



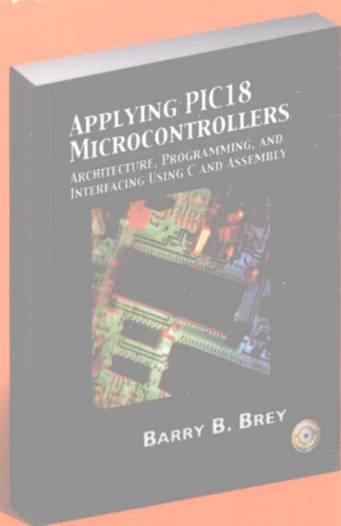
国外经典教材·电子信息

PEARSON  
Prentice  
Hall

Applying PIC18 Microcontrollers  
Architecture, Programming,  
and Interfacing using C and Assembly

# PIC18微控制器

体系结构、编程与接口设计(C语言与汇编语言版)



Barry B. Brey 著  
王俊华 肖亮 等译



清华大学出版社

国外经典教材·电子信息

# PIC18 微控制器

## 体系结构、编程与接口设计

### (C 语言与汇编语言版)

Barry B. Brey 著  
王俊华 肖亮 等译

清华大学出版社

北京

Simplified Chinese edition copyright © 2009 by PEARSON EDUCATION ASIA LIMITED and TSINGHUA UNIVERSITY PRESS.

Original English language title from Proprietor's edition of the Work.

Original English language title: Applying PIC18 Microcontrollers: Architecture, Programming, and Interfacing using C and Assembly, by Barry B. Brey, Copyright© 2009

EISBN: 0-13-088546-0

All Rights Reserved.

Published by arrangement with the original publisher, Pearson Education, Inc., publishing as Prentice-Hall.

This edition is authorized for sale only in the People's Republic of China (excluding the Special Administrative Region of Hong Kong and Macao).

本书中文简体翻译版由 Pearson Education(培生教育出版集团)授权给清华大学出版社在中国境内(不包括中国香港、澳门特别行政区)出版发行。

北京市版权局著作权合同登记号 图字: 01-2007-5191 号

本书封面贴有 Pearson Education(培生教育出版集团)激光防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话: 010-62782989 13701121933

#### 图书在版编目(CIP)数据

PIC18 微控制器: 体系结构、编程与接口设计(C语言与汇编语言版)/(美)贝里(Brey, B. B.)著;王俊华等译. —北京: 清华大学出版社, 2009. 4

书名原文: Applying PIC18 Microcontrollers: Architecture, Programming, and Interfacing using C and Assembly

(国外经典教材·电子信息)

ISBN 978-7-302-19335-7

I. P… II. ①贝… ②王… III. 单片微型计算机—微控制器—程序设计—教材  
IV. TP368.1

中国版本图书馆 CIP 数据核字(2009)第 010408 号

责任编辑: 龙啟铭

责任校对: 徐俊伟

责任印制: 何 芊

出版发行: 清华大学出版社

<http://www.tup.com.cn>

社 总 机: 010-62770175

投稿与读者服务: 010-62772015, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

地 址: 北京清华大学学研大厦 A 座

邮 编: 100084

邮 购: 010-62786544

印 刷 者: 三河市春园印刷有限公司

装 订 者: 三河市李旗庄少明装订厂

经 销: 全国新华书店

开 本: 185×260

印 张: 35.5

字 数: 857 千字

版 次: 2009 年 4 月第 1 版

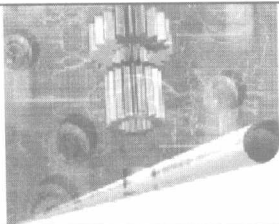
印 次: 2009 年 4 月第 1 次印刷

印 数: 1~3000

定 价: 69.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题,请与清华大学出版社出版部联系调换。

联系电话: 010-62770177 转 3103 产品编号: 027028-01



## 译者序

本书以 Microchip 公司的 PIC18 系列微控制器为例,讲解微控制器的结构、编程和接口技术,内容循序渐进、由浅入深、全面丰富、实例众多,既有适合初学者的入门知识,又有复杂的经典案例。另外,每章都有习题和简明扼要的总结,便于读者复习和检验学习效果。

本书主要内容如下:

(1) 第 1 章介绍计算机结构的基础知识,适合初学者入门。

(2) 第 2 章针对 Microchip 公司的 PIC18 系列微控制器,讲解 PIC 系列微控制器的结构、集成开发环境,以及汇编和连接过程。在对微控制器编程或接口之前,理解微控制器的结构是至关重要的。

(3) 第 3 章是一个重点章节,讲解 PIC18 微控制器的指令集中各指令的操作。

(4) 第 4 章提供了更多使用汇编语言编写的程序和算法。该章内容十分有价值,这里讲到的方法有助于加深读者对汇编器现有知识的理解。

(5) 第 5 章介绍如何使用 C 语言对 PIC 微控制器进行编程。

(6) 第 6 章详细讲解 PIC18 系列微控制器的 DC 和 AC 特性、操作 PIC18 微控制器所需的时钟电路,以及有关 PIC18 微控制器内部众多 I/O 设备操作的要点。

