

MEDIATEK

Sensor All In One

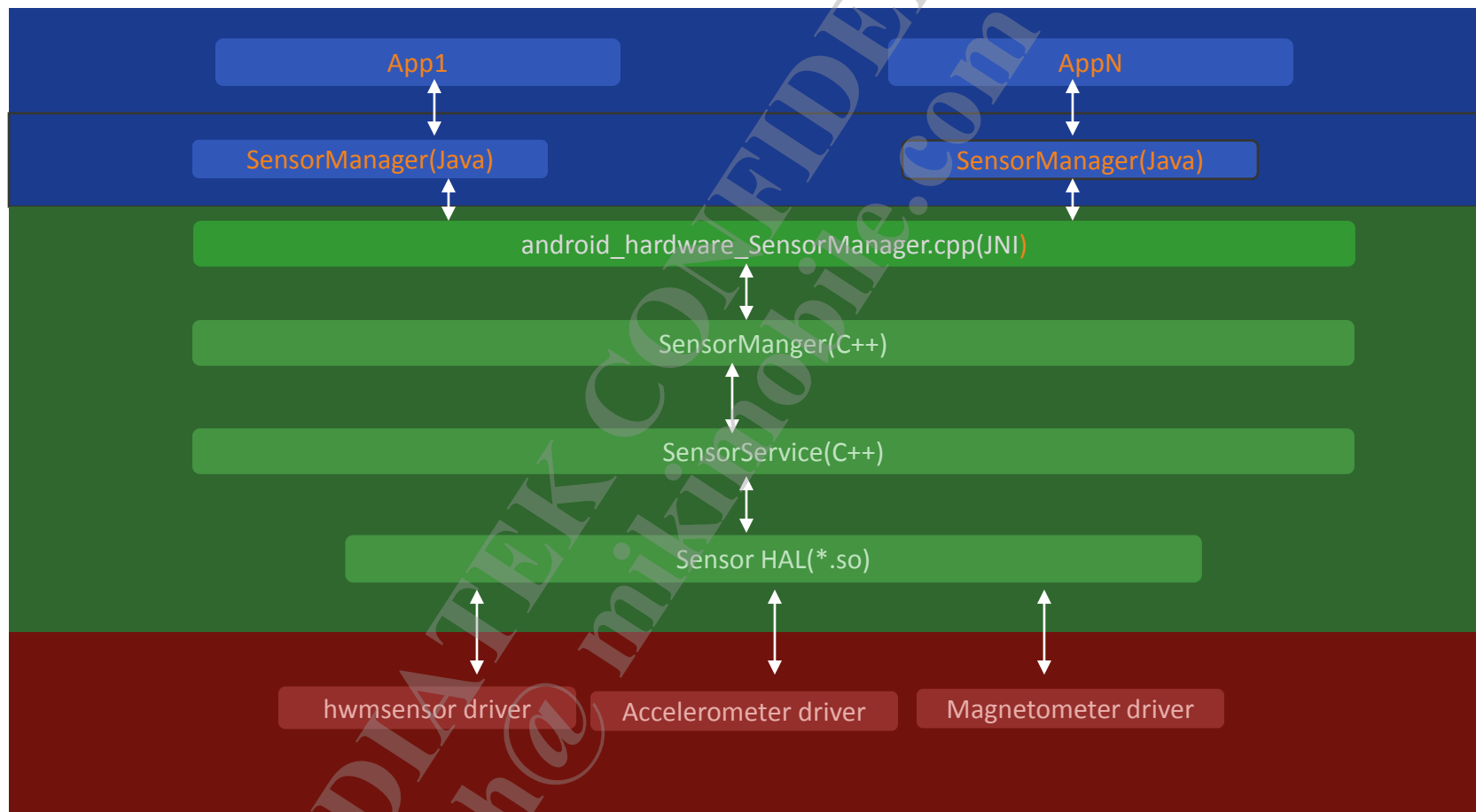
Outline

- Overview
- Sensor System Architecture
- OLD ARCTECTURE (旧架构)
- NEW ARCTECTURE (新架构)
- FACTORY MODE (工厂模式)
- Gsensor
- Msensor
- ALSPS
- L版本 Driver Porting Guide
- Test Apk

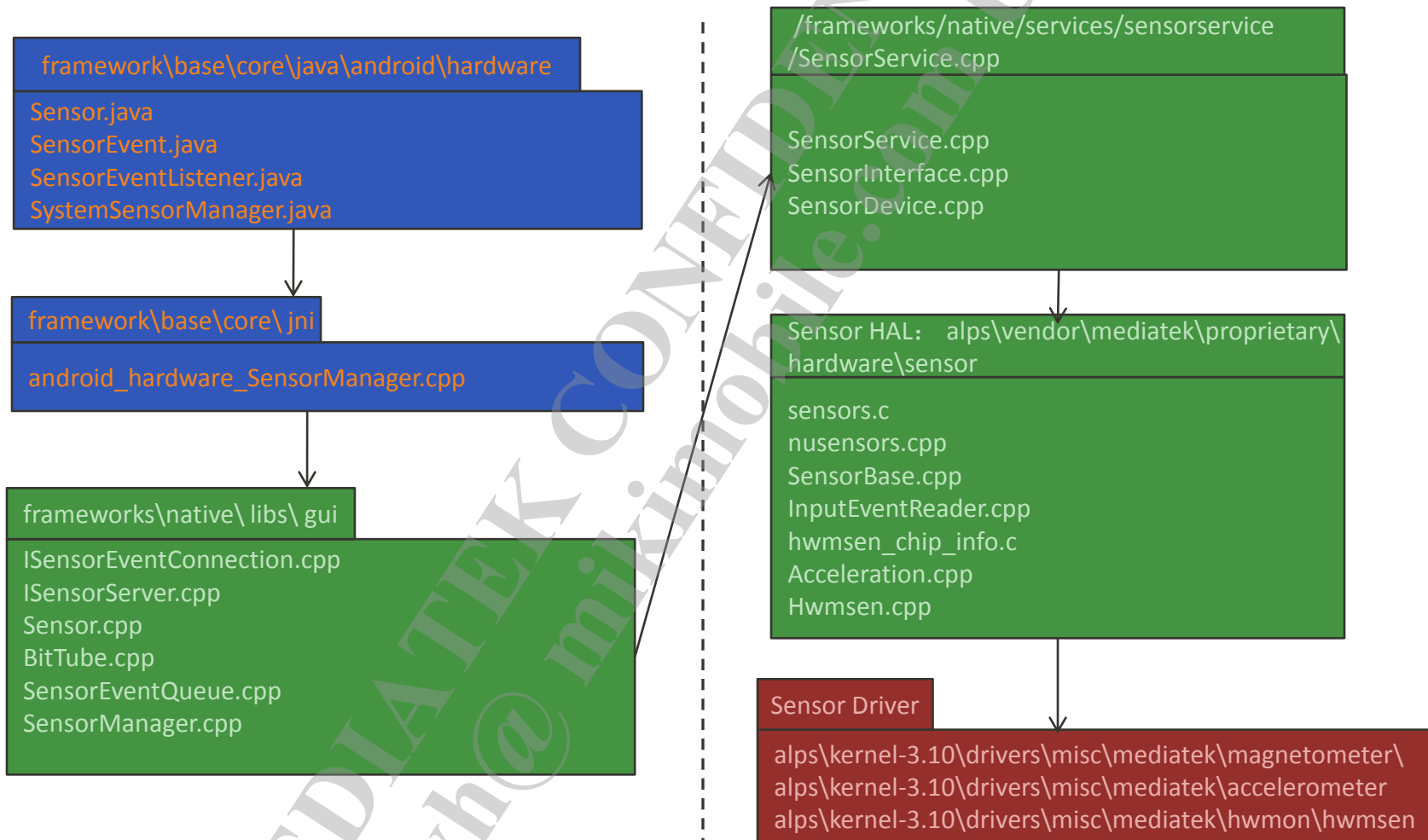
MEDIATEK

Sensor System Architecture

Sensor System Architecture



Sensor System Architecture



MEDIATEK

Sensor New Architecture

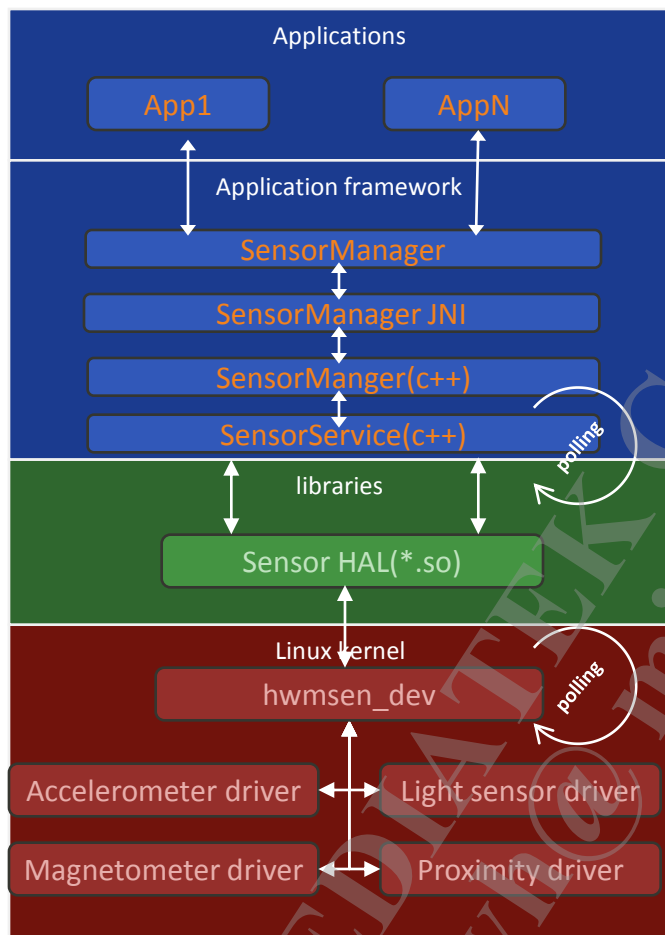
Agenda

- Sensor New Architecture Overview
- HAL New Architecture Introduction
 - Architecture description
 - APIs description
- Kernel New Architecture Introduction
 - Architecture description
 - APIs description
- Attachment

