计算机系统的自主设计

赵 刚 张 垒 高朔弋
夏俊杰 路 明 刘志刚 著

◎ 为您打开计算机设计的神秘之门
◎ 教您一步步地设计自己的计算机系统
◎ 使您深入理解计算机系统的本质

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

计算机系统的自主设计

赵 刚　张 垒　高朔弋
夏俊杰　路 明　刘志刚　著

電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 简 介

计算机系统分为通用计算机系统和嵌入式计算机系统，计算机的大量普及，极大地推动了人类社会的进步。但是，您是否觉得计算机系统的设计神秘至极？本书将为您打开计算机系统设计的神秘之门，教您如何去设计满足自己需求的计算机系统。

本书的重点是讲解计算机系统自主设计的方法，共分为四篇，全书以一个全部软硬件均自主定义与设计的教学用计算机系统的实现过程为主线，详细地介绍了自定义指令系统、CPU 芯片、汇编语言、高级语言、编辑器、汇编器、编译器、链接器、编程器，以及计算机系统的完整开发流程，试图让读者将往日所学到的零碎、割裂的知识通过书中的实例予以串联整合，使读者对计算机系统的本质得以深入理解。

本书特别适合 IT 领域或非 IT 领域的广大计算机爱好者阅读，读者群覆盖高中生、本科生、硕士生和博士生，可作为高校或职业学校的课程教材，也可作为研究院所和企事业单位的培训教材。

图书在版编目（CIP）数据

计算机系统的自主设计/赵刚等著. —北京：电子工业出版社，2011.10
ISBN 978-7-121-14553-7

Ⅰ. ①计… Ⅱ. ①赵… Ⅲ. ①计算机系统－系统设计 Ⅳ. ①TP302.1

中国版本图书馆 CIP 数据核字（2011）第 182681 号

责任编辑：田宏峰　　特约编辑：牛雪峰
印　　刷：
装　　订：北京京师印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
开　　本：787×1 092　1/16　印张：28.5　字数：760 千字
印　　次：2011 年 10 月第 1 次印刷
印　　数：4 000 册　　定价：59.00 元（含 CD 光盘 1 张）

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

计算机分为通用计算机和嵌入式计算机。前者追求高速运行、海量存储、多功能，人们日常使用的台式机及笔记本电脑、网络服务器等均属于通用计算机类。后者则常以一个产品中的内部控制器或以专用计算机的形式隐形地嵌入于具体的产品中，作为产品的一个部件，追求量体裁衣，最大限度地降低成本、体积与功耗。日常生活中常用的手机、电视机、空调、冰箱、汽车、电饭煲等，正因为其中的嵌入式计算机默默地运行，才使其具有了智能功能。近三十年来，计算机的出现极大地推动了人类生产和生活的进步，并使之对计算机形成了强烈的依赖感。目前，计算机已广泛地运用于工业、农业、商业、医疗、教育等各行各业，已进入千家万户。

三十年过去了，广大从事 IT 技术及产品研发人员经历了苹果机、单板机、单片机、286、386、奔腾、DSP、ARM 等硬件升级，经历了 BASIC、FORTRAN、PASCAL、COBOL、C、DELPHI、Java 等语言变迁。目前该行业的从业人员（含高校在校生）已超过 1 000 万人。但时至今日，在市面的书架上，在高校的教学中，仍在大量地介绍国外的各类商用 CPU、编程语言，以及开发系统软件该如何去选型、如何去使用。基于此，作者用了两年多的时间完成了本书，全书以一个全部软硬件均自主定义与设计的教学用计算机系统的实现过程为主线，详细地介绍了自定义指令系统、CPU 芯片、汇编语言、高级语言、编辑器、汇编器、编译器、链接器、编程器以及计算机系统的完整开发流程，试图让读者将往日所学到的零碎、割裂的知识，通过书中的实例予以串联整合，使读者对计算机系统的本质得以深入理解，由一名使用者提升至一名设计者和创造者，为社会开发出更多更好、具有自主知识产权的 IT 产品来。

尽管本书的内容覆盖了大量的 IT 专业及专业基础课程内容（如数字电路、微机原理与接口、单片机技术、编译方法、C++语言、EDA 技术等），但本书并不是仅仅针对 IT 专业技术人员和高年级学生写的。对于非 IT 专业的读者或 IT 专业的低年级学生，建议按书中的章节顺序及介绍的方法，动手制作并完成相应设计，这样会在不知不觉中掌握相关的专业理论知识和技能技巧。毕竟，最好的学习方法是带着问题去学习。本书将与实例相关的所有软硬件资料公开，其中硬件电路图和软件源代码都已经反复验证。

