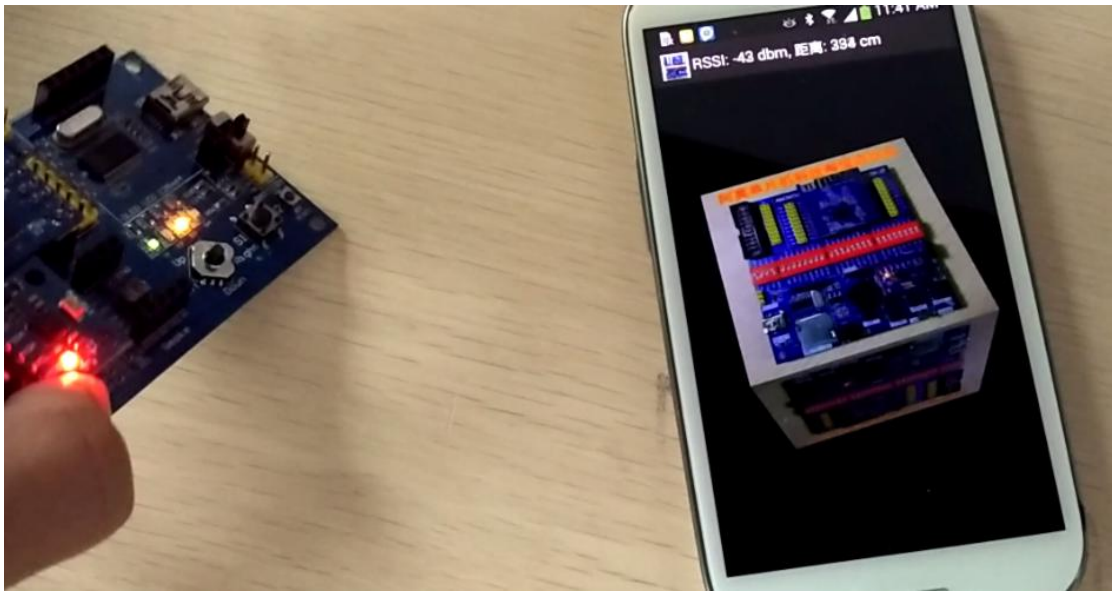
旋转魔方-运动姿态

www.AmoMcu.com

阿莫单片机 出品

2016-12-11 v1.1



目录

1，要实现的功能……2
2，开发环境……2
2.1 硬件……2
2.2 软件……2
3，源码位置……3
4，源码分析……4
4.1 上电就广播……5
4.2 按键上传……5
4.3 mpu6050 加速度数据读取与上传……5
4.4 蜂鸣器报警（LED2 闪）……8
4.5 Android 源代码分析……8

4.5.1 mpu6050 数据获取……9
5，连接设备……12
5.1 用 GPIO 模拟 IIC 的接法……12
5.2 用硬件 IIC 的接法……12
6，下载运行……13
6.1 下载……13
7，测试……13
8，联系我们……14

1，要实现的功能

实现读取 mpu6050 数据， 发送到手机， android 手机上根据四元数变换成欧拉角，并通过一个立方体进行姿态演示。

注意， 目前 CC254x 读取 mpu6050 数据有一些问题，导致数据缓慢，后面在优化解决。目前代码与防丢器的代码是一样的，后面需要优化加速度功能。

2，开发环境

2.1 硬件

1、 1 个开发板
2、 CC-Debugger 仿真器
3、 最好有一个 mpu6050 模块

2.2 软件

1、 ble 协议栈，版本：1.3.2
2、 IAR for 8051 开发环境，版本：8.10
3、 Flash Programmer 固件烧写软件。
4、 amokeyfob.apk

