



普通高等教育“十二五”规划教材

◎ 电子信息科学与工程类专业 规划教材

LPC1100

系列处理器原理及应用

◎ 桂电-丰宝联合实验室 编著



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY

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

电子信息科学与工程类专业规划教材

LPC1100

系列处理器原理及应用

桂电-丰宝联合实验室 编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书从基础的理论知识到实际的应用,详细介绍了LPC1100系列处理器的结构、指令、接口和应用开发实例等,是读者学习Cortex-M0应用开发的基础入门教程。全书共分5个部分,分别为:(1)嵌入式系统及ARM处理器的概述,Cortex-M0处理器的特点和技术规范;(2)LPC1100系列处理器的硬件结构和指令系统,包括LPC1100系列处理器基本结构、总线结构、存储器管理、寄存器组织、系统配置、电源管理、串行线调试、最小系统、Cortex-M0常用的指令集和寻址方式;(3)IAR公司的EWARM集成开发环境;(4)中断系统与LPC1100接口技术,详细介绍通用I/O口、定时/计数器、串行总线(UART、SPI、I²C和CAN)、A/D转换器;(5)基于LPC1100系列处理器的综合应用实例,介绍了SPI Flash存储器的读写、波形发生器的设计、矩阵键盘与显示、温度采集4个应用开发实例。本书的所有例程均已在上海丰宝电子科技有限公司开发的LINPO-PS-LPC11xx实验平台测试通过,该平台可与本书配合使用。

本书可作为高等院校电子信息类专业本科生及研究生“嵌入式系统”课程的入门教程,也可供从事嵌入式系统设计的研发人员参考。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

图书在版编目(CIP)数据

LPC1100 系列处理器原理及应用/桂电-丰宝联合实验室编著. —北京:电子工业出版社,2011.10

电子信息科学与工程类专业规划教材

ISBN 978-7-121-14848-4

I. ①L… II. ①桂… III. ①微处理器—高等学校—教材 IV. ①TP332

中国版本图书馆CIP数据核字(2011)第212618号

责任编辑:凌毅 特约编辑:张莉

印刷:北京市海淀区四季青印刷厂

装订:三河市鹏成印业有限公司

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:787×1092 1/16 印张:16 字数:410千字

印次:2011年10月第1次印刷

印数:4000册 定价:36.00元

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 zlt@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

随着我国信息化建设的不断发展,嵌入式技术成为发展最快、应用最广、最有前景的技术之一,嵌入式设备也不断应用到人们生活的各个领域,如电信、医疗、汽车电子、安全和消费等。广阔的应用前景导致了巨大的人才需求,据国内人才网统计,嵌入式工程师凭借其旺盛的需求荣登 IT 职场人气排行榜榜首,社会各界也对嵌入式人才的就业前景非常看好。

面对嵌入式系统如火如荼的发展形势,业界掀起了嵌入式系统理论和应用开发的学习热潮,各大高校也开设了相关课程。“桂电-丰宝联合实验室”在合作开发嵌入式教学平台的基础上,相继出版了嵌入式系列教材“ARM 原理与嵌入式应用——基于 LPC2400 系列处理器和 IAR 开发环境”、“基于 LPC3250 的嵌入式 Linux 系统开发”。本书是“桂电-丰宝联合实验室”长期合作的又一成果,本书以 LPC1100 系列处理器为例,介绍 Cortex-M0 处理器的工作原理与应用开发。

Cortex-M0 是 ARM 公司为了占领单片机市场而推出的一款极具竞争力的 MCU,该处理器是市场上现有的最小、功耗最低、最节能的具有 32 位架构性能的 ARM 处理器。与素有“体积小、功耗低、简单易学、性能价格比高”的 8051 单片机相比,Cortex-M0 的价格与之相当,但体积更小,功耗更低,性能更强,学习难度也与 8051 相当,因此,该款处理器极适合作为入门级的首选。目前已有多家公司获得 Cortex-M0 处理器授权,比如新唐科技、NXP(恩智浦)等。其中,NXP 的 LPC1100 系列处理器以其低功耗和高性能等特性,在 Cortex-M0 产品中占有较大市场份额。

本书从基础理论知识到实际应用,详细介绍了 LPC1100 系列处理器的结构、指令、接口和中断等,是读者学习 Cortex-M0 应用开发的基础入门教程。同时,实践是嵌入式系统课程的重要环节,缺乏实践的嵌入式系统课程是纸上谈兵。为了便于读者揣摩研习,本书的所有例程均已在上海丰宝电子科技有限公司开发的基于 LPC1100 系列处理器的实验平台——LINPO-PS-LPC11xx 实验平台测试过,该平台可与本书配合使用。

本书围绕着 NXP 的 LPC1100 系列处理器及应用开发展开,共 10 章,5 个部分,包括嵌入式系统概述、LPC1100 硬件结构和指令系统、IAR EWARM 集成开发环境、LPC1100 接口技术与中断、基于 LPC1100 系列处理器的应用开发实例。

第一部分为第 1 章,即嵌入式系统概述,主要对嵌入式系统及 ARM 处理器进行了概述,对 ARM 处理器的结构进行了介绍,并简要介绍 Cortex-M0 处理器的特点和技术规范。

第二部分包括第 2 章和第 3 章,其中第 2 章主要介绍 LPC1100 系列处理器的硬件结构。内容包括 LPC1100 系列处理器的简介、总线结构、存储器管理、寄存器组织、系统配置、电源管理、串行线调试和最小系统等。第 3 章为 Cortex-M0 指令系统,首先对 Cortex-M0 指令系统进行了概述,介绍了几种常用的寻址方式,并对 Cortex-M0 常用的指令集进行了详细介绍,包括存储器访问指令、通用数据处理指令、跳转与控制指令和综合指令等,最后对汇编程序基本结构进行了举例说明。