MEDIATEK

Sensor New Architecture Overview

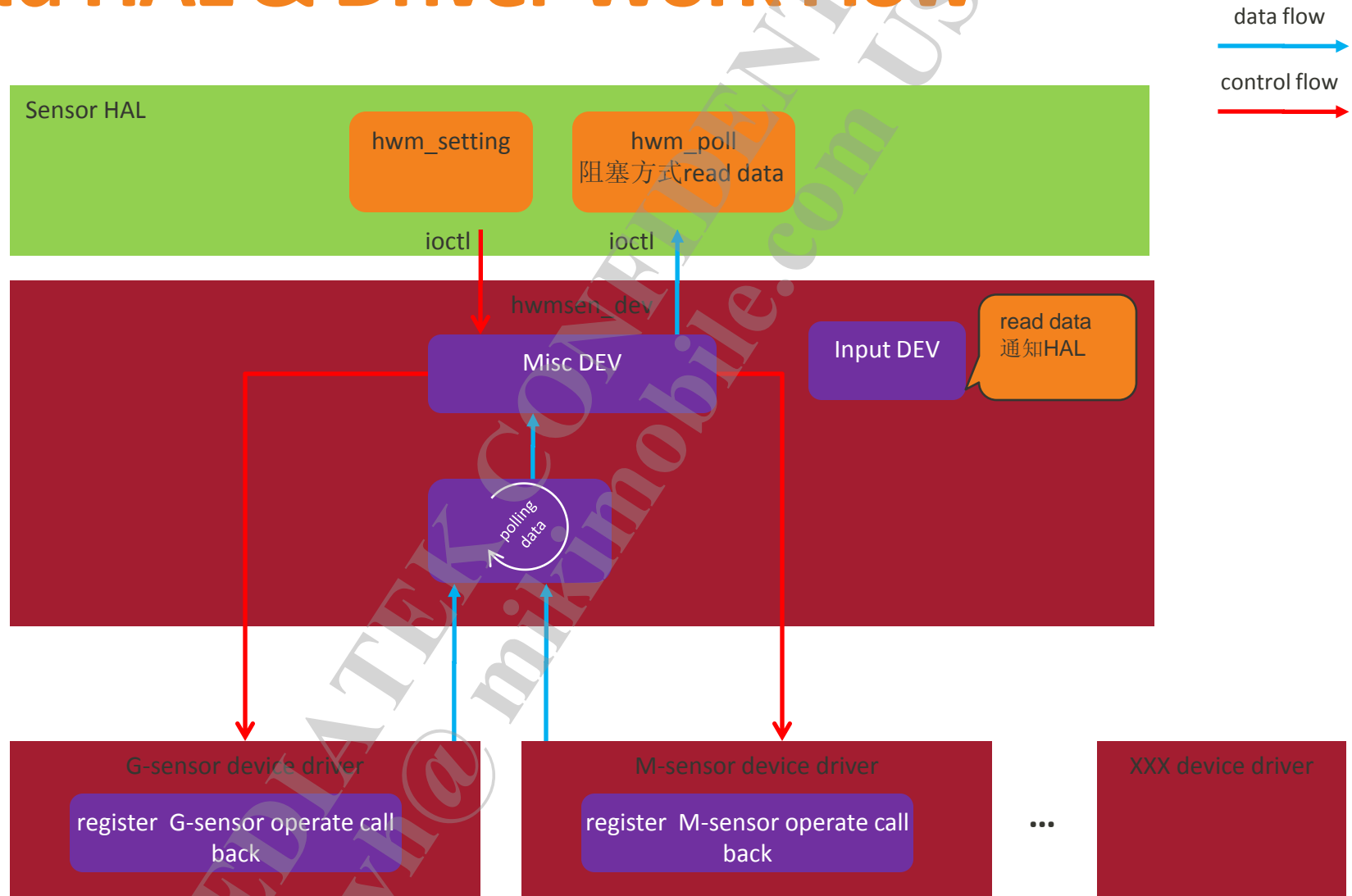
Old Sensor Structure



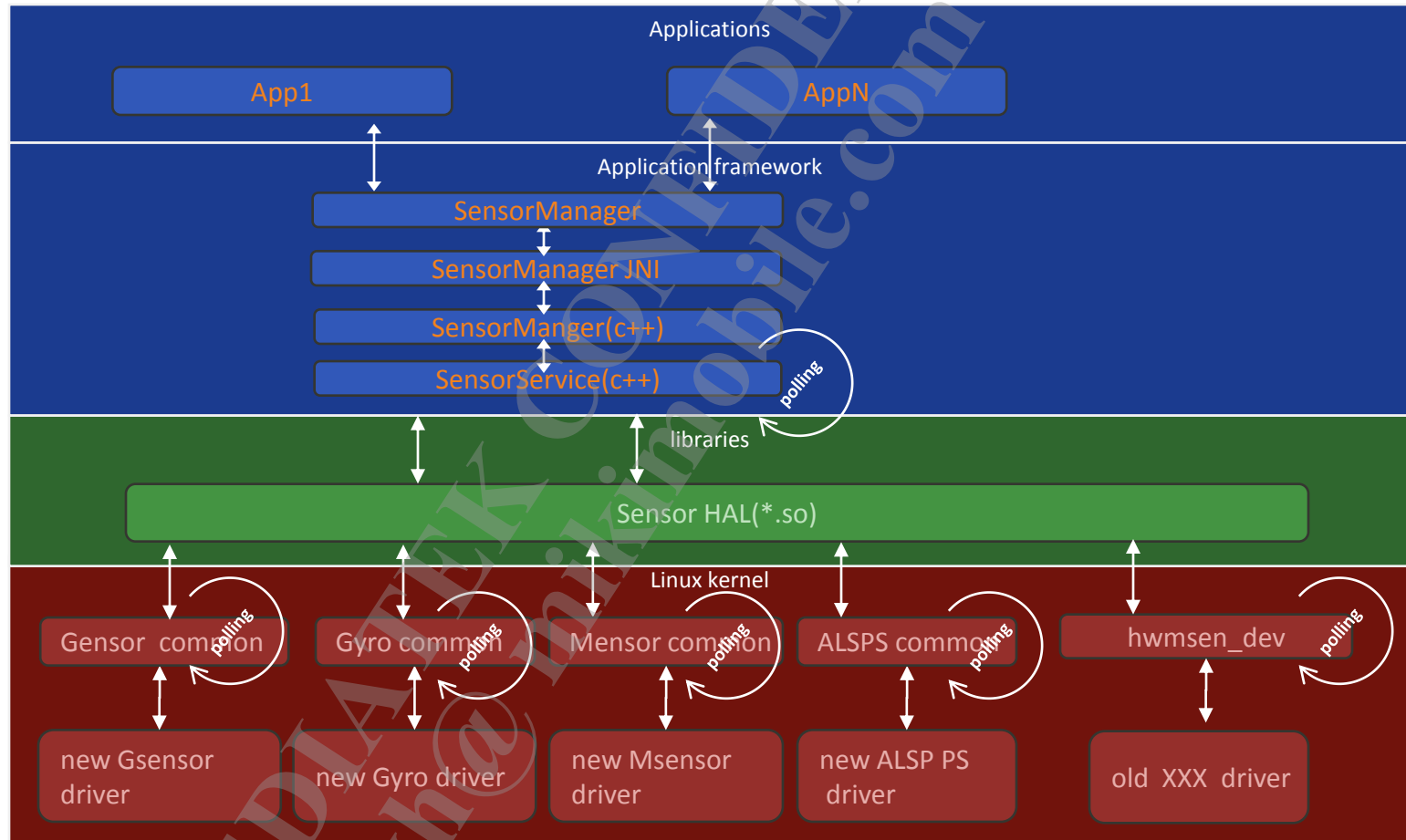
缺点:

- 1、所有sensor数据统一处理无法分别设置 report rate
- 2、其客户的sensor driver 无法直接应用到MTK solution
- 3、sensor 厂商都采用Google suggest 架构，提供support，对MTK 需单独修改架构 effort 较大

