



附赠1CD

- 程序源代码
- 电子课件PPT

# Cortex-A9

## 多核嵌入式系统设计

廖义奎 编著

- ☑ 多核处理器基础知识
- ☑ 多核处理器硬件设计
- ☑ 嵌入式系统软件设计基础

- ☑ Android 程序设计
- ☑ iOS 程序设计
- ☑ Hybrid App 开发



中国电力出版社  
CHINA ELECTRIC POWER PRESS

# Cortex-A9

## 多核嵌入式系统设计

廖义奎 编著



中国电力出版社  
CHINA ELECTRIC POWER PRESS

## 内 容 摘 要

本书分为多核处理器基础、多核处理器硬件设计、嵌入式系统软件设计基础、Android 程序设计、iOS 程序设计以及 Hybrid App 开发六大部分。本书共 19 章, 包括多核处理器基础、ARM 多核处理器体系结构、多核嵌入式系统硬件结构、存储器电路、音视频及触摸屏接口、通信接口、Bootloader 程序设计、Linux 软件开发平台及软件配置、Linux 驱动程序设计、Linux 及 Android 操作系统的编译、Android 程序设计基础、Android 图形用户界面设计、Activity、Intent 和 Service 系统结构设计、iOS 程序设计基础、Objective-C 编程语言、iOS 视图及视图控制器、Hybrid App 开发基础、PhoneGap 移动开发、jQuery Mobile 移动开发。

本书适合于嵌入式开发人员作为开发参考资料, 也适合于高校师生作为嵌入式系统、移动开发、物联网应用开发等课程的教材和教学参考书。

## 图书在版编目 (CIP) 数据

Cortex-A9 多核嵌入式系统设计/廖义奎编著. —北京: 中国电力出版社, 2014.9

ISBN 978-7-5123-6239-0

I. ①C… II. ①廖… III. ①微处理器—系统设计  
IV. ①TP332

中国版本图书馆 CIP 数据核字 (2014) 第 168731 号

中国电力出版社出版、发行

(北京市东城区北京站西街 19 号 100005 <http://www.cepp.sgcc.com.cn>)

北京丰源印刷厂印刷

各地新华书店经售

\*

2014 年 9 月第一版 2014 年 9 月北京第一次印刷

787 毫米×1092 毫米 16 开本 28.75 印张 770 千字

印数 0001—3000 册 定价 59.80 元 (1CD)

## 敬告读者

本书封底贴有防伪标签, 刮开涂层可查询真伪

本书如有印装质量问题, 我社发行部负责退换

版权专有 翻印必究

## 前 言

目前,嵌入式系统已经进入了一个全新的阶段,特别是在嵌入式系统的处理器方面,较之前具有了特别大的变化,已经从单核处理器进入了多核处理器时代。在嵌入式系统应用之中,从手机到平板电脑,从安卓机顶盒到智能电视机,从汽车多媒体中心到工业控制 HMI (Human Machine Interface, 人机交互界面) 系统,从智能路由器到物联网控制中心,无一例外地把多核处理器的应用作为产品的核心,是这些产品最大的特点和亮点。

多核处理器就是基板上集成有多个单核处理器,与单核处理器相比,多核处理器在体系结构、软件、功耗和安全性设计等方面面临着巨大的挑战,但也蕴含着巨大的潜能。多核处理器的结构主要分为同构和异构两类,同构是指内部核的结构是相同的,而异构是指内部的核结构是不同的。

多核处理器代表了计算技术的一次创新,在一个处理器中放入两个或多个功能强大的计算内核产生了一个重大的可能性,多核处理器能提供比单核处理器更好的性能和效率。为多任务应用的工作模式提供了一个很好的运行平台。

### 本书结构

全书共 19 章。主要包括:

#### (1) 多核处理器基础知识。

一方面详细介绍了嵌入式系统的基础知识,以及目前多核处理器的发展情况;另一方面详细介绍的 ARM 处理器多核体系结构,特别是深入分析了 Cortex-A9 内核的内部结构。

#### (2) 多核处理器硬件设计。

多核处理器作为硬件设计的中心,结合 Cortex-A9 多核处理器的特点,介绍了多核嵌入式系统硬件整体结构。另外还重点介绍了嵌入式系统的存储器电路设计、人机交互接口电路设计以及通信接口设计。

#### (3) 嵌入式系统软件设计基础。

Bootloader 程序部分主要介绍了 U-boot 的特点、功能与结构。Linux 软件开发平台及软件配置部分主要介绍了 VMware 虚拟机下的 Ubuntu 操作系统的安装与配置,以及 Linux 程序设计的基础,重点介绍的 makefile 文件的编写规则;同时还介绍了 ARM 的交叉编译器的配置方法。Linux 驱动程序设计部分介绍了 Linux 设备驱动程序的模块化结构,以及 Linux 驱动程序的编写、编译与加载方法。Linux 及 Android 操作系统的编译部分主要介绍了 Android 系统架构、Ubuntu 下的 Android 编译环境配置,以及重点介绍了如何从源代码构建 Android 系统。

