Android IceCreamSandwich

GPS porting Guide on the Android

TCCxxx-Android-IceCreamSandwich
-V1.00E-GPS Porting Guide

May. 2012

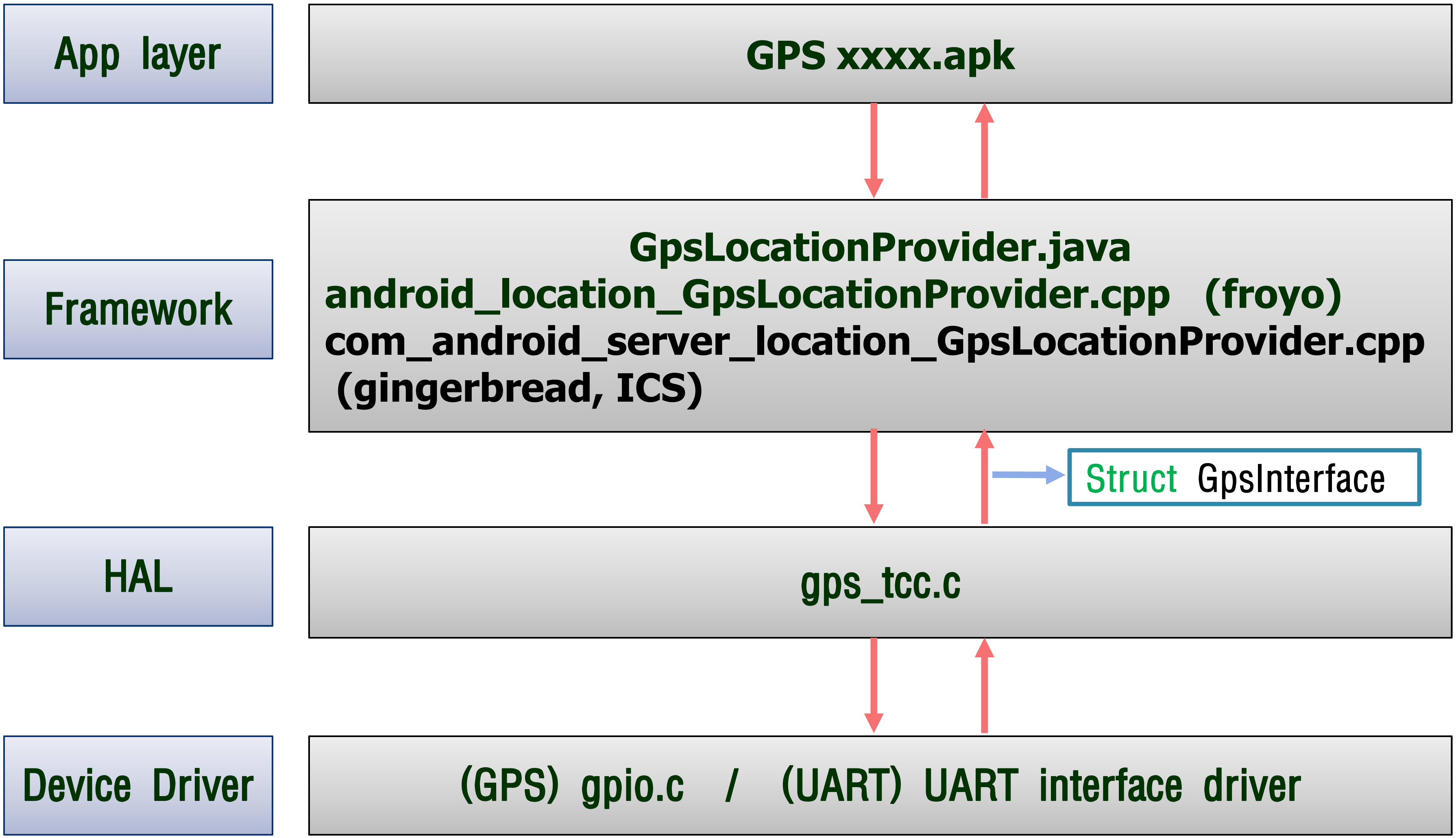
Index

1. Structure
2. Device Driver
3. HAL layer
4. FAQ

Revision History

Date	Version	Description
2012-03-02	1.00	Initial Release ICS

Structure



App layer

GPS xxxx.apk

Framework

GpsLocationProvider.java

android_location_GpsLocationProvider.cpp (froyo)

com_android_server_location_GpsLocationProvider.cpp (gingerbread, ICS)