全书分为四篇，共计 17 章。第 1 章以一个产品加工厂为例，通俗形象地对计算机系统结构、工作原理及开发运行流程进行了介绍；第 2 章对指令系统的基本概念、自定义指令系统设计方法进行了阐述，是 CPU 芯片的设计目标；第 3 章介绍了 VHDL 和 Verilog HDL 两种硬件描述语言的使用方法；第 4 章详细介绍了 FPGA 开发系统软件 QUARTUS II的使用方法、设计技巧，以及代码自动生成方法；第 5 章对自定义 CPU 的内核和片内外设的设计及 FPGA 实现进行了详细的介绍，并给出了用 VHDL 和 Verilog HDL 两种语言分别进行描述设计的方法；第 6 章介绍了用制作出的 CPU 构建计算机硬件系统的设计、测试方法；第 7 章介绍了什么是软件、软件对硬件的依赖关系以及软件的基本架构和开发流程；第 8 章和第 9 章介绍了汇编语言和高级语言的定义方法和基本组成；第 10 章通过一些编程实例介绍了 VC++ 6.0 工具与 C++语言的使用，并介绍了 Windows 系统平台上的应用程序的开发；第 11 章介绍了使用 MFC 设计编辑器的原理、步骤和技

巧；第 12 章、第 13 章和第 14 章简要地介绍了汇编器、编译器和链接器的设计原理和设计流程，并使用 C++语言将其实现；第 15 章介绍了编程器的制作原理和方法，包括计算机软件界面制作、串口通信和数据烧写；第 16 章对整个计算机系统使用自定义的 SCL 汇编语言进行了实例测试，详细地介绍了使用汇编语言进行开发的流程；第 17 章对整个计算机系统使用自定义的 SCH 高级语言进行了实例测试，详细地介绍了高级语言源程序的开发流程，并对三种基本程序结构、子程序，以及中断服务程序进行了实例分析。

本书特别适合 IT 领域或非 IT 领域的广大计算机爱好者阅读，读者群覆盖高中生、本科生、硕士生和博士生。可作为高校或职业学校的课程教材，也可作为研究院所和企事业单位的培训教材。

本书编写组由赵刚、张垒、高朔弋、夏俊杰、路明、刘志刚六位教师组成，四川大学电子信息学院研究生张金龙参与了部分章节的编写工作，编写工作由参加编写的教师集体讨论、分工编写、交叉修改，历经两年多完成。全书由赵刚担任主编，负责大纲拟定、组织编写与统稿工作。

在编写过程中，得到了华为技术有限公司成都研究所郑小燕工程师的大力支持和帮助，借本书出版之际，特表示感谢！

虽然本书是在作者多年教学和科研工作的基础上完成的，但由于作者水平有限，疏漏之处在所难免，敬请广大读者批评指正（E-mail:zhaogang@scu.edu.cn）。

