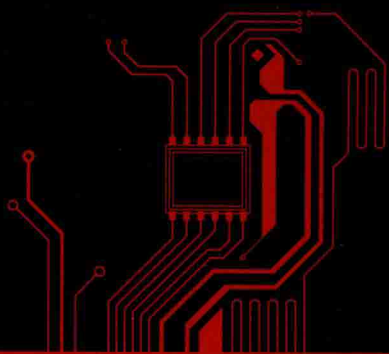




嵌入式系统设计与应用

—— 基于 ARM Cortex-A8 和 Linux

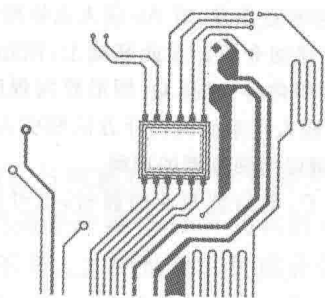
◎ 王剑 刘鹏 胡杰 文汉云 主编



- ◎ 基于 ARM Cortex-A8 处理器和嵌入式 Linux 操作系统，贴近主流
- ◎ 理论联系实际，原理阐述与三个经典实用案例有效结合
- ◎ 嵌入式系统开发从底层驱动到应用软件的全部覆盖

清华大学出版社





嵌入式系统设计与应用

—— 基于ARM Cortex-A8和Linux

© 王剑 刘鹏 胡杰 文汉云 主编



清华大学出版社
北京

内 容 简 介

本书选择当前嵌入式系统领域里具有代表性的 ARM Cortex-A8 处理器和嵌入式 Linux 操作系统作为分析对象,从嵌入式系统的发展历史和特点出发,对 A8 嵌入式处理器的体系结构、存储系统、中断系统、ARM 指令集和 S5PV210 微处理器进行详细介绍。在此基础上,阐述 ARM-linux 内核、Linux 文件系统、Linux 移植及调试方法、驱动程序的设计等内容;选择 Qt 图形界面程序和 SQLite 数据库这两个有代表性的嵌入式应用软件进行介绍;介绍传统嵌入式系统的设计方法和引入了“协同设计”概念的嵌入式系统设计方法,并介绍三个嵌入式系统应用案例以增强读者的理解。

本书可以作为高等学校计算机、电子、电信类专业的教材,也可以作为嵌入式开发相关人员的参考用书。

本书封面贴有清华大学出版社防伪标签,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

嵌入式系统设计与应用:基于 ARM Cortex-A8 和 Linux/王剑等主编. —北京:清华大学出版社,2017
(电子设计与嵌入式开发实践丛书)
ISBN 978-7-302-45806-7

I. ①嵌… II. ①王… III. ①微型计算机—系统设计 IV. ①TP360.21

中国版本图书馆 CIP 数据核字(2016)第 291652 号

责任编辑:刘 星 梅栾芳

封面设计:刘 键

责任校对:焦丽丽

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

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

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:24.5

字 数:594 千字

版 次:2017 年 2 月第 1 版

印 次:2017 年 2 月第 1 次印刷

印 数:1~2000

定 价:59.00 元

前言

嵌入式计算机技术是 21 世纪计算机技术重要发展方向之一,应用领域十分广泛且增长迅速。据估计,未来十年 95% 的微处理器和 65% 的软件都将被应用于各种嵌入式系统中。技术的发展和生产力的提高离不开人才的培养。目前业界对嵌入式技术人才的需求十分巨大,尤其在迅速发展的电子、通信、计算机等领域,这种需求更为显著。另外,企业对嵌入式系统开发从业者的工程实践能力、经验要求也越来越重视。因此目前国内外很多专业协会和高校都在致力于嵌入式相关课程体系建设,结合嵌入式系统的特点,在课程内容设计、师资队伍建设和教学方法探索、教学条件和实验体系建设等方面都取得了较好成效。

从国外嵌入式课程建设来看,2004 年,ACM(美国计算机协会)和 IEEE 联合制订了新的计算机学科的课程体系(2004 版),其中一个主要改革就是将 Embedded System 课程列为本科生的专业基础课,并且给出了基本课程体系。同时,美国卡内基-梅隆大学、加州大学伯克利分校等国外高校也不断在完善他们的嵌入式教育体系,欧盟也推出了面向欧盟高校和企业的嵌入式研究计划,这些信息为编写适合计算机专业使用的“嵌入式系统”教材提供了指导和参考。

