# Android USB转串口方案

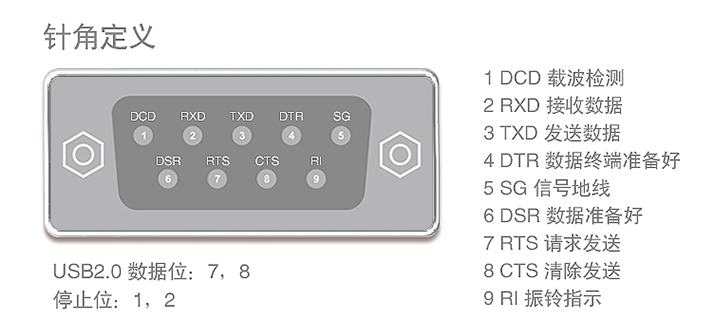
# 功能描述

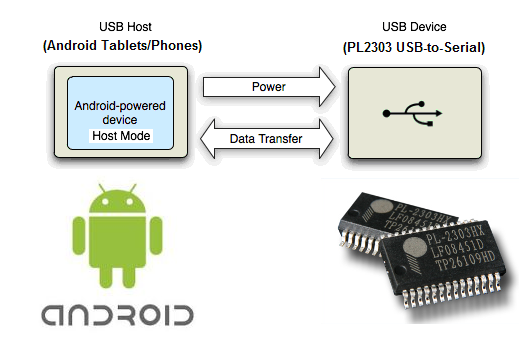
Android 手机或平板通过otg供电连接usb转串口芯片FT232/PL2303HXD实现Android设备与RS232设备的实时通讯

该方案适用于Android手机/平板的Micro USB/Type C接口转RS232串口以进行串口设备的访问及调试

Android端需求:无需root有otg功能即可







硬件串口线:参考Z-TEK ZE614 ,PL2303HXD

# 2.usb转串口芯片对比

FT232:最稳定

PL2303:用的多,最便宜

Android驱动可参考:

<http://www.prolific.com.tw/US/ShowProduct.aspx?p_id=230&pcid=41>

CH3404

稳定性和价格对比:FT232>CH340>PL2303

# 3.软件设计

1.内核驱动

一般高版本的内核都集成了usb转串口驱动

$ ls /sys/bus/usb-serial/drivers/

FTDI FT232驱动:drivers/usb/serial/ftdi\_sio.c

PL2303驱动:drivers/usb/serial/pl2303.c

驱动配置:

Device Drivers --->

[\*] USB support --->

<\*> USB Serial Converter support --->

[\*] USB Generic Serial Driver //通用usb转串口驱动

<M> USB FTDI Single Port Serial Driver //ft232驱动

<\*> USB Prolific 2303 Single Port Serial Driver //PL2302驱动

驱动装好后会出现/dev/ttyUSB0

chmod 777 /dev/ttyUSB0

操作/dev/ttyUSB0即可

1. 对于一般的手机我们无法配置其内核解决方安

可以不用root Android 设备

Ft232:下载Android Java Driver

<http://www.ftdichip.com/Drivers/D2XX.htm>

Android\_Java\_D2xx\_2.06.zip

1. FT232软件设计

1)集成d2xx.jar库



1. 检测USB设备插入

在AndroidManifest.xml的activity中加入

<intent-filter>

<action android:name="android.hardware.usb.action.USB\_DEVICE\_ATTACHED" />

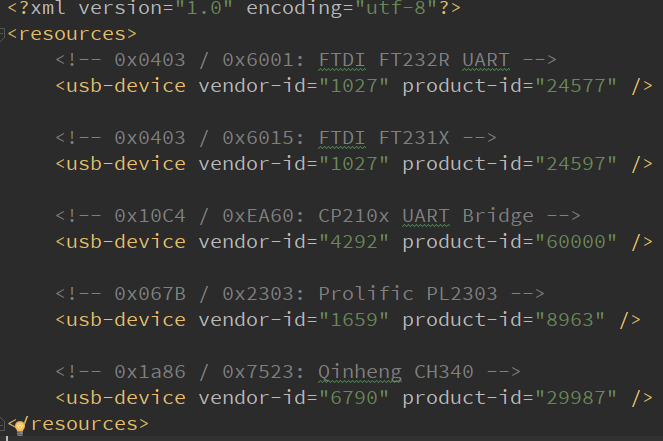
</intent-filter>

<meta-data android:name="android.hardware.usb.action.USB\_DEVICE\_ATTACHED"

android:resource="@xml/device\_filter" />

在xml目录下加入device\_filter.xml

声明对应的感兴趣的USB设备的资源文件



做好以上配置后当插入FT232 USB转串口后会弹出一个询问框



选择串口助手即可进入串口助手的应用

1. 点击FT232按钮进入主界面

打开设备:如果打开成功则默认初始化参数



打开成功则把默认值设成参数,修改参数设置右以更改默认值

//设置默认参数

public void setDefConfig(){

if (ftDevice.isOpen() == false) {

return;