作　　者
于四川大学

目录

CONTENTS

第一篇　计算机系统构成及自定义CPU芯片设计

第 1 章　计算机系统构成及工作原理 …… 3

1.1　从一个产品加工厂谈起 …… 3

1.2　计算机诞生及系统构成 …… 4

1.2.1　计算机诞生 …… 4

1.2.2　计算机系统结构 …… 4

1.3　计算机开发及运行流程 …… 5

1.3.1　计算机软硬件开发流程 …… 5

1.3.2　计算机运行流程 …… 6

第 2 章　自定义指令系统设计 …… 7

2.1　指令系统概述 …… 7

2.1.1　指令系统基本概念 …… 7

2.1.2　指令系统设计方法 …… 7

2.1.3　指令系统设计流程 …… 10

2.2　自定义指令系统 …… 10

2.2.1　指令分类与指令功能选择 …… 10

2.2.2　指令格式与寻址方式 …… 11

2.2.3　指令集 …… 13

2.2.4　CPU 芯片自动化设计流程 …… 19

第 3 章　HDL 硬件描述语言 …… 20

3.1　VHDL 硬件描述语言 …… 20

3.1.1　VHDL 程序的基本结构 …… 20

3.1.2　VHDL 语言的基本要素 …… 23

3.1.3　VHDL 程序的常用描述语句 …… 25

3.1.4　VHDL 程序的常用描述方式 …… 26

3.1.5　组合逻辑电路设计 …… 27

3.1.6　时序逻辑电路设计 …… 30

3.1.7　存储器设计 …… 32

3.1.8　状态机设计 …… 34

3.2　Verilog HDL 硬件描述语言 …… 36

3.2.1　Verilog HDL 语言的基本结构 …… 36

3.2.2　Verilog HDL 语言的基本要素 …… 38

3.2.3　Verilog HDL 程序的常用描述语句 …… 41

3.2.4　Verilog HDL 程序的常用描述方式 …… 43

3.2.5　组合逻辑电路设计 …… 44

3.2.6　时序逻辑电路设计 …… 46

3.2.7　存储器设计 …… 47

3.2.8　状态机设计 …… 49

3.3　VHDL 和 Verilog HDL 语言的对应关系 …… 52

第 4 章　QUARTUS II 开发工具 …… 54

4.1　QUARTUS II 开发系统软件使用入门 …… 54

4.1.1　QUARTUS II 软件安装 …… 54

4.1.2　QUARTUS II 平台下的芯片开发流程 …… 55

4.1.3　电子琴芯片设计实例 …… 56

4.2　QUARTUS II 设计技巧 …… 71

4.2.1　增量编译 …… 71

4.2.2　时序约束 …… 73

4.2.3　逻辑锁定 …… 75

4.2.4　嵌入式逻辑分析仪 …… 77

4.2.5　定制 LPM_ROM …… 81

4.3　DSP Builder 代码自动生成 …… 87

4.3.1　DSP Builder 代码自动生成流程 …… 87

4.3.2　直接数字频率合成器原理 …… 87

4.3.3　DDS 在 MATLAB/Simulink 中的建模与仿真 …… 89

4.3.4　DDS 在 QUARTUS II 中的 FPGA 实现 …… 92

第 5 章　自定义 CPU——SCU 芯片设计 …… 98
5.1　SCU 结构设计 …… 98
5.1.1　总线宽度设计 …… 98
5.1.2　SCU 结构设计 …… 99
5.2　SCU 内核设计 …… 103
5.2.1　控制单元设计 …… 103
5.2.2　运算单元设计 …… 127
5.2.3　寄存器堆设计 …… 132
5.2.4　SCU 内核模块连接 …… 141
5.3　SCU 片内外设设计 …… 143
5.3.1　中断控制器 INTERUPT 设计 …… 143
5.3.2　存储器接口电路设计 …… 146
5.3.3　I/O 接口电路设计 …… 146
5.4　SCU 整体电路 …… 152
5.5　SCU 芯片实现 …… 153
第 6 章　计算机硬件系统开发与机器语言编程 …… 158
6.1　计算机硬件系统设计 …… 158
6.1.1　计算机硬件系统结构设计 …… 158
6.1.2　计算机硬件系统原理图设计 …… 159
6.2　机器语言编程 …… 162
6.2.1　机器程序开发流程 …… 162
6.2.2　12 位二进制输入电路测试程序 …… 163
6.2.3　3 位十进制输出电路测试程序 …… 164
6.2.4　12 位二进制输入电路测试程序 …… 165
6.2.5　3 位十进制输入电路测试程序 …… 166
6.2.6　不同时钟频率对计算机运行速度的影响 …… 167

第二篇　软件开发流程及自定义编程语言

第 7 章　计算机软件的硬件基础 …… 171
7.1　软件概述 …… 171
7.2　数据——加工对象 …… 172
7.2.1　数据的硬件基础 …… 172
7.2.2　数据表达 …… 173
7.2.3　数据存储方式 …… 176
7.3　程序——加工流程 …… 177
7.3.1　程序的硬件基础 …… 177
7.3.2　程序加工流程 …… 178
7.3.3　程序的基本结构及其开发流程 …… 180
7.3.4　提高编程效率的有效途径——子程序与库 …… 181
第 8 章　自定义汇编语言——SCL 语言 …… 185
8.1　汇编语言概述 …… 185
8.1.1　从机器语言到汇编语言 …… 185
8.1.2　汇编语言组成 …… 186
8.2　指令语句与伪指令设计 …… 186
8.2.1　指令语句 …… 186
8.2.2　伪指令 …… 192
8.3　汇编语言结构设计 …… 193
8.3.1　数据段与代码段 …… 193
8.3.2　子程序 …… 194
8.3.3　中断服务程序 …… 194
第 9 章　自定义高级语言——SCH 语言 …… 196
9.1　高级语言概述 …… 196
9.1.1　高级语言与低级语言 …… 196
9.1.2　高级语言组成 …… 197
9.2　SCH 语言基本成分设计 …… 197
9.2.1　常量及变量 …… 197
9.2.2　运算与赋值 …… 198
9.2.3　输入/输出 …… 199
9.2.4　控制操作 …… 199
9.3　SCH 语言结构设计 …… 201
9.3.1　程序结构 …… 201
9.3.2　子程序 …… 203
9.4　SCH 语言单词与语法归纳 …… 204

第三篇　工具软件的自主开发

第 10 章　C++开发语言与 VC 开发工具 …… 207
10.1　熟悉 VC++6.0 软件开发工具 …… 207
10.1.1　VC++工具及帮助文档 MSDN …… 207
10.1.2　Windows 应用程序及 VC 编程流程 …… 208
10.1.3　编写第一个控制台应用程序 …… 210