(7) 第 7 章说明如何接口和使用 PIC18 微控制器系统中的许多基本 I/O 设备,提供了基于开关的设备和显示设备与微控制器接口所需的软件和硬件信息,以及控制这些设备的软件基础。

(8) 第 8 章讲解中断的各种应用。

(9) 第 9 章展示过程控制系统和几种应用的设计方法。

(10) 第 10 章探讨微控制器日常应用中几个高级话题,本章还结合示例应用说明如何使用 CAN(控制器局域网)和 USB(通用串行总线)。

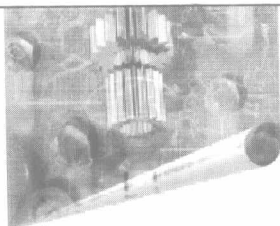
(11) 附录提供了 PIC18 系列微控制器指令集、常用 C 语言库函数以及习题答案。

本书主要由清华大学王俊华博士和华中科技大学肖亮老师翻译,参与本书翻译的人员还有:韦笑、王雷、李志云、李晓春、陈安华、孙宏、赵成璧、侯佳

## II PIC18微控制器：体系结构、编程与接口设计（C语言与汇编语言版）

宜、许伟、戴文雅、于樊鹏、刘朋、王嘉佳、李腾、邓卫、邓凡平、陈磊、李建锋、樊旭平、唐玮、周京平、李强、赵东辉、吴江华、孙燕、周刚、高强、吴江华、孙燕、周刚、王红亮、周峰、谢晖、李琳、刘明涛、孙向阳、李波、程云建、许晓哲、朱珂、曹锋、冯佳、张文波、赵楠等人。

虽竭尽所能，但由于水平有限，翻译过程中可能还会有错误和疏漏之处，敬请读者批评指正。



## 前 言

微控制器广泛应用于汽车、家电、工业控制、医疗设备等领域。本书全面讲解了这一现代奇迹的体系结构、编程和接口技术。本书以 Microchip 公司的 PIC18 系列微控制器为例讲解微控制器的体系结构、编程和接口技术。PIC18 是 Microchip 最新的 8 位微控制器,从这一系列的微控制器学到的知识,不仅适用于较早的 PIC 微控制器系列,也适用于其他制造商的微控制器。本书的难易程度几乎适合任何科学技术领域的教学大纲,而且,无论对于有丰富经验的从业人员,还是对于那些对微控制器深感兴趣的人,都具有宝贵的参考价值。

在过去 40 多年中,作者从事机电计算机、电子计算机、微处理器和微控制器方面的工作和教学活动。本书将结合作者多年总结出来的诸多技巧和方法,尽可能完善地向读者提供这一学科领域的经验和知识。本书的作者因在过去 22 年中写过 31 本著作而享有盛誉。这些著作涵盖了数字电子、微处理器和嵌入式系统等领域。其著作被译成多种语言版本,在世界知名大学广为流传。

### 本书组织结构

微控制器是由集成电路构成的完整计算机系统,其中包含了用于存储程序和数据的存储器,并集成了输入/输出端口设备。与其他计算机系统一样,人们通过对微控制器编程,使其执行特定的功能。

本书将全面讲解如何使用 C 语言和汇编语言对微控制器进行编程。尽管 C 语言更为常用,但汇编语言仍常常作为汇编代码块存在于 C 语言程序中。为了更好地理解编程,本书首先简要介绍计算机的大体结构,然后重点介绍 PIC18 系列微控制器的结构。第 1 章将介绍微处理器和微控制器领域的诸多术语。

介绍完微控制器的结构之后,后面的几章将讲解如何使用汇编语言编程来控制微控制器。这些章节将向读者展示许多常见的编程算法,同时配有大量的编程实例,演示如何在基于微控制器的系统中执行各种任务。第 3 章和第 4 章都将讲解汇编语言。第 4 章适合于希望深入研究汇编语言的读者,对于那些主要对 C 语言感兴趣的读者,可以将第 4 章作为选读章节。为了从本书学到尽可能多的知识,建议读者拥有初步的计算机编程知识和电子学基础。

## IV PIC18微控制器：体系结构、编程与接口设计 (C语言与汇编语言版)

详细讲解编程之后,将讲解微控制器和硬件的交互,同时讲解定时,以及微控制器与常见部件的接口技术。这些章节向读者展示了许多实际接口的例子,以及使用这些接口所需的软件驱动,包括许多外围设备的中断处理过程。常见的设备包括键盘、按键、开关、继电器、条形码阅读器、红外遥控器、螺线管、传感器、发动机、LCD和LED显示器、ADC、DAC、PWM,以及其他各种接口部件。对于连接设备,讲解了USB(通用串行总线)和CAN(控制器局域网),并配以适当的应用示例。在全书的自始至终,作者都努力地介绍和详尽地讲解尽可能多的设备,向读者演示诸多完整的系统应用。这样,读者可以从许多示例中剪切和粘贴代码,只需做些小的改动,就可以构建所需的应用。