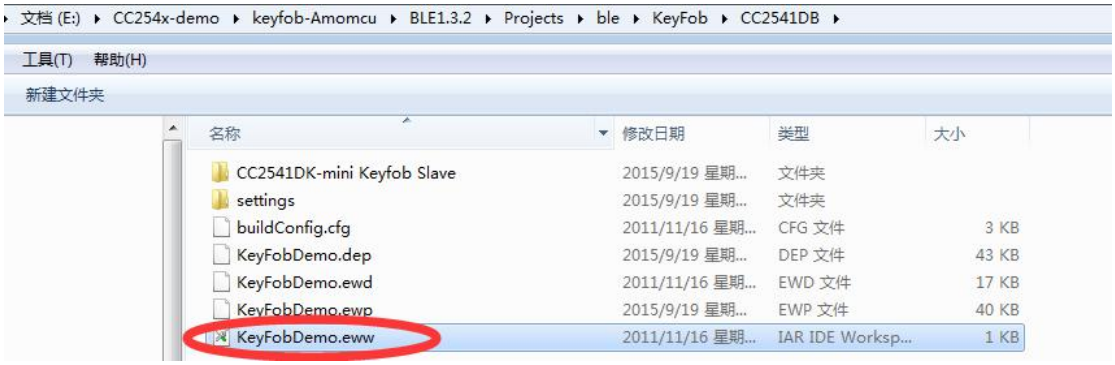
3，源码位置

【1】您下载这个源码后，如下图所示（版本号和日期可能有所不同）：

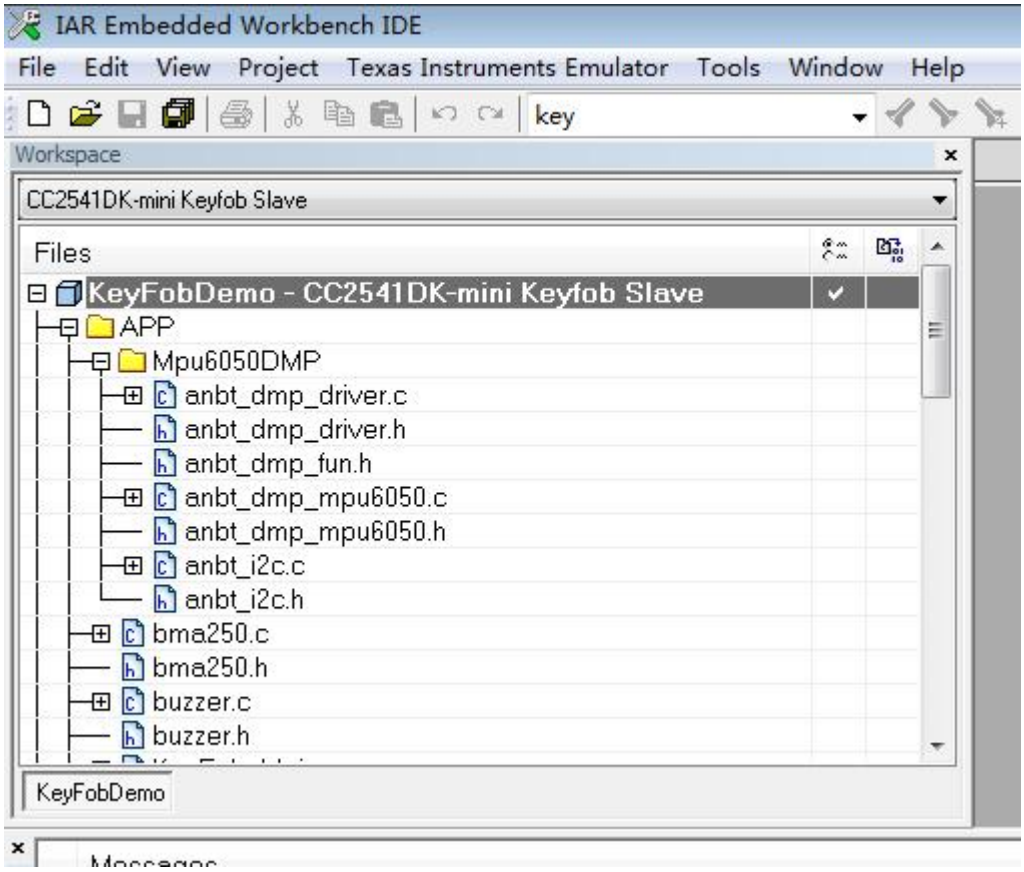
把 keyfob-Amomcu **复制到一个短路径，否则打开工程会出错。**

例如以下：

打开工程（以 CC2541 工程为例）：



打开后如下：



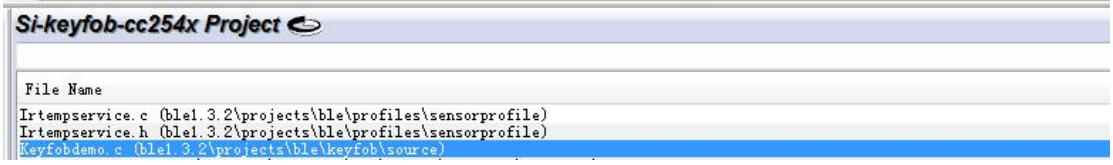
4，源码分析

实现旋转魔方的原理是： mpu6050 输出的欧拉角与四元数，手机实时读取该欧拉角与四元数，实现数据到空间角度的运动变化图形呈现。

至于欧拉角与四元数，不在我们的讨论范围，下面部分将描述如何实现数据的采集与上传。

4.1 上电就广播

```
00326: void KeyFobApp_Init( uint8 task_id )
00327: {
00328:   keyfobapp_TaskID = task_id;
00329:
00330:   // Setup the GAP
00331:   VOID GAP_SetParamValue( TGAP_CONN_PAUSE_PERIPHERAL, DEFAULT_CONN_PAUSE_PE
00332:
00333:   // Setup the GAP Peripheral Role Profile
00334:   {
00335:     // For the CC2540DK-MINI keyfob, device doesn't start advertising until
00336:     uint8 initial_advertising_enable = TRUE;
00337:
00338:     // By setting this to zero, the device will go into the waiting state a
00339:     // being discoverable for 30.72 second, and will not being advertising
00340:     // until the enabler is set back to TRUE
00341:     uint16 gapRole_AdvertOffTime = 0;
00342:
00343:     uint8 enable_update_request = DEFAULT_ENABLE_UPDATE_REQUEST;
00344:     uint16 desired_min_interval = DEFAULT_DESIRED_MIN_CONN_INTERVAL;
00345:     uint16 desired_max_interval = DEFAULT_DESIRED_MAX_CONN_INTERVAL;
```