从技术更新角度来看,近年来嵌入式系统技术得到了广泛应用和爆发性增长,普适计算、无线传感器、可重构计算、物联网、云计算等新兴技术的出现又为嵌入式系统技术的研究与应用注入了新的活力。这也对“嵌入式系统”课程教材的设计提出了更新更高的要求。

从国内嵌入式系统教材来看,一是有部分嵌入式硬件系统仍然采用 ARM9 体系结构作为核心处理器架构,甚至还有 ARM7 体系结构,这不仅和市场脱节极其严重,也和高校与时俱进的教学理念产生冲突;二是目前国内高校配合嵌入式系统理论教材的嵌入式系统实验实训平台普遍已经进入更新换代周期,根据调研发现,国内许多高校嵌入式实验平台多在 2006 年至 2009 年购入,普遍采用 ARM7、ARM9 架构的核心处理器,至今已工作六年以上,在未来两年内将迎来实验设备普遍更换的潮流,而目前的教材仍然以老平台作为实践依据,这也形成了理论和实践的脱节,也要求嵌入式理论教材能够跟上实验实训要求的步伐;三是从应用角度来看,操作系统和应用软件近年来都取得较大发展,开源 OS 和 APP 的迅猛发展以及物联网与嵌入式系统的紧密结合也要求嵌入式教材尽可能保持技术的敏感度。

在此前提下,本书对基于 Cortex-A8 处理器架构和嵌入式 Linux 的嵌入式系统进行了阐述。本书的特点如下。

① 参考了 ACM&IEEE 联合制订的计算机学科的课程体系(2004 版)关于 Embedded

Foreword

System 的课程要求,并结合了国内高校计算机学科课程大纲的要求。参考资料主要来自近几年国内外出版的嵌入式相关刊物、ARM 官网、嵌入式专业网站、著名嵌入式设备公司相关资料和编写小组近年来的科研项目与指导学生创新实践活动资料,具有较好的时效性和实用性。

② 采用 ARMv7 版本的 Cortex-A8 处理器架构作为系统核心处理器架构,取代原有 ARM7/ARM9 处理器架构。ARM9 架构作为国内嵌入式系统教学主要选择架构已经近十年,一方面市场上主流芯片已经难觅其踪,另一方面国内高校 ARM9 实验平台也已经普遍超期服役,在未来两年内将迎来实验设备普遍更换的高峰。在更新设备选择资源中,以 Cortex-A8 处理器架构的实验实训平台具有极高的性价比(良好的扩展性和众多嵌入式设备厂家支持)。Cortex-A8 处理器是 ARM 的第一款超标量处理器,具有提高代码密度和性能的技术,用于多媒体和信号处理的 NEON 技术,以及用于高效支持预编译和即时编译 Java 及运行时编译目标(RCT)技术。同时 Cortex-A8 处理器架构目前属于技术上稳定的处理器架构,有较多相关的嵌入式实验平台可供选择。

③ 增加 Qt 和 SQLite 数据库相关知识,以满足日益增长的嵌入式系统 UI 设计与数据处理需求,更符合计算机学科特点和满足智能移动平台需要。

④ 从编写小组自身从事的科研项目和实践活动出发,选择具有一定实用价值,包含交叉学科知识,反映嵌入式系统与物联网技术结合的三个项目实例。这些实例不仅从理论上深化拓展嵌入式系统设计方法和理念,也从实践角度提出“碰到问题如何运用所学知识解决问题”的观点,促进学生学以致用思想的升华。

本书受国家自然科学基金面上项目“随钻测量井下网络化光纤传感器及信息传输关键技术研究”(编号 41372155)的支持。编写过程中,王剑负责第 1、11 章的编写和全书的统稿;刘鹏负责第 2、3、4、9 章的编写工作;胡杰负责第 5、6 章的编写工作;文汉云负责第 10 章的编写工作;孟真玮负责第 7、8 章的编写工作。叶玲对本书进行了审校工作。同时本书的编写也得到了朱文霞和深圳博嵌科教仪器有限公司的大力支持和帮助,在此表示衷心的感谢。

