Hi3510 通过 GPIO 模拟实现触摸屏控制

Application Notes

文档版本 01

发布日期 2006-09-10

BOM编码 N/A

深圳市海思半导体有限公司

深圳市海思半导体有限公司为客户提供全方位的技术支持，用户可与就近的海思办事处联系，也可直接与公司总部联系。

深圳市海思半导体有限公司

地址： 深圳市龙岗区坂田华为基地华为电气生产中心 邮编：518129

网址： http://www.hisilicon.com

客户服务电话： 0755-28788858

客户服务传真： 0755-28788838

客户服务邮箱： support@hisilicon.com.

商标声明

 、Hisilicon、海思，均为深圳市海思半导体有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

插图目录

表格目录

前 言

概述

本节介绍本文档的内容、对应的产品版本、适用的读者对象、行文表达约定、历史修订记录等。

产品版本

与本文档相对应的产品版本如下所示。

产品名称	产品版本
Hi3510 通信媒体处理器芯片（简称 Hi3510）	Hi3510 V100
	Hi3510 V101
	Hi3510 V110
Hi3510 DVS 方案	Hi3510 DMS V100R001
Hi3510 VPhone 方案	Hi3510 DMS V200R001

读者对象

本指南为程序员和系统工程师而编写，描述了基于 Hi3510 媒体解决方案平台（Digital Media Solution，简称 DMS）如何通过 GPIO 模拟 SPI 接口实现触摸屏控制器的连接，以及触摸屏基本功能的实现方法。使用本书的程序员应该：

- 相关硬件基本知识
- 熟练掌握 C 语言
- 掌握基本的 Linux 环境编程

内容简介

本文描述通过 Hi3510 的 GPIO 模拟 SPI 接口，实现与触摸屏控制器连接，完成对触摸屏按下状态、弹起状态及运动轨迹的检测。实现菜单选取、手写输入等触摸屏基本功能。

章节	内容
1　概述	简单描述文档的主要内容。
2　GPIO 模拟 SPI 接口	详细描述 GPIO 引脚与 SPI 信号线的连接方法、GPIO 引脚模拟各个信号线需要满足的时序和注意事项。
3　触摸屏模块	详细描述触摸屏基本功能的方法和驱动模块的移植过程。

约定

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
危险	以本标志开始的文本表示有高度潜在危险，如果不能避免，会导致人员死亡或严重伤害。
警告	以本标志开始的文本表示有中度或低度潜在危险，如果不能避免，可能导致人员轻微或中等伤害。
注意	以本标志开始的文本表示有潜在风险，如果忽视这些文本，可能导致设备或器件损坏、数据丢失、设备性能降低或不可预知的结果。
窍门	以本标志开始的文本能帮助您解决某个问题或节省您的时间。
说明	以本标志开始的文本是正文的附加信息，是对正文的强调和补充。

通用格式约定

格式	说明
宋体	正文采用宋体表示。
黑体	一级、二级、三级标题采用黑体。
楷体	警告、提示等内容一律用楷体，并且在内容前后增加线条与正文隔离。
“Terminal Display”格式	“Terminal Display”格式表示屏幕输出信息。此外，屏幕输出信息中夹杂的用户从终端输入的信息采用加粗字体表示。

修改记录

修订记录累积了每次文档更新的说明。最新版本的文档包含以前所有文档版本的更新内容。

修改日期	版本	修改说明
2006-9-10	01	初始版本

1 概述

通过 Hi3510 的 GPIO 模拟 ADS7846 的数字接口，实现与触摸屏控制器连接，完成对触摸屏按下状态、弹起状态及运动轨迹的检测。

第 2 章和第 3 章将从以下两个方面介绍触摸屏模块：

- GPIO 模拟 ADS7846 数字接口
- 触摸屏驱动模块

2 GPIO 模拟 ADS7846 数字接口

关于本章

本章描述内容如下表所示。

标题	内容
2.1 触摸屏控制器（ADS7846）接口	介绍触摸屏控制器接口时序图和参数。
2.2 GPIO 与 ADS7846 数字接口连接	介绍 GPIO 与 ADS7846 数字接口的连接关系。
2.3 信号模拟	介绍 GPIO 引脚模拟信号。

