微處理機
軟體設計

林國富 譯

Software Design for Microprocessors

John G. Wester

雲陽出版社

微處理機軟體設計

作　者　林國富

出版者　雲陽出版社

台北市光復南路317巷4弄
台北郵政信箱第3-69
7629705
郵撥 18247
登記證局版台業字第0908號

發行人　陳文惠

台北市光復南路317巷46號
7629705

印刷者　建利印刷廠
台北市東園街260巷25號
3711088

基　價　精裝伍圓　平裝肆圓

版　次　中華民國69年 5 月初版

譯者序

計算機（*computer*）與計算器（*caculator* ）最大的區別就是計算機有記憶能力，一旦有記憶能力就能夠思考，所以計算機又有人譯爲「電腦」意義就在這裡。雖然計算機有記憶能力及思考能力，但是這些能力仍然是人們賦予機器的；早期的計算機由於軟體發展不足“較笨”只擁有計算能力，隨著硬體技術的演進軟體知識與技術也越來越豐富，計算機也變得越來越“聰明”，除了能夠計算以外還能做控制以及其他更複雜的工作，而且在配合軟體工程師的智慧運用各種程式語言及作業系統，計算機就名符其實可稱爲「電腦」。

本書討論的範圍是微計算機的軟體設計，其目的是協助使讀者所擁有的微計算機系統能更聰明、更完美。本書前四章是基本軟體設計觀念的說明，第五章是實際研討微處理機的軟體設計。大量的圖表及附錄提供讀者設計所需的資料是本書的一大特色，相信充份利用這些資料對於您的設計工作會有莫大的幫助。在此並感謝摯友黃漢邦、侯瑞甫的協助校稿以及雲陽出版社在各方面的配合。本書文稿雖一再校對，惟恐力有不迨之處仍多，盼各方賢達多多賜教。

林國富於台北
六十八年十一月

原序

在寫本書的同時，"微處理機"（*Microprocessor*）一詞已變成電子工業中最新的熱門字眼。數位積體電路的製造商，在製造改進大型積體電路（*LSI*）時，不斷地增加單位矽晶片上電閘（*gat*）的數目，增加的速率幾乎是與時間成對數關係。*LSI* 的首次主要的衝擊是在記憶器方面。當半導體記憶器變得足以與磁蕊記憶器相競爭，且由於使用日益廣泛，價格也就不斷下跌，這種新的數位技術的使用就出現了。由於 *LSI* 的改進（特別是 *MOS* 技術）使得計算機的算術和邏輯函數甚至控制函數可以製造在單一的 *LSI* 基片（*chip*）上（現在通常稱爲微處理機）。將計算機的這兩大基本的單位結合後，再加上半導體記憶器後，以 *LSI* 製造計算機的技術就實現了。由於市場的擴展以及越來越多半導體製造商不斷地加入製造各種微處理機，價格也隨著對傳統半導體的學習曲線的下降而下跌，使得微處理機以及 *LSI* 型計算機變得更經濟可行。預料微處理機的應用將會越來越廣。

微處理機和 *LSI* 型計算機的應用場合，其範圍由程序控制，到自動控制，到資料處理，甚至用於競賽、遊戲。微處理機以及其完整產品的市場在未來這個新時代將放射出不可估計的曙光。爲了瞭解這些，我們只需記住計算機及其同類的產品也會在這個時代大放光芒。

由於微處理機的用途如此廣泛，且由於其應用如此像計算機，因此必定會改變部份設計者的程序以利用所有這些新的計算元件的優點。設計工作的主要部份將是如何產生程式（軟體）。同一組微處理機和記憶裝置可用於很多種應用中，只需改變所要執行單位的

程式或指令序列即可。

本書的目的是協助技術或非技術人員，踏出以微處理機及其有關軟體來設計產品的第一步。本書討論的主題由基本的二進數一直到複雜的微處理機應用的實例。雖然本書主要是以很少或者沒有程式設計經驗的對象來寫的，但甚至老練的程式設計師也會發現其中的應用範例是很有挑戰性及教育性的。我希望讀者們能接受這些挑戰來學習更多有關微處理機及其軟體的設計，保證最後將會因加入這一新的、刺激的時代而獲得許多好處。

John . G . Wester
Willian D . Simpson

導言

自古以來，人類就想盡辦法來簡化日常的工作。的確，在整個發明界可發現最初的動機是人們想提昇有關日常呆板、費時、無效率、無聊而令人厭膩的工作。爲了協助人們計算，人們發明樹枝、石頭以及念珠等工具的計算系統。大約回溯至西元前 450 年，開始用算盤以木枝上的念珠來代表數字，一直到今日在世界的許多地方還在使用此種工具。

