MT6835 ใช้งานร่วมกับ STM32H723 Nucleo

ใช้งานไลบรารี่

.h	.c
mt6835.h	mt6835.c

1. ส่วน Header และการ Include ไฟล์

```
#include "main.h"
#include "memorymap.h"
#include "spi.h"
#include "usart.h"
#include "gpio.h"
#include "mt6835.h"
#include "string.h"
#include "stdio.h"
```

- นำเข้าไลบรารีที่จำเป็นสำหรับ STM32
- mt6835.h เป็นไลบรารีของเซ็นเซอร์ MT6835

2. การตั้งค่าการส่งข้อมูลผ่าน UART

```
int _write(int file, char *ptr, int len) {
    HAL_UART_Transmit(&huart3, (uint8_t*) ptr, len, HAL_MAX_DELAY);
    return len;
}
```

- ฟังก์ชัน _write ถูกใช้เพื่อช่วยให้ printf สามารถส่งข้อมูลออกทาง UART3 ได้

3. การกำหนดขา CS (Chip Select) สำหรับ SPI

```
#define SPI_INSTANCE    hspi3
#define SPI_M1_CS       SPI3_M1_Pin
#define SPI_M1_CS_PORT  SPI3_M1_GPIO_Port
#define SPI_M2_CS       SPI3_M2_Pin
#define SPI_M2_CS_PORT  SPI3_M2_GPIO_Port
```

- กำหนดตัวแปรสำหรับใช้ SPI3
- กำหนดขา CS สำหรับเซ็นเซอร์ตัวที่ 1 (SPI_M1_CS) และเซ็นเซอร์ตัวที่ 2 (SPI_M2_CS)

4. ฟังก์ชันควบคุมขา CS ของเซ็นเซอร์

```
static void mt6835_cs_control(mt6835_cs_state_enum_t state) {
    if (state == MT6835_CS_HIGH) {
        HAL_GPIO_WritePin(SPI_M1_CS_PORT, SPI_M1_CS, GPIO_PIN_SET);
    } else {
        HAL_GPIO_WritePin(SPI_M1_CS_PORT, SPI_M1_CS, GPIO_PIN_RESET);
    }
}
static void mt6835_cs2_control(mt6835_cs_state_enum_t state) {
    if (state == MT6835_CS_HIGH) {
        HAL_GPIO_WritePin(SPI_M2_CS_PORT, SPI_M2_CS, GPIO_PIN_SET);
    } else {
        HAL_GPIO_WritePin(SPI_M2_CS_PORT, SPI_M2_CS, GPIO_PIN_RESET);
    }
}
```

- ฟังก์ชัน mt6835_cs_control และ mt6835_cs2_control ใช้ควบคุมขา CS ของเซ็นเซอร์แต่ละตัว

```
void mt6835_link_spi_cs2_control(mt6835_t *mt6835, void (*spi_cs_control)
(mt6835_cs_state_enum_t state)) {
    if (mt6835 == NULL) {
        MT6835_DEBUG("%s mt6835 object is null", TAG);
        return;
    }
    if (spi_cs_control == NULL) {
        MT6835_DEBUG("%s mt6835 object use default spi_cs_control(null)", TAG);
        mt6835->func.spi_cs_control = mt6835_cs2_control;
    } else {
        mt6835->func.spi_cs_control = spi_cs_control;
    }
}
```

- เพิ่มฟังก์ชันคุมขา CS ใน mt6385.c

5. ฟังก์ชันส่งและรับข้อมูลผ่าน SPI

```
static void mt6835_spi_send_recv(uint8_t *tx_buf, uint8_t *rx_buf, uint8_t len) {
    HAL_StatusTypeDef status = HAL_OK;
    status = HAL_SPI_TransmitReceive_IT(&SPI_INSTANCE, tx_buf, rx_buf, len);
    if (status != HAL_OK) {
        printf("spi send_recv failed %d\n\r", status);
        return;
    }
    uint32_t tickstart = HAL_GetTick();
    while (HAL_SPI_GetState(&SPI_INSTANCE) != HAL_SPI_STATE_READY) {
        if (HAL_GetTick() - tickstart > 1) {
            printf("spi send_recv timeout\n\r");
            return;
```

```
        }
    }
}
```

- ใช้ HAL_SPI_TransmitReceive_IT เพื่อส่งและรับข้อมูลผ่าน SPI
- ตรวจสอบว่าการรับ-ส่งสำเร็จหรือไม่ ถ้าไม่สำเร็จจะแสดงข้อความ error

