基于 PowerPC 的多屏系统设计

何双江

（华东计算技术研究所，上海 200233）

摘　要：描述了一种在 PowerPC 平台和 Linux 操作系统上实现多屏显示的方法，对提高图形性能方面做了进一步分析。另外，所用设计思路完全基于开源软件，相比专用定制平台，提供了更好的兼容性和扩展性。

关键词：PowerPC；Linux；多屏显示

中图分类号：TP311.52　　文献标识码：A　　文章编号：1672-7800(2012)001-0109-03

1　系统设计

硬件部分的设计主要以 CPU 和系统控制器为核心，通过各种接口总线连接外围部件。具体的原理框图如图 1 所示：

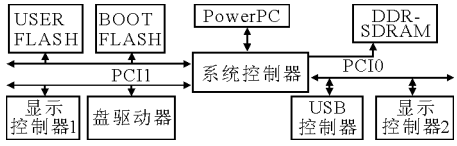

图 1　硬件系统结构图

显示控制器采用 ATI 公司的显示芯片，分别安装在系统控制器的两条 PCI 总线上。

系统软件全部采用源代码可控的软件。一方面获得开源软件的技术积累，另一方面也方便修改和定制。整个软件系统的结构如图 2。

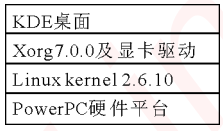

图 2　软件系统结构图

Linux 操作系统优点不再赘述。这里主要利用其图形功能。

XOrg 是 X Window System 的官方参考实现。它是开放源代码的自由软件。

在本文中，图形显示系统基于 Xorg 构建。Xorg 继承了 XFree86 的优点，同时又因跨平台兼容性强，硬件支持面广而被业界普遍采用。而正因为如此，各大显卡芯片厂商也相继推出了对应与 Xorg 的图形显示驱动，部分芯片还开放了驱动程序的源代码。另外，本文讨论的是一个多屏系统，而 Xorg 本身对多屏显示的支持是不错的。这也更有利于完成我们的研究计划。

2　关键技术

2.1　驱动程序及相关软件

2.1.1　显卡 BIOS 的作用

显卡 BIOS 固化在显示卡所带的一个专用存储器里。其中储存了显示卡的硬件控制程序和相关信息。显卡 BIOS 功能相当于主板 BIOS 对电脑的作用，它主要设定了显卡的一些参数、工作方式等等。此时时显卡能够按照事先设定好的程序进行工作，帮助显卡在没有驱动的情况下也能显示图像。

举个例子来说，我们经常听到有人通过刷新显卡 BIOS 的方式使显卡超频工作，其实就是修改了存储在显卡 BIOS 中的芯片最初的工作频率数值。

在 X86 平台上，一般显示卡只要正确地插在主板上就能开始工作，所以可以形象地说它是最典型的即插即用设备。开机后显示卡 BIOS 中的数据被映射到内存里并控制整个显卡的工作。在 DOS 下显示卡是不需要任何驱动程序的，Windows，Linux 进入到图形界面时也依赖于显示卡 BIOS 的支持。无论从哪个角度来说，显卡 BIOS 在显卡的工作工程中发挥了非常重要的作用。

2.1.2　PowerPC 上的 BIOS

前面说过，显卡 BIOS 最初的设计就是为 X86 平台专用的。所以显卡 BIOS 的二进制代码也是只能被 X86 架构的 cpu 直接读懂。那么，PowerPC 上应该如何利用 BIOS 里的信息来完成更多的工作呢？

要让 PowerPC 能够使用 X86 指令编写的 BIOS 代码，就需要一套机制来读懂 BIOS 内容，清楚 BIOS 执行过程中主机与显卡之间的交互，最后使显卡处于一个稳定完整的状态，并以此作为显卡最原始阶段的初始化驱动。这套过程需要的不仅仅是读取，更多的需要去理解指令所做的操作具体作用是什么，从而移植为当前需要的代码。

整个流程看起来大致如图 3 所示：

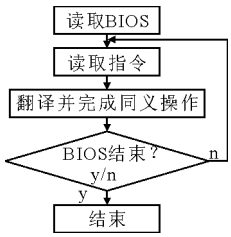

图 3　BIOS 流程

当然，仅仅有翻译指令的代码是不够的，因为 BIOS 代码中还有一些 I/O 指令，甚至一些中断操作。如何将 X86 的 I/O 指令完美的与 Linux 自身所带的 X86 虚拟 I/O 地址空间结合起来就需要更多的技术来实现。