### 讲解方法

每一章都包含一个内容简介,描述了该章中涉及的基本概念。每一章的主体部分将详细阐释这些概念,讲解中包含了许多应用示例。这些示例都很完整,包括示意图、对接口操作的讲解,以及每一接口示例涉及的所有软件的列表。通过这些示例,读者将得以更好地理解微控制器,并掌握微控制器在各种系统中的使用。

每一章的末尾,都有一个该章小结,回顾该章讲过的内容。为强化每一章中讲解的概念,章末包含了一系列的习题,为读者提供解决问题的机会,使读者经历基于微控制器的系统开发过程。学习完本书之后,读者将精通微控制器接口、编程和系统设计技术。

### 补充内容

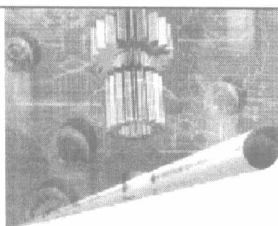
为访问在线的增补材料,教师需要申请一个教师访问码。浏览网页 <http://www.prenhall.com>,单击 Instructor Resource Center 链接,然后单击 Register Today 按钮获取教师访问码。注册 48 小时之后,您将会收到一封确认电子邮件,其中包含了一个教师访问码。收到访问码之后,浏览到上述网站并登录,那里有完整的操作指南,告诉您如何下载自己需要的材料。

### 联系作者

35年来,我一直教授计算机、微控制器和微处理器技术,至今,这些仍是我每天专注的事业,甚至在暑假也不例外。为了与读者保持联系和向读者提供额外信息,从1985年开始,我就已在电子学媒体上抛头露面。最初是在 CompuServe 公司(现为 America Online 公司所有),现在则是在互联网上。自从1995年以来,我在互联网上就一直有自己的网站。请读者访问我的网站,在那里可以看到有关许多话题的内容,还可以看到可能有助于学习计算机、微控制器和微处理器某方面知识的技术简报。

我的网站地址:<http://members.ee.net/brey>

我的 E-mail 地址: [bbrey@ee.net](mailto:bbrey@ee.net)



# 目 录

|                                    |    |
|------------------------------------|----|
| <b>第 1 章 计算机结构简介</b> .....         | 1  |
| 1.1 计算机基本结构 .....                  | 1  |
| 1.1.1 CPU .....                    | 2  |
| 1.1.2 存储器 .....                    | 5  |
| 1.1.3 I/O .....                    | 7  |
| 1.1.4 总线 .....                     | 7  |
| 1.1.5 微处理器和微控制器 .....              | 8  |
| 1.2 数制 .....                       | 9  |
| 1.2.1 数码 .....                     | 9  |
| 1.2.2 位置计数法 .....                  | 9  |
| 1.2.3 从其他进制转换为十进制 .....            | 11 |
| 1.2.4 从十进制转换成其他进制 .....            | 12 |
| 1.2.5 二进制编码的十六进制数 .....            | 14 |
| 1.2.6 补码和反码 .....                  | 14 |
| 1.3 计算机数据格式 .....                  | 16 |
| 1.3.1 ASCII 数据 .....               | 16 |
| 1.3.2 二进制编码的十进制数据 .....            | 17 |
| 1.3.3 字节数据 .....                   | 18 |
| 1.3.4 字数据 .....                    | 20 |
| 1.3.5 实数 .....                     | 22 |
| 1.4 本章小结 .....                     | 24 |
| 1.5 本章习题 .....                     | 25 |
| <b>第 2 章 PIC18 微控制器结构和编程</b> ..... | 28 |
| 2.1 PIC18 微控制器结构 .....             | 28 |
| 2.1.1 PIC 内部结构概览 .....             | 29 |
| 2.2 编程模型 .....                     | 35 |
| 2.2.1 寄存器文件 .....                  | 35 |
| 2.2.2 特殊功能寄存器 .....                | 36 |



