第 8 卷 第 4 期
2019 年 7 月
集 成 技 术
JOURNAL OF INTEGRATION TECHNOLOGY
Vol. 8 No. 4
Jul. 2019

**引文格式：**
李红义, 孔庆德, 方伟, 等. 基于 Android 模拟 P-Sensor 的设计与实现 [J]. 集成技术, 2019, 8(4): 63-68.
Li HY, Kong QD, Fang W, et al. Design and implementation of virtual P-Sensor based on android [J]. Journal of Integration Technology, 2019, 8(4): 63-68.

# 基于 Android 模拟 P-Sensor 的设计与实现

李红义[1,2] 孔庆德[1,2] 方 伟[3] 吴双卿[3] 胡 超[3]

[1](TCL 移动通信科技(宁波)有限公司 宁波 315100)
[2](TCL 通讯宁波研究院 宁波 315100)
[3](浙江大学宁波理工学院 宁波 315100)

**摘 要** 该文基于高通 MSM8939 平台，在不增加距离传感器(Proximity Sensor，P-Sensor)的前提下，提出了一种基于触摸屏的距离传感器模拟方法。该文首先介绍了电容触摸屏面板触点位置检测原理和数值计算方法，并给出模拟 P-Sensor 距离检测原理；然后，基于触摸屏模组和基带芯片的接口电路以及电源管理芯片，设计并实现了模拟 P-Sensor 的硬件电路；最后，采用 UML(统一建模语言)序列图的设计方式，实现了基于服务端-客户端框架层中的模拟 P-Sensor 通道控制程序设计，并根据距离检测结果实现了屏幕的亮灭自动控制以达到节省功耗的目的。

**关键词** Android；模拟 P-Sensor；Linux 内核驱动；框架层；UML 序列图
**中图分类号** TP 249 **文献标志码** A doi: 10.12146/j.issn.2095-3135.20190515001

## Design and Implementation of Virtual P-Sensor Based on Android

LI Hongyi[1,2] KONG Qingde[1,2] FANG Wei[3] WU Shuangqing[3] HU Chao[3]

[1]( *TCL CommunicationTechnology* (*Ningbo*) *Ltd*., *Ningbo* 315100, *China* )
[2]( *TCL Communication Research Institute*, *Ningbo* 315100, *China* )
[3]( *Ningbo Institute of Technology*, *Zhejiang University*, *Ningbo* 315100, *China* )

**Abstract** This paper proposed a solution to realize the virtual proximity sensor (P-Sensor) based on usual touch panel. The principle of the distance detection based on capacitance touch panel is introduced firstly. And the hardware system is developed according to the baseband and power management units. Using the UML (unified modeling language) sequence diagram, the control channel framework for the virtual P-Sensor is realized based on server-client system. With the distance detection results, the screen can be turn on or off automatically to save power consumption.

**Keywords** Android; virtual P-Sensor; Linux driver; framework; UML sequence diagram

**收稿日期**：2019-05-15 **修回日期**：2019-06-16
**基金项目**：浙江省自然科学基金项目(LQ17F030002)；宁波市自然科学基金项目(2017A610108)
**作者简介**：李红义，硕士，研究方向为嵌入式系统开发；孔庆德，硕士，研究方向为电路与硬件系统设计；方伟，讲师，研究方向为传感器与智能检测；吴双卿(通讯作者)，副教授，研究方向为机器视觉、光电检测与信息处理，E-mail：wsqing1999@163.com；胡超，教授，研究方向为自动化与传感器技术、图像处理和机器视觉。

## 1 引 言

Android 是一个基于 Apache License 的开放系统框架，它吸取了 Linux 开源社区的成果，并进行了一系列针对移动设备的优化[1-3]。目前常用的操作系统有 Android、黑莓和苹果的 IOS 等，而在国内生产的大部分智能终端(如手机、平板电脑(Pad)等)都是基于 Android 系统的，因此具有广泛的应用范围。为了发挥智能终端的多媒体特性及提升各项性能，各种传感器成为了智能终端不可或缺的组件之一[4-5]。比如，在通话过程中终端靠近人脸时，由于不进行屏幕操作，可以自动熄灭屏幕以节省功率消耗；当远离人脸(如拿在手上)时，一般会进行屏幕操作，则需点亮屏幕提升用户体验。通常会在顶部(屏幕上方)放置一颗距离传感器(Proximity Sensor，P-Sensor)实现这个功能，但在倒置拿时，距离传感器会变成在智能终端的底部，这时无法进行距离检测和判断，因此需要再增加一颗距离传感器或探讨其他检测手段。针对智能终端正置和倒置都可以正常通话，并根据检测距离自动控制屏幕以节省功耗的功能需求，本文基于高通 MSM8939 平台，在不额外增加硬件距离传感器的基础上，提出使用触摸屏(Touch Panel，TP)来实现一个模拟距离传感器(模拟 P-Sensor)。通过对模拟 P-Sensor 距离检测原理进行分析，设计实现了模拟 P-Sensor 的硬件电路，同时使用 UML(统一建模语言)序列图的方式设计了模拟 P-Sensor 的控制通道程序。