10.1.4 编写第一个窗口应用程序……215
10.1.5 程序调试及调试窗口观察……224
10.2 C++语言基础……229
10.2.1 C++语言组成与程序结构……229
10.2.2 常量与变量运用……231
10.2.3 运算与赋值……240
10.2.4 控制语句……244
10.2.5 函数的运用……247
10.2.6 预编译指令……253
10.2.7 类与对象的运用……255
10.2.8 类的继承与派生……258
10.3 窗口应用程序编程——使用 API……262
10.3.1 API 库……262
10.3.2 资源及资源编辑……264
10.3.3 菜单资源运用……266
10.3.4 对话框资源运用……267
10.3.5 控件……269
10.3.6 消息……270
10.3.7 计算器制作实例……273
10.4 窗口应用程序编程——使用 MFC……278
10.4.1 MFC 类库……279
10.4.2 MFC 应用程序运行原理……279
10.4.3 MFC 编程方法……287
10.4.4 计算器制作实例……290
第 11 章 编辑器的自主设计……294
11.1 编辑器概述……294
11.1.1 代码编辑器功能……294
11.1.2 代码编辑器制作流程……294
11.2 编辑器基本框架生成……295
11.3 关键字高亮与文件操作……296
11.3.1 关键字高亮……296
11.3.2 文件打开与保存……297
11.4 状态栏、快捷菜单与 SHELL 操作……303
11.4.1 在状态栏上显示光标位置……303
11.4.2 快捷菜单设计……308
11.4.3 编辑器打开不产生新文档……309
11.4.4 文件拖放……310
第 12 章 汇编器的自主设计……313
12.1 汇编器概述……313
12.1.1 汇编器功能……313
12.1.2 汇编器组成……313
12.2 汇编器工作原理……314
12.2.1 符号表……314
12.2.2 单词识别……317
12.2.3 数据定义处理……319
12.2.4 指令语句处理……320
12.2.5 目标代码文件生成……322
12.3 汇编器编程实现……323
12.3.1 汇编器模板及模块间调用关系……323
12.3.2 相关符号表和地址计数器的表示……325
12.3.3 汇编器细化模板及程序说明…327
第 13 章 编译器的自主设计……333
13.1 编译器概述……333
13.1.1 编译器功能……333
13.1.2 编译器组成……333
13.1.3 编译器与汇编器……334
13.2 编译器的工作原理……334
13.2.1 符号表——编译过程中有关符号的存储……334
13.2.2 单词识别——字符的组合……335
13.2.3 语法分析——语法关系的比较……335
13.3 指令代码生成……337
13.3.1 临时存储区及其运用……338
13.3.2 表达式的代码生成……339
13.3.3 赋值语句、输入/输出语句的代码生成……341
13.3.4 控制语句的代码生成……343
13.4 编译器的编程实现……346
13.4.1 编译器程序模板及模块调用关系……346
13.4.2 相关符号表的表示……349
13.4.3 编译程序细化模板及程序说明……351

第 14 章　链接器的自主设计 ····· 360
14.1 链接器概述 ····· 360
14.1.1 链接器功能 ····· 360
14.1.2 链接器组成 ····· 361
14.2 链接器工作原理 ····· 361
14.2.1 存储器模型 ····· 361
14.2.2 链接命令文件 ····· 361
14.2.3 重定位 ····· 362
14.2.4 可执行文件生成 ····· 364
14.3 链接器编程实现 ····· 365
14.3.1 主要功能模块及模块调用关系 ····· 365
14.3.2 相关符号表的表示 ····· 366
14.3.3 链接器程序分析 ····· 367
第 15 章　编程器的自主设计 ····· 372
15.1 编程器概述 ····· 372
15.1.1 编程器软件组成 ····· 372
15.1.2 编程器硬件组成 ····· 373
15.2 编程器硬件电路设计 ····· 373
15.2.1 编程器硬件总电路 ····· 373
15.2.2 EEPROM 芯片工作模式及时序 ····· 373
15.2.3 烧写控制电路 ····· 375
15.2.4 串口通信电路 ····· 376
15.3 RS-232 通信原理及自定义通信协议 ····· 378
15.3.1 RS-232 串口通信原理 ····· 378
15.3.2 自定义通信协议 ····· 378
15.4 单片机数据接收与烧写程序设计 ····· 379
15.4.1 MPLAB IDE 软件使用 ····· 379
15.4.2 单片机系统初始化 ····· 382
15.4.3 单片机数据收发 ····· 383
15.4.4 EEPROM 芯片烧写与读取 ····· 389
15.4.5 程序主体 ····· 392
15.5 编程器软件界面与数据发送功能设计 ····· 396
15.5.1 工程创建及二进制文件读取与显示 ····· 397
15.5.2 串口通信控件 ····· 400
15.5.3 利用串口通信控件进行数据接收与发送 ····· 403
15.6 编程器性能测试 ····· 410
15.6.1 测试原理 ····· 410
15.6.2 测试程序 ····· 410

第四篇　自定义计算机系统编程实例

第 16 章　自定义 SCL 汇编语言编程实例 ·· 417
16.1 SCL 汇编语言列表 ····· 417
16.2 汇编程序开发流程 ····· 418
16.3 汇编程序开发实例 ····· 421
16.3.1 顺序程序开发实例 ····· 421
16.3.2 分支程序开发实例 ····· 422
16.3.3 循环程序开发实例 ····· 424
16.3.4 子程序开发实例 ····· 425
16.3.5 中断服务程序开发实例 ····· 427
16.3.6 查询与中断方式传送数据 ····· 428
第 17 章　自定义 SCH 高级语言编程实例 ·· 430
17.1 SCH 高级语言单词列表及语法归纳 ····· 430
17.2 SCH 高级程序开发流程 ····· 431
17.3 高级程序开发实例 ····· 433
17.3.1 顺序程序开发实例 ····· 433
17.3.2 分支程序开发实例 ····· 434
17.3.3 循环程序开发实例 ····· 436
17.3.4 子程序开发实例 ····· 437
17.3.5 中断服务程序开发实例 ····· 439
附录 A　自定义计算机系统的 SOPC 实现 · 441
附录 B　随赠光盘文件列表 ····· 446
参考文献 ····· 447