第三部分为第 4 章,即 IAR 集成开发环境应用,主要介绍了 IAR EWARM 的特点,以及结合 LPC1100 系列处理器进行工程建立、编译与连接的方法,其中重点介绍了 IAR C-SPY 调

试器的使用。此外,还介绍了如何在上海丰宝电子科技有限公司的 LINPO-PS-LPC11xx 实验平台上进行应用程序的开发与调试。

第四部分为第 5~9 章,主要对 LPC1100 系列处理器的接口技术和中断系统进行了介绍。其中,第 5 章介绍中断控制系统,第 6 章介绍通用 I/O 口配置及其应用,第 7 章介绍定时/计数器及其应用,第 8 章介绍 LPC1100 系列处理器的串行总线通信及其应用,包括 UART、SPI、I²C 和 CAN,第 9 章对 LPC1100 系列处理器的 A/D 转换器及其应用进行介绍。在对接口技术进行介绍时,对接口配置的方法进行了详细描述,通过实例来引导读者更好地学习。

第五部分为第 10 章,是基于 LPC1100 系列处理器的应用开发实践。本部分介绍了 4 个应用开发实例,分别为 SPI Flash 存储器的读写、波形发生器的设计、矩阵键盘与显示、温度采集。SPI Flash 存储器的读写是针对 Winbond 公司生产的 W25X 系列 Flash 存储器的应用,波形发生器是采用 LPC1100 系列处理器具有的 PWM 功能而设计的,矩阵键盘与显示利用了 LINPO-PS-LPC11xx 实验平台上的矩阵键盘,温度采集是利用具有 I²C 接口的 LM75A 数字温度传感器和温度监控器而设计的温度采集系统。

附录部分介绍了 LPC1100 系列芯片的各封装引脚图、Cortex-M0 处理器指令集、IAR 下的汇编启动代码和 LINPO-PS-LPC11xx 实验平台的电路图。

本书可作为高等院校电子信息类专业本科生及研究生“嵌入式系统”课程入门教程,也可作为嵌入式爱好者或从事嵌入式系统应用的人员参考。本书也可以作为工程开发的入门指导书,还可作为 LPC1100 处理器的开发参考手册。另外,还可以作为 ARM Cortex M0 的编程入门指南。

本书由张红梅担任主编,由“桂电-丰宝联合实验室”的教师和工程师共同策划和编写,其中参与策划和编写的有张红梅、魏艺海、王鹏鹏、陈俊彦、张全君、吴阿沛。全书由张红梅负责统稿工作,郭远洋、吕兆辉、田庆阳、何燕、韦贤岁对本书进行了认真的校对。

感谢上海丰宝电子科技有限公司的朱玉峰总经理对本书工作的支持,感谢 NXP(北京)有限公司的王鹏鹏女士、NXP(上海)有限公司的王维先生对本书提出的许多建设性意见和不辞劳苦的审阅工作。同时,也非常感谢家人的理解与支持。

由于时间仓促及作者水平有限,书中难免存在遗漏和不足之处,恳请读者批评指正。

桂林电子科技大学

张红梅

2011 年 9 月

目 录

第 1 章 嵌入式系统概述	1	2.7.1 串行线调试概述	40
1.1 嵌入式系统	1	2.7.2 串行线调试的连接	40
1.1.1 嵌入式系统的定义	1	2.8 LPC1100 最小系统	41
1.1.2 嵌入式系统的特点	2	2.8.1 电源系统	41
1.1.3 嵌入式实时操作系统概述	2	2.8.2 复位系统	42
1.2 嵌入式处理器	4	2.8.3 调试接口和 ISP	43
1.2.1 嵌入式处理器的分类	4	2.8.4 时钟系统	44
1.2.2 ARM 微处理器	5	习题 2	44
1.2.3 Cortex-M0 处理器	7	第 3 章 Cortex-M0 指令系统	45
习题 1	8	3.1 Cortex-M0 指令概述	45
第 2 章 LPC1100 系列处理器的硬件		3.2 Cortex-M0 寻址方式	45
结构	9	3.2.1 立即寻址	46
2.1 LPC1100 系列处理器的简介	9	3.2.2 寄存器寻址	46
2.1.1 LPC1100 系列处理器的特点	9	3.2.3 寄存器间接寻址	46
2.1.2 LPC1110 系列处理器基本		3.2.4 基址加变址寻址	46
结构	11	3.2.5 多寄存器寻址	46
2.1.3 引脚描述	11	3.2.6 寄存器移位寻址	47
2.2 总线结构	16	3.2.7 相对寻址	47
2.3 存储器管理	16	3.2.8 堆栈寻址	47
2.3.1 LPC1100 系列处理器存储器		3.3 Cortex-M0 常用指令集	48
地址映射	17	3.3.1 存储器访问指令	48
2.3.2 异常向量表及其重映射	17	3.3.2 通用数据处理指令	51
2.3.3 Boot ROM	19	3.3.3 跳转与控制指令	56
2.4 寄存器组织	20	3.3.4 综合指令	56
2.4.1 通用寄存器	20	3.4 汇编应用程序举例	59
2.4.2 特殊功能寄存器	21	3.4.1 分支程序	59
2.5 系统配置	23	3.4.2 循环程序	59
2.5.1 时钟与 PLL 配置	23	3.4.3 子程序调用	60
2.5.2 外围电路复位配置	30	3.4.4 查表法	60
2.6 电源管理	32	3.4.5 汇编语言与 C/C++ 的混合	
2.6.1 系统工作模式	32	编程	61
2.6.2 电源管理单元及其他相关		习题 3	62
寄存器	34	第 4 章 IAR 集成开发环境应用	63
2.6.3 节电工作模式的配置	38	4.1 IAR EWARM 集成开发环境与仿真	
2.7 串行线调试(SWD)	40	工具	63

