XPE 在多功能显控台上的开发与应用

黄辽宁

(华中光电技术研究所,湖北 武汉 430074)

摘要:嵌入式计算机系统作为计算机应用的一个重要领域,应用广泛。XPE(Windows XP Embedded)具备 XP 与诸多嵌入式系统的优点。该文不仅简单介绍 XPE 操作系统的特点,而且详细说明了开发流程。结合 XPE 在海军多功能显控台上的具体应用,说明了其在复杂工作条件下的应用前景。

关键词:Windows XP Embedded;驱动程序组件;显控台;嵌入式;可靠性

中图分类号:TP316.7　　**文献标识码**:A　　**文章编号**:1009-3044(2010)28-8118-02

Development and Application of XPE in the Multifunctional Display Console

HUANG Liao-ning

(Institute of Optics and Electronics in Central China, Wuhan 430074, China)

Abstract: The embedded computer system takes an important domain in computer application, so it is widely applied. XPE(Windows XP Embedded)has some advantages of XP and many embedded system. This article describes Not only XPE operating system's features, and details in the development process. Take this the specific application of XPE in the Navy's Ship-bone multi-function display console as the example, Shows its good application prospects in complex working conditions.

Key words: Windows XP embedded; driver component; display console; embedded system; reliability

舰船上的装备种类繁多,具体要求也是各种各样。多功能显控台是各个装备的支撑平台,装备制造商只需要在其上编写软件以及制作特定的功能电路板即可。其运算处理单元采用 x86 架构的加固计算机,显示器采用 DVI 接口的加固显示器,其他功能电路板插接在计算机底板上通过 CPCI 总线、ISA 或自定义总线与运算处理单元通讯, 运算处理单元中的操作系统为 XPE 或 vxworks。Windows xp embedded 具有如下特点:1)完全兼容 windows xp 下开发的应用程序,可以满足各类程序的开发需要,开发工具丰富,可以降低开发成本;2)可以灵活定制,定制的操作系统只需要包括进需要用到的组件,系统占用空间小,运行速度快,适合于把系统安装在电子盘中;3)系统健壮性好,利用 XPE 中的 EWF 功能可以把系统配置成内存模式,这样对磁盘的读写次数会大大减少,且每次重新启动系统后所有对系统做的更改都会还原,这样系统永远都是新安装的状态,可以减少人为因素或突然掉电对系统的破坏。

1 开发环境

Windows XP Embedded Studio Tools 是一套完整的开发环境,包括开发工具和数据库,主要由以下四部分组成:1)目标分析器:包括 TA.exe 和 TAP.exe 两个应用程序。TA.exe 是 16 位应用程序,必须在 DOS 环境下运行,TAP.exe 是 32 位应用程序,可在 Windows 平台上使用;2)目标设计器:创建目标设备的新配置,并向其中添加相应组件检查相关性,确保配置具有创建运行时映像所需的适当组件,生成运行时的映像;3)组件设计器:用来设计新的组件并将其保存到组件数据库中,用来扩展嵌入式设备的功能以满足需求;4)组件数据库管理器:提供对组件设计器和目标设计器工具所使用的组件数据和存储库的管理功能。组件数据库可驻留在开发系统或服务器上。

2 开发步骤

开发步骤如图 1 所示,以下就按照这个顺序来论述。

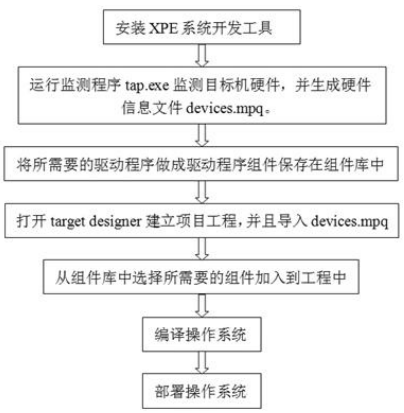

图 1　XPE 系统开发步骤

2.1 硬件分析

通过光盘的 winpe 帮助分析硬件,winpe 是一个运行在光盘上的操作系统, 可作为预安装环境,也可作为 TAP.exe 运行平台,用来分析目标机器的硬件配置情况。将 TAP.exe 拷贝到目标机器,运行检测目标机,生成一个默认文件名为 devices.pmq 的硬件列表定义文件,采用 XML 语言描述目标机器的硬件信息。