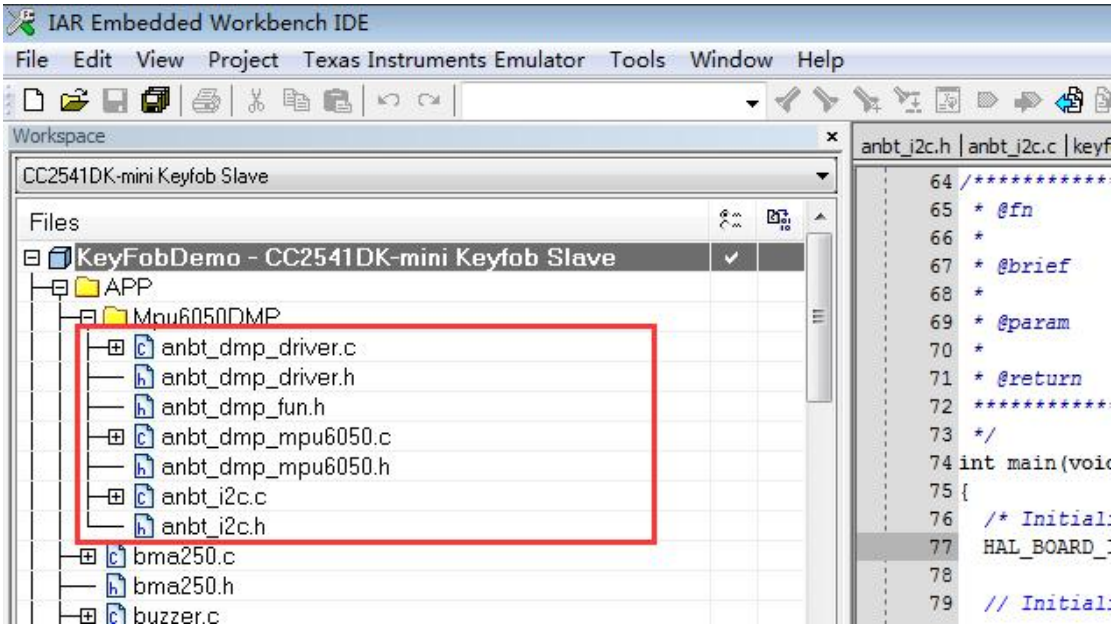


4.2 按键上传

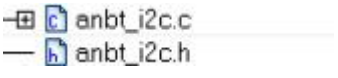
4.3 mpu6050 加速度数据读取与上传

Mpu6050 通过 iic 接口连接到 cc2540 或者 cc2541 单片机。

Mpu6050 驱动相关的文件如下：



其中：



这两个文件主要实现 iic 的读写驱动。

如果你要修改模拟 io 口做 iic， 或者硬件 iic， 主要修改这两个文件实现适配即可。

在 keyfobdemo.c 中， 我们主要用 宏定义 SENSOR_MPU6050_DMP 括住了与 mpu6050 传感器相关的代码。

例如，mpu6050 初始化代码如下：

```
00438: #if defined( SENSOR_MPU6050_DMP  )
00439:     g_is_mpu6050_ok_flag = AnBT_DMP_MPU6050_Init();        //6050DMP初始化
00440:     if(g_is_mpu6050_ok_flag)
00441:     {
00442:         HalLedSet( (HAL_LED_3), HAL_LED_MODE_ON );
00443:     }
00444:     else
00445:     {
00446:         HalLedSet( (HAL_LED_3), HAL_LED_MODE_OFF );
00447:     }
00448: #endif
```

然后启动一个定时器

KFD_ACCEL_MPU6050_READ_EVT

该定时器，的调用如下：

```
00549:     return (events ^ KFD_ACCEL_READ_EVT);
00550:   } ? end if events&KFD_ACCEL_READ... ?
00551:
00552: #if defined( SENSOR_MPU6050_DMP  )
00553:     if (events & KFD_ACCEL_MPU6050_READ_EVT)
00554:     {
00555:       events &= ~KFD_ACCEL_MPU6050_READ_EVT;
00556:
00557:       KeyFobDemo_mpu6050Read(); // 读取mpu6050数据到全局变量
00558:       if(g_is_mpu6050_timer_flag)
00559:       {
00560:         osal_start_timerEx( keyfobapp_TaskID, KFD_ACCEL_MPU6050_READ_EVT, ACC
00561:       }
00562:     }
00563: #endif
```

每隔 10ms （ACCEL_MPU6050_READ_PERIOD 定义）， 读取一次 mpu6050 的 dmp 输出，并保存到全局变量中， 全局变量如下：

```
static short mpu6050_gyro[3] = {0,0,0};
static short mpu6050_accel[3] = {0,0,0};
static short mpu6050_quat[4] = {0,0,0,0};//四元数存放数组
```

对应读取的函数如下:

```
01208: static uint8 KeyFobDemo_mpu6050Read()
01209: {
