基于嵌入式 Linux 下的 USB3.0 驱动程序开发方法研究

杨　军[1,2]

(1.青岛科技大学 信息科学学院,山东 青岛 266601;2.天水师范学院 物信学院,甘肃 天水 741000)

摘　要:USB3.0 是新一代通用串行总线,该总线下的设备目前还没有大规模生产,主要原因是它的通信标准及协议刚由 USB 3.0 开发小组(超过 200 家公司)制定出来。USB3.0 必将很快取代 USB2.0 成为今后市场的主流 USB 设备接口。本文针对嵌入式 Linux 操作系统内核提供编写设备驱动程序的基本框架结构给出了基于嵌入式 Linux 下的 USB3.0 驱动程序开发方法。

关键词:嵌入式 Linux;USB3.0;驱动程序

所谓嵌入式系统是指以应用为中心,以计算机技术为基础,软硬件都可裁剪,适用于应用系统对功能、可靠性、成本、体积和功耗有严格要求的专用计算机系统[1]。嵌入式系统由嵌入式微处理器、外围硬件设备、嵌入式操作系统以及用户应用程序 4 部分组成。搭建一个嵌入式系统是开发一个嵌入式产品的基础。由于嵌入式系统的灵活性和应用的广泛性使它的硬件形式也是多种多样层出不穷,这就给工程师们提出更高的要求。USB3.0 通信标准和协议的推出必将带动基于 USB3.0 接口设备的大量生产。对于每一种 USB3.0 产品在嵌入式应用中都必须开发其相应的设备驱动程序才可使该产品正常工作。因此,嵌入式系统工程师们对于新标准下的 USB3.0 产品驱动程序的设计与开发应该做好充分准备。

1　嵌入式 Linux 驱动开发

1.1 嵌入式 Linux

嵌入式 Linux 系统有两层含义,狭义的嵌入式 Linux 系统指的是嵌入式 Linux 操作系统,是指对 Linux 经过裁剪后,固化在容量只有几 K 到几 M 字节的存储器芯片或 MCU 中,应用于特定嵌入式场合的专用 Linux 操作系统。广义的嵌入式 Linux 系统指的是基于嵌入式 Linux 操作系统构建的嵌入式系统。

嵌入式 Linux 系统包括嵌入式 Linux 内核、文件系统和用户应用程序三部分。嵌入式 Linux 内核包含 Linux 的系统调用接口、设备驱动以及 Linux 内核机制;嵌入式文件系统是嵌入式操作系统的一部分,它的任务是对逻辑文件进行管理,其工作包括提供对逻辑文件的操作接口,方便用户操作文件和目录;用户程序通常指运行于用户空间能接受内核管理和调度的各种可执行程序。

1.2 Linux 驱动

驱动程序是 Linux 内核的重要组成部分,可以看作是应用程序和物理设备之间的一个软件层,由设备驱动程序来完成操作系统与硬件设备之间的交互。对于嵌入式开发而言由于没有通用的驱动程序,因而驱动程序开发便成为嵌入式系统设计过程的一个重要环节。

驱动程序包括配置初始化子程序和 I/O 请求子程序。配置初始化子程序在初始化时被调用一次,I/O 请求子程序的调用通过系统调用或硬件中断信号来触发。自动配置和初始化子程序一般在设备接入系统时或者加载设备驱动时调用,其主要负责检测所需驱动的硬件设备是否存在或是否能正常工作。如果该设备正常,则对这个设备及其相关的驱动程序需要的软件状态进行初始化。驱动程序所提供的与设备的打开、释放、读写和控制操作相对应的入口点函数都属于服务于 I/O 请求的子程序,并且通过 file_operations 结构向系统进行说明。中断服务子程序在嵌入式 Linux 系统中并不是直接从中断向量表中调用设备驱动程序的中断服务子程序,而是由 Linux 系统接收硬件中断,再由系统调用中断服务子程序。

与应用程序不同,设备驱动程序属于内核的一部分,所以驱动程序的开发就是 Linux 内核的开发。完成一个设备驱动程序后,用户可以动态地将该设备驱动程序加载到内核中或从内核中卸载。加载和卸载的入口函数为 init_module() 和 cleanup_module()。除此之外对于设备的每一种操作设备驱动程序中都包含其相应的入口函数。字符型和块设备驱动程序中包括打开设备函数 open()、关闭设备函数 close()、读数据函数 read()、写数据函数 write() 和 I/O 控制函数 ioctl() 等。

