基于触摸屏驱动的 Linux 内核输入子系统研究

华 明, 徐造林
(东南大学 计算机科学与工程学院, 江苏 南京 210096)

摘 要: Linux 是目前最为优秀的开源系统软件之一, 其完全开放的特性和优良的性能表现使其在程序开发领域广受欢迎, 而其不断完善的功能和极好的适应性更使其成为嵌入式领域的首选。根据应用的需要 Linux 推出了内核输入子系统的开发平台, 极大地方便了嵌入式领域的驱动开发。触摸屏是较常使用的外部输入设备之一, 尤其是在嵌入式领域里得到了广泛的应用。文中分析了在 Linux 2.6 内核输入子系统下触摸屏驱动程序设计, 并对比传统的驱动设计方法; 得出了基于 Linux 输入子系统的驱动程序设计优点。

关键词: Linux; 嵌入式; 输入子系统; 触摸屏; 驱动程序
中图分类号: TP311 文献标识码: A 文章编号: 1673-629X(2009)03-0005-04

Linux Kernel Input Sub-System Research Based on Touch Screen Driver

HUA Ming, XU Zao-lin
(Computer Science and Engineering College, Southeast University, Nanjing 210096, China)

Abstract: Linux is one of the most excellent open source system software, it is widely popular because of its complete open characteristic and fine performance, its continually improved functions and good flexibility make it be the preferred program in the embedded area. According with the development of application, Linux issues its kernel input sub-system, it makes most benefits for driver design in embedded area. Touch screen is one of the most normal input devices, and it is especially important in embedded area. By analyzing a touch screen driver program under Linux-2.6 input sub-system and comparing it to the normal driver program design way, show the advantages of Linux sub-system.

Key words: Linux; embedded; input sub-system; touch screen; driver program

0 引 言

输入功能是一个系统最常用也是最基本的系统功能之一, 人与系统的交互必须通过输入、输出功能来完成。常用的输入设备有鼠标、键盘、触摸屏等等, 系统开发中对输入设备的处理具有许多相同的特性, 鉴于对输入设备的处理方法有很多类似之处, Linux2.4 内核以后专门为输入设备定义了一个统一的驱动程序接口—输入子系统(Input sub-system), 由此增强了对系统输入功能的管理并方便了驱动程序的开发。

由于输入子系统的优越性, 这种设备程序接口得到了很好的应用, 并且在 Linux2.6 内核中得到普及和完善, Linux2.6 内核对输入子系统做了很大的扩充增加了对更多设备的支持(如触摸屏、键盘等)。但目前为止介绍输入子系统的相关资料却较为缺乏。

文中介绍基于 Linux 2.6 内核输入子系统的触摸屏驱动开发方法, 并通过对比传统的输入设备驱动开发方法, 得出基于 Linux 输入子系统的驱动程序设计优点。

1 Linux 内核与输入子系统

Linux 输入子系统由输入子系统核心层(Input Core)、驱动层和事件处理层(Event Handler)三部份组成。其关系如图 1 所示。

其中 Input Core 即 Input Layer 由 Linux 内核中 driver/input/input.c 及相关头文件实现, 其对下提供了设备驱动的接口, 对上提供了 Event Handler 层的编程接口。输入子系统的主要数据结构如表 1 所示。

输入子系统支持的输入事件如下:

ø EV_RST 0x00 Reset

收稿日期: 2008-06-23
基金项目: 国家"863"高技术研究发展计划(2007AA01Z141)
作者简介: 华 明(1982-), 男, 福建邵武人, 硕士研究生, 研究方向为嵌入式系统; 徐造林, 硕士, 副教授, 研究方向为嵌入式系统、计算机应用。

