基于ARM9的嵌入式系统设计及EPA设备实现的研究

邓俊华，杜玉晓，董 建
（广东工业大学 自动化学院，广东 广州 510006）

摘 要：针对EPA设备需要满足工业上实时性要求及与其他设备协调地工作等问题，研究了ARM微处理器和Linux操作系统的关键技术，设计以ARM微处理器为核心、Linux操作系统为软件平台的嵌入式系统。以ARM微处理器为核心的嵌入式系统应用于EPA设备能够满足工业实时性要求，并提供丰富的外围接口，为EPA设备的进一步开发奠定了基础。

关键词：EPA；嵌入式系统；Linux；文件系统；设备驱动
中图分类号：TP393 **文献标识码**：A

Research of the embedded system based on ARM9 and application in EPA device

DENG Jun Hua,DU Yu Xiao,DONG Jian
(Automation Colloege,Guangdong University of Technology , Guangzhou 510006,China)

Abstract: To satisfy the real-time attribute in industry field and cooperate with other devices better, this paper introduces the core technic about ARM microprocessor, Linux operating system and designs a set of embedded system. The EPA device based on embedded system can satisfy the real-time attribute in industry field and cooperate with other devices better. It makes the EPA device expand its functions easier.

Key words: EPA; embedded system; Linux; file system; device driver

嵌入式系统与Intenret技术的结合已经成为未来嵌入式系统的发展趋势，而基于ARM的嵌入式系统由于其低功耗、低成本、高性能等优势已经广泛地应用于工业控制领域。而与此同时，Linux由于性能优越、支持硬件平台广泛、源代码公开、具有极强的网络功能等优点成为设计嵌入式系统一种很好的选择。

本文从嵌入式系统的硬件电路和软件开发两个方面进行设计。在硬件设计上采用Atmel公司生产的AT91RM9200微处理器为CPU，选用8 MB的Flash和32 MB的SDRAM作为系统存储器，扩展了以太网接口、串行接口等外围通信设备以及输入输出接口，根据处理器和其他接口芯片的要求设计了电源电路、晶振电路、Flash及SDRAM存储器接口电路、以太网接口电路、串行接口电路和扩展I/O接口电路。使用4层贴片工艺设计了系统PCB印刷电路板，焊接和安装了贴片元件，并进行了电路调试等过程。在软件设计上基于Linux操作系统，分析了Linux操作系统的引导程序(Bootloader)的结构、工作流程及内核的启动过程； Bootloader移植和内核裁剪技术，移植了嵌入式Linux的引导过程；Linux文件系统的结构、根文件系统的层次和文件的管理方法；Linux设备管理方法和设备驱动程序的中断实现方法；Linux字符设备各驱动程序设计技术；编写了A/D转换的驱动程序和外扩I/O接口的驱动程序。

1 嵌入式EPA设备硬件设计

1.1 EPA设备结构框架

EPA（Ethenret for plant Au tomation）设备是基于工业以太网技术的分布式控制系统的底层设备，该设备具有模拟量输入输出、开关量输入输出和1个10Mb/s的以太网接口、1个RS485的通信接口。通过以太网及RS485通信方式，EPA设备在现场中既可作为主设备也可作为从设备，可方便地与其他设备进行通信。EPA设备的硬件结构图如图1所示。

原有的EPA设备以Z-World公司的RCM2210处理器为核心，完成电流、电压、温度传感器信号的输入，标准电流信号的输出和开关量电流输出的功能。由于RCM2210是以8位微处理器为基础并且可以使用的V0口资源有限，因此，对模块的速度以及功能扩展有很大的限制。

本文的主要目的就是设计以32位微处理器AT91RM9200为CPU外扩Flash、SDRAM存储器和以太网接口的嵌入式系统来代替RCM22100处理器。AT91RM 9200的主频达到180MHz时，其性能可以高达200MIP，这样的处理速度可以更好地满足工业上对实时性的要求。同时AT91RM9200有丰富的外围接口，为模块的功能扩展提供了更大的空间。

1.2 基于ARM的嵌入式系统硬件结构

嵌入式系统的硬件结构如图2所示，主要由以下各部分组成:

(1) 电源电路:输入5V，经过DC-DC变换转换为1.8V和3.3V，给系统内各器件提供工作电压。

(2) 晶振电路:18.432 MHz有源晶振经过倍频分别为ARM940T核/系统提供180 MHz的时钟频率。

(3) 微处理器: AT91RM9200是系统的工作和控制中心。

(4) Flash:可存放引导程序、嵌入式操作系统、用户应用程序或其他在系统掉电后需要保存的数据。

(5) SDRAM: 是系统代码的运行场所。

(6) 网络端口: 10/100(Mb/s)速率的RJ45接口，为系统

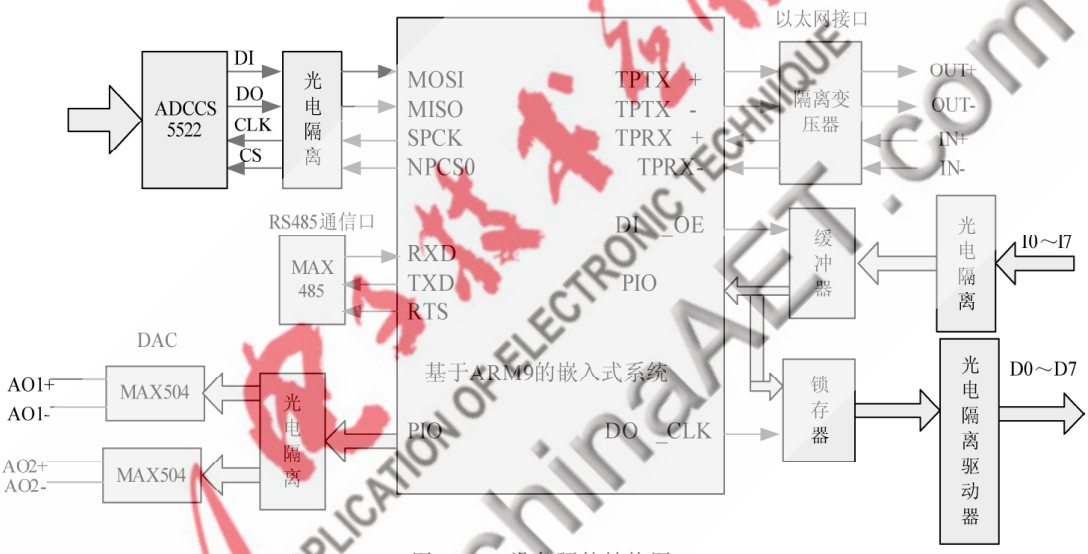

图1 EPA设备硬件结构图

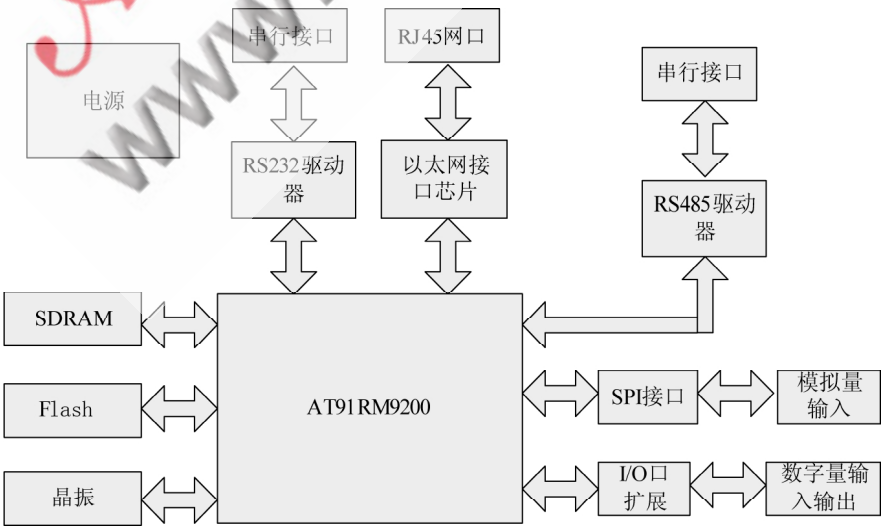

图2 系统硬件结构图

提供以太网接入的物理通道。

(7) 串行接口:用于 AT91RM9200 系统短距离双向串行通信。

1.3 以太网接口电路

作为一款优秀的网络控制器，基于 AT91RM9200 的系统若没有以太网接口，就会大大降低其应用价值，因此就整个系统而言，以太网接口电路应是必不可少的，但同时相对较复杂。从硬件的角度看，以太网接口电路主要由 MAC 控制器和物理层接口两大部分构成。