2　USB3.0 驱动程序设计

2.1 USB3.0 系统架构

目前国内对 USB 系统的研究主要在实际应用方面，一些以往的计算机通用外设和主流的数码电子产品中已经广泛采用 USB 技术。并成为当今嵌入式开发和应用中的一个热点。相比于 USB 2.0 总线 USB 3.0 是一个超高速总线并且具有和 2.0 总线相似的系统架构。USB 3.0 系统架构分为三个层次结构。即：USB 3.0 host 层；USB 3.0 hub 层；USB 3.0 的并行设备层。USB3.0 架构向下兼容 2.0 总线体系结构，因此 USB3.0 是一个双总线系统架构（见图 1）。

USB 3.0 host 是 USB 3.0 和 USB 2.0 设备连接和通信的主控制器。USB 3.0 继承了 USB 2.0 的核心结构，虽然做了一些外部扩展以适应双总线架构但基本的星形拓扑结构是和 USB2.0 一样的。对于 USB 3.0 总线任务是通过一个复合电缆及相关连接器构成的。USB 3.0 主控制器完成包括超高速总线和非超高速总线接口的控制。

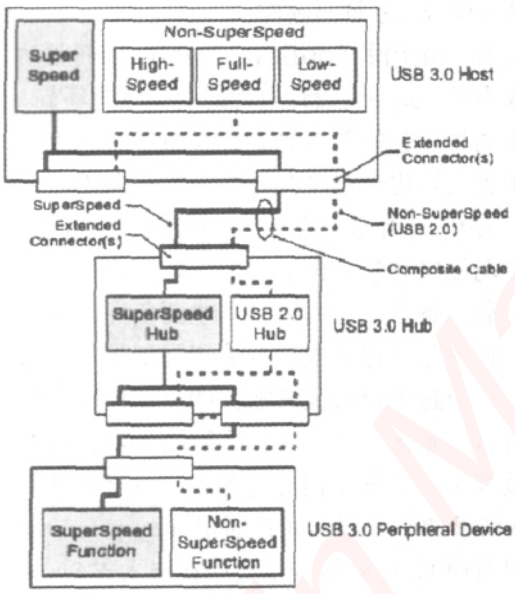

图 1 USB 3.0 双总线架构

USB3.0 集线器是用于连接更多 USB 设备到主控制器而提供更多接口的一个逻辑设备。USB 3.0 连接模式允许探测和配置 USB 设备的最高传输速度。在主机和设备间的最高信号传输速度和当前主控制器的功能及配置的探测以及 USB 设备的配置情况是通过所连接的集线器来实现探测和配置任务的。USB 3.0 主机同时包括超和非超高速总线接口，采用并行总线方式因此可使两种接口通过 hub 同时在主控制器的控制下有效工作。

USB 3.0 在应用层上至少能达到 300Mbyte/s 的数据吞吐量。新规范与前代版本兼容，然而新接口需要新的线缆和连接器，而且传输距离被限制在 3 米，而目前的 USB 产品可以支持 5 米长的线缆。3.0 标准，也被称作是超高速 USB（SuperSpeed USB），在一些特性上是独一无二的。它使用 5 个端口连线、两个用于发送，两个用于接收，一个是地线、来实现全双工从而达到 5 Gb/s 的物理层速率，目前的 USB 产品采用两线，半双工的架构。外观上 Type- A 的接头没有改变，但内部有 5 个连线来支持全双工，新的连接器兼容旧的插口。另外，3.0 版本在链路上采用了中断驱动，而不是目前的轮检方法，这样进一步降低功耗。

2.2 USB3.0 串口驱动设计实例

USB 驱动分为 USB 主机驱动和 USB 设备驱动，如果系统的 USB 主机控制器符合 OHCI 等标准，则主机驱动的绝大部分工作都可以沿用通用的代码。

对于一个 USB 设备而言，它至少具备两重身份：首先它是“USB”的，其次它是“自己”的。USB 设备是“USB”的，指它挂接在 USB 总线上，其必须完成 usb_driver 的初始化和注册；USB 设备是“自己”的，意味着本身可能是一个字符设备、tty 设备、网络设备等，因此，USB 设备驱动中也必须实现符合相应框架的代码。

