**ICM42688**

背景：使用nrf52832芯片，通过SPI和传感器icm42688通信；

1、void app\_imu\_init(void)

{

    nrf\_drv\_spi\_config\_t spi\_config = NRF\_DRV\_SPI\_DEFAULT\_CONFIG;

    //spi\_config.ss\_pin   = SPI\_SS\_PIN;

    spi\_config.miso\_pin = SPI\_MISO\_PIN;

    spi\_config.mosi\_pin = SPI\_MOSI\_PIN;

    spi\_config.sck\_pin  = SPI\_SCK\_PIN;

    spi\_config.mode = NRF\_DRV\_SPI\_MODE\_3;

    spi\_config.frequency = NRF\_DRV\_SPI\_FREQ\_8M;

    APP\_ERROR\_CHECK(nrf\_drv\_spi\_init(&spi, &spi\_config, NULL, NULL));

    icm42688\_sspin\_init();              //初始化片选引脚

    spi\_interface\_init(&spi);

    NRF\_LOG\_INFO("spi init success!");

    NRF\_LOG\_FLUSH();

    if (sensor\_init(&sensor, icm42688\_sensor\_event\_cb) != SNSR\_STATUS\_OK)

    {

        NRF\_LOG\_INFO("imu init result = %d", sensor.status);

        NRF\_LOG\_FLUSH();

    }

    if (sensor\_set\_config(&sensor) != SNSR\_STATUS\_OK)

    {

        NRF\_LOG\_INFO("imu configuration result = %d", sensor.status);

        NRF\_LOG\_FLUSH();

    }

    if (icm42688\_sensor\_set\_wom(&sensor) != SNSR\_STATUS\_OK)

    {

        NRF\_LOG\_INFO("imu set\_r2w result = %d", sensor.status);

        NRF\_LOG\_FLUSH();

    }

    icm42688\_in\_gpio\_init();                //设置中断脚，上下拉触发

    NRF\_LOG\_INFO("imu type is %s", SNSR\_NAME);

    NRF\_LOG\_INFO("sample rate set at %d%s", SNSR\_SAMPLE\_RATE, SNSR\_SAMPLE\_RATE\_UNIT\_STR);

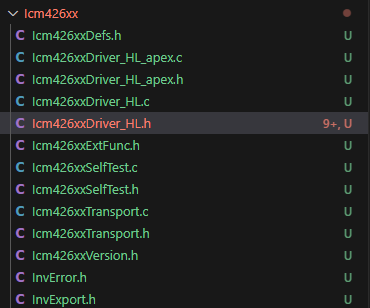
    NRF\_LOG\_INFO("SPI started.");

    NRF\_LOG\_FLUSH();

}

1. 首先是找到5个引脚，SCL(时钟引脚)，SDI(主机输入)，SDO(主机输出)，SS(片选引脚)，INT(中断引脚)。
2. 然后通过平台nrf\_drv\_spi.h接口，设置对应引脚
3. nrf\_drv\_spi\_config\_t spi\_config = NRF\_DRV\_SPI\_DEFAULT\_CONFIG;
4. //spi\_config.ss\_pin   = SPI\_SS\_PIN;
5. spi\_config.miso\_pin = SPI\_MISO\_PIN;
6. spi\_config.mosi\_pin = SPI\_MOSI\_PIN;
7. spi\_config.sck\_pin  = SPI\_SCK\_PIN;
8. spi\_config.mode = NRF\_DRV\_SPI\_MODE\_3;
9. spi\_config.frequency = NRF\_DRV\_SPI\_FREQ\_8M;
10. APP\_ERROR\_CHECK(nrf\_drv\_spi\_init(&spi, &spi\_config, NULL, NULL));
11. icm42688\_sspin\_init();              //初始化片选引脚

3、移植ICM42688官方库



4、然后初始化ICM42688

    if (sensor\_init(&sensor, icm42688\_sensor\_event\_cb) != SNSR\_STATUS\_OK)

    {

        NRF\_LOG\_INFO("imu init result = %d", sensor.status);

        NRF\_LOG\_FLUSH();

    }

其中通过宏，定义了另外一个名字

#define sensor\_init        icm42688\_sensor\_init

实质是初始化**icm42688\_sensor\_init()**函数。

其实该接口是通过inv\_icm426xx\_init()接口里的参数类型重新定义一个新的结构体类型，那外围的接口参数就能适配里面接口参数了

即是：

int inv\_icm426xx\_init(struct inv\_icm426xx \* s, struct inv\_icm426xx\_serif \* serif, void (\*sensor\_event\_cb)(inv\_icm426xx\_sensor\_event\_t \* event))

重新定义一个结构体

struct sensor\_device\_t {

#if SNSR\_TYPE\_BMI160

    struct bmi160\_dev device;

#elif SNSR\_TYPE\_ICM42688

    struct inv\_icm426xx device;

    struct inv\_icm426xx\_serif serif;

#endif

    volatile int status;

};

inv\_icm426xx\_configure\_fifo()

inv\_icm426xx\_get\_who\_am\_i()

//(struct inv\_icm426xx \* s, struct inv\_icm426xx\_serif \* serif, void (\*sensor\_event\_cb)(inv\_icm426xx\_sensor\_event\_t \* event))

int icm42688\_sensor\_init(struct sensor\_device\_t \*sensor, void (\*sensor\_event\_cb)(inv\_icm426xx\_sensor\_event\_t \* event)) {

    /\* Init ICM \*/

    memset(&sensor->serif, 0, sizeof(sensor->serif));

    sensor->serif.context   = 0;        /\* no need \*/

    sensor->serif.read\_reg  = icm42688\_spi\_read;

    sensor->serif.write\_reg = icm42688\_spi\_write;

    sensor->serif.max\_read  = SNSR\_COM\_BUF\_SIZE-1;

    sensor->serif.max\_write = SNSR\_COM\_BUF\_SIZE-1;

    sensor->serif.serif\_type = ICM426XX\_UI\_SPI4;

    sensor->status = SNSR\_STATUS\_OK;

    // Init and disable FIFO

    sensor->status = inv\_icm426xx\_init(&sensor->device, &sensor->serif, sensor\_event\_cb);        //主要是看这里的函数参数，然后再定义自己封装的函数，根据这里的结构体

    sensor->status |= inv\_icm426xx\_configure\_fifo(&sensor->device, INV\_ICM426XX\_FIFO\_DISABLED);  //再定义个大的结构体，作为自己封装的函数参数

    uint8\_t who\_am\_i;

    sensor->status |= inv\_icm426xx\_get\_who\_am\_i(&sensor->device, &who\_am\_i);

    if (who\_am\_i != ICM\_WHOAMI)

    {

        NRF\_LOG\_INFO("get who\_am\_i %x, not %s", who\_am\_i, SNSR\_NAME);

        sensor->status |= INV\_ERROR;

    }

    return sensor->status;

}