AT91RM9200 微处理器内嵌一个以太网控制器，支持媒体独立接口 MII(Media Independent Interface)，可在半双工或全双工模式下提供 10/100(Mb/s)的以太网接入。在本设计中，使用 DAVICOM 公司的 DM9161 作为以太网的物理层接口；DM9161 是一款低功耗、高性能的 CMOS 芯片，支持 10M 和 100M 的以太网传输，它起编码、译码输入和输出数据的作用[1]。提供了 IEEE802.3 标准定义的 MII，它包括接收数据总线和发送数据总线，来控制物理层和 MAC 的数据传输；使用一个简单的两线制串行接口来通过 MII 控制物理层并接收来自物理层的信息，其串行控制接口包括数据时钟(MDC)和数据输入输出(MDIO)；Mil 串行管理包括 1 个数据接口、基本寄存器设置和 1 个针对寄存器设置的串行接口。通过这个接口可以控制和配置很多物理层设备，得到状态和错误信息，并且确定 PHY 设备的工作方式和功能。

2 基于嵌入式 Linux 的 EPA 设备软件设计

由于 Linuxr 遵循 GPL,所以任何对 Linux 定制于嵌入式设备感兴趣的人都可以从 Internet 免费下载其内核和应用程序，并开始移植或开发，按照实际需要增减系统内核。

一个小型的嵌入式 Linux 系统需要下面 3 个基本元素：引导实用程序；Linux 微内核（由内存管理、进程管理和定时服务构成）；初始化过程。要实现最低限度的工作能力，还需添加：硬件驱动程序、1 个或多个应用进程，以提供所需功能。随着要求的增加，可能还需要：1 个文件系统（可以是在 ROM 或者是 RAM 里）；1 个图形用户接口（GUI）；TCP/IP 网络栈。由此可以归纳构建嵌入式 Linux 的步骤为[2]:

(1) 编写 Bootloader 用于加载嵌入式 Linux 内核到内存中。

(2) 重新编译 Linux 内核，去掉内核中不需要的模块。

(3) 构建文件系统。

(4) 运行应用程序。

可从以上 4 方面构建符合自己要求的嵌入式 Linux 的方法。嵌入式执行流程如图 3 所示。

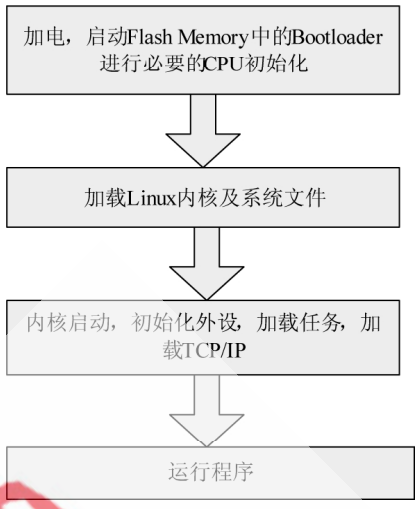

图 3 嵌入式系统 Linux 系统流程

2.1 引导加载程序

嵌入式 Linux 系统的引导过程如下：

(1) 处理器重新启动后,首先执行启动代码以初始化内存控制器以及片上设备,然后配置存储映射。

(2) 引导加载程序(Bootloader)把内核从 Flash 等固态存储设备加载到 RAM，然后跳转到内核的第一条指令处执行。

(3) 内核首先配置微处理器的寄存器，然后调用 start-kernel（它是与微处理器体系结构无关的开始点）。

(4) 内核初始化高速缓存和各种硬件设备。

(5) 内核挂装根文件系统。

(6) 内核执行 init 进程。

(7) init 进程加载运行时共享库，读取其配置文件。

(8) init 最后进入用户会话阶段。

在此过程中引导加载程序的作用是非常重要的。

2.2 嵌入式 Linux 内核配置

内核是一个操作系统的核心，它负责管理系统的进程、内存、设备驱动程序、文件和网络系统，决定着系统的性能和稳定性。Linux 的核心是 vmlinuz 文件，该文件控制着整套系统，通常在根目录下。