2.1 触摸屏控制器（ADS7846）接口

触摸屏控制器 ADS7846 是 TI 公司的触摸屏控制器，它的数字接口类似 SPI（Serial Peripheral Interface）接口。关于 ADS7846 的详细介绍，请参见 ADS7846 的 datasheet。

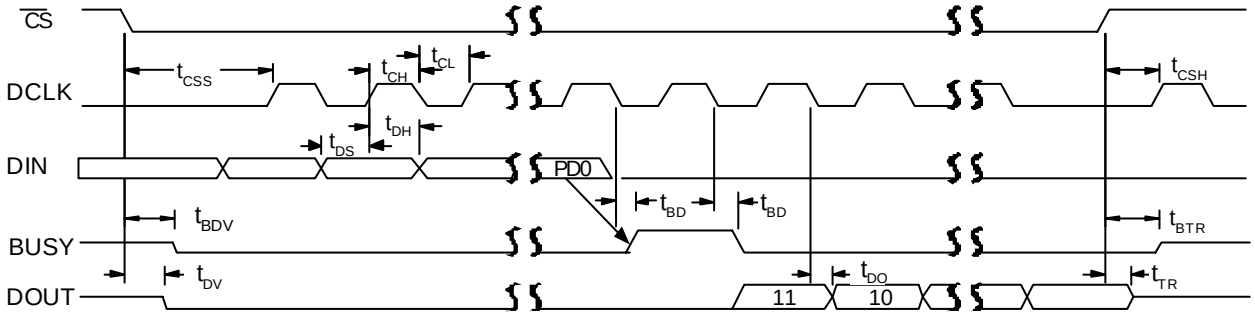
触摸屏控制器 ADS7846 数字接口信号描述如表 2-1 所示。

表2-1 触摸屏控制器 ADS7846 数字接口信号描述

信号名	方向	描述
SPICLK（DCLK）	I	串行时钟输入
SPICSN（CS）	I	片选
SPIDI（DIN）	I	串行数据输入
SPIDO（DOUT）	O	串行数据输出
SPIIRQ	O	中断引脚
SPIBUSY（BUSY）	O	状态指示

触摸屏控制器 ADS7846 数字接口的时序如图 2-1 所示。

图2-1 触摸屏控制器 ADS7846 数字接口的时序



注：此图参考 TI 公司的 ADS7846 的 datasheet。

图 2-1 中信号时序参数必须满足表 2-2。

表2-2 触摸屏控制器 ADS7846 数字接口的时序参数

符号	描述	最小时间	最大时间	单位
t_{ACQ}	捕获时间	1.5	-	μs

符号	描述	最小时间	最大时间	单位
t_{DS}	DIN 有效到 DCLK 上升沿	100	-	ns
t_{DH}	DCLK 变为高电平后，DIN 的保持时间	10	-	ns
t_{DO}	DCLK 下降沿到 DOUT 有效	-	200	ns
t_{DV}	片选下降沿到 DOUT 使能	-	200	ns
t_{TR}	片选上升沿到 DOUT 禁止	-	200	ns
t_{CSS}	片选下降沿到第一个时钟上升沿	100	-	ns
t_{CSH}	片选上升沿到忽略时钟信号	0	-	ns
t_{CH}	时钟高电平持续时间	200	-	ns
t_{CL}	时钟低电平持续时间	200	-	ns
t_{BD}	时钟下降沿到 BUSY 信号上升沿	-	200	ns
t_{BDV}	片选下降沿到 BUSY 信号使能	-	200	ns
t_{BTR}	片选上升沿到 BUSY 信号禁止	-	200	ns