4.1.1 IAR EWARM 软件的特点	63	6.2.1 GPIO 口的结构特点	98
4.1.2 仿真工具的介绍	64	6.2.2 GPIO 口的配置	99
4.2 基于 LINPO-PS-LPC11xx 实验环境的搭建	66	6.2.3 GPIO 应用示例	104
4.2.1 LINPO-PS-LPC11xx 实验平台概述	66	习题 6	111
4.2.2 J-Link 仿真器的物理连接与驱动安装	66	第 7 章 LPC1100 系列处理器定时/计数器及其应用	112
4.3 工程的创建、编译与连接	67	7.1 定时/计数器	112
4.3.1 在 IAR EWARM 生成项目	67	7.1.1 定时/计数器概述	112
4.3.2 在 IAR EWARM 编译项目	72	7.1.2 定时/计数器的配置	113
4.3.3 在 IAR EWARM 连接项目	72	7.1.3 定时/计数器应用示例	122
4.4 IAR C-SPY 调试器	73	7.2 系统节拍定时器	129
4.4.1 C-SPY 调试器的启动	73	7.2.1 系统节拍定时器概述	129
4.4.2 窗口介绍	73	7.2.2 系统节拍定时器的配置	130
4.4.3 断点的设置	74	7.2.3 系统节拍定时器应用示例	131
4.4.4 其他功能	75	7.3 看门狗定时器	134
4.5 C 语言与汇编语言混合编程模式	76	7.3.1 看门狗定时器概述	134
习题 4	77	7.3.2 看门狗定时器的配置	135
第 5 章 LPC1100 系列处理器中断控制系统	78	7.3.3 看门狗定时器应用示例	138
5.1 NVIC 概述	78	习题 7	140
5.2 中断控制过程	79	第 8 章 LPC1100 系列处理器串行总线通信及其应用	141
5.2.1 异常类型及中断向量表	79	8.1 UART 串口通信	141
5.2.2 中断输入及挂起行为	80	8.1.1 UART 概述	141
5.2.3 中断优先级	82	8.1.2 UART 接口电路	141
5.2.4 中断响应及返回过程	82	8.1.3 UART 功能寄存器	143
5.3 中断源及 NVIC 相关寄存器	85	8.1.4 UART 接口配置	149
5.3.1 中断源	85	8.1.5 接口函数	151
5.3.2 NVIC 相关寄存器	87	8.2 SPI 接口	153
5.4 Cortex 微控制器软件接口标准 (CMSIS) 中的 NVIC 编程	89	8.2.1 概述	153
习题 5	89	8.2.2 SPI 接口电路	156
第 6 章 LPC1100 系列处理器 I/O 口配置及其应用	90	8.2.3 SPI 功能寄存器	156
6.1 I/O 口的配置	90	8.2.4 SPI 接口配置	161
6.1.1 I/O 口的引脚模式	90	8.2.5 接口函数	162
6.1.2 I/O 口的配置	91	8.3 I ² C 总线	164
6.1.3 I/O 配置示例	96	8.3.1 I ² C 概述	164
6.2 GPIO 口结构及功能	98	8.3.2 I ² C 接口电路	166
		8.3.3 I ² C 功能寄存器	166
		8.3.4 I ² C 接口配置	170
		8.3.5 接口函数	171

8.4 CAN 总线	176	第 10 章 LPC1100 系列处理器应用	
8.4.1 CAN 概述	176	开发实践	210
8.4.2 CAN 接口电路	180	10.1 SPI Flash 存储器的读写	210
8.4.3 CAN 功能寄存器		10.1.1 W25X 系列 Flash 存储器	
描述	180	简介	210
8.4.4 CAN 接口配置	188	10.1.2 W25X 系列 Flash 存储器	
8.4.5 接口函数	189	功能描述	210
习题 8	196	10.1.3 W25X 系列 Flash 存储器	
第 9 章 LPC1100 系列处理器 A/D		的应用	211
转换器及其应用	197	10.2 波形发生器的设计	215
9.1 概述	197	10.2.1 程序流程	216
9.1.1 A/D 转换器的分类	197	10.2.2 波形发生器的实现	217
9.1.2 A/D 转换器的主要性能		10.3 矩阵键盘与显示	220
指标	198	10.3.1 矩阵键盘扫描	221
9.1.3 LPC1100 系列处理器 A/D		10.3.2 按键扫描	223
转换器特性	199	10.3.3 键盘工作方式	227
9.2 A/D 转换器接口电路	199	10.4 温度采集	228
9.3 A/D 转换器功能寄存器描述	199	10.4.1 LM75A 的功能介绍	228
9.3.1 寄存器总汇	199	10.4.2 LM75A 的工作模式	230
9.3.2 寄存器描述	200	10.4.3 LM75A 应用	230
9.4 A/D 转换器配置	203	习题 10	232
9.4.1 A/D 时钟配置	203	附录 A LPC1100 系列芯片各封装	
9.4.2 A/D 中断配置	203	引脚图	233
9.5 应用例程	203	附录 B Cortex-M0 指令系统	236
9.5.1 A/D 初始化	203	附录 C LPC1100 微处理器汇编启动	
9.5.2 A/D 数据采集	205	代码	238
9.5.3 A/D 中断服务程序	206	附录 D LINPO-PS-LPC11xx 实验平台	
9.5.4 A/D 转换主函数	208	电路图	242
习题 9	209	参考文献	248

Unit 1 Introduction to Software Engineering

本章主要介绍软件工程的背景知识，包括软件和软件工程的定义，软件的特征和应用分类，软件危机，软件工程学习和研究的主要问题等。学习本章后应该理解为什么需要软件工程，掌握软件工程涉及的主要内容。

“Software engineering is about making things work (done), not just think things up.”