本书参考了国内外的许多最新的技术资料,书末有具体的参考文献,有兴趣的读者可以查阅相关信息。本书配有电子课件,需要的读者可以登录清华大学出版社本书页面下载。

限于编者水平有限,书中不妥之处在所难免,敬请广大读者批评指正并提出宝贵意见。

作者

2016 年 12 月

目 录

第 1 章 嵌入式系统概述	1
1.1 嵌入式系统简介	1
1.1.1 嵌入式系统的产生	1
1.1.2 嵌入式系统的定义、特点和分类	2
1.1.3 嵌入式系统的两种应用模式	4
1.1.4 嵌入式系统的典型组成	5
1.2 嵌入式微处理器	6
1.2.1 嵌入式微处理器简介	6
1.2.2 主流嵌入式微处理器	8
1.3 嵌入式操作系统	9
1.3.1 嵌入式 Linux	9
1.3.2 Windows CE	11
1.3.3 Symbian	11
1.3.4 Android	12
1.3.5 μ C/OS-II	12
1.3.6 VxWorks	13
1.4 嵌入式系统的应用领域和发展趋势	13
1.4.1 嵌入式系统的应用领域	13
1.4.2 嵌入式系统的发展趋势	13
1.5 本章小结	14
习题	14
第 2 章 ARM 处理器体系结构	15
2.1 ARM 处理器概述	15
2.1.1 ARM 处理器简介	15
2.1.2 ARM 体系结构发展	16
2.1.3 ARM 处理器系列主要产品	19

Contents

2.1.4	ARM 开发工具简介	24
2.2	Cortex-A8 处理器架构	26
2.3	Cortex-A8 处理器工作模式和状态	30
2.3.1	Cortex-A8 处理器工作模式	30
2.3.2	Cortex-A8 处理器状态	31
2.4	Cortex-A8 存储器管理	32
2.4.1	ARM 的基本数据类型	32
2.4.2	浮点数据类型	32
2.4.3	大/小端存储模式	32
2.4.4	寄存器组	33
2.4.5	Cortex-A8 存储系统	37
2.5	Cortex-A8 异常处理	43
2.5.1	异常向量和优先级	43
2.5.2	异常响应过程	44
2.5.3	异常返回过程	45
2.5.4	Cortex-A8 处理器 S5PC100 中断机制	46
2.6	本章小结	48
	习题	48
第 3 章	ARM 指令集	49
3.1	ARM 指令集概述	49
3.1.1	指令格式	49
3.1.2	指令的条件码	50
3.2	ARM 指令的寻址方式	51
3.2.1	立即寻址	51
3.2.2	寄存器寻址	51
3.2.3	寄存器间接寻址	51
3.2.4	寄存器移位寻址	52
3.2.5	变址寻址	52
3.2.6	多寄存器寻址	52
3.2.7	相对寻址	53
3.2.8	堆栈寻址	53
3.2.9	块复制寻址	54
3.3	ARM 指令简介	54
3.3.1	跳转指令	54
3.3.2	数据处理指令	55
3.3.3	程序状态寄存器处理指令	59
3.3.4	加载/存储指令	60
3.3.5	协处理器指令	61

3.3.6 异常产生指令	62
3.4 Thumb 指令简介	63
3.5 ARM 汇编语言编程简介	63
3.5.1 伪操作	63
3.5.2 伪指令	65
3.5.3 汇编语句格式	66
3.5.4 汇编语言的程序结构	66
3.6 C 语言与汇编语言的混合编程	67
3.6.1 C 程序中内嵌汇编	67
3.6.2 汇编中访问 C 语言程序变量	67
3.6.3 ARM 中的汇编和 C 语言相互调用	68
3.7 本章小结	69
习题	69
第 4 章 S5PV210 微处理器与接口	71
4.1 基于 S5PV210 微处理器的硬件平台体系结构	71
4.1.1 S5PV210 处理器简介	71
4.1.2 S5PV210 内部各模块介绍	72
4.2 存储系统	76
4.2.1 S5PV210 的地址空间	76
4.2.2 S5PV210 启动流程	77
4.3 时钟系统	78
4.3.1 S5PV210 时钟概述	78
4.3.2 S5PV210 的时钟结构	79
4.4 GPIO 接口	81
4.4.1 GPIO 概述	81
4.4.2 GPIO 寄存器	83
4.4.3 GPIO 操作步骤	83
4.4.4 一个 LED 灯的例子	84
4.5 串行通信接口	89
4.5.1 串行通信方式	89
4.5.2 RS-232C 串行接口	90
4.5.3 S5PV210 的异步串行通信	91
4.5.4 S5PV210 的 UART 寄存器	92
4.5.5 UART 通信示例	92
4.6 A/D 转换器	97
4.7 本章小结	100
习题	100

