Linux

驱动开发入门与实战

（第3版）

郑强◎编著

清華大學出版社
北京

内 容 简 介

本书是获得大量读者好评的“Linux 典藏大系”中的《Linux 驱动开发入门与实战》（第 3 版）。本书内容充实，重点突出，实例丰富，实用性强，涵盖 Linux 驱动开发从基础知识到核心原理，再到应用实例的大部分核心知识。**本书专门提供教学视频、源代码、思维导图、习题参考答案和教学 PPT 等超值配套资料，可以帮助读者高效、直观地学习**。

本书共 19 章，分为 3 篇。第 1 篇“基础知识”涵盖 Linux 驱动开发概述、嵌入式处理器和开发板、构建嵌入式驱动程序开发环境、构建嵌入式 Linux 操作系统、构建第一个驱动程序、简单的字符设备驱动程序等内容；第 2 篇“核心技术”涵盖设备驱动的并发控制、设备驱动的阻塞和同步机制、中断与时钟机制、内外存访问等内容；第 3 篇“应用实战”涵盖设备驱动模型、RTC 实时时钟驱动程序、看门狗驱动程序、IIC 设备驱动程序、LCD 设备驱动程序、触摸屏设备驱动程序、输入子系统设计、块设备驱动程序、USB 设备驱动程序等内容。

本书适合所有想系统学习 Linux 驱动开发的入门与进阶人员阅读，也适合从事驱动开发的工程师阅读，还适合高等院校相关专业的学生和培训机构的学员作为学习用书。

图书在版编目（CIP）数据

Linux 驱动开发入门与实战 / 郑强编著. —3 版. —北京：清华大学出版社，2024.2
（Linux 典藏大系）
ISBN 978-7-302-65480-3

Ⅰ. ①L… Ⅱ. ①郑… Ⅲ. ①Linux 操作系统 Ⅳ. ①TP316.89

中国国家版本馆 CIP 数据核字（2024）第 043316 号

责任编辑：王中英
封面设计：欧振旭
责任校对：胡伟民
责任印制：沈 露

出版发行：清华大学出版社
网 址：https://www.tup.com.cn, https://www.wqbook.com
地 址：北京清华大学学研大厦 A 座 邮 编：100084
社 总 机：010-83470000 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
印 装 者：北京同文印刷有限责任公司
经 销：全国新华书店
开 本：185mm×260mm **印 张**：26 **字 数**：655 千字
版 次：2011 年 1 月第 1 版 2024 年 3 月第 3 版 **印 次**：2024 年 3 月第 1 次印刷
定 价：109.00 元

产品编号：101191-01

前言

Linux 驱动程序开发一直是一个非常热门的话题，大多数基于 Linux 操作系统的嵌入式系统都需要编写驱动程序。随着嵌入式系统的广泛应用，出现了越来越多的硬件产品必须编写驱动程序，以便让设备在 Linux 操作系统上能正常工作。但是 Linux 驱动程序开发难度较高，高水平的开发人员也较少，这导致驱动程序开发跟不上硬件的发展速度。基于此原因，笔者编写了本书，希望能帮助那些学习 Linux 驱动开发的相关人员尤其是入门人员，让他们更容易理解相关知识，从而能系统、高效地掌握 Linux 驱动程序开发的核心技术。

本书是获得大量读者好评的“Linux 典藏大系”中的《Linux 驱动开发入门与实战》（第 3 版）。本书在第 2 版的基础上进行了全面改版，升级了 Linux 系统的开发环境和内核版本，并对第 2 版中的一些疏漏进行了修订，也对书中的一些实例和代码重新进行了表述，从而更加易读。相信读者可以在第 3 版的引领下，轻松跨入 Linux 驱动开发的大门，最终成为一名驱动程序开发的高手。

关于“Linux 典藏大系”

“Linux 典藏大系”是专门为 Linux 技术爱好者推出的系列图书，涵盖 Linux 技术的方方面面，可以满足不同层次和各个领域的读者学习 Linux 的需求。该系列图书自 2010 年 1 月陆续出版，上市后深受广大读者的好评。2014 年 1 月，创作者对该系列图书进行了全面改版并增加了新品种。新版图书一上市就大受欢迎，各分册长期位居 Linux 图书销售排行榜前列。截至 2023 年 10 月底，该系列图书累计印数超过 30 万册。可以说，“Linux 典藏大系”是图书市场上的明星品牌，该系列中的一些图书多次被评为清华大学出版社“年度畅销书”，还曾获得“51CTO 读书频道”颁发的“最受读者喜爱的原创 IT 技术图书奖”，另有部分图书的中文繁体版在中国台湾出版发行。该系列图书的出版得到了国内 Linux 知名技术社区 ChinaUnix（简称 CU）的大力支持和帮助，读者与 CU 社区中的 Linux 技术爱好者进行了广泛的交流，取得了良好的学习效果。另外，该系列图书还被国内上百所高校和培训机构选为教材，得到了广大师生的一致好评。

关于第 3 版

随着技术的发展，本书第 2 版与当前 Linux 的几个流行版本有所脱节，这给读者的学习带来了不便。应广大读者的要求，笔者结合 Linux 技术的新近发展对第 2 版图书进行全面的升级改版，推出第 3 版。相比第 2 版图书，第 3 版在内容上的变化主要体现在以下几个方面：

- Linux 系统更新为 Ubuntu 22.04；
- Linux 内核版本由 Linux 2.6.34 更新为 Linux 5.15；
- 对一些函数及其形式进行了修改；
- 去除了一些弃用的函数；
- 新增对多队列 blk-mq 技术的讲解；
- 新增思维导图（提供电子版高清大图）和课后习题，以方便读者学习。

本书特色

1．配多媒体教学视频，学习效果好

本书重点内容提供大量的配套多媒体教学视频，可以帮助读者更加轻松、直观、高效地学习，取得更好的学习效果。

2．基于 Linux 5.15 内核讲解，内容新颖

本书基于 Linux 5.15 内核进行讲解。该内核是目前较为流行的 Linux 内核，它内置了大多数常用设备的驱动程序，便于读者学习和移植。

3．内容全面、系统、深入

本书全面、系统、深入地介绍 Linux 驱动开发的基础知识、核心技术和多个驱动程序开发案例，力求让读者通过一本书即可掌握 Linux 驱动开发的相关知识。

4．讲解由浅入深、循序渐进，适合各个层次的读者阅读

本书从 Linux 驱动程序开发的基础知识开始讲解，逐步深入 Linux 驱动开发的高级技术，内容安排从易到难，讲解由浅入深、循序渐进，适合各个层次的读者阅读。

5．贯穿多个典型实例和技巧，迅速提升开发水平

本书在讲解知识点时穿插多个典型的驱动程序开发实例，并给出大量的开发技巧，以便让读者更好地理解核心概念和知识点，并体验实际开发，从而迅速提高开发水平。

6．详解典型工程应用案例，有很强的实用性

本书详细介绍多个驱动开发典型应用案例。通过这些案例，可以提高读者的设备驱动开发水平，从而具备独立开发驱动程序的能力。

7．提供习题、源代码、思维导图和教学 PPT

本书特意在每章后提供多道习题，帮助读者巩固和自测该章的重要知识点，并提供源代码、思维导图和教学 PPT 等配套资源，以方便读者学习和老师教学。

本书内容

第 1 篇　基础知识

本篇涵盖第 1～6 章，主要内容包括 Linux 驱动开发概述、嵌入式处理器和开发板、构建嵌入式驱动程序开发环境、构建嵌入式 Linux 操作系统、构建第一个驱动程序、简单的字符设备驱动程序。通过学习本篇内容，读者可以掌握 Linux 驱动开发的基本概念、环境和注意事项等内容。