Learning Objectives

This unit is to introduce the background information of software engineering, including the origin of software engineering, formal definitions of software and software engineering, characteristics and categories of software, software crisis, and other major topics concerning of software engineering. Learning objectives include:

- Understand the concepts of software and software engineering
- Understand the characteristics and catalogs of software
- Understand the major topics introduced in this course

1.1 A Brief Introduction to Software Engineering

Software engineering has evolved steadily from its founding days in the 1940s until today in the 2010s. Its applications have evolved continuously. The ongoing goal of software engineering is to improve technologies and practices, seeks to improve the productivity of practitioners and the quality of applications to users. Most of content of this section is abstracted from wiki [wiki2011].

There are a number of areas where the evolution of software engineering is notable. People, process, product and project are four major factors affecting software engineering:

- Profession: By the early 1980s, software engineering had already emerged as a bona fide profession, to stand beside computer science and traditional engineering.
- Processes: Processes have become a big part of software engineering and are hailed for their potential to improve software and sharply criticized for their potential to constrict programmers.
- Product cost: The relative cost of software versus hardware has changed substantially over the last 50 years. When mainframes were expensive and required large support staffs, the few organizations buying them also had the resources to fund large, expensive custom software engineering projects. Computers are now much more numerous and much more powerful, which has much effects on software. The total cost of software can be much more than that of hardware.
- People: In the 1940s, 1950s, and 1960s, men often filled the more prestigious and better paying hardware engineering roles, but often delegated the writing of software to women.

1.1.2 嵌入式系统的特点

近年来掀起了嵌入式系统应用热潮的原因主要有两个方面:一方面是芯片技术的发展,使得单个芯片具有更强的处理能力,而且使集成多种接口已经成为可能,众多芯片生产厂商已经将注意力集中在这方面。另一方面的原因就是应用的需要,由于对产品可靠性、成本、更新换代要求的提高,使得嵌入式系统逐渐从纯硬件实现和使用通用计算机实现的应用中脱颖而出。

与通用计算机系统相比,嵌入式系统一般具有如下特点:

① 系统内核小。由于嵌入式系统一般是应用于小型电子装置的,系统资源相对有限,所以内核较之传统的操作系统要小得多。

② 专用性强。嵌入式系统的个性化很强,一般要针对硬件进行系统的移植,即使在同一品牌、同一系列的产品中也需要根据系统硬件的变化和增减不断进行修改。

③ 系统精简。嵌入式系统一般没有系统软件和应用软件的明显区分,不要求其功能设计及实现上过于复杂,这样利于控制系统成本,同时也利于实现系统安全。

④ 高实时性系统软件需求。嵌入式系统软件需要高实时性以实现任务调度、资源分配等功能。

⑤ 面向特定的应用。与通用 CPU 相比,嵌入式的 CPU 是为特定用户群设计的。如 ARM 系列 CPU 多用于手机产品开发。应用需求决定了嵌入式系统的设计,而决定嵌入式应用环境的主要因素是其提供的接口功能和处理速度。

⑥ 嵌入式系统需要专门的软、硬件开发工具和环境。由于其本身不具备自主开发能力,即使设计完成以后用户通常也是不能对其中的程序功能进行修改的,必须有一套开发工具和环境才能进行开发,这些工具和环境一般是基于通用计算机上的软、硬件设备,以及各种逻辑分析仪、混合信号示波器等。开发时往往有主机和目标机的概念,主机用于程序的开发,目标机作为最后的执行机,开发时需要交替结合进行。

1.1.3 嵌入式实时操作系统概述

操作系统的基本思想是隐藏底层不同硬件的差异,向在其上运行的应用程序提供一个统一的调用接口。应用程序通过这一接口实现对硬件的使用和控制,不必考虑不同硬件操作方式的差异。

嵌入式操作系统(Embedded Operation System, EOS)是一种专用于嵌入式设备的操作系统。EOS 负责嵌入系统的全部软、硬件资源的分配、任务调度,控制、协调并发活动。嵌入式操作系统在系统实时高效性、硬件的相关依赖性、软件固化及应用的专用性等方面具有较为突出的特点。嵌入式操作系统核心通常要求体积很小,因为硬件 ROM 容量有限,除了应用程序之外,不希望操作系统占用太大的存储空间。

嵌入式实时操作系统是嵌入式操作系统最主要的组成部分。实时操作系统是指具有实时性,能从硬件方面支持实时控制系统工作的操作系统。其中实时性是第一要求,需要调度一切可利用的资源完成实时控制任务,其次才着眼于提高计算机系统的使用效率,重点是要满足对时间的限制和要求。下面将介绍两种嵌入式实时操作系统。

1. μ C/OS-II

μ C/OS 是 Jean J. Labrosse 在 1990 年前后编写的一个嵌入式多任务实时操作系统。经过近 10 年的应用和修改,在 1999 年 Jean J. Labrosse 推出了 μ C/OS-II,并在 2000 年得到了

美国联邦航空管理局对用于商用飞机的、符合 RTCA /DO178B 标准的认证,从而证明 $\mu\text{C}/\text{OS-II}$ 具有足够的稳定性和安全性。

$\mu\text{C}/\text{OS-II}$ 是一个可裁减、源代码开放、结构小巧、可抢占式的实时多任务内核,是专为微控制器系统和软件开发而设计的,是控制器启动后首先执行的背景程序,并作为整个系统的框架贯穿系统运行的始终。它具有执行效率高、占用空间小、可移植性强、实时性能良好和可扩展性强等特点。采用 $\mu\text{C}/\text{OS-II}$ 实时操作系统可以有效地对任务进行调度;对各任务赋予不同的优先级可以保证任务及时响应,而且采用实时操作系统,降低了程序的复杂度,方便程序的开发和维护。

