在 Linux 平台上基于 QT 的动态图像采集系统的设计

杨国田* 毛小军
(华北电力大学自动化系,北京 102206)

摘 要:结合嵌入式 Linux 结构简单、灵活、低成本、高性能的特点以及图像采集系统的广泛应用,介绍了在 Linux 系统下基于 QT 开发图像采集系统的方法,给出了基于 AT89C2051 处理器以及 SAA7111A 视频采集芯片的嵌入式图像采集系统的设计方案,分析了图像采集卡的工作原理,并结合 CPLD 实现了对整个采集过程的控制。介绍了 QT 和 Linux 下串口通信的特点,并对如何编写基于 QT 的串口通信、图像传输、图像处理和图像显示等程序进行了讨论。

关键词:Linux;串口通信;QT;图像采集;图像处理

随着信息技术的飞速发展,信息采集不再停留在文字类型上,实时的、高品质的图像信息是许多决策者和科技人员获得动感和感性认识的源泉。视频采集在这方面发挥了很大的作用,越来越受到人们的重视。在众多的视频采集系统中,嵌入式的视频采集以其小巧、灵活、低成本、高性能的特点而独具优势。

Linux 操作系统,具有可移植性好、网络功能强、有优秀的 GNU 编译工具支持等优点。更重要的是 Linux 的开放源代码和免费的优点使得系统成本显著降低。一流的程序设计和开发加上测试的开放性使得 Linux 系统非常可靠和稳定,因而越来越多的人开始使用 Linux 开发应用程序。

Qt 是 Trolltech 公司推出的一个多平台的 C++ 图形用户界面应用程序框架。它提供给应用程序开发者建立艺术级的图形用户界面所需的所用功能。Qt 是完全面向对象的很容易扩展,并且允许真正地组件编程。自从 1996 年早些时候,Qt 进入商业领域,它已经成为全世界范围内数千种成功的应用程序的基础。Qt 也是流行的 Linux 桌面环境 KDE 的基础,KDE 是所有主要的 Linux 发行版的一个标准组件。

在计算机控制领域,经常需要实现计算机之间或计算机与其他设备之间的通信,串口通信作为一种灵活、方便、可靠的通信方式被广泛采用,在 Linux 下的串行通信编程主要是依靠内核提供的 termios 结构对串口进行参数设置,并对串口对应的设备文件进行读写。

1. 系统硬件设计

(1) 系统硬件结构

本系统通过 CCD 摄像头将模拟图像送入图像采集卡转换成数字图像信息,再送入图像采集终端,图像采集终端获得图像数据后进行处理并显示图像。本系统可应用于远程检测和远程监控等多个领域。系统总体方案如图 1 所示。

(2) 图像采集卡

图像采集卡完成模拟图像信号的采集输入,并将模拟图像信号转换为数字图像信号,并传输给图像采集终端接口。其内部逻辑框图如图 2 所示。

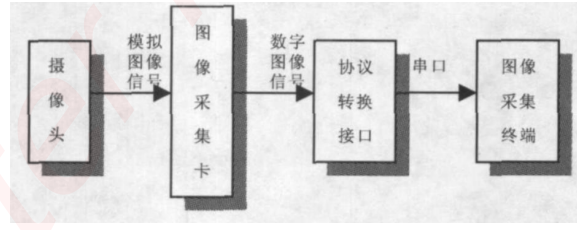

图 1 系统总体方案

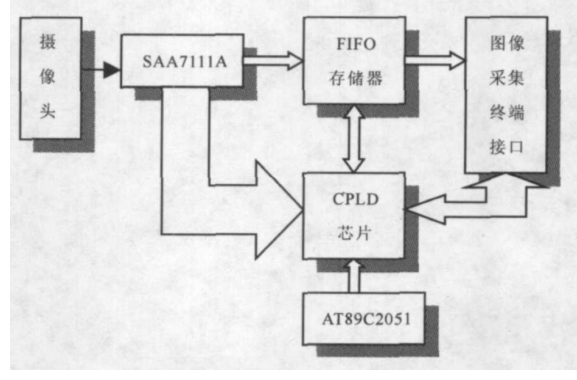

图 2 图像采集卡硬件逻辑框图

视频解码芯片采用 Philips 公司推出的增强型视频输入处理器 SAA7111A。本系统中单片机 AT89C2051 先对 CPLD 芯片初始化,对 SAA7111A 的初始化是通过 CPLD 芯片所提供的一对引脚 SDA 和 SCL 进行。SAA7111A 的初始设定为:一路模拟视频信号输入、自动增益控制、625 行 50Hz PAL 制式、YUV 422 16bits 数字视频信号输出、设置默认的图像对比度、亮度及饱和度。CPLD 实现的主要功能是将 SAA7111A 输出的图像数据送入 FIFO 存储器中暂存,待接收到图像采集终