第 5 章 ARM-Linux 内核	101
5.1 ARM-Linux 概述	101
5.1.1 GNU/Linux 操作系统的基本体系结构	101
5.1.2 ARM-Linux 内核版本及特点	102
5.1.3 ARM-Linux 内核的主要架构及功能	104
5.1.4 Linux 内核源码目录结构	105
5.2 ARM-Linux 进程管理	106
5.2.1 进程的表示和切换	107
5.2.2 进程、线程和内核线程	109
5.2.3 进程描述符 task_struct 的几个特殊字段	109
5.2.4 do_fork() 函数	110
5.2.5 进程的创建	112
5.2.6 线程和内核线程的创建	113
5.2.7 进程的执行——exec 函数族	113
5.2.8 进程的终止	114
5.2.9 进程的调度	115
5.3 ARM-Linux 内存管理	117
5.3.1 ARM-Linux 内存管理概述	117
5.3.2 ARM-Linux 虚拟存储空间及分布	118
5.3.3 进程空间描述	119
5.3.4 物理内存管理	123
5.3.5 基于 slab 分配器的管理技术	124
5.3.6 内核非连续内存分配(vmalloc)	125
5.3.7 页面回收简述	127
5.4 ARM-Linux 模块	127
5.4.1 LKM 的编写和编译	128
5.4.2 LKM 版本差异比较	128
5.4.3 模块的加载与卸载	129
5.4.4 工具集 module-init-tools	131
5.5 ARM-Linux 中断管理	132
5.5.1 ARM-Linux 中断的一些基本概念	132
5.5.2 内核异常向量表的初始化	133
5.5.3 Linux 中断处理	135
5.5.4 内核版本 2.6.38 后的中断处理系统的一些改变——通用中断 子系统	142
5.6 ARM-Linux 系统调用	143
5.7 本章小结	146
习题	146

第 6 章 Linux 文件系统	148
6.1 Linux 文件系统概述	148
6.2 ext2 文件系统格式	150
6.2.1 ext2 文件系统	150
6.2.2 目录结构	153
6.3 ext3 和 ext4 文件系统	154
6.3.1 ext3 文件系统	154
6.3.2 ex4 文件系统	155
6.4 嵌入式文件系统 JFFS2	155
6.4.1 嵌入式文件系统	155
6.4.2 JFFS2 嵌入式文件系统	157
6.5 YAFFS 与 YAFFS2 文件系统简介	164
6.5.1 YAFFS 文件系统	164
6.5.2 YAFFS2 文件系统简介	165
6.6 根文件系统	166
6.6.1 根文件系统概述	166
6.6.2 根文件系统的制作工具——BusyBox	168
6.6.3 YAFFS2 文件系统的创建	170
6.7 本章小结	171
习题	171
第 7 章 嵌入式 Linux 系统移植及调试	172
7.1 BootLoader 基本概念与典型结构	172
7.1.1 BootLoader 基本概念	172
7.1.2 BootLoader 的操作模式	173
7.1.3 BootLoader 的典型结构	174
7.1.4 常见的 BootLoader	174
7.2 U-Boot	175
7.2.1 U-Boot 概述	175
7.2.2 U-Boot 启动的一般流程	176
7.2.3 U-Boot 环境变量	180
7.2.4 U-Boot 命令	181
7.3 交叉开发环境的建立	183
7.4 交叉编译工具链	185
7.4.1 交叉编译工具链概述	185
7.4.2 工具链的构建方法	186
7.4.3 交叉编译工具链的主要工具	186
7.4.4 Makefile	192