2.2 把驱动程序做成组件

我们做好的 XPE 系统连同硬件平台一同作为产品交付给客户,当然不能要求客户自己去安装硬件的驱动程序。所以我们把硬件的驱动程序都做成了组件形式,并且加入到 XPE 工程中。这样当 FBA 完成后,所有的硬件的驱动程序就都已安装完毕。实现这种功能就需要把每个硬件的驱动程序都做成驱动程序组件,并且并加入到工程中。

制作驱动程序组件有以下几个步骤[1]：

1)将驱动程序的安装版本安装在目标机上；

2)利用驱动精灵把已安装的驱动备份出来；

3)打开组件设计器(Component Designer)，通过菜单 File->Import..导入备份出来的.inf 驱动配置文件[2]，如图 2 所示。

4)设置和发布组件

① 鼠标右键点击 Repositories 节点，选择 Add Repository，在 Repository Properties 中设置组件仓库的名称(Penmount Files)及源文件所在的路径。

② 设置完成后，在 Component Properties 上设置组件的 Repository 为刚才新建的组件仓库。

③ 点击组件下 Group Memberships 节点，设置组件所属的分类路径，该设置代表了该组件在目标设机器中的组件树中的位置。

④ 完成组件相关的设置后，鼠标点选组件名称，使焦点停留在该组件的名称上，然后通过菜单 Edit->Release Component 来发布该组件。

⑤ 发布完成后，该组件相关属性则变灰不可更改。点击菜单中 Files->Save 保存组件。

⑥ 完成组件的开发后，最后需要将该组件导入到组件数据库中，导入方法为点击菜单 Tools-> Component Database Manager，打开组件数据库管理器，选择 Import 按钮。在弹出的对话框中点击 Import 进行导入。

导入成功后，以后就可以在目标设计器里使用该组件。图 3 中目标设计器里加入的前几个组件就是本项目中制作的，包括主板、声卡和网卡的驱动。

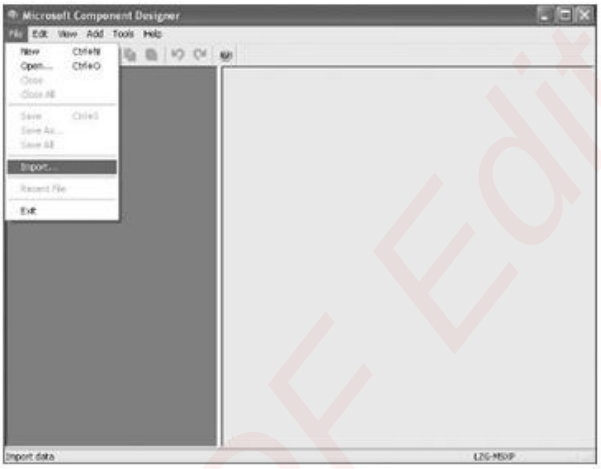

图 2　导入备份出来的.inf 驱动配置文件

图 3　本项目中所作的驱动程序组件

3 目标设计

运行目标设计器(Target Designer)，新建立一个工程，在本项目中名称是 7JCP1000。点击 File 菜单下的“import..”按钮打开“Choose File for Import”对话框，选择之前保存的 devices.mpq 文件，点击打开按钮；在出现的” Import File”对话框中点击开始按钮，开始导入硬件信息文件。导入过程中目标设计器会自动把跟硬件平台相关的组件加入到工程当中，包括我们在上段论述中自己做的驱动程序组件。