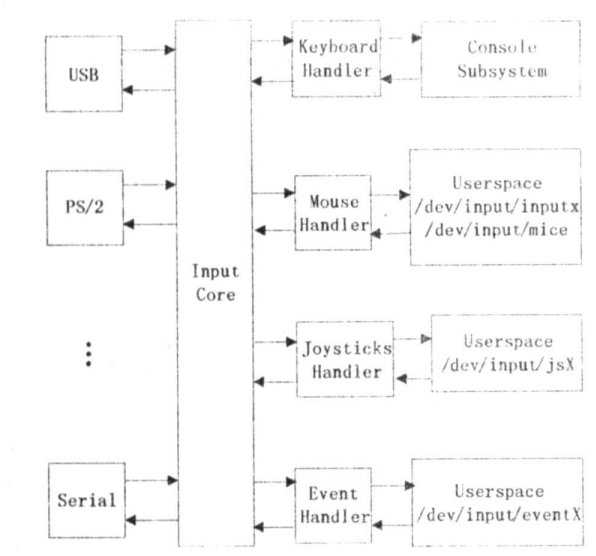

图 1 Linux 输入子系统结构图

表 1 输入子系统的主要数据结构

数据结构	用途	定义位置	具体数据结构的分配和初始化
Struct input_dev	驱动层物理 Input 设备的基本数据结构	Input.h	通常在具体的设备驱动中分配和填充具体的设备结构
Struct Evdev Struct Mousedev Struct Keybdev...	Event Handler 层逻辑 Input 设备的数据结构	Evdev.c Mousedev.c Keybdev.c	Evdev.c/Mouedev.c...中分配
Struct Input_handler	Event Handler 的结构	Input.h	Event Handler 层,定义一个具体的 Event Handler
Struct Input_handle	用来创建驱动层 Dev 和 Handler 链表的链表项结构	Input.h	Event Handler 层中分配,包含在 Evdev/Mousedev...中

- ø EV_KEY 0x01 按键
- ø EV_REL 0x02 相对坐标
- ø EV_ABS 0x03 绝对坐标
- ø EV_MSC 0x04 其它
- ø EV_LED 0x11 LED
- ø EV_SND 0x12 声音
- ø EV_REP 0x14 Repeat
- ø EV_FF 0x15 力反馈

输入子系统向应用程序和底层驱动分别提供了一个统一的接口,向输入子系统注册后的输入设备只要捕捉到输入事件并把事件的状态向子系统报告便完成了驱动程序设计的任务,对获取的事件的管理则已经由输入子系统完成,而上层应用程序也可以通过输入子系统的标准接口获取底层驱动捕获的输入事件,大大提高了驱动程序的通用性。

关于 Linux 输入子系统的更多原理及应用操作可参考文献[1,2]。

2 基于 AD7873 的触摸屏设计原理

触摸屏按其技术原理可分为五类:矢量压力传感式、电阻式、电容式、红外线式和表面声波式,其中电阻式触摸屏在嵌入式系统中用的较多。电阻式触摸屏结构简单、成本低廉、透光效果好、工作环境和外界完全隔离、不怕灰尘和水汽,可以用任何物体来触摸,可用来写字画画,同时,电阻式触摸屏具有高解析度、高速传输反应、一次校正、稳定性高、不漂移等特点,因此,电阻触摸屏应用广泛,尤其适合工业控制领域及办公室内使用。电阻式触摸屏的物理结构和触摸原理如图 2 所示。

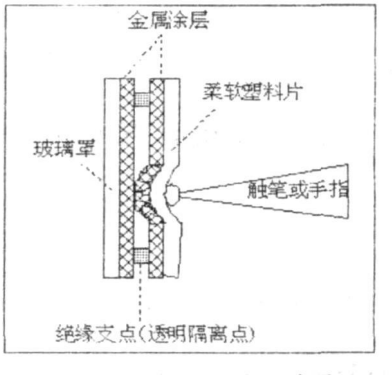

图 2 电阻式触摸屏工作原理

通过触摸笔的按压,可以导通触摸屏电路。电阻式触摸屏工作的实质是对 X Y 两个方向电阻分压的测量。只需要利用模/数转换(A/D)和电子开关来实现这个测量过程。利用普通的 A/D 转换器件或者集成在嵌入式处理器片上的 A/D 实现对触摸屏的测量,建议使用触摸屏专用的控制芯片,如 AD7873,ADS7843 和 UCB1400 等进行连接,这样可以提高触摸控制的精确性和编程的规范。笔者所实践的项目采用了 AD7873 芯片。