为了正确合理地设置内核编译配置选项，使制定编译的内核运行更快并且系统能够拥有更多的内存，因此需对内核进行剪裁，只编译系统需要的功能代码。配置完成后保存配置退出，执行内核编译：

```
make dep
make zImage
```

其中，make dep 命令用作建立内核源码的依存关系，make zImage 用于建立内存内核映像。编译成功后会在 linux/arch/arm/boot/ 下获得编译好的 Linux 内核映像 zImage。

在编译内核文件之前首先要修改根目录下的

Makefile 文件,在该文件中制定系统构架和交叉编译器:ARCH=arm、CROSS_COMPILE=/usr/local/arm/2.95.3/bin/arm-linux-(交叉编译器所在路径)[3-4]。Linux 内核提供多种配置工具,其中,make menuconfig 是以 curses 为基础的、终端式的配置菜单。在该配置菜单中可以选择目标系统的类型等,内核配置结束后会生成.config 文件,它保存了配置信息。

2.3 嵌入式 Linux 文件系统

Linux 文件系统的层次结构如图 4 所示[5]。其中,虚拟文件系统 VFS(Virtual Filesystem Switch)为用户程序提供一个统一的、抽象的、虚拟的文件系统界面,使用户程序可以通过同一个文件系统操作界面及对各种不同的文件系统进行操作,这样 Linux 可支持多种不同的文件系统。不同的文件系统通过不同的程序来实现其各种功能,但是与 VFS 之间的界面则是有明确定义的。这个界面就是 file_operation 数据结构。每种文件系统都有自己的 file_operation[6-7]。

本文件系统使用的根文件系统是 Ext2。Ext2 是 Linux 中使用最多的文件系统,因为它是专门为 Linux 设计的,拥有最快的速度和最小的 CPU 占有率。

2.4 系统运行

2.4.1 minicom 的建立

本系统的 PC 机通过 Linux 下的 minicom 与嵌入式系统通信,它为两者间通信提供操作的界面。minicom 相当于 Windows 下的超级终端,可通过串口与目标板通信。minicom 的设置为:波特率为 115 200 b/s,数据位为 8 位,无奇偶校验,停止位为 1,无数据流控制。

2.4.2 启动 U-Boot

U-Boot 烧写进 Flash 后,当系统启动时,系统会自动从 Flash 启动,运行 U-Boot。

2.4.3 下载 Linux 操作系统

在下载操作系统之前,首先用网线(交叉)直接把目标板和 PC 机相连,设置 PC 机的 IP 地址为 192.168.0.2,并调试好 tftp 功能。然后在 U-Boot 的命令行键入如下命令[8]:

```
Uboot>protect off all                    ;把 Flash 写保护去掉
Uboot>setenv ethaddr 12:34:56:78:9a:bc   ;设置目标板的 MAC 地址
Uboot>saveenv                            ;保存参数
```

环境变量设置好后就可以使用 tftp 下载内核和文件系统:

```
Uboot>tftp 0x20008000 zImage             ;下载 Linux 内核
Uboot>tftp 0x21000000 ramdisk.gz         ;下载 Linux 文件系统
Uboot<go 0x20008000                      ;开始运行
```

2.4.4 运行应用程序

(1) 当应用程序在调试阶段时可以通过 tfp 方式来运行程序。目标板上的 Linux 操作系统正常启动后首先要设置好目标板的 IP 地址为 192.168.0.2,其命令如下:

```
$ ifconfig eth 192.168.0.2
```

然后在 minicom 的命令行下键入如下命令登录 tfp 服务器(设 tfp 服务器的 IP 地址为 192.168.0.1):

```
$ tfp 192.168.0.1
```

键入该命令后输入 Linux 服务器的用户名,并在 Password 后面输入密码后回车,这时可以使用 ls、cd 等命令显示服务器上的文件并进入相应的目录。最后使用 get 命令就可以把所需要的程序下载到目标板上。

(2) 如果应用程序调试完毕,就要把编辑好的程序放到目标板的文件系统中。重新创建文件系统的方法如下:

```
$ gunzip ramdisk.gz                      ;解压缩原有的文件系统
$ mount  o loop ramdisk /mnt/newramdisk  ;解压后的文件系统映像挂在到指定的目录上
$ cd/mnt/newramdisk                      ;进入 /newramdisk 目录进行操作,随意增减文件
```