PART 1

第一篇

计算机系统构成及自定义 CPU 芯片设计

对于绝大多数人来说，一谈到计算机、CPU，总带有一种神秘、崇敬的感觉。

其实，计算机系统的早期构想就来源于人们经常见到的普通产品加工厂的样子。厂内加工车间的加工设备、物料货架就相当于CPU中的运算器、寄存器，厂子进出物料、成品的大门就相当于CPU的输入/输出端口，厂内的原材料、半成品、成品库房就相当于计算机系统中的数据存储器，厂内存放着加工各类产品所需的加工工艺及加工流程文件的资料室就相当于计算机系统中的程序存储器，连接厂内加工车间、库房、大门的道路就相当于计算机系统中的总线。从厂子大门进出、道路运输、库存内容、加工车间中的各种设备与货架都是由机关调度室统一控制指挥的，这里的调度室就对应于CPU中的控制器。

如果说加工厂与计算机有什么区别的话，前者加工的对象是有形的产品，而后者加工的对象则是无形的数据。

与工厂中按部就班的产品加工流水线一样，计算机并不具有创造性。加工数据时，控制器只要从程序存储器中取出加工工艺（文件）中所规定的加工操作流程，控制调度相关设备执行即可。由于人的一个动作往往是以秒、分、时来计算的，而计算机完成一个动作则是以纳秒来计算的，两者速度相差十亿倍以上，因而人们必然感觉到它特别神奇。对一个正在流水线上加工的产品，生产人员可根据产品当前的状态来决定下一步应该采取何种加工流程。计算机既能够对数据进行加工处理，又能够对数据状态做出迅速判断，从而能迅速对后续的加工流程进行调整，因而人们感觉到它具有智能。

加工厂按其所拥有的加工设备种类、库房大小、道路和大门宽窄可分为大型企业和小型企业，大型企业一般占用的场地更大，调度室可支配的资源会更多一些。但原则上，它们都应拥有最基本的加工设备，都能加工出相同的产品，只是所付出的时间代价有所区别。以上谈到的调度室可支配资源就对应CPU中的指令系统，指令条数决定了一个CPU的内部结构和复杂程度。

当你将待设计的CPU的指令系统通过硬件描述语言告诉计算机后，计算机中的软件会帮助你自动完成CPU内部数字电路的设计与优化工作。将得到的设计文件瞬间下载至FPGA芯片上，由你自定义的具有自主知识产权的CPU芯片就诞生了。

然后为所设计的CPU配上程序存储器、数据存储器、晶体振荡器等外围芯片后，一个自定义计算机系统的硬件部分就算完成了。

为了使读者能熟练掌握硬件描述语言编程和FPGA芯片开发技术，第3章和第4章以较大的篇幅对这两部分内容进行了详细介绍。

第 1 章

计算机系统构成及工作原理

1.1 从一个产品加工厂谈起

为了能够直观通俗地描述计算机系统构成及工作原理，下面将以一个产品加工厂为例，通过对产品加工厂结构与工作流程的剖析，帮助读者理解计算机系统的构成及其工作原理。

图 1.1 为一个产品加工厂的布局图，从图中可见，厂区内有四座建筑，分别为办公大楼、档案馆、加工车间和库房，以及用于建筑物间相互连通的道路。

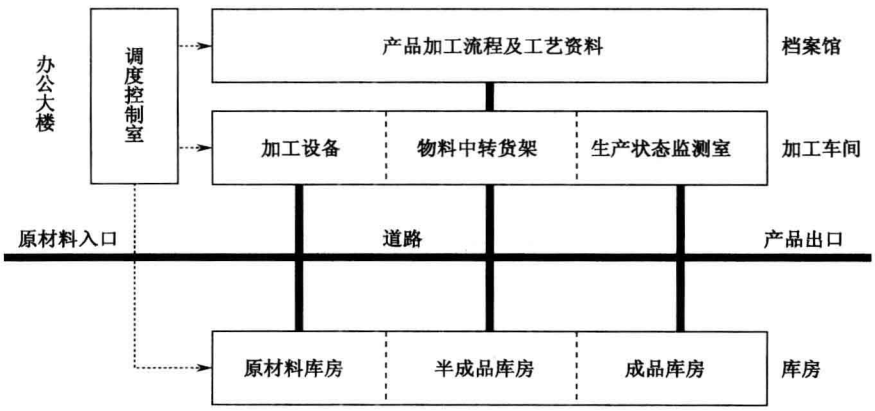

图 1.1 产品加工厂的布局图

产品加工厂的完整运行流程如下：

（1）首先工厂接收订单。如果该产品曾经加工过，则安排调度人员到档案馆取出相关加工资料，以便充分利用和调度工厂内各种储藏、运输和加工资源；如果接收的产品订单是工厂未曾加工过的，则安排技术人员草拟加工流程，并逐步对加工流程的各环节进行反复校验。在满足产品加工质量的前提下，尽量优化加工流程，充分利用厂内资源，以节省加工时间，提升产能。最后，要求将调试完成的产品加工流程及工艺资料送至档案馆保存。