AD7873 电阻触摸屏数字转换器专门针对电池供电的设备进行了优化,适用于触摸屏设备,如 PDA、手持设备、监视器、POS 终端和传呼机等。该数字转换器在一个 12 位逐次逼近式比较寄存器(SAR)ADC 架构上集成了用于驱动触摸屏的低通阻抗开关。AD7873 还添加其他功能,包括一个测量温度范围为- 40℃到 85℃的片上温度传感器,直接电池和触摸压力测量计,以及一个 2.5V 的板上基准电压。这些器件在不使用内部基准电压,并以大于 125kSa/s 的吞吐率运行时的最大功耗为< 1.4mW。它们还带有 10keV 到 12keV 的模拟输入 ESD 保护,增强了抗 ESD 能力,以避免关键的内部系统元件损坏。两种低功耗数字转换器均使用单 2.2V 到 5.25V 的电源工作。为了延长电池寿命,AD7873 的内部基准电压在不使用时可以关闭,且

两种器件均带有节电模式, 可将消耗电流降至< 1μA。详细的 AD7873 芯片原理及控制方式可参考相关的 datasheet 文档。

笔者的实践平台采用的是 Freescale 公司的 i. MX27 微处理器, 其上集成了可配置的同步串行接口(CSPI) 可用于与 AD7873 相连, CSPI 主要通过 3 线 MOSI、MISO、SCLK 控制处理器与外部的通信。同时还需要另选一个 GPIO 口作为触摸中断的输入, 实践中使用了 i. MX27 通用输入输出口第 3 组的第 29 口 PD28 作为笔中断, 在驱动程序中对此中断口做适当的配置并使用 request_ irq( ) 函数进行中断注册, 当触摸屏被按下时, 驱动程序可以捕捉到中断事件, 对输入事件进行处理通过 AD7873 便可及时获取相关的输入数据。

关于CSPI 与触摸屏控制芯片(如AD7873) 及触摸屏的连接原理及控制方式在大量的触摸屏驱动设计中都有介绍, 而这点在基于 Linux 输入子系统的驱动设计和常规的驱动设计中并没有任何区别, 这里不做详细介绍, 相关原理可参考文献[3] 。

3　结合 input 输入子系统的触摸屏驱动设计

在 Linux 内核中, 输入设备用 input_ dev 结构体描述, 使用输入子系统实现输入设备驱动的时候, 驱动的核心工作是向系统报告按键、触摸屏、键盘、鼠标等输入事件(event, 通过 input_ event 结构体描述) , 不再需要关心文件操作接口, 因为输入子系统已经完成了文件操作接口。驱动报告的事件经过 InputCore 和 Eventhandler 最终到达用户空间。

基于 Linux 输入子系统的触摸屏驱动主要有 3 个入口, 即驱动模块加载时调用的初始化入口函数、中断产生时调用的中断处理函数以及应用程序打开和关闭设备时调用的入口函数。将硬件电路按照正确的方式连接好后, 从 3 个入口开始通过输入子系统设计触摸屏驱动需要完成的主要工作流程如图 3 所示。

首先, 在驱动程序加载时被调用的初始化函数 static int_ init digi_ init(void) 中完成的工作包括:

1) 定义一个输入设备并进行初始化:

```
static struct input_ dev * ts_ input_ dev;
ts_ input_ dev= input_ allocate_ device( ) ; //分配空间
```

2) 告知输入子系统它将报告的数据格式。

```
ts_ input_ dev- > absbit[ 0]= BIT( ABS_ X) | BIT( ABS_ Y) | BIT( ABS_ PRESSURE) ;
//告知系统将使用绝对坐标进行计算, 需要获取的值有 X, Y 坐标以及压力值
input_ set_ abs_ params( ts_ input_ dev, ABS_ X, EDGE_ MIN_ LIMIT, EDGE_ MAX_ LIMIT, 4, 8) ;
input_ set_ abs_ params( ts_ input_ dev, ABS_ Y, EDGE_ MIN_ LIMIT, EDGE_ MAX_ LIMIT, 4, 8) ;
input_ set_ abs_ params( ts_ input_ dev, ABS_ PRESSURE, PRESS_ MIN, PRESS_ MAX, 4, 8) ;
//向输入子系统报告各个值的精度及边界, 不在此范围内的值将不被提交
```