Struct GpsInterface

HAL

gps_tcc.c

Device Driver

(GPS) gpio.c / (UART) UART interface driver

Device Driver

- File paths : android/kernel/drivers/gps/
- gpio.c (gps_gpio.ko)
  - This file have function to control GPIO port which is enable to GPS module.
  - Open() is function to initialization of GPS device driver.
    Please don’t enable the power of GPS module to reduce the current at sleep/idle state.
  - the ioctl() function controls the power of GPS module.

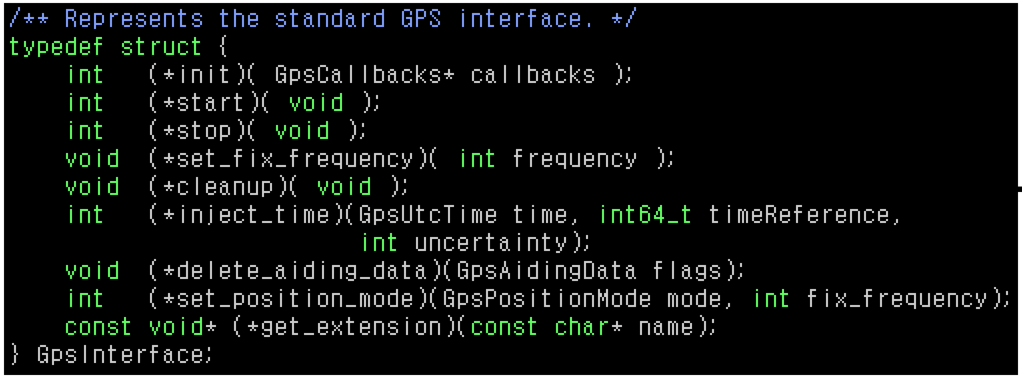
HAL

- File paths : android/hardware/telechips/common/libgps/
- gps_tcc.c
  - Refer to gps_qemu.c to porting GPS HAL layer.
  - The gps_stat_init() of this file calls open() to enable GPS device driver. (gps_gpio, $uart_port)

    ex) $uart_port -> ttyTCC3
  - When system stay on idle status, the gps_state_start() and gps_state_stop() call the ioctl() of gpio.C.