下面再手动加入一些需要的组件，比如本项目加入了 c 运行时库支持组件、中文语言支持组件、USB 设备支持组件、网络编程支持组件等等。一些组件还有设置选项，需要根据需求选择。

组件添加完毕后，要检查组件的关联性，因为所选中的组件需要其他组件的支持，反复检查关联性，并添加相应组件，直到无错误为止。

映像关联性检查无错误，就可以生成系统运行映像。

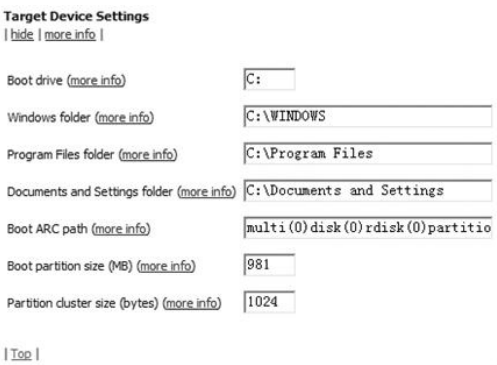

图 4　系统路径设置

4 系统发布和部署

在目标设计器里选择设置系统路径如图 4 所示。

系统部署在不同的文件系统下操作是不一样的。下面分两种情况进行说明[1]：

4.1 使用 FAT 或 FAT32 格式部署 XPE 操作系统

1)在 DOS 环境下，将 CF 卡或 IDE 磁盘分区，格式化并激活主分区；

2)通过 DOS 的 format c: /s 命令，将存储器的主分区格式化成能够进入 DOS 的引导盘；

3)拷贝 XPE 的 bootprep.exe 到 C:\ 下，启动进入 DOS，运行该程序创建引导区；

4)将制作好的 Windows XP Embeded 镜像拷贝到 C:\ ，以该分区引导重新启动，即可进入 FBA 阶段；

5)完成 FBA 后，系统自动重启，进入到 XPE 操作系统下；

4.2 使用 NTFS 格式部署 XPE 操作系统

该模式仅限于 IDE 磁盘，CF 卡由于被标示为可以动磁盘，使用 NTFS 构建可能会导致引导不成功：

1)如果原磁盘分区是 FAT 格式，则必须删除磁盘分区，重新创建；

2)通过 Win2000\XP 的磁盘管理工具或者 PQ，创建磁盘分区，格式化目标分区为 NTFS 格式；

3)激活磁盘主分区；

命令传输数据量为 256Byte(最大包)时,传输带宽为 318MByte/s。

b) 一次命令传输数据量大于 512Byte 时, 实际传输带宽基本趋于稳定,NWRITE 方式单向传输时, 最大实际传输带宽为 577MByte/s。

c)SWRITE 方式传输和 NWRITE 方式传输实际带宽基本相同。

d)在 2.5Gbps*4 速率情况下,SWRITE 方式和 NWRITE 方式传输的实际最大传输带宽和理论传输带宽比值为 64%。

2)NREAD 传输方式:

a)在 2.5Gbps*4 NREAD 方式单向传输时,512Byte 以内时,读传输带宽随数据量的增大而增长较快,一次读命令传输数据量为 512Byte 时基本接近最带带宽,再继续增加读命令的传输数据量,传输带宽增大幅度微小。

b)一次读命令传输数据量为一个 RapidIO 包的最大有效载荷 256Byte 时,实际传输带宽为 72MByte/s,2.5Gbps*4 NREAD 方式单向传输时,最大实际数据传输带宽为 90MByte/s。

c)NREAD 方式传输带宽只有 NWRITE 方式的 20%。

3 影响传输效率的因素分析

通过以上的分析,得出无论是 NWRITE 方式,SWRITE 方式还是 NREAD 方式,实际最大传输带宽和理论带宽都有一定的差距,在实际传输过程存在以下影响效率的因素:

a)发送方 DSP 处理数据的时间。接收方 FPGA 处理数据的时间,收到的数据先放到 DPRAM,处理内核再从 DPRAM 中取数据也要占用一定的时间。RapidIO 分包时间。

b)测试平台中包交换芯片处理 RapidIO 传输的延时。

c)RapidIO 协议传输时,通道中需要传输大量的控制符号和特殊码组字符等,用于链路维护、包定界、包确认、错误报告和错误恢复等。

d)尤其 NREAD 方式,传输效率很低,这是因为读操作由一个 NREAD 事物和一个 RESPONSE 事物组成,远端节点读本地存储器中的数据时,需要通过 RESPONSE 操作把请求数据和返回数据一起送到远端节点,所以花费的时间更长。

e)最后测试方法中存在计数计时的测量误差。

4 结论

文中设计了 RapidIO 串行通信的测试平台, 测试了 RapidIO 在 4 线 2.5Gbps NWRITE 方式、SWRITE 方式和 NREAD 方式下的实际传输带宽,通过实际传输带宽与计算的理论带宽对比,分析了影响传输性能的各种因素。同时得出 NWRITE 方式和 SRWITE 方式实际传输带宽基本一致,NREAD 方式传输的带宽比前两种方式小,效率较低。

参考文献:

[1] Sam Fuller.RapidIO 嵌入式系统互连[M].王勇,译.北京:电子工业出版社,2006.

[2] RapidIO Trade Association.RapidIO Interconnect Specification Rev.1.2[EB/OL].http://www.rapidio.org,2002.06.

[3] TMS320C6455 Serial RapidIO (SRIO)User's Guide(SPRU976a),TI Inc[EB/OL].http://www.ti.com,2006.05.

[4] TMS320C6455 Fixed-Point Digital Signal Processor Data Sheet(SPRS276H),TI Inc[EB/OL].http://www.ti.com,2005.05.

(上接第 8119 页)

4)拷贝镜像文件到该磁盘分区下,以该分区引导重新启动,即可进入 FBA 阶段;

5)完成 FBA 后,系统自动重启,进入到 XPE 操作系统下;

5 总结

本文给出了 windows xp embedded 在多功能显控台上的开发和应用。XPE 系统功能强大,运行速度快,实时性好,兼容性好,大大简化了程序的移植难度。而且还有对系统盘保护的 EWF 功能,非常适合于在恶劣的工作环境下使用。本系统进行了高低温循环实验,震动实验等环境可靠性实验,系统性能、可靠性表现良好。

参考文献:

[1] James Beau Cseri.Windows XP Embedded Step by Step[M].Annabooks/Rtc Books,2003.

[2] 丁露,陈家斌,吕少麟,等.基于 Windows XP Embedded 嵌入式车辆导航系统设计与实现[J].中国惯性技术学报,2006,14(6):27-29.

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
55. USB30 的高速信息传输瓶颈研究
56. 基于 IPv6 的 UDP 通信的实现

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
34. 基于 VxWorks 操作系统的 WindML 图形操控界面实现方法
35. 基于 GPU FPGA 芯片原型的 VxWorks 下驱动软件开发
36. VxWorks 下的多串口卡设计

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
40. 基于嵌入式 Linux 下的 USB30 驱动程序开发方法研究
41. Android 手机应用开发之音乐资源播放器
42. Linux 下以太网的 IPv6 隧道技术的实现
43. Research and design of mobile learning platform based on Android

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
22. Windows XP Embedded 在变电站通信管理机中的应用

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
20. 基于 PowerPC 的模拟量输入接口扩展

21. 基于 PowerPC 的车载通信系统设计

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
29. 基于 S3C6410 的 Uboot 分析与移植
30. 基于 ARM 嵌入式系统的高保真无损音乐播放器设计

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
25. 基于龙芯 2F 架构的 PMON 分析与优化
26. CPU 与 GPU 之间接口电路的设计与实现
27. 基于龙芯 1A 平台的 PMON 源码编译和启动分析

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