同时，我们不能忘记了寄存器操作。因此在这里，合适的虚拟出一套机制，完成用 PowerPC 内存映射 x86 CPU 内部通用寄存器，段寄存器等等也是一定要做的事情。

我们分别用 C 语言完成了加减乘除、逻辑运算、堆栈操作这些基础指令集的功能，

每当我们从 BIOS 里读到一条指令，就在我们的 X86 虚拟机里执行一条前面代码示例中的一条对应的 C 函数代码，完成对应的寄存器、存储器和 IO 操作，从而完成了初始化工作。

除去少量细节部分的注意，当上面有提到的部分都处理完成的时候。我们就能够理解和“执行”BIOS 代码了。完成了这些事情，这时候我们得到了一张被“处理过”的显卡。显卡芯片、显存等等一些与硬件系统紧密相关的设置都不用再去重置了。需要值得提醒的一点是，大家熟悉的同步，垂直刷新的相关工作数值。此时并不一定需要初始化好，当然这部分代码可以由 BIOS 来完成，也有一部分系统将这部分工作由显示控制程序来做。

2.1.3　多屏系统与 PowerPC

上面分析了我们在执行显卡初始化部分的一些机制和难点。如果需要的仅仅是一张显卡，那么这些就已经够了。然而在一个多屏系统中，我们需要两张甚至三张，四张显卡，这时候情况就不一样了。

因为硬件地址译码窗口对齐大小的关系及传统 X86 BIOS 的做法局限，显卡 I/O 地址空间往往被局限在一个很小的范围内，这个范围一般是 64K(0x0－0xffff)。当处于不同系统 PCI 总线窗口的各个显卡需要的 I/O 地址空间出现了重叠(这个问题在某些体系结构的处理器平台上并不存在)，而最终却需要映射到不同的 PCI 物理 I/O 空间，出现了地址复用的问题。

在讨论我们的解决方法之前，我们先谈谈 Linux 中对 IO 系统的处理方式。

X86 体系结构的 CPU 为外设专门实现了一个单独地地址空间，称为“I/O 地址空间”或者“I/O 端口空间”。这是一个与 CPU 地 RAM 物理地址空间不同的地址空间，所有外设的 I/O 端口均在这一空间中进行编址。CPU 通过设立专门的 I/O 指令(如 X86 的 IN 和 OUT 指令)来访问这一空间中的地址单元(也即 I/O 端口)。这就是所谓的“I/O 映射方式”(I/O－mapped)。与 RAM 物理地址空间相比，I/O 地址空间通常都比较小，如 X86 CPU 的 I/O 空间就只有 64KB(0－0xffff)。显然这是“I/O 映射方式”的一个主要缺点。

PowerPC 体系结构的 CPU 只实现一个物理地址空间，即 RAM。在这种情况下，外设 I/O 端口的物理地址就被映射到 CPU 的单一物理地址空间中，而成为内存的一部分。此时，CPU 可以象访问一个内存单元那样访问外设 I/O 端口，而不需要设立专门的外设 I/O 指令。这就是所谓的“内存映射方式”(Memory－mapped)。

那么如何解决上面提到的重叠问题呢？从原理上来说，我们为 Linux 的 memroy 映射 I/O 方式提供了另外一条路，即增加一条映射路径。从而解决了显卡 I/O 空间重叠时，BIOS 的初始化方法。这个方法看起来大致是这样的(图 4)：

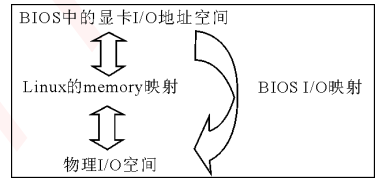

图 4　BIOS I/O 映射

上述这种方法可以解决这种重叠问题，当我们的 Xorg 在多块显卡上跑起来时，我们做了一些截屏。如图 5：

图 5　三屏系统

2.2　多屏显示中的触摸屏

2.2.1　优点

在我们的系统中，考虑到由于分辨率大，显示器多，鼠标来回移动会比较麻烦。在在这种特殊需求下，我们考虑加入触摸屏设计。通过触摸屏的控制，能够减轻对鼠标的依赖，方便快速的完成一些简单的操作，比如点击按钮，选中目标等。