#### (4) Android 程序设计。

Android 程序设计基础部分主要讲解了 Android 程序设计的开发平台,以及了解 Android 项目的文件目录结构和文件基本内容。Android 图形用户界面设计部分主要讲解了 Android 的界面布局以及 Android 的界面控件。Android 系统结构设计部分主要讲解了 Android 应用程序组件,包括 Activities、Services、Content Providers、Intents、Broadcast Receivers、Widgets、Notifications 等,以及 Android 的网络程序设计。

### (5) iOS 程序设计。

iOS 程序设计基础部分主要介绍了 iOS 开发环境的配置, 以及简单 iOS 项目的文件结构。Objective-C 编程语言部分主要介绍了 Objective-C 的特点, 以及与 C++ 之间的区别, 还介绍了 Objective-C 的内存管理机制。iOS 视图及视图控制器部分主要介绍了 iOS 的界面设计基础、iOS 常用控件以及 iOS 的视图控制器。

### (6) Hybrid App 开发。

Hybrid App 开发基础部分主要介绍了移动 Web 和 Hybrid App 开发的特点、开发框架、HTML5 以及 Sencha Touch 等内容。PhoneGap 移动开发部分主要介绍了 PhoneGap 的特点、PhoneGap 开发实例、PhoneGap 插件开发以及常用的 PhoneGap API。jQuery Mobile 移动开发部分介绍了 jQuery Mobile 的特点、jQuery 应用基础、jQuery 选择器、jQuery 事件、jQuery Mobile 应用实例。

### 读者对象

本书的读者需要具有一定的 Java、单片机以及电子线路设计基础, 适合于从事 ARM 嵌入式开发的工程开发人员作为参考资料, 更适合于以前从事 8 位、16 位 MCU 开发, 而又迫切需要跨越到 32 位 MCU 平台的工程开发人员。也适合于高校师生作为课程设计、毕业设计以及电子设计竞赛的培训和指导教材, 以及作为嵌入式系统、移动开发、物联网应用开发等课程的教材和教学参考书。

### 光盘使用

本书 ARM 程序的编译环境都是 GCC, Obtain\_Studio 集成开发环境(IDE)软件已带有 GCC, 并自动动态配置 GCC 的运行环境, 因此可以在 Obtain\_Studio 中直接编译本书的程序。

配套光盘中包括了本书各章的课件(PPT), 方便高校教师上课教学使用, 也可以作为读者学习本书的配套资料。还包括了部分章节的程序代码, 读者可以直接拷贝下来使用, 并仿照这些程序源代码去快速开发新的应用程序。

### 联系作者

如果读者在使用本书时遇到相关的技术问题, 或者对本书介绍的 ARM 开发板感兴趣或有疑问, 可以通过电子邮件与作者联系(javawebstudio@163.com), 作者将尽最大努力与广大读者一起解决学习和开发过程中遇到的问题, 共同进步。

### 致谢

在本书的编写过程中, 陆才志、苏宇、梁创英、许金、黄子芸、玉黄荣、王继平、韦运忠、徐卫怡、蓝艺峥、尤勇健、谢冬英、苏金秀分别审阅了本书全部或部分章节, 在此表示衷心的感谢。