## 2 电容式触摸屏距离感应原理

电容式触摸屏在玻璃表面上涂有一层特殊的透明导电材料，其工作原理是基于人体所产生的感应电流——当导体接触面板时落点处电容会随之改变，进而可以计算出触点位置[6-7]。电容式触摸屏主要有表面电容式和投射电容式[8]。其中，投射电容式可支持多点触摸和提供较高的精度。如图 1 所示，投射电容式触摸屏幕以矩阵方式排列导电材质，其中由行($X_1$-$X_n$)和列($Y_1$-$Y_n$)组成矩阵。当手指等导体接触屏幕时，扫描 $X$、$Y$ 轴检测接触电容的变化，即可计算出触摸位置。

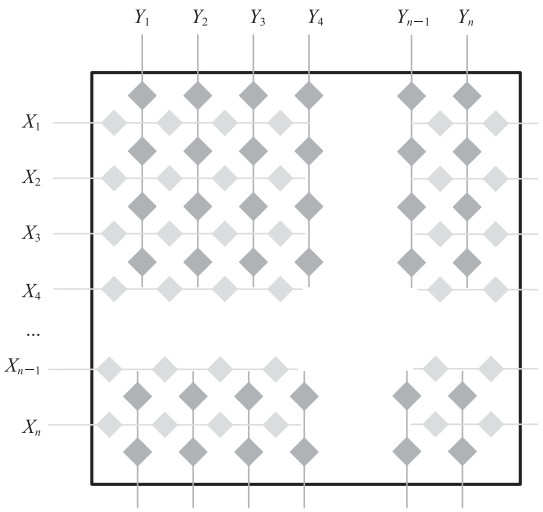


**图 1 触摸屏的电容矩阵**

**Fig. 1 Capacitance matrix of touch panel**

对于触摸屏来说，首先通过检测获取手指接触后自容或互容所产生的变化，然后通过触摸屏控制芯片内部的微控制单元(MCU)和数字信号处理器(DSP)程序来计算并确定位置。而实际对电容屏来说，除了接触外，一定距离的靠近也会影响电容屏幕的自容和互容，而通过触摸屏来模拟 P-Sensor 正是通过这一原理来实现的。

## 3 模拟 P-Sensor 硬件电路设计

模拟 P-Sensor 和基带芯片的硬件电路接口设计如图 2 所示，其中 Host 为主控 CPU 亦即基带芯片，而 P-Sensor 在这里实际上是采用触摸屏 TP 来模拟的，所以左侧是一个触摸屏 TP 模组来模拟实现 P-Sensor 的功能。基带芯片通过串口($I^2C$)和模组进行通讯[9]，同时也使用了部分的 GPIO(通用型之输入输出)引脚。从图 2 可以看

出，模组内有一个基于 MCU 的控制器，基带芯片本质上是和这个控制器来配合工作的，而控制器再去控制真实的触摸屏 TP。

在图 3 所示的硬件电路设计中，MSM8939 为基带控制芯片，通过内部的 GPIO 复用功能来实现。PM8916 为电源管理芯片，提供诸多模拟及电源相关功能。SDA/SCL 为模组和基带芯片之间传递控制信息和上报数据所使用的 $I^2C$ 串口通道。INT 为模组中断信号，当有数据需要上报时，这个引脚便被拉低以便触发基带芯片发生中断，然后上报数据到主控 CPU（即 MSM8939 基带控制芯片）。RESET 在模组初始化时使用，主控 CPU 通过输出低电平来复位此模组。

另外，这个电路由于是协同射频信号一起工作的，SDA/SCL $I^2C$ 信号工作时所产生的最高 4 MHz 多的频率会对射频电路产生一定的影响，所以需在 SDA/SCL 线上增加一个 *LC* 滤波电路，其中取电感 $L$＝27 nH、电容 $C$＝22 pF。INT 信号上所串的 *LC* 滤波电路也是同一个道理。*LC* 的取值与其本身的谐振频率和射频电路的工作频率

