[frizz Application Note]
frizz kernel driver & HAL Porting description

Rev.1.0

February 3, 2016

MegaChips Corporation

CAUTION

1. Prohibition on copying and disclosure.
   These specifications include intellectual property and know-how belonging to MegaChips Corporation ("MCC") and its partner companies.
   Therefore, do not use these specifications for any purpose other than the purpose expressly agreed to by MCC’s customer and MCC. Also, do not duplicate, copy, or reproduce these specifications, nor reveal these specifications to any third party without MCC’s prior written consent.

2. No guarantee of rights
   MCC does not warrant that it owns or controls any of the information, patents, copyrights or other intellectual property rights contained in these specifications.
   MCC grants no license of patents, copyrights or other intellectual property rights contained in these specifications. MCC disclaims all liability arising from the use of the specifications with regard to infringement of any patents, copyrights or other intellectual property rights.

3. Safety designs such as redundancy
   While MCC has been making continuous effort to enhance the quality and reliability of its devices, the possibility of malfunction cannot be eliminated entirely.
   To minimize risk of damage or injury to persons or property arising from a malfunction in a device, customer must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features. (If the customer intends to use the devices for applications other than those specified between the customer and MCC, please contact MCC sales representative before such use.)

4. Restriction on use
   MCC provides no warranty relating to the use of the devices, where there is risk of serious damage, environmental pollution, loss of life, injury or damage to property, or where reliability or special quality is essential, such as life support, military use, or space exploration.

5. Radiation-proof design
   This device has not been designed to be radiation-resistant.

6. Export restriction
   The export of this device may be regulated by the government under customs, anti-proliferation rules, or other regulations, and export may be prohibited without governmental license.

7. No prior notice of revision
   These specifications are based on materials dated January 23, 2015 and MCC reserves the right to revise these specifications without any prior notice. For mass production planning, please reference the latest version of the specifications.

Table of Contents

1. 簡介

本用戶指南介紹 frizz 驅動功能實，以及動作原理。
開發者能參考此文檔實現 frizz 運行在 Android 系統。

2. 源碼內容

MegaChips 所提供的 frizz Android 開發包內容物，包含以下使用參考指南以及代碼源。

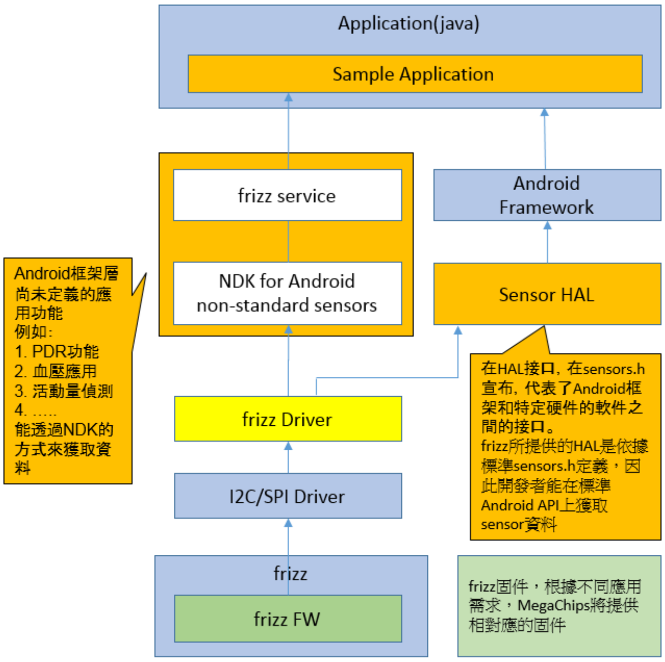
1. frizz-sensor-driver-frizz-kk-xxxxxxxx
2. frizz-sensor-hal-frizz-kk-xxxxxxx
3. [Application Note]frizz kernel driver and android HAL description.

3. 系統需求

1. CPU：ARM32/64
2. frizz 蕊片
3. 內核版本需高於 3.10.28

4. 系統架構

frizz 提供了標準 Android 框架層，開發人員除了透過 Android API 獲取數據外，另外也能透過 NDK 的方式取得 frizz 運算後的資料，下圖為 Android 與 frizz 應用軟體結構圖。



Android框架層尚未定義的應用功能
例如:
1. PDR功能
2. 血壓應用
3. 活動量偵測
4. .....
能透過NDK的方式來獲取資料

在HAL接口，在sensors.h宣布，代表了Android框架和特定硬件的軟件之間的接口。
frizz所提供的HAL是依據標準sensors.h定義，因此開發者能在標準Android API上獲取sensor資料

frizz固件，根據不同應用需求，MegaChips將提供相對應的固件

5. frizz 軟件應用規範

1. 使用標準的 Linux I2C or SPI 與 Timer 和 GPIO 驅動。
2. 透過 I2C or SPI 下載 frizz 固件到 frizz 的內部記憶體。
3. 獲取 frizz 資料的方式
   a. 通過 Timer 定時器，觸發系統讀取 frizz 硬體 FIFO。
   b. 通過 GPIO 中斷，當系統收到中斷觸發讀取 frizz 硬體 FIFO。
4. 基於標準的框架層控制（activate/setDelay/poll...）傳感器。
5. 提供非標傳感器 API，系統能通過此 API 獲取資料。

6. 標準 Sensor HAL 框架層

frizz 傳感器資料，依據 google android sensor HAL 的規範，提供標準 HAL 介面，開發人員也能透過這些標準介面獲取資料。

底下結構體，是 android sensor hal 標準資料結構體，frizz 上報資料格式繼承 sensors_event_t 標準界面。

```
 typedef struct sensors_event_t {
    int32_t version;                /* must be sizeof(struct sensors_event_t) */
    int32_t sensor;                 /* sensor identifier */
    int32_t type;                   /* sensor type */
    int32_t reserved0;              /* reserved */
    int64_t timestamp;              /* time is in nanosecond */
    union {
          union {
             float              data[16];
             sensors_vec_t acceleration;  // acceleration values are in meter per second per second (m/s^2)
             sensors_vec_t magnetic;      // magnetic vector values are in micro-Tesla (uT)
             sensors_vec_t orientation;   // orientation values are in degrees
             sensors_vec_t gyro;          // gyroscope values are in rad/s
             float   temperature;         // temperature is in degrees centigrade (Celsius)
             float   distance;            // distance in centimeters
             float   light;               // light in SI lux units
             float   pressure;            // pressure in hectopascal (hPa)
             float   relative_humidity;   // relative humidity in percent
             uncalibrated_event_t uncalibrated_gyro;           // uncalibrated gyroscope values are in rad/s

             uncalibrated_event_t uncalibrated_magnetic; //uncalibrated magnetometer values are in micro-Teslas
             heart_rate_event_t heart_rate;           // heart rate data containing value in bpm and status

             // this is a special event. see SENSOR_TYPE_META_DATA above.
             // sensors_meta_data_event_t events are all reported with a type of SENSOR_TYPE_META_DATA.
             // The handle is ignored and must be zero.
             meta_data_event_t meta_data;
          };
```

```
            union {
                uint64_t        data[8];
                /* step-counter */
                uint64_t        step_counter;
                } u64;
        };
        /* Reserved flags for internal use. Set to zero. */
        uint32_t flags;
        uint32_t reserved1[3];
    } sensors_event_t;

    /**
     * Heart rate event data
    */
    typedef struct {
         // Heart rate in beats per minute.
         // Set to 0 when status is SENSOR_STATUS_UNRELIABLE or ..._NO_CONTACT
         float bpm;
         // Status of the sensor for this reading. Set to one SENSOR_STATUS_...
         // Note that this value should only be set for sensors that explicitly define the meaning of this field. This field
         // is not piped through the framework for other sensors.
         int8_t status;
    } heart_rate_event_t;
```