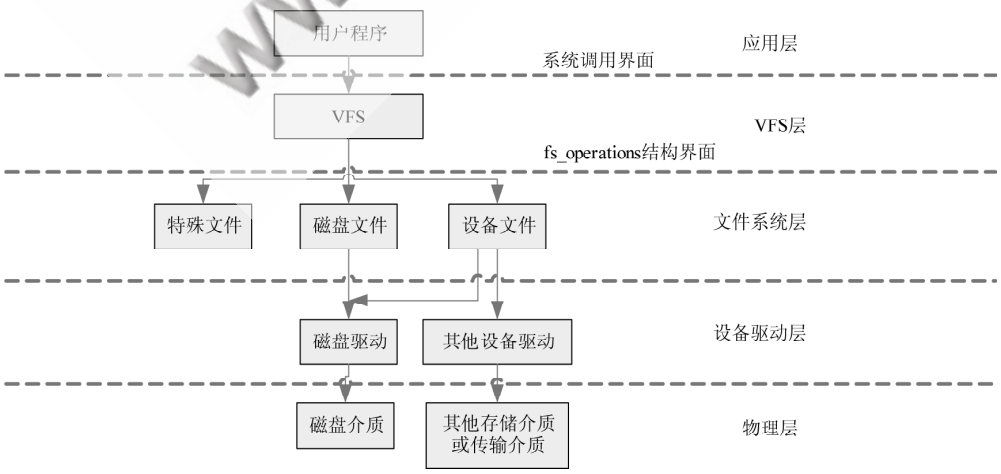

图 4 Linux 文件系统的层次结构

添加好自己的应用程序后修改 \etc\rc.d 下的 rc 文件，它是系统启动后自动调用的应用程序。

以上步骤完成后卸载文件系统映像，并重新压缩：

$ umount /mnt/newramdisk

$ gzip c v9 ramdisk>./newramdisk

3 嵌入式 Linux 驱动程序设计

本文主要完成了 EPA 设备驱动程序的模拟量输入及开关量输入输出的驱动程序。

3.1 模拟量输入驱动程序设计

EPA 设备具有多路模拟量输入测量通道，输入信号经过适当的处理，进入到 24 位 A/D 转换器 CS5522，转换后的数据会通过 SPI 接口输出给处理器。该功能的驱动程序的主要任务是合理地配置 AT91RM 9200 的 SPI 接口，使其能够正确地与 A/D 转换器 CS5522 进行数据通信。

AT91RM 9200 提供了很多寄存器用于控制 SPI 的操作，因此，在编写驱动程序之前要先定义与 SPI 操作有关的变量。定义指向 SPI 寄存器的结构体的代码如下：

AT91PS_SPI controller=(AT91PS_SPI)AT91C_VA-BASE_SPI

其中，AT91PS_SPI 在 \linux\include\asm-arm\arch-at91rm9200\AT91RM9200_SPI.h 中定义，用于指向 SPI 操作的所有寄存器，如配置寄存器、模式寄存器等：

```
Typedef struct_AT91S_SPI{
AT91_REG SPI_CR; // SPI 控制寄存器
AT91_REG SPI_MR; // SPI 模式寄存器
AT91_REG SPI_RDR; //SPI 接收数据寄存器
AT91_REG SPI_TDR; //SPI 发送数据寄存器
AT91_REG SPI_SR; // SPI 状态寄存器
AT91_REG SPI_IER; // SPI 中断使能寄存器
AT91_REG SPI_IDR; //SPI 中断禁用寄存器
AT91_REG SPI_IMR; //SPI 屏蔽寄存器
AT91_REG Reserved0[9];
AT91_REG S PI_CSR0; //SPI 片选寄存器 0
AT91_REG SPI_CSR1; //SPI 片选寄存器 1
AT91_ REG SPI_CSR2; //SP I 片选寄存器 2
AT91_ REG SP_CSR3; //SPI 片选寄存器 3
AT91_REG Reservedl[10];
}AT91S_SP I,*AT91PS_SPI
```

AT91C_VA_BASE_SPI 是在 \linux\include\asm-arm\arch-at91rm9200\hardware.h 中定义的 SPI 寄存器的地址：