第 2 篇　核心技术

本篇涵盖第 7～10 章，主要内容包括设备驱动的并发控制、设备驱动的阻塞和同步机制、中断与时钟机制、内外存访问等。通过学习本篇内容，读者可以掌握 Linux 设备驱动开发的基础知识和核心技术。

第 3 篇　应用实战

本篇涵盖第 11～19 章，主要内容包括设备驱动模型、RTC 实时时钟驱动程序、看门狗驱动程序、IIC 设备驱动程序、LCD 设备驱动程序、触摸屏设备驱动程序、输入子系统设计、块设备驱动程序、USB 设备驱动程序等。通过学习本篇内容，读者可以掌握编写常见设备驱动程序的方法。

读者对象

- Linux 驱动开发入门人员；
- Linux 驱动程序专业开发人员；
- Linux 内核研究者；
- 嵌入式开发工程师；
- 大中专院校的学生；
- 技术培训机构的学员。

配书资源获取方式

本书涉及的配套资源如下：

- 配套教学视频；
- 程序源代码文件；
- 高清思维导图；
- 习题参考答案；
- 配套教学 PPT；
- 书中涉及的开发工具。

上述配套资源有以下 3 种获取方式：

- 关注微信公众号“方大卓越”，然后回复数字“5”，即可自动获取下载链接；
- 在清华大学出版社网站（www.tup.com.cn）上搜索到本书，然后在本书页面上找到“资源下载”栏目，单击“网络资源”按钮进行下载；
- 还可在本书技术论坛（www.wanjuanchina.net）上的 Linux 模块进行下载。

技术支持

虽然笔者对书中所述内容都尽量予以核实，并多次进行文字校对，但因时间所限，可能还存在疏漏和不足之处，恳请读者批评与指正。

读者在阅读本书时若有疑问，可以通过以下方式获得帮助：

- 加入本书 QQ 交流群（群号：302742131）进行提问；
- 在本书技术论坛（网址见上文）上留言，会有专人负责答疑；
- 发送电子邮件到 book@ wanjuanchina.net 或 bookservice2008@163.com 获得帮助。

编　者

2024 年 2 月

目录

第 1 篇　基础知识

第 2 篇　核心技术

第 3 篇 应用实战

第 1 篇
基础知识

第 1 章　Linux 驱动开发概述

设备驱动程序是计算机硬件与应用程序的接口，是软件系统与硬件系统沟通的桥梁。如果没有设备驱动程序，那么硬件设备就只是一堆废铁，没有什么功能。本章将对 Linux 驱动开发进行简要的介绍，让读者对一些常见的概念有所了解。

1.1　Linux 设备驱动基础知识

刚接触 Linux 设备驱动的读者可能会对 Linux 设备驱动中的一些基本概念不太理解。从而影响后续的学习，因此本节将集中讲解一些 Linux 设备驱动的基本概念，为进一步学习打下良好的基础。

1.1.1　设备驱动程序概述

设备驱动程序（Device Driver）简称驱动程序（Driver）或驱动。它是一个允许计算机软件（Computer Software）与硬件（Hardware）交互的程序。这种程序建立了一个硬件与硬件或硬件与软件沟通的界面。CPU 经由主板上的总线（Bus）或其他沟通子系统（Subsystem）与硬件形成连接，这样的连接使得硬件设备（Device）之间进行数据交换成为可能。

依据不同的计算机架构与操作系统平台差异，驱动程序可以是 8 位、16 位、32 位和 64 位。不同平台的操作系统需要不同的驱动程序。例如，32 位的 Windows 系统需要 32 位的驱动程序，64 位的 Windows 系统需要 64 位驱动程序；在 Windows 3.11 的 16 位操作系统盛行时期，大部分的驱动程序都是 16 位；到 32 位的 Windows XP，则大部分是使用 32 位驱动程序；至于 64 位的 Linux 或 Windows 7、Windows 8、Windows 10、Windows 11 平台，则必须使用 64 位驱动程序。

1.1.2　设备驱动程序的作用

设备驱动程序是一种可以使计算机与设备进行通信的特殊程序，相当于硬件的接口。操作系统只有通过这个接口才能控制硬件设备的工作。如果某个设备的驱动程序未能正确安装，那么该设备便不能正常工作。正因为这个原因，驱动程序在系统中的地位十分重要。一般，操作系统安装完毕后，首先便是安装硬件设备的驱动程序。并且，当设备驱动程序有更新的时候，新的驱动程序比旧的驱动程序性能更优。这是因为新的驱动程序对内存和 I/O 等进行了优化，使硬件能够发挥更好的优势。

大多数情况下并不需要安装所有硬件设备的驱动程序。例如磁盘、显示器、光驱、键盘和鼠标等就不需要安装驱动程序，而显卡、声卡、扫描仪、摄像头和 Modem 等就需要安装驱动程序。不需要安装驱动程序并不代表这些硬件不需要驱动程序，而是这些设备所需的驱动程序已经内置在操作系统中。另外，不同版本的操作系统对硬件设备的支持也是不同的，一般情况下，操作系统的版本越高，其支持的硬件设备也越多。

设备驱动程序用来将硬件本身的功能告诉操作系统（通过提供接口的方式），完成硬件设备电子信号与操作系统及软件的高级编程语言之间的互相翻译。当操作系统需要使用某个硬件时，如让声卡播放音乐，它会先发送相应指令到声卡的某个 I/O 端口，声卡驱动程序从该 I/O 端口接收到数据后，马上将其翻译成只有声卡才能听懂的电子信号命令，从而让声卡播放音乐。因此简单地说，驱动程序为硬件到操作系统提供了一个接口，并且负责协调二者之间的关系。因为驱动程序有如此重要的作用，所以有“驱动程序是硬件的灵魂、硬件的主宰”这种说法，同时，驱动程序也被形象地称为“硬件和系统之间的桥梁”。

1.1.3　设备驱动的分类

计算机系统的主要硬件由 CPU、存储器和外部设备组成。驱动程序的对象一般是存储器和外部设备。随着芯片制造工艺的提高，为了节约成本，通常将很多原属于外部设备的控制器嵌入 CPU 内部。例如，Intel 的酷睿 i5 3450 处理器就内置了 GPU 单元，配合“需要搭配内建 GPU 的处理器”的主板，就能够起到显卡的作用，相比独立显卡，其在性价比上有明显的优势。因此，目前的驱动程序基本都支持 CPU 中的嵌入控制器。Linux 将计算机硬件设备分为 3 大类，分别是字符设备、块设备和网络设备。

1．字符设备

字符设备是指那些能一个字节一个字节读取数据的设备，如 LED 灯、键盘和鼠标等。字符设备一般需要在驱动层实现 open()、close()、read()、write()、ioctl()等函数。这些函数最终将被文件系统中的相关函数调用。内核会为字符设备提供专用的操作文件（如字符设备文件/dev/console）实现对字符设备的操作。对字符设备的操作可以通过字符设备文件/dev/console 来完成。这些字符设备文件与普通文件没有太大的差别，差别之处是字符设备一般不支持寻址，但在特殊情况下，有很多字符设备也是支持寻址的。寻址是对硬件中的一块寄存器进行随机访问。不支持寻址就是只能对硬件中的寄存器进行顺序读取，读取数据后，由驱动程序自己分析需要哪一部分数据。

