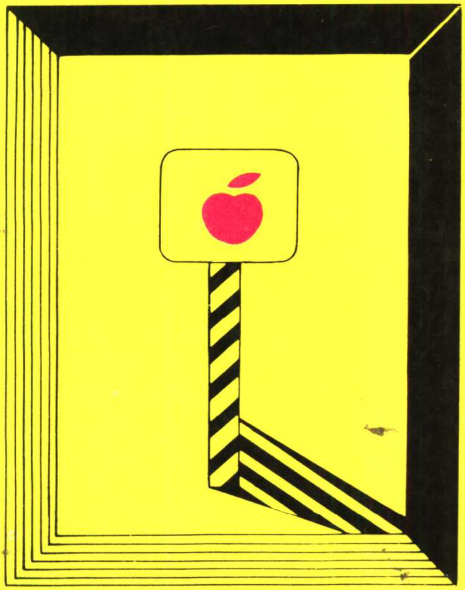
Apple II

Apple FORTRAN

Language Reference Manual

譯者：任清華



Apple FORTRAN 福傳

譯　者：**任清華**
出版者：聯星圖書公司
發行者：聯星圖書公司
地　址：香港洛克道168地下
印刷者：美華印刷廠
地　址：香港柴灣工業大厦二樓128號

前　言

這本手册所描述的，是一種使用於Apple Ⅱ或Apple Ⅱ－plus 上的Apple 福傳語言，它符合美國國家標準福傳子集，也就是ANSI subset FORTRAN 77 。

Apple 福傳包含了ANSI 標準子集以外的一些特性，如包含了完整語言才有的一些特性，及適應Apple 的一些特性。而ANSI 標準子集包含了對完整ANSI FORTRAN66 的許多重要修訂。

這本手册的目的是：

※使你熟悉Apple 福傳與標準FORTRAN 77 的一些差別及擴展

※使你熟悉Apple 福傳的作業環境，它使用Apple Pascal 作業系統。

※向你簡介新的ANSI FORTRAN 77 與舊的ANSI FORTRAN 66 之間的差異。

※提供Apple 福傳的完整語言規格。

完整的Apple 福傳文件包括另一本手册：

※ Apple 語言系統裝置及操作手册
（ Apple Language System Installation and Operation Manual ）

本書中提到的Pascal 文件，包括Apple 語言系統的Pascal 手册（作業系統參考手册及語言參考手册）。

注意：爲裝置你的福傳系統以便開始工作，福傳系統有兩片磁碟，即FORT1: 和FORT2:，前者包含了FORTLIB.CODE；後者包含SYSTEM.COMPILER及SYSTEM.LIBRARY。配置福傳系統的方法寫在附錄A中，第一部份是給使用單一磁碟機的人，第二部份是給使用二台以上磁碟機的人。附錄A假定你所用的啓動磁碟（ boot diskette ）是FORT2:，你也可以作不同的配置，但必須小心處理。另外一提的是如果你沒有用過Pascal 作業系統，附錄A中將提到簡單的使用法。

pple 福傳係基於下列標準：

目　錄

第 1 章
概論

手冊內容說明

什麼是Apple 福傳

Apple 福傳與ANSI77子集

ANSI77 與完整語言

ANSI77 與 ANSI66

簡介

Apple 福傳是一種在 Apple Pascal 作業系統上執行的程式語言。這種強而有力的作業系統可以支援 Pascal 以外之語言。將福傳建在 Pascal 作業系統之上，有許多好處：

※完整的 FORTRAN 程式發展設備（包括文字編輯程式，檔案處理程式，庫存程式，組合語言編譯程式及各種實用程式）。
※易於建立一轉鑰（ trunkey ）系統，亦即開機時可執行指定的程式。
※福傳和 Pascal 都在同一 Apple 上操作。
※使用福傳和 Pascal 只需學習一種作業系統。
※ Pascal 和福傳副程式皆可互相連結。
※組合語言副程式可與福傳或 Pascal 連結。

其他好處包括：Pascal 作業系統提供 Apple 福傳許多理想的特性，使得使用者易於利用 Apple 福傳寫作程式。但將福傳建立在 Pascal 作業系統上有兩個小限制，本章末將有討論。

Apple 福傳和蘋果 FORTRAN 本質上只是編譯程式不同，此外一些小差別，這本手册將有所說明。蘋果 FORTRAN 和 Pascal 的編譯程式輸出碼相同，可使用於單一的作業系統上。

程式發展系統包含編輯程式（ Editor ），連結程式（ Linker ），檔案管理程式（ Filer ），及其他實用和庫存程式。發展程式的步驟如下：

※用編輯程式來寫福傳程式。
※用檔案管理程式讀取或儲存磁碟上之檔。
※用福傳編譯程式將文字檔編譯成碼檔（ code file ）。
※用連結程式把碼檔轉變爲可執行的碼檔。
※執行。