本书在编写过程中参考了大量的文献资料, 一些资料来自互联网和一些非正式出版物, 书中的参考文献无法一一列举, 在此对原作者表示诚挚的谢意。

限于作者水平, 并且编写时间比较仓促, 书中难免存在错误和疏漏之处, 敬请读者批评指正。

编著者

# 目 录

## 前言

## 第1章 多核处理器基础

1

|                        |    |
|------------------------|----|
| 1.1 嵌入式系统概述            | 1  |
| 1.2 嵌入式系统的发展           | 2  |
| 1.2.1 从单片机到嵌入式系统       | 2  |
| 1.2.2 从芯片级设计到系统级设计     | 3  |
| 1.2.3 从单处理器设计到多处理器设计   | 3  |
| 1.3 多核处理器              | 3  |
| 1.3.1 单核处理器的局限性        | 3  |
| 1.3.2 处理器执行指令的效率       | 4  |
| 1.3.3 CPU的主频与功耗的关系     | 4  |
| 1.3.4 从单核处理器到多核处理器设计   | 6  |
| 1.3.5 多任务应用的工作模式       | 6  |
| 1.4 嵌入式系统软件            | 7  |
| 1.4.1 嵌入式软件设计的特点       | 7  |
| 1.4.2 嵌入式操作系统          | 8  |
| 1.5 嵌入式系统应用            | 12 |
| 1.6 ARM 处理器            | 14 |
| 1.6.1 ARM 的历史          | 14 |
| 1.6.2 ARM 的发展          | 15 |
| 1.7 ARM 处理器分类          | 15 |
| 1.7.1 ARM 处理器架构        | 15 |
| 1.7.2 ARM Cortex 处理器   | 16 |
| 1.7.3 Cortex-A8 处理器    | 17 |
| 1.7.4 Cortex-A5 处理器    | 17 |
| 1.7.5 Cortex-A7 处理器    | 18 |
| 1.7.6 Cortex-A9 处理器    | 19 |
| 1.7.7 Cortex-A12 处理器   | 19 |
| 1.7.8 Cortex-A15 处理器   | 20 |
| 1.8 ARMv8 架构           | 22 |
| 1.8.1 ARMv8 架构介绍       | 22 |
| 1.8.2 Cortex-A50 系列处理器 | 23 |

|                 |    |
|-----------------|----|
| 1.8.3 苹果 A7 处理器 | 24 |
| 1.9 多核处理器时代     | 25 |

## 第2章 ARM 多核处理器体系结构

28

|                              |    |
|------------------------------|----|
| 2.1 微处理器的体系结构基础              | 28 |
| 2.2 ARM 处理器体系结构              | 32 |
| 2.3 Cortex-A9 处理器体系结构        | 37 |
| 2.3.1 Cortex-A9 内核结构         | 37 |
| 2.3.2 Cortex-A9 单核处理器        | 38 |
| 2.3.3 Cortex-A9 MPCore 多核处理器 | 39 |
| 2.3.4 Cortex-A9 主要功能         | 40 |

## 第3章 多核嵌入式系统硬件结构

42

|                        |    |
|------------------------|----|
| 3.1 嵌入式系统硬件结构          | 42 |
| 3.2 Exynos4412 开发板介绍   | 43 |
| 3.2.1 Exynos4412 处理器介绍 | 43 |
| 3.2.2 Exynos4412 开发板   | 44 |
| 3.3 OMAP4460 开发平台      | 45 |
| 3.3.1 OMAP4460 双核处理器介绍 | 45 |
| 3.3.2 OMAP4460 开发板介绍   | 48 |
| 3.3.3 OMAP4460 开发板结构   | 49 |
| 3.3.4 JTAG 接口          | 50 |
| 3.4 电源管理               | 52 |
| 3.4.1 电源管理概述           | 52 |
| 3.4.2 TWL6030 电源管理 IC  | 54 |
| 3.4.3 OMAP4460 开发板电源管理 | 56 |

## 第4章 存储器电路

57

|                       |    |
|-----------------------|----|
| 4.1 OMAP4460 存储空间     | 57 |
| 4.1.1 OMAP4460 存储空间概要 | 57 |
| 4.1.2 OMAP4460 存储映射   | 58 |
| 4.2 RAM 存储器           | 60 |
| 4.2.1 RAM 存储器介绍       | 60 |
| 4.2.2 SDRAM 工作原理      | 62 |
| 4.2.3 DDR SDRAM 分类    | 65 |
| 4.2.4 DDR SDRAM 工作原理  | 69 |
| 4.2.5 DDR3 新增特点       | 71 |
| 4.3 ROM 存储器           | 72 |
| 4.3.1 ROM 存储器类型       | 72 |
| 4.3.2 Flash 存储器类型     | 75 |
| 4.4 存储卡接口电路设计         | 80 |

|                 |    |
|-----------------|----|
| 4.4.1 存储卡分类     | 80 |
| 4.4.2 存储卡接口电路设计 | 81 |

## 第5章 音视频及触摸屏接口

83

|                        |     |
|------------------------|-----|
| 5.1 显示屏                | 83  |
| 5.1.1 液晶显示技术的发展        | 83  |
| 5.1.2 TFT 屏            | 84  |
| 5.1.3 IPS 硬屏           | 85  |
| 5.1.4 其他液晶屏            | 88  |
| 5.1.5 OLED 屏           | 89  |
| 5.1.6 AMOLED 屏幕        | 91  |
| 5.1.7 Super AMOLED 屏幕  | 91  |
| 5.1.8 等离子显示屏           | 92  |
| 5.1.9 OMAP4460 显示屏接口设计 | 93  |
| 5.2 触摸屏                | 95  |
| 5.2.1 触摸屏介绍            | 95  |
| 5.2.2 电阻触摸屏            | 97  |
| 5.2.3 电容触摸屏            | 98  |
| 5.3 HDMI 接口            | 102 |
| 5.3.1 HDMI 接口介绍        | 102 |
| 5.3.2 HDMI 接口类型        | 103 |
| 5.3.3 HDMI 接口电路        | 104 |
| 5.4 摄像头接口              | 106 |
| 5.4.1 摄像头种类            | 106 |
| 5.4.2 CMOS 摄像头接口电路     | 108 |
| 5.5 音频接口               | 109 |
| 5.5.1 音频接口介绍           | 109 |
| 5.5.2 音频接口电路           | 110 |

