# FDC2214 串口转接模块使用手册 V1.3（完整版）

淘宝店铺：https://shop125570758.taobao.com

## 1 模块简介

(1). 主要功能：当正确连接传感器，上电后模块会自动通过串口实时发送传感器四个通道的采集值（FDC224 有四个通道），同学们可以用 51 单片机、STM32 单片机、430 单片机等进行接收，快速进行开发，无需调试麻烦的 I2C 协议。
(2). 如果有其他需求，可以调用模块配套的函数来配置传感器，整个开发过程无需翻阅 FDC2214 的英文芯片手册，根据我们的介绍和配置步骤，可以快速熟悉 FDC2214 这款芯片，快速进行开发（具体看功能号介绍，芯片配置）

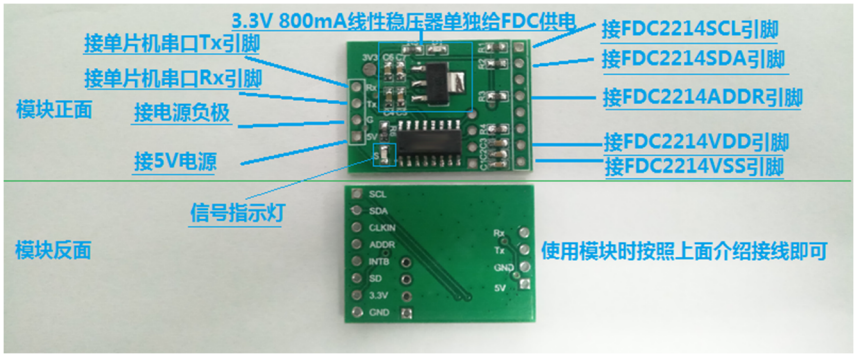
**模块使用注意事项：**

1. 串口波特率为 115200，自己配置单片机串口时需要设置波特率为 115200，否则接收不到数据！
2. 绿色指示灯快速闪烁：表示模块正常工作，正在通过串口发送传感器采集的数据。
3. 绿色指示灯两秒左右闪烁一次，同时通过串口发送 0x23：表示 FDC2214 传感器和本模块连线错误，需要检查传感器的 VDD、VSS、SDA、SCL 连接线是否连接可靠，然后重新上电。

**拿到模块后，按照下面介绍使用模块！**

**第一步：使用串口助手测试本模块的功能。**
**1. 安装“串口助手和驱动”文件夹里的驱动程序（WIN7 系统）**
**2. 按照下图连接传感器，传感器连接好之后，连接 USB 转串口模块（需要自己单独购买）：**

**本模块的 Tx 引脚 接 USB 转串口模块的 Rx 引脚**
**本模块的 Rx 引脚 接 USB 转串口模块的 Tx 引脚**
**本模块的 5V 引脚 接 USB 转串口的 5V 输出**
**本模块的 GND 引脚 接 USB 转串口的 GND**
特别注意正负极不要接反！SCL 和 SDA 引脚不要接反！



连接好之后，把 USB 转串口模块插上电脑，打开“串口助手和驱动”文件夹里的串口助手，设置波特率为 115200，然后选择 USB 转串口模块对应的 COM 口，打开串口，此时可以看到串口助手接收窗口接收到数据。
当接收到数据，格式为：0x76 开头 0x55 结尾的数据帧时（见后面的模块发送协议），表明 FDC2214 串口转接板是可以工作的

1. 如果未收到数据，检查 Tx 和 Rx 信号线是否连接可靠，波特率是否为 115200
2. 如果收到的数据是 0x23，表示传感器连线有问题，按照模块使用注意事项中的说明检查连线
3. 如果接收到的通道数据为 0x00000000，说明传感器未开始探测，或 FDC 无外接晶振（外部时钟）（本模块默认使用外部时钟，需要配置为使用内部时钟（配置详见例程或本文档后面的说明）
4. 如果传感器没有探测电路或探测极板，接收到的通道数据为 0x00FFFFFF，这个是没有问题的，连接传感器探测电路和探测极板后，就会接收到传感器探测数据