2．块设备

块设备与字符设备类似，一般是磁盘类设备。在块设备中还可以容纳文件系统并存贮大量的信息，如 U 盘、SD 卡。在 Linux 系统中进行块设备读写时，每次只能传输一个或者多个块。Linux 可以让应用程序像访问字符设备一样访问块设备，一次只读取一个字节。因此块设备从本质上看更像一个字符设备的扩展，它能完成很多工作，如传输一块数据。

综合来说，相比字符设备，块设备要求更复杂的数据结构，其内部实现也不一样。在 Linux 内核中，与字符设备驱动程序相比，块设备驱动程序具有完全不同的 API 接口。

3．网络设备

计算机连接到互联网上需要一个网络设备，网络设备主要负责主机之间的数据交换。与字符设备和块设备完全不同，网络设备主要是面向数据包的接收和发送而设计的。网络设备在 Linux 操作系统中是一种非常特殊的设备，其没有实现类似块设备和字符设备的 read()、write()和 ioctl()等函数，而是实现了一种套接字接口，任何网络数据传输都可以通过套接字来完成。

1.2　Linux 操作系统与驱动的关系

Linux 操作系统与设备驱动之间的关系如图 1.1 所示。用户空间包括应用程序和系统调用两层。应用程序一般依赖于函数库，而函数库是由系统调用编写的，因此应用程序间接地依赖于系统调用。

系统调用层是内核空间和用户空间的接口层，就是操作系统提供给应用程序最底层的 API。通过这个系统调用层，应用程序不需要直接访问内核空间的程序，增加了内核的安全性。同时，应用程序也不能直接访问硬件设备，只能通过系统调用层来访问硬件设备。如果应用程序需要访问硬件设备，则先访问系统调用层，由系统调用层访问内核层的设备驱动程序。这样的设计保证了各个模块的功能独立性，也保证了系统的安全。

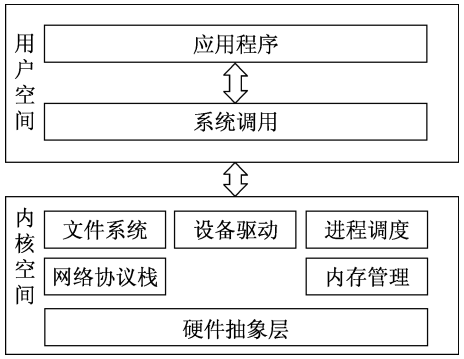

图 1.1　Linux 操作系统与设备驱动程序的关系

系统调用层依赖内核空间的各个模块来实现。在 Linux 内核中包含很多实现具体功能的模块，如文件系统、网络协议栈、设备驱动、进程调度和内存管理等，这些模块是在内核空间实现的。

最底层是硬件抽象层，这一层是实际硬件设备的抽象。设备驱动程序的功能就是驱动这一层的硬件。设备驱动程序可以工作在有操作系统的场景下，也可以工作在没有操作系统的场景下。如果只需要实现一些简单的控制设备操作，那么可以不使用操作系统。如果嵌入式系统完成的功能比较复杂，往往需要操作系统来帮忙。例如，单片机程序就不需要操作系统，因为其功能简单，内存、处理器能力弱，没有必要为其开发操作系统。

1.3　Linux 驱动程序开发简介

Linux 驱动程序的开发与应用程序的开发有很大的差别。这些差别导致编写 Linux 设备驱动程序与编写应用程序有本质的区别，因此对于应用程序的设计技巧很难直接应用在驱动程序开发上。最经典的例子是应用程序如果发生错误，可以通过 try catch 等方式避免程序崩溃，而驱动程序就没有这么方便的处理方法了。本节将对 Linux 驱动程序的开发进

行简要的讲解。

1.3.1　用户态和内核态

Linux 操作系统分为用户态和内核态。用户态处理上层的软件工作；内核态用来管理用户态的程序，完成用户态请求的工作。驱动程序与底层的硬件交互，因此工作在内核态。

简单来说，内核态大部分时间在完成与硬件的交互，比如读取内存，将磁盘上的数据读取到内存中，调度内存中的程序到处理器中运行等。相对于内核态，用户态则自由得多，其实，用户态中的用户可以狭隘地理解为应用程序开发者，他们很少与硬件直接打交道，他们的工作一般是编写 Java 虚拟机中的应用程序，或者.NET 框架中的应用程序。即使他们的编程水平不高，经常出现程序异常，充其量也是使 Java 虚拟机崩溃，想让操作系统崩溃还是很难的。这一切都归功于内核态对操作系统有很强大的保护能力。

另一方面，Linux 操作系统分为两个状态的原因是为应用程序提供一个统一的计算机硬件抽象。工作在用户态的应用程序完全可以不考虑底层的硬件操作，这些操作由内核态程序来完成。这些内核态程序大部分是设备驱动程序。一个好的操作系统的驱动程序对用户态应用程序应该是透明的，也就是说，应用程序可以在不了解硬件工作原理的情况下很好地操作硬件设备，并且不会使硬件设备进入非法状态。Linux 操作系统很好地做到了这一点。在 Linux 编程中，程序员经常使用 open()方法读取磁盘中的数据，调用这个方法时，无须关心磁盘控制器怎么读取数据并将其传递到内存中。这些工作都是驱动程序完成的，这就是驱动程序的透明性。

值得注意是，工作在用户态的应用程序不能因为一些错误而破坏内核态的程序。现代处理器已经充分考虑了这个问题。处理器提供了一些指令，分为特权指令和普通指令。特权指令只有在内核态下才能使用；普通指令既可以在内核态下使用，也可以在用户态下使用。通过这种限制，用户态程序就不能执行只有在内核态下才能执行的程序了，从而起到保护的作用。

另外值得注意的是，用户态和内核态是可以互相转换的。每当应用程序执行系统调用或者被硬件中断挂起时，Linux 操作系统都会从用户态切换到内核态。当系统调用完成或者中断处理完成后，操作系统会从内核态返回用户态，继续执行应用程序。

1.3.2　模块机制

模块在运行时可以加入内核代码，这是 Linux 的一个很好的特性。这个特性使内核可以很容易地扩大或缩小，一方面扩大内核可以增加内核的功能，另一方面缩小内核可以减小内核的大小。

Linux 内核支持很多种模块，驱动程序就是其中最重要的一种，甚至文件系统也可以写成一个模块，然后加入内核。每一个模块由编译好的目标代码组成，可以使用 insmod（insert module 的缩写）命令将模块加入正在运行的内核，也可以使用 rmmod（remove module 的缩写）命令将一个未使用的模块从内核中删除。如果删除一个正在使用的模块，则是不允许的。对 Windows 熟悉的朋友，可以将模块理解为 DLL 文件。