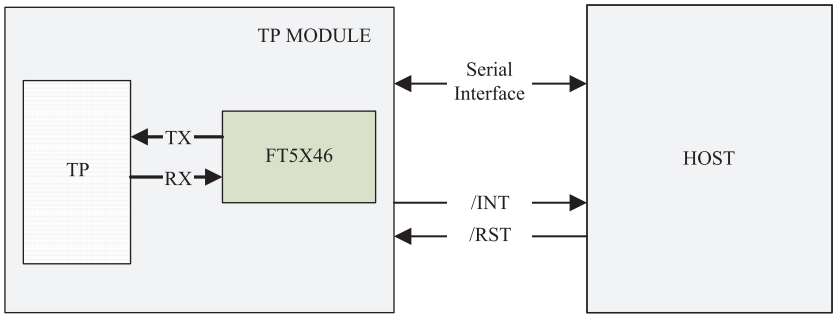


**图 2　P-Sensor 硬件接口**

**Fig. 2　P-Sensor hardware interface**

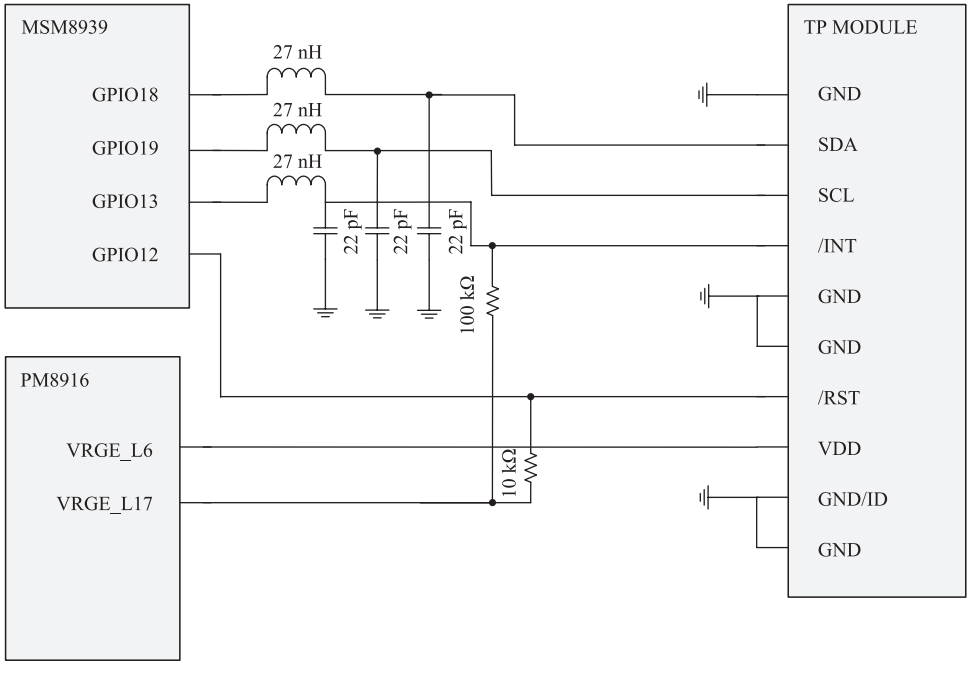


**图 3　P-Sensor 硬件电路设计原理**

**Fig. 3　The hardware schematic of the P-Sensor**

有关。由于 INT、RESET 信号是低有效使能，因此这里进行了上拉，使其正常工作时去使能。值得说明的是，除连接 VREG_L17（$V$=1.8 V）外，还串联了个 10 kΩ 的电阻。

## 4 模拟 P-Sensor 的软件系统设计

Android 平台 Sensor 的架构分为具体硬件、Linux 内核驱动、硬件抽象层、本地框架（C/C++）、Java 层框架和应用层。通常 Sensor 的开发会涉及 Linux 内核设备驱动程序的编写和硬件抽象层，而框架层则不做修改。但本文是通过触摸屏 TP 来模拟 P-Sensor 的功能，故还需要对框架层做一定的开发设计来实现控制通路。

### 4.1 P-Sensor 驱动程序设计

模拟 P-Sensor 的数据传输设计使用输入设备节点/dev/input/inputx 通知用户空间程序。另外，还需要对模拟 P-Sensor 功能打开与关闭进行控制，如仅当智能终端处于倒置状态时，才打开模拟 P-Sensor 功能。Linux 内核提供了 sysfs 虚拟文件系统（挂载点为/sys），把内核空间的设备驱动信息传到用户空间。模拟 P-Sensor 的控制通道采用这一技术来实现[10-12]。

```
static DEVICE_ATTR(enable, SYSFS_
AUTHORITY, virtual_proximity_enable_
show, virtual_proximity_enable_store);
static intsys_create_file(void)
{
 struct class *virtual_proximity = NULL;
 struct device *virtual_proximity_device =
NULL;
 virtual_proximity = class_create(THIS_
MODULE, "virtual-proximity");
 virtual_proximity_device = device_
create(virtual_proximity, NULL, 0, NULL,
"device");
 device_create_file(virtual_proximity_device,
& dev_attr_enable)
}
```