}

//reset to UART mode for 232 devices

ftDevice.setBitMode((byte)0,D2xxManager.FT\_BITMODE\_RESET);

ftDevice.setBaudRate(baud);

ftDevice.setDataCharacteristics(dataBits,stopBits,parity);

ftDevice.setFlowControl(flowControl,(byte)0x0b,(byte)0x0d);

}

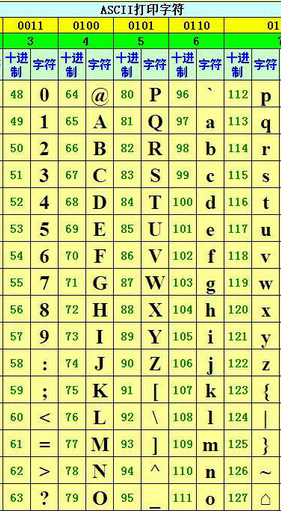
1. 串口的发送与接收

发送字符串与发送十六进制的区别:

在计算机中，所有的数据在[存储](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8" \t "/tmp/wps-jz/x/_blank)和运算时都要使用[二进制数](https://baike.baidu.com/item/%E4%BA%8C%E8%BF%9B%E5%88%B6%E6%95%B0" \t "/tmp/wps-jz/x/_blank)表示

ASCII 码使用指定的7 位或8 位[二进制数](https://baike.baidu.com/item/%E4%BA%8C%E8%BF%9B%E5%88%B6%E6%95%B0" \t "/tmp/wps-jz/x/_blank)组合来表示128 或256 种可能的[字符](https://baike.baidu.com/item/%E5%AD%97%E7%AC%A6" \t "/tmp/wps-jz/x/_blank)。标准ASCII 码也叫基础ASCII码

在计算机中一个汉字通常用两个字节的编码表示，我国制定了“中华人民共和国国家标 准信息交换汉字编码字符集(基本集GB2312—1980)”，简称国标码



串口发送的两种方式:ASCII(字符串)和HEX(十六进制)发送



在使用串口发送数据时可以选择字符串发送或者十六进制发送，通常情况下我们习惯选用字符串发送数据,默认的是字符串(ASCII)发送和接收,当选择hex显示时,表示用十六进制发送和接收

在计算机中，数据是以二进制的形式存储的，例如十进制 1（10）在计算机中用 0000 0001（2）来表示。我们在用串口发送数据的时候首先将待数据转换为对应的ASCII码，然后再将这些ASCII码按照二进制的方式一位一位的发送出去。

例如我们要发送一串数据“A852010100000000A91A”，以字符串和十六进制两种方式发送：

（1）字符串发送

         串口以字符串发送数据，首先将字符串转化为二进制，格式如下：

          0000 1010     0000 1000    0000 0110    0000 0010    0000 0000   0000 0001   0000 0000    0000 0001

                A                   8                  5                      2                    0                1                    0                    1

          0000 0000     0000 0000   0000 0000    0000 0000     0000 1010    0000 1001    0000 0001    0000 1010

                          0                   0                     0                   0                    A                   9                   1                   A

         然后按照8位（串口设置数据位为8位）形式将数据发送出去。

         串口接收的数据格式如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 数据内容 | 编号 | 数据内容 |
| 0 | A | 8 | 0 |
| 1 | 8 | 9 | 0 |
| 2 | 5 | 10 | 0 |
| 3 | 2 | 11 | 0 |
| 4 | 0 | 12 | A |
| 5 | 1 | 13 | 9 |
| 6 | 0 | 14 | 1 |
| 7 | 1 | 15 | A |

 （2）十六进制发送数据

          串口以十六进制发送数据，首先将数据转化为：

          1010 1000     0110 0010    0000 0001     0000 0001    0000 0000    0000 0000     1010 1001     0001 1010

              0XA8            0X52               0X01             0X01              0X00           0X00                0XA9            0X1A