第一種眞正的計算機械是由巴斯卡（*Pascal*）設計在1642 年製造成的。完成計算的單位是以十爲底使用的是一些齒輪系統。在大約 1671 年李柏尼玆（*Leibniz*）發明了一種更精確的機械包括了做乘法和除法的能力。這種早期的機械的一個更突出的特點是使用了二進算數，也就是以二爲底的數字系統（參考附錄 *A*）。

近代（從 1800年），歷史學家記述發明家巴伯傑（*Charles Babbage*）不僅完成了機械計算裝置的發展，而且指出這時才首次可使用"計算機"（*computer*）一詞。事實上他的分析機（*analysical engine*）是現代數位計算機的先驅。

現代計算機械的另一次重大進步，是1944年在哈佛大學(*Harvard University*)的一個小組發展成功的 *Mark I*。這部機器有一個房子大，是電機序列控制（*sequence-controlled*）的計算機器。它的結構和巴伯傑的分析機相似是使用二進算術，且包含有相同的基本計算機械元件，但是它另外有更好的輸入 / 輸出能力。

在 *Mark I* 之後不久就發明了 *ENIAC*。在這個系統裡前一種機器的電機邏輯元件以眞空管來取代，這大大的增加了計算的速率。而且，在大約這同時，1940年的年末，另外在數位系統的發展

上又有新的突破。紐曼（*Jon Von Neumann*）首先發展出預儲程式（*stored program*）的觀念。接著貝爾實驗室（*Bell Labs*）又在1948發展成功電晶體。紐曼的預儲程式極爲重要，解除了早期將計算機器由一種程式改變爲另一種時的困難。

爲了分別計算機的實質元件（硬體），和預儲程式或記憶器的指令內容，於是發明了軟體（*software*）一詞，軟體代表預儲程式的資料，儲存在記憶器中的零和壹指示機器依其運算序列操作。

預儲程式的基本觀念再更深入說明後，其重要性就可完全了解。儲存的程式是一列零和壹的形式，就像用以儲存資料的二進數字，其意義是機器的步驟是按特殊程式來執行，可以很容易地對記憶器載入新的指令而加以改變。事實上，記憶器用來儲存指令和資料字時並無任何差別。因此，計算機的控制單位，是以直接由記憶器中讀出一序列指令來執行其功能或"程式"（*program*）。這最顯著的益處是快速且可完全自動的做計算機控制。

預儲程式的觀念促成了大型記憶器的發展，因爲現在顯然需要較大的單一功用的記憶器來擔任儲存資料和機器的指令。第一個使用預儲程式的機器之一是 *EDVAC*。這架機器和先前的相似，包含四個計算機的基本單位（記憶器、算術單位、控制單位、和輸入/輸出）且使用二進算術。*EDVAC* 這一類的機器，使此工業產生了一種特徵，以硬體和軟體來說明所完成系統的能力。

與機器中製造技術的進步相比較，半導體技術的重要性不能誇大地單以其對計算機和計算機器的衝擊來說。由巴伯傑的分析機中的發明到第一架電子數位計算機發展成功（1946）大約花了一百年。仔細追研這件事是1948年電晶體的發明，那是電子技術的重大成功。電晶體的經濟使得適合用來製造大型數位機器。但是，從電晶體的發明到第一個積體電路（*integrated circuit*）發展成功只經過短短十年。1958年不但是積體電路成功的一年，也是第一架以電晶體製造的商業計算機問世的一年。換句話說，將電晶體技術完全用在計算機中足足費了十年。在1960年中期，不到五年的時間

，計算機就開始使用積體電路。表 1-1 列出了促使數位硬體演進的主要的電子發展的里程碑。

表 1　電子與計算機發展的里程碑

1883	半導體二極體——硒整流
1904	眞空管二極管
1906	眞空管三極管——電子放大
1946	*ENIAC*——第一架電子計算機
1948	*EDVAC*——第一架預儲程式計算機
1948	電晶體——半導體放大
1951	*Univac*——第一架商用的計算機
1958	電晶體計算機
1958	積體電路
1964	積體電路計算機
1964	大型積體電路
1965	迷你計算機
1970	*LSI* 半導體記憶器
1971	微處理機

當系統首先使用積體電路技術時，也正在發展其他的技術，即在製造以前爲了要完全測試或除錯而先模擬。這種設計上的困難是昂貴及費時。但是，當半導體技術進步且較高密度的積體電路成功時，電路元件變成不昂貴而數位計算機就爆炸性地出現。積體電路的密度進步得更高時，就有了大型積體電路或 *LSI*。*LSI* 的發展結合了預儲程式的觀念開創了計算機的新紀元，開啓了微處理機和微計算機的時代。