|            |                            |           |
|------------|----------------------------|-----------|
| 2.3        | IDE .....                  | 42        |
| 2.3.1      | IDE 概览 .....               | 42        |
| 2.3.2      | 将程序下载到微控制器 .....           | 50        |
| 2.4        | 汇编语言、汇编器和连接器 .....         | 51        |
| 2.4.1      | 汇编器和连接器 .....              | 51        |
| 2.4.2      | 汇编语言语句 .....               | 52        |
| 2.4.3      | 伪指令 .....                  | 53        |
| 2.5        | 本章小结 .....                 | 57        |
| 2.6        | 本章习题 .....                 | 58        |
| <b>第3章</b> | <b>PIC18 微控制器指令集</b> ..... | <b>60</b> |
| 3.1        | 立即数指令 .....                | 60        |
| 3.1.1      | 立即数指令详述 .....              | 60        |
| 3.2        | 位操作指令 .....                | 64        |
| 3.3        | 字节指令 .....                 | 66        |
| 3.4        | 程序控制指令和间接寻址 .....          | 73        |
| 3.4.1      | 程序结构 .....                 | 75        |
| 3.4.2      | 间接寻址 .....                 | 80        |
| 3.5        | 表指令 .....                  | 82        |
| 3.6        | 宏序列 .....                  | 86        |
| 3.6.1      | 定义宏 .....                  | 86        |
| 3.7        | 本章小结 .....                 | 88        |
| 3.8        | 本章习题 .....                 | 89        |
| <b>第4章</b> | <b>汇编语言编程</b> .....        | <b>91</b> |
| 4.1        | 堆栈和队列结构 .....              | 91        |
| 4.1.1      | 数据堆栈 .....                 | 91        |
| 4.1.2      | 队列 .....                   | 95        |
| 4.2        | 复杂算术 .....                 | 98        |
| 4.2.1      | BCD 算术 .....               | 98        |
| 4.2.2      | 乘法 .....                   | 101       |
| 4.2.3      | 除法 .....                   | 105       |
| 4.3        | 十进制和二进制之间的转换 .....         | 107       |
| 4.3.1      | 二进制转换成 BCD .....           | 107       |
| 4.3.2      | BCD 转换成二进制 .....           | 111       |
| 4.4        | 延时 .....                   | 112       |
| 4.4.1      | 延时程序 .....                 | 112       |
| 4.5        | 程序示例 .....                 | 115       |
| 4.5.1      | 程序示例 1 .....               | 115       |
| 4.5.2      | 程序示例 2 .....               | 120       |

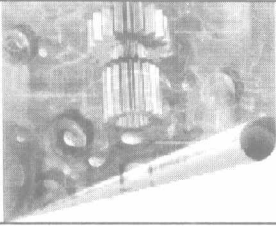


|              |                             |            |
|--------------|-----------------------------|------------|
| 4.6          | 本章小结 .....                  | 126        |
| 4.7          | 本章习题 .....                  | 126        |
| <b>第 5 章</b> | <b>PIC18 C 语言编程 .....</b>   | <b>129</b> |
| 5.1          | C18 C 语言编译器 .....           | 129        |
| 5.1.1        | C 语言和 IDE .....             | 130        |
| 5.1.2        | 示例程序 .....                  | 133        |
| 5.2          | 使用 C 语言头文件 .....            | 137        |
| 5.2.1        | 转换 .....                    | 137        |
| 5.2.2        | 存储器和字符串函数 .....             | 140        |
| 5.3          | C 语言示例程序 .....              | 150        |
| 5.3.1        | 示例 1 .....                  | 150        |
| 5.3.2        | 示例 2 .....                  | 154        |
| 5.4          | 数学函数库 .....                 | 160        |
| 5.5          | 本章小结 .....                  | 163        |
| 5.6          | 本章习题 .....                  | 163        |
| <b>第 6 章</b> | <b>PIC18 系列微控制器详解 .....</b> | <b>165</b> |
| 6.1          | 引脚输出和基本操作特性 .....           | 165        |
| 6.1.1        | 引脚输出 .....                  | 165        |
| 6.1.2        | 电源连接 .....                  | 166        |
| 6.1.3        | 输入/输出连接 .....               | 169        |
| 6.1.4        | 复位 .....                    | 169        |
| 6.1.5        | 时钟 .....                    | 173        |
| 6.2          | I/O 端口引脚 .....              | 177        |
| 6.2.1        | Port A .....                | 178        |
| 6.2.2        | Port B .....                | 180        |
| 6.2.3        | Port C、Port D、Port E .....  | 181        |
| 6.2.4        | I/O 示例 .....                | 181        |
| 6.3          | 中断简介 .....                  | 191        |
| 6.4          | 其他内部外围设备 .....              | 205        |
| 6.4.1        | 定时器 .....                   | 205        |
| 6.4.2        | ADC .....                   | 214        |
| 6.4.3        | EEPROM .....                | 217        |
| 6.4.4        | 比较和捕捉单元(CCP) .....          | 219        |
| 6.4.5        | 其他内部设备 .....                | 222        |
| 6.5          | 本章小结 .....                  | 222        |
| 6.6          | 本章习题 .....                  | 223        |
| <b>第 7 章</b> | <b>基本输入和输出 .....</b>        | <b>226</b> |
| 7.1          | 基于开关的输入 .....               | 226        |
| 7.1.1        | 开关接口 .....                  | 226        |