6. การกำหนดตัวแปรสำหรับมุมหมุน

```
uint32_t raw_angle_1 = 0, raw_angle_2 = 0;
float radian_angle_1 = 0.0f, radian_angle_2 = 0.0f;
float degree_angle_1 = 0.0f, degree_angle_2 = 0.0f;
uint8_t id_1, id_2;
```

- ตัวแปร raw_angle_X เก็บค่ามุมดิบจากเซ็นเซอร์
- ตัวแปร radian_angle_X และ degree_angle_X ใช้เก็บค่ามุมในหน่วยเรเดียนและองศา

7. การเริ่มต้นทำงานของ STM32 ใน main()

```
HAL_Init();
SystemClock_Config();
MX_GPIO_Init();
MX_SPI3_Init();
MX_USART3_UART_Init();
```

- HAL_Init() -> รีเซ็ตฮาร์ดแวร์และเปิดใช้งาน HAL
- SystemClock_Config() -> ตั้งค่านาฬิกาของระบบ
- MX_GPIO_Init(), MX_SPI3_Init(), MX_USART3_UART_Init() -> ตั้งค่า GPIO, SPI3 และ UART3

8. การสร้างและตั้งค่าเซ็นเซอร์ MT6835

```
mt6835_t *mt6835_1;
mt6835_t *mt6835_2;
mt6835_1 = mt6835_create();
mt6835_2 = mt6835_create();

mt6835_link_spi_cs_control(mt6835_1, mt6835_cs_control);
mt6835_link_spi_cs_control(mt6835_2, mt6835_cs2_control);

mt6835_link_spi_send_recv(mt6835_1, mt6835_spi_send_recv);
mt6835_link_spi_send_recv(mt6835_2, mt6835_spi_send_recv);

mt6835_enable_crc_check(mt6835_1);
mt6835_enable_crc_check(mt6835_2);
```

- สร้างอินสแตนซ์ของเซ็นเซอร์ mt6835_1 และ mt6835_2
- เชื่อมโยงฟังก์ชันควบคุม CS และ SPI กับเซ็นเซอร์แต่ละตัว
- เปิดใช้งาน CRC Check เพื่อป้องกันข้อผิดพลาด

9. การอ่านค่ามุมจากเซ็นเซอร์ในลูป

```
while (1) {
    raw_angle_1 = mt6835_get_raw_angle(mt6835_1,MT6835_READ_ANGLE_METHOD_NORMAL);
    radian_angle_1 = raw_angle_1 * (M_PI * 2.0f) / MT6835_ANGLE_RESOLUTION;
    degree_angle_1 = raw_angle_1 * (360.0f / MT6835_ANGLE_RESOLUTION);

    raw_angle_2 = mt6835_get_raw_angle(mt6835_2,MT6835_READ_ANGLE_METHOD_NORMAL);
    radian_angle_2 = raw_angle_2 * (M_PI * 2.0f) / MT6835_ANGLE_RESOLUTION;
    degree_angle_2 = raw_angle_2 * (360.0f / MT6835_ANGLE_RESOLUTION);
```

- อ่านค่ามุมดิบจากเซ็นเซอร์
- แปลงค่ามุมเป็น เรเดียน และ องศา

10. ตรวจสอบความถูกต้องของข้อมูล

```
if (!mt6835_1->crc_res) {
    printf("CRC error on MT6835 1\n\r");
}
if (!mt6835_2->crc_res) {
    printf("CRC error on MT6835 2\n\r");
}
```

- ตรวจสอบว่า CRC ของเซ็นเซอร์ถูกต้องหรือไม่

11. แสดงผลข้อมูลผ่าน UART

```
printf("Raw Angle 1: %lu, Radian 1: %f, degree_angle1: %.2f deg\n\r",
    raw_angle_1, radian_angle_1, degree_angle_1);
printf("Raw Angle 2: %lu, Radian 2: %f, degree_angle2: %.2f deg\n\r",
    raw_angle_2, radian_angle_2, degree_angle_2);
HAL_Delay(500);
```

- ส่งค่ามุมที่อ่านได้ออกทาง UART
- ใช้ HAL_Delay(500) เพื่อรอ 500 มิลลิวินาที

สรุป

- ใช้ SPI3 ติดต่อกับเซ็นเซอร์ MT6835
- อ่านค่ามุมจากเซ็นเซอร์ 2 ตัว
- แปลงค่าเป็นเรเดียนและองศา

- แสดงผลออกทาง UART3