7.5	嵌入式 Linux 系统移植过程	194
7.5.1	U-Boot 移植	194
7.5.2	内核的配置、编译和移植	198
7.6	GDB 调试器	205
7.7	远程调试	209
7.7.1	远程调试工具的构成	209
7.7.2	通信协议——RSP	210
7.7.3	远程调试的实现方法及设置	211
7.7.4	远程调试应用实例方法	212
7.8	内核调试	213
7.8.1	printk()	213
7.8.2	KDB	214
7.8.3	Kprobes	216
7.8.4	KGDB	216
7.9	本章小结	217
	习题	218
第 8 章	设备驱动程序设计	219
8.1	设备驱动程序开发概述	219
8.1.1	Linux 设备驱动程序分类	220
8.1.2	驱动程序的处理过程	221
8.1.3	设备驱动程序框架	230
8.1.4	驱动程序的加载	233
8.2	内核设备模型	234
8.2.1	设备模型功能	235
8.2.2	sysfs	235
8.2.3	sysfs 的实现机制 kobject	237
8.2.4	设备模型的组织——platform 总线	239
8.3	字符设备驱动设计框架	243
8.3.1	字符设备的重要数据结构	243
8.3.2	字符设备驱动框架	248
8.4	GPIO 驱动概述	252
8.4.1	gpiolib 关键数据结构	252
8.4.2	GPIO 的申请和注册	253
8.5	I ² C 总线驱动设计	255
8.5.1	I ² C 总线概述	256
8.5.2	I ² C 驱动程序框架	257
8.5.3	关键数据结构	258
8.5.4	I ² C 核心接口函数	261

8.5.5 I ² C 设备驱动的通用方法	265
8.6 块设备驱动程序设计概述	267
8.6.1 块设备驱动整体框架	267
8.6.2 关键数据结构	268
8.6.3 块设备的请求队列操作	275
8.7 嵌入式网络设备驱动设计	276
8.7.1 网络设备驱动程序框架	277
8.7.2 网络设备驱动程序关键数据结构	278
8.7.3 网络设备驱动程序设计方法概述	282
8.8 网络设备驱动程序示例——网卡 DM9000 驱动程序分析	282
8.9 本章小结	291
习题	292
第 9 章 Qt 图形界面应用程序开发基础	293
9.1 Qt 简介	293
9.2 Qt 5 概述	294
9.2.1 Qt 5 简介	294
9.2.2 通过“帮助”菜单了解 Qt 5 的组成——模块	295
9.2.3 Linux 下 Qt 开发环境的安装与集成	297
9.2.3 Qt Creator 功能和特性	299
9.3 信号和插槽机制	299
9.4 Qt 程序设计	302
9.4.1 helloworld 程序	302
9.4.2 多窗口应用程序	303
9.5 Qt 数据库应用	309
9.5.1 数据库驱动	309
9.5.2 Qt 与 SQLite 数据库的连接	310
9.5.3 SQL 模型	311
9.6 本章小结	317
习题	318
第 10 章 SQLite 数据库	319
10.1 SQLite 数据库概述	321
10.1.1 基于 Linux 平台的嵌入式数据库概述	321
10.1.2 SQLite 的特点	322
10.1.3 SQLite 的体系结构	323
10.2 SQLite 安装	324
10.3 SQLite 的常用命令	326
10.4 SQLite 的数据类型	328

10.5	SQLite 的 API 函数	329
10.5.1	核心 C API 函数	329
10.5.2	扩充 C API 函数	335
10.6	本章小结	337
	习题	337
第 11 章	嵌入式系统的开发设计案例	338
11.1	嵌入式系统设计方法介绍	338
11.1.1	传统的嵌入式系统设计方法	338
11.1.2	“协同设计”概念的嵌入式系统设计方法	342
11.2	基于 ARM 的嵌入式 Web 服务器设计实例	344
11.2.1	系统环境搭建	344
11.2.2	Web 服务器原理	349
11.2.3	嵌入式 Web 服务器设计	350
11.3	物联网网关设计实例	355
11.3.1	背景介绍——环境监测系统平台整体架构	355
11.3.2	网关节点硬件设计方案	357
11.3.3	系统软件设计	360
11.3.4	数据库建设	360
11.4	智能无人值守实验室监控系统设计实例	361
11.4.1	系统总体框架	361
11.4.2	学生选课预约	364
11.4.3	门禁系统	365
11.4.4	ZigBee 网络的网络拓扑及路由协议	365
11.4.5	Qt 的使用	369
11.5	本章小结	375
	参考文献	376