微處理機的時代其特徵是數位技術深入現代生活的各面，且這只是個開始。*LSI* 技術（最明顯的是微處理機）的經濟跳板，將提供越來越便宜的數位硬體，而產生一些爆炸性的應用。爲了這些應用的出現，許多經營者、設計者、學生以及使用者，要學習更多有關高密度的電路技術，和如何經由軟體來使最後的元件可以完成許多事情的觀念。經由儲存在現代數位機器中的程式，機器變得好

像有生命似的來完成所要做的事。經由了解這些程式是如何運算，以及如何產生它們，人們將參與此種刺激的演進。

本書所說的＂如何＂是以＂做中學習＂的方式來學習。第一章說明一些名詞的定義，基本機器結構的研究，以及基本的機器指令。下兩章是說明寫程式的技術以及後勤工作（*logistics*），以及發展軟體所需的支持。第四章經由簡單的計算機器的發明談到其在通訊問題中的應用。在第五章中說明四種典型的微處理機的應用，並比較不同種類微處理機的結構能力與限制。範圍廣泛的附錄包含了一些基本材料以及有用的參考資料。

在本書中的軟體說明僅限制在機器階（*machine - level*）和組合階（*assembly - level*）的程式設計。較高階語言的程式設計留至這方面的主要課程中討論。

目　　錄

第二章 如何建立軟體

第三章 軟體的支援與體實

第一章
基本觀念

1—1 概說

我們可以把軟體的世界想成一張地圖。首先，我們必須了解如何讀畫在圖上的符號。這就必須學習語言和特殊的術語。其次，我們必須了解由圖上的一點到另一點時所表示的是什麼。我們必須了解如何由許多可用的路中採取一條最好的路來達到所要求的功能。最後，我們必定可由基本的圖達到熟練並完全了解一些特殊圖中的意義，例如所描繪的地形。因此，由學習語言和特殊術語以及由研究基本程式設計的技術，我們就可獲得整個軟體中更複雜的東西。

1—2 定義

首先我們先定義一些名詞：

數元（*bit*）　"二進數位"（*binary digit*）的簡稱，用來代表二進數中的一個位置（*position*）。如同二進數元，它只能是"0"或"1"（

也是資訊的基本單位）。

字
（*word*）　一列數元，通常用來代表計算機中 *ALU*、暫存器、記憶器、或資料路徑中所能處理的並行數元數目。

數元組
（*byte*）　一列數元，通常是計算機中能夠操作的最小的數元數目（例如：16 數元字可包含兩個 8 數元數元組。）

字元
（*character*）　用來代表資料中的一個字母、數位、或符號。在計算機工作時通常是使用 8 數元字元碼（八個數元可代表 $2^8 = 256$ 個字元或者一個校正數元和 $2^7 = 128$ 字元）。

在定義上面的第一個名詞時，曾用到＂1＂和＂0＂這兩個名詞。這些名詞本身是什麼意思呢？因爲我們所指定的二進數元＂1＂和＂0＂實際上的意思是指邏輯的＂導通＂或＂不導通＂兩種狀態，所以我們是採用較容易製作成電路的兩位階組織方式，而不用像十進制計算法所需十位階組織方式的電路。當指定電子電路中的二進準位時，首先要接受兩個定義：

正邏輯　＂1＂＝電壓高準位
　　＂0＂＝電壓低準位

負邏輯　＂1＂＝電壓低準位
　　＂0＂＝電壓高準位

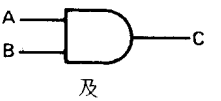
較常使用的是正邏輯表示法。

二進算術的這個世界聯想到有用的工作究竟是怎麼回事呢？那是經由**布氏代數**（*Boolean algebra*）做媒介。布氏代數是以零和壹來組成函數的規則。圖 1-1 中所顯示的是一些布氏函數的代表＂**及**＂（*AND*）、＂**或**＂（*OR*）、**取補數**函數，並以眞值表來說

明它們的輸入／輸出關係。一些更完整的定義說明於附錄B。當這些函數以電子電路（硬體）來製作且連用組成複雜問題的整個解時，這種電路的整個組合稱爲"**硬線**"（*hardwired*）解。如果電路製作成具有以控制字來影響其功能的改變能力，我們就有了硬體的軟體控制。利用一列這些控制字，我們就可用同樣的電路以軟體的修改一步步完成整個複雜的函數。這就產生了"**程式**"（*program*）。

1—3　機器結構

在了解如何建立軟體程式時，重要的是要有硬體的基本知識。它們最基本的部份是什麼呢？它們的特性是什麼呢？它們是如何工作的呢？它們整個系統的特性如何呢？這些可由研究機器結構而明

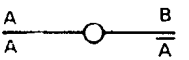


及

及函數		
A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

或

或函數		
A	B	C
0	0	0
0	1	1
1	0	1
1	1	1



取補數

取補數函數	
A	B
0	1
1	0

圖 1-1　基本布氏觀念