## 第二步：连接单片机使用

模块上电默认的配置如下（更改配置见后面的功能号介绍）：

四个通道的传感器参考计数都为：4000
四个通道的传感器稳定计数都为：15
四个通道的传感器驱动电路都为：12
四个通道的参考频率分频系数都为：8
四个通道的传感器测量分频系数都为 1
使能多通道模式
多通道模式下选用的通道为：ch0 ch1 ch2 ch3 四个通道
抗尖峰脉冲滤波器带宽选择为 3.3MHz
选用外部时钟
退出睡眠模式

注意：传感器正常工作需要设置时钟源和退出睡眠模式，本模块默认退出睡眠模式，使用外部时钟源（外面焊接了晶振），如果自己的 FDC2214 使用的是内部时钟源（外面没有焊接晶振），需要设置为内部时钟源 FDC2214 才能正常探测

# 2 模块发送协议（可跳过不看）：

| 帧字节 | DATA[0] | DATA[1] | DATA[2] | DATA[3] | DATA[4] | DATA[5] | DATA[6] | DATA[7] | DATA[8] |
|---|---|---|---|---|---|---|---|---|---|
| 内容 | 0x76 | CH0_B3 | CH0_B2 | CH0_B1 | CH0_B0 | CH1_B3 | CH1_B2 | CH1_B1 | CH1_B0 |
| 帧字节 | DATA[9] | DATA[10] | DATA[11] | DATA[12] | DATA[13] | DATA[14] | DATA[15] | DATA[16] | DATA[17] |
| 内容 | CH2_B3 | CH2_B2 | CH2_B1 | CH2_B0 | CH3_B3 | CH3_B2 | CH3_B1 | CH3_B0 | 0x55 |

帧头：0x76
帧尾：0x55
CH0、CH1、CH2、CH3 分别对应传感器通道 0、1、2、3

数据合成：通道 0 采集值=( CH0_B3<<24)+ (CH0_B2<<16)+ (CH0_B1 <<8)+ CH0_B0。其他通道类似

**同学们在使用时，直接调用我们提供的接收函数，就可以获得四个通道的实时采集数据。**

## 3 模块接收协议（可跳过不看，直接看本模块例程）：

| 帧字节 | DATA[0] | DATA[1] | DATA[2] | DATA[3] | DATA[4] |
|---|---|---|---|---|---|
| 内容 | 0X56 | ID | Data_HSB | Data_LSB | 0X55 |

帧头：0x56
帧尾：0x55
Data_HSB：数据高八位
Data_HSB：数据低八位
ID：功能号，取值为模块配套的例程中的 ClientID 枚举中的元素，通过功能号对传感器各个参数进行设置

**同学们在使用时，直接调用我们提供的发送函数，修改例程中的参数就可以快速按照自己的需求配置传感器**

## ID（功能号，用来配置传感器！模块配套的例程中也有详细介绍，可直接看例程）

唯一一个配置函数 FDC_Set(Client_ID　ID, uint16_t　Data);
　　ID:功能号（例程中有定义，下面有介绍）
　　Data:设置的参数或命令（例程中有定义，下面有介绍）

**功能号详解：**
ID：sleep= 0x1d,
睡眠模式设置
取值范围：
　BOXDisable =0，//退出睡眠模式
　BOXEnable =1,　//进入睡眠模式
调用举例：FDC_Set(reset_device, BOXDisable);//退出睡眠模式 传感器开始探测

ID：reset_device= 0x1c,
复位设备
取值范围：
　BOXEnable =1,//复位设备
调用举例：FDC_Set(reset_device, BOXEnable);//复位 FDC2214 芯片，所有配置复位为芯片出厂默认设置（一般无需调用）

ID：clk_src = 0x18,
传感器时钟源设置
取值范围：
　Clk_src_external　=0x01,外部时钟（选用外部时钟，需要外接 40MHz 晶振）
　Clk_src_internal　=0x02，内部时钟
配置举例：FDC_Set(clk_src, Clk_src_internal);//传感器使用外部时钟

ID：autoscan= 0x15,//自动扫描设置
自动扫描使能位，FDC2214 有三种工作模式：睡眠模式，单通道模式，多通道模式。当使能自动扫描时，为多通道模式：轮流使用指定的[见后面的 autoscan 功能号]通道探测，同一时间只有一个通道探测，分时复用。否则为单通道模式
取值范围：BOXDisable =0 或 BOXEnable =1,
配置举例：FDC_Set(autoscan, BOXEnable);//使能自动扫描，开启多通道模式