01210: //    char stringArray[128];
01211:
01212:     HalLedSet(HAL_LED_1, HAL_LED_MODE_ON);
01213:     if(TRUE == g_is_mpu6050_ok_flag)
01214:     {
```

.....

```
01231:           mpu6050_gyro[0] = gyro[0];
01232:           mpu6050_gyro[1] = gyro[1];
01233:           mpu6050_gyro[2] = gyro[2];
01234:
01235:           mpu6050_accel[0] = accel[0];
01236:           mpu6050_accel[1] = accel[1];
01237:           mpu6050_accel[2] = accel[2];
01238:
01239:           q0=quat[0] >> 16;
01240:           q1=quat[1] >> 16;
01241:           q2=quat[2] >> 16;
01242:           q3=quat[3] >> 16;
01243:
01244:           //四元数存放数组
01245:           mpu6050_quat[0] = q0;
01246:           mpu6050_quat[1] = q1;
01247:           mpu6050_quat[2] = q2;
01248:           mpu6050_quat[3] = q3;
```

，如果手机 app 连接上我们开发板后，手机上的 app 启动读取函数(4.5 节有介绍)，触发 keyfobdemo.c 中的函数如下：

```
01286: static uint8 KeyFobDemo_accelReadCB( uint16 param, void *value )
01287: {
01288:     uint8 len = ACCEL_MPU6050_OUT_LEN;
01289:
01290:     if(param == ACCEL_MPU6050_OUT_UUID)
01291:     {
01292:         // 未启动则要先启动
01293:         if(FALSE == g_is_mpu6050_timer_flag)
01294:         {
01295:               //osal_start_timerEx( keyfobapp_TaskID, KFD_ACCEL_MPU6050_READ_EVT
01296:               osal_start_timerEx( keyfobapp_TaskID, KFD_ACCEL_MPU6050_READ_E
01297:               //osal_start_reload_timer( keyfobapp_TaskID, KFD_ACCEL_MPU6050_REA
01298:               g_is_mpu6050_timer_flag = TRUE;
01299:         }
01300:
01301:         HalLedSet(HAL_LED_2, HAL_LED_MODE_ON);
01302:
01303:         len = ACCEL_MPU6050_OUT_LEN;
```