#define AT91C_VA_BASE_SPI AT91_IO_P2V (AT91C_BASE_SPI)

本驱动程序需要 SPI 接口的操作是打开 / 关闭 SPI 设备，接收 / 发送数据，因此定义 file operations 结构体如下：

```
int spi_open(struct inode* inode,struct file*filp);
int spi_release(struct inode*inode,struct file*filp);
ssize_t spi_read(struct file*filp,ch ar*buf,size_t count,lofft *1);
ssize_t spi_write(struct file*filp,const char* buf,size_t count, loff_t * 1) ;
struct file_operations spi_fops={
open : spi_open,
release: spi_release,
read: spi_read,
write: spi_write,
};
```

spi_open 用于打开 SPI 设备，这个函数主要完成了 SPI 设备的初始化。AT91RM9200 的每条 I/O 线都可以复用 2 个外设功能，可用于 SPI 接口的口线是 PA0 ~ PA3，因此要把这 3 个口线设置为 SPI 控制，实现的函数是 AT91_Cf9PIO_SPI( )，该函数定义在 linux/include/asm-ann/arch-at91rm 920io.h，其原形如下：

```
static inline void AT91_CfgPIO_SPI(void){
AT91_SYS->PIOA_PDR=AT91C_PAO_MISO| AT91C_PAl_MOSI | AT91C_P A2_SPCK;
}
```

SPI 口线设置好后就要设置 SPI 的工作方式，主要设置控制寄存器(SPI_CR)、模式寄存器(SPI_MR)和片选寄存器(SPI_CSR0)：

```
controller->SPI_CR=AT91C_SPI_SWRST; // 软件复位 SPI
controller->SPI_MR=AT91C_SPI_MSTR| AT91C_SPI_MODFDIS | AT91C_SPI_PCS;
                                    // 设置 SPI 模式寄存器为主机模式，固
                                                        //定片选
controller->SPI_CSR0=(AT91C_SPI_NCPHA | (AT91C_SPI_DLYBS&0x100000) | (SPI_CLOCK <<8) | CSR0_BITS_AD);
                                    // 设置 SPI 片选寄存器包括时钟极性、
                                    // 相位、时钟波特率、传输数据位数
Controller->SPI_CR= AT91C_SPI_SPIEN;
                                    // 设置 SPI 控制寄存器，使能 SPI/
                                    // 发送与接收数据
```

当用户程序想用 SPI 接收或发送数据时，系统会调用 spi_write 和 spi_read 函数完成读写操作。AT91RM 9200 提供了接收数据寄存器(SPI_RDR)、发送数据寄存器(SPI_TDR)和状态寄存器(SPI_SR)。其中，SPI_RDR 和 SPI_TDR 分别存放接收到的数据和发送数据。SPI_SR 用于表示 SPI 当前操作的状态，其中 RDRF 标识接收数据

寄存器满。当RDRF=0时，表示上次读SPI_RDR后未收到数据；RDRF=1时，表示上次读SPI_RDR后已收到数据并由串行器发送到SPI_RDR:TDRE标识发送数据寄存器空。当TDRE=0时，表示数据已写入SPI_TDR但仍未传输到串行器；当TDRE=1时，表示最后写入发送数据寄存器的数据已传输到串行器。驱动程序通过检测RDRF和TDRE的值来完成接收和发送数据的操作。读写操作主要的实现代码如下：

```
while( !(controller->SPI_SR&AT91C_SPI_RDRF));
ret_ad = ( controller -> SPI_RDR); // 存储接收到的数据
adp =( char *)& ret_ad;
copy_ to_user( buf,adp, count);          // 把接收到的数据
                                         // 传给用户程序
copy _from_user(bf,buf,count);           // 接收用户需要发
                                         // 送的数据
while(!(controller->SPI_SR&AT91C_SPI_TDRE));
contro ller -> SPI_TDR = (ad_cmd&0xFFFF); // 向SPI设
                                         // 备发送数据
```