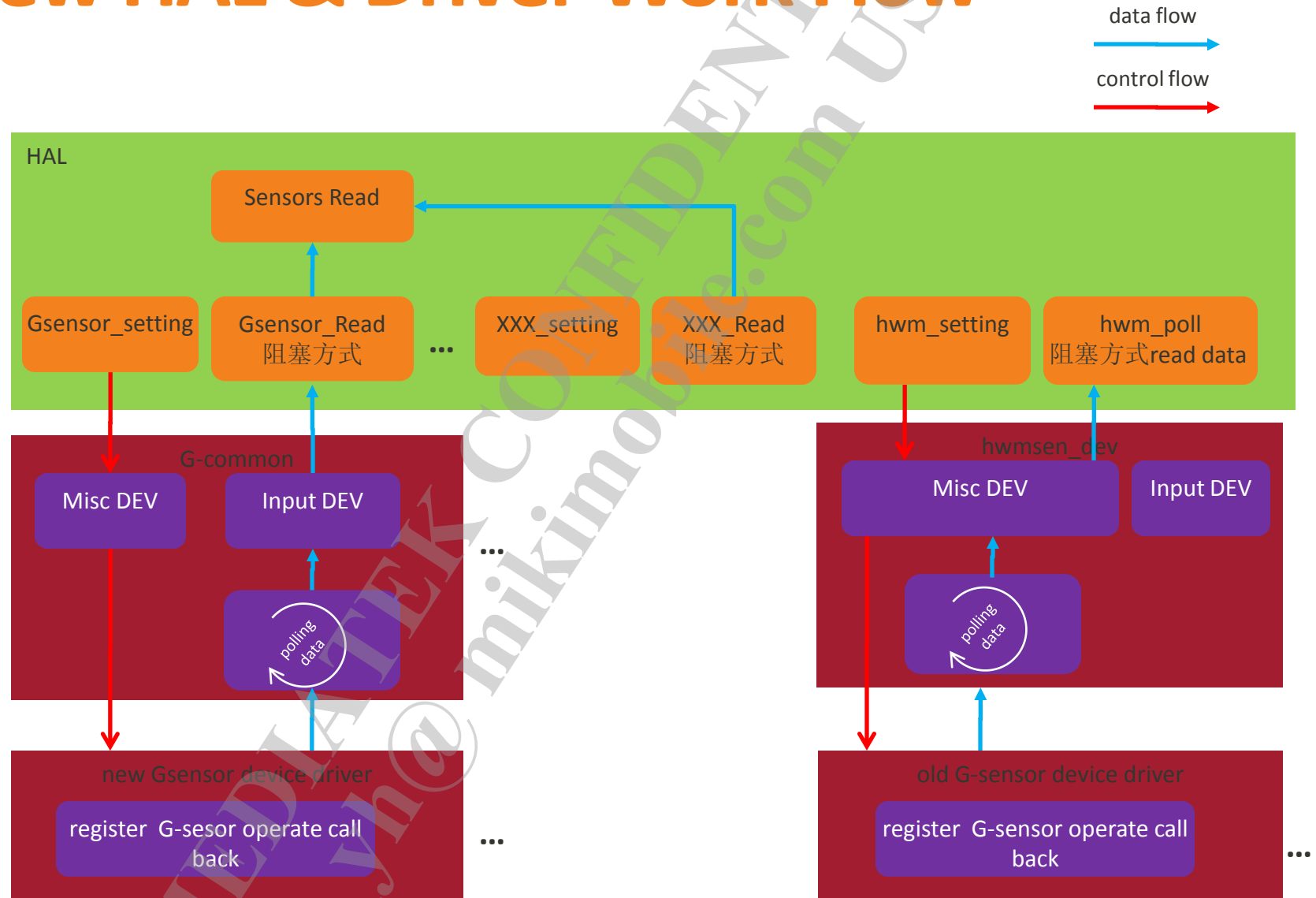
Old HAL & Driver Work Flow



New Sensor Structure



New HAL & Driver Work Flow



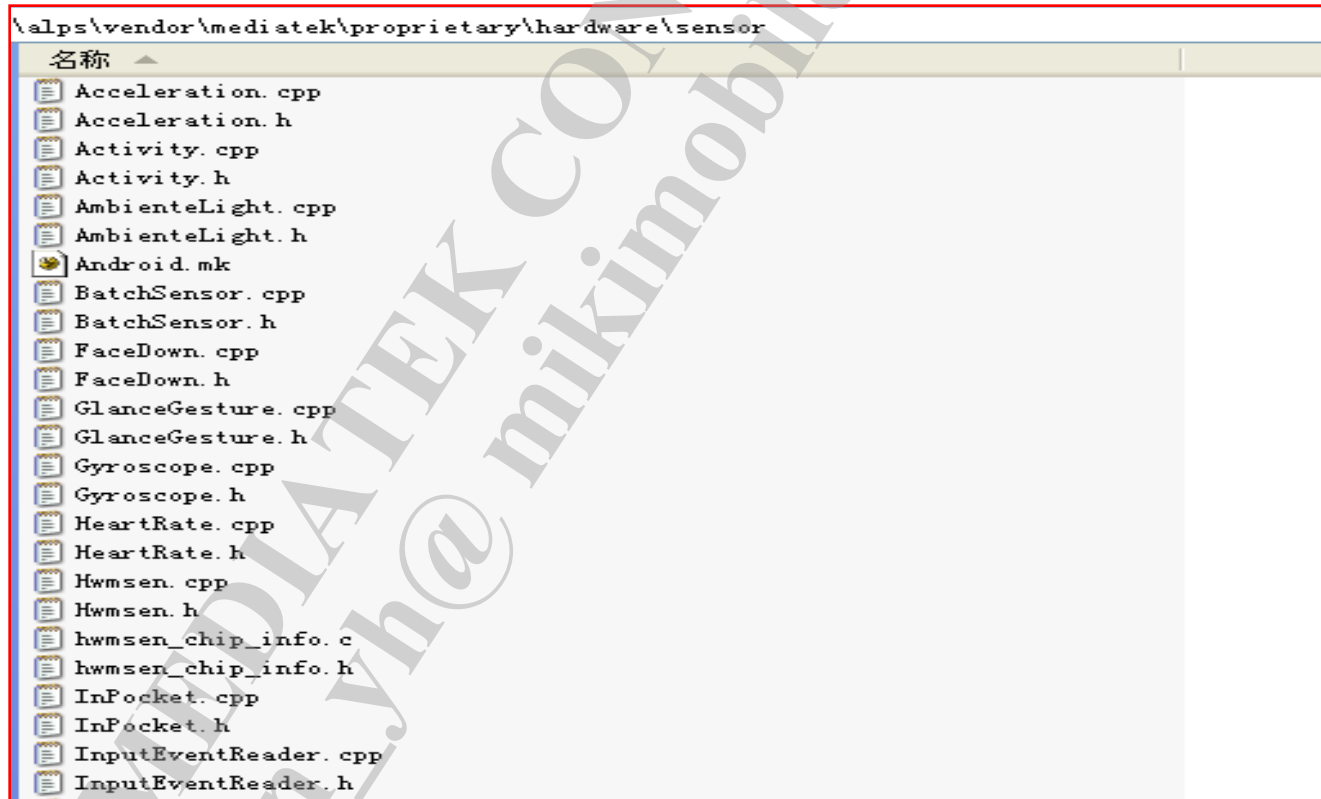
MEDIATEK

HAL New Architecture Introduction

New HAL Detail Design

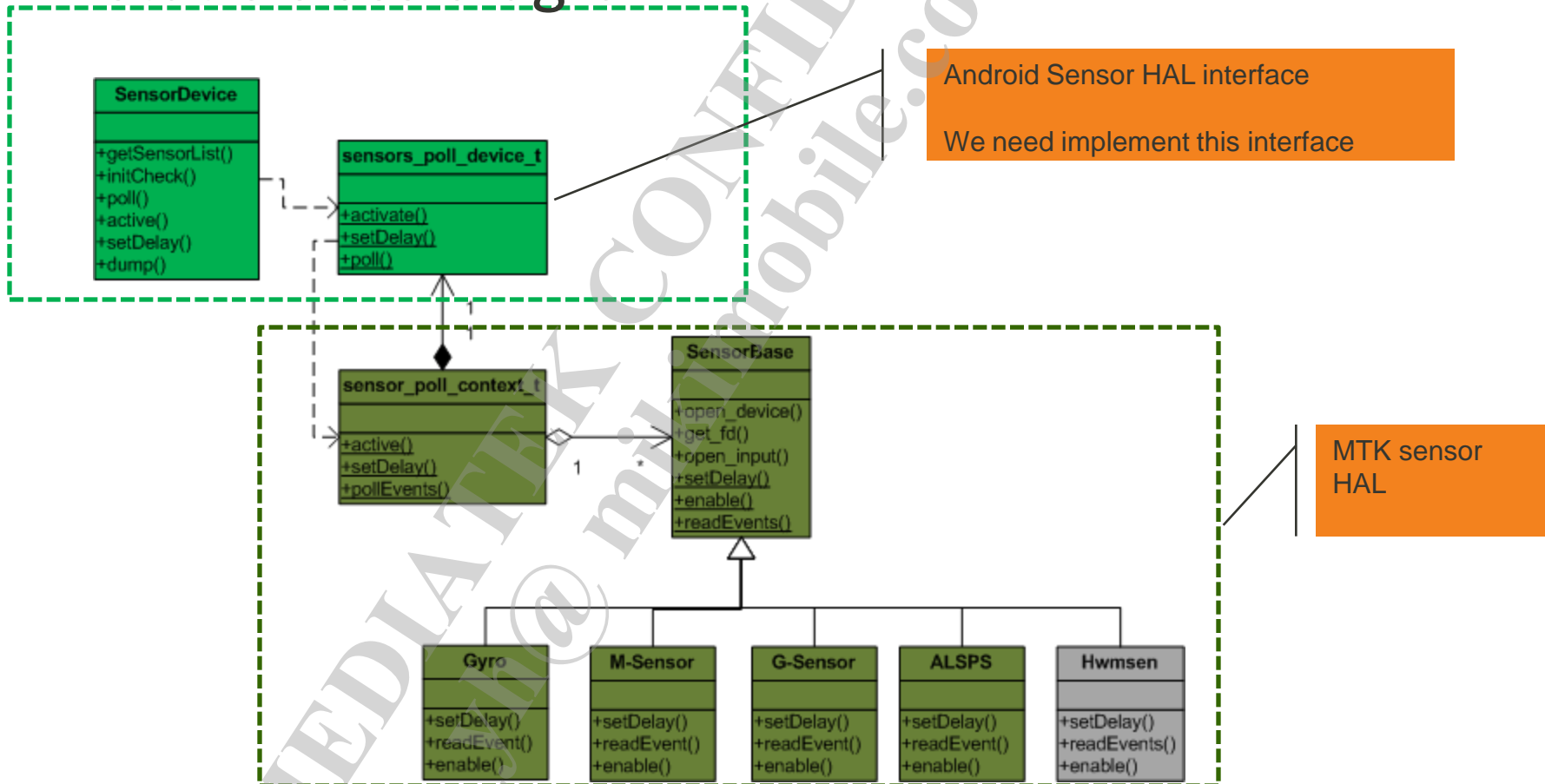
Code path说明:

- 1、新架构的HAL层仍位于原HAL层code路径
alps\vendor\mediatek\proprietary\hardware\sensor
- 2、新架构中的每个sensor都对应于一个CPP file



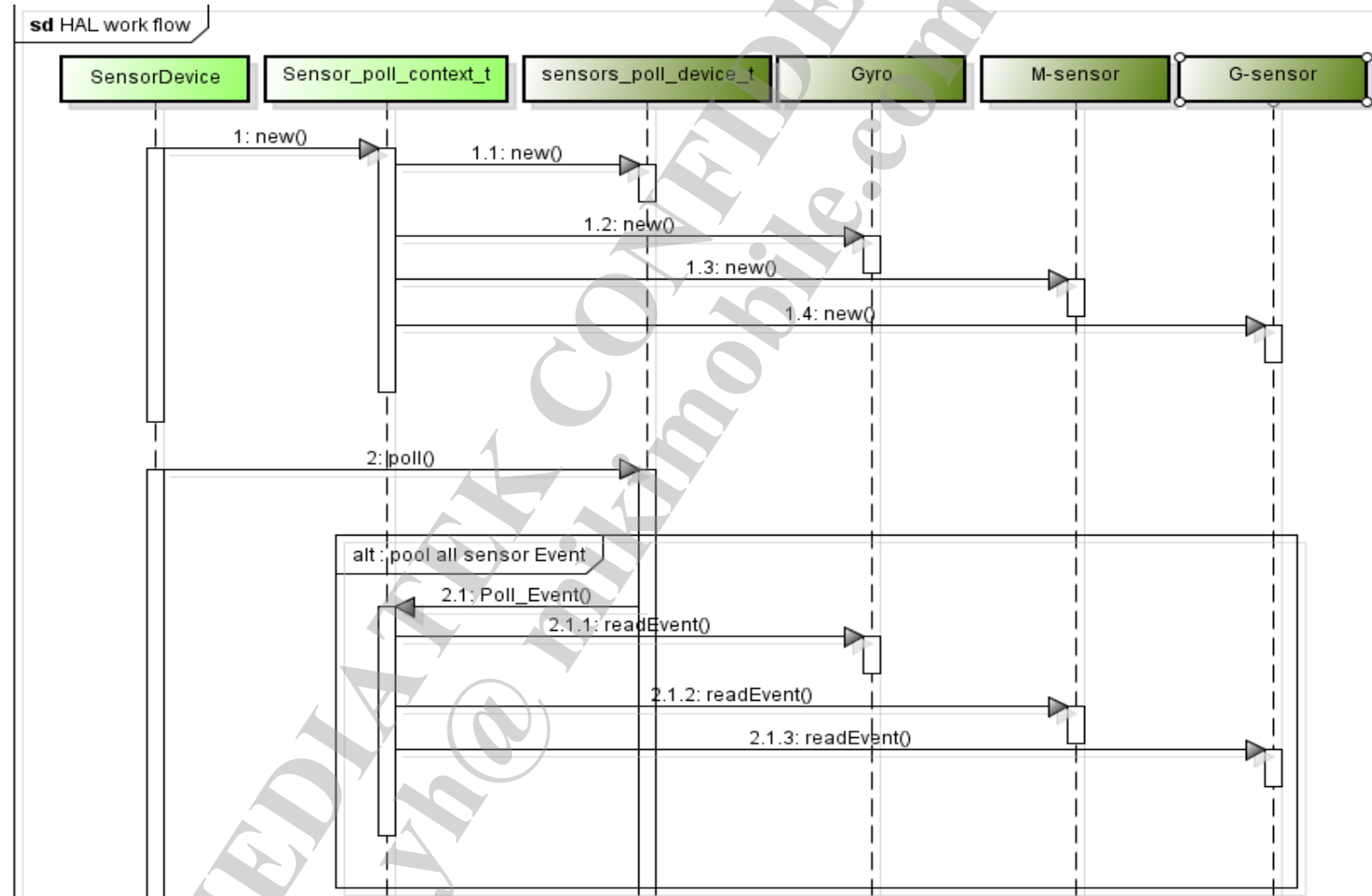
New HAL Detail Design

■ static class diagram



New HAL Detail Design

- HAL sequence program



New HAL Detail Design

- HAL层data flow以及control flow的新老架构对比情况

	新架构	老架构
Data flow	Input event	ioctl
Control flow	Misc device attribute file	ioctl

New HAL API Description

- Attribute file path sys/class/misc/
- HAL control & data interface
 - G-sensor common
 - Misc dev: "m_acc_misc"
 - Input dev: "m_acc_input"
 - M-Sensor Common
 - Misc dev: "m_mag_misc"
 - Input dev: "m_mag_input"
 - Gyro-Sensor Common
 - Misc dev: "m_gyro_misc"
 - Input dev: "m_gyro_input"
 - Light-Sensor Common
 - Misc dev: "m_light_misc"
 - Input dev: "m_light_input"
 - Proximity-Sensor Common
 - Misc dev: "m_prox_misc"
 - Input dev: "m_prox_input"
 - hwmsen-dev
 - Misc dev: "hwmsensor"
 - Input dev: "hwmdata"



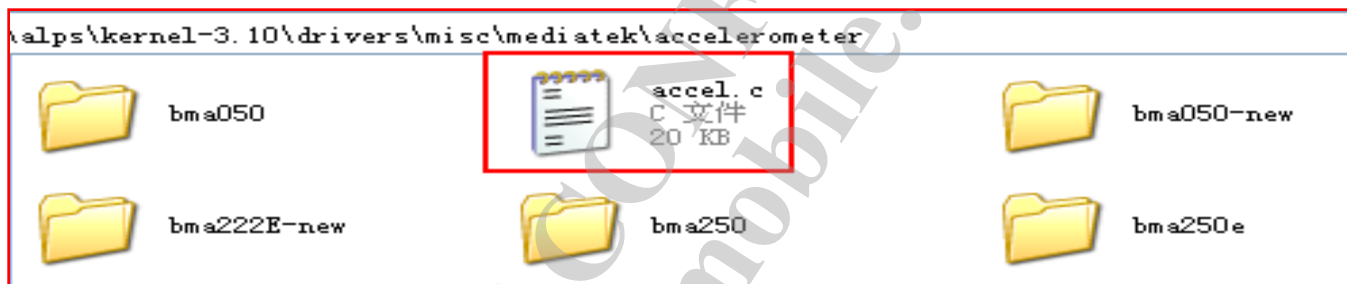
MEDIATEK

Kernel New Architecture Introduction

New Kernel Driver Design Detail

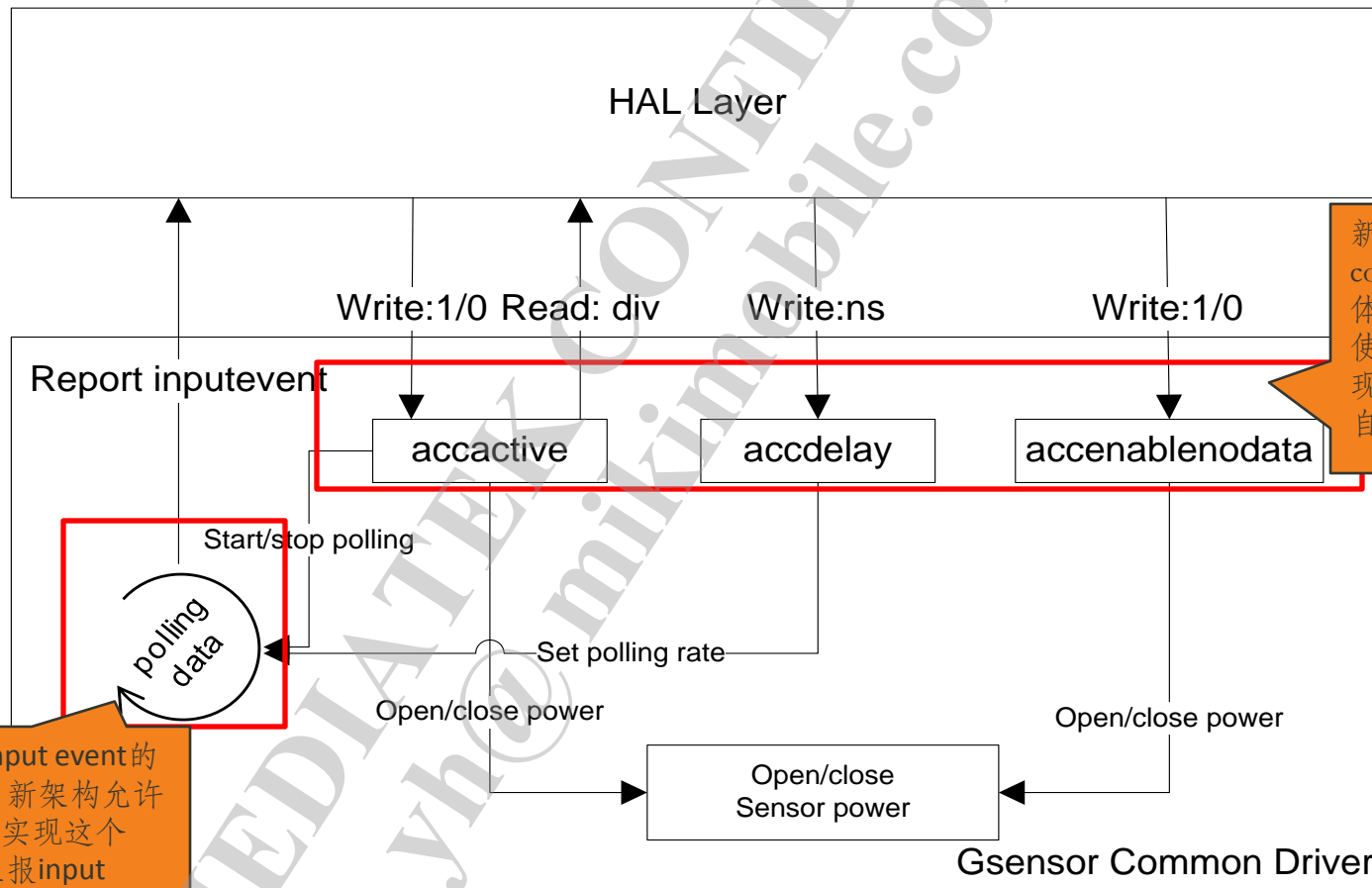
Code path说明:

1、新架构的Kernel层common driver layer为用于下图中的路径中



New Kernel Driver Design Detail

架构说明：下图为我们所期望的single sensor driver的一般结构



New Driver API Description-Gsensor

API Name	Parameter description
acc_driver_add(struct acc_init_info* obj)	acc_init_info包含了sensor的相关信息，通过把这个结构体注册到common driver实现sensor auto detect的功能
acc_register_data_path(struct acc_data_path *data)	acc_data_path中包含了一个获取sensor数据的function和一个用于归一化不同厂商sensor raw data的算术值，这两个数据将被注册到common sensor driver的架构中去，为MTK架构获取sensor数据
acc_register_control_path(struct acc_control_path *ctl)	acc_control_path 包含了3个function和一个布尔型参数，用于控制sensor的enable/disable， setDelay等

New Driver API Description-Gsensor

Struct Name	Parameter	Description
acc_data_path	int (*get_data)(int x, int y, int z, int status)	用于挂载single sensor driver的用来获取sensor数据的function，x，y，z代表三轴上的数据，status表示数据精度
	int vender_div	用于归一化不同厂商sensor raw data的算术值，raw data除以这个数据得到以m/s^2为单位的一个数据
acc_control_path	Int (*open_report_data)(int open)	如果single driver使用自己work queue或者其他方式上报input event给MTK platform，则在这个function中去实现开启上报input event的相关功能，参数open：1表示开启此功能，0表示关闭此功能
	Int (*enable_nodata)(int en)	用于实现开启sensor power的功能，参数en：1表示设置sensor为measure mode，0表示设置sensor为suspend mode
	Int (*set_delay)(u64 delay)	用于设置sensor的polling rate，参数delay的单位为ns(纳秒)
	bool is_report_input_direct	用于提供给MTK driver 判定single driver是否自己上报input event
acc_init_info	char *name	当前sensor chip的名称
	int (*init)(void)	调用i2c_add_driver来加载对应sensor的i2c driver
	int (*uninit)(void)	调用i2c_del_driver来删除对应sensor的i2c driver
	struct platform_driver* platform_driver_addr	用于关联platform driver以便于attribute file的加载

New Driver API Description

- About div

- Div用于将sensor data的单位进行归一化处理
- Div将被HAL层用来做除数，HAL层将从input event中读到的数据除以div，得到归一化单位的sensor数据
- 因此，对于third party driver来说，其上报的div必须和上报的数据相匹配，这两组数据通过除法得到android所规定的每种sensor的标准数据单位
- 对于新架构中的Gsensor来说，我们的common driver上报mg(毫g)单位，同时上报div=1024，这两个数据在HAL进行除法运算得到以 m/s^2 为单位的加速度值

New Driver API Description

- 对于driver上报的input event的数据格式也必须遵循以下的原则：

Gsensor

```
00030: #define EVENT_TYPE_ACCEL_X ABS_X
00031: #define EVENT_TYPE_ACCEL_Y ABS_Z
00032: #define EVENT_TYPE_ACCEL_Z ABS_Y
00033: #define EVENT_TYPE_ACCEL_STATUS ABS_WHEEL
```

Msensor

```
#define EVENT_TYPE_MAGEL_X ABS_X
#define EVENT_TYPE_MAGEL_Y ABS_Y
#define EVENT_TYPE_MAGEL_Z ABS_Z
#define EVENT_DIV_MAGEL ABS_RUDDER
#define EVENT_TYPE_MAGEL_STATUS ABS_WHEEL

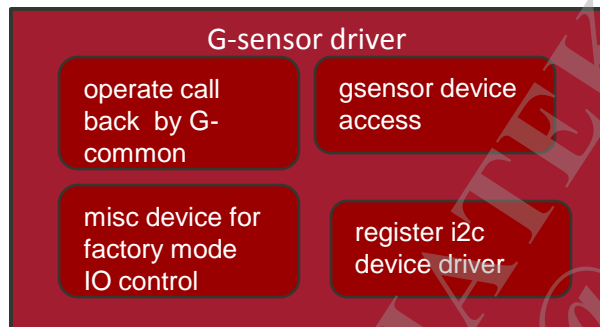
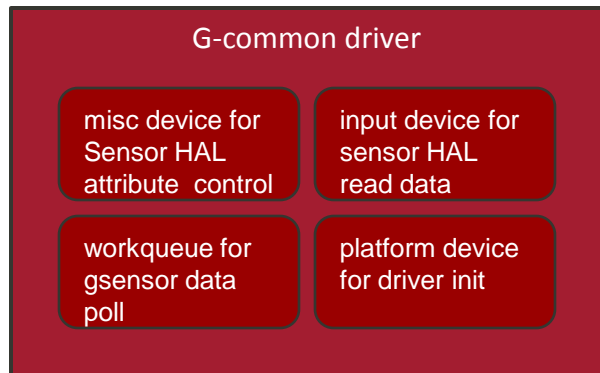
#define EVENT_TYPE_O_X ABS_RX
#define EVENT_TYPE_O_Y ABS_RY
#define EVENT_TYPE_O_Z ABS_RZ
#define EVENT_DIV_O ABS_GAS
#define EVENT_TYPE_O_STATUS ABS_THROTTLE
```

- 上报的input event的类型是定死的，MTK driver/third party driver都必须遵循这个准则，否则sensor无法上报数据

New Driver API Description

- 新的sensor架构在kernel layer的设计上支持三种连接方式：
 1. HAL + third party driver
 2. HAL + MTK common kernel driver + third party driver
 3. HAL + MTK common kernel driver + MTK new driver
- 相比较而言，我们更建议客户使用2，3两种搭配方式，因为这两种方式under MTK架构，便于我们更好的管控driver的行为

MTK Old Driver Porting to New Driver

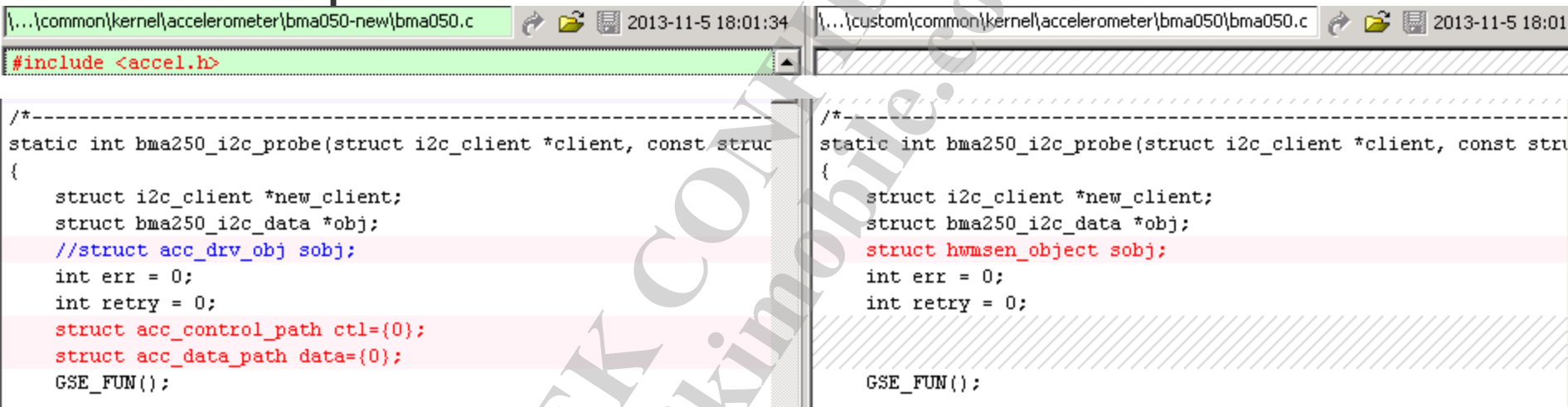


做法：

- 1、在不对原有driver进行较大改变的前提下进行
- 2、只需要进行一些call back function的注册即可

MTK Old Driver Porting to New Driver-Gsensor

■ Example



The image displays two side-by-side code editors comparing the `bma250_i2c_probe` function in an old driver (left) and a new driver (right). The left editor shows the original code with structures `acc_drv_obj` and `acc_control_path`. The right editor shows the ported code where these are replaced with `hwmsen_object` and `acc_data_path`. The file paths in the editors indicate the old driver is in a 'new' directory and the new driver is in a standard directory.

```
Left Editor (Old Driver):
File: [...]common\kernel\accelerometer\bma050-new\bma050.c
#include <accel.h>

/*-----
static int bma250_i2c_probe(struct i2c_client *client, const struct i2c_device_id *id)
{
    struct i2c_client *new_client;
    struct bma250_i2c_data *obj;
    //struct acc_drv_obj sobj;
    int err = 0;
    int retry = 0;
    struct acc_control_path ctl={0};
    struct acc_data_path data={0};
    GSE_FUN();
}

Right Editor (New Driver):
File: [...]custom\common\kernel\accelerometer\bma050\bma050.c
/*-----
static int bma250_i2c_probe(struct i2c_client *client, const struct i2c_device_id *id)
{
    struct i2c_client *new_client;
    struct bma250_i2c_data *obj;
    struct hwmsen_object sobj;
    int err = 0;
    int retry = 0;
    GSE_FUN();
}
```

MTK Old Driver Porting to New Driver-Gsensor

■ Example

```
if((err = bma250_create_attr(&(bma050_init_info.platform_driver)
{
    GSE_ERR("create attribute err = %d\n", err);
    goto exit_create_attr_failed;
}

ctl.open_report_data= bma050_open_report_data;
ctl.enable_nodata = bma050_enable_nodata;
ctl.set_delay = bma050_set_delay;
ctl.is_report_input_direct = false;

err = acc_register_control_path(&ctl);
if(err)
{
    GSE_ERR("register acc control path err\n");
    goto exit_kfree;
}

data.get_data = bma050_get_data;
data.vender_div = 1000;
err = acc_register_data_path(&data);
if(err)
{
    GSE_ERR("register acc data path err\n");
    goto exit_kfree;
}

if((err = bma250_create_attr(&bma250_gsensor_driver.driver))
{
    GSE_ERR("create attribute err = %d\n", err);
    goto exit_create_attr_failed;
}

sobj.self = obj;

sobj.polling = 1;
sobj.sensor_operate = gsensor_operate;
if((err = hwmsen_attach(ID_ACCELEROMETER, &sobj)))
{
    GSE_ERR("attach fail = %d\n", err);
    goto exit_kfree;
}
```