注意该函数 ：

static uint8 KeyFobDemo_accelReadCB(uint16 param, void *value)

其返回的是数据长度 （为 20 个字节），也就是 void *value 的有效长度，而 value 相当于一个数组的首地址，该数组存放的是欧拉角与四元数等数据。

数据经过处理如下：

```
01305: #if defined( SENSOR_MPU6050_DMP  )
01306:         ((uint8*)value)[1] = (mpu6050_gyro[0]>>8) & 0xff;   ((uint8*)value)[0] = mpu6050_gyro[0] & 0xff;
01307:         ((uint8*)value)[3] = (mpu6050_gyro[1]>>8) & 0xff;   ((uint8*)value)[2] = mpu6050_gyro[1] & 0xff;
01308:         ((uint8*)value)[5] = (mpu6050_gyro[2]>>8) & 0xff;   ((uint8*)value)[4] = mpu6050_gyro[2] & 0xff;
01309:
01310:         ((uint8*)value)[7] = (mpu6050_accel[0]>>8) & 0xff;   ((uint8*)value)[6] = mpu6050_accel[0] & 0xff;
01311:         ((uint8*)value)[9] = (mpu6050_accel[1]>>8) & 0xff;   ((uint8*)value)[8] = mpu6050_accel[1] & 0xff;
01312:         ((uint8*)value)[11] = (mpu6050_accel[2]>>8) & 0xff;   ((uint8*)value)[10] = mpu6050_accel[2] & 0xff;
01313:         //
01314:         ((uint8*)value)[13] = (mpu6050_quat[0]>>8) & 0xff;   ((uint8*)value)[12] = mpu6050_quat[0] & 0xff;
01315:         ((uint8*)value)[15] = (mpu6050_quat[1]>>8) & 0xff;   ((uint8*)value)[14] = mpu6050_quat[1] & 0xff;
01316:         ((uint8*)value)[17] = (mpu6050_quat[2]>>8) & 0xff;   ((uint8*)value)[16] = mpu6050_quat[2] & 0xff;
01317:         ((uint8*)value)[19] = (mpu6050_quat[3]>>8) & 0xff;   ((uint8*)value)[18] = mpu6050_quat[3] & 0xff;
01318: #else
```

总结如下：

1， 初始化 mpu6050.

2， 开始 ble 广播，此时读取 mpu6050 的定时器也可以不启动，如本例程就没有启动。

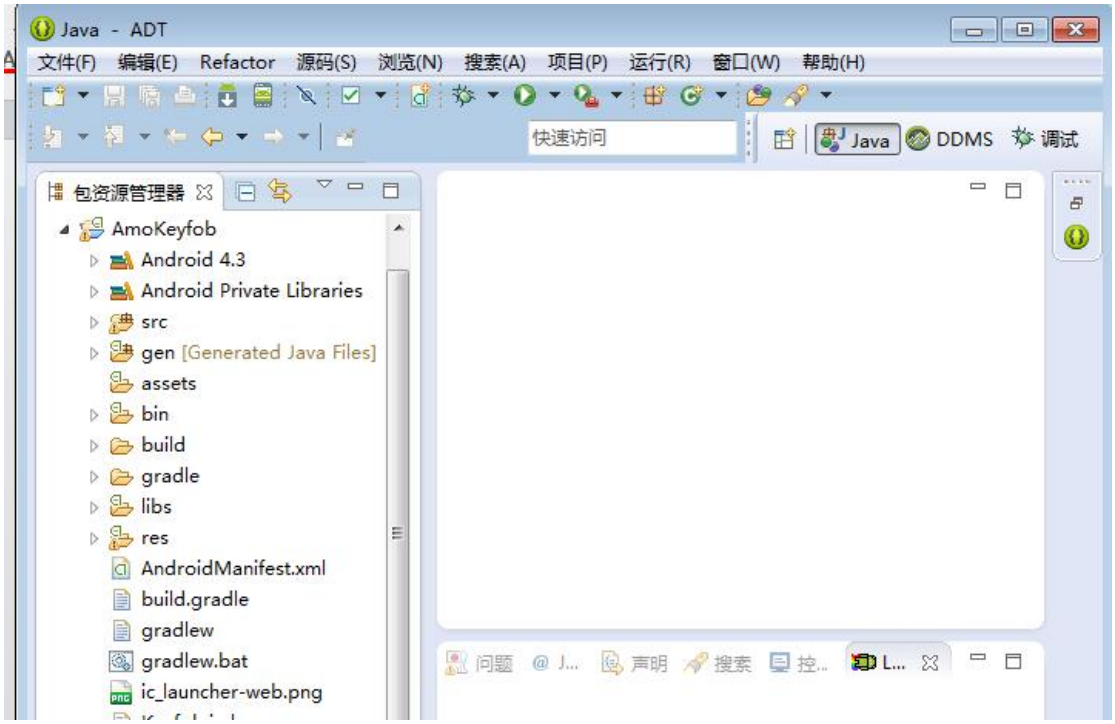
3， 手机 app 连接开发板，连接后第一次读取 mpu6050 数据时，启动定时器（读取 mpu6050 的定时器 ）KFD_ACCEL_MPU6050_READ_EVT， 这是一个 10ms 的重复调用的定时器，每间隔 10ms 读取一次 mpu6050 的数据存放到全局变量中，方便以后手机 app 读取最新的数据。实测发现每隔 100ms，手机 app 读取一次数据。

