DMARD03 Android Driver V1.38 Porting Guide

Files

Name	Version	Description
dmard03_AD_V1.38_PG.pdf	2012_10_03	This document
src/Driver/dmt030818.c	version 1.35	Linux I2C device driver
src/Driver/dmt030818.h	version 1.35	Linux I2C device driver header
src/libsensors/*	version 1.34	Android HAL
Calibration_applications_Operations_Guide.pdf	2012_09_11	APP Calibration document
DMT_Calibration/*	2012_09_11	APP Calibration source
DMT_Calibration_LAUNCHER.apk	2012_09_11	LAUNCHER APP
DMT_Calibration_DEFAULT.apk	2012_09_11	DEFAULT APP

Android Version

DMARD03 Android Driver V1.38 supports Android 2.3 (Gingerbread) system
And Android 4.0 (Ice-Cream-Sandwich) system.

Index

Revision History

Ver.	Date	Updates	Descriptions
1.38	Oct/03th, 2012	2.3 Gingerbread 4.0 Ice-Cream-Sandwich	1. remove IOCTL(_IOC_NR(cmd) > SENSOR_MAXNR) 2. Unified 1g = 1024 LSB 3. Default close AUTO_CALIBRATION FUNCTION 4. SET BANDWIDTH_REG to No filter 5. Fix sensorlayout's define
1.37	May/31th, 2012	2.3 Gingerbread 4.0 Ice-Cream-Sandwich	ADD device_i2c_txdata FUNCTION SET CONTROL_REGISTER_2 to No filter Note : D06a slave address "0x1d"
1.36	Mar/27th, 2012	2.3 Gingerbread 4.0 Ice-Cream-Sandwich	Revision for schedule_delay, Enhance driver performance, Reduce the use of resources.
1.35	Jan/31th, 2012	2.3 Gingerbread 4.0 Ice-Cream-Sandwich	Gsensor driver combination in order to facilitate follow-up maintenance D03 & of D08 combined src/Driver/dmt0308.c D05 & the D06 & D07 combination src/Driver/dmt050607.c Observing the src/Hal/libsensors /sensors.c Observing the src/Hal/libsensors /Android.mk
1.34	Jan/11th, 2012	2.3 Gingerbread 4.0 Ice-Cream-Sandwich	Fix device_i2c_probe() use device_i2c_xyz_read_reg() replace i2c_master_send() & i2c_master_recv()
1.33	Dec/20th, 2011	2.3 Gingerbread 4.0 Ice-Cream-Sandwich	supports Android 2.3 (Gingerbread) system And Android 4.0 (Ice-Cream-Sandwich) system.
1.32	Dec/01th, 2011	2.3 Gingerbread	first time open device:offset read from the file"/data/misc/dmt/offset.txt" offset save to the file "/data/misc/dmt/offset.txt" offset read from the file "/data/misc/dmt/offset.txt"
1.31	Nov/16th, 2011	2.3 Gingerbread	Fix unlocked_ioctl function in Driver/dmard03.c (v1.21) Be changed to static long device_ioctl (struct file * filp, unsigned int cmd, unsigned long arg)
1.3	Nov/15th, 2011	2.3 Gingerbread	Beagleboard-xM compiler test adjustment, verify that the complet ed version.
1.2	Nov/11th, 2011	2.3 Gingerbread	Unable to complete the DMA-210L burned verify Hsu teacher to modify the generatedvers ion of the code not verified.
1.11	Dec/30th, 2011	2.1 Eclair 2.2 FroYo	first time open device: offset read from the file"/data/misc/dmt/offset.txt" offset save to the file "/data/misc/dmt/offset.txt"

Domintech Co., Ltd. Tel: +886-2-2290-1288 Fax: +886-2-2290-1266 http://www.domintech.com.tw

1.1	Set/08th, 2011	2.1 Eclair 2.2 FroYo	1. Layout pattern selection support. 2. Add 5 ioctl interface: SENSOR_RESET, SENSOR_CALIBRATION, SENSOR_GET_OFFSET, SENSOR_SET_OFFSET, SENSOR_READ_ACCEL_XYZ
1.0	Aug/22th, 2011	2.1 Eclair 2.2 FroYo	First release

Build Kernel Image

1. Copy “src/Driver/dmt030818.c” to “$KERNEL/drivers/misc/”
   Copy “src/Driver/dmt030818.h” to “$KERNEL/include/linux/
   For example in our system it’s in “$KERNEL/drivers/misc/”

   $ cp src/Driver/dmt030818.c $KERNEL/drivers/misc/
   $ cp src/Driver/dmt030818.h $KERNEL/include/linux/

2. In the example code, we have D03’s CE (pin#12) connected to Pandaboard ES GPIO_137. Refer to the file in the “src/board_omap4panda.c”
   Modify the following code segment in “dmt0308.c” to reflect your real configuration.
   i. Function “config_ce_pin”:

```
#include "../../arch/arm/mach-omap2/mux.h"
#define CHIP_ENABLE      137
```

   ii. Function “ce_on”:

```
void ce_on(void){
        omap_mux_set_gpio(OMAP_PIN_INPUT_PULLUP, CHIP_ENABLE);
}
```

   iii. Function “ce_off”:

```
void ce_off(void){
        omap_mux_set_gpio(OMAP_PIN_INPUT_PULLDOWN, CHIP_ENABLE);
}
```