         然后按照8位（串口设置数据位为8位）形式将数据发送出去。

         串口接收的数据格式如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 编号 | 数据内容 | 编号 | 数据内容 |
| 0 | A8 | 4 | 00 |
| 1 | 52 | 5 | 00 |
| 2 | 01 | 6 | A9 |
| 3 | 01 | 7 | 1A |

public static String BytestoHexString(byte[] array, int length)

当以hex显示时要把byte数组转为hex字符串

public static byte[] hexStringToByteArray(String hexString)

当输入的是hex命令时,要把hexString转成byte数组

1. PL2303HXD软件设计

官方网址参考:

<http://www.prolific.com.tw/US/ShowProduct.aspx?p_id=230&pcid=41>

Download [PL2303HXDSimpleTest sample code for Android Studio](http://www.prolific.com.tw/UserFiles/files/PL2303HXDSimpleTest_v20216_20170309_AndroidStudio.zip).

Android主机模式:主机模式是指Android设备充当USB主机并为总线供电。此模式下，Android设备需支持USB主机功能或OTG功能，此时Android设备的USB主机称为USB嵌入式主机EH（Embedded Host）。与PC上的USB主机相比，EH设备可能无法为连接到其总线上的未识别外围设备加载驱动程序，因此它们对其目标外围设备列表TPL（Target Peripheral List）进行了定义。这些外围USB设备大部分为HID设备（Human Interface Device）、BOMS设备（Bulk Only Mass Storage，如U盘）和CDC设备（Comm-  
unication Device Class，USB通信设备类，如打印机），其驱动程序已存在于Android平台的系统中（Linux Kernel），因此Android设备可以与其直接通信。

确定Android设备的内核驱动是否支持USB Host功能

$ ls system/lib/libusbhost.so

system/lib/libusbhost.so

确认Android系统是否已开放了usb host权限[重要]

$ ls system/etc/permissions/android.hardware.usb.host.xml

system/etc/permissions/android.hardware.usb.host.xml

内容如下:

<permissions>

<feature name="android.hardware.usb.host" />

</permissions>

查找是否存在:

system/etc/permissions/handheld\_core\_hardware.xml

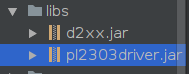
或者tablet\_core\_hardware.xml

在permissions目录下的handheld\_core\_hardware.xml中添加

<feature name=”android.hardware.usb.host”>

在AS中集成pl2303hxd java 层的驱动库

目前集成的库是20170512/lib/1port/pl2303driver.jar



1. 打开串口

private PL2303Driver mSerial;

private static final String ACTION\_USB\_PERMISSION =

"com.bshui.androidserial.USB\_PERMISSION";

private PL2303Driver.BaudRate baud = PL2303Driver.BaudRate.B115200;//波特率

private PL2303Driver.DataBits dataBits= PL2303Driver.DataBits.D8;//8bit

private PL2303Driver.StopBits stopBits = PL2303Driver.StopBits.S1;//1bit

private PL2303Driver.Parity parity = PL2303Driver.Parity.NONE;//none

private PL2303Driver.FlowControl flowControl = PL2303Driver.FlowControl.OFF;//none

mSerial = new PL2303Driver((UsbManager)getSystemService(Context.USB\_SERVICE),

this,ACTION\_USB\_PERMISSION);

//check USB host function

if(!mSerial.PL2303USBFeatureSupported()){

Toast.makeText(this,

"不支持USB host API",Toast.LENGTH\_LONG).show();

mSerial = null;

}

if(!mSerial.enumerate()){

Toast.makeText(this,

"no more devices found",Toast.LENGTH\_LONG)

.show();

}

try{

//在进入界面后打开串口

Thread.sleep(1500);

startOpenDev();

}catch (Exception e){

e.printStackTrace();

}

private void startOpenDev(){

if(mSerial==null){

return;

}

mSerial.InitByDefualtValue();

if(mSerial.isConnected()){

if(!mSerial.InitByBaudRate(baud, 700)){

if(!mSerial.PL2303Device\_IsHasPermission()){

Toast.makeText(this,

"打开失败,没有权限",

Toast.LENGTH\_LONG).show();

//打开成功

tvPlstatus.setText("串口状态:打开失败");

}

}else{

//打开成功

tvPlstatus.setText("串口状态:打开");

//默认配置

try {

mSerial.setup(baud, dataBits, stopBits, parity, flowControl);

}catch (IOException e){

e.printStackTrace();

}

}

}else{

tvPlstatus.setText("串口状态:连接失败");