注：此表中参数来自 TI 公司的 ADS7846 的 datasheet。

2.2 GPIO 与 ADS7846 数字接口连接

GPIO(General Purpose I/O)用于生成和采集特定应用的输入或输出信号。Hi3510 的 GPIO 引脚与 ADS7846 的数字接口连接，模拟接口时序，GPIO 引脚与数字接口信号的连接关系如表 2-3 所示。

在 Hi3510 VSEVB（Video Solution Evaluation Board）中使用表 2-3 中的 GPIO 的 6 个引脚与 ADS7846 的数字接口的 6 个信号线连接。

表2-3 GPIO 端口与 ADS7846 的数字接口的连接关系

Hi3510		ADS7846	
GPIO	方向	SPI	方向
GPIO0_7	I	SPIBUSY	O
GPIO0_6	I	SPIIRQ	O
GPIO0_4	O	SPICLK	I
GPIO0_5	O	SPICSN	I
GPIO1_0	I	SPIDO	O
GPIO1_1	O	SPIDI	I

2.3 信号模拟

合理分配 GPIO 的 6 个引脚，模拟出如图 2-1 的时序，并满足表 2-2 的时序参数。此时，Hi3510 可以通过 GPIO 与 ADS7846 的数字接口通信。即 ADS7846 的数字接口可正确接收到从 GPIO 传送来的命令；并根据接收到的命令，通过 SPIDO 信号线传送触摸屏上的坐标信息。

下面将分别介绍 GPIO 的 6 个引脚的模拟过程。

GPIO0_5

GPIO0_5：模拟片选信号，与 SPICSN 连接。

根据图 2-1 中的时序，片选信号是低电平有效。在以下两种过程中，片选信号应一直保持低电平。

- 通过 GPIO 向 ADS7846 发送命令的过程
- 触摸屏对命令作出响应并送出坐标值的过程

通过软件实现模拟片选信号时，有 3 种实现方式：

- 插入模块时，将片选置为低电平，并保持到卸载模块。
- 打开设备时，将设备置为低电平，并保持到关闭设备。
- 在发送获取坐标数据的命令前，将片选置为低电平，并保持到获取坐标数据结束。

在 GPIO 引脚不冲突的情况下，3 种实现方式都可以得到正确的结果，没有明显区别。但是推荐使用第 3 种。

GPIO0_4

GPIO0_4：模拟串行时钟，与 SPICLK 连接。

根据图 2-1 中的时序，需要各时序参数满足表 2-2。从表 2-2 中可以看出最小时钟周期为 400ns。

因为仅在发送命令和读取数据时，需要用时钟来驱动 SPIBUSY、SPIDO 和 SPIDI3 个信号线。每次获取坐标，最多需要 21 个时钟周期，因此当模拟的时钟周期略大于 400ns 时，占用的 CPU 资源并不是很多。

由于 Linux 内部定时精度不高，因此采用软件循环的方式实现串行时钟的模拟。

GPIO1_1

GPIO1_1：模拟命令输入引脚，与 SPIDI 连接。

根据图 2-1 中的时序，在时钟的下降沿更新该引脚电平；在时钟的上升沿，ADS7846 在 SPIDI 信号线上采集数据。

ADS7846 的命令长度为一个字节，因此发送命令需要 8 个时钟周期。

GPIO0_7

GPIO0_7：连接 SPIBUSY，用于指示触摸屏控制器的忙、闲状态。

当触摸屏控制器接收到获取坐标的命令后，在下一个时钟下降沿，将 SPIBUSY 置为高电平，并在第 2 个时钟下降沿，将 SPIBUSY 恢复为低电平。

软件实现时，只需要在发送完命令以后，继续发送时钟信号，在第 2 个时钟下降沿，触摸屏控制器会将获得的坐标数据送到 SPIDO，软件可以在时钟的上升沿进行读取。软件模拟时，如果 GPIO 资源不多，不需要对此信号模拟，即将 SPIBUSY 信号线置空。