在 Linux 内核中，串口属于 tty 设备，对于一个 USB 串口设备而言，其驱动主要由两部分组成：usb_driver 的成员函数和 tty 设备的 tty_operations 结构体成员函数。

在 USB 串口设备驱动的模块加载函数中，将注册对应于 USB 串口的 usb_driver，利用 static int __init usb_serial_init（void）函数初始化和注册 tty 驱动。利用 USB 串口设备驱动的模块卸载函数 static void __exit usb_serial_exit（void）将注销对应于 USB 串口的 usb_driver，并注销 tty 驱动。

在 usb_driver 的探测成员函数 usb_serial_probe（）中，将初始化 USB 端点等信息，并通过 usb_set_intfdata（）设置接口私有数据，它也将初始化 urb（USB 请求块）。相反，在断开成员函数 usb_serial_ disconnect（）中将设置接口私有数据为 NULL，并释放引用计数。

在 tty_operations 的各 write（）、read（）等成员函数中，将调用 usb_serial_driver 结构体中的相应函数，usb_serial_driver 结构体中封装了串口的各函数（读写、读写中断端点完成函数、读写批量端点完成函数等）。

目录 drivers/usb/serial/ 下 generic.c 文件中提供了 USB 串口驱动的通用打开 / 关闭、读 / 写函数等，如 usb_serial_generic_write（）、usb_serial_generic_write_bulk _callback（）、usb_serial_generic_read_bulk_ callback（）等。

3 结束语

本论述归纳了 Linux 驱动程序设计的一般方法，并根据工程实践给出了一个基于嵌入式 （下转 39 页）

集中于中低端用户群。他们使用 CMMB 手机电视的目的就是通过手机来观看电视节目，从而获得简单的文化生活。这个群体对节目清晰度以及节目的片源都没有太高的要求，需求主要停留在“有和无”的层次。而 TD 标准的手机电视已经不仅能够满足电视信号覆盖，还能够满足中高端用户群体对手机电视内容点播服务的需求。用 CMMB 观看电视节目，价格便宜，不走流量。而如果需要互动，或者收看在传统电视节目里没有的短视频电视节目，则可以通过 TD 制式的流媒体方式来观看。

⑶ 虽然目前 TD 与 CMMB 的合作从技术上来说，并没有做到融合，严格意义上只是两种功能的相互组合。但是 TD+CMMB 迈出良好的开端之后，电信运营商与广电运营商的下一步合作具有非常广阔的空间。在目前双方进行合作的手机电视业务方面，由于双方共用一个屏幕，那么可以将单向广播与双向交互结合起来，也就是将广播电视业务与移动通信业务相结合，将需要大量下行的数字电视节目传输由广电的 CMMB 网络来承载，而在数据上行和用户信息收集方面，则由具有上行回路的 TD 网络来运行[2]。

⑷ 在收费渠道方面，中国移动、中国联通、中国电信等经过多年的建设，不仅拥有遍布大街小巷的营业网点，而且已经建成全国联网的 BOSS（电信业务运营支持系统）系统，能够做到实时计费、融合计费。收费渠道方面的优势，完全可以帮助广电解决 CMMB 收费方面存在的难题。

4 用户关心的问题

4.1 如何收看电视节目

手机用户无论采用哪家公司的 3G 服务都需要更换成带有 CMMB 芯片 3G 手机才能收看电视节目。在使用 3G 手机的前提下，移动和电信用户不需更换手机号就能升级使用 3G。其中，移动推出“三不政策”，即所有中国移动用户不用换手机号、不用换 SIM 卡、也不用到营业厅办理登记手续，只需将原手机 SIM 卡插入 3G 手机即可。中国电信 133/153 在内的全部用户均可以使用目前 189“天翼”业务。中国联通 130、131、132、156 用户无需换号可直接升级 3G。

没有手机的用户想收看电视节目必需购买电视终端，其价格在四佰元到几千元不等。

4.2 费用问题

中国移动将在统一定价的基础上，针对使用 3G 的用户给予折扣。中国电信在 189 号段推出了商旅套餐和畅聊套餐。中国联通 3G 的资费水平将不会超过 2G 时代。中国多媒体广播电视推出包月 12 元收费标准。

