Vxworks下的冗余CAN通讯模块设计

Designing of Redundant CAN Model Based on the RTOS vxworks

北京邮电大学 杨 君 孙汉旭 贾庆轩 史国振
YANG JUN SUN HANXU JIA QINGXUAN SHI GUOZHEN

摘要 本文介绍了航天应用领域中CAN总线的多种冗余方法，重点论述了在实时操作系统vxworks下基于ARM处理器的二模(DRM) CAN冗余的硬件设计和软件驱动程序设计过程，并在空间六自由度机器人通信平台中得到实现。

关键词：CAN总线；vxworks; ARM处理器
中图分类号：TP302.1 **文献标识码**：B

Abstract: Many redundant methods of CAN bus in spaceflight are introduced, and it emphasized the designing of hardware architecture of the double redundant model (DRM) CAN based on ARM processor, in addition, the realization of the CAN software driver based on RTOS vxworks is given, it completed in the 6 DOF space robot telecommunication platform.

Key words: CAN bus, vxworks, ARM process

CAN(controller area network)属于现场总线，是一种有效支持分布式控制串行通信局域网络，由于其高性能、高可靠性、实时性好及其独特的设计，已应用于控制系统中的各检测和执行机构之间的数据通信，在工业控制领域、汽车、航天航空领域中得到了广泛的应用。

本研究是基于航天863机械手系统项目，CAN模块用于整个机械手系统和外界的数据通讯，由于中央控制器将长期工作于外层空间，空间辐射环境如高能粒子产生的单粒子翻转和单粒子锁定会使星载计算机出现大量瞬态故障。对于CAN通讯模块而言，接插件的损坏或者驱动器损坏都可以影响通讯的可靠性，为了保证通讯的可靠性和稳定性，一种有效方法就是对总线进行硬件冗余设计。对于航天系统，为了保证其稳定运行，不仅要在硬件上保证安全性，在软件上也要保证其实时性和可靠性。为了有效管理整个系统资源利用、提高系统的可移植性，我们引入了实时操作系统Vxworks，在航天上已得到验证。

1 CAN冗余方法分析与设计

1.1 CAN冗余方法分析

在航空领域，为了保证通讯的可靠性和稳定性，在通讯领域都不同程度的使用了冗余技术。我国发射的返回式卫星首先采用三模冗余(TMR)技术，二模冗余(DMR)等多种星载计算机体系结构和容错方法。

三模冗余是由3个功能相同的模块和1个表决器组成的系统，表决器按三取二方式工作，只要3个模块中任何两个模块的输出一致时，两个模块的输入"与"函数作为表决器的输出。

二模冗余就是两个功能相同的模块同时工作，二模冗余又可以分为二模热冗余和二模冷冗余。二模热冗余就是在两个相同模块之间加一个判断电路，自动选择其中一个模块工作屏蔽另一个模块。二模冷冗余可以将两个模块连在电子模拟开关上，系统初始化时，默认选择其中一路为当班模块，有效工作模块。当这路模块出现运行故障时，通过软件判定，系统切换到另一路模块工作并复位出错模块。

1.2 CAN冗余方法设计

航天控制系统在功能实现的情况下，要求硬件设计尽量简单，虽然三模冗余可以提高系统的可靠性，但是三个模块之间要连接一个输出表决单元，二模热冗余两个模块要连接一个判断电路，而且只要CAN驱动器接收数据则表决单元和判断电路都要频繁处于工作状态，所以决定采用二模冷冗余设计方法。

通信模块处理器采用以ARM7TDMI为内核AT91FR40162芯片，CAN控制器选用philips公司的SJA1000，它可以自动完成链路层以下的收发工作，驱动器选用PCA82C250，它可以增加总线挂接节点个数。整个系统采用双机冷热备份，单机CAN模块采用驱动器二模冷冗余，系统冗余设计框图如下：

