TouchSense® Android Developer Kit for DRV2605

Driver Integration Guide

1 Driver Integration Overview

The process of integrating the DRV2605 device driver into a mobile handset consists of the following major steps:

1. Modify driver sample code to include the information about the actuator, effect bank, and GPIO port, i2c address and i2c bus
2. Rebuild the driver with your changes
3. Modify Android operating system to load driver at boot time
4. Build Android images for deployment

1.1 I2C Communication

1. Determine the address of the device and the i2c bus that it sits on
2. In drv2605.c, replace the value of macro 'DEVICE_BUS' and 'DEVICE_ADDR' with the obtained value. For example, if the drv2605 chip sits on bus number **3** and has the address of **0x58**, the following lines should appear in the code:

```
/* Address of our device */
#define DEVICE_ADDR 0x58


/* i2c bus that it sits on */
#define DEVICE_BUS  3
```

1.2 GPIO Port

The driver sample code has its GPIO port set to 'GPIO_VIBTONE_EN1'. On a Nexus S phone, this is the GPIO that controls the LRA actuator which sits on GPJ1(1). This value needs to be set to the correct GPIO port that the drv2605 chip connected to.

```
/*
   GPIO port that enable power to the device
*/
#define GPIO_PORT GPIO_VIBTONE_EN1
```

1.3 Effect Bank and Actuator Voltage Selection

The haptic firmware supports several classes of actuators with different characteristics and contains a different bank of effects tuned for each actuator class. OEMs/ODMs must configure the haptic processor or firmware to use the appropriate effect bank for the target actuator. There is one way to configure the haptic processor or firmware for the effect bank:

1. In drv2605.c, the default effect bank is set to bank A (index 0). The default effect bank is defined with the macro "EFFECT_LIBRARY". In order to change this, simply change the macro to a desired effect bank defined in drv2605.h
2. For example, to change from effect bank A to effect bank B, replace 'LIBRARY_A' with 'LIBRARY_B'

```
/*
   DRV2605 built-in effect bank/library
```

```
 */
#define EFFECT_LIBRARY LIBRARY_B;
```

When the effect bank is determined and set, the rated voltage and overdrive voltage of the actuator are automatically set to default values. However, these values can be set manually. To do this, simply change the value of the two macros defined as 'ERM_RATED_VOLTAGE' and 'ERM_OVERDRIVE_CLAMP_VOLTAGE' to the desired values for ERM actuators. For example, if the ERM rated voltage is 3.0v and the ERM overdrive voltage is 3.0v, the two macros should be:

```
#define ERM_RATED_VOLTAGE 0x90
#define ERM_OVERDRIVE_CLAMP_VOLTAGE 0x90
```

If LRA actuator is being used, the two macros 'LRA_RATED_VOLTAGE' and 'LRA_OVERDRIVE_CLAMP_VOLTAGE' will contain the voltage for that actuator. So therefore, they need to be set to the desired values.

For SEMCO LRA with 2.1v RMS rating for drive voltage:

```

#define LRA_RATED_VOLTAGE 0x60
#define LRA_OVERDRIVE_CLAMP_VOLTAGE 0x9E
```

Failure to select the effect bank by either of the above methods may result in sub-optimal effects and damage to the actuator.

Table 1: Effect Banks

Effect Bank Index	Characteristics	Actuator Rated Voltage (V)	Actuator Overdrive Voltage (V)	90% Rise Time at 3V (ms)	90% Brake Time at -3V (ms)
0	Best – with overdrive	1.3	3[1]	40-60	10-20
1	Very good – fast for light devices	3	3	40-60	5-15
2	Good – strong	3	3	60-80	10-20
3	Acceptable	3	3	100-140	15-25
4	Marginal	3	3	>140	>30

[1]Overdrive voltage applied for less than 60ms at a time.

Immersion has characterized and tested certain actuators and determined suitable effect banks for those actuators. Table 2 groups the tested actuators according to effect bank. Immersion used the first actuator in each group as the reference actuator to tune the effects in the effect bank.