4.4 蜂鸣器报警（LED2 闪）

4.5 Android 源代码分析

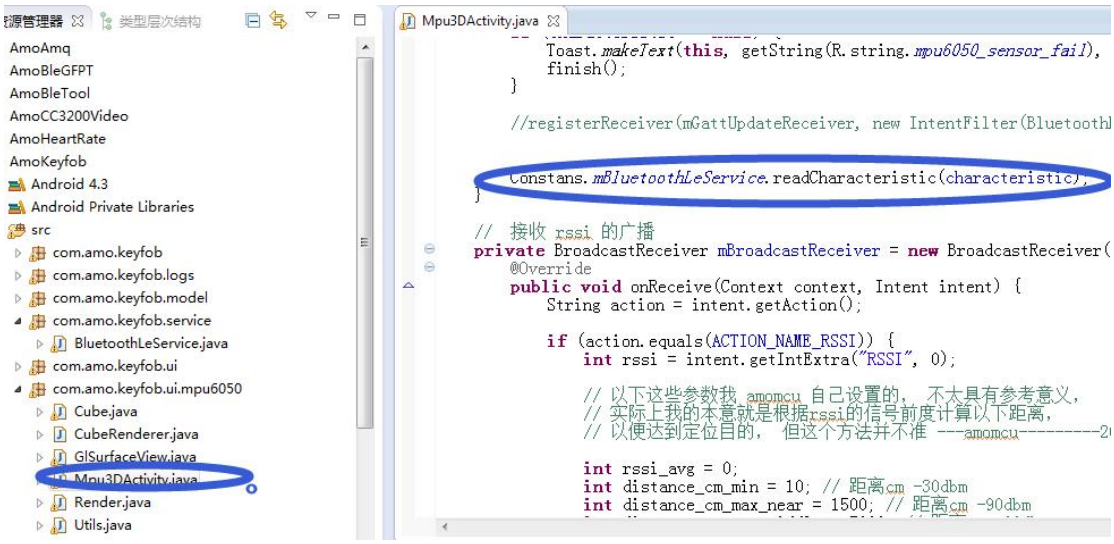
Apk 与对应的源代码如下：

用 eclipse 导入下图;