關於 frizz 已支持 HAL 介面如下

1. TYPE_ACCELEROMETER

```
    / *
     *   SENSOR_TYPE_ACCELEROMETER
     *   reporting-mode: continuous
     *
     *   All values are in SI units (m/s^2) and measure the acceleration of the device minus the force of gravity.
     *
     *   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
     *   possesses a wake up fifo.
     */
    #define SENSOR_TYPE_ACCELEROMETER                  (1)
    #define SENSOR_STRING_TYPE_ACCELEROMETER       "android.sensor.accelerometer"
```

2. TYPE_MAGNETIC_FIELD

```
    / *
     *   SENSOR_TYPE_GEOMAGNETIC_FIELD
     *   reporting-mode: continuous
     *
     *   All values are in micro-Tesla (uT) and measure the geomagnetic field in the X, Y and Z axis.
     *
```

```
 *   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
 *   possesses a wake up fifo.
 */
 #define SENSOR_TYPE_GEOMAGNETIC_FIELD            (2)
 #define SENSOR_TYPE_MAGNETIC_FIELD               SENSOR_TYPE_GEOMAGNETIC_FIELD
 #define SENSOR_STRING_TYPE_MAGNETIC_FIELD        "android.sensor.magnetic_field"
```

3. TYPE_ORIENTATION

```
/*
*   SENSOR_TYPE_ORIENTATION
*   reporting-mode: continuous
*
*   All values are angles in degrees.
*
*   Orientation sensors return sensor events for all 3 axes at a constant rate defined by setDelay().
*
*   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
*   possesses a wake up fifo.
*/
#define SENSOR_TYPE_ORIENTATION                   (3)
#define SENSOR_STRING_TYPE_ORIENTATION            "android.sensor.orientation"
```

4. TYPE_GYROSCOPE

```
/*
*   SENSOR_TYPE_GYROSCOPE
*   reporting-mode: continuous
*
*   All values are in radians/second and measure the rate of rotation around the X, Y and Z axis.
*
*   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
*   possesses a wake up fifo.
*/
#define SENSOR_TYPE_GYROSCOPE                     (4)
#define SENSOR_STRING_TYPE_GYROSCOPE              "android.sensor.gyroscope"
```

5. TYPE_PRESSURE

```
/*
*   SENSOR_TYPE_PRESSURE
*   reporting-mode: continuous
*
*   The pressure sensor return the athmospheric pressure in hectopascal (hPa)
*
*   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
*   possesses a wake up fifo.
*/
```

```
#define SENSOR_TYPE_PRESSURE                      (6)
#define SENSOR_STRING_TYPE_PRESSURE               "android.sensor.pressure"
```

6. TYPE_GRAVITY

```
/*
*   SENSOR_TYPE_GRAVITY
*   reporting-mode: continuous
*
*   A gravity output indicates the direction of and magnitude of gravity in the devices's coordinates.
*
*   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
*   possesses a wake up fifo.
*/
#define SENSOR_TYPE_GRAVITY                       (9)
#define SENSOR_STRING_TYPE_GRAVITY                "android.sensor.gravity"
```

7. TYPE_LINEAR_ACCELERATION

```
/*
*   SENSOR_TYPE_LINEAR_ACCELERATION
*   reporting-mode: continuous
*
*   Indicates the linear acceleration of the device in device coordinates, not including gravity.
*
*   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
*   possesses a wake up fifo.
*/
#define SENSOR_TYPE_LINEAR_ACCELERATION           (10)
#define SENSOR_STRING_TYPE_LINEAR_ACCELERATION    "android.sensor.linear_acceleration"
```

8. TYPE_ROTATION_VECTOR

```
/*
*   SENSOR_TYPE_ROTATION_VECTOR
*   reporting-mode: continuous
*
*   The rotation vector symbolizes the orientation of the device relative to the East-North-Up coordinates frame.
*
*   Implement the non-wake-up version of this sensor and implement the wake-up version if the system
*   possesses a wake up fifo.
*/
#define SENSOR_TYPE_ROTATION_VECTOR               (11)
#define SENSOR_STRING_TYPE_ROTATION_VECTOR        "android.sensor.rotation_vector"
```

9. TYPE_GAME_ROTATION_VECTOR

```
/*
* SENSOR_TYPE_GAME_ROTATION_VECTOR
* reporting-mode: continuous
*
*   Similar to SENSOR_TYPE_ROTATION_VECTOR, but not using the geomagnetic field.
*
*   Implement the non-wake-up version of this sensor and implement the wake-up
*   version if the system possesses a wake up fifo.
*/
#define SENSOR_TYPE_GAME_ROTATION_VECTOR                    (15)
#define SENSOR_STRING_TYPE_GAME_ROTATION_VECTOR "android.sensor.game_rotation_vector"
```

10. TYPE_SIGNIFICANT_MOTION

```
/*
*   SENSOR_TYPE_SIGNIFICANT_MOTION
*   reporting-mode: one-shot
*
*   A sensor of this type triggers an event each time significant motion is detected and automatically disables
*   itself.
*   For Significant Motion sensor to be useful, it must be defined as a wake-up sensor.
*   (set SENSOR_FLAG_WAKE_UP). Implement the wake-up significant motion sensor.
*   A non wake-up version is not useful. The only allowed value to return is 1.0.
*/
#define SENSOR_TYPE_SIGNIFICANT_MOTION                    (17)
#define SENSOR_STRING_TYPE_SIGNIFICANT_MOTION     "android.sensor.significant_motion"
```

11. TYPE_STEP_DETECTOR

```
/*
*   SENSOR_TYPE_STEP_DETECTOR
*   reporting-mode: special
*
*   A sensor of this type triggers an event each time a step is taken by the user.
*   The only allowed value to return is 1.0 and an event is generated for each step.
*
*   Both wake-up and non wake-up versions are useful.
*/
#define SENSOR_TYPE_STEP_DETECTOR                         (18)
#define SENSOR_STRING_TYPE_STEP_DETECTOR           "android.sensor.step_detector"
```

12. TYPE_STEP_COUNTER

```
/*
*   SENSOR_TYPE_STEP_COUNTER
*   reporting-mode: on-change
*
*   A sensor of this type returns the number of steps taken by the user since the last reboot while activated.
*   The value is returned as a uint64_t and is reset to zero only on a system / android reboot.
*
```

```
* Implement the non-wake-up version of this sensor and implement the wake-up
* version if the system possesses a wake up fifo.
*/
#define SENSOR_TYPE_STEP_COUNTER                    (19)
#define SENSOR_STRING_TYPE_STEP_COUNTER             "android.sensor.step_counter"
```

13. TYPE_GEOMAGNETIC_ROTATION_VECTOR

```
/*
* SENSOR_TYPE_GEOMAGNETIC_ROTATION_VECTOR
* reporting-mode: continuous
*
* Similar to SENSOR_TYPE_ROTATION_VECTOR, but using a magnetometer instead of using a gyroscope.
*
* Implement the non-wake-up version of this sensor and implement the wake-up version if the system
* possesses a wake up fifo.
*/
#define SENSOR_TYPE_GEOMAGNETIC_ROTATION_VECTOR       (20)
#define SENSOR_STRING_TYPE_GEOMAGNETIC_ROTATION_VECTOR
"android.sensor.geomagnetic_rotation_vector"
```

14. TYPE_LIGHT

```
/*
* SENSOR_TYPE_LIGHT
* reporting-mode: on-change
*
* The light sensor value is returned in SI lux units.
*
* Both wake-up and non wake-up versions are useful.
*/
#define SENSOR_TYPE_LIGHT                           (5)
#define SENSOR_STRING_TYPE_LIGHT                    "android.sensor.light"
```

15. TYPE_PROXIMITY

```
/*
* SENSOR_TYPE_PROXIMITY
* reporting-mode: on-change
*
* The proximity sensor which turns the screen off and back on during calls is the wake-up proximity sensor.
* Implement wake-up proximity sensor before implementing a non wake-up proximity sensor. For the wake-up
* proximity sensor set the flag
* SENSOR_FLAG_WAKE_UP.
* The value corresponds to the distance to the nearest object in centimeters.
*/
#define SENSOR_TYPE_PROXIMITY                       (8)
#define SENSOR_STRING_TYPE_PROXIMITY                "android.sensor.proximity"
```