### 4.2 P-Sensor 框架层的设计开发

在本文系统中，框架层被实现成客户端-服务端（C-S）的架构，同时新增一个类 SymmetricalPolicyManager 来封装具体的控制逻辑，然后在客户端新增加接口，供 JNI（Java Native Interface）调用。此外，服务端在接口（ISensorServer）、代理（BpSensorServer）和具体的实现（BnSensorServer, SensorService）上也会新增接口（SetProximitySensorStatus()），通过 SensorService 来最终调用驱动层所暴露出来的虚拟文件系统节点，最终形成本文的控制通道设计，具体的序列图如图 4 所所示。

另外，新增加的控制逻辑会被 WindowManagerService.java 中屏幕旋转逻辑所调用，来达到在真正的 P-Sensor 和模拟 P-Sensor 之间的切换。

## 5 系统测试与结果

为对系统功能进行测试，采用函数 virtual_proximity_enable_show() 返回当前模拟 P-Sensor 的状态，当用户对节点 enable 写值时调用 virtual_proximity_enable_store()，本文约定可写入的有效值为 0 和 1。其中，节点 enable 写入 0 值时表示关闭模拟 P-Sensor，通过写 0 到寄存器 0xB0 关闭功能；节点 enable 写入 1 值时表示打开模拟 P-Sensor，先在芯片的 RESET 引脚上加一低电压硬件复位芯片，然后写 1 到寄存器 0xB0 打开功能。本文通过 ls 和 cat 来测试上述设备节点是否创建成功和工作是否正常，如下终端输出所示。enable 节点已经被正常建立，

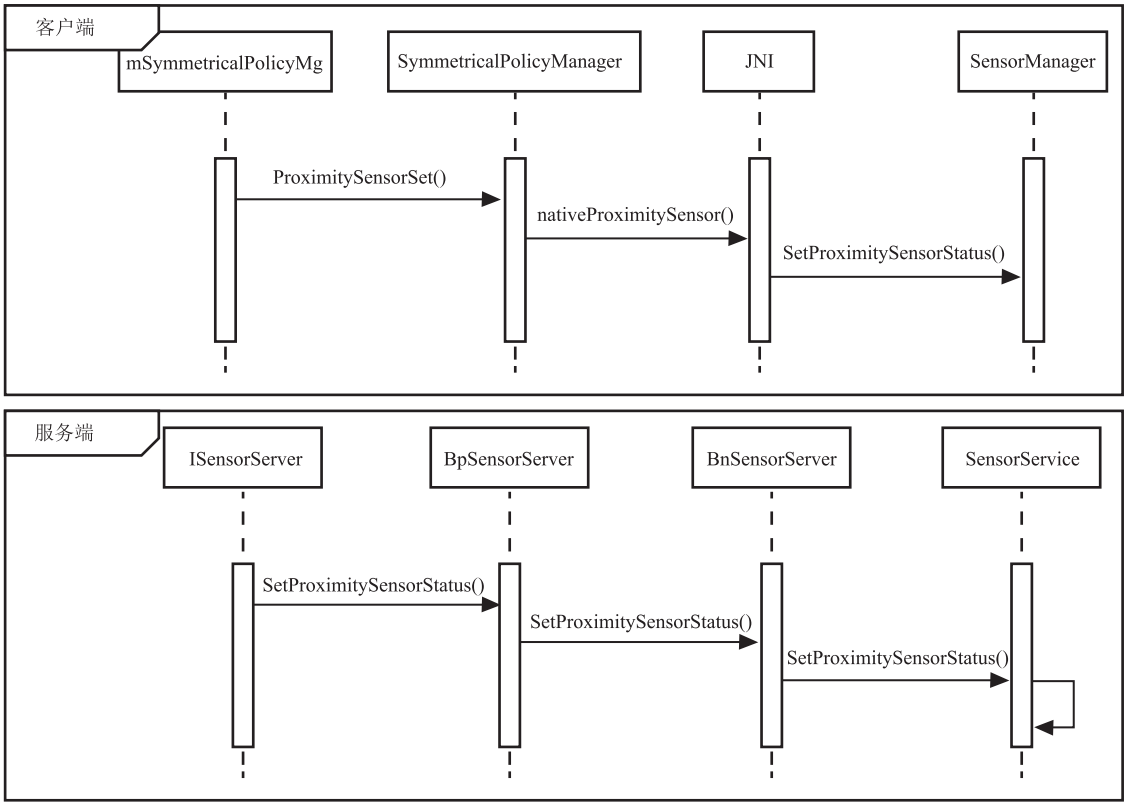


**图 4　模拟 P-Sensor 序列图**

**Fig. 4　UML sequence diagram**

并且当前状态是 0，说明模拟 P-Sensor 处于关闭状态。

```
/sys/class/virtual-proximity/device # ls
enable
power
subsystem
uevent
/sys/class/virtual-proximity/device # cat enable
cat enable
0
```