}

}

}

读写:

mSerial.read(readData);

//写数据

mSerial.write(data,data.length);

1. 通用串口软件设计

Github下载地址:

<https://github.com/cepr/android-serialport-api>

1. 生成.so

新建一个Android Project勾选include C++ support

包名:android.serialport.api

自定义一个java文件生成所需要的接口

SerialPort.java

// JNI

private native static FileDescriptor open(String path, int baudrate);

public native void close();

static {

System.loadLibrary("serial\_port");

}

通过alt+enter键在cpp/native-lib.cpp中生成相应的接口

实现open和close的接口

修改CMakeLists.txt

生成.so

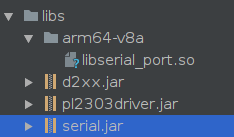
./app/build/intermediates/cmake/debug/obj/arm64-v8a/libserial\_port.so

arm64-v8a设备兼容arm64-v8a、armeabi-v7a、armeabi；

生成.jar

./mylibrary/build/intermediates/bundles/debug/classes.jar

把.so和.jar添加到工程中



build.gradle app的android中加入

sourceSets {

main {

jniLibs.srcDir 'libs'

}

}

1. 初始化

private SerialPort mSerialPort;

private OutputStream mOutputStream;

private InputStream mInputStream;

try {

mSerialPort = new SerialPort(new File(serialnum), baudrate);

mOutputStream = mSerialPort.getOutputStream();

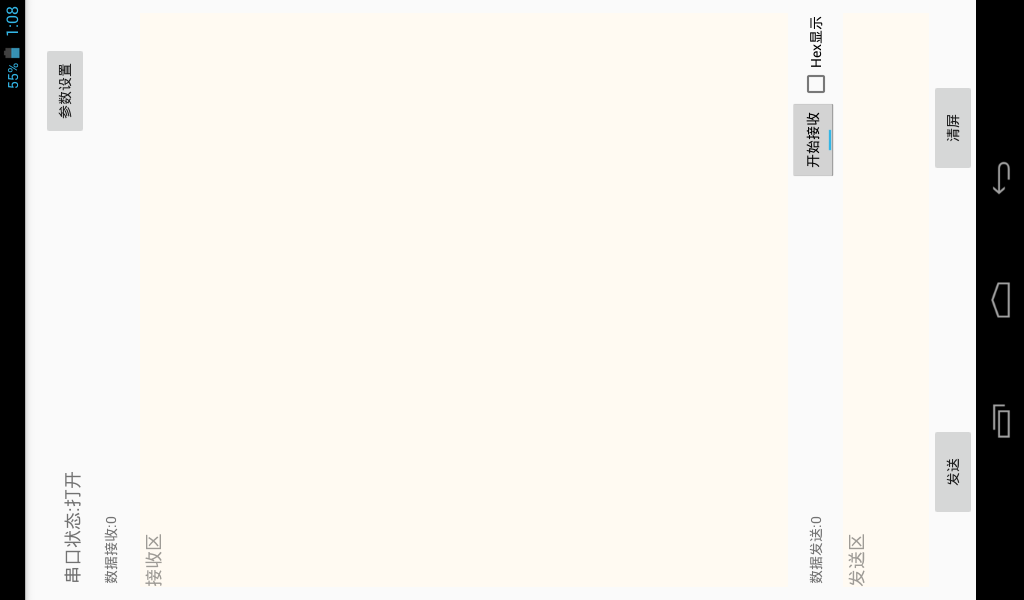
mInputStream = mSerialPort.getInputStream();

}catch (IOException e){

e.printStackTrace();

}

在imx6 平台上操作ttyUSB0成功



1. 写操作

String writeData = edSend.getText().toString().trim().replace(" ", "");

try {

if (mOutputStream != null) {

mOutputStream.write(writeData.getBytes());

sendlen += writeData.length();

tvSend.setText("发送数据:"+sendlen);

} else {

return;

}

}catch (IOException e){

e.printStackTrace();

}

4)读操作

while(bReadThreadGoing==true){

try{

Thread.sleep(50);

}catch (InterruptedException e){

e.printStackTrace();

}

try{

if(mInputStream == null)

return;

iavailable = mInputStream.read(readData);

if(iavailable > 0) {

//字符串显示

for (int i = 0; i < iavailable; i++) {

readDataToText[i] = (char) readData[i];

}

Message msg = handler.obtainMessage();

handler.sendMessage(msg);

}

}catch (IOException e){

e.printStackTrace();

return;

}

}