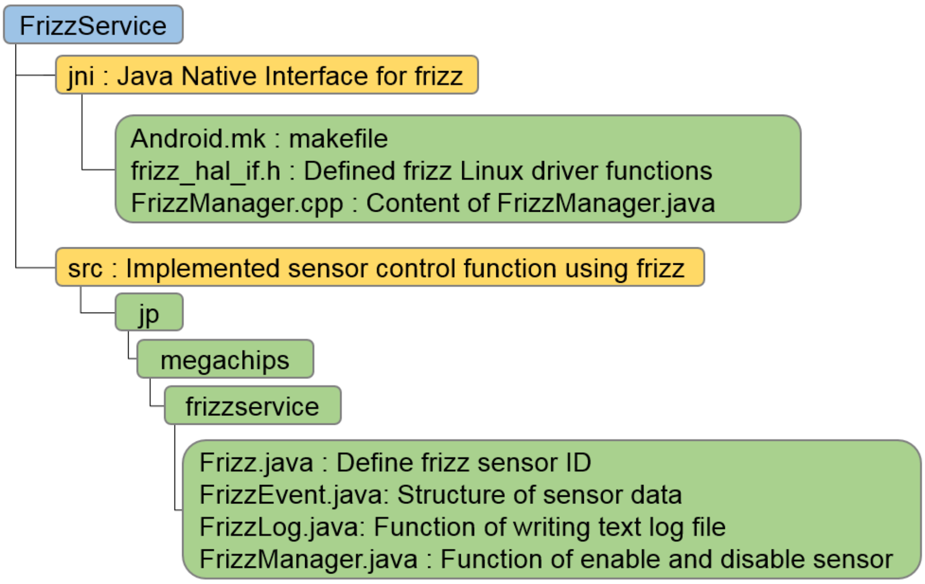
16. TYPE_HEART_RATE

```
/*
*   SENSOR_TYPE_HEART_RATE
*   reporting-mode: on-change
*
*   A sensor of this type returns the current heart rate. The events contain the current heart rate in beats per *
*   minute (BPM) and the status of the sensor during the measurement. See heart_rate_event_t for more
*   details.
*
*   Because this sensor is on-change, events must be generated when and only when heart_rate.bpm or
*   heart_rate.status have changed since the last event.
*   In particular, upon the first activation, unless the device is known to not be on the body, the status field of the
*   first event must be set to SENSOR_STATUS_UNRELIABLE. The event should be generated no faster than
*   every period_ns passed to setDelay() or to batch(). See the definition of the on-change reporting mode for
*   more information.
*   sensor_t.requiredPermission must be set to SENSOR_PERMISSION_BODY_SENSORS.
*
*   Both wake-up and non wake-up versions are useful.
*/
#define SENSOR_TYPE_HEART_RATE                          (21)
#define SENSOR_STRING_TYPE_HEART_RATE                    "android.sensor.heart_rate"
```

Sensor HAL 會因不同的 android 版本或許會有些許修改，因此詳細說明請參閱 android 網頁說明
https://source.android.com/devices/sensors/index.html

7. NDK 跟 frizz service 架構

1. 下圖為 frizz API 的檔案架構


FrizzService
jni : Java Native Interface for frizz
Android.mk : makefile
frizz_hal_if.h : Defined frizz Linux driver functions
FrizzManager.cpp : Content of FrizzManager.java
src : Implemented sensor control function using frizz
jp
megachips
frizzservice
Frizz.java : Define frizz sensor ID
FrizzEvent.java: Structure of sensor data
FrizzLog.java: Function of writing text log file
FrizzManager.java : Function of enable and disable sensor

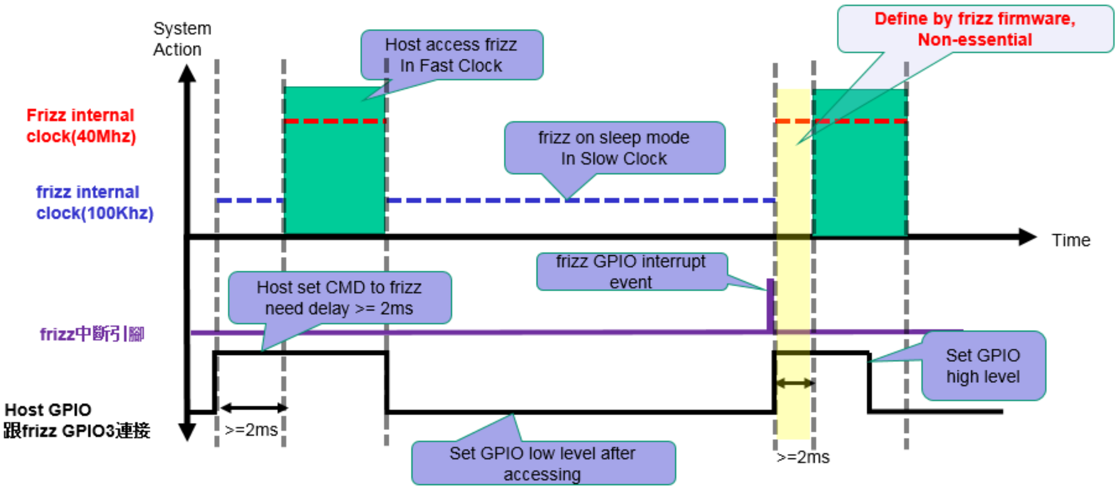
2. 開發者透過 JNI 直接呼叫 frizz ioctl 驅動控制 frizz，並獲取資料，實際應用請參考 MegaChips 所提供的範例 APK

8. frizz 驅動移植

1. 節點樹配置(Device tree)
   設備樹是一種數據結構，用於描述硬件。設備樹的配置，需根據開發平台自行配置，請先將此配置完成。

2. I2C or SPI 配置
   a. 確認 I2C 或 SPI 的組態，修改相對應的組態
      frizz_serial.h
      #define FRIZZ_I2C_BUS (3) → 請根據不同的平台做相對應的更改
   b. I2C 一次存取最大數量，請根據主控晶片硬件做相對應修改
      frizz_serial_i2c.c
      #define MAX_ACCESS_LIMIT (32) → I2C 一次存取最大 Bytes 的數量

3. GPIO 配置，請根據系統平台修改 frizz_gpio.c
   a. Host 獲取 frizz 資料能透過中斷或輪尋的方式
      使用中斷需配置 GPIO
   b. frizz 低功耗模式
      當 frizz 進入低功耗模式，內部會處於 100Khz，此時 frizz 跟 host 之間的介面(SPI or I2C)是無法使用的，需透過 GPIO 將 frizz 喚醒(40Mhz)，才能存取 frizz，如應用上需要此模式，請配置 GPIO 來控制 frizz，下圖為低功耗模式控制方式。在開發階段請確認平台控制流程符合此規範。


System Action
Host access frizz In Fast Clock
Define by frizz firmware, Non-essential
Frizz internal clock(40Mhz)
frizz on sleep mode In Slow Clock
frizz internal clock(100Khz)
Time
frizz GPIO interrupt event
Host set CMD to frizz need delay >= 2ms
frizz中斷引腳
Set GPIO high level
Host GPIO 跟frizz GPIO3連接
>=2ms
Set GPIO low level after accessing
>=2ms

4. 除錯 log，在 frizz 驅動 frizz_debug.h 內定義了 log 打印的種類，請根據平台需求修改所需觀察的 log 訊息
5. 編譯 frizz 驅動
   a. 將修改完成的源碼放置於 char 資料夾，底下是使用 DragonBoard410c 開發版做描述
      不同的平台會有所差異，請依據環境的差異做相對應的配置
      $ mv frizz <android-source>/kerneldrivers/char/

```
$ vim <android-source>/kernel/drivers/char/Kconfig
```

//編輯 Kconfig 檔案，將 frizz 驅動的 Kconfig 連結，修改如下

```
+
+ source "drivers/char/frizz/Kconfig"
+
```

```
$ vim <android-source>/kernel/drivers/char/Makefile
```