本文在系统里设定设备接近和远离之间的阈值为 1.2 cm，同时由于使用过程中距离会发生小幅度的波动，在阈值附近会产生屏幕忽亮忽暗的情况，需要做滞回区间±0.3 cm。检查设备节点 /sys/class/virtual-proximity/device/enable 为 1 说明模拟 P-Sensor 处于正常工作状态。在整机测试过程中倒置智能终端，使用智能终端逐渐远离和逐渐接近人脸，测试结果如表 1 所示(测试过程中电容式触摸屏需要感应人脸皮肤，距离数值本身有一定的误差)。

根据测试结果分析，电容式触摸屏的感应距离比普通的 P-Sensor 要近(普通的 P-Sensor 阈值一般设成 6 cm 左右)，但本文采用触摸屏 TP 来

**表 1　触摸屏的亮/暗状态测试结果**

**Table 1　The test of the touch panel screen with varied distance**

| 远离 | | 接近 | |
|---|---|---|---|
| 距离(cm) | 屏幕状态 | 距离(cm) | 屏幕状态 |
| 1.0 | 暗 | 1.2 | 亮 |
| 1.1 | 暗 | 1.1 | 亮 |
| 1.2 | 暗 | 1.0 | 亮 |
| 1.3 | 暗 | 0.9 | 亮 |
| 1.4 | 亮 | 0.8 | 暗 |
| 1.5 | 亮 | 0.7 | 暗 |

实现的模拟 P-Sensor，能较好地根据检测距离自动控制屏幕的点亮或熄灭，在一定程度上能达到节省功耗的目的。

## 6 总结与展望

本文研究了电容触摸屏面板的触点位置检测原理和数值计算方法，给出了模拟 P-Sensor 的距离检测原理，在不增加硬件距离传感器的基础上，利用现有的电容触摸屏 TP 来实现距离检测。基于触摸屏模组和基带芯片的接口电路以及电源管理芯片，开发并实现了模拟 P-Sensor 的硬件电路，考虑协同射频信号同时工作，增加了滤波电路设计。实现了基于客户端-服务端架构的模拟 P-Sensor 控制通道程序设计，并根据距离检测结果实现屏幕的亮灭自动控制以达到节省功耗的目的。

## 参 考 文 献

[1] Meier R, Lake L. Professional Android (4th Edition) [M]. Wrox, 2018.

[2] 韩超, 梁泉. Android 系统原理及开发要点详解 [M]. 北京: 电子工业出版社, 2010.

[3] 姚昱旻, 刘卫国. Android 的架构与应用开发研究 [J]. 计算机系统应用, 2008(11): 110-112.

[4] 徐科军. 传感器与检测技术(第 4 版) [M]. 北京: 电子工业出版社, 2016.

[5] Chikaraishi T, Minato T, Ishiguro H. Development of an Android system integrated with sensor networks [C] // Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008.

[6] 薛建设, 刘翔, 鲁成祝, 译. 触摸屏技术与应用 [M]. 北京: 机械工业出版社, 2014.

[7] 秦丹丹, 夏志强. 低成本内置触控技术 [J]. 光电子技术, 2019, 39(1): 58-62.

[8] 彭本贤, 俞挺, 于峰崎. 低噪声电容式超声传感器的结构与电路设计 [J]. 集成技术, 2012, 1(3): 15-19.

[9] NXP Semiconductors. UM10204: $I^2C$-bus specification and user manual (Rev.6) [OL]. 2014-04-04[2019-05-13]. http://www.nxp.com/documents/user_manual/UM10204.pdf.

[10] Corbet J, Rubini A, Kroah-Hartman G. Linux Device Drivers (3rd Edition) [M]. O'Reilly Media, 2005.

[11] Love R. Linux Kernel Development (3rd Edition) [M]. Addison-Wesley Professional, 2010.

[12] Henry-Stocker S. How to manage your Linux environment [OL/J]. Network World, 2019. https://www.networkworld.com/article/3385516/how-to-manage-your-linux-environment.html.