严格来讲, $\mu\text{C}/\text{OS-II}$ 只是一个实时操作系统内核,它仅仅包含了任务调度、任务管理、时间管理、内存管理和任务间通信与同步等基本功能,没有提供输入/输出管理、文件管理、网络等额外的服务。但由于 $\mu\text{C}/\text{OS-II}$ 良好的可扩展性和源码开放,这些功能完全可以由设计人员根据需求自己实现。

$\mu\text{C}/\text{OS-II}$ 获得广泛使用不仅是因为它的源码开放,还有它较强的可移植性。 $\mu\text{C}/\text{OS-II}$ 的代码大部分是用 C 编写的,只有与处理器硬件相关的一部分代码是用汇编编写的。 $\mu\text{C}/\text{OS-II}$ 在最初设计时即考虑了系统的可移植性,为其广泛应用奠定了良好的基础。

2. 实时线程操作系统

实时线程操作系统(RT-Thread)是一款来自国内的开源实时操作系统,并且商业许可证非常宽松,由国内 RT-Thread 开发团队进行开发与维护。它不仅是一款高效、稳定的实时操作系统内核,也是一套面向嵌入式系统的软件平台。起初 RT-Thread 只有一个实时的内核,但在发展过程中,RT-Thread 实时操作系统得到了来自全国嵌入式开发工程师的鼎力支持,为 RT-Thread 添砖加瓦,现在它已慢慢变成一个完善的、全功能的操作系统。目前,RT-Thread 已经被国内十多所企业所采用,被证明是一款能够稳定持续运行的操作系统。

RT-Thread 不仅是一个单一的实时操作系统内核,它也是一个完整的应用系统,包含了嵌入式系统相关的各个组件:TCP/IP 协议栈,文件系统,图形用户界面 RTGUI,Finsh Shell 等。

RT-Thread 内核部分包括了 RT-Thread 的核心代码,包括对象管理器,线程管理及调度,线程间通信等的微小内核实现。内核 C 库是为了保证内核能够独立运作的一套小型 C 库(在 RealView MDK 等系统自带部分 C 库函数的情况下,这部分不会被使能)。CPU 及板级支持包包含了 RT-Thread 支持的各个平台移植代码,通常会包含两个汇编文件,一个是系统启动初始化文件,一个是线程进行上下文切换的文件,其他的都是 C 源文件。

RT-Thread 是基于面向对象的方式开发的实时内核,标准内核体积为 9KB 左右,主要有如下特点:

- ① 面向对象方式的实时核心(但依然保留了 C 语言的优雅、小巧风格);
- ② 默认 32 线程优先级的全抢占式实时内核(也可配置成 256 线程优先级),相同优先级线程时间片轮转调度;
- ③ 相同优先级线程实施时间片可配置的分时时间片轮转调度;
- ④ 线程间同步机制——信号量和防止优先级翻转的互斥锁;
- ⑤ 完善高效的线程间通信机制,包括邮箱、消息队列和事件;
- ⑥ 支持线程挂起和唤醒的固定内存块管理及线程安全的动态内存堆管理;
- ⑦ 向上层提供基于名字的统一接口设备驱动模型;

- ⑧ 支持各个内核对象(如信号量对象、消息队列对象)的裁减;
- ⑨ 支持可嵌套的中断;
- ⑩ 支持线程堆栈溢出检测。

1.2 嵌入式处理器

嵌入式处理器是嵌入式系统的核心,是控制、辅助系统运行的硬件单元。目前世界上具有嵌入式功能特点的处理器已经超过 1000 种,包括早期的 4 位处理器、8 位单片机以及当前受到广泛青睐的 32 位、64 位嵌入式 CPU。鉴于嵌入式系统广阔的发展前景,很多半导体制造商都大规模生产嵌入式处理器,并且公司自主设计处理器也成为了未来嵌入式领域的一大趋势,其中从单片机、DSP 到 FPGA 有着各式各样的品种,速度越来越快,性能越来越强,价格也越来越低。嵌入式处理器的寻址空间一般为 64KB~4GB,处理速度为 0.1~2000MIPS,封装从 8 个引脚到 208 个引脚不等。

1.2.1 嵌入式处理器的分类

嵌入式处理器大致可以分为以下 4 个大类。

1. 嵌入式微处理器(Micro Processor Unit, MPU)

嵌入式微处理器是由通用计算机中的 CPU 演变而来的。它的特征是具有 32 位以上的处理器,具有较高的性能,当然其价格也相应较高。在实际嵌入式应用中,只保留和嵌入式应用紧密相关的功能硬件,去除其他的冗余功能部分,这样就以最低的功耗和资源实现嵌入式应用的特殊要求。和工业控制计算机相比,嵌入式微处理器具有体积小、重量轻、成本低、可靠性高的优点。目前主要的嵌入式处理器类型有 Am186/88、386EX、SC-400、Power PC、63000、MIPS、ARM/StrongARM 系列等。其中 ARM/StrongARM 是专为手持设备开发的嵌入式微处理器,属于中档的价位。

MPU 分为复杂指令集计算机(CISC)和精简指令集计算机(RISC)。RISC 和 CISC 是目前设计制造微处理器的两种典型技术,为达到高效的目的,采用的方法不同。

CISC 内部为将较复杂的指令译码,也就是指令较长,分成几个微指令去执行,正是如此开发程序比较容易(指令多的缘故),但是指令复杂,执行工作效率较差,处理数据速度较慢,如 PC 中 Pentium 的结构都为 CISC CPU。

RISC 是精简指令集 CPU,指令位数较短,内部还有快速处理指令的电路,使得指令的译码与数据的处理较快,所以执行效率比 CISC 高,不过,必须经过编译程序的处理,才能发挥它的效率,如 IBM 的 Power PC 为 RISC CPU 的结构,CISCO 的 CPU 也是 RISC 的结构。