3.2 开关量输出驱动程序设计

EPA设备具有8路开关量输出，使用的8个引脚为PA24、P A26、P A27、PB0、PB1、PB2、PB3、PC15。在本驱动程序中用到的输入输出(PIO)控制器的用户接口有:PIO使能寄存器(PIO_PER)、输出使能寄存器(PIO_OER)、PIO置位输出数据寄存器(PIO_SODR)、清零输出数据寄存器(PIO_CODR)。以下为控制PA24引脚的输出的部分代码：

```
AT9I_ SYS->PIOA_PER=(unsignedint)(1<<24); // 使能
                                         //PIO来控制PA24
AT91_ SYS->PIOA_OER=(unsignedin t)(1<<24); // 定义
                                         //PA24为输出引脚
if(ad_cmd&0x80)
AT91_ SYS-> PI OA _SODR=(unsignedin t)(1<<24); // 使
                                         //PA24输出为高电平
else
AT 91_ SY S->PIOA CODR=(unsignedin t)(1<<24); // 使
                                         //PA24输出为低电平
```

3.3 开关量输入驱动程序设计

EPA设备具有8路数字量输入，其中有2路输入要求可以产生中断，外设通过这2路向CPU提出数据输入请求。本系统所使用作为输入的引脚有PA23、 PA25、PC0、PC10、PC11、PA19、PA20、PA22，其中PA23、PA25作为外部中断输入。当引脚被设为输入时，每个I/O线的电平都可以通过外设数据状态寄存器PIO_PDSR读出，而此时必须将PIO控制器的时钟使能。AT91RM9200的电源控制器(PMC)独立提供和控制多达30个外设时钟，其用户接口为外设时钟使能寄存器PMC_PCER。AT91RM9200的每个外设都有自己的外设标识，PMC_CER就是通过外设标识来使能其时钟的。在include\asm-arm\arch-at91rm9200\AT91RM9200.h文件中定义了各个外设的标识，部分定义如下：

```
#define AT91C_ID_PIOA     (2) //Parallel IO Controller A
#define AT91C_ID_P IOB    (3) // P arallelIO C ontroller B
#define AT91C_ID_PIOC     (4 ) // P arallelIO C ontroller C
#define AT91C_ID_IRQ0     (25) // A dvanced Interrupt
                                //Controller(IRQ0)
#define AT91C_ID_IRQ1     (26) // A dvanced Interrupt
                                //Controller(IRQ1)
#define AT91C_ID_IRQ2     (27) // A dvanced Interrupt
                                //Controller(IRQ2)
#define AT91C_ID_IRQ3     (28) // A dvanced Interrupt
                                //Controller(IRQ3)
以下为P A19作为输入引脚的主要代码:
AT91_SYS->PIOA _PER=(unsignedin t)(1<<19); // 使能
                                //PIO控制PA19
AT91_SYS->PMC_PCER=((unsignedin t)1<<AT91C
_ID_PIOA);                      // 使能PIO控制器时钟
adcmd =(AT91_SYS->PIOA_PDSR);// 通过引脚数据状
                                // 态寄存器来读引脚的输出状态
```

PA 23、PA25除了作为输入引脚外还要产生中断，AT91RM9200使用高级中断控制器(AIC)来对中断进行管理。本驱动程序用到的AIC寄存器有:AIC源模式寄存器(AIC SMR0-AIC_SMR31)，用来设置各个中断源的优先级和中断源类型;中断使能命令寄存器(AIC_IECR)，根据每个设备的标识号控制其使能。

```
AT91_ SYS->AIC_SMR[27]=(unsigned int)(1<<5); // 设置
                                //IRQ2为边沿触发
AT91_ SYS->AIC_IECR=((unsigned int)1<<AT91C_ID
_IRQ2);                         // 使能IRQ2
result= request_irq(EXTERNAL_IRQ2,&testirq_interrupt2,
NULL,"testiol",NULL);
// 注册中断并激活中断处理程序，返回值为0时表
// 示注册成功
if( result==-1)
{
printk ("Can 't get assigned irq %d\n",EXTERNAL_IRQl);
return result ;
}
void test irq_interruptl(void)  // 中断处理程序
{
printk ("INTERRUPT1 \n");
```

```
AT91_SYS ->A IC_IECR=((unsignedin t)1
<<AT91C_ID_IRQI);
                //中断处理程序结束后再次使能该中断
    }
```

以上研究了Linux操作系统设备管理方法和Linux设备中断机制，通过分析Linux设备驱动程序中断机制实现的方法及字符设备驱动程序的开发过程，编写了EPA设备的驱动程序。

本文基于ARM嵌入式开发技术研制了EPA设备的核心控制部分，完成了从硬件平台的设计、调试到Linux操作系统的BootLoader与内核的移植，文件系统和设备驱动程序分析以及针对EPA设备驱动程序的编写。

参考文献

[1] 马学文，朱名日，程小辉.基于μcLinux和53C4510B的网络通信设计.单片机与嵌入式系统应用，2004,4(6):30.