模块在内核启动时装载称为静态装载，在内核已经运行时装载称为动态装载。模块可

以扩充内核所期望的任何功能，但通常用于实现设备驱动程序。一个模块的最基本框架代码如下：

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/init.h>

int __init xxx_init(void)
{
    /*这里是模块加载时的初始化工作*/
    return 0;
}

void __exit xxx_exit(void)
{
    /*这里是模块卸载时的销毁工作*/
}
module_init(xxx_init);                    /*指定模块的初始化函数的宏*/
module_exit(xxx_exit);                    /*指定模块的卸载函数的宏*/
```

1.3.3　编写设备驱动程序需要了解的知识

目前，Linux 操作系统有七八百万行代码，其中，驱动程序代码就占四分之三左右。所以对于驱动程序开发人员来说，学习和编写设备驱动程序都是一个漫长的过程。在这个过程中，应该掌握以下知识：

（1）驱动程序开发人员应该有良好的 C 语言基础，并能灵活地应用 C 语言的结构体、指针、宏等基本语言结构。另外，Linux 系统使用的 C 编译器是 GNU C 编译器，因此对 GNU C 标准的 C 语言也应该有所了解。

（2）驱动程序开发人员应该有良好的硬件基础。虽然不要求驱动开发人员具有设计电路的能力，但是应该对芯片手册上描述的接口设备非常了解。常用的设备有 SRAM、Flash、UART、IIC（也称 I2C）和 USB 等。

（3）驱动程序开发人员应该对 Linux 内核源代码有所了解，如一些重要的数据结构和函数等。

（4）驱动程序开发人员应该具有多任务程序设计的能力，并且会使用自旋锁、互斥锁等。

本书的内容基本涵盖上述几方面，读者通过对本书的学习可以快速掌握这些知识。

1.4　编写设备驱动程序的注意事项

大部分程序员都比较熟悉应用程序的编写，但是对于驱动程序的编写可能不是很熟悉。关于应用程序的很多编程经验不能直接应用到驱动程序的编写中，下面给出编写驱动程序的一些注意事项，希望引起读者的注意。

1.4.1　应用程序开发与驱动程序开发的差异

Linux 中的程序开发一般分为两种，一种是内核及驱动程序开发，另一种是应用程序

开发。这两种开发种类对应 Linux 的两种状态，分别是内核态和用户态。内核态管理用户态的程序，完成用户态请求的工作；用户态处理上层的软件工作。驱动程序与底层的硬件交互，因此工作在内核态。

大多数程序员致力于应用程序的开发，少数程序员则致力于内核及驱动程序的开发。相比于应用程序开发，内核及驱动程序开发有明显差别，主要包括以下几点：

- 在进行内核及驱动程序开发时必须使用 GNU C，因为 Linux 操作系统从一开始就使用的就是 GNU C，虽然也可以使用其他编译工具，但是需要对以前的代码做大量的修改。需要注意的是，32 位的机型和 64 位的机型在编译时也有一定的差异。
- 在进行内核及驱动程序开发时不能访问 C 库，因为 C 库是使用内核中的系统调用来实现的，而且是在用户空间实现的。驱动程序只能访问有限的系统调用或者汇编程序。
- 在进行内核及驱动程序开发时缺乏像用户空间这样的内存保护机制，稍不注意就可能读写其他程序的内存。
- 内核只有一个很小的定长堆栈。
- 在进行内核及驱动程序开发时要考虑可移植性，因为对于不同的平台，驱动程序是不兼容的。
- 内核支持异步中断、抢占和 SMP，因此在进行内核及驱动程序开发时必须时刻注意同步和并发。
- 在进行内核及驱动程序开发时浮点数很难使用，应该使用整型数。

1.4.2　使用 GUN C 开发驱动程序

GUN C 语言最早起源于一个 GUN 计划，GUN 的意思是 GUN is not UNIX。GUN 计划开始于 1984 年，这个计划的目的是开发一个类似 UNIX 并且软件自由的完整操作系统。这个计划一直在进行，到 Linus 开发 Linux 操作系统时，GNU 计划已经开发出来了很多高质量的自由软件，其中就有著名的 GCC 编译器，GCC 编译器能够编译 GUN C 语言。Linus 考虑到 GUN 计划的自由和免费，因此选择了 GCC 编译器来编写内核代码，之后的很多开发者也使用这个编译器，直到现在，驱动开发人员依旧使用 GUN C 语言来开发驱动程序。

1.4.3　不能使用 C 函数库开发驱动程序

与用户空间的应用程序不同，内核不能调用标准的 C 函数库，主要原因在于对于内核来说完整的 C 函数库太大了。一个编译的内核大小可以是 1MB 左右，而一个标准的 C 语言库大小可能操作 5MB。这对于存储容量较小的嵌入式设备来说，是不实用的。缺少标准 C 语言库，并不是说驱动程序就只能做很少的事情。因为标准 C 语言库是通过系统调用实现的，驱动也是通过系统调用等实现的，两者都有相同的底层，因此驱动程序不需要调用 C 语言库也能实现很多功能。

大部分常用的 C 库函数在内核中都已经实现了。例如，操作字符串的函数就位于内核文件 lib/string.c 中，只要包含<linux/string.h>，就可以使用它们；又如，内存分配的函数也包含在 include/linux/slab_def.h 中实现了。

注意：内核程序中包含的头文件是指内核代码树中的内核头文件，不是指开发应用程序时的外部头文件。在内核中实现的库函数的打印函数 printk()，是 C 库函数 printf()的内核版本。printk()函数和 printf()函数的用法和功能基本相同。

1.4.4　没有内存保护机制

当一个用户应用程序由于编程错误，试图访问一个非法的内存空间时，操作系统内核会结束这个进程并返回错误码。应用程序可以在操作系统内核的帮助下恢复原状，而且应用程序并不会对操作系统内核有太大的影响。如果操作系统内核访问了一个非法内存，那么就有可能破坏内核的代码或者数据。这将导致内核处于未知状态，内核会通过 oops 错误给用户一些提示，但是这些提示都是不支持或者难以分析的。

在内核编程过程中，不应该访问非法内存，特别是空指针，否则内核会忽然“死掉”，没有任何机会向用户提示。对于不好的驱动程序，引起系统崩溃是很常见的，因此驱动开发人员应该重视对内存的正确访问。一个好的建议是，在申请内存后，应该对返回的地址进行检测。

1.4.5　小内核栈

用户空间的程序可以从栈上分配大量的空间来存放变量，甚至用栈存放大型的数据结构或者数组也没问题。之所以能这样做是因为应用程序内存是非常驻的，它们可以动态地申请和释放所有可用的内存空间。而内核要求使用固定常驻的内存空间，因此要求尽量少地占用常驻内存，应尽量多地留出内存提供给用户程序使用。因此内核栈的长度是固定大小且不可动态增长的，32 位机的内核栈是 8KB，64 位机的内核栈是 16KB。

由于内核栈比较小，所以编写程序时应该充分考虑小内核栈问题。尽量不要使用递归调用，在应用程序中，递归调用 4000 多次就有可能溢出，在内核中，递归调用的次数非常少，几乎不能完成程序的功能。另外，使用完内存空间后应该尽快地释放内存，以防止资源泄漏，引起内核崩溃。

1.4.6　重视可移植性

对于用户空间的应用程序来说，可移植性一直是一个重要的问题。一般，可移植性通过两种方式来实现。一种方式是定义一套可移植的 API，然后对这套 API 在这两个需要移植的平台上分别实现。应用程序开发人员只要使用这套可移植的 API，就可以写出可移植的程序。在嵌入式领域，比较常见的 API 套件是 QT。另一种方式是使用类似 Java、Actionscript 等可移植到很多操作系统上的语言。这些语言一般通过虚拟机执行，因此可以移植到很多平台上。

对于驱动程序来说，可移植性需要注意以下几个问题：

- 考虑字节顺序，一些设备使用大端字节序，一些设备使用小端字节序。Linux 内核提供了大小端字节序转换的函数，具体如下：

