AS－T1=T2+T3

APPLESOFT只把前兩個字元視爲變數

• 亂數函數

IB－X=RND(16)

AS_X=INT(16*RND(1))

在APPLESOFT中，還有一種用來表示亂數變數的方式，X%

AS－X%=16*RND(1)

• 整數　(INTEGERS)

IB－TRY1=TRY2

AS－T1%=T2%

這和前面用亂數INTEGER 所得的結果相同，當然並不需要將變數都化爲INTEGER，只是程式以INTEGER 來寫時，速度較快，也較節省記憶位置。

• DIM 指令

IB－DIM A$(20)

表示有一個 20 字元長的字串

AS－DIM A$(20)

表示有 20 個字串，每字串可達255個字元，在程式中我們若將DIM 指令去掉，此時APPLE SOFT 字串數量若超過 10 個字串時必需用DIM 來標示數量。

• TAB 指令

將所有整數BASIC中的TAB 改成HTAB即可適合APPLE

SOFT用。

• GOTO指令中可計算

IB-GO TO 1000+X*100

AS-ON X GOTO 1100,1200,1300,1400

若有四個可選擇的副程式，我們可由所選擇的數字（X＝1到4）來執行跳躍動作。

• 零頁（PAGE 0）

將整數BASIC所用的機器碼排列在零頁，第三頁中有些部份也可以用來排列。或者，將LOMEM提高超過$800的位址，依序改變所有CALL的對應位置。此時可看出要將INTEGER 程式轉成APPLESOFT可不是說著玩的，可以做到，但我認爲若有一塊INTEGER BASIC卡，就方便多了。

一些APPLE Soft沒有的功能

• AUTO（自動編行號功能）

當你寫程式時可利用此功能使APPLE Ⅱ 自動提供下一個行號。但是如程式有錯誤，或你並沒有輸入任何敍述，在按下RETURN鍵後，APPLE Ⅱ 不提供下一行號。此命令的寫法如下：

AUTO N

N爲起始的行號數，下一行號將是N＋10，

AUTO敍述的另一種寫法如下：

AUTO N,D

你除了必須告訴APPLE Ⅱ 行號的起始數以外，還可指定行號增加的間隔數。N是行號自動增加的起始數目，D則是增加的間隔數。

• MAN （自己定行號）

當你不希望由APPLE II 幫你定行號時，可用CTRL X 取消APPLE II的AUTO 功能，接著打入MAN 指令，以達到取消自動編行號的目的。

• DSP 除錯功能

DSP 敍述後面必須跟著一個變數，當此變數值改變以後，APPLE II將會印出在那一個行號此變數值被改變了，並且還會印出改變後的變數值（這裏所謂的改變，是指重新設定變數值，所以變數值可能與以前的值相同）。由於DSP 敍述會被RUN 令命取消，因此你必須將DSP 敍述放在程式中。DSP 敍述例子如下：

```
>10  DSP  N
```

只要N值重新被設定APPLE II 馬上會印出在那一個行號時變數N的值改變了，接著告訴你目前N值。

• NODSP

你可用NO DSP 敍述取消DSP 的功能，NOSDP敍述的例子如下：

```
.     >100  NODSP
```

組合語言（Assembly Language）

APPLE內部已建立了組合語言的能力，若我們不會用實在是令人惋惜的一件事，許多書籍談到APPLE II內部屬於處理機用的組合語言時，大多假設讀者已具有許多相關的知識，故許多基本而且有用的觀念沒有談到。因此，我們以另一個角度，以一個新學習者，來

接觸組合語言。

背景（Back ground）

若我們要用APPLE Ⅱ 中的監督程式，我們必須知道此兩機型具有與不具有自動起始（Autostart）到底有什麼分別。監督程式包含了許多組合語言程式，其中包含了讀鍵盤輸入的副程式，將指令轉換成電腦函數及在螢光幕上顯示結果的副程式。事實上，我們很容易操作微處理，原因就是監督程式已幫助我們執行了一些程式。可以想像到，當我們在鍵盤上敲一個字時，信號送入監督程式中，然後把對應的字元顯示出來。

下面談到在APPLE各種機型中，如何接觸到監督程式。若接觸到，螢光幕會立刻顯示（＊）記號。

- APPLE Ⅱ — 若無APPLESOFT的ROM 時，接上電源，在任何時刻按RESET鍵皆可。
- APPLE Ⅱ — 具有APPLESOFT的ROM 時；先確定ROM卡是在 Integer Basic 位置，然後按RESET鍵。（可用 Control B 來檢察位置是否正確）。
- APPLE Ⅱ — 具有自動起始作用，開機時，會自動將你放在BASIC語言中，此時要鍵入CALL —151 並RETURN，以進入監督程式。此時ROM 卡上的開關位置與前面相同。
- APPLE Ⅱ PLUS — 具有INTEGER 卡；若沒有INTEGER卡，也能夠進入APPLE的監督程式，但是不能寫組合語言程式，因爲沒有迷你組合程式。CALL —151 將使你進入監督程，由此進入點，可以儲存資料，修改資料及搬動資料。