外加步驟包括銜接 Pascal 和福傳程式，或銜接福傳及組合語言程式。

手冊內容說明

下一章討論的是如何在 Pascal 作業系統中用福傳，將告訴

你如何由 Pascal 作業系統的文件中，找出其重要的特色。

手册前段的其餘部份，第三～五章討論如何有效組成福傳程式、福傳編譯程式所需之輸入、及連結程式之使用。第六～十四章包括蘋果福傳語言規格及描述，包含資料形成，陳式、敘述、輸出入、格式等細節。

第十五～十六章討論的是銜接福傳和 Pascal 的程式，調用組合語言副程式，及產生彩色圖形等。

附錄包含這本手册的摘要及一些有用的事情。剛開始使用 Apple Pascal 或福傳的人，於使用前必定要讀附錄 A。

關於福傳和 Pascal 作業系統的資料，包含在 Pascal 作業系統的文件中，所以學習編輯、執行福傳程式，你必須有一份 Apple Pascal 參考手册或 Apple Pascal 作業系統參考手册。

附錄 A 教你如何開始用程式發展系統，以及如何使用單一磁碟機或多磁碟機的系統。

Pascal 作業系統的文件描述了程式發展系統及作業系統本身，其中的例子雖然是 Pascal 程式，但學習作業系統及程式發展系統時，和所用的語言無關。

Pascal 文件其他部份主要針對 Pascal 使用者，所以一些部份我們於這本書中重新改寫，其他部份只須以福傳代替 Pascal 即可通用。

什麼是Apple 福傳

福傳是一個歷史很久的高階語言，它經歷了多次的發展。1966 年美國國家標準協會（ANSI）提出一種標準福傳，被稱爲 ANSI FORTRAH 66。這語言又經多次增訂，於 1977 年又製訂了一種新的標準，被稱爲 ANSI FORTRAN 77，目前已被廣泛接受。Apple 福傳則是基於 ANSI FORTRAN 77 的子集，並加入 Apple 電腦的一些特色。

所以，使用者應該了解 Apple 福傳與其他福傳的是同點，在此提出三個問題：

※ Apple 福傳與 ANSI FORTRAN 77 標準子集有何不同？

※ ANSI 子集和整個語言有何不同。

※ANSI 77 和ANSI 66有何不同？

以下將逐步討論。

Apple福傳與ANSI77子集：

它們大致相同，只有少許差異。

Apple福傳不符合ANSI 77子集的地方有二：

※Apple福傳中整數與實數所佔記憶量不同，而ANSI子集相同。實數佔4 bytes，數值由±5.8E−39至±1.7E+38；整數佔2 bytes，數值由−32768至+32767；邏輯值則佔2 bytes。

※子程式名稱不能經由另一子程式作爲正式參數。

另一些地方Apple福傳允許一些完整的ANSI FORTRAN之特性，而不只是子集：

※註標陳式：允許調用函數，陣列。

※DO變數陳式：DO的範圍不限制陳式的使用，可使用所有的整數陳式。此外，READ、WRITE中隱示的DO迴圈也適用這個特性。

※輸出入單位號碼：可使用陳式。

※輸出列可使用陳式。但左括號不能作爲開頭，故（A+B）※（C+D）須寫爲+（A+B）※（C+D）。

※計值GOTO中的值可使用陳式。

※一般化的輸出入：Apple福傳對循序檔與直接檔皆可使用有格式規定或無格式規定兩種方式。原有子集規定直接檔無格式規定，而循序檔有格式規定。OPEN敍述可接受完整語言中的附加參數。可用CLOSE敍述。輸出入細節於第十一章有詳細描述。

※包含內部函數CHAR。

此外，Apple福傳包含另一些特性，如控制編譯程式的指令，以及EOF函數等。

ANSI77與完整語言

爲澄清ANSI標準子集的特色，這本書有兩個附錄摘要這個子集：附錄D爲子集及Apple福傳的語法結構圖；附錄E列出子集所有的敍述及語法結構。

ANSI77 與 ANSI66

二者的差別如ANSI 77 廢除荷氏（Hollerith）資料形式，可參考附錄F。此外新的能力加入於ANSI 77中，並澄清了ANSI 66中的未定義區域。

第 2 章
FORTRAN讀者指南

閱讀方向
Pascal 文件指南
指令 (COMMAND) 階層
檔案管理程式 (Filer)
編輯程式 (Editor)
6502組合語言編譯程式 (Assembler)
連結程式 (Linker)
實用程式