如何实现两套架构兼容

■ 1. HAL层如何实现兼容

- HAL sensor_poll_context_t 会对所有种类的sensor 进行 pollEvent, 有Event 就上报无Event 就bypass

■ 2. kernel层如何实现兼容

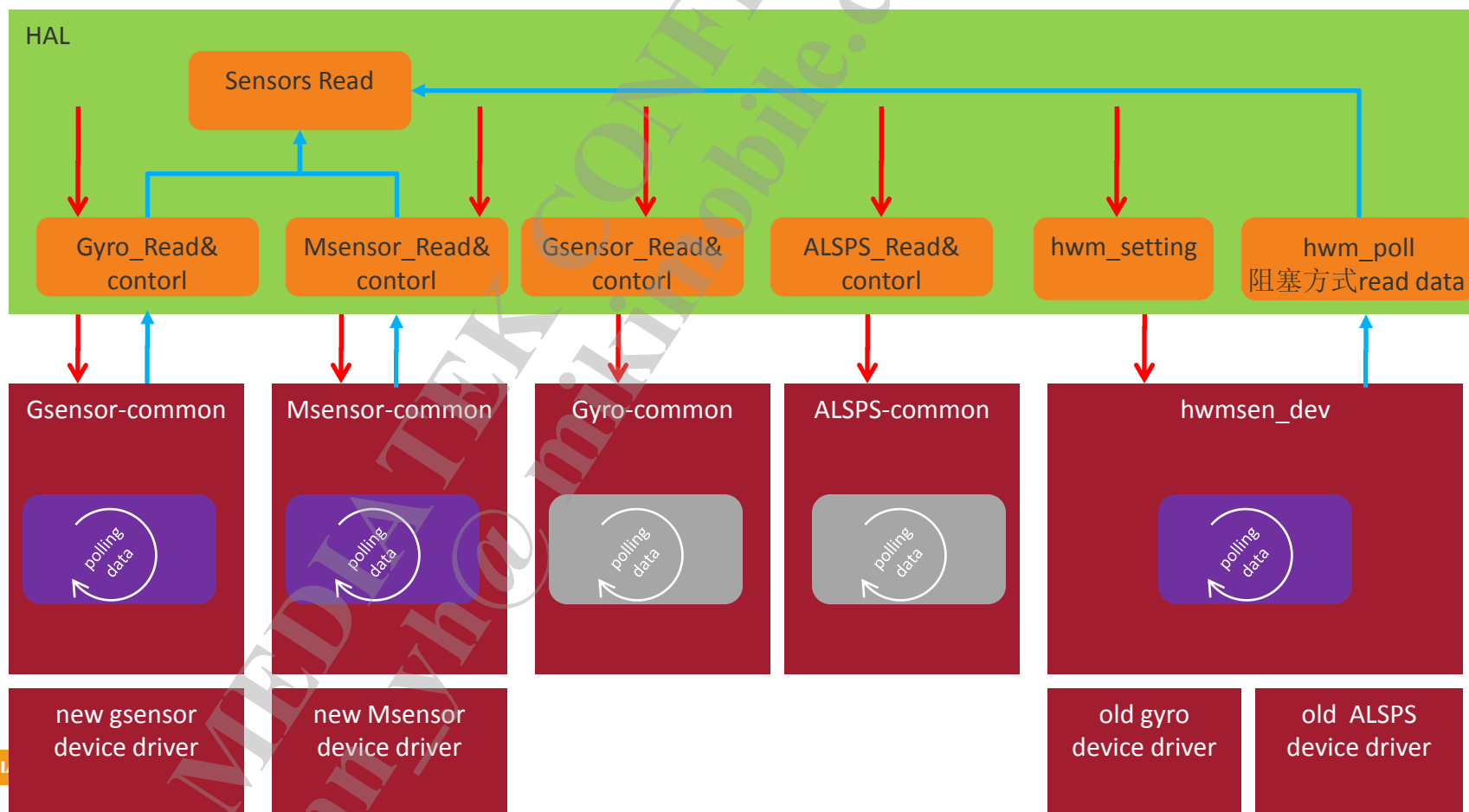
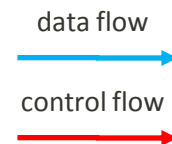
- 旧架构hwmsen_dev 根据注册的call back poll sensor, 有注册才polling
- 新架构XXX_common根据注册的call back poll sensor, 有注册才polling

■ 例子

- 使用旧架构编写的driver
 - Gyroscope
 - ALSPS
- 使用新架构编写的driver
 - Gsensor
 - Msensor

如何实现两套架构兼容

■ 例子执行flow

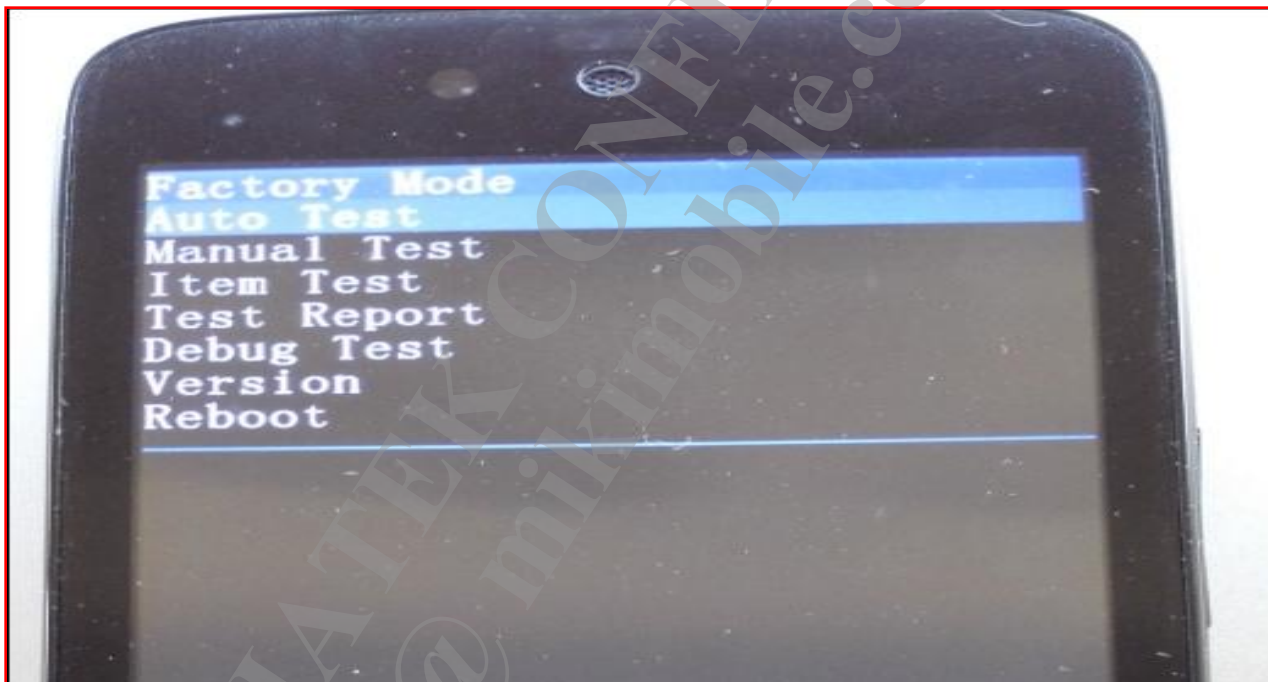


MEDIATEK

Sensor Factory Mode Test

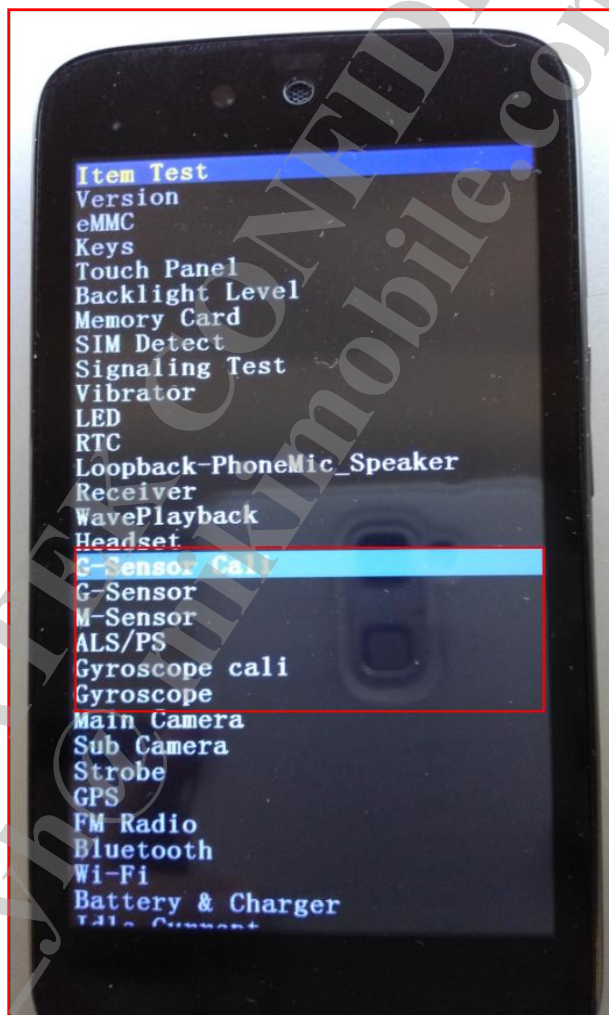
测试步骤

- 1. 音量下键与powerkey同时按下进入工厂模式，如下图



测试步骤

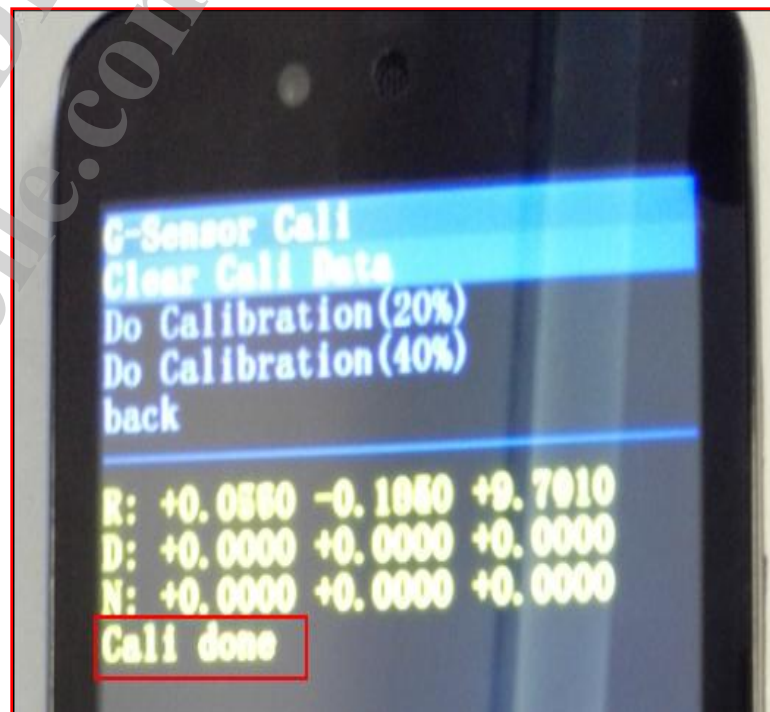
- 2. 点击Item Test进入如下界面，按音量下键选择相应的sensor测试类型
- 即可对相应sensor进行校准测试



测试步骤

- 3. 例如选择G-sensor Cali，首先clear cali data，然后做20%或者40%测试，若显示Cali done则表示校准成功

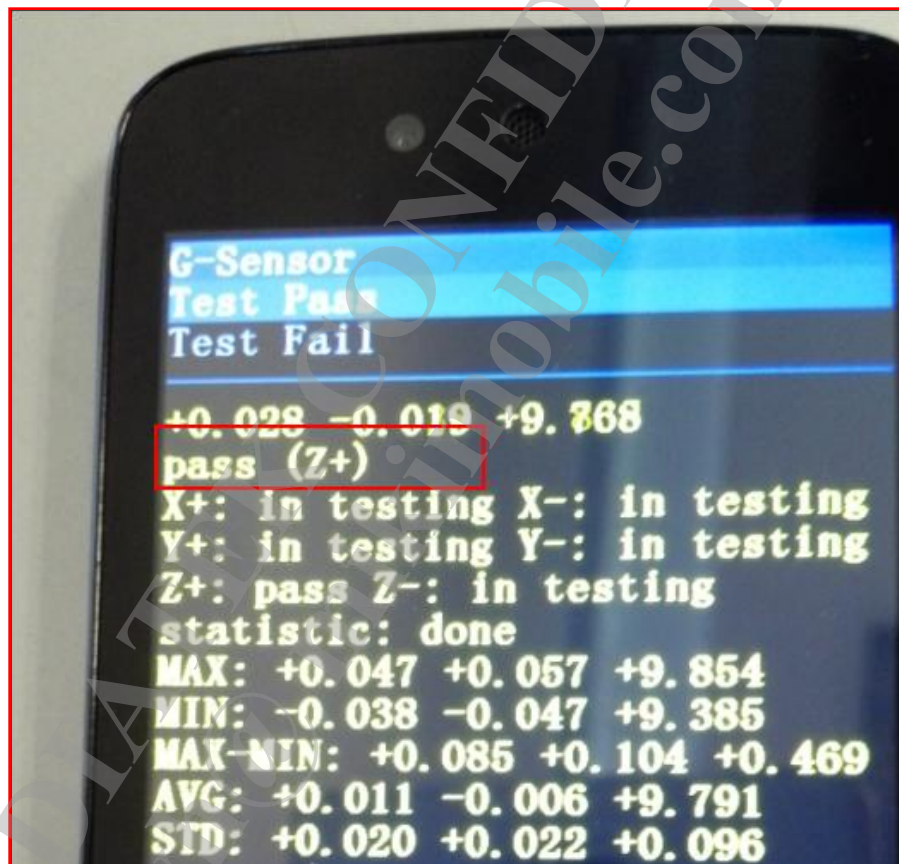
- As shown in the image, there are three strings:
 - R: +0.0560 -0.1050 + 9.7910
 - The real gravity data reported by sensor
 - D: -0.0000 +0.0000 -0.0000
 - The calibration data read from driver
 - N: -0.0000 +0.0000 -0.0000
 - The calibration data read from NVRAM
 - The calibration data will be saved NVRAM first.
 - And, it will be written to driver during booting and calibration.
 - Hence, the 2nd and 3rd value should be the same
- Functionality
 - Clear Calibration
 - Clear the calibration both in NVRAM & driver
 - Do calibration