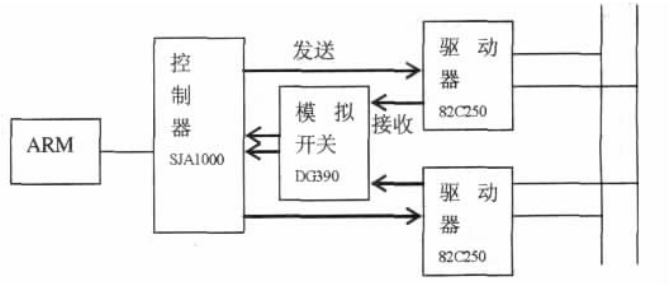

图1 单机CAN二模驱动器冷冗余简图

中央控制器单机采用CAN驱动器冷冗余，即一个CAN总线控制器SJA1000控制两个CAN总线驱动器PCA82c250，两个CAN驱动器分别连在两路物理总线上。单机发送时，控制器将数据同时发送到两路总线上，单机接收时，控制器通过电子模拟开关DG390芯片选择一路驱动器数据进行接收。一路如果长时间收不到数据或者出现错误则由软件判定切换到另一路驱动器进行接收。由于中央控制器是双机冷热备份，所以整个系统间接实现了can控制器冗余。

2 CAN驱动程序的编写

2.1 I/O系统与标准设备驱动

vxworks下标准设备驱动程序的管理可划分成三种模块：字符设备、块设备和网络设备驱动程序模块。本文所介绍的CAN驱动属于字符设备驱动程序模块。

为了屏蔽底层硬件，增加应用程序可移植性，vxworks操作系统为应用程序提供了一个统一的接口——I/O系统。vxworks的I/O系统为字符型设备提供了一个基本的I/O操作函数：open()、creat()、read()、write()、ioctl()、close()和remove()，应用程序调用7个基本函数，由基本函数调用真正的底层驱动程序，实现对底层硬件的操作。I/O系统如图2所示：

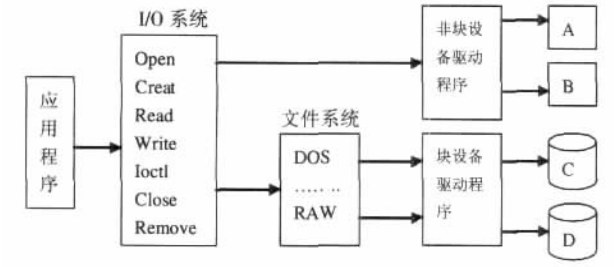

图2 I/O系统与设备驱动程序

2.设备驱动访问过程

基本I/O操作函数与真正的驱动程序是通过设备列表、驱动程序描述表、文件描述符表连接起来的。在应用程序调用I/O函数进行设备操作之前，首先需要编写与之对应的一个底层设备驱动程序：xxOpen()、xxCreat()、xxRead()、xxWrite()、xxIoctl()、xxClose()和xxRemove()，然后通过xxDrv()调用iosDrvInstall()函数将上述6个设备驱动程序填入到驱动程序表中并返回驱动程序索引号Drvnum，接着通过xxDevCreate()调用iosDevAdd()函数将设备加入到设备列表中，其中包括设备名（如"/Can/0"）、设备驱动程序索引号和设备号。

操作设备之前，必须先用open()函数打开设备，如fd=open("/Can/0", O_RDWR,0)。是open()返回的文件描述符，表1所示通过fd可以找到设备号和驱动程序索引号，将fd作为句柄传递给I/O读写和控制函数完成对设备的操作。打开设备后，每次对设备操作都是通过fd查找文件描述符表找到相应的设备头结构和驱动程序入口地址。

表1 文件描述符表

文件描述符(fd)	驱动程序索引号(Drvnum)	设备号
3	2	1
4	5	3
5	5	3

2.3 CAN驱动程序具体实现

(1)调用CanDrv()将底层驱动函数挂接在I/O系统上。在CanDrv()中调用CanHrdInit()对CAN底层硬件进行初始化，CanDevCreate()将设备添加到设备列表。函数示意代码如下：

```
STATUS CanDrv (void)
{
CanHrdInit (); //can硬件初始化
CanDrvNum = iosDrvInstall (canOpen, (FUNCPTR) NULL, canOpen,(FUNCPTR) NULL, CanSend, CanRecv, NULL);将驱动程序入口地址填入I/O系统
}
STATUS CanDevCreate(char * name,设备名
int channel 设备号 )
{
return (iosDevAdd (&CanDv [channel].devHdr, name, CanDrvNum);将设备添加到设备列表
}
```