|              |                  |            |
|--------------|------------------|------------|
| 7.1.2        | 开关去抖             | 227        |
| 7.1.3        | 键盘               | 231        |
| 7.2          | 显示设备             | 234        |
| 7.2.1        | LED              | 234        |
| 7.2.2        | 七段码 LED 显示器      | 239        |
| 7.2.3        | LCD 显示器          | 247        |
| 7.2.4        | C18 对 LCD 显示器的支持 | 259        |
| 7.2.5        | 真空荧光显示器          | 261        |
| 7.3          | 控制电机             | 262        |
| 7.3.1        | 步进电机             | 262        |
| 7.3.2        | DC 电机            | 265        |
| 7.4          | 继电器、螺线管和传感器      | 275        |
| 7.4.1        | 继电器              | 275        |
| 7.4.2        | 螺线管              | 276        |
| 7.4.3        | 传感器              | 278        |
| 7.4.4        | 转轴编码器            | 281        |
| 7.4.5        | 流量传感器            | 285        |
| 7.4.6        | 红外遥控设备           | 287        |
| 7.4.7        | 感知气体             | 292        |
| 7.5          | 本章小结             | 296        |
| 7.6          | 本章习题             | 297        |
| <b>第 8 章</b> | <b>中断</b>        | <b>300</b> |
| 8.1          | 中断回顾             | 300        |
| 8.1.1        | 中断服务程序           | 302        |
| 8.2          | USART 和中断        | 313        |
| 8.2.1        | 串行数据             | 314        |
| 8.2.2        | 控制 USART         | 315        |
| 8.3          | 输入改变中断           | 323        |
| 8.4          | 中断系统示例           | 331        |
| 8.4.1        | 示例系统 1           | 331        |
| 8.4.2        | 示例系统 2           | 363        |
| 8.5          | 本章小结             | 369        |
| 8.6          | 本章习题             | 370        |
| <b>第 9 章</b> | <b>控制系统</b>      | <b>373</b> |
| 9.1          | 规划控制系统           | 373        |
| 9.2          | 示例系统             | 378        |
| 9.2.1        | 示例 1             | 378        |
| 9.2.2        | 示例 2             | 396        |



|                                     |            |
|-------------------------------------|------------|
| 9.2.3 示例 3 .....                    | 399        |
| 9.3 本章小结 .....                      | 415        |
| 9.4 本章习题 .....                      | 416        |
| <b>第 10 章 高级话题 .....</b>            | <b>418</b> |
| 10.1 存储器扩展 .....                    | 418        |
| 10.1.1 增加串行 EEPROM .....            | 418        |
| 10.1.2 使用附加存储器的应用程序 .....           | 424        |
| 10.2 引导模块 .....                     | 442        |
| 10.2.1 引导加载器 .....                  | 442        |
| 10.2.2 写程序闪存 .....                  | 445        |
| 10.3 扩展 I/O .....                   | 451        |
| 10.3.1 增加输出连接 .....                 | 451        |
| 10.3.2 增加输入连接 .....                 | 453        |
| 10.3.3 使用双线接口 .....                 | 454        |
| 10.3.4 数模转换器 .....                  | 455        |
| 10.4 CAN .....                      | 458        |
| 10.4.1 CAN 互联 .....                 | 458        |
| 10.4.2 CAN 应用 .....                 | 460        |
| 10.5 USB .....                      | 475        |
| 10.5.1 USB 连接器 .....                | 476        |
| 10.5.2 USB 数据 .....                 | 476        |
| 10.5.3 USB 命令 .....                 | 477        |
| 10.5.4 PIC18 和 USB .....            | 479        |
| 10.6 PIC18 扩展指令 .....               | 499        |
| 10.7 本章小结 .....                     | 501        |
| 10.8 本章习题 .....                     | 502        |
| <b>附录 A PIC18 系列微控制器指令集 .....</b>   | <b>504</b> |
| <b>附录 B 常用 C 语言库函数 .....</b>        | <b>509</b> |
| <b>附录 C 偶数编号的习题答案 .....</b>         | <b>522</b> |
| <b>附录 D 使用 USB 接口的 PIC 系统 .....</b> | <b>537</b> |



# 第 1 章

## 计算机结构简介

PIC(Programmable Interface Controller, 可编程接口控制器)微控制器是微芯科技公司(Microchip Technology Incorporated, <http://www.microchip.com>)生产的 RISC(Reduced Instruction Set Computer, 精简指令集计算机)系列集成电路。目前可用的 PIC 微控制器,其封装尺寸小到 18 只引脚,大到 128 只引脚。Parallax 公司(<http://www.parallax.com>)为 PIC 微控制器生产了一系列被称为 BASIC Stamp<sup>®</sup>的产品,这些产品使用 BASIC 语言或 Java 语言进行编程,而不是使用汇编语言。本章介绍微控制器和计算机系统的结构,以此作为使用微控制器进行控制的基础。本章还定义了微控制器技术方面的许多术语,以此作为进入这一神奇领域的入门性介绍。