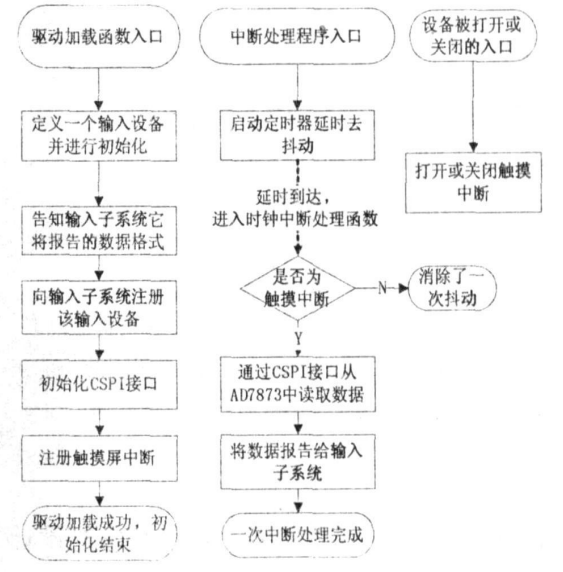

图 3　基于输入子系统的触摸屏驱动实现流程

3) 注册该输入设备。

输入设备的注册函数为: input_ register_ device (ts_ input_ dev) ;

4) 初始化 CSPI 接口。

void spi_ init(void) , 在该函数中配置 CSPI 的各个寄存器, 使之开始数据采集工作。

5) 注册触摸屏中断。

request_ irq(ts_ input_ dev- > irq_ gpio, digi_ isr, TPNL_ INTR_ MODE, MODULE_ NAME, & ts_ dev) ; //digi_ isr 是中断处理函数, 触摸中断产生后该函数将被调用。

其次, 实现触摸屏的中断处理函数。在触摸屏被触摸/ 抬起/ 移动时通过下列函数将输入事件报告给输入子系统。

```
input_ report_ abs( ts_ input_ dev, ABS_ X, 0) ; //X 坐标
input_ report_ abs( ts_ input_ dev, ABS_ Y, 0) ; //Y 坐标
input_ report_ abs( ts_ input_ dev, ABS_ PRESSURE, 0) ; //压力
input_ sync( ts_ input_ dev) ; //同步, 告知事件的接收者驱动已经发出了一个完整的报告
```

触摸屏驱动的大部分工作实际上是在中断的处理程序中完成, 程序中的中断处理函数是 static irqreturn_ t digi_ isr(int irq, void * dev_ id) 。中断的产生是由电平高低的跳变形成, 硬件中断的产生具有极高的

精度当触摸屏被按下时,由于硬件的电气及物理特性,电平的跳变在极短的时间内实际上是一个抖动的过程,而硬件完全能够捕捉到这些快速的电平跳变,并在短时间内产生多个中断,然而这却干扰了我们对输入事件的识别。因此在中断处理函数中往往采用延时的策略来跳过抖动达到正确的判断。所以笔者的程序中在触摸屏被按下所产生的中断处理程序中启用了一个定时器 add_ timer(ts_ input_ dev- > pen_ timer),进入中断后将产生 10ms 的延时,而真正的触摸屏按键参数的读取及报告是在时钟的中断处理程序 static void digi_ sam_ callback(unsigned long data) 中进行。

最后驱动向应用层提供的接口函数是:

```
ts_ input_ dev- > open = digi_ open;
ts_ input_ dev - > close = digi_ release;
```

当应用程序读取和关闭该触摸屏的设备节点时,这两个函数将被调用。digi_ open 函数完成的功能主要是初始化 CSPI 接口,使能触摸屏及开中断。而 digi_ release 函数完成的功能则刚好相反。