端的采集命令后通过接口将图像送出。与图像采集终端的接口主要由 16 位宽的数据总路线、读信号控制线 RD、写信号控制线 WR 组成。CPLD 芯片采用 Lattice 公司生产的 ispGAL22V10。

(3) 图像采集终端

图像采集终端完成接收图像采集卡的数字图像数据,传输图像数据以及图像处理和显示。其内部逻辑框图如图 3 所示。

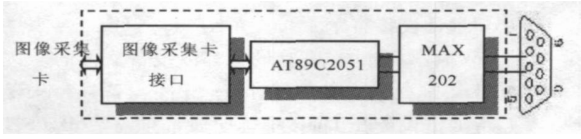

图 3　图像采集终端硬件逻辑框图

图像采集终端利用数据总线和通用 I/O 口实现与图像采集卡的连接。MAX202 用于完成 TTL 到 RS-232C 的电平转换,可通过两根收发线接至 PC 机的 RS-232C 串行口,实现图像数据和收发指令的传输功能。

2. 系统软件设计

(1) 图像采集卡软件设计

图像采集卡程序控制着整个图像采集工作的全过程。其程序流程如图 4 所示。采集终端给串口发送命令后,AT89C2051 开始初始化过程,然后,初始化图像采集卡模块,设定图像采集模式,启动采集过程。接下来,系统将连续查询图像采集控制寄存器的值,查询是否完成一幅图像的采集,如果没有完成,继续查询图像采集控制寄存器,直到查询到采集完成一幅图像,启动图像读取子程序,把图像数据从 FIFO 中读取到串口送出。

(2) 图像处理与显示程序设计

①图像处理与显示程序框架

本程序共由 QApplication、QWidget、QBmpShow 和 QYUVToBmp 四个大类组成,其中 QBmpShow 和 QYUVToBmp 是自定义的类,其程序框架如图 5 所示。

QApplication 是 QT 程序必须有的一个类,每一个 QT 应用程序都要包含 QApplication 对象。QApplication 管理应用程序各种各样的资源。

QWidget 类是所有用户界面对象的基类。

QBmpShow 类实现 BMP 图像的创建、保存及显示。

QYUVToBmp 实现 YUV 格式的图像格式转换成 RGB 格式。

②图像处理程序

图像处理程序主要完成将 YUV 格式的图像格式转换成 RGB 格式,并创建和保存 BMP 文件,以实现图像的显示。

A. 图像格式转换

由于 SAA7111A 输出的数字图像信号是 YUV 格式,我们需要把 YUV 三个分量经过矩阵运算变换为 RGB 信号,以便在显像管上显示。RGB 和 YUV 的对应关系用代数方程式表示如下:

R = Y + 1.371 * (V - 128)

G = Y - 0.698 * (V - 128) - 0.336 * (U - 128)

B = Y + 1.732 * (U - 128)