（2）从库房取出原材料，运送至加工车间加工。

（3）在加工过程中，半成品和常用物件可暂时堆放在加工设备附近的物料中转货架上，以节省加工车间与库房之间来回运转所消耗的时间。由于物料中转货架容量较小，因此如果半成品需要占用大量空间，仍须运送至半成品库房存放。

（4）如果加工过程中出现意外情况，相关信息将反映至生产状态监测室，生产状态监控室将情况报告给调度控制室，调度室将安排调度人员从档案馆中调阅相关处置预案。

（5）如需提高产品加工进度，在保障产品质量的前提下，调度人员可调高整个工作流程的节

奏，直至某一环节处于满负荷状态为止。

（6）产品加工完毕，将成品从车间运送至成品库。

1.2 计算机诞生及系统构成

1.2.1 计算机诞生

早在1936年，24岁的英国数学家图灵（1912年—1954年）发表了题为“论数字计算在决断难题中的应用”的论文，首次提出了“图灵机”的概念。图灵机的思想是用机器来模拟人们用纸和笔来完成的数学运算，它为计算机的诞生奠定了理论基础。

1946年，第一台电子计算机ENIAC在美国诞生，ENIAC计算机由18000只电子管组成，体积庞大、耗电量惊人、功能非常有限，但是，ENIAC确实起到了节省人力和节省时间的作用，它为计算机科学技术发展开辟了新纪元。

ENIAC计算机有两个致命的缺陷：一是采用十进制运算；二是没有内部存储器。十进制的缺点是电路结构复杂、可靠性低。由于没有内部存储器，控制机器运算和操作的指令和数据分散存储在电路的许多部件中。因此，每进行一次运算，就要更改一次电路，非常麻烦、费时，这为计算机的发展和应用带来了瓶颈。

针对这两方面缺陷，美国科学家冯·诺依曼提出了采用二进制运算和程序存储的思想。二进制运算降低了电路的复杂度，提高了计算机的可靠性。程序存储的思想是把控制计算机运算和操作的指令、数据存储在独立的存储器中。这样，每进行一次运算，只需更改存储器的内容，使用简单快捷。二进制和程序存储的思想为现代计算机奠定了实际基础。

计算机的核心是处理器。1971年，美国英特尔公司成功研制Intel4004，这是第一块用于通用计算机的微处理器。Intel4004具有2300个晶体管，它的功能相当有限，且工作速度慢，它的诞生在当时并没有引起人们足够的注意。然而正是这块小小的芯片，翻开了微处理器发展的新篇章，微处理器和计算机发展从此日新月异。时至今日，计算机已广泛应用于国防、航天、工业、教育等各个领域，计算机已进入千家万户，极大地造福于人类社会。

1.2.2 计算机系统结构

计算机系统内部构成一般分为冯·诺依曼结构和哈佛结构两种形式，如图1.2（a）和图1.2（b）所示。

这两种结构的主要区别是程序存储器和数据存储器在物理空间上的分布差异。冯·诺依曼结构将程序和数据集中存放在内存条DRAM芯片中，该结构常用于桌面计算机和服务器系统。哈佛结构一般将程序存放在Flash芯片中，将数据存放在DRAM或SRAM芯片中，该结构主要应用于嵌入式系统。

计算机与产品加工厂的系统结构类似。将图1.2（b）与图1.1相比较，程序存储器相当于档案馆；数据存储器相当于库房；数据总线相当于道路；控制器相当于调度控制室；运算器相当于加工设备；寄存器一方面相当于物料中转货架，另一方面保存生产状态信息。

计算机与产品加工厂相比较，最大区别是加工对象不同，前者是传输加工数据，后者是传输加工产品。在计算机中，通常将控制器、运算器、寄存器集成在同一个单元内，称之为计算机的处理器，简称为CPU（Central Processing Unit）。

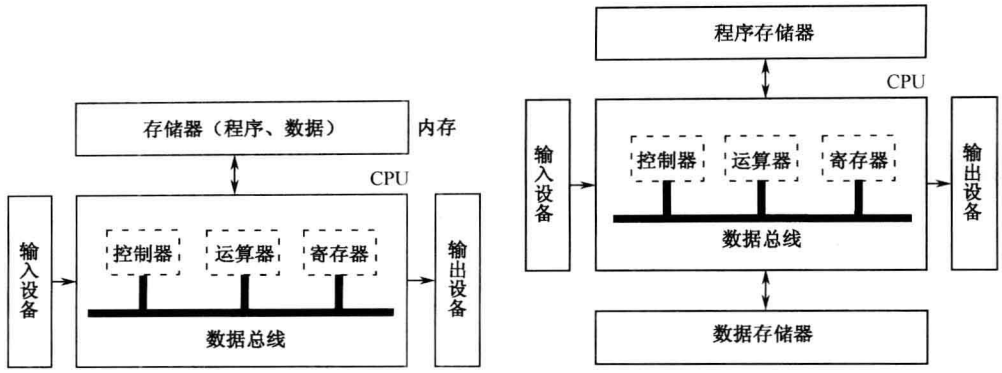

图 1.2 计算机系统内部构成

1.3 计算机开发及运行流程

要使一台计算机正常运转，需要软件和硬件的协调工作。硬件构成计算机的物质基础，犹如产品加工厂的厂房、设备和道路，物质基础在很大程度上决定了系统的性能。软件则类似产品加工厂中的生产加工流程，加工流程有助于改善系统的性能。本节将讲述计算机硬件开发、软件开发，以及计算机运行流程。