```
#define cpu_to_le16(val) (val)
#define cpu_to_le32(val) (val)
#define cpu_to_le64(val) (val)
#define le16_to_cpu(val) (val)
#define le32_to_cpu(val) (val)
#define le64_to_cpu(val) (val)
```

- 即使是同一种设备驱动程序，如果使用的芯片不同，也应该编写不同的驱动程序，但是应该给用户提供一个统一的编程接口。
- 尽量使用宏代替设备端口的物理地址，并且可以使用 ifdefine 宏确定版本等信息。
- 针对不同的处理器，应该使用相关的处理器函数。

1.5　Linux 驱动的发展趋势

随着嵌入式技术的发展，使用 Linux 的嵌入式设备也越来越多，特别是 Android 设备，相应的，工业上对 Linux 驱动开发也越来越重视。本节将对 Linux 驱动的发展进行简要的介绍。

1.5.1　Linux 驱动的发展前景

Linux 和嵌入式 Linux 软件在过去几年里已经被越来越多的 IT、半导体、嵌入式系统等公司认可和接受，它已经成为一个可以替代微软的 Windows 和众多传统的 RTOS 的重要的操作系统。Linux 内核和基本组件及工具已经非常成熟。面向行业、应用和设备的嵌入式 Linux 工具软件和嵌入式 Linux 操作系统平台是未来发展的必然趋势。符合标准，遵循开放是大势所趋，人心所向，嵌入式 Linux 也不例外。

使嵌入式 Linux 不断发展的一个核心条件是提供大量的稳定和可靠的驱动程序。每天都有大量的芯片被生产出来，芯片的设计和原理不一样，因此驱动程序就不一样。这样，就需要大量的驱动程序开发人员开发驱动程序，可以说，Linux 驱动程序的发展前景是很光明的。

1.5.2　驱动的应用

计算机系统已经融入各行各业和各个领域；计算机系统在电子产品中无处不在，从手机、游戏机、冰箱、电视、洗衣机，到汽车、轮船、火车、飞机等都有它的身影。这些产品都需要驱动程序使之运行，可以说驱动程序的运用前景是非常广泛的。每天都有很多驱动程序需要编写，因此驱动程序开发人员的前途是无比光明的。

1.5.3　相关学习资源

学习 Linux 设备驱动程序，只学习理论是不够的，还需要亲自动手编写各种设备的驱动程序。编写驱动程序不仅需要软件知识，还需要硬件知识。这里，笔者推荐一些国内外优秀的驱动开发网站，希望对读者的学习有所帮助。

- Linux 内核之旅网站：http://www.kerneltravel.net/；
- 知名博客：http://www.lupaworld.com/；
- 一个不错的 Linux 中文社区：https://linux.cn/；
- Linux 伊甸园：http://www.linuxeden.com/。

1.6　小　　结

本章首先对 Linux 设备驱动程序的基本概念进行了详细的讲述，并且讲述了设备驱动程序的作用；接着讲述了设备驱动程序的分类、特点及其与操作系统的关系等；然后讲述了驱动程序开发的一些重要知识和一些注意事项；最后讲述了 Linux 驱动程序的发展趋势。通过本章的学习，读者可以对 Linux 设备驱动程序的开发有一个大概的了解。

随着嵌入式设备的出现，有越来越多的驱动程序需要程序员去编写，因此学习驱动程序的开发对个人的发展是非常有帮助的。

1.7　习　　题

一、填空题

1．设备驱动程序简称_____。
2．用户空间包括_____和系统调用两层。
3．GUN C 语言最早起源于一个_____计划。

二、选择题

1．在驱动程序的分类中不包含（　　）。
A．字符设备　　B．块设备　　C．网络设备　　D．输入设备
2．应用程序开发与驱动程序开发的差异不包含（　　）。
A．在进行内核及驱动程序开发时不能访问 C 库。
B．在进行内核及驱动程序开发时必须使用 GNU C。
C．在进行内核及驱动程序开发时必须使用浮点数。
D．在进行内核及驱动程序开发时要考虑可移植性。
3．将模块加入正在运行的内核的命令是（　　）。
A．insert module　　B．insmod　　C．add　　D．insert

三、判断题

1．在处理器提供的指令中特权指令既可以在内核态下使用，也可以在用户态下使用。（　　）
2．内核支持异步中断、抢占和 SMP。（　　）
3．模块在内核启动时装载称为动态装载。（　　）

第 2 章　嵌入式处理器和开发板

在实际的工程项目中，Linux 驱动程序一般是为嵌入式系统而写的。嵌入式系统因用途、功能和设计厂商不同，硬件之间存在很大的差异。这些差异，不能通过编写一个通用的驱动程序来解决，需要针对不同的设备编写不同的驱动程序。要编写驱动程序，必须了解处理器和开发板的相关信息，本章将对这些信息进行详细讲解。

2.1　处　理　器

处理器的内部控制器不一样，其驱动程序的开发也是不一样的。例如，Intel 和 ARM 的处理器，它们的驱动开发就不一样。本节将对处理器的概念进行简要的讲解，并介绍一些常用的处理器种类，以使读者对嵌入式系统的处理器有初步的认识。

2.1.1　处理器简介

处理器是解释并执行指令的功能部件。每个处理器都有一组独特的诸如 mov、add 或 sub 这样的操作命令集，这组操作集称为指令系统。在计算机诞生初期，设计者喜欢将计算机称为机器，因此该指令系统有时也称作机器指令系统。

处理器以惊人的速度执行指令指定的工作。一个称作时钟的计时器准确地发出定时电信号，该信号为处理器工作提供有规律的脉冲，通常叫时钟脉冲。测量计算机速度的术语引自电子工程领域，称作兆赫（MHz），兆赫意指每秒百万个时钟周期。在一个 8MHz 的处理器中，指定执行速度高达每秒 800 万次。这个概念相信读者并不陌生，衡量 CPU 的计算速度一般就是这个单位。

2.1.2　处理器的种类

处理器作为一种高科技产品，其技术含量非常高，因此处理器厂商也非常少。这些厂商主要有 Intel、AMD、ARM、中国威盛、Cyrix 和 IBM 等。目前，处理器在嵌入式领域的应用十分广泛，各大厂商都推出了自己的嵌入式处理器，主要有 ARM、MIPS、Power PC、Intel ATOM 等。了解这些嵌入式处理器的特性，是驱动开发人员必须要补的一课，下面对这些常用的处理器进行简要的介绍。

1．ARM处理器

ARM 处理器是英国 Acorn 有限公司设计的低功耗成本的第一款 RISC 微处理器，全称

为 Advanced RISC Machine。对应此处理器的详细介绍可以参考 2.2 节。

2．MIPS处理器

MIPS 实际上是芯片设计商 MIPS Technologies 公司的名字。MIPS Technologies 公司并不生产芯片，它只是把设计许可给其他公司，由其他公司制造、生产。例如，NEC 就是其主要的合作厂商。NEC 是所有用于 Pocket PC 的 MIPS 处理器的制造商。所有 Pocket PC 上使用的 MIPS 处理器都是 64 位处理器。MIPS Vr4121 处理器内建 8KB 的高速数据缓存和 16KB 的高速代码缓存；MIPS Vr4122 处理器内建 16KB 的高速数据缓存和 32KB 的高速代码缓存；而 MIPS Vr4181 处理器内建 4KB 的高速数据缓存和 4KB 的高速代码缓存。因缓存的大小不同，价格也有所不同，应该根据需要选择合适的处理器类型。Pocket PC 上使用的 MIPS 处理器的时钟频率范围为 70～150MHz，其应用十分广泛。

3．Power PC

Power PC 架构的特点是可伸缩性好，方便灵活。Power PC 处理器的品种很多，既有通用的处理器，又有嵌入式控制器和内核，应用范围从高端的工作站、服务器到桌面计算机系统，从消费类电子产品到大型通信设备等各个方面，非常广泛。