RISC 主要把精力放在常用指令上,不常用的功能通过组合指令实现,而 CISC 则由专用指令完成特定的功能。

2. 嵌入式微控制器(Micro Control Unit, MCU)

嵌入式微控制器的典型代表是单片机,从 20 世纪 70 年代末单片机出现到今天,虽然已经经过了 30 多年的历史,但这种 8 位的电子器件目前在嵌入式设备中仍然有着极其广泛的应用。单片机芯片内部集成 ROM/EPROM、RAM、总线、总线逻辑、定时/计数器、看门狗、I/O、串行口、脉宽调制输出、A/D、D/A、Flash RAM、EEPROM 等各种必要功能和外设。

和嵌入式微处理器相比,微控制器的最大特点是单片化,体积大大减小,从而使功耗和成

本下降、可靠性提高。微控制器是目前嵌入式系统工业的主流。微控制器的片上外设资源一般比较丰富,适合于控制,因此称为微控制器。

由于 MCU 低廉的价格,优良的功能,所以拥有的品种和数量最多,比较有代表性的包括 8051、MCS-251、MCS-96/196/296、P51XA、C166/167、68K 系列,以及 MCU 8XC930/931、C540、C541,并且有支持 I²C、CAN-Bus、LCD 及众多专用 MCU 和兼容系列。

3. 嵌入式 DSP(Embedded Digital Signal Processor, DSP)

DSP 处理器是专门用于信号处理方面的处理器,其在系统结构和指令算法方面进行了特殊设计,具有很高的编译效率和指令的执行速度。在数字滤波、FFT、谱分析等各种仪器上 DSP 获得了大规模的应用。

推动 DSP 发展的一个重要因素是嵌入式系统的智能化。智能化算法的运算量一般都比较较大,特别是向量运算、指针线性寻址等较多,而这些正是 DSP 具备的、其他嵌入式处理所没有的长处。

随着大规模集成电路技术发展,1982 年世界上诞生了首个 DSP 芯片,其运算速度比 MPU 快了几十倍,在语音合成和编码解码器中得到了广泛应用。至 20 世纪 80 年代中期,随着 CMOS 技术的进步与发展,第二代基于 CMOS 工艺的 DSP 芯片应运而生,其存储容量和运算速度都得到成倍提高,成为语音处理、图像硬件处理技术的基础。到 20 世纪 80 年代后期, DSP 的运算速度进一步提高,应用领域也从上述范围扩大到了通信和计算机方面。20 世纪 90 年代后, DSP 发展到了第五代产品,集成度更高,使用范围也更加广阔。

4. 片上系统(System on Chip, SoC)

超大规模集成电路(Very Large Scale Integration)设计的普及和半导体工艺的迅速发展,使在单块硅片上实现一个更复杂的系统成为可能,这就是所谓的片上系统(SoC)。

SoC 最大的特点是成功实现了软、硬件无缝结合,直接在处理器片内嵌入操作系统的代码模块。而且 SoC 具有极高的综合性,在一个硅片内部运用 VHDL 等硬件描述语言,实现一个复杂的系统。用户不需要再像传统的系统设计一样,绘制庞大复杂的电路板,一点点的连接焊制,只需要使用精确的语言,综合时序设计直接在器件库中调用各种通用处理器的标准,然后通过仿真之后就可以直接交付芯片厂商进行生产。

SoC 分为通用和专用两类。通用系列包括 Motorola 的 M-Core 和 Neuron、某些 ARM 系列器件、Echelon 等。专用 SoC 一般专用于某类系统中,不为一般用户所知。例如, NXP 的 Smart XA,将 XA 单片机内核和支持超过 2048 位复杂 RSA 算法的 CCU(Communication Control Unit)单元制作在一块硅片上,形成一个可加载 Java 或 C 语言的专用 SoC,可用于 Internet 安全领域。

1.2.2 ARM 微处理器

ARM(Advanced RISC Machines),既可以认为是一个公司的名字,也可以认为是对一类微处理器的通称,还可以认为是一种技术的名字。ARM 公司于 1991 年在英国剑桥成立, ARM 公司是专门从事基于 RISC 技术芯片设计开发的公司,作为知识产权供应商,本身不直接从事芯片生产,靠转让设计许可由合作公司生产各具特色的芯片。目前,采用 ARM 技术知识产权核的微处理器,即我们通常所说的 ARM 微处理器,已遍及工业控制、消费类电子产品、通信系统、网络系统、无线系统等各类产品市场,基于 ARM 技术的微处理器应用约占据了 32 位 RISC 微处理器 75%以上的市场份额,ARM 技术正在逐步渗入到我们生活的各个方面。

ARM 公司开发了很多系列的 ARM 处理器核。目前常用的 ARM 微处理器有 ARM7 系

列、ARM9 系列、ARM9E 系列、ARM10E 系列、ARM11 系列、SecurCore 系列、Intel 的 StrongARM 和 XScale、Cortex-M、Cortex-A、Cortex-R 等。其中，Cortex-M、Cortex-A、Cortex-R 系列符合 ARMv7 标准的处理器的指令集、内存模型和程序员模型。

1. ARM 微处理器的特点

- ① 体积小、低功耗、低成本、高性能；
- ② 支持 Thumb(16 位)/ARM(32 位)双指令集,能很好地兼容 8 位/16 位器件；
- ③ 大量使用寄存器,指令执行速度更快；
- ④ 大多数数据操作都在寄存器中完成；
- ⑤ 寻址方式灵活简单,执行效率高；
- ⑥ 指令长度固定。

2. ARM 微处理器的结构

(1) 体系结构

ARM 是 Advanced RISC Machines 的缩写,顾名思义,ARM 处理器也是一种精简指令集计算机,其简单的结构使 ARM 内核非常小,器件的功耗也非常低。而且它具有经典 RISC 的特点:

- ① 大的、统一的寄存器文件；
- ② 装载/保存结构,数据处理操作只针对寄存器的内容,而不直接对存储器进行操作；
- ③ 简单的寻址方式；
- ④ 统一和固定长度的指令域,简化了指令的译码。

ARM 体系结构从最初开发到现在有了巨大的改进,并仍在完善和发展。为了清楚地表达每个 ARM 应用实例所使用的指令集,ARM 公司定义了 7 种主要的 ARM 指令集体系结构版本,以版本号 v1~v7 表示。ARM 微处理器的核心及体系结构见表 1.1。

表 1.1 ARM 微处理器核心及体系结构

核 心	体 系 结 构
ARM1	v1
ARM2	v2
ARM2As, ARM3	v2a
ARM6, ARM600, ARM610, ARM7, ARM700, ARM710	v3
StrongARM, ARM8, ARM810	v4
ARM7TDMI, ARM710T, ARM720T, ARM740T, ARM9TDMI, ARM920T, ARM940T	v4T
ARM9E2S, ARM10TDMI, ARM1020E	v5TE
ARM1136J(F)2S, ARM1176J Z(F)2S, ARM11 MPCore	v6
ARM1156T2(F)2S	v6T2
ARM Cortex-M, ARM Cortex-R ,ARM Cortex-A	v7

v1 是 ARM1 使用的,只能寻址 26 位空间,目前已废弃不用。

v2 也是只能寻址 26 位空间,但是相对 v1 增加了乘法和协处理器支持,现在也不再使用。

v3 的寻址空间增大为 32 位,使用了当前程序状态寄存器 CPSR,并增加了程序状态保存寄存器 SPSR。另外还增加了两种处理器模式,以便更有效地处理异常。

v4 是 v3 的扩展,相对 v3 增加了多种加载/存储指令,并增加了特权模式。

v5 是在 v4 的基础上进行了修改,增加了指令,提高了切换效率,增加了软件断点功能。

v6 通过 SIMD 功能扩展,优化了应用系统的音频/视频的处理功能,将处理性能提高到原来的 4 倍。它支持多微处理器内核,适合使用电池供电的高性能的便携式设备。

v7 架构是在 v6 架构的基础上诞生的,采用了 Thumb-2 技术,并保持了对现有 ARM 解决方案的完整的代码兼容性。还采用了 NEON 技术,将 DSP 和媒体处理能力提高了近 4 倍,并支持改良的浮点运算,满足下一代 3D 图形、游戏物理应用及传统嵌入式控制应用的需求。

(2) 寄存器结构

ARM 处理器共有 37 个寄存器,被分为若干个组(BANK),这些寄存器包括程序计数器(PC 指针)在内的 31 个 32 位通用寄存器和 6 个用于标示 CPU 工作状态及程序运行状态的 32 位状态寄存器。

(3) 指令结构

ARM 微处理器在较新的体系结构中支持两种指令集:ARM 指令集和 Thumb 指令集。其中,ARM 指令为 32 位长度,Thumb 指令为 16 位长度。Thumb 指令集为 ARM 指令集的功能子集,但与等价的 ARM 代码相比较,可节省 30%~40%的存储空间,同时具备 32 位代码的所有优点。

3. ARM 微处理器的应用领域

(1) 工业控制

作为 32 位的 RISC 架构,基于 ARM 核的微控制器芯片不但占据了高端微控制器市场的大部分市场份额,同时也逐渐向低端微控制器应用领域扩展,ARM 微控制器的低功耗、高性价比,向传统的 8/16 位微控制器提出了挑战。特别是 Cortex 系列处理器的诞生,其代码密度和能效优势意味着它是各种应用中 8/16 位设备的高性价比换代产品。

(2) 无线通信

目前已有超过 85%的无线通信设备采用了 ARM 技术,ARM 以其高性能和低成本,在该领域的地位日益巩固。

(3) 网络应用

随着宽带技术的推广,采用 ARM 技术的 ADSL 芯片正逐步获得竞争优势。此外,ARM 在语音及视频处理上进行了优化,并获得广泛支持,也对 DSP 的应用领域提出了挑战。

(4) 消费类电子

ARM 技术在目前流行的数字音频播放器、数字机顶盒和游戏机中得到了广泛采用,如银行等的取号机也是采用 ARM 技术。

(5) 成像及安全产品

现在流行的数码相机和打印机中绝大部分采用 ARM 技术。手机中的 32 位 SIM 智能卡也采用了 ARM 技术。

1.2.3 Cortex-M0 处理器

ARM Cortex-M0 处理器是现有的体积最小、能耗最低和能效最高的 ARM 处理器,也是符合 ARMv6-M 标准的处理器的指令集、内存模型和程序员模型的 ARM 处理器。该处理器的硅面积小、能耗极低并且所需的代码量极少,这使得开发人员能够以 8 位的设备实现 32 位设备的性能,从而省略 16 位设备的研发步骤。Cortex-M0 处理器超低的门数也使得它可以部署在模拟和混合信号设备中。

1. Cortex-M0 处理器的特点

(1) 性价比高

Cortex-M0 的代码密度和能效优势意味着它是各种应用中 8/16 位设备的自然高性价比换代产品,同时保留与功能丰富的 Cortex-M3 处理器的工具和二进制向上兼容性。

(2) 超低的能耗

Cortex-M0 处理器在不到 12K 门的面积内能耗仅有 $85\mu\text{W}/\text{MHz}$,所凭借的是作为低能耗技术的领导者和创建超低能耗设备的主要推动者的无与伦比的 ARM 专门技术。

