# 一、IIO子系统简介

以前驱动编写方法所带来的缺点

以前的写法，就是驱动人员自己定义传感器数据的上报格式，自己定义的格式，那么就只有自己知道，自己编写应用。缺点就是不灵活，封闭。没有一个标准，不统一

理想的驱动应该是：

1、黑匣子，应用直接能读取到，而且人性化。

## 1.1、为什么使用IIO子系统

 为了规范化日益庞大的ADC/DAC类传感器

 使用IIO子系统系统以后，会在跟文件系统得到一大堆文件：



 我们得到in\_accel\_z\_raw是ICM20608加速度Z轴原始值。对应的g是？

位数始终是16位ADC，测量范围不同，每一位代表的实际g值就不同。



 我需要知道加速度计实际值得时候，只需要将in\_accel\_x\_raw\*in\_accel\_scale。

## 1.2、iio\_dev结构体

 1、申请

使用iio\_dev来表示一个具体的iio设备。devm\_iio\_device\_alloc或iio\_device\_alloc申请iio\_dev

 2、注册iio\_dev

 iio\_device\_register

 1、通道：

 表示传感器的每一个测量通道，

## 1.3、iio\_info

核心

 icm20608\_read\_raw

icm20608\_write\_raw

icm20608\_write\_raw\_get\_fmt

static int bma180\_read\_raw(struct iio\_dev \*indio\_dev,

 struct iio\_chan\_spec const \*chan, int \*val, int \*val2,

 long mask)

## 1.4、iio\_chan\_spec

 IIO设备通道重要的成员变量

 1、type

 2、modified和channel2

 3、address

 4、scan\_type

 5、info\_mask\_separate

 6、info\_mask\_shared\_by\_type

# 二、IIO驱动程序编写

# 三、应用编写



这些文件读到都是字符串

“0.000488281”，需要将字符串转换为实际的数值

可是使用atof将浮点字符串转换为实际的浮点数据，atoi就是完成整数的转换