//編輯 Makefile，將 frizz 驅動連結，修改如下

```
obj-$(CONFIG_MSM_RDBG)          += rdbg.o
obj-$(CONFIG_MSM_SMD_PKT)       += msm_smd_pkt.o
+ obj-$(CONFIG_FRIZZ)           += frizz_driver/
```

b. 編譯 kernel config

```
$ cd <android-source>
$ source build/envsetup.sh
$ lunch msm8916_64-userdebug
$ make kernelconfig
```

編譯後會跳出視窗，能經由此對話視窗，選擇所需要的應用

Device Drivers ---> [Enter]

Character devices ---> [Enter]

Mega Chips support ---> [Enter]

<M> Frizz sensor hub support

[<*> is kernel implementation. <M> is generated ko file]

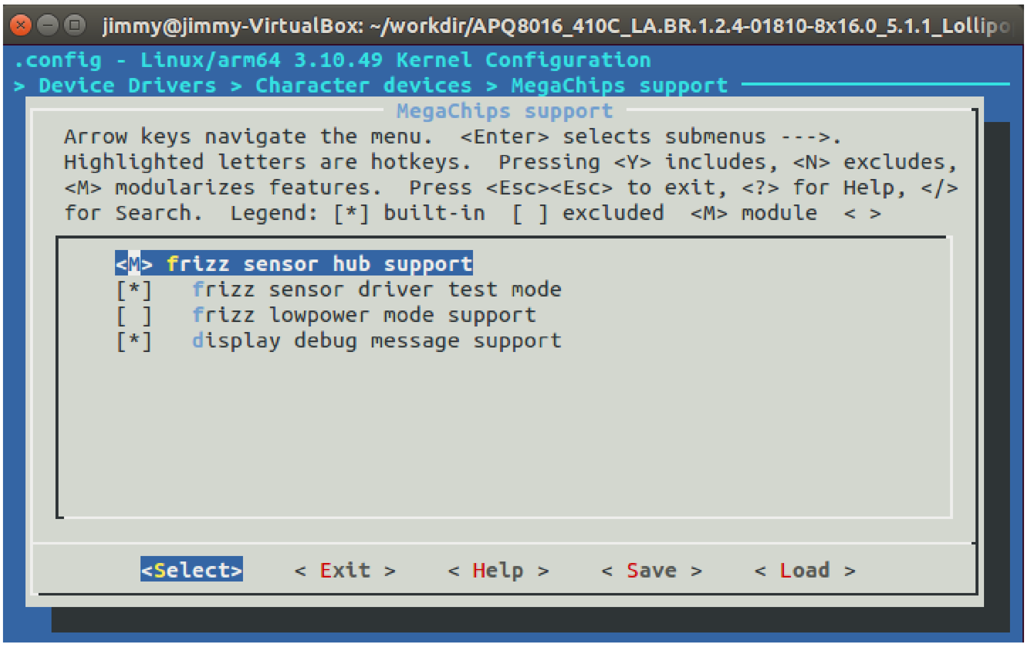
[*] Frizz sensor driver test mode

[ ] frizz lowpower mode support

[[*] boot frizz low power mode. we need to connect gpio.(host -> frizz)]

[*] display debug message support

[[*] display debug message support can output debug message.]



```
jimmy@jimmy-VirtualBox: ~/workdir/APQ8016_410C_LA.BR.1.2.4-01810-8x16.0_5.1.1_Lollipo
.config - Linux/arm64 3.10.49 Kernel Configuration
> Device Drivers > Character devices > MegaChips support
                        MegaChips support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

      <M> frizz sensor hub support
      [*]   frizz sensor driver test mode
      [ ]   frizz lowpower mode support
      [*]   display debug message support

      <Select>    < Exit >    < Help >    < Save >    < Load >
```

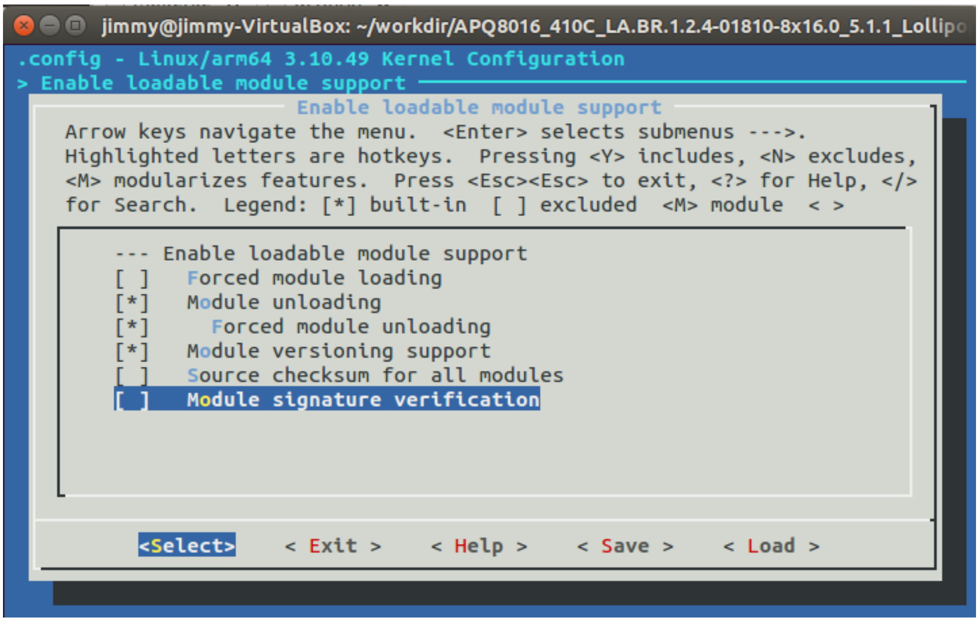
完成設定後，從新編譯內核

$ make –j8 bootimage

Linux 的內核有檢查模塊的簽名功能，並且如果 frizz 驅動程序沒有簽名，便無法透過『insmod frizz.ko』的方式來使用。

Enable loadable module support ---> [Enter]

[ ]　Module signature verification



```
jimmy@jimmy-VirtualBox: ~/workdir/APQ8016_410C_LA.BR.1.2.4-01810-8x16.0_5.1.1_Lollipo
.config - Linux/arm64 3.10.49 Kernel Configuration
> Enable loadable module support ──────────────────────────────────
              Enable loadable module support
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

      --- Enable loadable module support
      [ ]   Forced module loading
      [*]   Module unloading
      [*]     Forced module unloading
      [*]   Module versioning support
      [ ]   Source checksum for all modules
      [ ]   Module signature verification

         <Select>    < Exit >    < Help >    < Save >    < Load >
```

c.　請在初次移植先透過 MegaChips 所提供的測試工具，在系統測試是否有正常工作，底下將說明，如何測試

$ adb shell　　　//切換至 DragonBoard410c

root@msm8916_64:/data/frizz:/ # cd data

root@msm8916_64:/data/frizz:/ # ls

root@msm8916_64:/data/frizz:/ # mkdir frizz

//如果開發版內沒有/data/frizz 目錄，請創建 frizz 目錄。之後一些測試的檔案都將存放於此。並非一定要在此資料夾內，能自行使用其它名稱

//請開啟 Terminal Tool

$ adb shell cat /proc/kmsg　　　　// 透過 log，來觀察 frizz 動作狀態

//將編譯成功的檔案推至開發版內

$ adb push

<android-source>/out/target/product/msm8994/obj/KERNEL_OBJ/drivers/ char/
　　　　frizz_driver/frizz.ko /data/frizz