本章包含以下内容:

- 定义微控制器领域所使用的术语。
- 讲解计算机系统各组成部分的作用。
- 讲解计算机系统各部分的操作及其与其他系统部件的相互关系。
- 定义微控制器所使用的数据类型。
- 讲解常见数制之间的转换。

### 1.1 计算机基本结构

最常见的计算机结构如图 1-1 中的结构图所示。1856 年,查尔斯·巴贝奇(Charles Babbage)在其设计的分析机(A analytical Engine)中首次提出和使用了这种结构(<http://www.cbi.mn.edu>)。此后,这种结构继续作为许多现代数字计算机的基本系统结构而存在。分析机是一种机械式的计算机系统,它使用手柄提供动力。后来,在查尔斯·巴贝奇的工作被人们重新发现之前,这种结构归功于约翰·冯·诺依曼(John von Neumann),他于 1945 年初描述了这种结构,而如今这种结构常被称为冯·诺依曼结构(von Neumann Architecture)。第 2 章将讲解另一种结构,被称为哈佛结构(Harvard Architecture)。虽然这种结构并没有冯·诺依曼结构那么常见,却是 PIC 系列微控制器内部所采用的结构。

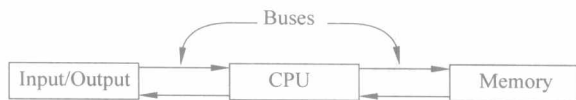


图 1-1 计算机系统结构框图

### 1.1.1 CPU

如图 1-1 所示的计算机系统结构虽然简单，却包含了 3 个主要的模块，代表了大多数现代数字计算机系统的结构。结构图中的核心部分是 CPU(中央处理器)，它是整个系统的控制元素。早期的 CPU 是机械式的系统，如分析机、计算器和会计机，广泛应用于 20 世纪 70 年代。早期的电子数字计算机最初使用真空管构造，后来使用晶体管构造。现代计算机系统则使用 MOSFET(或 CMOS, 互补金属氧化物半导体场效应管)集成电路构造。

中央处理器的位宽可以是 4 位到 64 位之间的任何位数。“位”是一个二进制数，其取值只能是 0 或 1。因为目前所采用的电路只能精确地表示二进制数据，所以大多数电子数字计算机系统都基于二进制数字系统。4 位处理器的例子是 Intel 的早期的微处理器 4004 或 4040。目前，最新的微处理器是 AMD(Advanced Micro Devices)和 Intel 的 64 位的微处理器。CPU 的大小(位数)取决于存储器地址宽度以及算术和逻辑单元。算术和逻辑单元用来对系统中的整数执行算术和逻辑运算。4 位的微处理器对 4 位的整数执行算术和逻辑运算，而 64 位的微处理器最大可以对 64 位的整数执行这些运算。64 位机也可以对 32 位、16 位和 8 位的整数执行算术和逻辑运算。如今，最常见的最小直接寻址数一般是 8 位的二进制数，被称为一个字节(Byte)。微处理器和微控制器的位宽可以是 1 位到任意位数，尽管如此，大多数微处理器的位宽是 8 的倍数，而许多微控制器的位宽则要么是 8 位的，要么是 16 位的。

CPU 的另一个参数是时钟频率。时钟频率大小不一，目前，微控制器的时钟频率一般在几兆赫，而用于桌面计算机的微处理器更强大，其时钟频率高达数千兆赫。兆赫(MHz)的系统基于每秒百万个脉冲的时钟频率，而千兆赫(GHz)的系统基于每秒十亿个脉冲的时钟频率。为纪念赫兹(Heinrich Rudolf Hertz, <http://www.idealfinder.com/history/inventors/hertz.htm>)，人们用其名字定义交流电每秒钟的脉冲数(最初被称为每秒循环数或 cps)。尽管微处理器或微控制器的时钟频率不一定代表每秒所执行的指令数，但是，时钟频率直接决定了系统的速度。每秒钟执行指令的精确数目取决于 CPU 的内部设计、所执行指令的复杂度和所执行的程序。但是，有一点是确定的：如果拿 10MHz 的机器与 2GHz 的机器相比，2GHz 的机器将是两台机器中速度较快的。

微处理器的类型也可以决定处理器的速度。RISC(Reduced Instruction Set Computer, 精简指令集计算机)每个时钟周期执行一条指令，而且只包含基本的指令。与 RISC 机器相比，CISC(Complex Instruction Set Computer, 复杂指令集计算机)可执行更多的指令，但有些指令需要多个时钟周期才能完成。如今，这些术语已经模糊不清，因为大多数新型的微处理器使用 CISC 和 RISC 混合技术，在单个时钟周期内执行大多数指令。而且，有些微处理器(如 Intel 的 Pentium®系列)可以在三分之一的时钟周期内执行许多种指令。需要明白的是，这些新型机器通常包含同时运行的整数乘法单元。