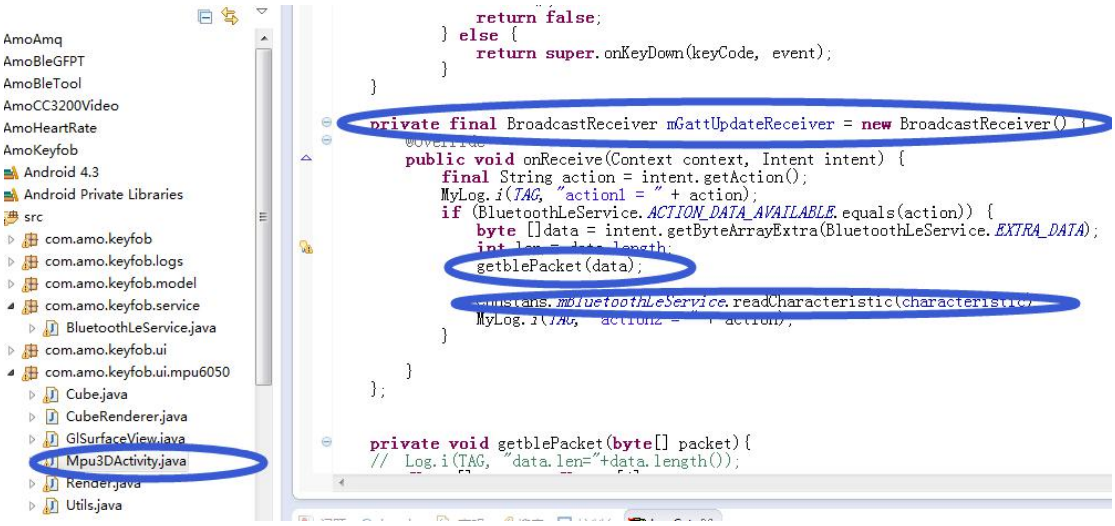
4.5.1 mpu6050 数据获取

启动 app 后并连接开发板后， 将执行以下文件 **Mpu3DActivity.java** 中的函数：

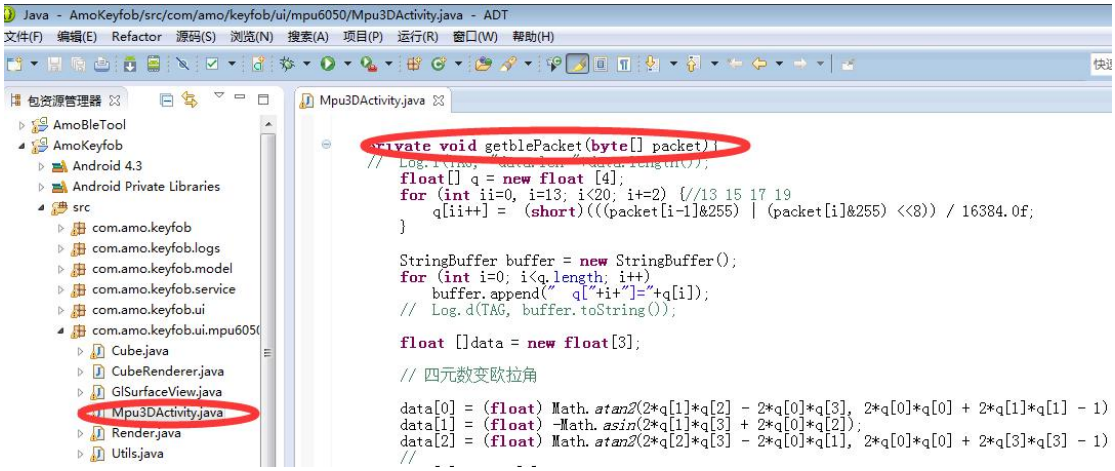


函数 Constans.*mBluetoothLeService*.readCharacteristic(characteristic) 就是启动一个读取操作，读取开发板上 20 个字节的数据，格式如 4.3 节描述。

然后，读取到数据后，会触发回调函数：
private final BroadcastReceiver mGattUpdateReceiver = **new** BroadcastReceiver()
的 **public void** onReceive(Context context, Intent intent)



getblePacket(data); 该函数处理读取到的 20 个字节数据，并显示图形，如下图：



如上图：
getblePacket(**byte**[] packet)中的 packet 是数组，共 20 个字节；

其中都是 16 位表示，高位在后：
packet[0]~packet[1] 是加速度的 ax
packet[2]~packet[3] 是加速度的 ay
packet[4]~packet[5] 是加速度的 az

packet[6]~packet[7] 是陀螺仪的 gx
packet[8]~packet[9] 是陀螺仪的 gy
packet[10]~packet[11] 是陀螺仪的 gz

packet[12]~packet[19] 是 mpu6050 的 dmp 输出的四元数

代码内 **float**[] q = **new float** [4];
后面 q 出来的就是四元数。

后面计算出欧拉角是：
float []data = **new float**[3];

最后，根据欧拉角调用一个立方体显示：
mGLSurfaceView.onMpu6050Sensor(data[2], data[1], data[0]);

处理完成 图形之后，又马上执行一次读数据 的函数 ：
Constans.*mBluetoothLeService*.readCharacteristic(characteristic)
下一次又能读取数据了。