GPIO0_6

GPIO0_6：连接 SPIIRQ，用于指示中断。

该引脚初始为高电平。

- 当触摸屏被按下时，引脚电平跳变为低电平。
- 当触摸屏弹起时，引脚电平恢复为高电平。

该引脚设置为下降沿触发方式。

GPIO1_0

GPIO1_0：连接 SPIDO，用于获取触摸屏控制器发送的坐标值。

在 SPIBUSY 信号变为低电平时，触摸屏控制器将坐标信息发送到 SPIBUSY 信号线。

- 在时钟的上升沿，软件采集该信号。
- 在时钟的下降沿，控制器更新信号线上的数据。

在模拟的过程中，由于时钟周期较短，软件的函数调用和 for 循环的条件判断都会对时钟信号造成影响。SPIDO、SPIDI 上的数据更新都由时钟驱动，时钟信号的变化不会对命令的发送和坐标的获取产生影响，但会降低性能，占用较多的 CPU 资源。

在触摸屏方案中将循环展开，对触摸屏的一次操作，包括发送命令和获取坐标，共需要 21 个时钟周期。

触摸屏控制器支持的坐标格式包括 8bit 和 12bit 两种。因当前方案中发送命令固定，触摸屏控制器仅支持 12bit 的坐标格式。在后续版本中可提供支持 8bit 的坐标格式。