3. Specify the g-sensor layout by defining one of the following macro in “dmt030818.h”. The example code uses layout pattern 1. See Figure 1 for other g-sensor layout pattern orientations.

g-senor layout configuration, choose one of the following configuration

```
#define CONFIG_GSEN_LAYOUT_PAT_1        1
#define CONFIG_GSEN_LAYOUT_PAT_2        0
#define CONFIG_GSEN_LAYOUT_PAT_3        0
#define CONFIG_GSEN_LAYOUT_PAT_4        0
#define CONFIG_GSEN_LAYOUT_PAT_5        0
#define CONFIG_GSEN_LAYOUT_PAT_6        0
#define CONFIG_GSEN_LAYOUT_PAT_7        0
#define CONFIG_GSEN_LAYOUT_PAT_8        0
```

4. Modify "$KERNEL/driver/misc/Makefile" with addition of the following line.
   Refer to patch file in the "src/Makefile.patch"

```
obj-$(CONFIG_SENSORS_DMARD03)      += dmt030818.o
```

5. Modify "$KERNEL/driver/misc/Kconfig" with addition of the following line.
   Refer to patch file in the "src/Kconfig.patch"

```
config SENSORS_DMARD03
        tristate "DMARD03 GSENSOR support"
        default y
        depends on I2C=y
        ---help---
          If you say yes here you get support for accelemeter
          sensor DMARD03.
```

6. Add I2C information to the board information. For example in our system it's in "$KERNEL/arch/arm/mach-omap2/board-omap4panda.c". Locate your own board information file instead. DMARD03 has the option of selecting one of two slave addresses by setting DA (pin#17) to high or low. Sample codes illustrate the case of setting DA (pin#17) to low. In the case of DA (pin#17) set to high, g-sensor's 7-bit slave address would be *0x1c*.
   Refer to patch file in the "src/board-omap4panda.c.patch"

```
static struct i2c_board_info __initdata panda_i2c_boardinfo[]= {
#if (defined(CONFIG_SENSORS_DMARD03) ||
defined(CONFIG_SENSORS_DMARD03_MODULE))
        {I2C_BOARD_INFO("dmt", 0x1c), },
#endif
};
```

7. Rebuild the kernel image

Build Android Image

1. Modify “src/libsensors/Android.mk” with addition of the following line.

```
LOCAL_CFLAGS    := -DLOG_TAG=\"Sensors\"
                   -DSENSORHAL_ACC_D03 \
                   -Wall \
```

2. Copy “src/libsensors” directory to “$ANDROID/hardware/libhardware/module/libsensors”

$ cp –r src/libsensors hardware/libhardware/module/

3. Add the following to the file “$ANDROID/system/core/rootdir/init.rc”

```
chown system system /sys/class/accelemeter/dmt/enable_acc
chown system system /sys/class/accelemeter/dmt/delay_acc
mkdir /data/misc/dmt 0777 system system
chmod 0755 /dev/dmt
```

4. Rebuild the Android image

I2C Connection Circuit

Please refer to “AN001: Application Circuit Examples and Basic Operations of DMARD03” for connection circuit example.

Coordinate System

The coordinate system used by the Android SensorEvent API is defined relative to the screen of the phone in its default orientation, as shown in the Figure 1. Specify the g-sensor layout to one of the layout patterns illustrated below.

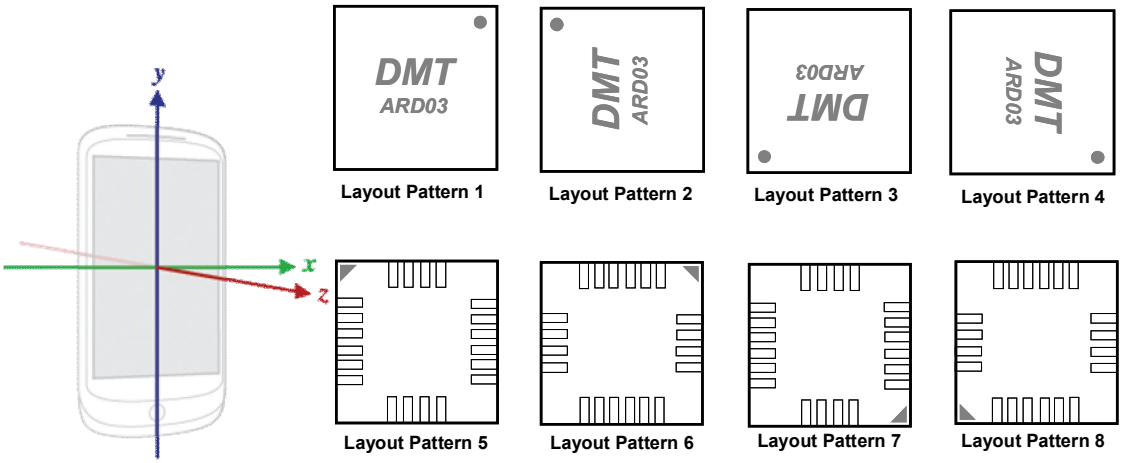

Figure 1: Android SensorEvent Coordinate System and DMARD03 Layout Patterns

Domintech Co., Ltd. Tel: +886-2-2290-1288 Fax: +886-2-2290-1266 http://www.domintech.com.tw

IOCTL Interface

DMARD03 Android Driver implements the following five specific operations to the device.