(2)编写open程序用于设备打开，提供读写操作对设备进行操作设备头指针。

```
LOCAL int canOpen (DEV_HDR *pCanDv,设备结构指针
char *name, 设备名
int mode 打开模式
)
{
return (int pCanDv);返回设备结构头指针
}
```

(3)CAN底层接收用中断形式，在sysHwInit2中实现CAN中断挂接到vxworks上。

```
void sysHwInit2 (void)
{
打开CAN的I/O控制端口；
(void)intConnect (INUM_TO_IVEC (CAN0_INT_VEC), Can0Recv, 0);挂接CAN中断
}
```

(4)编写底层发送和接收驱动程序，Can0Recv()用于中断接收中断服务程序读取接收缓冲的数据，成功接收一帧数据后，释放接收缓冲区并返回接收数据字节数。CanSend()用于查询发送，发送时一定要等到发送缓冲区可用再填写发送缓冲区，然后置位发送命令。

```
LOCAL int CanSend (int canDevId, 设备结构指针
)
{
CAN_DEV * pcanDev = (CAN_DEV *)canDevId;
发送缓冲区可用 )
{
向发送缓冲区写数据；
置位发送；
}
}

LOCAL int Can0Recv (int canDevId, 设备结构指针
)
{
CAN_DEV * pcanDev = (CAN_DEV *)canDevId;
从接收缓冲区读取数据放入内存；
释放接收缓冲区；
}
```

3.结束语

本文作者创新点在于提出并实现了一种高效可靠的硬件CAN总线二模冗余方法，成功移植vxworks下CAN底层驱动软件，并在8自由空间机器人项目中得到验证，该冗余方法使系统的可靠性得到很大提高，可供后续航天工作者参考与借鉴。

参考文献

[1]饶运涛，现场总线CAN原理与应用技术[M].北京：北京航天航空大学出版社，2003.

块完成的 这也是 SoC设计的另一个显著特点。

行为描述就是把系统分成多个功能块 由这几个模块完成系统所要求的功能 结构描述主要是把已有的 IP核或者自行设计的 IP核描述成模块 使之与功能模块相匹配 通过不断的系统仿真从而确定与系统行为描述相匹配的系统结构描述。这主要涉及到系统描述 行为描述和结构描述 的规范性。为了便于系统性能仿真 系统描述必须是规范的 这样才能解决描述不匹配的问题 才能使性能仿真人员快速进行仿真 从而节省系统描述到性能仿真的过渡时间。

软硬件划分是软硬件协同设计的关键技术。软硬件划分是指在设计系统时 确定各个模块是采取软件还是硬件的实现方式。软件实现的特点 灵活、成本低 而硬件实现的特点 性能高 但同时成本也增高。如何兼顾系统的速度和成本 达到成本和性能的最佳结合 是软硬件划分所要解决的问题。应遵循的基本原则是高速、低功耗由硬件实现 多品种、小批量由软件对应 处理器和专用硬件并用以提高处理速度和降低功耗。划分的方法应该从两方面着手 面向软件 从软件到硬件满足时序要求 面向硬件 从硬件到软件降低成本。

软硬件并行综合与 IP设计综合相比 增加了许多约束和限制 其中最大的问题就是 SoC与众多 IP核之间接口的综合。软硬件并行仿真就是用软件控制硬件的仿真 在系统级芯片上 硬件和软件之间密切相关 但在系统做出之前 软硬件之间的相互作用通常很难精确测出 一些设计错误也不会明显表示出来。为了解决这个问题 必须采用软硬件协同验证技术。软硬件的协同仿真始终是设计中的关键 需利用相应的 EDA工具采取先进的协同仿真技术 才能达到协同仿真的目的。

软硬件协同设计与传统的单流程设计有着本质的区别 其软件和硬件的设计不再是两个独立的设计单元 在设计之初便相互交织在一起 相互提供设计平台 相互作用 真正实现二者的并行性。不管是整体设计还是局部设计 并行的思想始终贯穿于设计之中 这也是软硬件协同设计技术的核心。协同设计流程中目前解决较好的是算法级和系统级的行为描述及模拟 难点在于软硬件划分。在硬件流程中 高级层次的行为描述与 RTL级之间的接口 尚待完善。