软件模拟 SPI 接口的整个时序过程如图 2-2 所示。

图2-2 软件模拟 SPI 接口过程

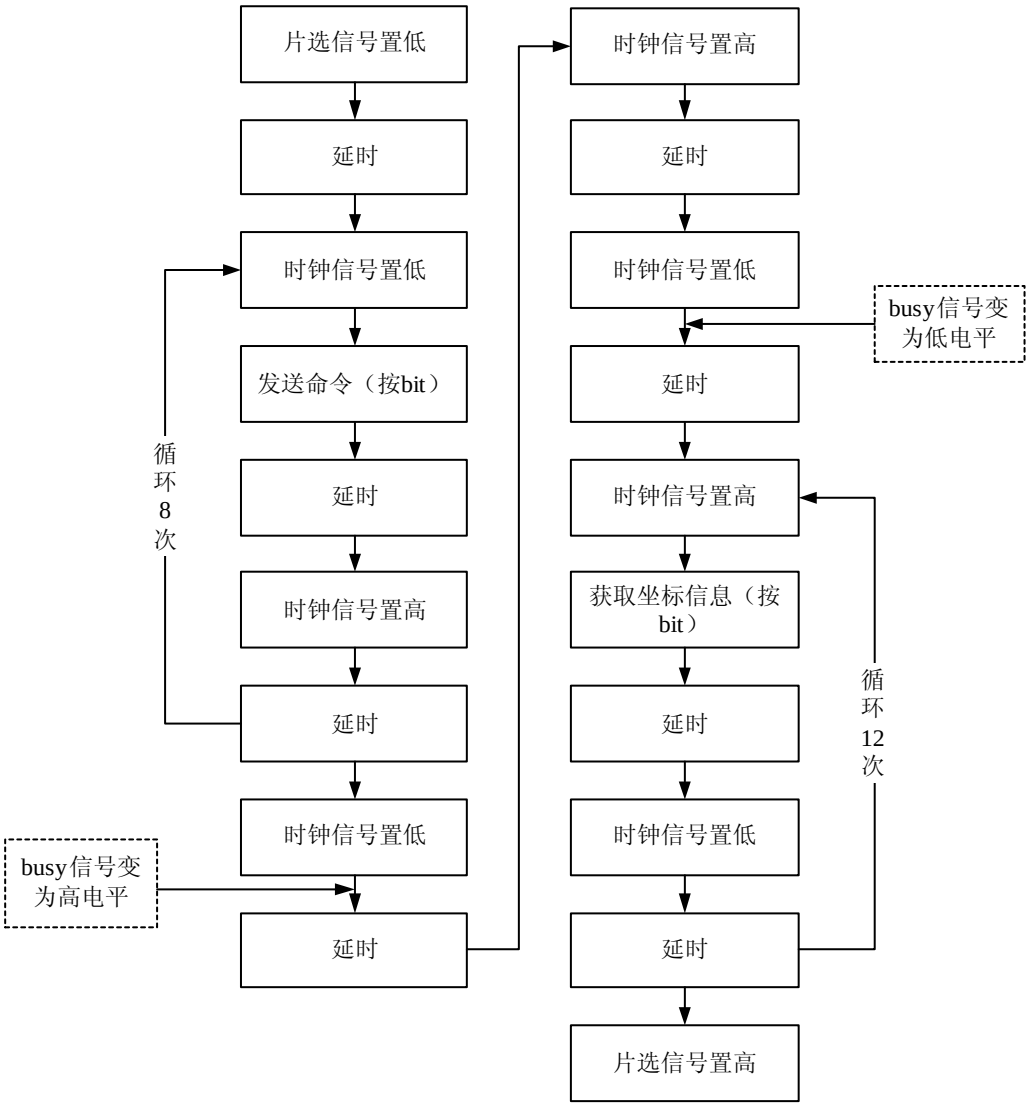


图 2-2 中的虚线框表示不需软件干预。延时的长度可以根据需要调整，只要满足表 2-2 中的要求即可。

3 触摸屏驱动模块

关于本章

本章描述内容如下表所示。

标题	内容
3.1 触摸屏功能	介绍触摸屏功能。
3.2 工作过程	介绍触摸屏工作过程。
3.3 接口说明	介绍触摸屏接口说明。
3.4 移植说明	介绍触摸屏移植说明。
3.5 应用示例	介绍触摸屏应用示例。

3.1 触摸屏功能

在 Linux 开发环境下，触摸屏作为一个字符型设备。用户通过字符型设备的标准 I/O 接口来控制触摸屏控制器，实现触摸屏基本功能。

触摸屏驱动模块可以实现：

- 检测触摸屏的按下状态
- 检测触摸屏的弹起状态
- 检测运动轨迹

触摸屏控制器的功能是：接收命令，并根据命令返回坐标数据。

触摸屏控制器的命令格式请参见 ADS7846 的 datasheet。

I/O 接口对触摸屏控制器的控制，实际上是控制内核层驱动程序向触摸屏控制器发送不同的命令。当前版本中，没有提供 IOCTL 控制接口，驱动程序发送到触摸屏控制器的命令是固定的；在以后的版本中可提供 IOCTL 控制接口。