## 第6章 通信接口

111

|                        |     |
|------------------------|-----|
| 6.1 无线通信               | 111 |
| 6.1.1 WLAN 通信          | 111 |
| 6.1.2 Wi-Fi 通信         | 112 |
| 6.1.3 蓝牙通信             | 112 |
| 6.1.4 嵌入式系统中的无线通信解决方案  | 113 |
| 6.2 以太网通信              | 115 |
| 6.2.1 以太网接口介绍          | 115 |
| 6.2.2 LAN9514 以太网接口的设计 | 116 |
| 6.2.3 LAN8720 以太网接口的设计 | 117 |
| 6.2.4 DM9621 以太网接口的设计  | 118 |
| 6.3 其他通信接口             | 119 |

|              |     |
|--------------|-----|
| 6.3.1 其他无线通信 | 119 |
| 6.3.2 其他有线接口 | 120 |
| 6.4 物联网      | 121 |

## 第7章 Bootloader 程序设计

123

|                     |     |
|---------------------|-----|
| 7.1 Bootloader 程序   | 123 |
| 7.1.1 Bootloader 介绍 | 123 |
| 7.1.2 U-Boot 介绍     | 123 |
| 7.1.3 Vivi 介绍       | 124 |
| 7.2 U-Boot 引导程序分析   | 124 |
| 7.2.1 U-Boot 功能与结构  | 124 |
| 7.2.2 U-Boot 命令     | 125 |
| 7.2.3 U-Boot 工作流程   | 131 |

## 第8章 Linux 软件开发平台及软件配置

133

|                               |     |
|-------------------------------|-----|
| 8.1 Linux 体系结构                | 133 |
| 8.1.1 Linux 介绍                | 133 |
| 8.1.2 Linux 系统结构              | 134 |
| 8.1.3 Linux 内核结构              | 135 |
| 8.2 Linux 开发基础                | 137 |
| 8.2.1 在 Windows 上安装虚拟机        | 137 |
| 8.2.2 在虚拟机上安装 Linux           | 139 |
| 8.3 Linux 嵌入式系统开发平台配置         | 153 |
| 8.3.1 GCC 编译开发环境              | 153 |
| 8.3.2 在 Ubuntu 上安装 ARM 交叉编译环境 | 154 |
| 8.4 Linux 程序设计                | 155 |
| 8.4.1 make 常用命令               | 155 |
| 8.4.2 makefile 文件规则           | 157 |

## 第9章 Linux 驱动程序设计

162

|                                   |     |
|-----------------------------------|-----|
| 9.1 Linux 设备驱动程序                  | 162 |
| 9.1.1 Linux 设备分类                  | 162 |
| 9.1.2 Linux 驱动程序的模块化              | 163 |
| 9.1.3 Linux 设备驱动程序结构              | 165 |
| 9.1.4 Linux 驱动程序常用的数据结构           | 166 |
| 9.1.5 Linux2.6 与 2.4 内核驱动程序的区别    | 171 |
| 9.2 Linux 经典 Hello World 驱动程序     | 174 |
| 9.2.1 驱动程序代码                      | 174 |
| 9.2.2 module_init 与 module_exit 宏 | 174 |
| 9.2.3 printk 函数                   | 177 |
| 9.2.4 驱动程序的编译与加载                  | 177 |

|                           |     |
|---------------------------|-----|
| 9.3 Linux 字符设备驱动程序实例..... | 179 |
| 9.3.1 字符设备驱动程序.....       | 179 |
| 9.3.2 字符设备驱动测试.....       | 185 |
| 9.3.3 字符设备驱动的加载与测试.....   | 186 |

## 第10章 Linux 及 Android 操作系统的编译

187

|                          |     |
|--------------------------|-----|
| 10.1 Android 系统架构.....   | 187 |
| 10.2 Ubuntu 编译环境配置.....  | 190 |
| 10.2.1 主机环境.....         | 190 |
| 10.2.2 初步设置.....         | 192 |
| 10.3 快速入门与预编译二进制文件.....  | 193 |
| 10.4 从源代码构建.....         | 194 |
| 10.4.1 编译平台设置.....       | 194 |
| 10.4.2 创建 AOSP 主分支.....  | 195 |
| 10.4.3 编译 Linaro 内核..... | 197 |