I. SENSOR_RESET: Conduct g-sensor reset by first pulling CE (pin#12) to low and then setting CE to high. All g-sensor internal registers will be restored to default values after the reset. A typical program segment to reset g-sensor is

```
int fd = open("/dev/dmard03", O_RDONLY);
ioctl(fd, SENSOR_RESET);
```

II. SENSOR_CALIBRATION: Conduct static g-sensor offset calibration. Offset values will be estimated and returned to caller. Furthermore, the offset values will be saved as "/data/misc/dmt/offset.txt" and subsequent g-sensor readings will be automatically compensated by these offset values.
A typical program segment to conduct static calibration is

```
int fd = open("/dev/dmt", O_RDONLY);
int v[3] = {0, 0, 0};
//the first element is set to the static calibration orientation
v[0] = CONFIG_GSEN_CALIBRATION_GRAVITY_ON_Z_NEGATIVE;
ioctl(fd, SENSOR_CALIBRATION, &v);
//the estimated offset values will be returned in v
printf("Offset@X/Y/Z: %04d , %04d , %04d\n", v[0], v[1], v[2]);
```

The static g-sensor offset calibration requires the device to be static. And the caller needs to inform which static orientation the device is positioned when conducting such calibration. The example code illustrates the case in which gravity is acting on the -Z axis. Please refer to Figure 2 for all other static orientations.

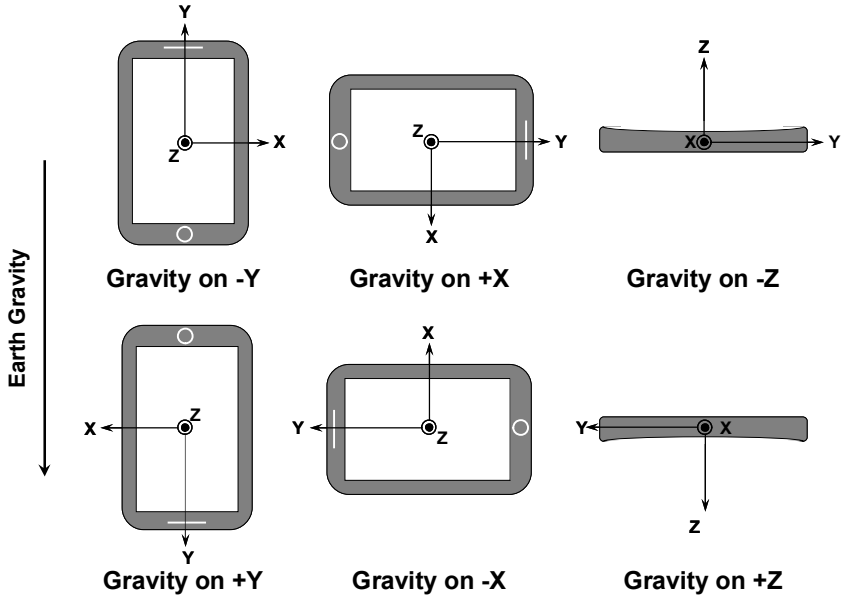

Figure 2: Static Calibration Orientations

III. SENSOR_GET_OFFSET: Get the offset values currently used in the device driver. The

g-sensor readings will be automatically compensated by these offset values. A typical program segment to get offset is

```
int fd = open("/dev/dmt", O_RDONLY);
int v[3] = {0, 0, 0};
ioctl(fd, SENSOR_GET_OFFSET, &v);
//the offset values used in driver will be returned in v
printf("Offset@X/Y/Z: %04d , %04d , %04d\n", v[0], v[1], v[2]);
```

IV. SENSOR_SET_OFFSET: Set the offset values into the device driver. Furthermore, the offset values will be saved as "/data/misc/dmt/offset.txt" and subsequent g-sensor readings will be automatically compensated by these offset values. A typical program segment to set offset is

```
int fd = open("/dev/dmt", O_RDONLY);
int v[3] = {20, -10, 30};    //the offset values to be set to device driver
ioctl(fd, SENSOR_SET_OFFSET, &v);    //the device driver offset is set
```

V. SENSOR_READ_ACCEL_XYZ: Read the acceleration values from the device driver. Note the g-sensor readings will be automatically compensated by the offset values set in the device driver. A typical program segment to read XYZ is

```
int fd = open("/dev/dmt", O_RDONLY);
int v[3];
ioctl(fd, SENSOR_READ_ACCEL_XYZ, &v);
//the acceleration values will be returned in v
printf("Acceleration@X/Y/Z: %04d , %04d , %04d\n", v[0], v[1], v[2]);
```