監督程式的指令（Monitor Commands）

當組合語言的星號＊出現時，我們就需要利用指令使監督程式進一步爲我們工作。

- 在記憶體中讀出或寫入資料
- 修改或搬動資料
- 檢查記憶體內的資料
- 將資料存入卡帶或自卡帶中讀出資料
- 十六進制的運算
- 迷你組合語言編譯器（僅適用於INTEGER 系統）
- 其它的應用

若欲更多了解監督程式，可閱讀APPLE Ⅱ 參考手册（APPLE Ⅱ REFFERENCE MANUAL）40 頁至 66 頁，此手册較深入論及微處理機內部工作情形。

現在回到組合語言上

二進制和十六進制

若我們對二進制、十進制和十六進制間的關係已了解，將更容易討論組合語言。我們常以$0000 到$FFFF來描述記憶的地址。這$符號用以表示十六進制，在十進制中，它們的範圍便從 0 到65536或是說具有65536的記憶體位置。可以不必注意記憶體地址用二進制所表示的值，我們利用記憶體位址找出在此處所存的資料，而這些資料是用二進制來表示的。

在單元記憶體內所存之資料稱爲位元組（Byte），它是由 8 個

位元（Bit）組成，而每一個位元由1或0來表示，1表示此位元在ON 狀態，0表示此位元在OFF 狀態，4個位元稱爲一個Nybble ，表示一個十六進制的數字，二個十六進制的數字表示存在一個記憶體內的二進制資料位元組。當我們自組合語言程式得到較多的技巧，就會更習慣二進制和十六進制的數字。在APPLE參考手册和程式書籍中參閱記憶體分佈圖（Memory map），將幫助你了解如何利用APPLE的記憶體。

組合語言（Assembly Language）

在APPLE Ⅱ 系統中，我們至少有兩種方法可以利用組合語言寫程式。一種方法是寫好了程式後，查表翻譯成機械語言，再利用監督程式將一個個的位元組依序放至記憶體內。另一種方法是利用APPLE中的迷你組合語言編譯程式。在我們試著寫組合語言程式之前，我們尚需了解程式各部份的意義，然後我們會在螢光幕上看見寫好的程式語言如何轉換成機械語言。同時可試驗APPLE的許多特徵。

指令羣（The Instruction）

微處理機以一群密碼來表示它的指令，6502 微處理機使用 55 個指令密碼，這些密碼又稱爲簡字指令（Mnemonics）。僅有這些密碼指令，尚不能夠告訴6502 如何工作，大多數要加上另一種稱爲運算元（operand）的資料才能工作。

```
Mnemonic        Operand
  LDA             #$C1
  JSR             $FDFE
  RTS
```

此例子就是組合語言程式，爲了要利用這小小的程式，需要將它轉換成機械語言。這是組合語言轉譯器的工作，稍後我們將用APPLE II內部的迷你組合語言編譯器寫一個程式。

運算碼（Operation Cade ）

每個指令都有一個以十六進制表示的運算碼。這些運算碼經由系統的監督程式分辨，並轉換二進制數以輸給6502 CPU。事實上，電腦只認識二進制數，若我們測試正在執行程式的記憶體單元時，僅能得著0或1的資料。爲了簡化人與電腦交談的方式，二進制數被轉換成機械語言。較十六進制機械語言高一級的是組合語言，它利用簡字指令（運算碼）和資料（運算元）來簡化程式設計，在參考手册第121頁至第126頁包含了所有6502 簡字型指令的運算碼和地址模式，這些資料在其它參考書籍中亦有。

住址模式（Address Modes）

在指令中使用運算元（Operand）就像在指令中使用運算碼（opcode ）一樣。運算元告知電腦要使用何種住址模式。由此住址模式所描述的運算元，使得電腦在此住址上做一些指定的工作。在6502 系統中有十三種住址模式，這些住址模式的使用時機全看程式設計的型態與結果而定，每一種指令皆可能使用幾種不同的住址模式，待會我們要舉個例子，不過只用了三種住址模式。

微處理機，類似6502 機型，CPU 本身具有讀／寫功用的記憶體（RAM）稱之暫存器（registers）。由於這些暫存器使得程式師能將指令和資料輸出並輸入微處理機中。其中有一暫存器稱爲累積器