- Press "Do Calibration(20)" to perform calibration with 20% tolerance
- Press "Do Calibration(40)" to perform calibration with 40% tolerance

测试步骤

- 4. 选择G-sensor 测试项测试，看校准后数据是否OK，若为pass则表示通过



Gsensor 校准流程

■ Gsensor校准

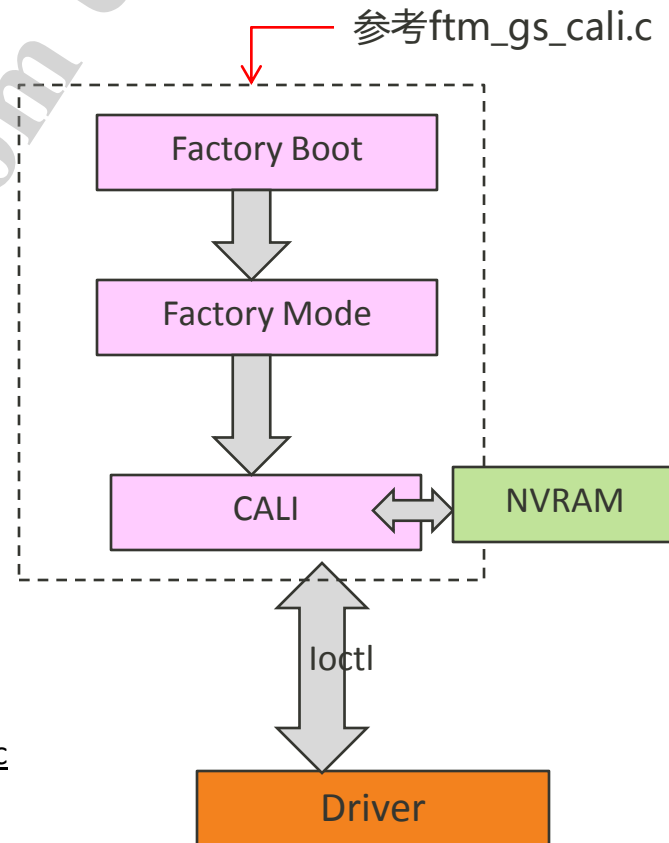
- 正常情况下，平放朝上时，x/y/z数值应该为0/0/9.8；如果hw受外力影响，会发生偏移，需要校准。
- 在工厂模式下，进入gsensor校准界面，手机静止平放朝上，选择校准即可。
- 如果数值偏移不大，即使不校准，对性能影响不大。

■ 校准过程

- 进入工厂模式后，单项测试中，选择 gsensor校准选项，平放手机，校准即可。
- 工模下校准流程，工模下的数据通过ioctl和driver层进行交互
 - 在第1步中进行校准后，通过ioctl读取20个数据，取平均做校准
 - 计算得到校准数据后，通过ioctl把校准的数据写入driver内
 - 同时把校准数据保存到nvrnm中
- 在做校准时，会把数据保存到nvrnm中；开机时，会先从nvrnm中读取之前校准的数据，然后通过ioctl把校准数据设置到driver，所以校准一次就可以实现sensor的使用。

另外GYROSCOPE的校准流程与Gsensor基本一样

可以参考 /vendor/mediatek/proprietary/factory/src/test/ftm_gyroscope.c
/vendor/mediatek/proprietary/factory/src/test/ftm_gyro_cali.c



MEDIA TEK

Accelerometer

Bma050 Driver Porting – Overview

- IOCTL interface is implemented for
 - Accessed by factory mode (factory test & calibration)
 - Accessed by meta mode (calibration)
- call back function is implemented for
 - Accel.c
- SYSFS interface is implemented for
 - Read raw data
 - Read remapped data
 - Adjust the filter length of low pass filter
 - Set debug trace
 - For msensor daemon
- power control
- driver customization

Bma050 Driver Porting – IOCTL (1/2)

■ IOCTL

Cmd	GSENSOR_IOCTL_INIT
Description	Initial bma050.
Arg	None
Return	0 : the operation succeeds; otherwise : the operation fails
Cmd	GSENSOR_IOCTL_READ_CHIPINFO
Description	Read chip information.
Arg	A string buffer to store chip information
Return	0 : the operation succeeds; otherwise : the operation fails
Cmd	GSENSOR_IOCTL_READ_SENSORDATA
Description	Read remapping sensor data. The sensor data is remapped to Android coordinate system
Arg	A string buffer to store remapped sensor data (printed as hex number).
Return	0 : the operation succeeds; otherwise : the operation fails
Cmd	GSENSOR_IOCTL_READ_RAW_DATA
Description	Read original sensor data.
Arg	A string buffer to store original sensor data (printed as hex number)
Return	0 : the operation succeeds; otherwise : the operation fails

Bma050 Driver Porting – IOCTL (2/2)

■ IOCTL

Cmd	GSENSOR_IOCTL_SET_CALI
Description	Set calibration data. The calibration data will be added to original calibration data because the current measurement is based on original calibration data
Arg	An integer array with 3 elements for setting calibration data
Return	0 : the operation succeeds; otherwise : the operation fails
Cmd	GSENSOR_IOCTL_GET_CALI
Description	Get calibration data.
Arg	An integer array with 3 elements for getting calibration data
Return	0 : the operation succeeds; otherwise : the operation fails
Cmd	GSENSOR_IOCTL_CLR_CALI
Description	Reset calibration data to all zero
Arg	None
Return	0 : the operation succeeds; otherwise : the operation fails

Bma050 Driver Porting -call back function

- how to use call back function

```
ctl.open_report_data= bma050_open_report_data;
ctl.enable_nodata = bma050_enable_nodata;
ctl.set_delay = bma050_set_delay;
ctl.is_report_input_direct = false;

err = acc_register_control_path(&ctl);
if(err)
{
    GSE_ERR("register acc control path err\n");
    goto ↓exit_kfree;
}

data.get_data = bma050_get_data;
data.vender_div = 1000;
err = acc_register_data_path(&data);
if(err)
{
    GSE_ERR("register acc data path err\n");
    goto ↓exit_kfree;
}
```

```
int acc_register_control_path(struct acc_control_path *ctl)
{
    struct acc_context *cxt = NULL;
    int err =0;
    cxt = acc_context_obj;
    cxt->acc_ctl.set_delay = ctl->set_delay;
    cxt->acc_ctl.open_report_data= ctl->open_report_data;
    cxt->acc_ctl.enable_nodata = ctl->enable_nodata;
```

```
int acc_register_data_path(struct acc_data_path *data)
{
    struct acc_context *cxt = NULL;
    int err =0;
    cxt = acc_context_obj;
    cxt->acc_data.get_data = data->get_data;
    cxt->acc_data.vender_div = data->vender_div;
```

Bma050 Driver Porting – SYSFS (1/2)

■ Attributes List

```
static DRIVER_ATTR(chipinfo, S_IWUSR | S_IRUGO, show_chipinfo_value, NULL);  
static DRIVER_ATTR(sensordata, S_IWUSR | S_IRUGO, show_sensordata_value, NULL);  
static DRIVER_ATTR(cali, S_IWUSR | S_IRUGO, show_cali_value, store_cali_value);  
static DRIVER_ATTR(firlen, S_IWUSR | S_IRUGO, show_firlen_value, store_firlen_value);  
static DRIVER_ATTR(trace, S_IWUSR | S_IRUGO, show_trace_value, store_trace_value);  
static DRIVER_ATTR(status, S_IRUGO, show_status_value, NULL);
```

- chipinfo
 - Read chip information
- sensordata
 - Remapped sensor data
- cali
 - Read calibration data (integer format)

Bma050 Driver Porting – SYSFS (2/2)

– firlen

- Read the current filter length of low pass filter (averaging filter)
- Write a number N to enable/disable low pass filter
 - < 32: enable averaging filter with FIR = N
 - 0: disable averaging filter

– Trace

- ADX_TRC_FILTER: bit mask for filter related log
- ADX_TRC_IOCTL: bit mask for printing remapped data

```
typedef enum {  
    ADX_TRC_FILTER    = 0x01,  
    ADX_TRC_RAWDATA   = 0x02,  
    ADX_TRC_IOCTL     = 0x04,  
    ADX_TRC_CALI      = 0x08,  
    ADX_TRC_INFO       = 0x10,  
} ADX_TRC;
```

Bma050 Driver Porting - Customization (1/3)

■ Change file list

File Name	Location
cust_acc.h	alps\kernel-3.10\drivers\misc\mediatek\mach\mt67XX\\${PROJ}\accelerometer\
cust_acc.c	alps\kernel-3.10\drivers\misc\mediatek\mach\mt67XX\\${PROJ}\accelerometer\

■ Customization item

– Overview

```
struct acc_hw {  
    int i2c_num;  
    int direction;  
    int power_id;  
    int power_vol;  
    int firlen;  
};
```

– i2c_num

- Customer can define the I2C number used by G-sensor
- The value could be defined as 0 ~ 2

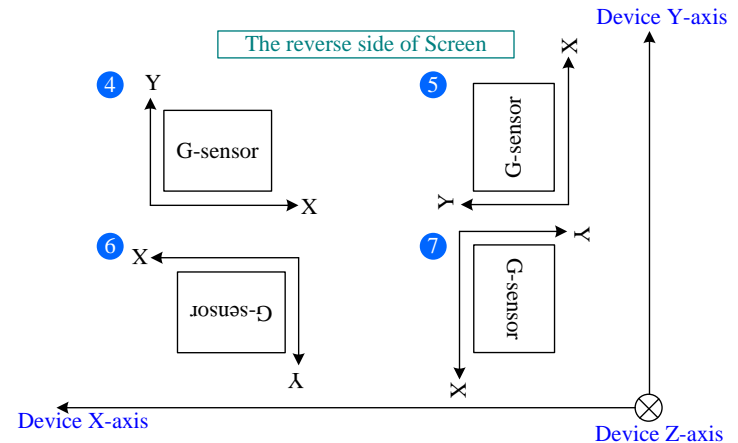
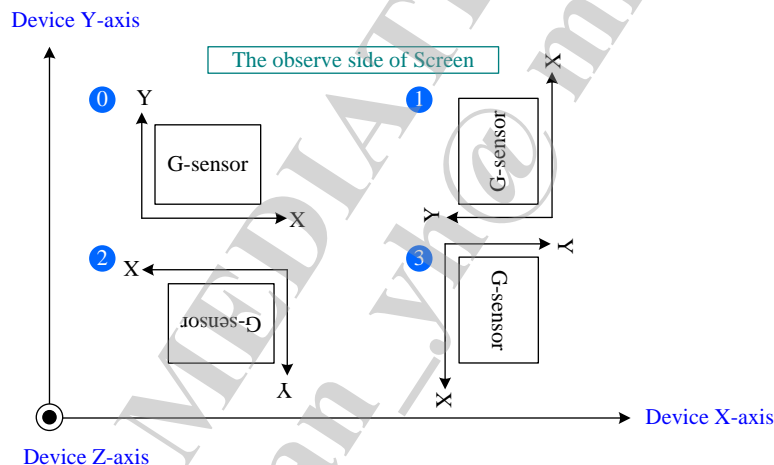
– firlen

- Customer can define the filter length of SW low pass filter.
- The value could be defined as 0 ~ 32. 0 will disable the functionality.