ID：autoscan_ch= 0x16
自动扫描选用的通道
取值范围：
　　Multi_2 =0x01,　　//选用 ch0 ch1
　　Multi_3 =0x02,　　//选用 ch0 ch1 ch2
　　Multi_4 =0x03,　　//选用 ch0 ch1 ch2 ch3
配置举例 FDC_Set(autoscan_ch, Multi_2);//设置在自动扫描模式下，选用通道 0 和通道 1

ID：single_ch= 0x17,
单通道模式下（只有一个通道工作），使用的通道设置//当选用单通道模式时，需要配置此项
取值范围：
　Chan0 = 0x01,//选用通道 0
　Chan1 = 0x02,// 选用通道 1
　Chan2 = 0x03,// 选用通道 2
　Chan3 = 0x04,// 选用通道 3
配置举例：FDC_Set(single_ch, Chan0);//单通道模式下选用通道 0

ID：high_current_drive= 0x1b,
　　高电流驱动设置：选用高电流驱动，在其他配置不变的情况下，采集的数据值会变大（传感器振荡的振幅更大）
取值范围：
　BOXDisable =0，
　BOXEnable =1,
调用举例：FDC_Set(high_current_drive, BOXEnable);使高电流驱动传感器

ID：ref_count_ch0 = 0x01,//通道 0 参考计数设置
ID：ref_count_ch1 = 0x02,
ID：ref_count_ch2 = 0x03,
ID：ref_count_ch3 = 0x04,
四个通道的参考计数，此参数决定

取值范围：0x0100-0xFFFF
配置举例：FDC_Set(ref_count_ch0, 0x0123); //设置通道 0 的参考计数值为 0x0123

ID：settle_count_ch0= 0x05,//通道 0 传感器稳定计数
ID：settle_count_ch1= 0x06,
ID：settle_count_ch2= 0x07,
ID：settle_count_ch3= 0x08,
四个通道的稳定计数值，此参数
取值范围：0x0002 - 0xFFFF
配置举例：FDC_Set(settle_count_ch0, 0x0003); //设置通道 0 的稳定计数值为 0x0003

ID：drive_current_ch0= 0x09,//驱动电流
ID：drive_current_ch1= 0x0a,
ID：drive_current_ch2= 0x0b,
ID：drive_current_ch3= 0x0c,
四个通道的传感器驱动电流，用来设置传感器工作时的驱动电流
取值范围：0~31
配置举例：FDC_Set(drive_current_ch0, 12); //设置通道 0 的驱动电流为 12

ID：fref_dividers_ch0= 0x0d,// 参考频率分频设置
ID：fref_dividers_ch1= 0x0e,
ID：fref_dividers_ch2= 0x0f,
ID：fref_dividers_ch3= 0x10,
传感器测量时的参考频率分频系数，用来对传感器参考频率进行分频
取值范围：1~1023
配置举例：FDC_Set(fref_dividers_ch0, 56); //设置通道 0 的参考频率分频系数为 56

ID：fin_dividers_ch0= 0x11,//传感器输入分频设置
ID：fin_dividers_ch1= 0x12,
ID：fin_dividers_ch2= 0x13,
ID：fin_dividers_ch3= 0x14,
传感器测量输入分频系数，对传感器的信号进行分频
取值范围：1 或 2
配置举例：FDC_Set(fin_dividers_ch0, 1); //设置通道 0 分频系数为 1

deglitch_bandwidth = 0x19,
抗尖峰脉冲滤波器带宽设置
取值范围：
Bandwidth_1M =0x01, //设置带宽为 1MHz
Bandwidth_3D3M =0x02, //3.3MHz
Bandwidth_10M =0x03, //10MHz
Bandwidth_33M =0x04, //33MHz
调用举例：FDC_Set(deglitch_bandwidth, Bandwidth_1M);//设置抗尖峰脉冲滤波器带宽为 1MHz

ID：activate_sensor= 0x1a,

传感器激活设置

取值范围：

BOXDisable =0，

BOXEnable =1,

调用举例：FDC_Set(activate_sensor, BOXDisable);使能传感器激活