## 第11章 Android 程序设计基础

200

|                                      |     |
|--------------------------------------|-----|
| 11.1 Eclipse 开发环境配置.....             | 200 |
| 11.1.1 环境搭建.....                     | 200 |
| 11.1.2 使用 Eclipse 开发 Android 程序..... | 202 |
| 11.2 Obtain_Studio 使用方法.....         | 203 |
| 11.2.1 Obtain_Studio 介绍.....         | 203 |
| 11.2.2 创建 Android 项目.....            | 205 |
| 11.2.3 Obtain_Studio 集成开发系统常用技巧..... | 211 |
| 11.3 Android 常用命令.....               | 214 |
| 11.3.1 ADB 命令.....                   | 214 |
| 11.3.2 Shell 命令.....                 | 217 |
| 11.4 Android 项目.....                 | 219 |
| 11.4.1 Android 项目结构.....             | 219 |
| 11.4.2 Android 项目文件.....             | 220 |
| 11.4.3 Android 项目编译配置文件.....         | 222 |
| 11.4.4 Android 项目全局配置文件.....         | 224 |
| 11.5 Android 资源.....                 | 227 |
| 11.5.1 Android 资源类型.....             | 227 |
| 11.5.2 字符串资源文件.....                  | 228 |
| 11.6 Android 应用程序签名.....             | 234 |
| 11.6.1 关于 Android 应用程序签名.....        | 234 |
| 11.6.2 ProGuard 工具.....              | 235 |
| 11.6.3 keytool 程序.....               | 236 |
| 11.6.4 应用程序签名.....                   | 237 |

|         |                         |     |
|---------|-------------------------|-----|
| 12.1    | Android 界面布局            | 239 |
| 12.1.1  | 界面布局文件                  | 239 |
| 12.1.2  | 界面布局类型                  | 240 |
| 12.2    | Android 常用控件            | 243 |
| 12.2.1  | Button 控件               | 244 |
| 12.2.2  | CheckBox 控件             | 245 |
| 12.2.3  | EditText 控件             | 246 |
| 12.2.4  | ImageButton 控件          | 248 |
| 12.2.5  | ImageView 控件            | 249 |
| 12.2.6  | ListView 控件             | 249 |
| 12.2.7  | ProgressBar 控件          | 251 |
| 12.2.8  | RadioButton 控件          | 252 |
| 12.2.9  | SeekBar 控件              | 253 |
| 12.2.10 | Spinner 控件              | 254 |
| 12.2.11 | TabHost/TabWidget       | 255 |
| 12.2.12 | Gallery 与 ImageSwitcher | 256 |
| 12.2.13 | GridView                | 260 |
| 12.3    | 自定义按钮背景                 | 263 |

|        |                    |     |
|--------|--------------------|-----|
| 13.1   | Java 语言编程基础        | 267 |
| 13.1.1 | Java 语言的特点         | 267 |
| 13.1.2 | Java 与 C++的区别      | 268 |
| 13.2   | Android 应用程序组件     | 270 |
| 13.3   | Activity           | 270 |
| 13.3.1 | Activity 类         | 270 |
| 13.3.2 | Android 事件侦听器      | 274 |
| 13.4   | Intent             | 278 |
| 13.4.1 | Intent 介绍          | 278 |
| 13.4.2 | 常见 Intent 应用       | 279 |
| 13.4.3 | 两个 activity 之间切换实例 | 281 |
| 13.5   | Service            | 285 |
| 13.5.1 | Service 介绍         | 285 |
| 13.5.2 | Service 启动流程       | 285 |
| 13.5.3 | Service 应用实例       | 285 |
| 13.6   | 对话框                | 286 |
| 13.6.1 | Toast 消息提示框        | 286 |
| 13.6.2 | 对话框                | 289 |
| 13.7   | Android 程序生命周期     | 290 |

|                          |     |
|--------------------------|-----|
| 13.8 广播接收器               | 292 |
| 13.9 Android 网络编程        | 293 |
| 13.9.1 Android HTTP 通信   | 294 |
| 13.9.2 Android Socket 通信 | 297 |

## 第14章 iOS 程序设计基础

300

|                              |     |
|------------------------------|-----|
| 14.1 iOS 开发环境搭建              | 300 |
| 14.1.1 iOS 介绍                | 300 |
| 14.1.2 iOS 开发环境搭建            | 300 |
| 14.2 iOS 入门实例                | 302 |
| 14.2.1 创建 iOS 项目             | 302 |
| 14.2.2 编辑 main.storyboard 文件 | 304 |
| 14.2.3 程序代码分析                | 307 |
| 14.2.4 main 函数及程序启动过程        | 308 |
| 14.2.5 UIResponder 类         | 309 |