在这里对加入触摸屏的优点稍微提一下。信息技术和自动化技术的高速发展使得直接人机交互过程越来越重要，触摸屏技术正是为了满足这种需求而产生的。触摸屏是一种最新的电脑输入设备，它可以让使用者只用手指轻轻地碰计算机显示屏上的图符或文字，就能实现对主机的操作，这样摆脱了键盘和鼠标操作，使人与机交互更为直截了当。它具有坚固耐用、反应速度快、节省空间、易于交流等许多优点。触摸屏是集信息显示、处理、通信和控制于一体的综合信息系统，在工程控制和商业服务等领域都有大量的应用。从大型飞行器的控制，到人们手中的手机，都可以看到它的身影。

2.2.2　实现方式及难点分析

我们的设计思路是一个触摸屏对应一个显示器。所

以，如果是三个显示器，则加入三个触摸屏的设计。

既然是多屏系统，多触摸屏的设计，那么多个触摸屏在同一个图形系统中的工作需要协调。触摸屏驱动程序需要能够支持多屏协同工作，这就涉及到对不同硬件事件的处理及图形系统的坐标翻译问题。同时，该驱动程序还要能够模拟双击，拖动，交换左右键等动作。

另外，由于当前大部分触摸屏驱动程序并没有为我们的硬件系统设计，要将驱动程序移植到 PowerPC 上，需要一定的移植调试工作。

3 性能分析

3.1 屏幕数量

到目前为止，我们已经验证过三屏显示系统。该系统通过两块显卡的三路输出接口，成功完成三路输出不同却又是统一桌面的图形系统。整个系统由一套鼠标键盘控制。鼠标能够在三个显示器上来回移动，键盘则根据屏幕焦点来确定输入对象。从连续的角度看来，应该是下面这种样子（图 6）：

图 6 三屏连续桌面

3.2 绘制性能

在开启扩展桌面显示功能后，我们对多屏显示系统的 2D、3D 性能做了全面的测试。测试结果如表 1。

表 1 绘制性能测试

	单屏	第一屏	第二屏	第三屏
绘制时间	8.1s	10.3s	10.5s	7.3s

我们重复绘制一副图像多次。根据整个过程的绘制时间来判断性能高低。很显然，耗费时间越少，绘制性能越高。

从结果中对比单屏和三屏的现实性能，三屏系统明显耗费时间更多。可以想象，多屏系统因为扩展了分辨率，增加了显示面积，耗费了更多显存等原因，性能较单屏系统应该会有所下降。这个结果虽然令我们难过，却也在情理之中。然而，无意中测试到的另一些数据却引起了我们的注意。也许你已经发现，在三屏中第三个屏绘制图像的性能明显好过其他两个屏，甚至好过单屏的性能。这是为什么呢？

我前面讲过，我们的三屏系统是由两块显卡搭建而成的，第一块显卡双输出，分别输出到第一个和第二个屏，第二块显卡输出到第三个屏。如果图形测试跑在第二块显示卡上的时候，所带来的性能提高。我们分析认为，这主要是由于第二块显卡在图形系统中处于副显示卡的角色，从而负载相比主显示卡比较轻，所以性能相对更好。毕竟，主显示卡还要为 FrameBuffer 模式预留一些资源，还要显示更多的桌面控件等等，这些都会导致性能的浪费。无论如何，我们发现的这个结果不失为一种提高显示卡工作性能的方法，可以作为一些对显示性能有较高要求的特殊系统一种设计思路。

3.3 功能支持

通过本文描述方法搭建出来的多屏系统。功能支持非常完成。除了对 3D 性能支持不是很好之外，该系统能够全面支持 Xorg 的各种 2D 加速及扩展功能。包括 XAA，EXA，硬件光标，帧缓冲模式，OpenGL 图形库... 等等。

在各扩展功能支持全面的基础上，我们的多屏系统在分辨率方面，相对单屏系统，有着无可比拟的优势。以目前的三屏系统为例，已经实现了 4800×1200 的分辨率（如图 7），这种分辨率在单屏系统中很难实现，即使显示控制器支持，显示器也很难达到要求。

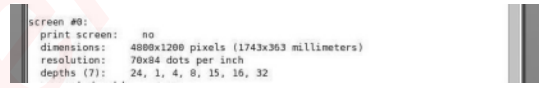

图 7 高分辨率

多屏系统的解决方案采用将屏幕分割的思路，在多个显示器上实现统一桌面，既方便了应用程序开发和使用，也增加了显示面积，不可不谓一箭双雕。

4 应用前景

针对于当前计算机领域日益增大的显示性能需求，本文中描述的思路主要用于在基于国产化硬件平台上研究如何增强图形显示的综合性能，如何扩展更多实用的显示功能。多屏领域是今后图形显示方面一个比较重要的方向，在高性能显示设备，显示控制设备方面有着较大的市场前景。

参考文献：

[1] Jonathan Corbet：Linux 设备驱动程序[M]. 北京：中国电力出版社，1991.