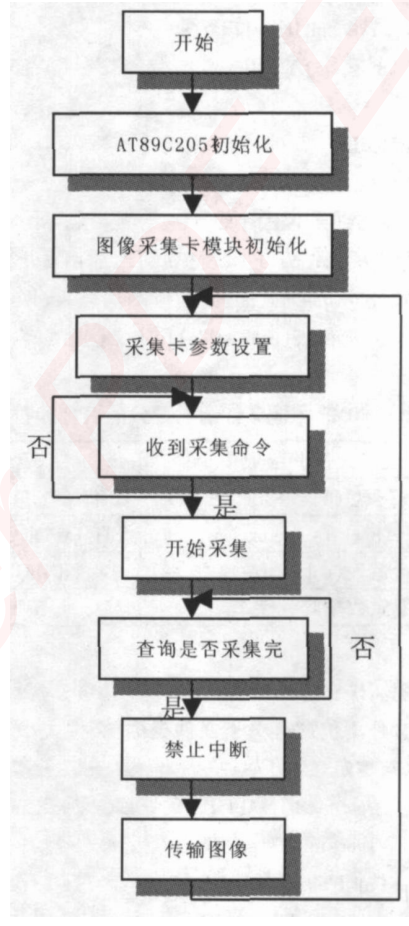

图 4　图像采集卡程序流程图

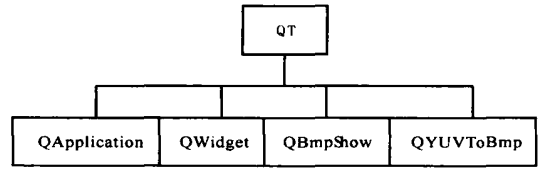

图 5　图像处理与显示程序框架图

程序示例如下:

```
BOOL QYUVToBmp::YUV12BMP(BYTE * pYUVBuf, BYTE *pRGBBuf, int i)
{
pRGBBuf[2] = pYUVBuf[1] + 1.371 * (pYUVBuf[2] - 128);
pRGBBuf[1] = pYUVBuf[1] - 0.698 * (pYUVBuf[2] - 128) - 0.336 * (pYUVBuf[0] - 128);
pRGBBuf[0] = pYUVBuf[1] + 1.732 * (pYUVBuf[0] -
```

```
128);
    pRGBBuf[5] = pYUVBuf[3] + 1.371 * (pYUVBuf[2] -
128);
    pRGBBuf[4] = pYUVBuf[3] - 0.698 * (pYUVBuf[2] - 128)
- 0.336 * (pYUVBuf[0] - 128);
    pRGBBuf[3] = pYUVBuf[3] + 1.732 * (pYUVBuf[0] -
128);
    return TRUE;
    }
```

B. 创建和保存 BMP 图像

位图文件(Bitmap - File,BMP)可看成由 4 个部分组成:位图文件头(bitmap - file header)、位图信息头(bitmap - information header)、彩色表(color table)和定义位图的字节阵列,它们的名称和符号如表 1 所示。

表 1 BMP 图像文件组成部分的名称和符号

位图文件的组成	结构名称
位图文件头(bitmap - file header)	BITMAPFILEHEADER
位图信息头(bitmap - information header)	BITMAPINFOHEADER
彩色表(color table)	RGBQUAD
图像数据阵列字节	BYTE

a. 位图文件头

位图文件头包含有关于文件类型、文件大小、存放位置等信息,其结构定义如下:

```
typedef struct tagBITMAPFILEHEADER { /* bmfh */
unsigned short int bfType;
unsigned int bfSize;
unsigned short int bfReserved1;
unsigned short int bfReserved2;
unsigned int bfOffBits;
} BITMAPFILEHEADER;
```

由于在 Linux 下不能使用 WORD 和 DWORD,所以我们将将 WORD 换为 unsigned short 将 DWORD 换为 unsigned int,在这种情况下,为了防止读取图像时数据丢失,最好在程序中依此读取每个成员,如:

```
fread(&BmpHeader.type,sizeof(BmpHeader.type),1,fp);
fread(&BmpHeader.size,sizeof(BmpHeader.size),1,fp);
fread(&BmpHeader.reserved1,sizeof(BmpHeader.reserved1),
1,fp);
fread(&BmpHeader.reserved2,sizeof(BmpHeader.reserved2),
1,fp);
fread(&BmpHeader.offset,sizeof(BmpHeader.offset),1,fp);
```

而不是将整个文件头一次读取到结构中,如:

```
fread(&BmpHeader,sizeof(FILEHEADER),1,fp);
```

b. 位图信息头

位图信息用 BITMAPINFO 结构来定义,它由位图信息头(bitmap - information header)和彩色表(color table)组成,前者用 BITMAPINFOHEADER 结构定义,后者用 RGBQUAD 结构定义。BITMAPINFO 结构具有如下形式:

```
typedef struct tagBITMAPINFO {/* bmi */
BITMAPINFOHEADER bmiHeader;
RGBQUAD bmiColors[1];
} BITMAPINFO;
```

(3)串口通信软件设计

①串口通信设备简介

在 Linux 操作系统中,所有设备都以设备文件的形式存在于/dev 目录下,每一个设备都对应一个主设备号和从设备号。串口等通信设备在 Linux 系统中通常对应于/dev/ttyS0、/dev/ttyS1、/dev/ttyS2、...,其中编号为 0 的是第一个串口(即 COM1),标准的串口和可扩展多口串行通信卡都以这种方式映射,其主设备号都为 4,从设备号各不相同,如表 1。

表 2 Linux 系统串行设备文件及设备号

串口名称	设备文件	主设备号	次设备号	I/O 地址
COM1	/dev/ttyS0	4	64	3F8
COM2	/dev/ttyS1	4	65	2F8
COM3	/dev/ttyS2	4	66	3E8
COM4	/dev/ttyS3	4	67	2E8

②串行通信编程的主要数据结构

在 Linux 中,串行口的属性参数全部反映在一个 struct termios 中,它定义在头文件 termios. h 中,其结构如下:

```
#define NCCS 19
struct termios{
tcflag-t c-iflag; /*输入选项 */
tcflag-t c-oflag; /*输出选项 */
tcflag-t c-cflag; /*控制选项 */
tcflag-t c-lflag; /*端口信号线选项 */
cc-t c-line; /* 线路规章 */
cc-t c-cc[NCCS]; /*控制字 */
speed-t c-ispeed; /* 输入速率 */
speed-t c-ospeed; /* 输出速率 */
};
```