$ adb root

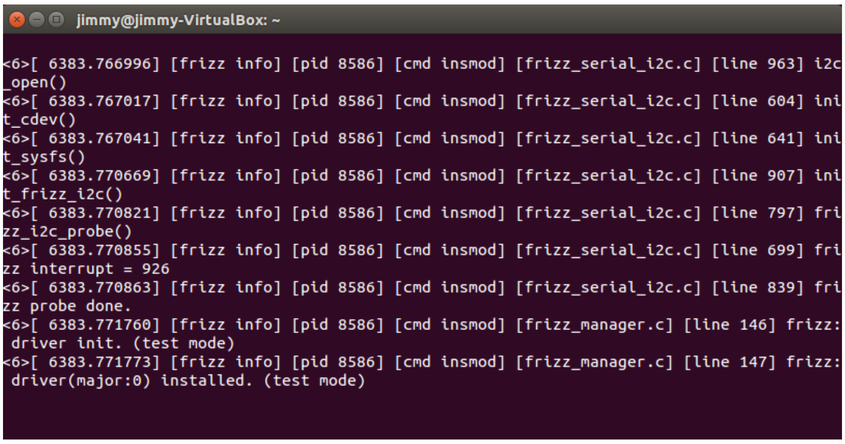
$ adb shell

//切換至開發版

root@msm8916_64:/data/frizz:/ # cd /data/frizz/

root@msm8916_64:/data/frizz:/ # insmod frizz.ko

//手動掛載驅動，掛載成功

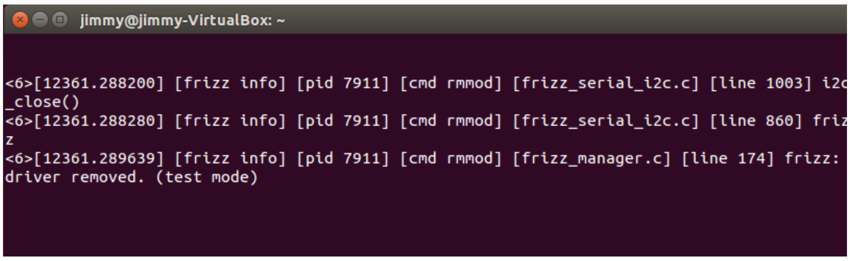


jimmy@jimmy-VirtualBox: ~

```
<6>[ 6383.766996] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 963] i2c
_open()
<6>[ 6383.767017] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 604] ini
t_cdev()
<6>[ 6383.767041] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 641] ini
t_sysfs()
<6>[ 6383.770669] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 907] ini
t_frizz_i2c()
<6>[ 6383.770821] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 797] fri
zz_i2c_probe()
<6>[ 6383.770855] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 699] fri
zz interrupt = 926
<6>[ 6383.770863] [frizz info] [pid 8586] [cmd insmod] [frizz_serial_i2c.c] [line 839] fri
zz probe done.
<6>[ 6383.771760] [frizz info] [pid 8586] [cmd insmod] [frizz_manager.c] [line 146] frizz:
 driver init. (test mode)
<6>[ 6383.771773] [frizz info] [pid 8586] [cmd insmod] [frizz_manager.c] [line 147] frizz:
 driver(major:0) installed. (test mode)
```

root@msm8916_64:/data/frizz:/ # rmmod frizz.ko

//手動卸載驅動，卸載成功



jimmy@jimmy-VirtualBox: ~

```
<6>[12361.288200] [frizz info] [pid 7911] [cmd rmmod] [frizz_serial_i2c.c] [line 1003] i2c
_close()
<6>[12361.288280] [frizz info] [pid 7911] [cmd rmmod] [frizz_serial_i2c.c] [line 860] friz
z
<6>[12361.289639] [frizz info] [pid 7911] [cmd rmmod] [frizz_manager.c] [line 174] frizz:
driver removed. (test mode)
```

frizz 測試工具源碼編譯

i. 讀取 frizz 寄存器測試

$ cd <android-source>/vender/

$ cd tools/frizz_reg_test/

$ mm –B

編譯成功後，將產生出來的檔案放於開發版內

$ adb push <android-source>/out/target/product/msm8994/system/bin/frizz_reg_test /data/frizz

$ adb root

$ adb shell //切換至開發版

root@msm8916_64:/data/frizz:/ # cd /data/frizz/

root@msm8916_64:/data/frizz:/ # chmod 555 frizz_reg_test

root@msm8916_64:/data/frizz:/ # insmod frizz.ko

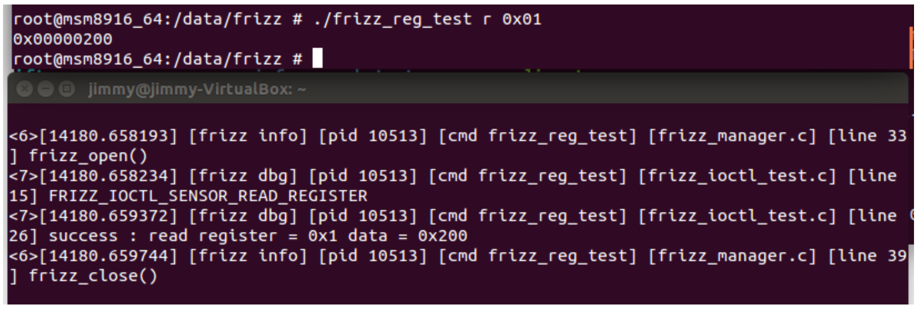
Read Format

# ./frizz_reg_test r [reg_addr]

[read_result]

root@msm8916_64:/data/frizz:/ # ./frizz_reg_test r 0x01 // 讀取 frizz ID 寄存器指令

0x00000200 // 讀取結果



```
root@msm8916_64:/data/frizz # ./frizz_reg_test r 0x01
0x00000200
root@msm8916_64:/data/frizz # 
```

```
<6>[14180.658193] [frizz info] [pid 10513] [cmd frizz_reg_test] [frizz_manager.c] [line 33
] frizz_open()
<7>[14180.658234] [frizz dbg] [pid 10513] [cmd frizz_reg_test] [frizz_ioctl_test.c] [line
15] FRIZZ_IOCTL_SENSOR_READ_REGISTER
<7>[14180.659372] [frizz dbg] [pid 10513] [cmd frizz_reg_test] [frizz_ioctl_test.c] [line
26] success : read register = 0x1 data = 0x200
<6>[14180.659744] [frizz info] [pid 10513] [cmd frizz_reg_test] [frizz_manager.c] [line 39
] frizz_close()
```