**CPU 的任务。** CPU 的任务是执行一连串的命令，这些命令组织在一起，称为程序(Program)。实际上，从 CPU 上电开始到 CPU 断电，CPU 唯一能做的事情就是执行指令。它不停地从存储系统中取出指令，然后执行指令。CPU 的设计思想和运作方式，最早由查



尔斯·巴贝奇提出,至今仍未改变,而且很有可能继续存在相当长的时间。较新型的机器能够同时执行多条指令,因为这些机器的 CPU 内部有多个算术和逻辑处理器。某些最新出现的机器有多个内核,多内核机器的集成电路中包含了不止一个微处理器。

计算机中的程序是以连续的形式排列在一起的指令集合,CPU 也以连续的形式执行这些指令。例如,假设程序需要将 6 与 2 相加,则第一步(或第一条指令)取得数字 6,第二步取得数字 2,第三步将 6 和 2 加在一起,最后一步的操作是将和保存在存储系统中的某个地方。正如这个例子所演示的,将 6 与 2 相加的简单任务被分割成了多个步骤,这些步骤被称为指令(Instruction)。这些指令则组成程序,程序由微处理器或微控制器执行。

- (1) 取得数字 6。
- (2) 取得数字 2。
- (3) 将 6 和 2 相加。
- (4) 将和保存在存储器中。

这个简单程序的每一步都被转换成 CPU 能够理解的语言,这种语言被称为机器语言(Machine Language)。机器语言是一系列代表指令的数字。假设一个 CPU 的“取指令”指令为数字 00,“相加”指令为数字 01,“保存”指令为数字 02,那么,上述程序数字形式的机器语言为 00 06 00 02 01 02 00,或者如例 1-1 所示。

#### 例 1-1:

```
00 06
00 02
01
02 00
```

从这个例子可以看出,写数字形式的机器代码十分麻烦,而且代码不易辨认。虽然在计算处理的早期阶段人们使用这种代码编写程序,但现在人们已几乎不再那样做了。取而代之的是一种被称为汇编器(Assembler)的工具,人们使用这种工具,以一种助记码(Mnemonic Opcode)的形式编写程序,如 GET 和 ADD。汇编器本身也是一个程序,它将符号形式的汇编语言程序转换为数字形式的机器代码。生成例 1-1 中程序的汇编语言程序,如例 1-2 所示。

#### 例 1-2:

```
GET 6
GET 2
ADD
STORE 0
```

尽管这样依然有些麻烦,但至少比一串数字易读多了。因为汇编语言程序十分冗长,写起来也很困难,现代编程通常使用高级语言(High-level Language)生成机器代码。高级语言程序接受类似英语的语言输入,然后将其转换为数字形式的机器代码。BASIC、C、C++ 和 Java 等都是高级语言。同样的程序用 C 语言编写,如例 1-3 所示。

## 例 1-3:

```
char answer;           //set aside a place for the answer
answer=6+2;          //perform the addition and store sum in answer
```

从这段 C 程序可以看出, C 程序的输入更少, 不需要知道机器代码或数字形式的存储地址, 而且易于阅读。C 语言程序生成的机器代码, 其执行效率可能不如汇编器生成的机器代码效率高。但是, C 语言程序更容易编写、理解和维护, 而且写起来需要的时间更短, 技巧更少。正是由于这个原因, 软件通常使用高级语言编写。计算机系统的软件 (Software) 指计算机中的程序, 计算机系统的硬件 (Hardware) 指计算机系统的电路。有时, 软件被称为可变硬件 (Variable Hardware), 这是因为软件改变了硬件起作用的方式。对基于微控制器的系统而言, 情况尤为如此。

读者应该知道的一件有趣的事情是, 奥古斯塔·爱达·拜伦 (Augusta Ada Byron, 与拜伦伯爵结婚之后被称为 Lovelace 伯爵夫人) 获得了第一个计算机程序设计师的美誉。她获得这一声誉的原因是她曾经为查尔斯·巴贝奇的分析机用数字形式的机器语言编写了程序。有一种计算机语言被称为 Ada, 正是为了纪念她而命名的。这种语言是美国国防部使用的计算机语言。

**CPU 的功能。**在计算机系统中, CPU 主要执行 3 种任务: (1) 数据传送; (2) 算术和逻辑运算; (3) 程序流程控制。数据传送是 CPU 最主要的任务。CPU 大部分时间都用来传送数据。数据传送包括从存储器中提取数据、在寄存器或存储器不同位置之间传送数据、以及在存储器和 CPU 之间, 或者 I/O (输入/输出) 设备和 CPU 之间传送算术或逻辑运算的结果。数据传送中的指令提取是 CPU 最常见和最重要的操作。指令提取使 CPU 能够以很高的速度从存储系统执行程序。这种存储程序的概念使计算机变得十分强大。存储程序的概念最初是由查尔斯·巴贝奇预言的。大多数程序用 50% 的时间甚至更短的时间进行数据传送。表 1-1 列出了多数 CPU 最常见的数据传送操作。