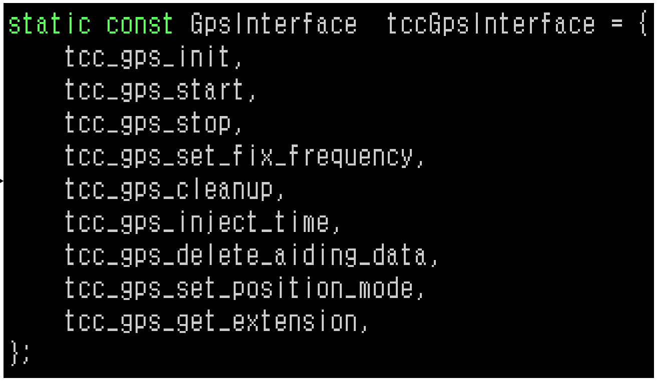
HAL

- File paths : android/hardware/libhardware/include
- gps.h
  - This file has defines and structures and it is related to GPS. (callback-related)
  - Refer to gps_tcc.c of HAL layer.
  - Refer to android_location_GpsLocationProvider.cpp(froyo) of Frameworks
    com_android_server_location_GpsLocationProvider.cpp(gingerbread, ICS)



```
/** Represents the standard GPS interface. */
typedef struct {
    int   (*init)( GpsCallbacks* callbacks );
    int   (*start)( void );
    int   (*stop)( void );
    void  (*set_fix_frequency)( int frequency );
    void  (*cleanup)( void );
    int   (*inject_time)(GpsUtcTime time, int64_t timeReference,
                         int uncertainty);
    void  (*delete_aiding_data)(GpsAidingData flags);
    int   (*set_position_mode)(GpsPositionMode mode, int fix_frequency);
    const void* (*get_extension)(const char* name);
} GpsInterface;
```

gps.h



```
static const GpsInterface  tccGpsInterface = {
    tcc_gps_init,
    tcc_gps_start,
    tcc_gps_stop,
    tcc_gps_set_fix_frequency,
    tcc_gps_cleanup,
    tcc_gps_inject_time,
    tcc_gps_delete_aiding_data,
    tcc_gps_set_position_mode,
    tcc_gps_get_extension,
};
```

tcc_gps.c

  - The init(GpsCallbacks *) register the structure of GpsInterface and link with Frameworks and HAL layer.

HAL

File path : android/hardware/telechips/common/libgps/**gps_tcc.c**

- Checking opening UART3(ttyTCC3) & GPIO

```
state->fd = open(state->device, O_RDWR | O_NONBLOCK | O_NOCTTY);          // UART3
D("tcc : %s Device Open FDescriptor %d", state->device, state->fd);

if (state->fd < 0) {
    D("tcc : no gps Hardware detected");
    return;
}
```

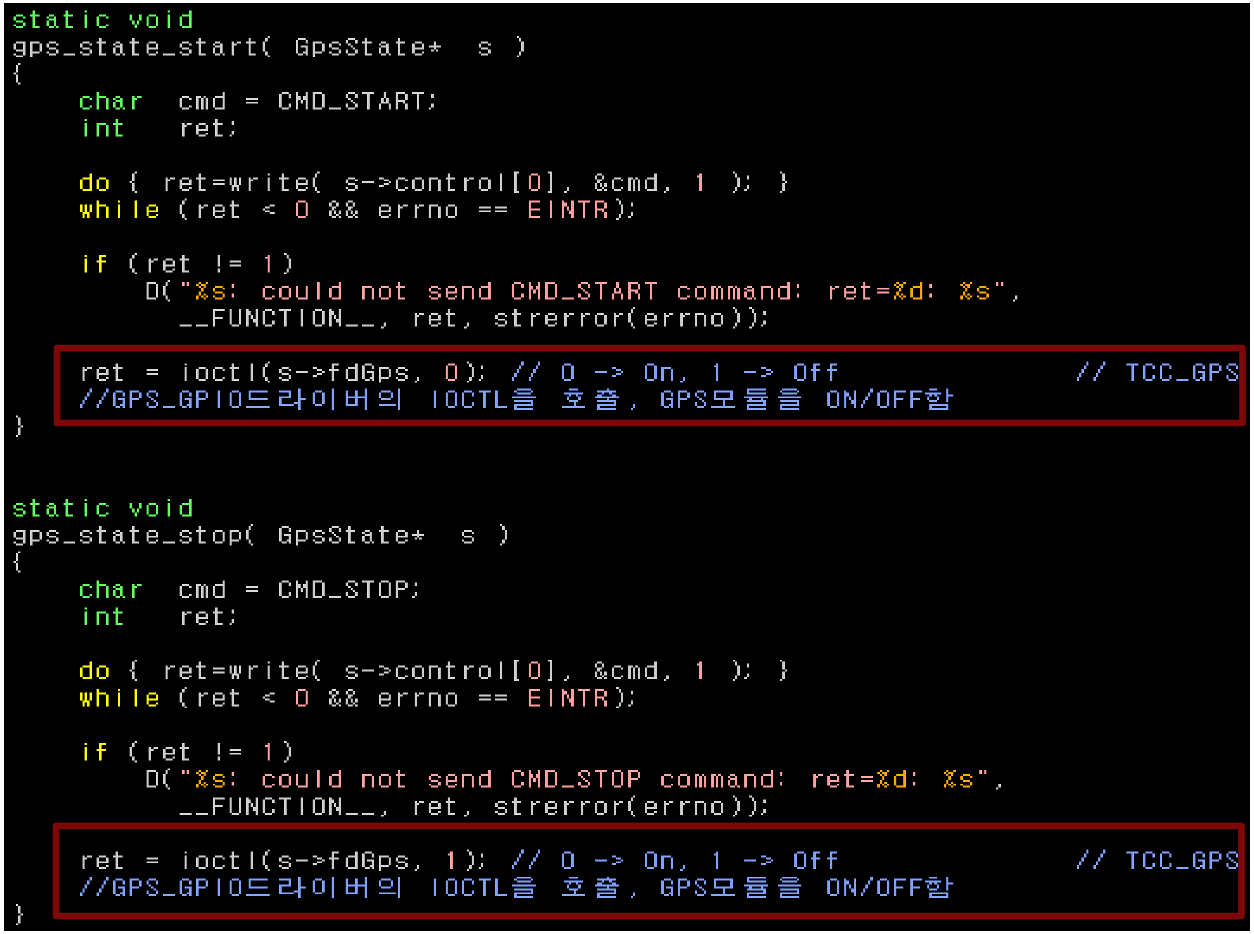
```
state->fdGps = open("/dev/gps_gpio", O_RDWR);                              // Gps_GPIO
D("tcc : Gps_GPIO Device Open FDescriptor %d",state->fdGps);

if (state->fdGps < 0) {
    D("tcc : Couldn't open gps_gpio");
    return;
}
```