ii. 寫入 frizz 寄存器測試

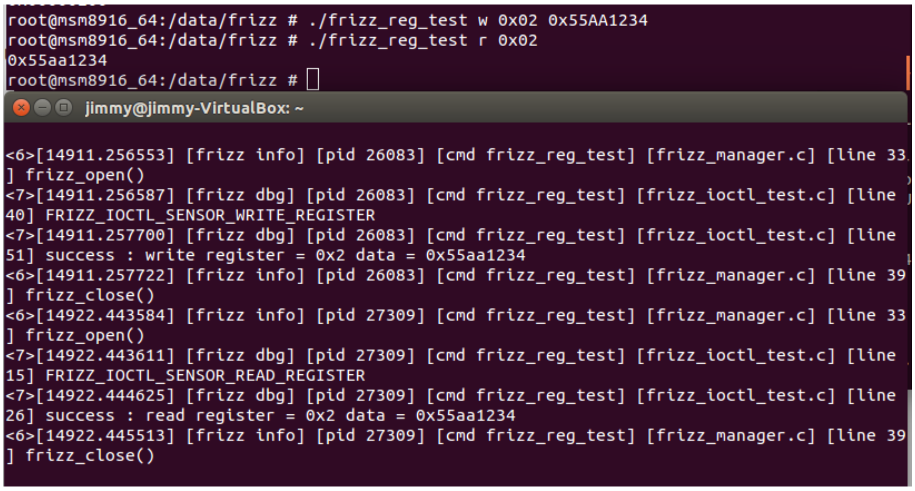
Write Format

# ./frizz_reg_test w [reg_addr] [reg_data]

root@msm8916_64:/data/frizz:/ # ./frizz_reg_test w 0x02 0x55AA1234 //寫入寄存器指令

root@msm8916_64:/data/frizz:/ # ./frizz_reg_test r 0x02 //讀取結果

0x55aa1234



```
root@msm8916_64:/data/frizz # ./frizz_reg_test w 0x02 0x55AA1234
root@msm8916_64:/data/frizz # ./frizz_reg_test r 0x02
0x55aa1234
root@msm8916_64:/data/frizz # 
```

```
<6>[14911.256553] [frizz info] [pid 26083] [cmd frizz_reg_test] [frizz_manager.c] [line 33
] frizz_open()
<7>[14911.256587] [frizz dbg] [pid 26083] [cmd frizz_reg_test] [frizz_ioctl_test.c] [line
40] FRIZZ_IOCTL_SENSOR_WRITE_REGISTER
<7>[14911.257700] [frizz dbg] [pid 26083] [cmd frizz_reg_test] [frizz_ioctl_test.c] [line
51] success : write register = 0x2 data = 0x55aa1234
<6>[14911.257722] [frizz info] [pid 26083] [cmd frizz_reg_test] [frizz_manager.c] [line 39
] frizz_close()
<6>[14922.443584] [frizz info] [pid 27309] [cmd frizz_reg_test] [frizz_manager.c] [line 33
] frizz_open()
<7>[14922.443611] [frizz dbg] [pid 27309] [cmd frizz_reg_test] [frizz_ioctl_test.c] [line
15] FRIZZ_IOCTL_SENSOR_READ_REGISTER
<7>[14922.444625] [frizz dbg] [pid 27309] [cmd frizz_reg_test] [frizz_ioctl_test.c] [line
26] success : read register = 0x2 data = 0x55aa1234
<6>[14922.445513] [frizz info] [pid 27309] [cmd frizz_reg_test] [frizz_manager.c] [line 39
] frizz_close()
```

iii. 透過測試腳本能測試 frizz 大量讀寫狀態

在所提供的代碼內包含 FrizzRegTest.sh，這個 shell 腳本可以確認大量讀寫狀態

$ adb root

$ adb push FrizzRegTest.sh /data/frizz

$ adb push FrizzRegTestFunc.sh /data/frizz

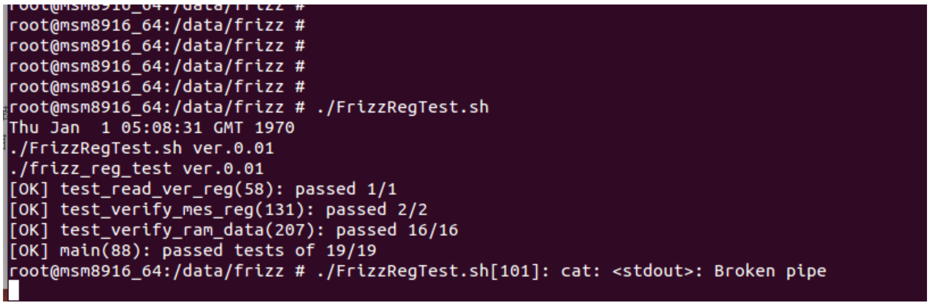
$ adb shell

root@msm8916_64:/data/frizz:/ # cd /data/frizz/

root@msm8916_64:/data/frizz:/ # chmod 555 *.sh

root@msm8916_64:/data/frizz:/ # ./FrizzRegTest.sh

//結果顯式如下



```
root@msm8916_64:/data/frizz #
root@msm8916_64:/data/frizz #
root@msm8916_64:/data/frizz #
root@msm8916_64:/data/frizz #
root@msm8916_64:/data/frizz #
root@msm8916_64:/data/frizz # ./FrizzRegTest.sh
Thu Jan  1 05:08:31 GMT 1970
./FrizzRegTest.sh ver.0.01
./frizz_reg_test ver.0.01
[OK] test_read_ver_reg(58): passed 1/1
[OK] test_verify_mes_reg(131): passed 2/2
[OK] test_verify_ram_data(207): passed 16/16
[OK] main(88): passed tests of 19/19
root@msm8916_64:/data/frizz # ./FrizzRegTest.sh[101]: cat: <stdout>: Broken pipe
```

執行至此步驟都能成功，已經能確認 host 跟 frizz 溝通上是沒問題的。

6. 實際運行請將 config 的 test mode 關閉，如果未關閉，請重新編譯 Kconfig，並且重新編譯 kernel，並更新編譯好的內核。

9. frizz HAL 移植

1. 將源碼移動至平台源碼包

$ mv frizz-sensor-hal <android-source>/hardware/frizz_hal
$ source build/envsetup.sh
$ lunch msm8916_64-userdebug
$ cd hardware/frizz_hal

//確認 frizz 固件路徑
$ vim FrizzSensor.cpp
//開發者能修改此路徑，底下是 HAL 預設抓取 frizz 固件路徑
#define D_FRIZZ_FIRMWARE_PATH "*/data/frizz/from.bin*" ///< frizz firmware full path

//編譯 HAL 碼源
$ mm –B

//HAL 編譯成功後會產生"sensors.msm8916.so"，能將檔案推入開發版內運行
$ adb remount
$ adb push sensors.msm8916.so /vendor/lib64/hw
// 請注意如果開發平台為 32 位元，請放入 lib 資料夾內

//或者將 system.img 的打包，更新 system.img
$ cd <android-source>
$ source build/envsetup.sh
$ lunch msm8916_64-userdebug
$ make snod

//更新 HAL system.img
$ adb reboot bootloader
$ fastboot flash system <android-source>/out/target/product/msm8916_64/system.img
$ fastboot reboot

//補充說明：透過系統開機自動掛載，因此需要修改 init.target.rc 檔案，請根據平台做相對應修改

2. 框架層測試

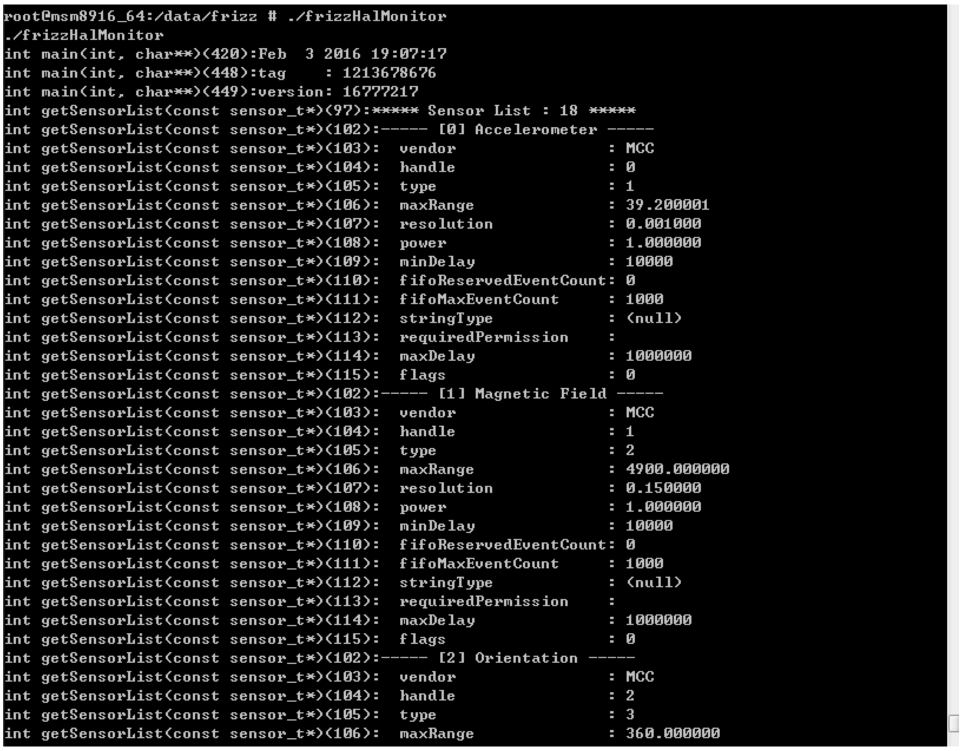
   Frizz HAL 測試工具源碼編譯

   讀取 frizz 寄存器測試

```
$ cd <android-source>/hardware/frizz_hal
$ cd frizzHalChecker
$ mm –B
```