表 1-1 最常见的数据传送操作

| 操 作         | 说 明                  |
|-------------|----------------------|
| 存储器→操作码寄存器  | 从存储器取操作传送到操作码寄存器     |
| 存储器→数据寄存器   | 从存储器取数据传送到数据寄存器      |
| 数据寄存器→存储器   | 将寄存器数据保存到其存储单元中      |
| 数据寄存器→数据寄存器 | 将数据寄存器的内容传送到另一个数据寄存器 |
| 数据寄存器→I/O   | 将数据寄存器的内容传送到输入/输出设备  |
| I/O→数据寄存器   | 将输入/输出设备的内容传送到数据寄存器  |

与数据传送相比, CPU 用来对数字进行算术和逻辑运算的时间就少得多。通常, 程序只需要 CPU 花费小部分时间进行算术和逻辑运算。表 1-2 列出了 CPU 执行的常见算术和逻辑操作。通常情况下, 这些操作由整数算术单元 (Integer Arithmetic Unit) 执行。如果没有程序或专门的处理单元 (称为算术单元 (Arithmetic Unit) 或数字处理器 (Numeric Processor)), CPU 就不能对实数进行操作。微处理器通常包含一个数字处理器和一个整数处理器, 但很少有微控制器包含数字处理器。





表 1-2 常见的算术和逻辑操作

| 操 作 | 说 明  |
|-----|--|
| 加法  | 寄存器之间、寄存器和存储器之间,或与立即数相加                        |
| 减法  | 寄存器之间、寄存器和存储器之间,或与立即数相减                        |
| 乘法  | 寄存器之间、寄存器和存储器之间,或与立即数相乘;结果具有两倍位宽               |
| 除法  | 寄存器之间、寄存器和存储器之间,或与立即数相除;被除数具有两倍位宽(有些微控制器不提供除法) |
| 取负  | 改变数字的正负符号                                      |
| 与   | 寄存器之间、寄存器和存储器之间,或与立即数按位进行逻辑与操作                 |
| 或   | 寄存器之间、寄存器和存储器之间,或与立即数按位进行逻辑或操作                 |
| 异或  | 寄存器之间、寄存器和存储器之间,或与立即数按位进行逻辑异或操作                |
| 取反  | 寄存器之间、寄存器和存储器之间,或与立即数按位进行逻辑取反操作                |

CPU 最强大的功能是通过简单的数字决策改变程序的流程。程序流程控制指令(Program Flow Control Instruction)使程序中的代码块能够多次重复使用。程序流程控制指令还可以将程序流程传送给函数或过程。程序流程控制指令分为无条件程序流程控制指令和有条件程序流程控制指令。无条件程序流程控制指令有 GOTO 指令和 CALL 指令。有条件程序流程控制指令通过测试一个数字满足的条件,然后决定是否改变程序流程。

有条件程序流程控制指令通过测试一个数字,确定该数字是否为零,是否在相加之后出现进位,或者运算结果是正还是负。表 1-3 列出了大多数 CPU 中常见的可测试条件。有条件程序流程控制指令代表了程序中的小部分指令,正是这一小部分指令使 CPU 成为计算机系统中的重要组成部分,使计算机具备了思考的能力。例如,计算机如何确定键盘上输入的数字是不是 3? 为确定是不是 3,计算机从输入的数字中减去 3,然后测试结果是否为零。如果结果为零,程序流程改变,执行某种对输入数字 3 的响应。从这里得到的启发是计算机能够对键盘上输入的数字是不是 3 进行思考和推理,而实现起来则是从输入的数字中减去 3,然后测试结果是否为零。是什么使计算机成为如此强大的工具呢? 软件,程序设计师编写的软件才是关键的因素,因为软件能使计算机进行思考和推理。

表 1-3 常见的 CPU 测试条件

| 条 件 | 说 明    | 条 件 | 说 明           |
|-----|--------|-----|---------------|
| 零   | 数字是否为零 | 符号  | 数字符号是正还是负     |
| 进位  | 是否发生进位 | 溢出  | 有些 CPU 测试算术溢出 |

### 1.1.2 存储器

存储器(Memory)是计算机系统极其重要的组成部分,因为它为程序和程序使用的数据提供存储的场所。没有存储器,计算机系统也只不过就像一个能进行四则运算的计算器那样。有了存储器,计算机能以很高的速度访问存储器中的程序指令。顺便提一下,查尔斯·巴贝奇的分析机包含了存储器,能够存储 1000 个数字,每个数字的宽度是 20 个十进制位。该分析机也执行保存在存储器中的程序。