4.3 网络覆盖问题

中国移动表示 2011 年 TD 网络将覆盖全国所有地市。电信今年 7 月将在全国开展 3G 服务。中国联通计划于上半年在中国 55 个城市推出 3G 服务，并且计划在今年年底前将该服务的覆盖范围扩大至 282 个城市。广电部门已经在全国地级以上市区建设了地面增补网络[1]。

5 “手机电视”的应用前景

采取移动多媒体广播方式，可以高效率、低成本地向流动人群和交通工具提供 20 多套电视、30 多套广播以及各类信息服务，满足这些用户群的需求，市场潜力巨大，发展前景良好。同时，作为广电行业推荐标准的 CMMB 完全具有成为国标的核心优势广电借助卫星通信，能极好地解决手机电视信号流畅的问题。手机电视将成为广电部门更好开展业务，传播优秀广播电视节目的又一信息化平台。

参考文献：

[1] 张忠明.什么是 3G[EB/OL].http://www.sogu.com/web?query=158614272&_asf=www.sgou.com&w=01029901&num=10&p=40040101&dp=1.

[2] 马龙.TD 与 CMMB 双方技术优势互补[EB/OL].http://www.lokmw.cn/html/jgdiahang/wz-59.html.

[3] 王其武.中国移动广播多媒体 CMMB 应用[EB/OL].http://www.jpw8.com/cteye/2000/312/1031289581FICE8J04H20AA27E52.html.

（上接 34 页）Linux 平台下 USB3.0 驱动程序的设计方法。实践表明：开发一个稳定、高效的设备驱动程序需要工程师把握一个原则就是不要给用户强加任何策略，因为不同的用户有不同的需求，驱动程序应该处理如何使硬件工作的问题，而将怎样使用硬件的问题留给上层应用。总而言之，驱动程序应当为上层应用提供一个使用设备的工具箱。

参考文献：

[1] 杨恒.ARM 嵌入系统设计及实践[M].西安:西安电子科技大学出版社,2005.

[2] 李驹光,郑耿,江泽明.嵌入式 Linux 系统开发详解[M].北京:清华大学出版社, 2006.

[3] Universal Serial Bus Specification.Http://www.usb.org,2000.4.

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用
33. 高可用的磁盘阵列 Cache 的设计和实现
34. RAID 阵列中高速 Cache 管理的优化

35. 一种新的基于 RAID 的 CACHE 技术研究与实现
36. 基于 PCIE-104 总线的高速数据接口设计
37. 基于 VPX 标准的 RapidIO 交换和 Flash 存储模块设计
38. 北斗卫星系统在海洋工程中的应用
39. 北斗卫星系统在远洋船舶上应用的研究
40. 基于 CPCI 总线的红外实时信号处理系统
41. 硬件实现 RAID 与软件实现 RAID 的比较
42. 基于 PCI Express 总线系统的热插拔设计
43. 基于 RAID5 的磁盘阵列 Cache 的研究与实现
44. 基于 PCI 总线的 MPEG2 码流播放卡驱动程序开发
45. 基于磁盘异或引擎的 RAID5 小写性能优化
46. 基于 IEEE1588 的时钟同步技术研究
47. 基于 Davinci 平台的 SD 卡读写优化
48. 基于 PCI 总线的图像处理及传输系统的设计
49. 串口和以太网通信技术在油液在线监测系统中的应用
50. USB30 数据传输协议分析及实现
51. IEEE 1588 协议在工业以太网中的实现
52. 基于 USB30 的设备自定义请求实现方法
53. IEEE1588 协议在网络测控系统中的应用
54. USB30 物理层中弹性缓冲的设计与实现

VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用
14. 基于 MPC8270 的 VxWorks BSP 的移植
15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现

17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发
23. Vxworks 下的冗余 CAN 通讯模块设计
24. WindML 工业平台下开发 S1d13506 驱动及显示功能的实现
25. WindML 中 Mesa 的应用
26. VxWorks 下图形用户界面开发中双缓冲技术应用
27. VxWorks 上的一种 GUI 系统的设计与实现
28. VxWorks 环境下 socket 的实现
29. VxWorks 的 WindML 图形界面程序的框架分析
30. VxWorks 实时操作系统及其在 PC104 下以太网编程的应用
31. 实时操作系统任务调度策略的研究与设计
32. 军事指挥系统中 VxWorks 下汉字显示技术
33. 基于 VxWorks 实时控制系统中文交互界面开发平台