嵌入式系统概述

进入 21 世纪,随着各种手持终端和移动设备的发展,嵌入式系统(Embedded System)的应用已从早期的科学研究、军事技术、工业控制和医疗设备等专业领域逐渐扩展到日常生活的各个领域。在涉及计算机应用的各行各业中,几乎 90%左右的开发都涉及嵌入式系统的开发。嵌入式系统的应用,为社会的发展起到了很大的促进作用,也给人们的日常生活带来了极大便利。

本章主要介绍嵌入式系统的基本知识,包括嵌入式系统的基本概念和特点,嵌入式微处理器和嵌入式操作系统,并在此基础上介绍嵌入式系统的应用领域和发展趋势。

1.1 嵌入式系统简介

1.1.1 嵌入式系统的产生

电子数字计算机诞生于 1946 年。在随后的发展过程中,计算机始终是供养在特殊机房中、实现数值计算的大型昂贵设备。直到 20 世纪 70 年代,随着微处理器的出现,计算机才出现了历史性的变化,以微处理器为核心的微型计算机以其小型、廉价、高可靠性等特点,迅速走出机房,演变成大众化的通用计算装置。

另一方面,基于高速数值计算能力的微型计算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,实现对对象体系的智能化控制。例如,将微型计算机经电气、机械加固,并配置各种外围接口电路,安装到大型舰船中构成自动驾驶仪或轮机状态监测系统。于是,现代计算机技术的发展,便出现了两大分支:以高速、海量的数值计算为主的计算机系统和嵌入到对象体系中、以控制对象为主的计算机系统。为了加以区别,人们把前者称为通用计算机系统,而把后者称为嵌入式计算机系统。

通用计算机系统以数值计算和处理为主,包括巨型机、大型机、中型机、小型机、微型机等。其技术要求是高速、海量的数值计算,技术方向是总线速度的无限提升、存储容量的无限扩大等。

嵌入式计算机系统以对象的控制为主,其技术要求是对对象的智能化控制能力,技术发展方向是与对象系统密切相关的嵌入性能、控制能力与控制的可靠性等。

1.1.2 嵌入式系统的定义、特点和分类

1. 嵌入式系统的定义

嵌入式系统诞生于微型机时代,其本质是将一个计算机嵌入到一个对象体系中去,这是理解嵌入式系统的基本出发点。目前,国际国内对嵌入式系统的定义有很多。例如,国际电气和电子工程师协会(the Institute of Electrical and Electronics Engineers, IEEE)对嵌入式系统的定义为:嵌入式系统是用来控制、监视或者辅助机器、设备或装置运行的装置。而国内普遍认同的嵌入式系统定义是:嵌入式系统是以应用为中心、以计算机技术为基础,软、硬件可裁剪,适应于应用系统对功能、可靠性、成本、体积、功耗等方面有特殊要求的专用计算机系统。

国际上对嵌入式系统的定义是一种广泛意义上的理解,偏重于嵌入,将所有嵌入到机器、设备或装置中,对宿主起控制、监视或辅助作用的装置都归类为嵌入式系统。而国内则对嵌入式系统的含义进行了收缩,明确指出嵌入式系统其实是一种计算机系统,围绕“嵌入到对象体系中的专用计算机系统”加以展开,使其更加符合嵌入式系统的本质含义。“嵌入性”“专用性”与“计算机系统”是嵌入式系统的三个基本要素,对象体系则是指嵌入式系统所嵌入的宿主系统。

与个人计算机这样的通用计算机系统不同,嵌入式系统通常执行的是带有特定要求的预先定义的任务,由于嵌入式系统通常都只针对一项特殊的任务,所以设计人员往往能够对它进行优化、减小尺寸、降低成本。