其中含有大约 50 个标志位,这些标志位告诉串口驱动程序如何处理收发的字符,如何设置通信参数等,在头文件 termios. h 中给出了所有标志位的定义。需要注意的是,不能直接初始化 termios 结构,而应该用位运算(AND、OR 和 NOT)设置或清除所有的标志位。

③串口通信设置函数

为了方便编程,Linux 系统还包含了一系列针对于 termios 结构的设置函数,用来完成获取和设置串口属性,设置波特率等功能。另外,在 Linux 下串口是以设备文件的形式存在,因此其 I/O 函数与文件 I/O 函数是一致的,可以用 open 函数来打开一个串口,用 read 和 write 完成发送与接收,用 close

函数关闭串口等。

④图像采集串口通信的程序设计

部分程序示例如下:

```
#include <termios.h>
... ...
/* 包含必要的头文件 */
#define com1 "/dev/ttyS0"
... ...
int main (int argc, char **argv)
{
QApplication App(argc,argv);
myWindow w;
int fd, res, i;
struct termios oldtio, newtio;
/* 定义 termios 结构, oldtio 用于保存原有的串口属性, newtio 用于设置新的串口属性 */
BYTE buf[512];
Speed-t ispeed, ospeed;
/* 分别定义输入和输出速率变量 */
... ...
fd = open (com1,O RDWR|O NOCTTY); /* 打开串口 */
if (fd < 0)
{
printf ("serial errorn"); exit (1);
}
tcgetattr(fd, &oldtio); /* 将原有的串口属性保存在 oldtio 中 */
bzero (&newtio, sizeof (newtio)); /* 清空 newtio 结构 */
newtio.c cflag& = ～PARENB; /* 忽略奇偶校验 */
newtio.c cflag& = ～CSTOPB; /* 使用一个停止位 */
newtio.c cflag& = ～CSIZE; /* 屏蔽字符大小 */
newtio.c cflag| = CS8|CREAD|CLOCAL; /* 设置数据宽度、使能接收等 */
newtio.c oflag| = OPOST; /* 选择输出处理 */
newtio.c lflag = 0; /* 选择非行规模式、不回显字符等 */
newtio.c cc[VM IN] = 1; /* 指定返回需读取的最小字符数 */
newtio.c cc[VTIME] = 0; /* 设置超时时间 */
ispeed = B115200;
ospeed = B115200;
cfsetispeed (&newtio, ispeed); /* 设置输入波特率 */
cfsetospeed (&newtio, ospeed); /* 设置输出波特率 */
tcflush (fd, TCIFLUSH); /* 刷新串口缓冲区 */
tcsetattr(fd, TCSANOW, &newtio); /* 将新的设置立即生效 */
QObject::connect(mod0, SIGNAL(clicked()), SLOT(write(fd, buf, 5));
/* 发送 FF00 命令,开始采集 */
... ...
tcflush (fd, TCOFLUSH);
tcsetattr(fd, TCSANOW, &oldtio); /* 恢复串口的原始属性 */
close (fd); /* 关闭串口 */
myApp.setMainWidget(&w);
w.show();
return myApp.exec();
}
```

3. 结语

本系统可以应用于电厂图像监测、交通、防盗等多个领域。Linux 环境下硬件设备的操作过程基本上和 Windows 平台使用 API 函数编程差不多,但 Linux 提供了更多的灵活性,结构简单,成本低,充分利用它们可以方便地实现各种复杂的应用。Qt 库中使用信号和槽机制,而不是 Windows 下的回调机制,使组件之间的连接更加容易实现。

参考文献:

[1] Xteam(中国) 软件技术有限公司. QT 程序设计,清华大学出版社, 2002.

[2] Gary Frerking, Serial Programming HOWTO [EB/OL], Aug 26th. 2001.

[3] 于明俭,陈向阳,方汉. Linux 程序设计权威指南[M]. 北京:机械工业出版社,2001. 109 - 120.

[4] 怀石工作室. Linux 上的 C 编程[M]. 北京: 中国电力出版社, 2000. 389 - 391.

[5] 王鑫,陈晓竹. 徐倩. USB 接口的嵌入式图像采集与显示系统. 中国计量学院信息工程学院,浙江杭州,2006 - 01 - 012.

[6] 贺科学,李鸿. 基于 CPLD 和接触式图像传感器的图像采集系统. 单片机与嵌入式系统应用, 2006 - 04 - 018.

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
37. VxWorks 内存管理机制的研究

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

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

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
7. 数据结构考题 - 第 1 章 绪论