Table 2: Actuators

Effect Bank Index	Tested Actuator Models
0	AWA GT-4168 ZLIFE RP1342 Sanyo NRS-2574i
1	ZLIFE RF2323
2	LNLON Y0408A-400050572-M021 KOTL/Jinlong Z4TH5B1241993 DMEGC DM-YK421-7G DMEGC DM-YK407-6F2 KOTL/Jinlong Z4TL2B124167S
3	DMEGC DM-YX403 DMEGC DM-YX402-A
4	AWA GH-4342 (bar) Jinlong C1030B028F (coin)

The process of selecting the appropriate effect bank for a target actuator can be as follows:

1. If the actuator is listed in Table 2, select the effect bank from the table.
2. If the actuator is rated for 1.3V operation and can be overdriven with 3V for up to 60ms at a time, select effect bank 0.
3. If the actuator’s 90% rise time at 3V and 90% brake time at −3V can be determined from the datasheet and/or by measurement, select the effect bank using Table 1. To measure the rise time:
    a. Attach the actuator to a mass equivalent to the handset mass.
    b. Attach an accelerometer to the mass and view the accelerometer output on an oscilloscope.
    c. Rest the mass on a compliant surface such as silicone or soft rubber.
    d. Apply a 3V (positive) pulse to the actuator.
    e. Note how long it takes for the acceleration amplitude to reach 90% of the steady-state (maximum) value.

   To measure the brake time:
    a. Use the same setup as for the rise time.
    b. Apply a positive pulse as for the rise time allowing the acceleration to reach steady-state.
    c. Apply a −3V (negative) pulse and note the time it takes for the acceleration to fall to 10% of the steady-state value. Do not

apply long brake pulses that will cause the motor to spin in the reverse direction. Begin with short brake pulses and increase their duration until the brake time is determined.

4. If the effect bank cannot be determined by step 1, 2, or 3, select each effect bank in turn and determine which effect bank feels best while ensuring that brake pulses from effects never cause the actuator to spin backward:
   a. Attach the actuator to a mass equivalent to the handset mass.
   b. Select an effect bank in the drv2605 driver and rebuild the driver.
   c. Play a few different effects and note how they feel on the target mass. Use the same set of effects in each effect bank for comparison.
   d. Play effects 0 through 5 in the effect bank. Effects 0 through 5 have brake pulses. If the actuator changes direction (spins backward) while playing or stopping an effect, select a different effect bank.
   e. Repeat steps b through d for each effect bank.
   f. After trying all the effect banks, select the effect bank that felt best without the actuator changing direction (spinning backward).

Spinning an actuator backward can damage the actuator. It is important to ensure that brake pulses from playing or stopping effects in the selected effect bank never cause the actuator to spin backward. If effects in the selected effect bank cause the motor to spin backward, the effect bank is inappropriate for the actuator and a different effect bank should be selected.

2 Compiling the driver

To compile the driver, simply locate the drivers folder and execute the following command in the linux shell:

```
ARCH=arm CROSS_COMPILE=[PATH_TO_CCOMPILER] make -C [PATH_TO_KERNEL SOURCE] M=${PWD}
modules
```

where [PATH_TO_CCOMPILER] is replaced with the absolute path to the cross compiler that was used to compile the linux kernel and [PATH_TO_KERNEL_SOURCE] is replaced with the absolute path the kernel source code.

For example:

```
ARCH=arm CROSS_COMPILE=~/Android/prebuilt/linux-x86/toolchain/arm-eabi-4.4.3/bin/arm-
eabi- make -C ~/Code/Kernel/samsung M=${PWD} modules
```

The compilation process should produce a kernel module named **'drv2605.ko'**.

3 Modifying the Android Operating System

This section applies to the Android operating system version 4.0.4 on Nexus S and describes how to change the default haptic implementation on Android to use the drv2605 driver

3.1 Modifying device_base.mk

It is necessary to modify device/samsung/crespo/device_base.mk (relative to the root of the Android source code) to copy **drv2605.ko** to the system image. A copy of drv2605.ko also needs to be copied into device/samsung/crespo/ in order for this to happen. The default **device_base.mk** file for Android 4.0.4 (ICS) at line 219 is as follows.

```
PRODUCT_COPY_FILES += \
        $(LOCAL_WIFI_MODULE):system/modules/bcm4329.ko
```