Bma050 Driver Porting - Customization (2/3)

- Customer can define the device direction of g-sensor in device.
- The value could be defined as 0 ~ 7

Value	Description
0	$\{x, y, z\} \Rightarrow \{x, y, z\}$
1	$\{x, y, z\} \Rightarrow \{-y, x, z\}$
2	$\{x, y, z\} \Rightarrow \{-x, -y, z\}$
3	$\{x, y, z\} \Rightarrow \{y, -x, z\}$
4	$\{x, y, z\} \Rightarrow \{-x, y, -z\}$
5	$\{x, y, z\} \Rightarrow \{y, x, -z\}$
6	$\{x, y, z\} \Rightarrow \{x, -y, -z\}$
7	$\{x, y, z\} \Rightarrow \{-y, -x, -z\}$



Bma050 Driver Porting - Customization (2/3)

■ Sensor方向 客制化

```
/*-----*/
static struct acc_hw cust_acc_hw = {
    .ifc_name = 0,
    .direction = 1,
    .power_id = MT65XX_POWER_NONE, /*!
    .power_vol= VOL_DEFAULT, /*!
    .firlen = 0, //old value 16
};

/*-----*/
struct hwmsen_convert map[] = {
    { { 1, 1, 1}, {0, 1, 2} },
    { {-1, 1, 1}, {1, 0, 2} },
    { {-1, -1, 1}, {0, 1, 2} },
    { { 1, -1, 1}, {1, 0, 2} },
    { {-1, 1, -1}, {0, 1, 2} },
    { { 1, 1, -1}, {1, 0, 2} },
    { { 1, -1, -1}, {0, 1, 2} },
    { {-1, -1, -1}, {1, 0, 2} },
};

/*-----*/
int hwmsen_get_convert(int direction, struct hwmsen_convert *cvt)
{
    if (!cvt)
        return -EINVAL;
    else if (direction >= sizeof(map)/sizeof(map[0]))
        return -EINVAL;

    *cvt = map[direction];
    return 0;
};

/*-----*/
if (!err = hwmsen_get_convert(obj->hw->direction, &obj->cvt))
    GSE_ERR("invalid direction: %d\n", obj->hw->direction);
```

obj

8个方向的数组，前面为正负号，后面为坐标映射

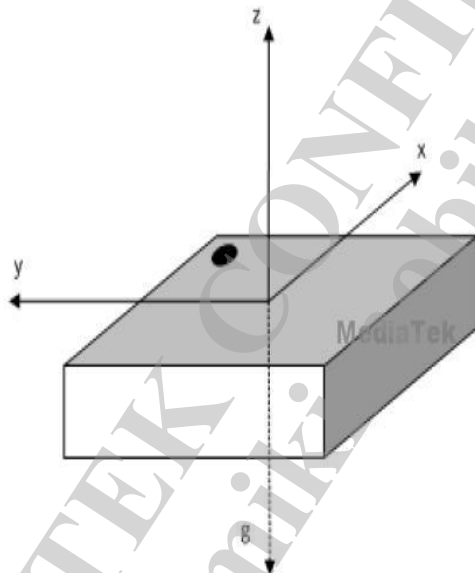
acc[] = {100, 0, 100}
sign[] = {1, -1, 1};
map[] = {1,0,2};

acc_x = sign[0]*acc[map[0]]
acc_y = sign[1]*acc[map[1]]
acc_z = sign[2]*acc[map[2]]

MEDIATEK
INTERNAL USE

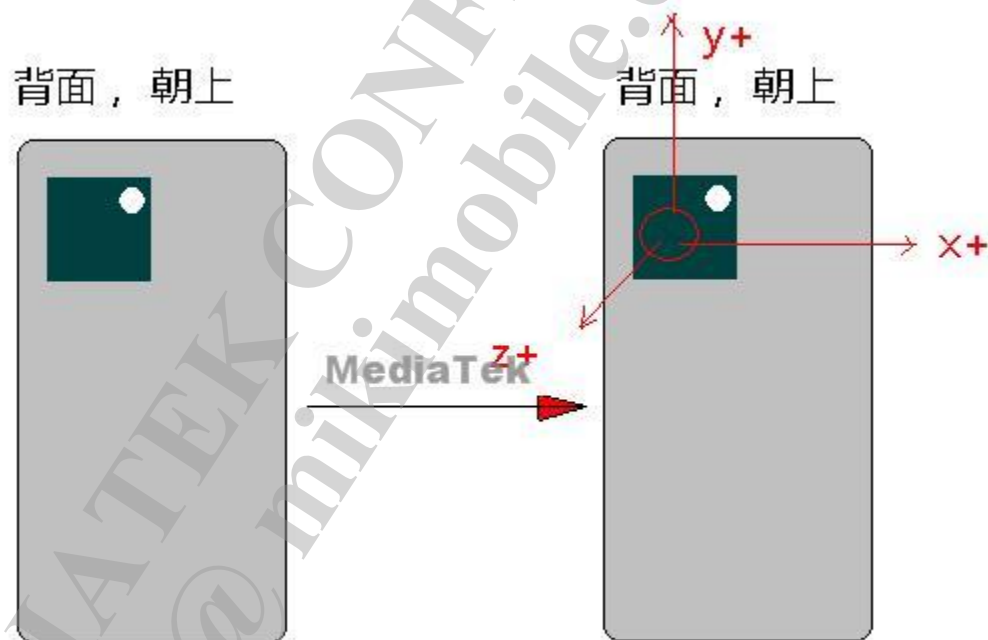
Example

1. 首先查阅sensor的spec，看spec中对坐标系的定义（IC上的圆点为参照原点）
举例，假设spec中定义如下



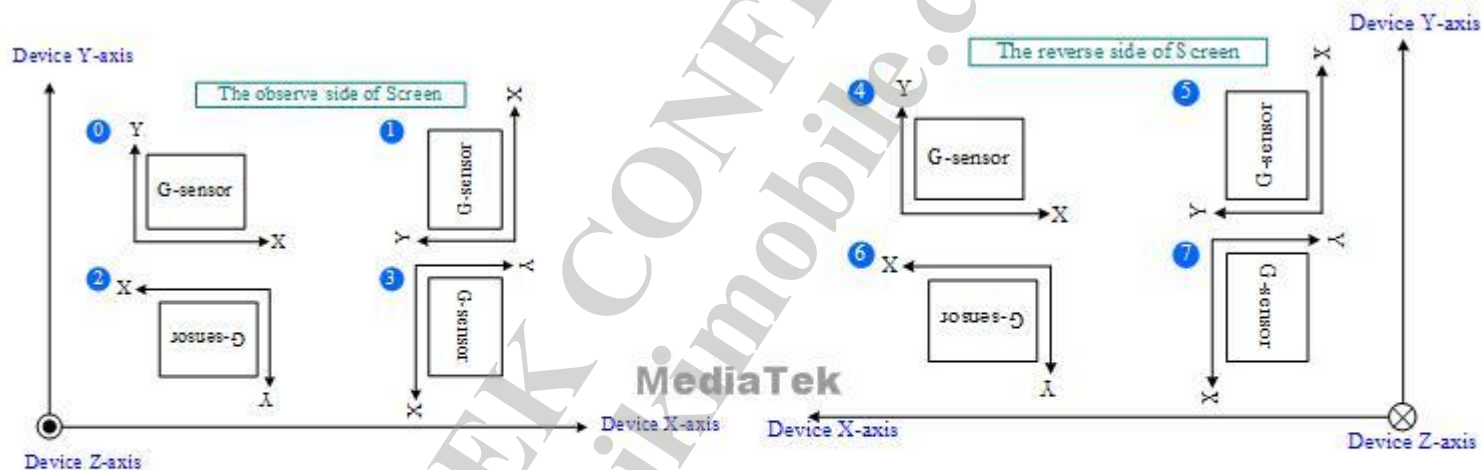
Example

- 2. 查看手机上IC的贴片方向，假设贴片方向如下图左侧，IC贴在背面，圆点在右上角，那么我们可以结合spec上的坐标系定义，画出手机上的坐标系，见下图右侧



Example

3. 根据上个步骤画出来的手机上的坐标系，对比下面的坐标系与软件定义的方向的对应关系，即可知道direction的取值在这个例子中 $\text{direction} = 4$

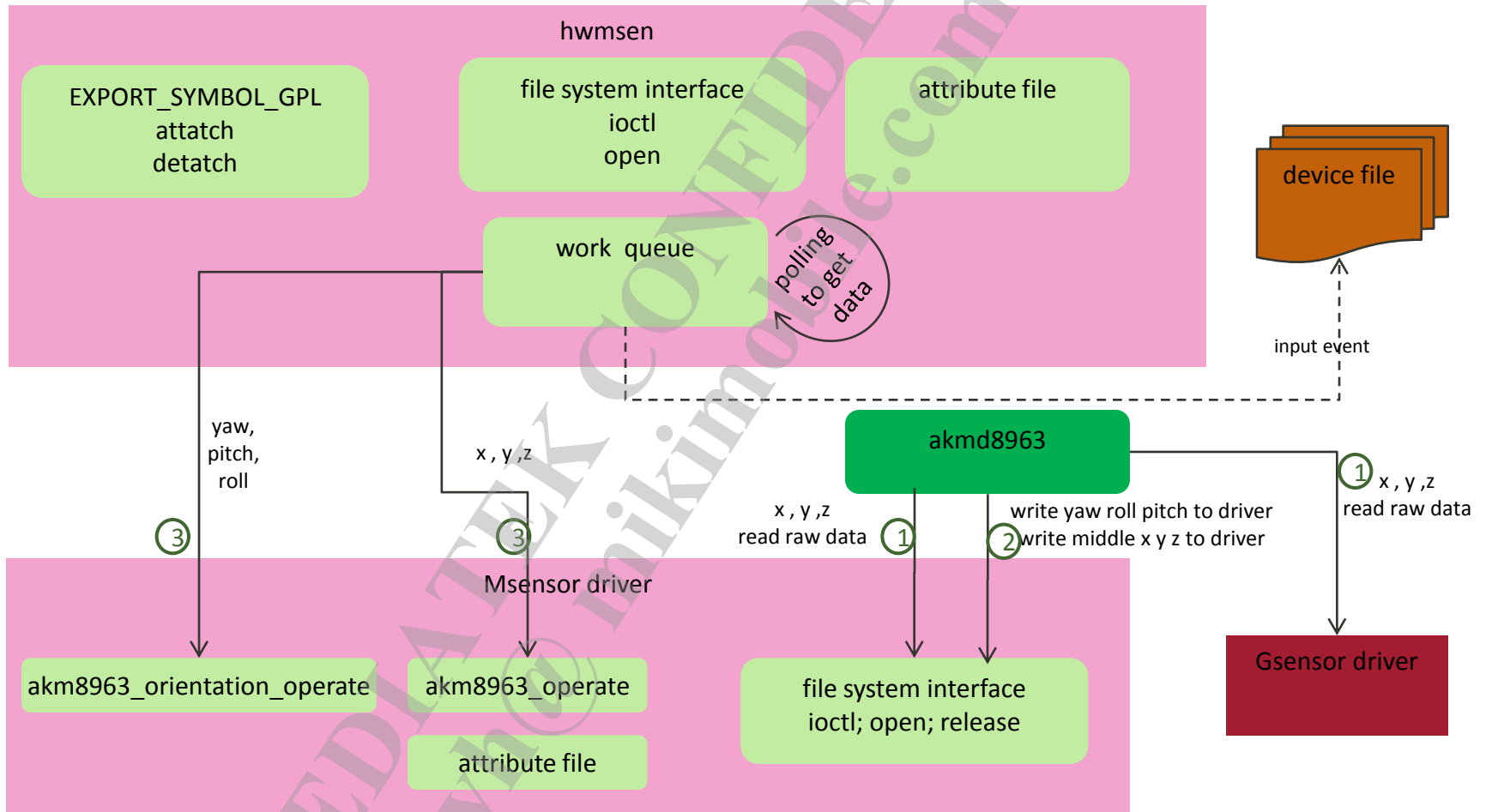


MEDIATEK

Msensor

akm8963 driver work flow

- M-sensor driver architecture



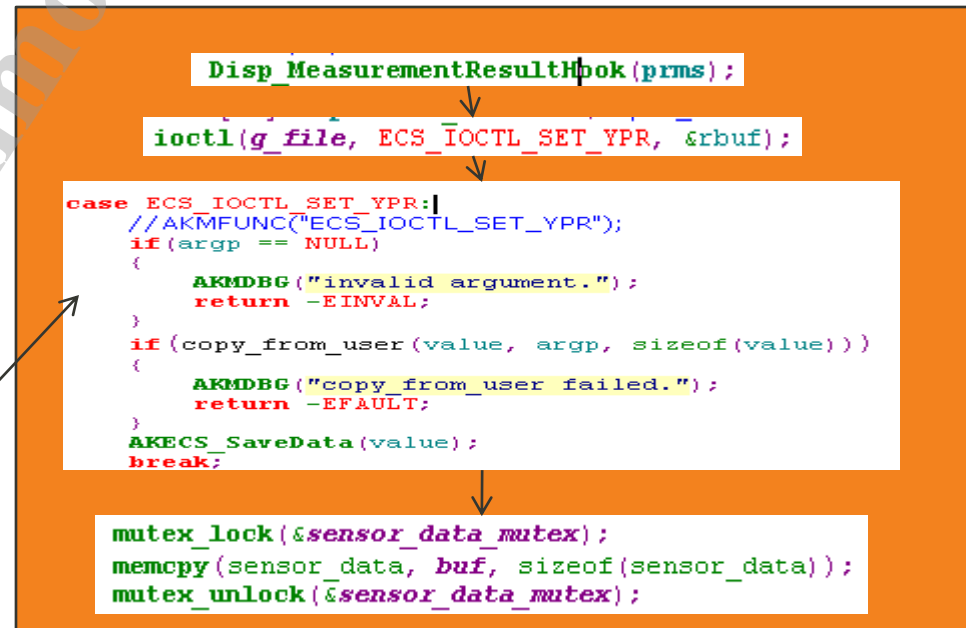
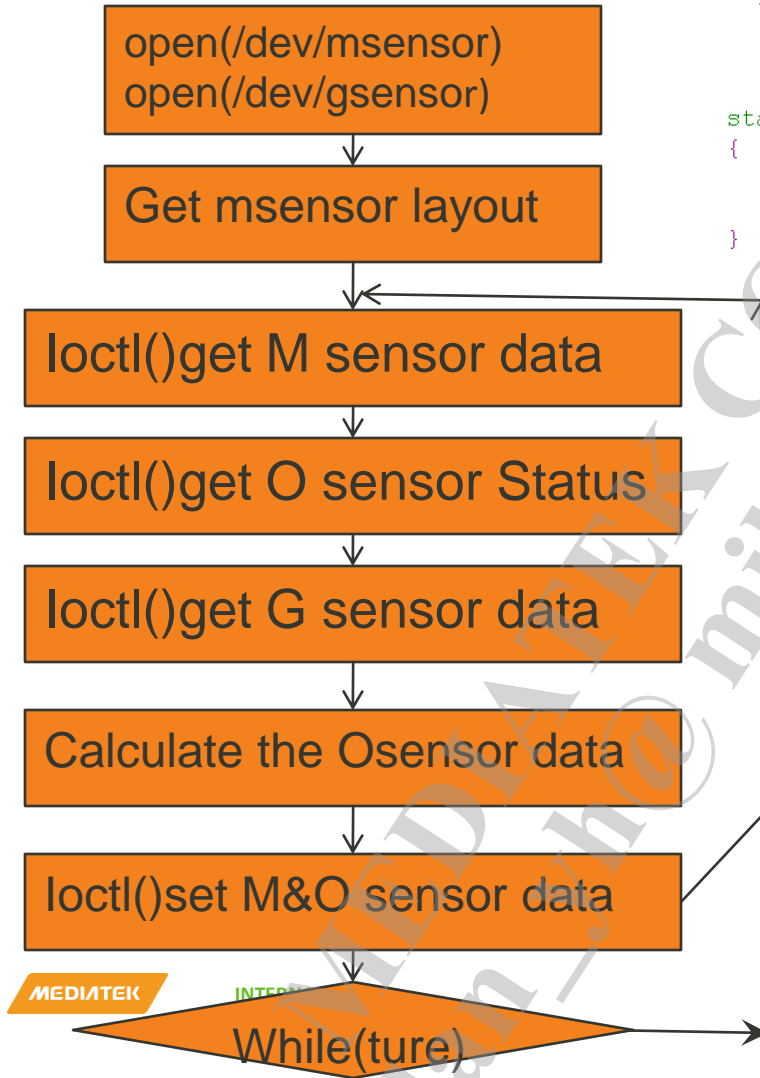
Msensor daemon