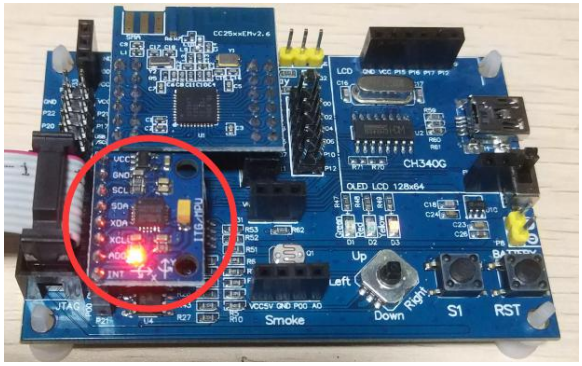
总结： android app 读取数据流程如下：

1， Constans.*mBluetoothLeService*.readCharacteristic(characteristic) ，启动读函数
2， 读取到数据，则执行： **private final** BroadcastReceiver mGattUpdateReceiver = **new** BroadcastReceiver()
的 **public void** onReceive(Context context, Intent intent)
然后在该函数中执行 getblePacket(data)， 实现数据处理与图形显示。
然后再执行 Constans.*mBluetoothLeService*.readCharacteristic(characteristic) ，启动读函数
.....
是一个循环呀.....
readCharacteristic ->
onReceive ->
getblePacket ->
readCharacteristic ->
.....

5，连接设备

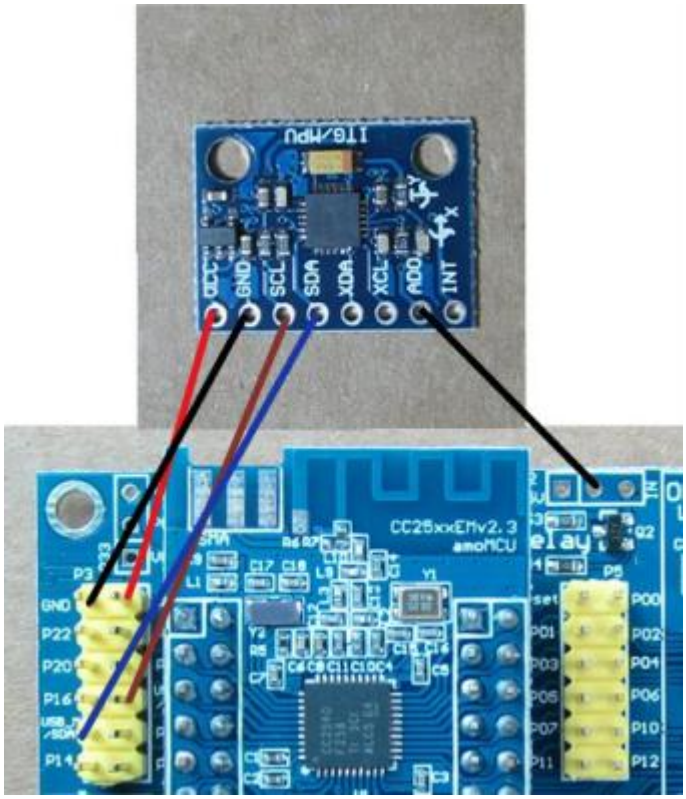
5.1 用 GPIO 模拟 IIC 的接法

适合 cc2540 与 cc2541 芯片， 直接把 mpu6050 模块插入到开发板的 P21 座子上即可，如下图：



5.2 用硬件 IIC 的接法

仅适合 cc2541 芯片（cc2540 芯片没有硬件 iic 功能）， 需要用杜邦线接线，接线如下图：



6，下载运行

6.1 下载

请参考《BLE 入门与提高教程》。

7，测试

见视频演示：

用 CC254x 实现的旋转魔方视频演示（目前存在一点问题、反应不灵敏）：

http://cloud.video.taobao.com/play/u/51731980/p/1/e/6/t/1/30385925.mp4

在我们 CC26x0 平台上的反应是灵敏的，见以下视频：

http://cloud.video.taobao.com/play/u/51731980/p/1/e/6/t/1/29791787.mp4

8，联系我们

QQ 群：257318688
QQ: 11940507
Tel： 18588220515 阿莫
网站支持： www.AmoMcu.com 阿莫单片机社区网
淘宝店铺： http://amomcu.taobao.com
公司地址： 深圳市宝安区宝安电子数码城 4 楼 4F10