1.3.1 计算机软硬件开发流程

1）硬件开发流程

硬件是计算机的物质基础，它决定了计算机的性能。计算机硬件开发步骤包括：设计指令系统、设计 CPU、设计硬件系统，流程如图 1.3 所示。

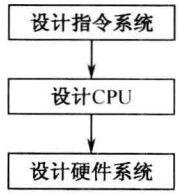

图 1.3 硬件开发流程图

①设计指令系统。设计指令系统是硬件开发的首要步骤。在设计指令系统时，首先要对硬件进行准确的定位，明确硬件需要完成的功能和性能，根据功能和性能需求设计指令系统。如果指令太少，则不能完成需要的功能；如果指令太多，则会大大增加硬件的复杂度。因此在设计指令系统时，应该充分考虑相关情况。在满足硬件功能和性能的前提下，尽量优化指令系统。

②设计 CPU。CPU 是为指令系统服务的。设计 CPU 时，必须认真分析指令运行过程，在此基础上，确定 CPU 的结构和组成模块。如果 CPU 的组成模块太少，则不能顺利地完成指令功能；如果 CPU 的组成模块太多，则会造成了无谓的资源浪费。

③设计硬件系统。CPU 是计算机硬件的核心部件。如果将计算机硬件比喻成人的身体，则 CPU 就是人的大脑，仅仅有 CPU，计算机也不能正常工作，所以在设计完成 CPU 后，须将 CPU 与外围设备连接，形成计算机系统。

在硬件开发流程中，指令系统设计和 CPU 设计是至关重要的。没有好的指令系统就不能有好的 CPU，没有好的 CPU 就不能有好的计算机硬件系统。因此，在计算机硬件开发中必须着重注意指令系统设计和 CPU 设计两个步骤。

2）软件开发流程

在软件开发中，无论是设计大型软件，还是编写只有几行代码的小程序，软件开发步骤都是

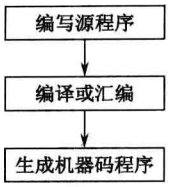

图 1.4　软件开发流程图

基本相同的。软件开发步骤包括：编写源程序、对源程序进行编译或汇编、生成机器码程序，流程如图 1.4 所示。

①编写源程序。利用高级语言或汇编语言编写具有一定功能的源程序，在编辑器中输入源程序，生成相应的源程序文件。

编写源程序时，应根据不同用途选用不同的语言。在科学计算中，一般采用高级语言进行编程，高级语言程序具有程序模块短小、易阅读等优点。在实时处理中，一般采用汇编语言进行编程，汇编语言程序具有代码效率高、控制灵活等特点。

②编译或汇编。源程序利用文字符号来表示程序，这类程序的编写、阅读和修改都比较方便，但是，源程序无法被计算机执行。因此，需要利用编译器或汇编器将源程序转化为机器代码，机器代码是计算机唯一能够识别执行的代码。

③生成机器码程序。在软件开发流程中，编译器是至关重要的，它是设计的重点和难点。可以说，没有编译器，就没有计算机的普及和快速发展。除了编译器，调试工具也是不可缺少的，调试工具能够帮助程序员快速发现软件设计中存在的问题、缩短开发周期。

1.3.2　计算机运行流程

开发完成计算机软硬件后，利用软件调度硬件工作。软件调度硬件工作的主要步骤包括状态监测、取指令和执行指令，如图 1.5 所示。

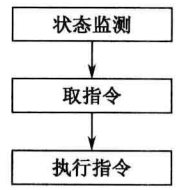

图 1.5　软件调度硬件工作流程图

1）状态监测

计算机定期地监测当前运行状态。如果运行中出现意外情况，则调用应急预案，处理意外情况。如果运行中没有出现意外情况，则进入取指令阶段。

2）取指令

计算机从程序存储器中取出指令，并将指令传送至 CPU。

3）执行指令

CPU 依据接收到的不同指令，调用不同硬件资源，完成相关操作。

第 2 章

自定义指令系统设计

2.1 指令系统概述

2.1.1 指令系统基本概念

计算机能够按照用户预先设计好的步骤有条不紊地运行，是因为用户事先编写了指令代码，并将指令代码存储在存储器中。计算机按照存储器中的指令代码逐条运行。此处所说的指令就是控制计算机各模块协调工作的命令，是一组能实现某个基本操作的二进制代码。指令是程序设计的最小语言单位。

指令系统是所有能对计算机进行操作控制的指令集合，是标志计算机功能是否强大的重要指标，反映了计算机的功能和性能。指令系统越丰富完备，编制程序就越方便灵活。指令系统是根据计算机使用要求来设计的，不同种类的计算机，指令系统所包含的指令数目与格式也不同。