## 第15章 Objective-C 编程语言

311

|                                  |     |
|----------------------------------|-----|
| 15.1 Objective-C 概述              | 311 |
| 15.1.1 Objective-C 介绍            | 311 |
| 15.1.2 Objective-C 特点            | 311 |
| 15.1.3 Objective-C 和 C++/java 比较 | 312 |
| 15.1.4 运行 Objective-C            | 314 |
| 15.2 类和对象                        | 316 |
| 15.2.1 类的声明                      | 316 |
| 15.2.2 类的实例化                     | 317 |
| 15.2.3 类的继承                      | 318 |
| 15.2.4 Objective-C 分类和扩展         | 319 |
| 15.3 内存管理                        | 321 |
| 15.3.1 内存管理基础                    | 321 |
| 15.3.2 ARC 内存管理                  | 323 |

## 第16章 iOS 视图及视图控制器

326

|                          |     |
|--------------------------|-----|
| 16.1 iOS 的窗口与视图介绍        | 326 |
| 16.1.1 iOS 窗口            | 326 |
| 16.1.2 iOS 视图            | 327 |
| 16.2 iOS 控件              | 328 |
| 16.2.1 iOS 基本控件          | 328 |
| 16.2.2 UIToolBar 工具条使用示例 | 331 |
| 16.3 iOS 视图控制器           | 333 |
| 16.4 视图控制器应用实例           | 335 |

|        |                  |     |
|--------|------------------|-----|
| 17.1   | Hybrid App 开发介绍  | 348 |
| 17.1.1 | 移动开发类型           | 348 |
| 17.1.2 | Native App       | 348 |
| 17.1.3 | Web App          | 349 |
| 17.1.4 | Hybrid App       | 350 |
| 17.1.5 | 常见移动 Web 开发框架    | 350 |
| 17.1.6 | 常见 Hybrid App 平台 | 351 |
| 17.2   | HTML5            | 353 |
| 17.2.1 | HTML5 介绍         | 353 |
| 17.2.2 | 语义特性             | 354 |
| 17.2.3 | 本地存储             | 359 |
| 17.2.4 | WebSockets       | 365 |
| 17.3   | Sencha Touch     | 365 |
| 17.3.1 | Sencha Touch 概要  | 365 |
| 17.3.2 | Sencha Touch 应用  | 366 |

|         |                      |     |
|---------|----------------------|-----|
| 18.1    | PhoneGap 概述          | 370 |
| 18.1.1  | PhoneGap 介绍          | 370 |
| 18.1.2  | PhoneGap 基本知识        | 370 |
| 18.1.3  | PhoneGap 的特点         | 370 |
| 18.2    | PhoneGap 开发实例        | 371 |
| 18.2.1  | 配置 PhoneGap          | 371 |
| 18.2.2  | PhoneGap 自带例子        | 372 |
| 18.2.3  | 创建一个新的实例             | 376 |
| 18.2.4  | PhoneGap 插件开发        | 378 |
| 18.3    | PhoneGap API         | 381 |
| 18.3.1  | Accelerometer        | 382 |
| 18.3.2  | Camera               | 383 |
| 18.3.3  | Compass              | 385 |
| 18.3.4  | Contacts             | 386 |
| 18.3.5  | Device               | 387 |
| 18.3.6  | Events               | 387 |
| 18.3.7  | File                 | 389 |
| 18.3.8  | GeoLocation          | 397 |
| 18.3.9  | Media                | 398 |
| 18.3.10 | Notification         | 400 |
| 18.3.11 | Storage              | 401 |
| 18.4    | PhoneGap 开发 iOS 应用程序 | 402 |

|        |                    |     |
|--------|--------------------|-----|
| 19.1   | jQuery Mobile 概要   | 404 |
| 19.1.1 | jQuery 介绍          | 404 |
| 19.1.2 | jQuery Mobile 介绍   | 405 |
| 19.2   | jQuery 应用          | 405 |
| 19.2.1 | jQuery 应用基础        | 405 |
| 19.2.2 | jQuery 选择器         | 407 |
| 19.2.3 | jQuery 事件          | 409 |
| 19.3   | jQuery.ajax        | 416 |
| 19.3.1 | jQuery.ajax 介绍     | 416 |
| 19.3.2 | JSON               | 417 |
| 19.3.3 | jQuery.ajax 应用     | 419 |
| 19.4   | jQuery Mobile 应用   | 423 |
| 19.4.1 | jQuery Mobile 页面结构 | 423 |
| 19.4.2 | jQuery Mobile 页面链接 | 424 |
| 19.4.3 | jQuery Mobile 内容格式 | 427 |
| 19.4.4 | 对话框                | 428 |
| 19.4.5 | jQuery Mobile 导航   | 429 |
| 19.4.6 | jQuery Mobile 工具栏  | 431 |
| 19.4.7 | jQuery Mobile 按钮   | 435 |
| 19.4.8 | jQuery Mobile 主题化  | 436 |
| 19.4.9 | jQuery Mobile 列表视图 | 440 |
|        | 参考文献               | 445 |