閱讀方向

正如前面所提到的，這本手册必須參照Pascal 手册，才能對Apple福傳有完整的認識。這一章的目的，是在告訴你這些手册中應閱讀的部份，並幫你將這些手册用於福傳。

在Pascal 文件中，對編輯程式，檔案管理程式，連結程式及作業系統中的許多現點，均有完整的描述。其中有許多Pascal程式範例，大致上我們只須改變磁碟片名稱，並以“ FORTRAN ”代替“ Pascal ”即可。

在Pascal 作業系統上，Pascal 與福傳的關係簡述如下：

※當 Pascal文件提及作業系統檔案名稱，如SYSTEM.A APPLE，在福傳與Pascal 中的功能相同。主要的例外是SYSTEM.COMPILERY現在是福傳編譯程式；以及SYSTEM LIBRARY 中包含有特殊的單位：RTUNIT.CODE，此外，福傳中沒有可和 Pascal的SYSTEM.S SYNTAX相對等的程式。

※Pascal文件中提到Pascal虛擬碼，稱爲Pcode 福傳和Pascal編譯程式都產生這種碼，而不是產生機器碼，Pcode有它自己的解釋程式，可將它以機器碼的形式執行，所以福傳和Pascal都可在Pascal作業系統上執行。

※磁碟檔案有兩種基本形式：TEXT和CODE。TEXY（文字）檔是人可閱讀的格式；而CODE（碼）檔是機器閱讀的。TEXT檔是一串字元，英文或福傳程式；CODE檔則是福傳或Pascal 編譯程式所產生的。前者具有.CODE字尾，後者具有.CODE字尾。編輯程式只編輯具.TEXT字尾的檔案，而連結程式只連結具有.CODE或.LIBRARY 字尾的檔案。使用者須小心處理這些事項。

Pascal 文件指南

以下是如何閱讀Pascal 作業系統文件的指南。讀者可先閱讀這些手册的簡介部份，以便對整個作業系統有概觀的了解。

指令 (COMMAND) 階層

請讀完關於指令階層的這一章，以下補充有於建立轉鑰（

turnkey）系統的方法。

Apple Pascal 系統可建立轉鑰系統，以便於開機時立即執行某一指定程式。爲建立轉鑰系統，首先將磁碟FORT2：複製，但其中不需要SYSTEM.COMPIELR，再用C（hange指令將複製出的磁碟改名，譬如說改爲TURNKEY ：。然後用T（ransfer 將開機時所要執行的碼檔（code file）放入這一磁碟，這個檔的名稱，必須爲SYSTEM.STARTUP。

檔案管理程式 (Filer)

請閱讀這一章，並記住檔案管理程式位於磁碟FORT1：之上，所以使用前請放入這一片磁碟。

編輯程式 (Editor)

請閱讀這一章，並記住編輯程式在磁碟FORT1：之上。

在“改變文字內容的指令”這段，於I（nsert插入方式時，平常爲A（uto-indent TRUE，F(illing FALSE。

6502組合語言編譯程式 (Assembler)

福傳程式可調用組合語言副程式，這章告訴你如何寫它與如何連結。將福傳而言，程式ASMDEMO必須作少許修改，本書第第十五章將有說明。

連結程式 (Linker)

本書第五章取代了Pascal 書中的這章。

連結程式將編譯過的一些碼檔連結成可執行的碼檔。雖然福傳和 Pascal 所產生的碼檔格式相同，且連結程式也相同，但其用法卻不同。請參照本書第五章。

實用程式

關於福傳和 Pascal 磁碟格式均相同，並使用一個格式化程式（Formatter），這程式係於於 Pascal 的磁碟APPLE3:之上（可參考附錄A）。

關於系統的庫存程式，主要庫存是SYSTEM.LIBRARY，福傳與 Pascal 均使用它，雖然二者所使用的有少許不同，譬如福傳庫存中有RTUNIT。另外，福傳使用的庫存管理程式爲FORT1：上的FORTLIB.CODE，而不用 Pascal 的LIBRARY.CODE。

爲管理庫存程式，請閱關於庫存內容圖（ Library mapping ）的部份。

假如不使用標準的操作台（ console ，包括鍵盤和螢幕），則必需將操作系統一部份重新配置（ recofigure ），可參考Pascal作業系統中系統重新配置與改變GOTOXY 通訊的部份。

第 3 章
程式片段處理

簡介
部份編譯
原始碼片段
目的碼片段
單元、節和庫存程式

簡介

福傳系統提供許多完成程式的方法，其中最重要的就是可將一個大程式分爲許多子程式，這些子程式可獨自發展，並可供許多程式調用。將程式分爲許多片段的單元，將有下列許多好處：