The following shows how **device_base.mk** can be modified to copy **drv2605.ko** onto the system image.

```
PRODUCT_COPY_FILES += \
        $(LOCAL_WIFI_MODULE):system/modules/bcm4329.ko \
        device/samsung/crespo/drv2605.ko:system/modules/drv2605.ko
```

3.2 Modifying init.herring.rc

At boot time, Android will read a script containing instructions on what to do. We want to be able to load up our driver during this time. The **init.herring.rc** file will allow us to do this. In the 'on post-fs-data' section, the default **init.herring.rc** (located in device/samsung/crespo) is as follows.

```
on post-fs-data
# wi-fi
    mkdir /data/misc/wifi/sockets 0770 wifi wifi
    mkdir /data/misc/dhcp 0770 dhcp dhcp

# create radio & log for ril daemon
    mkdir /data/radio 0775 radio radio
    mkdir /data/radio/log 0775 radio radio

    setprop vold.post_fs_data_done 1
```

The following shows how **init.herring.rc** can be modified to load up **drv2605.ko** at boot time.

```
on post-fs-data
# wi-fi
    mkdir /data/misc/wifi/sockets 0770 wifi wifi
    mkdir /data/misc/dhcp 0770 dhcp dhcp

# create radio & log for ril daemon
    mkdir /data/radio 0775 radio radio
    mkdir /data/radio/log 0775 radio radio
```

```
    setprop vold.post_fs_data_done 1

# load drv2605 driver
    insmod /system/modules/drv2605.ko
```

3.3 Modifying ueventd.herring.rc

When the drv2605.ko driver is loaded into the linux kernel, it will register a device. We want to create this device on the file system and change the permission so that applications can access. To do this, we modify **ueventd.herring.rc** (located in device/samsung/crespo) with the file name and permission that we want. The default ueventd.herring.rc is as follows.

```
/dev/pvrsrvkm             0666   system     system
/dev/uwibro               0660   system     system
/dev/swmxctl              0660   system     system
/dev/video0               0660   system     camera
/dev/video1               0660   system     camera
/dev/video2               0660   system     camera
/dev/s3c-jpg              0660   system     camera
/dev/s3c-mem              0660   system     system
/dev/s3c-mfc              0660   media      media

/dev/modem_ctl            0660   radio      radio
/dev/modem_fmt            0660   radio      radio
/dev/modem_rfs            0660   radio      radio
/dev/s3c2410_serial3      0660   radio      radio
/dev/block/mtdblock5      0660   radio      radio
/dev/mtd/mtd5ro           0660   radio      radio
/dev/mtd/mtd5             0660   radio      radio

# for Sensor HAL
/dev/akm8973              0660   system     system
/dev/accelerometer        0660   system     system

# for GPS
/dev/s3c2410_serial1      0600   gps        gps
```

The following shows how **ueventd.herring.rc** can be modified to create a file in /dev and change its permission:

```
/dev/pvrsrvkm             0666   system     system
/dev/uwibro               0660   system     system
/dev/swmxctl              0660   system     system
/dev/video0               0660   system     camera
/dev/video1               0660   system     camera
/dev/video2               0660   system     camera
/dev/s3c-jpg              0660   system     camera
/dev/s3c-mem              0660   system     system
/dev/s3c-mfc              0660   media      media

# for drv2605
/dev/drv2605              0666   system     system

/dev/modem_ctl            0660   radio      radio
/dev/modem_fmt            0660   radio      radio
/dev/modem_rfs            0660   radio      radio
/dev/s3c2410_serial3      0660   radio      radio
```

```
/dev/block/mtdblock5      0660   radio      radio
/dev/mtd/mtd5ro           0660   radio      radio
/dev/mtd/mtd5             0660   radio      radio

# for Sensor HAL
/dev/akm8973              0660   system     system
/dev/accelerometer        0660   system     system

# for GPS
/dev/s3c2410_serial1      0600   gps        gps
```

4. Rebuilding the Operating System

After making the changes described above, Android can be rebuilt normally by executing **make** from the root of the Android source code.

Immersion Corporation
30 Rio Robles,
San Jose, CA 95134 USA
T: +1 408.467.1900
F: +1 408.467.1901
www.immersion.com