[2] Daniel P. Bovet and Marco Cesati：深入理解 Linux 内核[M]. 北京：中国电力出版社，1989.

[3] Karim Yaghmour：构建嵌入式 Linux[M]. 北京：中国电力出版社，2009.

（责任编辑：王 钊）

Design of a Multi-Screen System Based on PowerPC

Abstract：Describe a way to implement multiscreen system on PowerPC and Linux platform. Analyse the performance of the graphic processing. Further more，all the used software is open source software which was widely used. Compared with those customized system，This solution provide more compatibility and expansibility.

Key Words：PowerPC；Linux；Multi-Screen Display

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
57. 一种基于 IPv6 的流媒体传送方案研究与实现
58. 基于 IPv4-IPv6 双栈的 MODBUS-TCP 协议实现
59. RS485CAN 网关设计与实现
60. MVB 周期信息的实时调度
61. RS485 和 PROFINET 网关设计
62. 基于 IPv6 的 Socket 通信的实现
63. MVB 网络重复器的设计
64. 一种新型 MVB 通信板的探究
65. 具有 MVB 接口的输入输出设备的分析
66. 基于 STM32 的 GSM 模块综合应用
67. 基于 ARM7 的 MVB CAN 网关设计

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
37. VxWorks 内存管理机制的研究
38. T9 输入法在 Tilcon 下的实现
39. 基于 VxWorks 的 WindML 图形界面开发方法
40. 基于 Tilcon 的 IO 控制板可视化测试软件的设计和实现
41. 基于 VxWorks 的通信服务器实时多任务软件设计
42. 基于 VXWORKS 的 RS485MVB 网关的设计与实现
43. 实时操作系统 VxWorks 在微机保护中的应用
44. 基于 VxWorks 的多任务程序设计及通信管理

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
44. 基于 linux 和 Qt 的串口通信调试器调的设计及应用
45. 在 Linux 平台上基于 QT 的动态图像采集系统的设计
46. 基于 Android 平台的医护查房系统的研究与设计
47. 基于 Android 平台的软件自动化监控工具的设计开发
48. 基于 Android 的视频软硬解码及渲染的对比研究与实现
49. 基于 Android 移动设备的加速度传感器技术研究
50. 基于 Android 系统振动测试仪研究
51. 基于缓存竞争优化的 Linux 进程调度策略

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

23. XPE 在多功能显控台上的开发与应用
24. 基于 Windows XP Embedded 的 LKJ2000 仿真系统设计与实现
25. 虚拟仪器的 Windows XP Embedded 操作系统开发
26. 基于 EVC 的嵌入式导航电子地图设计
27. 基于 XPEmbedded 的警务区 SMS 指挥平台的设计与实现
28. 基于 XPE 的数字残币兑换工具开发

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
22. 基于 PowerPC 的嵌入式系统中通用 IO 口的扩展方法
23. 基于 PowerPC440GP 型微控制器的嵌入式系统设计与研究
24. 基于双 PowerPC 7447A 处理器的嵌入式系统硬件设计
25. 基于 PowerPC603e 通用处理模块的设计与实现
26. 嵌入式微机 MPC555 驻留片内监控器的开发与实现
27. 基于 PowerPC 和 DSP 的电能质量在线监测装置的研制
28. 基于 PowerPC 架构多核处理器嵌入式系统硬件设计

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
31. UBoot 在 Mini6410 上的移植
32. 基于 ARM11 的嵌入式 Linux NAND FLASH 模拟 U 盘挂载分析与实现
33. 基于 ARM11 的电源完整性分析
34. 基于 ARM S3C6410 的 uboot 分析与移植
35. 基于 S5PC100 移动视频监控终端的设计与实现

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
28. 基于 PC104 工控机的嵌入式直流监控装置的设计
29. GPGPU 技术研究与发展
30. GPU 实现的高速 FIR 数字滤波算法
31. 一种基于 CPUGPU 异构计算的混合编程模型
32. 面向 OpenCL 模型的 GPU 性能优化
33. 基于 GPU 的 FDTD 算法
34. 基于 GPU 的瑕疵检测
35. 基于 GPU 通用计算的分析与研究

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
7. 数据结构考题 - 第 1 章 绪论
8. 数据结构考题 - 第 2 章 线性表
9. 数据结构考题 - 第 2 章 线性表 - 答案