※將大的工作分割爲片段，並獨立發展，再加綜合，因爲每次只專注於一項問題，發展速度將會更快。

※子程式可被許多程式調用，卻只須寫一遍，並可儲存於庫存內備用。

※許多實用的子程式已包含於系統庫存中，你的程式可直接使用。系統庫存位於磁碟FORT2：

※福傳主程式可和Pascal或組合語言程式連結。

※作業系統提供了許多處理子程式的方法，可減小程式所佔空間，這些方法本章將會提到。

部份編譯

福傳程式包含了一個主程式及一些子程式。在最簡單的情況，它們存在於同一個文字（TEXT）檔，稱爲原始碼。它們同時編譯成碼（CODE）檔，稱爲目的碼，執行時一次放入記憶後再開始。放入記憶的程式稱爲碼像（code image）。

執行程式時，將程式分斷爲單元，有三個階層：原始碼階層，目的碼階層，碼像（code image）階層。每一階層各有其用途及優點，如下：

原始碼片段

可將原始程式任何部份分離於不同的文字（TEXT）檔中，只須加入一些特殊敍述以取代所移出的部份，編譯程式即可處理妥當。但唯一限制是必須於每行的邊界分割（不可把敍述分割）。編譯程式將它們重新組合後，仍是一個完整的福傳程式。

將程式分割成單元（module）有兩個目的：第一，不必每次都編輯全部程式；第二，把不同目標的程式放於不同的檔中，可使程式結構與組織更清晰。各項細節可參考第四章的$INCLUDE敍述。

目的碼片段

若文字檔案是一個完整的子程式，則可個別編譯，此種過程稱爲分離編譯。此外，若有需要亦可再分爲不同的文字檔案。

如果有需要，一個完整的子程式仍可分割成許多文字檔，但不論如何分割，仍須提供編譯程式一個完整子程式。

在最簡單的情形下，每個文字檔包含了一個或以上的子程式，分離編譯分爲兩個步驟。首先，子程式個別地被編譯，而產生CODE檔，其中不但包含編譯過的程式，也包含描述這個子程式的一些資料。

第二步驟是編譯主程式，PROGRAM敘述告訴編譯程式，這是個主程式，接著必須用$USES指出子程式所在的檔案，$USES必須緊接在PROGRAM後，任何可執行敘述之前。於是，編譯程式將讀進子程式中所用的名稱和其他的描述，此時編譯程式才能編譯那些牽涉到子程式的部份，$USES指令的詳情請看第四章及第十四章。

假如編譯程式沒有發現任何$USES或$EXT敘述，編譯程式將認爲這一個文字檔已包含了完整的程式。

福傳編譯程式並不眞實地把子程式的碼放入主程式剛編譯過的碼檔中，它只產生一些訊息，以提供連絡程式將主程式與子程式連結。

以上只是最簡單的情形，我們可以利用這種結構來做許多事情，包括分離編譯的子程式調用其他分離編譯的子程式。

當我們把主程式與子程式連結成爲一個可執行的碼檔時，你必須告訴連結程式全部所需檔的名稱。

與福傳的$USES相對的是Pascal的USES敘述，它們可以把Pascal和福傳程式相連結。至於連結程式的使用法，請看第五章。

單元、節和庫存程式

一群包含有一個或以上的子程式，一起編譯時，稱爲一個編譯單元（compilation unit），但不可和輸出入單元（I/O unit）相混淆。輸出入單元是一個特定的輸出入裝置。

使用編譯單元有兩種主要方法：正規單元或覆蓋單元（

OVERLAY unit ）。當正規單元與主程式連結時，單元中的子程式將放入所產生的碼檔中，所以這個碼檔，不但包含了主程式，也包含了子程式。

覆蓋單元的目的是把程式分割，以避免過大。覆蓋單元平常並不放在記憶器裏，而只是當需要用到它時，才由SYSTEM、LIBRARY或其他碼檔中取出，放入記憶器，使用完畢後，這段記憶器空門又可以作其他使用。覆蓋單元必須預先加以分離編輯，使用時，只須在$USES敍述中加上OVERLAY就可以了，請參照第十四章。

分離編譯可以讓許多主程式共用一個子程式，通常這些子程式都放在一個庫存碼檔FORTLIB·CODE之中，位於FORT1：之上，它們的詳情請看第十四章。此外，你的系統中還有一個庫存，稱爲SYSTEM·LIBRARY，它包含了許多單元，包括一個許多福傳程式執行時都需要的單元，叫做RTUNIT，除了RTUNIT之外，這個庫存中所包含的子程式，可以作彩色圖形等。還有，你也可以把你自己的單元放入庫存書。