[2] 王亚军，刘金刚.Linux运用于嵌入式系统的技术分析.计算机应用研究，2005,20(5):102-105.

[3] SCHACH S R ,JIN B ,WRIGHT D R. Maintainability of the Linux kernel.Software Engineering,2002,149(1):18-23.

[4] 魏平，夏良正，王岩.Linux体系结构及嵌入式Linux的移植方法.东南大学学报，2004,34(1):126-131.

[5] 毛德操，胡希明.Linux内核源代码情景分析.杭州:浙江大学出版社，2003.

[6] 胡宁，张德运，王福豹.基于Linux的流媒体文件系统.计算机工程，2005,31(14):196-198.

[7] 史芳丽，周亚莉.Li nux系统中虚拟文件系统内核机制研究.陕西师范大学学报(自然科学版)，2005,33(1):29.

[8] 刘斐，王文君，杨建民.U-Boot在ARM系统中的启动及应用.陕西师范大学学报(自然科学版)，2005,33(6):213-215.

[9] 兰晓红.嵌入式Linux中断设备驱动程序设计.计算机应用研究，2003,23(5):96-98.

[10]ALESSANDRO R,JONATHAN C.Linux device driver(2nd edition). USA: O'Reilly, 2001.

(收稿日期：2008-12-11)

(上接第26页)

以上是以C8051F单片机作为主MCU在自动机器人控制电路中的应用，它通过PID算法和光电检测器构成双闭环的控制电路，实现了对驱动电机的精确控制，能够很好地满足比赛的控制要求。通过比赛实践证明了该电路具有体积小、功耗低、控制精度高、成本低的优点，非常适合大学生参赛使用，同时也是一个很好的训练和培养学生实践创新能力的试验项目。

参考文献

[1] 张迎新，雷文，姚静波.C8051F系列SOC单片机原理及应用.北京：国防工业出版社，2005：3-4.

[2] 新华龙电子有限公司.C8051F310/1/2/3/4/5/8/16 KB ISP Flash微控制器数据手册.Rev 1.5.潘琢金译. 2004.

(收稿日期：2008-11-20)

IT动态

Lighthouse（兆光科技）
高清晰LED屏幕为武汉天河机场带来极致视觉体验

2009年1月20日，中国武汉——全球领先的LED显示解决方案供应商Lighthouse（兆光科技）宣布,该公司为武汉天河机场提供的2台6 mm高分辨率LED显示屏已安装完毕。此项目的成功进一步巩固了Lighthouse（兆光科技）在中国LED大屏幕显示市场的领先地位。

武汉天河机场选用的是Lighthouse（兆光科技）P6-S显示面板。该产品像素间距仅为6 mm，拥有出色的显示亮度，即使在强光照射下也能实现超高的可视性和清晰的图像显示。该显示屏的亮度高达2000尼特，对比度为1000:1。M4颜色均匀性控制功能可以确保图像清晰、色彩逼真。此外，P6-S的新型超轻薄设计使安装更加简易。

“Lighthouse（兆光科技）已在中国乃至全世界许多地区完成许多LED显示屏项目，我们对此感到无比自豪”。Lighthouse（兆光科技）销售运营部门总监林泽志表示：“我们期待继续获得像武汉天河机场这样的项目，进而推动中国数码标牌市场的快速增长。在公共交通、商场、广告牌、办公室外墙等应用领域，Lighthouse（兆光科技）的产品与方案均可以发挥重要作用”。

目前，Lighthouse（兆光科技）在中国的交通、体育、创意活动、商场、办公楼宇等众多行业发挥着越来越重要的作用。Lighthouse（兆光科技）将秉承不断创新，追求卓越的精神，继续保持其在业内的领先地位。

(万卓环球供稿)