|  |  |  |
| --- | --- | --- |
| Driver Version | Description | Author |
| V3.0 | Fix Raw A+G issues.  Fix Apex issues.  Key API function optimization | TDK Invensense  Aaron Tan |
| V2.0 | Raw A+G Optimization  Support APEX Features(Pedo, Freefall…) | TDK Invensense  Aaron Tan |
| V1.0 | Support Raw A+G and Selftest | Aaron Tan  TDK Invensense |

1. ***Key APIs Notes***

void inv\_icm4x6xx\_set\_serif

***Function***:

*Set serial interface to read/write chip register in driver.*

***Parameters***:

int (\*read)(void \*, uint8\_t, uint8\_t \*, uint32\_t);

*The function pointer of read API depending on platform system via SPI or I2C.*

int (\*write)(void \*, uint8\_t, uint8\_t \*, uint32\_t);

The function pointer of write API depending on platform system via SPI or I2C.

***Return***:

*NULL*.

void inv\_icm4x6xx\_set\_delay

***Function***:

*Set delay API for register read/write setting.*

***Parameters***:

void (\*delay\_ms)(uint16\_t);

*The function pointer of delay\_ms depending on platform system.*

void (\*delay\_us)(uint16\_t);

The function pointer of delay\_us depending on platform system.

***Return***:

*NULL*.

int inv\_icm4x6xx\_initialize

***Function***:

Sensor initialization: Get WHOAMI, ACC/Gyro/FIFO initial setting etc.

***Parameters***:

NULL

***Return***:

*0: Success Others: Fail.*

int inv\_icm4x6xx\_acc\_enable

***Function***:

Power on Acc.

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_acc\_disable

***Function***:

*Power off Acc.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_acc\_set\_rate

***Function***:

*Set Acc ODR and FIFO Watermark.*

***Parameters***:

float odr\_hz;

*Acc ODR Value. Unit: Hz.*

uint16\_t watermark;

*FIFO Watermark Value. PS: Just work under FIFO Mode.*

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_gyro\_enable

***Function***:

*Power on Gyro.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_gyro\_disable

***Function***:

*Power off Gyro.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_gyro\_set\_rate

***Function***:

*Set Gyro ODR and FIFO Watermark.*

***Parameters***:

float odr\_hz

*Gyro ODR Value. Unit: Hz.*

uint16\_t watermark

*FIFO Watermark Value. PS: Just work under FIFO Mode.*

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_get\_rawdata

***Function***:

*Get sample data from sensor.*

***Parameters***:

struct \*accGyroDataPacket

*The packet pointer to save sample data from sensor.*

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_acc\_selftest

***Function***:

*Run acc selftest case.*

***Parameters***:

bool \*result

*The bool pointer to save test result.* ***1: Test OK. 0: Test Fail.***

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_gyro\_selftest

***Function***:

*Run gyro selftest case.*

***Parameters***:

bool \*result

*The bool pointer to save test result.* ***1: Test OK. 0: Test Fail.***

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_pedometer\_enable

***Function***:

*Enable hw pedometer.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_pedometer\_disable

***Function***:

*Disable hw pedometer.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_pedometer\_get\_stepCnt

***Function***:

*Get step counter.*

***Parameters***:

uint64\_t \*step\_cnt;

uint64\_t pointer to save step count.

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_lowG\_enable

***Function***:

*Enable lowG/freeFall feature.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_lowG\_disable

***Function***:

*Disable lowG/FreeFall.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_highG\_enable

***Function***:

*Enable highG feature.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_highG\_disable

***Function***:

*Disable highG feature.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_low\_high\_G\_get\_event

***Function***:

*Get lowG or highG event.*

***Parameters***:

uint8\_t \*event;

uint8\_t pointer to save lowG and highG event. ***0x04: highG event. 0x02:lowG event***

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_wom\_enable

***Function***:

*Enable wom feature.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_wom\_disable

***Function***:

*Disable wom feature.*

***Parameters***:

NULL

***Return***:

*0: Success. Others: Fail.*

int inv\_icm4x6xx\_wom\_get\_event

***Function***:

*Get wom event.*

***Parameters***:

bool \*event;

bool pointer to get wom event.