Linux:

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通
12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核
15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究

20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings
25. Android 应用逆向分析方法研究
26. Android 操作系统的课程教学
27. Android 智能手机操作系统的研究
28. Android 英文朗读功能的实现
29. 基于 Yocto 订制嵌入式 Linux 发行版
30. 基于嵌入式 Linux 的网络设备驱动设计与实现
31. 如何高效学习嵌入式
32. 基于 Android 平台的 GPS 定位系统的设计与实现
33. LINUX ARM 下的 USB 驱动开发
34. Linux 下基于 I2C 协议的 RTC 驱动开发
35. 嵌入式下 Linux 系统设备驱动程序的开发
36. 基于嵌入式 Linux 的 SD 卡驱动程序的设计与实现
37. Linux 系统中进程调度策略
38. 嵌入式 Linux 实时性方法
39. 基于实时 Linux 计算机联锁系统实时性分析与改进

Windows CE:

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]
16. 基于 WinCE 的 BootLoader 研究

17. Windows CE 环境下无线网卡的自动安装
18. 基于 Windows CE 的可视电话的研究与实现
19. 基于 WinCE 的嵌入式图像采集系统设计
20. 基于 ARM 与 WinCE 的掌纹鉴别系统
21. DCOM 协议在网络冗余环境下的应用

PowerPC：

1. Freescale MPC8536 开发板原理图
2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计
7. 用 PowerPC860 实现 FPGA 配置
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植
12. 基于 PowerPC 双核处理器嵌入式系统 UBoot 移植
13. 基于 PowerPC 的雷达通用处理机设计
14. PowerPC 平台引导加载程序的移植
15. 基于 PowerPC 嵌入式内核的多串口通信扩展设计
16. 基于 PowerPC 的多网口系统抗干扰设计
17. 基于 MPC860T 与 VxWorks 的图形界面设计
18. 基于 MPC8260 处理器的 PPMC 系统
19. 基于 PowerPC 的控制器研究与设计

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理

5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的μC-OSⅡ及 LwIP 协议栈的移植与实现
16. 基于 ARM 的嵌入式 Linux 无线网卡设备驱动设计
17. ARM S3C2440 Linux ADC 驱动
18. ARM S3C2440 Linux 触摸屏驱动
19. Linux 和 Cortex-A8 的视频处理及数字微波传输系统设计
20. Nand Flash 启动模式下的 Uboot 移植
21. 基于 ARM 处理器的 UART 设计
22. ARM CortexM3 处理器故障的分析与处理
23. ARM 微处理器启动和调试浅析
24. 基于 ARM 系统下映像文件的执行与中断运行机制的实现
25. 中断调用方式的 ARM 二次开发接口设计
26. ARM11 嵌入式系统 Linux 下 LCD 的驱动设计
27. Uboot 在 S3C2440 上的移植
28. 基于 ARM11 的嵌入式无线视频终端的设计

Hardware：

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究

13. 基于 UEFI 固件的恶意代码防范技术研究
14. MIPS 架构计算机平台的支持固件研究
15. 基于 UEFI 固件的攻击验证技术研究
16. 基于 UEFI 的 Application 和 Driver 的分析与开发
17. 基于 UEFI 的可信 BIOS 研究与实现
18. 基于 UEFI 的国产计算机平台 BIOS 研究
19. 基于 UEFI 的安全模块设计分析
20. 基于 FPGA NiosⅡ的等精度频率计设计
21. 基于 FPGA 的 SOPC 设计
22. 基于 SOPC 基本信号产生器的设计与实现
23. 基于龙芯平台的 PMON 研究与开发
24. 基于 X86 平台的嵌入式 BIOS 可配置设计

Programming:

1. 计算机软件基础数据结构 - 算法
2. 高级数据结构对算法的优化
3. 零基础学算法
4. Linux 环境下基于 TCP 的 Socket 编程浅析
5. Linux 环境下基于 UDP 的 socket 编程浅析
6. 基于 Socket 的网络编程技术及其实现