计算机指令一般由操作码和操作数两部分组成，指令格式如图 2.1 所示。

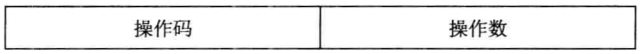

图 2.1 指令格式

① 操作码。操作码说明指令功能，用于规定指令要执行的操作，比如加法运算、移位运算、空操作等。

② 操作数。操作数说明指令运行所需操作数的位置。在操作数部分，可以存放操作数本身，也可以存放操作数的位置。值得注意的是，有的指令不需要操作数，比如空操作指令。

2.1.2 指令系统设计方法

1. 指令分类

维持 CPU 的正常运转需要运算类、通信类和控制类三类操作。

运算类指令负责对数据加工，可细分为算术运算、逻辑运算和移位运算三类。

通信类指令用于实现 CPU 与存储器、CPU 与端口之间的通信。通信类指令是计算机最基本、最重要的操作。在程序编制过程中，通信类指令使用比例最高。图 2.2 是 CPU、存储器和端口之间的通信框图。

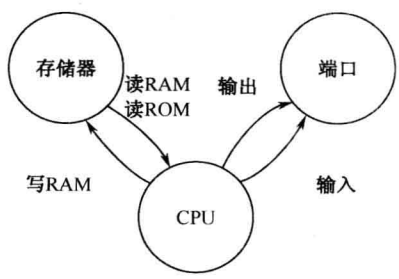

图 2.2 计算机各部件之间的通信框图

控制类指令用于控制程序的运行顺序。如果没有控制指令，程序只能顺序运行，有了控制指令，才能实现循环程序、分支程序、子程序等。常用的控制指令有跳转、子程序调用、

子程序返回和中断返回等。

2. 指令格式与硬件结构

完成同样的指令功能，可以有不同的指令格式；而不同的指令格式又对应着不同的硬件结构。下面将针对指令格式和硬件结构进行讲解。常用的硬件结构有基本机型、栈式机型、累加器机型和通用寄存器机型。

1）基本机型

图 2.3 所示是基本机模型，模型中的程序计数器 PC 用于保存指令地址。CPU 内部不包含通用寄存器，因此 CPU 运行所需的数据全部存放在存储器中。运行指令时，CPU 首先将数据从存储器中取出，然后将运算结果存回到存储器中。下面以加法指令来说明。

假如加法指令格式为“ADDD addr3, addr2, addr1”，加法指令中，addr1 和 addr2 表示两个源操作数在存储器中的位置，addr3 表示运算结果在存储器中的位置。基本机执行该指令的示意图如图 2.4 所示。从图 2.4 中看出，基本机在运行指令时，需要频繁地访问存储器，从而严重影响运行速度。在实用的计算机中，几乎不使用基本机型。

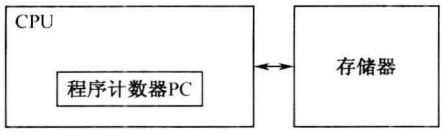

图 2.3　基本机模型

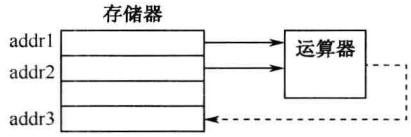

图 2.4　基本机型加运算示意图

2）栈式机型

图 2.5 所示是栈式机模型，栈式机模型与基本机模型的相同之处是 CPU 内部包含程序计数器 PC 用于指向指令地址，不包含通用寄存器。与基本机不同的是，栈式机在存储器中开辟出一块专用的存储空间，这块存储空间称为“栈”，栈地址由寄存器 BP 和寄存器 SP 组成，其中 BP 用于指向堆栈的段基地址，SP 用于指向栈顶位置。栈式机运行所需的数据全部存放在堆栈中。运行指令时，CPU 首先将数据从堆栈中取出，然后将运算结果存回到堆栈中。同样以加法指令进行说明。

假如加法指令格式为“ADDD”，在加法指令中，两个源操作数存放在 SP 和 SP+1 所指定的堆栈位置中，运算结果被回存至 SP+1 所指定的堆栈位置。栈式机执行该指令的示意图如图 2.6 所示。由于栈式机内部没有通用寄存器，因此，在运行指令时，需要频繁地访问存储器，影响运行速度。在实际应用中，栈式机类型也比较少见。

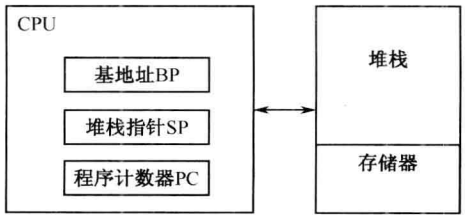

图 2.5　栈式机模型

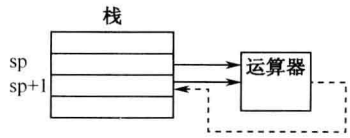

图 2.6　栈式机型加运算示意图

3）累加器机型

图 2.7 所示是累加器机模型，该模型中，CPU 内部包含程序计数器 PC、一个通用寄存器即累加器。累加器机型在运行过程中可以将其中一个操作数存放在累加器中，其他所需操作数存放在存储器中，运算结果可存放至累加器，也可存放至存储器。同样以加法指令来说明。