***Return***:

*0: Success. Others: Fail.*

1. ***Key Setting in Driver***

|  |  |  |
| --- | --- | --- |
| Marco | Description | Value (default as red) |
| FIFO\_WM\_MODE\_EN | Switch FIFO and DRDY | 1: Fifo mode 0: DRDY mode |
| SPI\_MODE\_EN | Switch SPI and I2C | 1: SPI 0: I2C |
| INT\_LATCH\_EN | Switch INT Mode | 1: Latch mode 0:Pulse mode |
| INT\_ACTIVE\_HIGH\_EN | Switch INT Active Level | 1: Active High 0: Active Low |
| SUPPORT\_SELFTEST | Selftest feature en/dis | 1: Enable 0: Disable |
| SUPPORT\_PEDOMETER | Hw pedo feature en/dis | 1: Enable 0: Disable |
| SUPPORT\_LOWG | LowG feature en/dis | 1: Enable 0: Disable |
| SUPPORT\_HIGHG | HighG feature en/dis | 1: Enable 0: Disable |
| SUPPORT\_WOM | Wom feature en/dis | 1: Enable 0: Disable |
| SENSOR\_DIRECTION | Index of sensor direction | 0 ~ 7 (0) |
| SENSOR\_LOG\_LEVEL | Driver log level | 0 ~ 4 (3) |
| SENSOR\_REG\_DUMP | Register dump | 1: Enable 0: Disable |

1. ***Sample Code for customer***

|  |
| --- |
| ***Feature: Raw A+G***  int ret = 0;  inv\_icm4x6xx\_set\_serif(idd\_io\_hal\_read\_reg, idd\_io\_hal\_write\_reg);  inv\_icm4x6xx\_set\_delay(sm\_delay\_ms, sm\_delay\_us);  ret += inv\_icm4x6xx\_initialize();  ret += inv\_icm4x6xx\_acc\_set\_rate(200, 2);//200Hz, watermark 2.  ret += inv\_icm4x6xx\_acc\_enable();  ret += inv\_icm4x6xx\_gyro\_set\_rate(100, 4);//100Hz, watermark 4.  ret += inv\_icm4x6xx\_gyro\_enable();  /\* Get A+G data in irq handler \*/  struct accGyroDataPacket datapacket;  memset(&datapacket, 0, sizeof(struct accGyroDataPacket));  inv\_icm4x6xx\_get\_rawdata(&datapacket);  ret += inv\_icm4x6xx\_acc\_disable();  ret += inv\_icm4x6xx\_gyro\_disable(); |

|  |
| --- |
| ***Feature: Selftest***  int ret = 0;  inv\_icm4x6xx\_set\_serif(idd\_io\_hal\_read\_reg, idd\_io\_hal\_write\_reg);  inv\_icm4x6xx\_set\_delay(sm\_delay\_ms, sm\_delay\_us);  bool result = false;  ret = inv\_icm4x6xx\_acc\_selftest(&result);  ret = inv\_icm4x6xx\_gyro\_selftest(&result); |

|  |
| --- |
| ***Feature: Pedometer***  int ret = 0;  inv\_icm4x6xx\_set\_serif(idd\_io\_hal\_read\_reg, idd\_io\_hal\_write\_reg);  inv\_icm4x6xx\_set\_delay(sm\_delay\_ms, sm\_delay\_us);  ret += inv\_icm4x6xx\_initialize();  ret += inv\_icm4x6xx\_pedometer\_enable();  /\* get step count in irq handle \*/  uint64\_t step\_cnt;  ret += inv\_icm4x6xx\_pedometer\_get\_stepCnt(&step\_cnt);  ret += inv\_icm4x6xx\_pedometer\_disable (); |

|  |
| --- |
| ***Feature: LowG/HighG.***  int ret = 0;  inv\_icm4x6xx\_set\_serif(idd\_io\_hal\_read\_reg, idd\_io\_hal\_write\_reg);  inv\_icm4x6xx\_set\_delay(sm\_delay\_ms, sm\_delay\_us);  ret += inv\_icm4x6xx\_initialize();  ret += inv\_icm4x6xx\_lowG\_enable();  ret += inv\_icm4x6xx\_highG\_enable();  /\* get lowG event in irq handle \*/  uint8\_t \*event;  ret += inv\_icm4x6xx\_low\_high\_G\_get\_event(&event);  if(event & 0x04)  //lowG detected.  If(event & 0x02)  //highG detectd.  ret += inv\_icm4x6xx\_lowG\_disable ();  ret += inv\_icm4x6xx\_highG\_disable(); |

|  |
| --- |
| ***Feature: WOM.***  int ret = 0;  inv\_icm4x6xx\_set\_serif(idd\_io\_hal\_read\_reg, idd\_io\_hal\_write\_reg);  inv\_icm4x6xx\_set\_delay(sm\_delay\_ms, sm\_delay\_us);  ret += inv\_icm4x6xx\_initialize();  ret += inv\_icm4x6xx\_wom\_enable();  /\* get wom event in irq handle \*/  bool event;  ret += inv\_icm4x6xx\_wom\_get\_event(&event);  ret += inv\_icm4x6xx\_wom \_disable (); |