嵌入式系统与对象系统密切相关,其主要技术发展方向是满足嵌入式应用要求,不断扩展对象系统要求的外围电路(如 ADC、DAC、PWM、日历时钟、电源监测、程序运行监测电路等),形成满足对象系统要求的应用系统。嵌入式系统作为一个专用计算机系统,要不断向计算机应用系统发展。因此,可以把定义中的专用计算机系统引申成满足对象系统要求的计算机应用系统。

2. 嵌入式系统的特点

嵌入式系统的特点与定义不同,它是由定义中的三个基本要素衍生出来的。不同的嵌入式系统其特点会有所差异。

- ▶ 与“嵌入性”相关的特点:由于是嵌入到对象系统中,因此必须满足对象系统的环境要求,如物理环境(小型)、电气/气氛环境(可靠)、成本(价廉)等要求。
- ▶ 与“专用性”的相关特点:软硬件的裁剪性;满足对象要求的最小软、硬件配置等。
- ▶ 与“计算机系统”相关的特点:嵌入式系统必须是能满足对象系统控制要求的计算机系统。与前两个特点相呼应,这样的计算机必须配置有与对象系统相适应的接口电路。

需要注意的是:在理解嵌入式系统定义时,不要与嵌入式设备相混淆。嵌入式设备是指内部有嵌入式系统的产品、设备,如内含单片机的家用电器、仪器仪表、工控单元、机器人、手机、PDA 等。

3. 嵌入式系统的分类

嵌入式微处理器不能叫做真正的嵌入式系统,因为从本质上说嵌入式系统是一个嵌入式的计算机系统,只有将嵌入式微处理器构成了一个计算机系统,并作为嵌入式应用时,这样的计算机系统才可称为嵌入式系统。因此,对嵌入式系统的分类不能以微处理器为基准进行分类,而应以嵌入式计算机系统为整体进行分类。根据不同的分类标准,可按形态和系统的复杂程度进行分类。

按其形态的差异,一般可将嵌入式系统分为:芯片级(MCU、SoC)、板级(单板机、模块)和设备级(工控机)等三级。

按其复杂程度的不同,又可将嵌入式系统分为以下四类:

① 主要由微处理器构成的嵌入式系统,常常用于小型设备中(如温度传感器、烟雾和气体探测器及断路器);

② 不带计时功能的微处理器装置,可在过程控制、信号放大器、位置传感器及阀门传动器等中找到;

③ 带计时功能的组件,这类系统多见于开关装置、控制器、电话交换机、包装机、数据采集系统、医药监视系统、诊断及实时控制系统等;

④ 在制造或过程控制中使用的计算机系统,也是由工控机级组成的嵌入式计算机系统,是这四类中最复杂的一种,也是现代印刷设备中经常应用的一种。

4. 嵌入式系统的独立发展道路

(1) 单片机开创了嵌入式系统的独立发展道路

嵌入式系统虽然起源于微型计算机时代,然而,微型计算机的体积、价位、可靠性都无法满足广大对象系统的嵌入式应用要求,因此,嵌入式系统必须走独立发展道路。这条道路就是芯片化道路:将计算机做在一个芯片上,从而开创了嵌入式系统独立发展的单片机时代。

在探索单片机的发展道路时,有过两种模式,即“ Σ 模式”与“创新模式”。“ Σ 模式”本质上是通用计算机直接芯片化的模式,它将通用计算机系统中的基本单元进行裁剪后,集成在一个芯片上,构成单片微型计算机;“创新模式”则完全按嵌入式应用要求设计全新的,满足嵌入式应用要求的体系结构、微处理器、指令系统、总线方式、管理模式等。

Intel公司的MCS-48、MCS-51就是按照创新模式发展起来的单片形态的嵌入式系统(单片微型计算机)。MCS-51是在MCS-48探索基础上,进行全面完善的嵌入式系统。历史证明,“创新模式”是嵌入式系统独立发展的正确道路,MCS-51的体系结构也因此成为单片嵌入式系统的典型结构体系。