   編譯成功後會產生"frizzHalMonitor"，將產生出來的檔案放於開發版內

```
$ adb push frizzHalMonitor /data/frizz
$ adb root
$ adb shell                    //切換至開發版
root@msm8916_64:/data/frizz:/ # cd /data/frizz/
root@msm8916_64:/data/frizz:/ # chmod 777 frizzHalMonitor
root@msm8916_64:/data/frizz:/ # ./frizzHalMonitor
```



```
root@msm8916_64:/data/frizz # ./frizzHalMonitor
./frizzHalMonitor
int main(int, char**)(420):Feb  3 2016 19:07:17
int main(int, char**)(448):tag    : 1213678676
int main(int, char**)(449):version: 16777217
int getSensorList(const sensor_t*)(97):***** Sensor List : 18 *****
int getSensorList(const sensor_t*)(102):----- [0] Accelerometer -----
int getSensorList(const sensor_t*)(103):  vendor                   : MCC
int getSensorList(const sensor_t*)(104):  handle                   : 0
int getSensorList(const sensor_t*)(105):  type                     : 1
int getSensorList(const sensor_t*)(106):  maxRange                 : 39.200001
int getSensorList(const sensor_t*)(107):  resolution               : 0.001000
int getSensorList(const sensor_t*)(108):  power                    : 1.000000
int getSensorList(const sensor_t*)(109):  minDelay                 : 10000
int getSensorList(const sensor_t*)(110):  fifoReservedEventCount: 0
int getSensorList(const sensor_t*)(111):  fifoMaxEventCount        : 1000
int getSensorList(const sensor_t*)(112):  stringType               : (null)
int getSensorList(const sensor_t*)(113):  requiredPermission       :
int getSensorList(const sensor_t*)(114):  maxDelay                 : 1000000
int getSensorList(const sensor_t*)(115):  flags                    : 0
int getSensorList(const sensor_t*)(102):----- [1] Magnetic Field -----
int getSensorList(const sensor_t*)(103):  vendor                   : MCC
int getSensorList(const sensor_t*)(104):  handle                   : 1
int getSensorList(const sensor_t*)(105):  type                     : 2
int getSensorList(const sensor_t*)(106):  maxRange                 : 4900.000000
int getSensorList(const sensor_t*)(107):  resolution               : 0.150000
int getSensorList(const sensor_t*)(108):  power                    : 1.000000
int getSensorList(const sensor_t*)(109):  minDelay                 : 10000
int getSensorList(const sensor_t*)(110):  fifoReservedEventCount: 0
int getSensorList(const sensor_t*)(111):  fifoMaxEventCount        : 1000
int getSensorList(const sensor_t*)(112):  stringType               : (null)
int getSensorList(const sensor_t*)(113):  requiredPermission       :
int getSensorList(const sensor_t*)(114):  maxDelay                 : 1000000
int getSensorList(const sensor_t*)(115):  flags                    : 0
int getSensorList(const sensor_t*)(102):----- [2] Orientation -----
int getSensorList(const sensor_t*)(103):  vendor                   : MCC
int getSensorList(const sensor_t*)(104):  handle                   : 2
int getSensorList(const sensor_t*)(105):  type                     : 3
int getSensorList(const sensor_t*)(106):  maxRange                 : 360.000000
```

   //測試 HAL 前，請先確保驅動已經能工作

10. frizz JNI

frizz 除了提供標準 android HAL 層讓系統能獲取資料外，另外在 android 尚未定義之功能，系統也能透過 NDK 的介面獲取 sensor 的資料

1. 將源碼移動至平台源碼包

```
$ mv frizz-jni <android-source>/package/apps
$ source build/envsetup.sh
$ lunch msm8916_64-userdebug
$ cd package/apps/frizz-jni
//編譯 JNI 碼源
$ mm –B

//JNI 編譯成功後會產生"FrizzManager.so"，能將檔案推入平台運行，但是這是屬於系統服務層，
因此需要在將檔名修改為"libFrizzManager.so"
$ cp FrizzManager.so libFrizzManager.so
$ adb remount

//推到 lib or lib64 請取決於需系統 host
$ adb push libFrizzManager.so /system/lib64
```

2. MegaChips 所提供的 JNI 介面是透過 ioctl 的方式直接呼叫驅動，因此在使用上，因此也能將 JNI 透過 import 的方式，直接將 JNI 的服務與 APK 綁訂一起，如此能省略第 1 點的說明

11. frizz sensor 輸出格式&命令下發格式

1. 實體 sensor 資料格式

a. Accelerometer Introduction

Data	Type	Description
Accelerometer	float	時間戳記
		[0]：[G] AccRaw Data X-Axis [1]：[G] AccRaw Data Y-Axis [2]：[G] AccRaw Data Z-Axis

b. Gyro Introduction

Data	Type	Description
Gyroscope	float	時間戳記
		[0]：[rad/s] Gyro Data X-Axis [1]：[rad/s] Gyro Data Y-Axis [2]：[rad/s] Gyro Data Z-Axis [3]：[rad/s] Gyro Temperature

c. Magnetic Introduction

Data	Type	Description
Magnetic	float	時間戳記
		[0]：[μT] Magnetic Data X-Axis [1]：[μT] Magnetic Data X-Axis [2]：[μT] Magnetic Data X-Axis

d. Pressure Introduction

Data	Type	Description
Pressure	float	時間戳記

		[0]：[hPa] Pressure Data

e. Light Sensor Introduction

Data	Type	Description
Light Sensor	float	時間戳記
		[0]：[lx] illuminance

f. Proximity Sensor Introduction

Data	Type	Description
Proximity Sensor	float	時間戳記
		[0]：[cm] Distance from object

g. Humidity Sensor Introduction

Data	Type	Description
Humidity Sensor	float	時間戳記
		[0]：[%] ambient relative humidity

h. Temperature Sensor Introduction

Data	Type	Description
Temperature Sensor	float	時間戳記
		[0]：[°C] ambient relative temperature

i. PPG Sensor Introduction

Data	Type	Description
PPG Sensor	float	時間戳記
		[0]：Defined by different specification

j. ECG Sensor Introduction

Data	Type	Description
ECG Sensor	float	Time stamp
		[0]：Defined by different specification

2. 虛擬 sensor 資料格式

a. Pedometer Data Output Introduction

Data	Type	Description
Pedometer Sensor	unsigned int	時間戳記
		[0]：算法所計算出的步數 [1]：步頻，實際步數輸出時間間隔，算法算出步數以及下一次算初步數的時間差，單位 msec

a-1. Pedometer Command → 清除計步器數值

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	88	01	Command ID：0x00 計步數值是持續累加的，能透過這個命令，將輸出數值清除為 0
Command	00	00	00	00	

a-2. Pedometer Command → 設定計步上報步數

Command Format	**Byte3**	**Byte2**	**Byte1**	**Byte0**	**Description**
Header	FF	81	88	02	Command ID：0x01 Parameter： 計步資料上報通知 0：計步數值存放於 frizz 的記憶體內，不會寫入 FIFO，Host 需透過命令來取計步資料 >0：計步演算法會以設定的數值更新。舉例說明，如果此參數設定 10，算法將以>=10 來更新計步數據。
Command	01	00	00	00	
Parameter[0]	00	00	00	01	

b. Gesture Data Output Introduction

Data	**Type**	**Description**
Gesture Sensor	unsigned int	時間戳記
		[0]：手勢結果 1：搖晃手部 2：抬手看手錶姿態 3：看完手錶，將手放下 4：翻轉手腕

b-1. Gesture Command → 手勢辨識功能開啟關閉

Command Format	**Byte3**	**Byte2**	**Byte1**	**Byte0**	**Description**
Header	FF	81	A6	02	Command ID：0x01 Parameter： 手勢判斷開啟/關閉 Bit0：搖晃手部 Bit1： 抬手看手錶姿態 Bit2：看完手錶，將手放下 Bit3： 翻轉手腕 能根據所需應用場景開啟或關閉
Command	01	00	00	00	
Parameter[0]	00	00	00	0F	

c. Motion Sensing Data Output Introduction

Data	**Type**	**Description**
Motion Sensing Sensor	unsigned int	時間戳記
		[0]：Motion status 0：停止模式(裝置放於桌上不動) 1：休息模式(停止不動) 2：走路模式 3：跑步模式