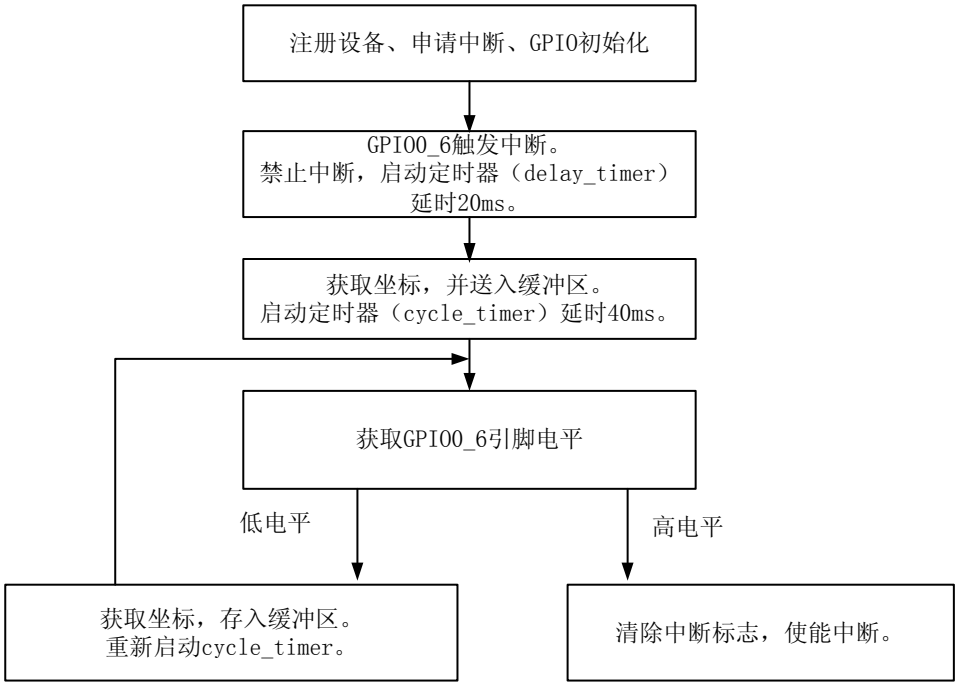
3.2 工作过程

当前的触摸屏驱动模块实现的功能是：检测触摸屏的按下状态和运动轨迹。

- 当触摸屏被按下时，获取按下点的坐标并传送到用户层。
- 当触摸屏被按下并滑动时，模块每隔 40ms 向用户层传送一个坐标数据（传送坐标数据的时间间隔可以通过修改 margin 的值调整），以便跟踪运动轨迹。

触摸屏模块具体的工作流程如图 3-1 所示。

图3-1 触摸屏模块工作流程



当触摸屏被按下时，GPIO0_6 的电平会有抖动现象。为了防止中断的误触发，需要先禁止中断，延时 20ms 以后再读取电平状态。用户层可以通过 read 接口从缓冲区中读取坐标数据。由坐标数据可以解析出触摸屏的运动轨迹。

3.3 接口说明

在用户层只能通过标准的 I/O 接口使用触摸屏模块，所用接口原型如下所示。

```
int open(const char *pathname, int flags);
int close(int fd);
ssize_t read(int fd, const void *buf, size_t count);
```

3.4 移植说明

触摸屏模块的移植包括两方面：

- 选用不同的 GPIO 引脚
- 选用不同的触摸屏控制器

选用不同的 GPIO 引脚

当选用不同的 GPIO 引脚时，只需要修改 hi_touchScreen.h 文件中的信号线宏定义，具体内容如下所示。

```
#define SPICLK_G    GPIO_0
#define SPICLK_B    4
#define SPICSN_G    GPIO_0
#define SPICSN_B    5
#define SPIDI_G     GPIO_1
#define SPIDI_B     1
#define SPIIRQ_G    GPIO_0
#define SPIIRQ_B    6
#define SPIBUSY_G   GPIO_0
#define SPIBUSY_B   7
#define SPIDO_G     GPIO_1
#define SPIDO_B     0
```

其中 xxx_G 表示 GPIO 引脚组号，xxx_B 表示 GPIO 引脚位号。GPIO_x 等组号在 hi_gpio.h 中定义，所以在包含 hi_touchScreen.h 前应先包含 hi_gpio.h。

选用不同的触摸屏控制器

选用不同的触摸屏控制器芯片时，应根据芯片的要求修改时序和命令。时序的调整可以通过修改 gpio_send_command(unsigned char command)函数中的延时长度来实现。命令可以在 touchscreen_readx()和 touchscreen_ready()函数中指定。如下所示，修改 command 的值即可。

```
command = 0xD0;
data = gpio_send_command(command);
```

3.5 应用示例

```
#include <fcntl.h>

static int  fd = -1;

int  main(void)
{
    int result,i;
    int buf;

    /* open keypad in O_RDONLY mode */
    fd=open("/dev/misc/hi_touchscreen",O_RDONLY);
    if(fd<0)
    {
        printf("Can't open\n");
        return -1;
    }
    for(i=0;i<1000;i++)
```

```
    {

            result = read(fd,&buf,sizeof(buf));
            if(result>0)
            {

                printf("the x=%x  ",buf);

            }

            result = read(fd,&buf,sizeof(buf));
            if(result>0)
            {

                printf("the y=%x\n",buf);

            }

      }
   close(fd);
    return 0;
}
```