HAL

File path : android/hardware/telechips/common/libgps/**gps_tcc.c**
android/kernel/drivers/gps/**gpio.c**

- Checking GPIO setting of ioctl() in gps_state_start() & gps_state_stop()



```
static void
gps_state_start( GpsState*  s )
{
    char  cmd = CMD_START;
    int   ret;

    do { ret=write( s->control[0], &cmd, 1 ); }
    while (ret < 0 && errno == EINTR);

    if (ret != 1)
        D("%s: could not send CMD_START command: ret=%d: %s",
          __FUNCTION__, ret, strerror(errno));

    ret = ioctl(s->fdGps, 0); // 0 -> On, 1 -> Off            // TCC_GPS
    //GPS_GPIO드라이버의 IOCTL을 호출, GPS모듈을 ON/OFF함
}


static void
gps_state_stop( GpsState*  s )
{
    char  cmd = CMD_STOP;
    int   ret;

    do { ret=write( s->control[0], &cmd, 1 ); }
    while (ret < 0 && errno == EINTR);

    if (ret != 1)
        D("%s: could not send CMD_STOP command: ret=%d: %s",
          __FUNCTION__, ret, strerror(errno));

    ret = ioctl(s->fdGps, 1); // 0 -> On, 1 -> Off            // TCC_GPS
    //GPS_GPIO드라이버의 IOCTL을 호출, GPS모듈을 ON/OFF함
}
```

FAQ-01 Setting of UART

- Referenced baud rate of UART5 : 9600 baud rate (bps), at sirf.

There are 5EA by UART channels in 89X,88X,892x

but only UART0 ~ 3 can use DMA to transmit data.

The remaining UART channels (4,5) can not use DMA and there is a baud rate limitation. in case of using UART channels (4.5), the transmission data loss can be occurred by overrun of UART while data is transmitted faster than 9600 baud-rate.

If you want to use a GPS or other device which support baud rate faster than 9600, and if there is no unused UART channel which can support DMA transmission,

please refer to "Android-ALL-V1.00K-Uart-User Guide.pdf"