目前，Power PC 独立微处理器与嵌入式微处理器的主频从 25～700MHz 不等，它们的能量消耗、大小、整合程度和价格差异悬殊，主要产品模块有主频为 350～700MHz 的 Power PC 750CX 和 750CXe，以及主频为 400MHz 的 Power PC 440G 等。

嵌入式的 Power PC 405（主频最高为 266MHz）和 Power PC 440（主频最高为 550MHz）处理器内核可以用在各种集成的系统芯片（SoC）设备上，在电信、金融和其他行业有广泛的应用。

4．Intel ATOM

ATOM 是 Intel 公司出品的凌动处理器，它是 Intel 公司历史上体积最小和功耗最小的处理器。ATOM 基于新的微处理架构，专门为小型和嵌入式系统而设计，和其他嵌入式处理器相比，它的最大优势是采用 X86 体系结构，可以运行 Windows 操作系统（也可以运行 Android 操作系统），能提供更好的通用性，因此在平板电脑等消费类电子产品中得到了广泛的应用。

2.2　ARM 处理器

在所有处理器中，ARM 处理器是应用最广泛的一种处理器。ARM 处理器价格便宜，功能相对较多，是目前流行的嵌入式处理器之一。ARM 处理器分为很多种类，适用于不同的应用，下面对其进行详细介绍。

2.2.1　ARM 处理器简介

1．研发厂商ARM

ARM 是微处理器行业的一家知名企业，其设计了大量高性能、廉价、耗能低的 RISC

处理器。ARM 处理器具有性能高、成本低和功耗低的特点。其中，功耗低是其流行的主要原因之一。嵌入式设备一般使用电池，设备续航能力的高低主要取决于 CPU 的功耗。因为 ARM 处理器功耗低，所以适用于多个领域，如嵌入控制、消费/教育类多媒体、DSP 和移动式应用等。

ARM 公司将其技术授权给世界上许多著名的半导体、软件和 OEM 厂商。每个厂商得到的都是一套独一无二的 ARM 相关技术及服务。利用这种合作关系，ARM 很快成为许多 RISC 标准的缔造者。技术授权就是 ARM 公司为其他 CPU 生产商提供技术支持。

目前，总共有上百家半导体公司与 ARM 签订了硬件技术使用许可协议，其中包括 Intel、IBM、LG 半导体、NEC、SONY、菲利浦和国民半导体这样的大公司。至于软件系统的合伙人，则包括微软、IBM 和 MRI 等一系列知名公司。

2．ARM处理器的特点

ARM 处理器的优点很多，因此得到了广泛的使用，这些优点包括：

- 16/32 位双指令集，节省存储空间。
- 小体积、低功耗、低成本、高性能。
- 支持 DSP 指令集，支持复杂的算数运算，对多媒体处理非常有用。
- 使用 Jazelle 技术，对 Java 代码运行速度进行了优化。
- 全球众多的合作伙伴，ARM32 位体系结构被公认为业界领先的 32 位嵌入式 RISC 处理器结构，所有 ARM 处理器共享这一体系结构。这可确保开发者转向更高性能的 ARM 处理器时，由于所有产品均采用一个通用的软件体系，所以基本上可以在所有产品中运行，从而使开发者在软件开发上获得最大的回报。

2.2.2　ARM 处理器系列

ARM 处理器常用的有 6 个产品系列，分别是 ARM7、ARM9、ARM9E、ARM10、ARM11 和 SecurCore。一些产品来自于合作伙伴，如 Intel Xscale 微体系结构和 StrongARM 产品。ARM7、ARM9、ARM9E 和 ARM10 是 4 个通用的处理器系列，每个系列提供了一套特定的性能来满足设计者对功耗、性能、体积的需求。SecurCore 是第 5 个产品系列，是专门为安全设备而设计的。目前，我国市场应用较成熟的 ARM 处理器以 ARM7TDMI、ARM9 和 ARM11 核为主，主要的厂家有 SAMSUNG、ATMEL、OKI 等。下面对各系列处理器做简要的介绍。

1．ARM7系列

ARM7 系列包括 ARM7TDMI、ARM7TDMI-S、带有高速缓存处理器宏单元的 ARM720T 和扩充了 Jazelle 的 ARM7EJ-S。该系列处理器提供 Thumb 16 位压缩指令集和 EmbeddedICE JTAG 软件调试方式，适合应用于更大规模的 SoC 设计。其中，ARM720T 高速缓存处理宏单元还提供 8KB 缓存、读缓冲和具有内存管理功能的高性能处理器，支持 Linux、Symbian OS 和 Windows CE 等操作系统。

ARM7 系列处理器广泛应用于多媒体和嵌入式设备，包括 Internet 设备、网络和调制解调器设备及移动电话、PDA 等无线设备。无线信息设备领域的前景广阔，因此，ARM7

系列也瞄准了下一代智能化多媒体无线设备领域的应用。

2．ARM9系列

ARM9 系列有 ARM9TDMI、ARM920T 和带有高速缓存处理器宏单元的 ARM940T。所有的 ARM9 系列处理器都具有 Thumb 压缩指令集和基于 EmbeddedICE JTAG 的软件调试方式。ARM9 系列兼容 ARM7 系列，而且能够比 ARM7 进行更加灵活的设计。

ARM9 系列主要应用于引擎管理（如在自动挡汽车中的应用）、仪器仪表、安全系统、机顶盒、高端打印机、PDA、网络计算机及带有 MP3 音频和 MPEG4 视频多媒体格式的智能电话中。

3．ARM9E系列

ARM9E 系列为综合处理器，包括 ARM926EJ-S、带有高速缓存处理器宏单元的 ARM966E-S 和 ARM946E-S。该系列强化了数字信号的处理功能，可应用于需要 DSP 与微控制器结合使用的情况，将 Thumb 技术和 DSP 都扩展到 ARM 指令集中，并具有 EmbeddedICE-RT 逻辑（ARM 的基于 EmbeddedICE JTAG 软件调试的增强版本），更好地适应了实时系统开发的需要。同时，其内核在 ARM7 处理器内核的基础上使用了 Jazelle 增强技术，该技术支持一种新的 Java 操作状态，允许在硬件中执行 Java 字节码。

4．ARM10系列

ARM10 系列包括 ARM1020E 和 ARM1020E 微处理器核。其核心是使用向量浮点（VFP）单元 VFP10 提供的高性能浮点解决方案，极大提高了处理器的整型和浮点数据的运算能力，为用户界面的 2D 和 3D 图形引擎应用夯实了基础，如视频游戏机和高性能打印机等。

5．SecurCore系列

SecurCore 系列涵盖 SC100、SC110、SC200 和 SC210 处理器。该系列处理器主要针对新兴的安全市场，以一种全新的安全处理器设计为智能卡和其他安全 IC 设备开发提供独特的 32 位系统设计方案，并具有特定的反伪造方法，从而有助于防止对硬件和软件的盗版。

6．StrongARM系列和Xscale系列

StrongARM 处理器将 Intel 处理器技术和 ARM 体系结构融为一体，致力于为移动通信和消费电子类设备提供理想的解决方案。Intel Xscale 微体系结构则提供全性能、高性价比和低功耗的解决方案，支持 16 位 Thumb 指令和 DSP 指令。

2.2.3　ARM 处理器的应用

虽然 8 位微控制器仍然占据着低端嵌入式产品的大部分市场，但是随着应用的增加，ARM 处理器的应用也越来越广泛。这里将普遍的应用以一个表格列出，如表 2.1 所示。

表 2.1　ARM处理器的应用

产　　品	主 要 应 用
无线产品	手机、PDA，目前80%以上的手机是基于ARM的产品
汽车产品	车上娱乐系统、车上安全装置和导航系统等
消费类娱乐产品	数字视频、Internet终端、交互电视、机顶盒和网络计算机等；数字音频播放器和数字音乐板；游戏机