# 多核处理器基础

## 1.1 嵌入式系统概述

### 1. 单片机

嵌入式系统的出现最初是基于单片机。单片微型计算机简称单片机，是典型的嵌入式微控制器（Microcontroller Unit），常用英文字母的缩写 MCU 表示单片机，单片机又称单片微控制器，它不是完成某一个逻辑功能的芯片，而是把一个计算机系统集成到一个芯片上。

单片机是一种集成电路芯片，是采用超大规模集成电路技术把具有数据处理能力的中央处理器 CPU、随机存储器 RAM、只读存储器 ROM、多种 I/O 口和中断系统、定时器/计时器等（可能还包括显示驱动电路、脉宽调制电路、模拟多路转换器、A/D 转换器等电路）集成到一块硅片上构成的一个小而完善的微型计算机系统，广泛应用于工业控制领域。

### 2. 嵌入式系统起源

第一种来源：简单的单片机已经不能满足复杂的和智能的应用系统，需要加上操作系统实现多任务；需要更快的运算速度、更加大的内存容量；需要更强的网络功能。从一个单片机主板变成了复杂的可嵌入到其他应用之中的系统，也就是嵌入式系统。

第二种来源：微型机表现出的智能化水平引起了控制专业人士的兴趣，要求将微型机嵌入到一个对象体系中，实现对象体系的智能化控制。这样一来，计算机便失去了原来的形态与通用的计算机功能。为了区别原有的通用计算机系统，把嵌入到对象体系中、实现对象体系智能化控制的计算机，称为嵌入式计算机系统。

### 3. 嵌入式系统的定义

据 IEEE（国际电动机工程师协会）所述，嵌入式系统即“控制、监视或者辅助装置、机器和设备运行的装置”（原文为 devices used to control, monitor, or assist the operation of equipment, machinery or plants）。这主要是从应用上加以定义的，从中可以看出嵌入式系统是软件和硬件的综合体，还可以涵盖机械等附属装置。

不过，IEEE 的定义并不能充分体现出嵌入式系统的精髓，目前国内一个普遍被认同的定义是：以应用为中心，以计算机技术为基础，软件硬件可裁剪，适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统。

一般认为嵌入式系统（Embedded System）就是可以嵌入到其他系统中的微处理器应用系统。在构成上，嵌入式系统以微处理器及软件为核心部件，这两者缺一不可。在特征上，嵌入式系统可以方便、灵活地嵌入到其他应用系统，即具有很强的可嵌入性。嵌入式系统本身是一个可独立执行的系统，但更重要的是它可作为一个部件嵌入到其他应用系统中。

### 4. 嵌入式系统特点

由于嵌入式系统的内容很广，在含义上与传统的单片机系统和计算机系统有很多重叠部分。

为了方便区分,在实际应用中通常给嵌入式系统加了一些不成文的限定:

(1) 嵌入式系统的微处理器通常是由 32 位以及 32 位以上的 RISC 处理器组成的,例如 ARM、MIPS 等。

(2) 嵌入式系统的软件系统通常是以嵌入式操作系统为核心,外加用户应用程序。

(3) 嵌入式系统在特征上具有明显的可嵌入性。

从狭义上说,嵌入式系统都应该具备上述三个特征。从广义上说,只要具备上述三个特征中的部分特征,也可以称为嵌入式系统。

#### 5. 常见的数字芯片

(1) CPU (Central Processing Unit) 中央处理器。

(2) MCU (Micro Control Unit) 微控制器。

(3) MPU (Micro Processing Unit) 微处理器。

(4) DSC (Digital signal controller) 数字信号控制器。

(5) DSP (Digital Signal Processing) 数字信号处理。

(6) SOC (System on a Chip) 系统级芯片。

(7) CPLD (Complex Programmable Logic Device) 复杂可编程逻辑器件。

(8) FPGA (Field Programmable Gate Array) 现场可编程门阵列。

(9) SOPC (System-on-a-Programmable-Chip) 可编程片上系统。

这些数字芯片都可以用于构建嵌入式系统。嵌入式系统可分为以单片机 (MCU) 为核心组成的嵌入式单片机系统,以工业计算机板 (x86 处理器) 为核心组成的嵌入式计算机系统,以 DSP 为核心组成的嵌入式数字信号处理器系统,以 FPGA 及 CPU (SOPC) 为核心组成的嵌入式 SOPC 系统等。目前,在手机和平板电脑上,用得最多的还是以 MPU 为核心的嵌入式系统。