(2) 单片机的技术发展史

单片机诞生于20世纪70年代末,经历了SCM、MCU、SoC三大阶段。

SCM即单片微型计算机(Single Chip Microcomputer)阶段,主要是寻求单片形态嵌入式系统的最佳体系结构。“创新模式”获得成功,奠定了SCM与通用计算机完全不同的发展道路。在开创嵌入式系统独立发展道路上,Intel公司功不可没。

MCU即微控制器(Microcontroller Unit)阶段,主要的技术发展方向是:不断扩展满足嵌入式应用时,对象系统要求的各种外围电路与接口电路,凸显其对象的智能化控制能力。它所涉及的领域都与对象系统相关,因此,发展MCU的重任不可避免地落在电气、电子技术厂家。从这一角度来看,Intel逐渐淡出MCU的发展也有其客观因素。在发展MCU方

面,最著名的厂家当属 Philips 公司。

Philips 公司以其在嵌入式应用方面的巨大优势,将 MCS-51 从单片微型计算机迅速发展 to 微控制器。因此,当我们回顾嵌入式系统发展道路时,不要忘记 Intel 和 Philips 的历史功绩。

单片机开启了嵌入式系统的独立发展之路,向 MCU 阶段发展的重要因素,就是寻求应用系统在芯片上的最大化解。专用单片机的发展自然形成了 SoC 化的趋势,随着微电子技术、IC 设计、EDA 工具的发展,基于 SoC 的单片机应用系统设计会有较大的发展。因此,对单片机的理解可以从单片微型计算机、单片微控制器延伸到单片应用系统。

1.1.3 嵌入式系统的两种应用模式

嵌入式系统的嵌入式应用特点,决定了它多学科交叉的特点。作为计算机的内行,要求计算机领域人士介入其体系结构、软件技术、工程应用方面的研究。然而,了解对象系统的控制要求,实现系统控制模式必须具备对象领域的专业知识。因此,从嵌入式系统发展的历史过程,以及嵌入式应用的多样性中,可以了解到客观上形成的两种应用模式——以电子技术应用工程师为主体的电子系统智能化应用和以计算机专业人士为主体,基于嵌入式系统软硬件平台,以网络、通信为主的嵌入式上层应用。

(1) 客观存在的两种应用模式

嵌入式计算机系统起源于微型机时代,但很快就进入到独立发展的单片机时代。在单片机时代,嵌入式系统以器件形态迅速进入到传统电子技术领域中,以电子技术应用工程师为主体,实现传统电子系统的智能化,而计算机专业队伍并没有真正进入单片机应用领域。因此,电子技术应用工程师以自己习惯性的电子技术应用模式,从事单片机的应用开发。这种应用模式最重要的特点是:软硬件的底层性和随意性;对象系统专业技术的密切相关性;缺少计算机工程设计方法。

虽然在单片机时代,计算机专业淡出了嵌入式系统领域,但随着后 PC 时代的到来,网络、通信技术得以发展;同时,嵌入式系统软、硬件技术有了很大的提升,为计算机专业人士介入嵌入式系统的应用开辟了广阔天地。计算机专业人士的介入所形成的计算机应用模式带有明显的计算机工程应用特点,即基于嵌入式系统软、硬件平台,以网络、通信为主的非嵌入式底层应用。

(2) 两种应用模式的并存与互补

由于嵌入式系统最大、最广、最底层的应用是传统电子技术领域的智能化改造,因此,以通晓对象专业的电子技术队伍为主,用最少的嵌入式系统软、硬件开销,以 8 位机为主,带有浓重的电子系统设计色彩的电子系统应用模式会长期存在下去。另外,计算机专业人士会愈来愈多地介入嵌入式系统应用,但囿于对象专业知识的隔阂,其应用领域会集中在网络、通信、多媒体、商务电子等方面,不可能替代原来电子工程师在控制、仪器仪表、机械电子等方面的嵌入式应用。因此,客观存在的两种应用模式会长期并存下去,在不同的领域中相互补充。电子系统设计模式应从计算机应用设计模式中,学习计算机工程方法和嵌入式系统软件技术;计算机应用设计模式应从电子系统设计模式中,了解嵌入式系统应用的电路系统特性、基本的外围电路设计方法和对象系统的基本要求等。