数字影像产品	信息家电、数字照相机和数字系统打印机
工业产品	机器人控制、工程机械和冶金控制等
网络产品	PCI 网络接口卡、ADSL调制解调器、路由器和无线LAN访问点等
安全产品	电子付费终端、银行系统付费终端、智能卡和32位SIM卡等
存储产品	PCI 到Ultra2 SCSI 64位RAID控制器和磁盘控制器

2.2.4　ARM 处理器的选型

随着国内外嵌入式应用领域的发展，ARM 芯片必然会获得广泛的重视和应用。但是，由于 ARM 芯片有多达十几种的芯核结构，100 多家芯片生产厂家以及千变万化的内部功能配置组合，给开发人员在选择方案时带来了一定的困难。下面从应用角度介绍 ARM 芯片选择的一般原则。

1．ARM处理器核

如果希望使用 WinCE 或 Linux 等操作系统以减少软件开发时间，就需要选择 ARM720T 以上带 MMU（Memory Management Unit）功能的 ARM 处理器芯片。ARM720T、StrongARM、ARM920T、ARM922T 和 ARM946T 都带有 MMU 功能。而 ARM7TDMI 没有 MMU，不支持 Windows CE 和大部分的 Linux，但目前有 uCLinux 等少数几种 Linux 不需要 MMU 的支持。

2．系统时钟控制器

系统时钟决定了 ARM 芯片的处理速度。ARM7 的处理速度为 0.9MIPS/MHz，常见的 ARM7 芯片系统主时钟为 20～133MHz，ARM9 的处理速度为 1.1MIPS/MHz，常见的 ARM9 的系统主时钟为 100～233MHz，ARM10 最高可以达到 700MHz。不同芯片对时钟的处理不同，有的芯片只有一个主时钟频率，这样的芯片可能不能同时兼顾 UART 和音频时钟的准确性，如 Cirrus Logic 的 EP7312 等；有的芯片内部时钟控制器可以分别为 CPU 核和 USB、UART、DSP、音频等功能部件提供不同频率的时钟，如 PHILIPS 公司的 SAA7550 等芯片。

3．内部存储器容量

当不需要大容量存储器时，可以选择内部集成存储器的 ARM 处理器，这样可以有效地节省成本。包含存储器的芯片如表 2.2 所示，不过这类芯片的存储空间较少，使用上也有很多局限性。

表 2.2　内置存储器的芯片

芯 片 型 号	生　产　商	Flash存储器容量	ROM存储器容量	SRAM存储器容量
AT91F40162	ATMEL	2MB	256KB	4KB
AT91FR4081	ATMEL	1MB		128KB
SAA7750	Philips	384KB		64KB
PUC3030A	Micronas	256KB		56KB
HMS30C7202	Hynix	192KB		
ML67Q4001	OKI	256KB		
LC67F500	Snayo	640KB		32KB

4．GPIO数量

在某些芯片供应商提供的说明书中，往往申明的是最大可能的 GPIO 数量，但是有许多引脚是和地址线、数据线、串口线等引脚复用的。这样在进行系统设计时需要计算实际可以使用的 GPIO 数量。

5．中断控制器

ARM 内核只提供快速中断（FIQ）和标准中断（IRQ）两个中断向量。但各个半导体厂家在设计芯片时加入了自己不同的中断控制器，以便支持诸如串行口、外部中断和时钟中断等硬件中断。

6．IIS接口

如果设计师想开发音频应用产品，则 IIS（Integrate Interface of Sound）总线接口是必备的，其支持音频输入和输出。

7．nWAIT信号

nWAIT 信号即外部总线速度控制信号。不是每个 ARM 芯片都提供这个信号引脚，利用这个信号与廉价的 GAL 芯片就可以实现与符合 PCMCIA 标准的 WLAN 卡和 Bluetooth 卡的接口，而不需要外加高成本的 PCMCIA 专用控制芯片。另外，当需要扩展外部的 DSP 协处理器时，此信号也是必需的。

8．RTC实时时钟

很多 ARM 芯片都提供了实时时钟的功能，但方式不同。例如，Cirrus Logic 公司的 EP7312 的 RTC（Real Time Clock）只是一个 32 位计数器，需要通过软件计算出年、月、日、时、分、秒；而 SAA7750 和 S3C2410 等芯片的 RTC 直接提供了年、月、日、时、分、秒格式。

9．LCD控制器

有些 ARM 芯片内置了 LCD 控制器，有的甚至内置了 64K 彩色 TFT LCD 控制器。在设计 PDA 和手持式显示记录设备时，选用内置 LCD 控制器的 ARM 芯片如 S1C2410 较为适宜。

10．PWM输出

有些 ARM 芯片有 2～8 路 PWM 输出，可以用于电机控制或语音输出等场合。

11．ADC和DAC模数转换

有些 ARM 芯片内置了 2～8 通道的 8～12 位通用的 ADC，可以用于电池检测、触摸屏和温度监测等。PHILIPS 的 SAA7750 更是内置了一个 16 位立体声音频 ADC 和 DAC，并且带耳机驱动。

12．扩展总线

大部分 ARM 芯片具有外部 SDRAM 和 SRAM 扩展接口，不同的 ARM 芯片可以扩展的芯片数量即片选线数量不同，外部数据总线有 8 位、16 位或 32 位。某些特殊应用的 ARM 芯片如德国 Micronas 的 PUC3030A 则没有外部扩展功能。

13．UART和IrDA

几乎所有的 ARM 芯片都具有 1～2 个 UART 接口，可以用于和 PC 通信或用 Angel 进行调试。一般的 ARM 芯片通信波特率为 115 200bps，少数专为蓝牙技术应用设计的 ARM 芯片的 UART 通信波特率可以达到 920Kbps，如 Linkup 公司的 L7205。

14．DSP协处理器

首先，需要了解一下什么是协处理器。协处理器是协助 CPU 完成相应功能的处理器，它除了与 CPU 通信之外，不会与其他部件进行通信。DSP 是协处理器中的一种，全称为数字信号处理器（Digital Signal Processor）。在进行图像处理和音频处理时，这种处理器用得很多。大多数 MP3 就使用 DSP 协处理器。常用的协处理器如表 2.3 所示。

表 2.3　常用的协处理器

芯 片 型 号	生　产　商	DSP处理器核心	DSP MIPS	应　　用
TMS320DSC2X	TI	16bits C5000	500	Digital Camera
Dragonball MX1	Motorola	24bits 56000		CD-MP3
SAA7750	Philips	24bits EPIC	73	CD-MP3
VWS22100	Philips	16bits OAK	52	GSM
STLC1502	ST	D950		VOIP
GMS30C3201	Hynix	16bits Piccolo		STB
AT75C220	ATMEL	16bits OAK	40	IA
AT75C310	ATMEL	16bits OAK	40x2	IA
AT75C320	ATMEL	16bits OAK	60X2	IA
L7205	Linkup	16bits Piccolo		Wireless
L7210	Linkup	16bits Piccolo		wireless
Quatro	OAK	16bits OAK		Digital Image

15．内置FPGA

有些 ARM 芯片内置有 FPGA，适合于通信等领域。常用的 FPGA 芯片如表 2.4 所示。

表 2.4　常用的FPGA芯片

芯 片 型 号	供　应　商	ARM芯片核心	FPGA门数	引　脚　数
EPXA1	Altera	ARM922T	100×2^{10}	484
EPXA4	Altera	ARM922T	400×2^{10}	672