Daemon work flow

会根据layout做yaw,pitch,roll的转换

```
static int AKECS_GetOpenStatus(void)
{
    wait_event_interruptible(open_wq, (atomic_read(&open_flag) != 0));
    return atomic_read(&open_flag);
}
```

利用等待队列来通知Daemon程序是否osensor is running



Msensor Factory Mode

FTM test

Orientation sensor data range is list in below table. Please rotate your phone around X,Y,Z axes, and check the output data is right or not.

YAW	Pitch	Roll	Accuracy
0-360°	+180 —> -180	+90 ———> -90	0~3

Msensor data is RAW data , In FTM, we can only monitor the data is changed if you move your devices. This will be enough.

```
Captured Image:
M-Sensor
Test Pass
Test Fail

shipment test: Not support
status: 0
Yaw: 270
Pitch: 24
Roll: 38
Msensor Raw data:
X: -509
Y: 594
Z: -258
```

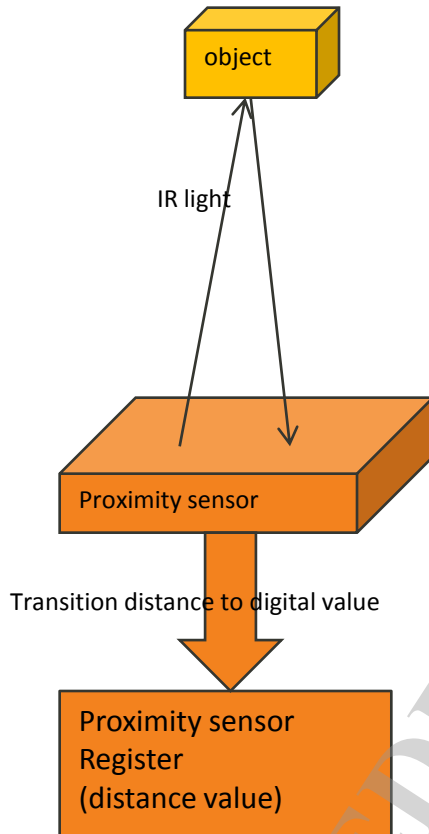
Orientation data

MEDIATEK

Proximity & Ambient Light Sensor

Device introduce

- sensor work mechanism



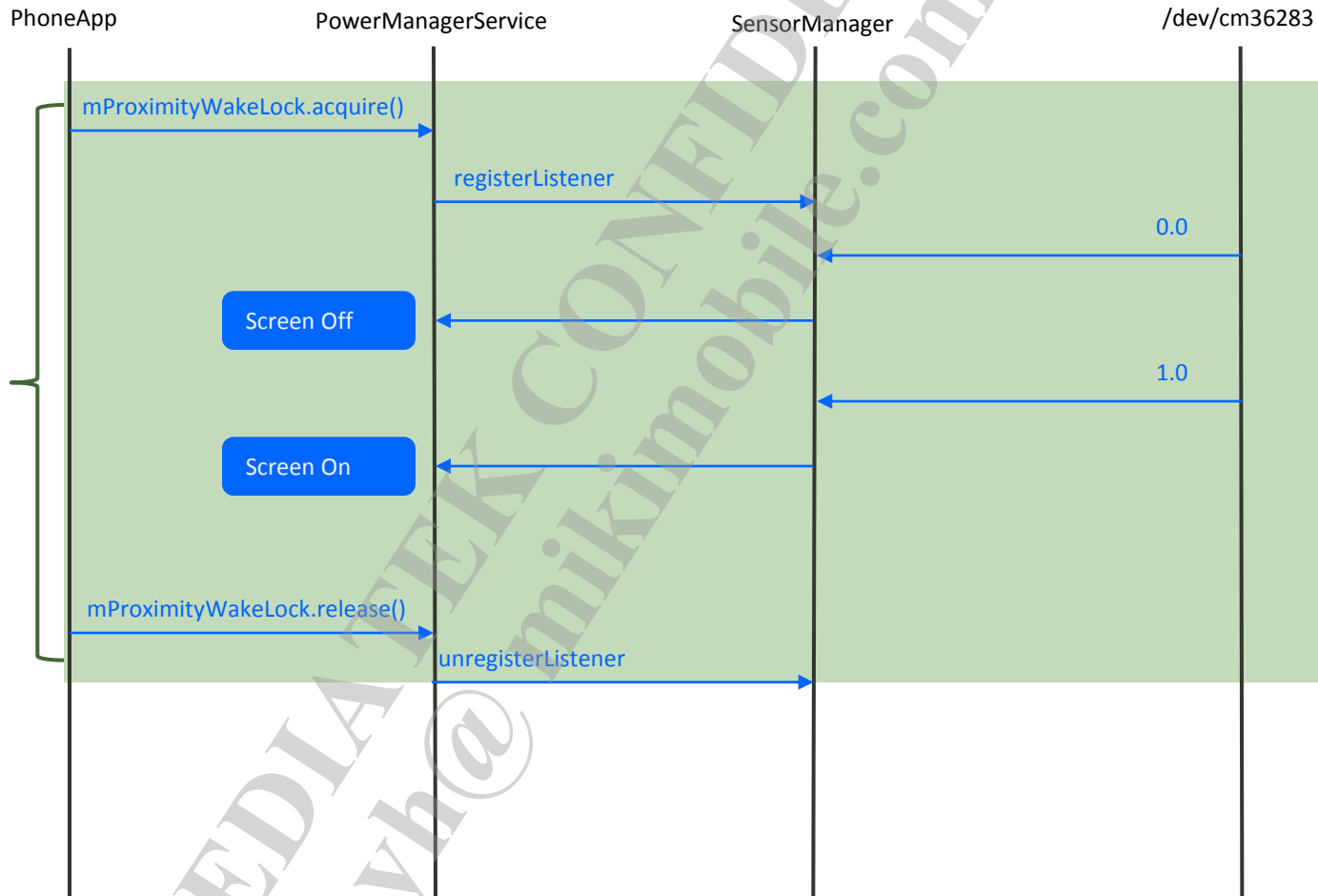
Proximity Sensor(PS) convert distance to digital value through IR LED. Through PS, we can Set system power state when there is a active call.

When there is an active call and user moves the phone close to ear, we can set the system power state to suspend mode. And when user moves the phone away from ear, we can set the system power state to on mode.

Ambient light sensor(ALS) convert light intensity to digital value. We can change back light in different environment through ambient light sensor. The SI unit of ALS is LUX

Android Integration

- How does proximity work when calling?



Driver porting – Customization

■ Change file list

File Name	Location
cust_alsps.h	alps\kernel-3.10\drivers\misc\mediatek\mach\mt67XX\\${PROJ}\alsps\
cust_alsps.c	alps\kernel-3.10\drivers\misc\mediatek\mach\mt67XX\\${PROJ}\alsps\

■ Customization item

```
static struct alsps_hw cust_alsps_hw = {  
    .i2c_num      = 2,  
    .polling_mode_ps = 0,  
    .polling_mode_als = 1,  
    .power_id      = MT65XX_POWER_NONE, /*LDO is not used*/  
    .power_vol      = VOL_DEFAULT, /*LDO is not used*/  
    //.i2c_addr     = {0x0C, 0x48, 0x78, 0x00},  
    .als_level      = { 0, 1, 1, 7, 15, 15, 100, 1000,  
    .als_value       = {40, 40, 90, 90, 160, 160, 225, 320,  
    .ps_threshold_high = 130,  
    .ps_threshold_low  = 120,  
};
```

0为中断模式，1为轮询模式，ps一般都为中断模式

传感器IC寄存器出来的数据是raw data

上报给framework层的数值

靠近时灭屏的阈值

远离时亮屏的阈值

CM36283 Driver Porting Notice (ambient light)

Mapping table

- For different board, only adjust alslv
- The alsval remains no change because Android framework has one mapping table for all board

different device
and different
phone may have
different alslv

no need to change
this values

note:

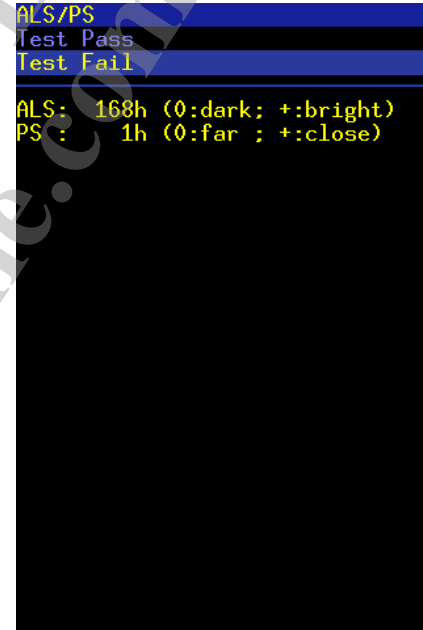
In general, there are 4 types of ambient lighting conditions,

Out door : ($> 10\text{Lux} \pm 720\text{lux}$)
Office : ($\sim 500\text{lux} \pm 112.5\text{lux}$)
Home lighting : ($\sim 200\text{lux} \pm 56.25\text{lux}$)
Dark : ($< 50\text{lux} \pm 11.25\text{lux}$)

Cm 36283 driver level	Driver Value (Unit is LUX)	Framework Level (Unit is LUX)	Framework Value (no unit)
0	40	0	30
1	40	16	40
1	90	32	50
7	90	50	60
15	160	100	70
15	160	140	80
100	225	180	102
1000	320	240	102
2000	640	300	102
3000	1280	600	102
6000	1280	1000	102
10000	2600	2000	180
14000	2600	3000	200
18000	2600	4000	210
20000	10240	8000	230
20000	10240	10000	255

Alsps Factory Mode

- Factory mode
 - Show the raw values continuously in phone UI.



- For ALS: the value is more bigger while sensor is more close to bright.
- For PS: the value is more bigger while sensor is more close to some object

Appendix

- [FAQ09802][Sensor]如何添加新的sensor type (重要)
- [FAQ13477]L版本Msensor SELinux限制问题总结 (重要经常遇到)
- [FAQ13345]Android L版本上指南针apk读取不到sensor数据的原因分析(重要)
- [FAQ04551][sensor] Msensor Daemon该如何检查
- [FAQ13298]Android L版本中实现32bit userspace程序能通过ioctl()系统调用与64bit的kernel driver通信的方法(重要, 经常遇到)
- [FAQ14188]SensorService发生NE分析
- [FAQ04525][sensor] p sensor 阈值如何设置
- [FAQ04122][G/M sensor]如何确定cust_acc/cust_mag中的direction值
- [FAQ08173][sensor]陀螺仪, 重力传感器校准原理及数据格式

MEDIATEK

L版本 Driver Porting Guide 可在dcc上下载