(3) 简单

指令只有 56 个,这样用户便可以快速掌握整个 Cortex-M0 指令集(如果需要);但其 C 语言友好体系结构意味着这并不是必需的。可供选择的具有完全确定性的指令和中断计时使得计算响应时间十分容易。

(4) 优化的连接性

设计为支持低能耗连接,如 Bluetooth Low Energy (BLE)、IEEE 802.15 和 Z-wave,这些模拟设备正在增加其数字功能,以有效地预处理和传输数据。

2. Cortex-M0 处理器的技术规范

ARM Cortex-M0 处理器采用 ARMv6-M(冯·诺依曼)体系结构,执行 Thumb 指令集,包括少量使用 Thumb-2 技术的 32 位指令,是高性能的 32 位 CPU,具有确定性的运算和低延迟的 3 阶段管道,整数运算速度达到 0.9DMIPS/MHz。

ARM Cortex-M0 处理器集成了睡眠状态和多电源域的支持,并使用基于构架的软件进行睡眠状态和功耗的控制。

嵌套向量中断控制器(NVIC)是 Cortex-M0 处理器不可或缺的部分,它为处理器提供了卓越的中断处理能力,中断延迟 16 个周期。Cortex-M0 处理器使用一个向量表,其中包含要为特定中断处理程序执行的函数的地址。接收中断时,处理器会从该向量表中提取地址。为了减少门数并增强系统灵活性,Cortex-M0 处理器使用一个基于堆栈的异常模型。

Cortex-M0 处理器的 CoreSight 调试与跟踪,支持 JTAG 和 2 针串行线调试(SWD),支持多处理器和实时跟踪。

Cortex-M0 处理器支持广泛的第三方工具,采用 Cortex 微控制器软件接口标准(CM-SIS),最大限度地增加了软件成果的重用,缩短了微控制器开发人员的学习过程。

基于 ARM Cortex 处理器的微控制器的软件开发可能比 8 位微控制器产品的开发容易得多。Cortex 处理器不但是完全可通过 C 语言进行编程的,而且还附带各种高级调试功能以帮助定位软件中的问题。除 MCU 开发工具包中包括的所有附加资源外,Internet 上还有大量示例和教程,其中许多来自基于 ARM 处理器的 MCU 供应商的网站。

习 题 1

- 1.1 请说说你对嵌入式系统的理解。
- 1.2 嵌入式系统应用为什么会成为热潮?
- 1.3 概述嵌入式处理器的分类。
- 1.4 概述 ARM 微处理器的结构。
- 1.5 Cortex-M0 处理器有哪些特点?
- 1.6 查找资料,概述 NVIC 对中断的处理。

第2章 LPC1100 系列处理器的硬件结构

LPC1100 系列微处理器具有高效的处理能力,这与其硬件结构配置密切相关。本章将对 LPC1100 系列处理器的硬件结构进行介绍,除了对总线结构、存储器及寄存器的介绍外,还对 LPC1100 系列处理器出色的电源管理功能及其特有的串行线调试(SWD)进行详细介绍,最后对 LPC1100 系列处理器的最小系统分模块进行详细介绍。

2.1 LPC1100 系列处理器的简介

2009 年 11 月,NXP 公司推出的 LPC1100 系列 Cortex-M0 处理器,它们是为嵌入式系统

应用而设计的高性能、低功耗的 32 位微处理器。该处理器是市场上定价最低的 32 位微控制器,其价值和易用性比现有的 8/16 位微控制器更胜一筹。该控制器性能卓越、简单易用、功耗低,更重要的是,它能显著降低所有 8/16 位应用的代码长度。

LPC1100 系列 Cortex-M0 微处理器的主频时钟高达 50MHz,拥有高性能的运算控制能力,每秒可执行 4500 多万条指令,而 8 位 MCU 每秒只能执行不到 100 万条指令,16 位 MCU 也只能每秒执行 300~500 万条指令;该处理器在高效地执行运算和控制任务的同时,还能实现低功耗,支持睡眠、深度睡眠和深度掉电 3 种低功耗模式,其平均电流不足 10mA。同时,它还拥有丰富的外设组件:高达 32KB 片内 Flash 程序存储器、8KB 片内 SRAM、一路 I²C(FM+),一路 RS-485/EIA-485 UART、两路 SSP、4 个通用定时器及多达 42 个通用 I/O 口。

2010 年,第一批面市的 LPC1100 系列处理器,有 4 种产品,分别是:LPC1111、LPC1112、LPC1113、LPC1114,能满足所有那些寻求用可扩展 ARM 架构来进行整个产品开发过程的 8/16 位用户,满足其产品开发无缝整合需求。2010 年 6 月,LPC1100 系列再添两款新器件:LPC11C12 和 LPC11C14,这两款器件针对控制局域网(CAN)2.0B 标准研制的控制器,可以满足工业和嵌入式网络应用需求。之后的 2011 年 1 月,NXP 公司宣布推出业界首款内嵌易用型片上 CANopen 驱动,集成高速 CAN 物理层收发器的微处理器 LPC11C22 和 LPC11C24。2011 年 2 月,NXP 公司宣布推出其基于 ARM Cortex-M0 处理器的 LPC1200 工业控制系列。LPC1200 进一步拓展了恩智浦 32 位 ARM 微控制器的产品范围,适用于工业和家庭自动化领域一系列广泛的工业应用,如白色家电、电机控制、功率转换和电源等。2011 年 4 月,NXP 公司宣布推出基于 ARM Cortex-M0 的低成本微处理器 LPC11Uxx,该系列采用高度灵活的全新 USB 架构和智能卡接口。本节将介绍 LPC1100 系列处理器的性能特点、基本结构和引脚功能等。

2.1.1 LPC1100 系列处理器的特点

LPC1100 系列 Cortex-M0 微处理器具有以下主要特性。

(1) 系统部件

- Cortex-M0 内核工作频率高达 50MHz;