EPXA10	Altera	ARM922T	1000×2^{10}	1020
TA7S20系列	Triscend	ARM7TDMI	多种	多种

16．时钟计数器和看门狗

一般，ARM 芯片都具有 2～4 个 16 位或 32 位时钟计数器，以及一个看门狗计数器。

17．电源管理功能

ARM 芯片的耗电量与工作频率成正比，ARM 芯片一般有 3 种模式，分别是低功耗模式、睡眠模式和关闭模式。

18．DMA控制器

有些 ARM 芯片内部集成有 DMA，可以和磁盘等外部设备高速交换数据，并且可以减少数据交换时对 CPU 资源的占用。另外，还可以选择的内部功能部件有 HDLC、SDLC、CD-ROM Decoder、Ethernet MAC、VGA controller、DC-DC，可以选择的内置接口有 IIC、SPDIF、CAN、SPI、PCI 和 PCMCIA。最后需要说明的是封装问题。ARM 芯片现在主要的封装有 QFP、TQFP、PQFP、LQFP、BGA 和 LBGA 等形式，BGA 封装具有芯片面积小的特点，可以减少 PCB 板的面积，但是需要专用的焊接设备，无法手工焊接。另外，一般 BGA 封装的 ARM 芯片无法用双面板完成 PCB 布线，需要多层 PCB 板布线。

2.2.5　ARM 处理器选型举例

在选择处理器的过程中，应该选择合适的处理器。所谓合适，就是在能够满足功能的前提下选择价格尽量便宜的处理器，这样开发出来的产品更具有市场竞争力。消费者也可以从合适的搭配中找到性价比高的产品。这里列出了一些常用的选择方案供读者参考，如表 2.5 所示。

表 2.5　处理器应用方案

应　　用	方　案　1	方　案　2	说　　明
高档PDA	S3C2440	Dragon ball MX1	
便携式CD/MP3播放器	SAA7750		USB和CD-ROM解码器
Flash MP3播放器	SAA7750	PUC3030A	内置USB和Flash
WLAN和BT应用产品	L7205，L7210	Dragon ball MX1	高速串口和PCMCIA接口
Voice Over IP	STLC1502		

续表

应　　用	方　案　1	方　案　2	说　　明
数字式照相机	TMS320DSC24	TMS320DSC21	内置高速图像处理DSP
便携式语音email机	AT75C320	AT75C310	内置双DSP，可以分别处理MODEM和语音
GSM 手机	VWS22100	AD20MSP430	专为GSM手机开发
ADSL Modem	S5N8946	MTK-20141	
电视机顶盒		GMS30C3201	VGA控制器
3G 移动电话机	MSM6000	OMAP1510	
10G 光纤通信	MinSpeed公司系列ARM芯片	多ARM核+多DSP核	

2.3　S3C2440 开发板

S3C2440 开发板上集成了一块 S3C2440 处理器。S3C2440 处理器是 ARM 处理器中的一款，广泛使用在无线通信、工业控制和消费电子领域。本节将对 S3C2440 开发板进行详细介绍，当然，如果你手上有任何一款开发板，本书的内容也是通用的。

2.3.1　S3C2440 开发板简介

目前大多数拥有 ARM 处理的开发板都是基于 S3C2440 处理器的。基于 S3C2440 的开发板由于资料全面、扩展功能好、性能稳定 3 大特点，深受广大嵌入式学习者和嵌入式开发工程师的喜爱。这种开发板由于性能较高，一般可以应用于车载手持、GIS 平台、Data Servers、VOIP、网络终端、工业控制、检测设备、仪器仪表、智能终端、医疗器械和安全监控等产品中。

2.3.2　S3C2440 开发板的特性

基于 S3C2440 开发板包含许多实用的特性，这些特性都是驱动开发人员练习驱动开发的好“材料”。下面对这些开发板的一般特性进行介绍。

1．CPU处理器

Samsung S3C2440A，主频 400MHz，最高 533MHz。

2．SDRAM内存

- 主板 64MB SDRAM。
- 32B 数据总线。
- SDRAM 时钟频率高达 100MHz。

3．Flash存储

- 主板 64MB Nand Flash，掉电非易失。
- 主板 2M Nor Flash，掉电非易失，已经安装 BIOS。

4．LCD显示

- 板上集成 4 线电阻式触摸屏接口，可以直接连接四线电阻触摸屏。
- 支持黑白、4 级灰度、16 级灰度、256 色、4096 色 STN 液晶屏，尺寸从 3.5 英寸到 12.1 英寸，屏幕分辨率可以达到 1024×768 像素。
- 支持黑白、4 级灰度、16 级灰度、256 色、64×2^{10} 色、真彩色 TFT 液晶屏，尺寸从 3.5 英寸到 12.1 英寸，屏幕分辨率可以达到 1024×768 像素。
- 标准配置为 NEC 256×2^{10} 色，分辨率为 240×320，尺寸为 3.5 英寸的 TFT 液晶显示屏，带触摸屏。
- 板上引出一个 12V 的电源接口，可以为大尺寸 TFT 液晶的 12V CCFL 背光模块（Inverting）供电。

5．接口和资源

- 1 个 100M 以太网 RJ-45 接口（采用 DM9000 网络芯片）。
- 3 个串行口。
- 1 个 USB Host。
- 1 个 USB Slave B 型接口。
- 1 个 SD 卡存储接口。
- 1 路立体声音频输出接口，一路麦克风接口。
- 1 个 2.0mm 间距 10 针的 JTAG 接口。
- 4USER Leds。
- 6USER buttons（带引出座）。
- 1 个 PWM 控制蜂鸣器。
- 1 个可调电阻，用于 AD 模数转换测试。
- 1 个 I^2C 总线 AT24C08 芯片，用于 I^2C 总线测试。
- 1 个 2.0 mm 间距 20pin 摄像头接口。
- 板载实时时钟电池。
- 电源接口（5V），带电源开关和指示灯。

6．系统时钟源

- 12MHz 无源晶振。

7．实时时钟

- 内部实时时钟（带后备锂电池）。

8．扩展接口

- 1 个 34 pin 2.0mmGPIO 接口。
- 1 个 40 pin 2.0mm 系统总线接口。

9．操作系统支持

- Linux 2.6.x 及以上的版本。
- Windows CE.NET。
- Android。

2.3.3　其他开发板

也许读者手中有不同种类的开发板，这并不是说本书的内容就不适合你。读者只需要找到开发板对应的芯片手册，就能够使用本书介绍的方法来学习驱动程序的开发。

2.4　小　　结

本章简单介绍了驱动开发人员必备的处理器知识，详细介绍了 S3C2440 处理器构建的开发板。对驱动开发人员来说，重要的是处理器选型问题。本章不仅给出了详细的处理器选型准则，而且对常见应用的选型进行了举例，相信读者通过本章的学习会有所收获。

2.5　习　　题

一、填空题

1．处理器是解释并执行_____的功能部件。
2．ARM 内核只提供_____中断和标准中断两个中断向量。
3．系统时钟决定了 ARM 芯片的_____速度。

二、选择题

1．测量计算机速度的术语是（　　）。
A．赫兹　　B．m/s　　C．兆赫　　D．兆
2．ARM 芯片选择的一般原则不包含（　　）。
A．ARM 处理器核　　B．系统时钟控制器
C．内部存储器容量　　D．LCD 显示
3．ARM 处理器的特点不包含（　　）。
A．16/32 位双指令集　　B．支持 DSP 指令集
C．Jazelle 技术　　D．大体积、低功耗

三、判断题

1．ARM7TDMI 有 MMU。（　　）

2．ATOM 是 Intel 公司出品的凌动处理器。（　　）

3．S3C2440 开发板有 2 个 2.0mm 间距 10 针的 JTAG 接口。（　　）