与此同时 EDA环境也将不断进步 以适应 SOC开发的需要 一些新兴的 EDA公司还提供出了重点发展 C语言、标准单元库和网上设计服务等新的发展策略 并且开发基于 C语言的硬件描述工具 采用 C为设计语言 不但运行速度比 HDL快 2个数量级 而且可为 IP供应商提供知识产权保护 ,这些又为 SOC技术的发展注入了新的活力。

3 结论与展望

本文作者创新点 目前的 SOC的主要研究方法是将现有的器件和电路模块进行集成化 将来会根据系统集成的特点 将传感器、执行器嵌入系统之中 这样 SOC将不断走向高性能、高功能 在模块融合的基础上产生新的结构 以便产生高附加价值。

SOC设计的热点将集中在两个方面 一个是采用硬件 软件协调设计的高抽象度设计与验证体系 另一方面是与深亚微米对应的 CAD体系 前者帮助我们克服复杂性的危机 后者为进入微细化时代解决好功耗、布线延迟、可靠性等物理量的挑战所必须。这些设计工具与设计人员的创造力结合 将会不断地推进系统 LSI(规模集成电路 的进步 满足社会发展的需要。

杨刚,杨晞,汪道辉 .SoC与芯片设计方法 [J].微计算机信息,2003 19(2):56-57 72

郭鹏飞 .SoC设计中的软硬件协同设计 [J].中国电子产品,2004 6:73-75

陈庆德,苏桂兰 系统芯片设计的关键技术 [J].大理学院学报, 2005 4(5):7-9

王红,邢建辉,杨士元 .SoC测试访问机制 [J].微计算机信息,2006 22(2):117-119

作者简介 张艳 (1959-),女 汉族 江苏南京人 湖南信息职业技术学院讲师 本科 主要从事基础教学研究 胡桂 (1971-),男 汉族 湖南双峰人 湖南信息职业技术学院讲师 本科 主要从事基础教学研究。

Biography:Zhang yan(1959-),female(the Han nationality),Nanjing JiangShu Province,Lectuer at Hunan College of Information, BS, engaging in the reaseach of elements instruction;Hugui (1971-),male(the Han nationality), HuNan ShuangFeng Province, Lectuer at Hunan college of Information, BS, engaging in the reaseach of elements instruction.

(410208)湖南长沙 湖南信息职业技术学院 张艳 胡桂

通讯地址 :(410208)长沙望城 湖南信息职业技术学院基础实验中心 张艳

收稿日期 :2007.8.2(修稿日期 :2007.9.25)

(上接第 4页)

陈智育、温彦军、陈琪 vxworks程序开发实践 [M].人民邮电出版社 2004.5

[3]Philips Semiconductor. SJA1000 Standalone CAN Controller DataSheet[S].2002.

[4]VxWorks Programmer's Guide [S].WindRiver System,Inc.1998.

尚涛,施亮,吴智铭 实时操作系统 VxWorks下 I/O设备驱动程序的编写技巧 [J].微计算机信息, 2001 17-8:24-26

周春来 许化龙 刘根旺等 .CAN总线冗余方法研究 [J].2003 (卷第 期)

作者简介 杨君 (1982- 男 北京邮电大学自动化学院硕士研究生 主要研究方向现场总线技术和嵌入式技术。孙汉旭 (1960- 男 北京邮电大学教授 主要研究方向机器人技术及嵌入式技术 贾庆轩 (1964- 男 北京邮电大学教授 主要研究方向机器人技术及虚拟现实技术 史国振 (1974- 男 博士研究生 主要研究方向现场总线技术及嵌入式开发。

Biography:YangJun(1982-), Male, Beijing, Beijing university of post and telecom, postgraduate, automation, Main research field is BUS technology and Embed skill;Sun Han-xu(1960-), Male, Beijing, Beijing university of post and telecom, Professor, mentor of a Ph. D. student, Main research field is robot technology and Embed technique;Jia Qing-xuan(1964-), Male, Beijing, Beijing university of post and telecom, Professor, mentor of a Ph. D. student, Main research field is robot technique and virtual reality technology;Shi Guo-zhen(1974-), Male, Beijing, Beijing university of post and telecom, Ph D, Main research field is BUS technology and Embed skill

(424400)北京 北京邮电大学 杨君 孙汉旭 贾庆轩 史国振

通讯地址 :(424400)北京 北京邮电大学自动化学院 杨君

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

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

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

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

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

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

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

Programming:

1. 计算机软件基础数据结构 - 算法