c-1. Motion Sensing Command → 設定停止多久進入停止模式

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	BD	02	Command ID：0x00 Parameter： 此命令用於判斷多久裝置將檢測為停止模式，預設為 5 分鐘，開發者能透過此命令改變判斷時間，單位：分鐘。
Command	00	00	00	00	
Parameter[0]	00	00	00	05	

c-2. Motion Sensing Command → Motion 狀態上報開啟/關閉

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	BD	02	Command ID：0x01 Parameter： Bit0：停止狀態 Bit1：休息狀態 Bit2：走路狀態 Bit3：跑步狀態 舉例說明，如果只需要判斷走路狀態，能將參數設定為 0x04。 預設為 0x0F，4 個狀態都開啟
Command	01	00	00	00	
Parameter[0]	00	00	00	0F	

d. Activity Data Output Introduction

Data	Type	Description
Motion Sensing Sensor	unsigned int	時間戳記
		[0]：Activity 狀態 0 : ACTIVITY_DEEPSLEEP 1 : ACTIVITY_SLEEP 2 : ACTIVITY_REST 3 : ACTIVITY_WALK 4 : ACTIVITY_RUN [1]：計步步數 [2]：進入睡眠時的翻身狀態

e. Fall Down Data Output Introduction

Data	Type	Description
Motion Sensing Sensor	unsigned int	時間戳記
		[0]：跌倒狀態，此數值為累加數值

e-1. Fall Down Command → 跌倒記數清除

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	BD	02	Command ID：0x00 跌倒數據會是持續累加數值，能透過此命令將跌倒記數清除
Command	00	00	00	00	

e-2. Fall Down Command → Fall down detection sensitivity setting.

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	BD	02	Command ID：0x01 Parameter： 跌倒偵測，靈敏度判斷 此明令提供了 1～6 個階段的靈敏度設定，1 是最靈敏，6 是最不靈敏。預設值為 5。
Command	01	00	00	00	
Parameter[0]	00	00	00	05	

f. Calorie Data Output Introduction

Data	Type	Description
Calorie Sensor	float	時間戳記
		[0]：卡路里輸出

f-1. Calorie Command → Calorie is reset automatically.

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	BD	02	Command ID：0x00 清除卡路里數據
Command	00	00	00	00	

f-2. Calorie Command → Height and weight parameter.

Command Format	Byte3	Byte2	Byte1	Byte0	Description
Header	FF	81	BD	03	Command ID：0x01 設定個人身高體重，卡路里 sensor 將使用此參數計算卡路里數據 Parameter[0]：身高，單位：cm Parameter[1]：體重，單位：kg
Command	01	00	00	00	
Parameter[0]	00	00	00	AA	
Parameter[1]	00	00	00	46	

g. Orientation Data Output Introduction

Data	Type	Description
Orientation Sensor	float	時間戳記
		[0]：Yaw 方位，當裝置朝向北方時為 0 度，東方為 90 度，南方為 180 度，西方為 270 度。 [1]：Roll 滾翻：裝置水平放置時為 0 度；裝置向右側傾斜（左側較高）時會漸增到 90 度，反之則漸減到 90 度。 [2]：Pitch 俯仰，裝置水平放置時為 0 度；裝置向前端傾斜（尾巴較高）時會漸增到 90 度，整個翻面則為 180 度。反之則漸減到 90 度，反向翻面則為-180 度。

h. PDR Data Output Introduction

Data	Type	Description
PDR Sensor	float	時間戳記
		[0]：計步步數。 [1]：跟原點 X，相對位置，單位公尺。 [2]：跟原點 Y，相對位置，單位公尺。

		[3]：X 方向的加速度，單位 m/sec。 [4]：Y 方向的加速度，單位 m/sec。 [5]：累積的總距離，單位公尺。

i. HR and Blood Pressure Data Output Introduction

Data	Type	Description
HR and Blood Pressure Sensor	Struct define	時間戳記
		[0]：Bit15~Bit0：版本= 0x0004 Bit31~Bit16：資料長度 = 0x0014 [1]：Bit15~Bit0：FFT 心跳 Bit31~Bit16：血壓高壓 [2]：Bit15~Bit0：血壓低壓 Bit31~Bit16：心跳 [3]：卡路里數值 [4]：Bit7~Bit0：資料狀態 0：PPG 數據 NG 1：心跳 OK, 2：血壓 OK Bit15~Bit8：預留 Bit23~Bit16：資料類型 BP_DETAIL_PARAM_TM_DETAIL BP_DETAIL_PARAM_FFT_DETAIL BP_DETAIL_PARAM_FFT_DETAIL2 Bit31~Bit24：預留

j. HR and Blood Pressure Learn Data Output Introduction

Data	Type	Description
HR and Blood Pressure Learn Sensor	Struct define	時間戳記
		[0]：Bit15~Bit0：版本 Bit31~Bit16：資料長度 [1]：Bit15~Bit0：sab[0], learning data Bit31~Bit16：sab[1], learning data [2]：Bit15~Bit0：sab[2], learning data Bit31~Bit16：dab [0], learning data [3]：Bit15~Bit0：dab [1], learning data Bit31~Bit16：dab [2], learning data [4~39]：st[24], dt[24], ps[24], learning source data [40]：Bit15~Bit0：學習血壓高壓最大值 Bit31~Bit16：學習血壓高壓最小值 [41]：Bit15~Bit0：學習血壓低壓最大值 Bit31~Bit16：學習血壓低壓最小值 [42]：Bit15~Bit0：學習進度指標，輸出 24 代表學習完成 Bit31~Bit16：預留 [43]：結束檔尾 0x12345678

j-1. HR and Blood Pressure Learn Command → 血壓數值設定

Command Format	**Byte3**	**Byte2**	**Byte1**	**Byte0**	**Description**
Header	FF	81	AF	0C	Command ID：0x00 Set BP parameter values. Parameter[0~3]：預留 Parameter[4]： Bit15~Bit0：預留 Bit31~Bit16：血壓高壓 Parameter[5]： Bit15~Bit0：血壓低壓 Parameter[6~9]：預留 Parameter[10]：結束檔尾 (0x12345678)
Command	00	00	00	00	
Parameter[0]	00	00	00	00	
Parameter[1]	00	00	00	00	
Parameter[2]	00	00	00	00	
Parameter[3]	00	00	00	00	
Parameter[4]	00	92	00	00	
Parameter[5]	00	00	00	41	
Parameter[6]	00	00	00	00	
Parameter[7]	00	00	00	00	
Parameter[8]	00	00	00	00	
Parameter[9]	00	00	00	00	
Parameter[10]	12	34	56	78	

j-2. HR and Blood Pressure Learn Command → Blood Pressure parameter setting command.

Command Format	**Byte3**	**Byte2**	**Byte1**	**Byte0**	**Description**
Header	FF	81	AF	2D	Command ID：0x01 [0]：Bit15~Bit0：版本 Bit31~Bit16：資料長度 [1]：Bit15~Bit0：sab[0] Bit31~Bit16：sab[1] [2]：Bit15~Bit0：sab[2] Bit31~Bit16：dab [0] [3]：Bit15~Bit0：dab [1] Bit31~Bit16：dab [2] [4~39]：st[24], dt[24], ps[24] [40]：Bit15~Bit0：學習血壓高壓最大值 Bit31~Bit16：學習血壓高壓最小值 [41]：Bit15~Bit0：學習血壓低壓最大值 Bit31~Bit16：學習血壓低壓最小值 [42]：Bit15~Bit0：學習進度指標 Bit31~Bit16：預留 [43]：結束檔尾 0x12345678
Command	01	00	00	00	
Parameter[0] ～ Parameter[43]	將 frizz 血壓學習的 sensor 資料填入				

12. 參考文獻

Table 4-1 Reference document

No.	Name	Overview	Publisher
1			
2			
3			
4			
5			

13. 歷史紀錄

Date	Revision	Description
Dec,23,2015	1.0	Initial Release