## 1.2 嵌入式系统的发展

如今手机向智能化发展,电视向网络化发展,这些都得益于嵌入式系统日新月异的进步和发展。

### 1.2.1 从单片机到嵌入式系统

20 世纪 80 年单片机开始进入人们的视线,它使用简单方便,功能强大。在随后的二三十年里,已经渗透到各个领域,给人们的生活和工业生产带来极大方便。从信号采集、处理到传输都能由单片机来完成。

但是,随着网络时代的来临,许多电子设备需要联网和更智能化、更强的计算能力,比如音频、视频的数据采集、处理和传输,丰富的图形界面等。单片机越来越不能满足应用对象的需求,开发工作也变得越来越复杂且庞大。

随着微电子技术的进步,芯片的制造成本大大降低,而功能却大大增强,16 位和 32 位的嵌入式微处理器逐渐成为嵌入式系统开发设计的主流。不过,只有嵌入式微处理器是不够的,OEM (原始设备制造商) 还需要有一个运行于嵌入式微处理器上的操作系统。

嵌入式操作系统具有良好的可移植性;软件开发工作变得规范,容易测试,可实现模块化编程,同时由多人共同完成 1 个任务;解决以往开发产品存在的诸多安全隐患。很多软件厂商迎合嵌入式系统发展的需要,推出了多种不同特点的嵌入式操作系统。例如,Microsoft 公司的 Windows CE、3COM 公司的 Palm OS、Symbian 公司的 EPOC、中科院凯思集团的 HOpen,以及 Linux 等。

### 1.2.2 从芯片级设计到系统级设计

嵌入式控制系统需要具有能跨越传统硬件和软件设计界限的新型设计人员,这是嵌入式系统技术与传统单片机和微处理器系统技术的重要区别。嵌入式系统具有系统级应用开发技术的基本特征,而以往的微处理器、单片机和 DSP 处理器应用设计,则属于器件或芯片级应用开发技术。以操作系统和硬件系统提供的接口作为应用系统设计的基础,其开发技术与 PC 的应用开发颇为相似。

在嵌入式系统应用开发中,设计者不必了解很多底层知识和技术,只需要在操作系统之上完成应用系统设计即可。嵌入式系统随设计需要来选择,如成本、性能、功率、体积和灵活性等。

(1) 随着各种各样的微处理器、微控制器、可编程逻辑器件功能的增强以及成本的降低,电子电路设计与十年以前相比,已经发生了本质的变化。最根本的变化不在于技术,也不在于器件,而是在于设计思想的变化。只有思想上的突破,才可能有设计上的突破;只有设计上的突破,才可能有产品上的突破。

(2) 过去电子线路的设计,由于器件成本较高,种类也比较单一,许多设计都是先看器件能实现什么功能,再决定去做什么任务,即面向器件的设计。现在,由于器件品种非常丰富,器件性能从低端到高端一应俱全,不管是从性能还是从成本方面考虑,都有非常大的选择余地。因此,所需要实现的任务才是核心问题,可以根据任务目标去选择最恰当的器件,这就是面向任务的设计。

### 1.2.3 从单处理器设计到多处理器设计

多微片器和核心微处理器在并行工作的能力要比单处理器或单核强。过去电子线路的设计,多以单一微处理器为核心,目前已发展为双处理器以及多处理器或多核心处理器的设计。任务的执行也由顺序执行发展为并行执行。

通常情况下,一个应用系统都以一个微处理器为核心,外加键盘驱动、显示驱动以及控制驱动,以及一些 ADC、DAC 芯片进行数据采集及数据输出。目前,在许多情况下,这样安排不一定是合理的。例如,ADC 转换,用一片专用的芯片,有时不如用一片带 ADC 的 AVR 单片机简单,有时不如用一片 ARM 处理实现效果好(用 ARM 实现的优点是同时可以把 A/D 转换、数字滤波、异常处理、输出通信协议等都放在一片 ARM 中实现,这是专用 A/D 芯片所不能实现的);键盘以及 LED/LCD 驱动电路,采用专用芯片也不如用一片 MCS51 或 AVR 单片机来实现成本更低;做快速事务处理及流程控制,用一片 FPGA (SOPC) 远比用一片 MCS51、AVR 以及 ARM 处理器有效,即相当于以可编程逻辑器件为核心,而把 AVR、ARM 等作为 FPGA (SOPC) 的外围器件,实现通信、A/D 转换等外设功能,这一点与常规的以 AVR、ARM 微处理器为核心的应用系统正好相反。以上都是随着电子技术、电子器件的不断进步所带来的设计思想上的突破。

## 1.3 多核处理器

### 1.3.1 单核处理器的局限性

#### 1. 处理器采用多核结构

1971 年,英特尔推出的全球第一颗通用型微处理器 4004,由 2300 个晶体管构成。当时,公