4 基于输入子系统的驱动设计较常规设计方法的优越性

有较多文献都介绍了常规的触摸屏驱动程序设计方法(参考文献[4~ 6]),常规的设备驱动程序设计需要处理更多的事务,对上层要实现更多的入口,通常除以上介绍的入口外还要向应用层实现设备的读取入口函数,而该入口的实现具有相当的复杂性。在基于输入子系统的驱动设计中,输入事件被采集后将报告给输入子系统来管理,程序设计者不需要关心这个部分,而常规的设备驱动程序中由于没有输入子系的支持,对采集来的输入数据需要在驱动中自行设计管理方案,通常是采用自定义的循环缓冲的方法将连续采集到的输入数据加入缓冲链表中,并向上层应用提供一个自定义的接口将缓冲中的输入事件报告给读取设备的应用程序。同时还需考虑当缓冲区中的数据为空时对读取输入数据的应用程序的响应方式,通常采用等待队列来阻塞应用程序。而缓冲区大小的设置也需要考虑诸多因素的影响。

对比采用输入子系统的触摸屏驱动程序设计方法与常规的触摸屏驱动设计方法,可以总结出以下几点:

(1)基于输入子系统的驱动程序设计只需要考虑较少的模块,使程序设计难度降低。笔者实践的基于输入子系统的触摸屏驱动设计中需要向上层实现的设备接口如下:

```
ts_ input_ dev- > open = digi_ open;
ts_ input_ dev- > close = digi_ release;
```

即设备打开和关闭时的初始化和注销工作。而常规的驱动设计中对上层有大量的接口可以自定义,触摸屏驱动中通常还需要实现的设备接口有:

```
ssize_ t (* read) (struct file * , char_ user * , size_ t, loff_ t * );
```

(2)基于输入子系统的驱动程序设计具有更高执行效率。常规的驱动程序设计需要自行设计对输入事件的管理方案,对采集到的输入事件的管理是否合理将影响到驱动程序的效率,而基于输入子系统的驱动设计把驱动的设计从对事件的管理中解脱出来,使程序设计者可以花更多的精力来致力于提高输入设备采集的效率和精度上来。

(3)基于输入子系统的驱动程序设计具有更广泛的移植性和适应性。常规的驱动程序设计需要自定义对上层的接口,而自定义的接口在应用层并不一定具有通用性,这大大限制了驱动程序的使用范围,而基于输入子系统的驱动程序设计可以不用考虑向上层报告输入设备的接口设计,将此工作交给了输入子系统来完成,而且输入子系统对上层的接口具有通用性,可以使驱动程序的使用范围大大扩展。如基于输入子系统的驱动程序可以直接驱动 Qt 的图形界面。

(4)基于输入子系统的驱动程序设计规范了输入设备驱动程序的设计,使程序的设计更为简洁规范。

5 结束语

通过对比两种触摸屏驱动程序的设计方法,可以看出基于 Linux Input 子系统的驱动程序设计无论是从程序的复杂性、通用性、高效性等各方面来衡量都具有明显的优势。可以肯定的是随着 Linux 内核的不断更新和 Linux 的使用在嵌入式领域的不断扩展输入子系统在 Linux 的内核中将得到不断的完善,将会有更多的输入事件得到 Linux 输入子系统的支持。

参考文献:

[1] Hards B. Using the Input Subsystem, Part I[EB/OL]. 2003-02. http://www.linuxjournal.com/article/6396.

[2] Hards B. Using the Input Subsystem, Part II[EB/OL]. 2003 - 02. http://www.linuxjournal.com/article/6429.

[3] 李春萍,李颉思. 嵌入式 Linux 中对触摸屏驱动的设计[J]. 计算机工程与设计, 2007, 28(6): 1387- 1389.

[4] 强新建,田 泽,刘天时. 基于 S3C2440 的触摸屏驱动程序实现[J]. 航空计算技术, 2007, 37(4): 85- 87.

[5] 畅卫功,丁忠林. 嵌入式 Linux 系统中触摸屏驱动的研究[J]. 微计算机信息, 2007, 23(1- 2): 103- 105.

[6] 刘 淼. 嵌入式系统接口设计与 Linux 驱动程序开发[M]. 北京:北京航空航天大学出版社, 2006: 83- 101.