

1 Hướng dẫn sử dụng source code Drone



Mình viết ra file này để hỗ trợ ae Việt Nam ta:

1.1 main_quad là source của khối xử lý

+core:Thư mục thư viện

+Inc:chứa fild header

+main.h Thư viện

+mpu6050.h Thư viện IMU + giải thuật Kalman

+nRF24L01.h Thư viện nRF24L01 điều khiển

+Src:chứa file soure

+main.c Source trương trình

+mpu6050.c Source IMU + giải thuật Kalman cho Drone

+nRF24L01.c Source nRF24L01 điều khiển Drone

+Debug:Thư mục debug

+core :chứa file liên quan

+main_quad.elf: source sau khi build sẽ sinh ra (dùng để nạp trương trình và debug bằng cube IDE))

+main_quad.hex: source sau khi build sẽ sinh ra (chỉ dùng để nạp trương trình stlink)

1.2 control_stmf103 là source của khối điều khiển

+core:Thư mục thư viện

+Inc:chứa fild header

+main.h Thư viện

+nRF24L01.h Thư viện nRF24L01 điều khiển

+Src:chứa file soure

+main.c Source trương trình

+nRF24L01.c Source nRF24L01 điều khiển Drone

+Debug:Thư mục debug

+core :chứa file liên quan

+main_quad.elf: source sau khi build sẽ sinh ra (dùng để nạp trương trình và debug bằng cube IDE))

+main_quad.hex: source sau khi build sẽ sinh ra (chỉ dùng để nạp chương trình stlink)

2 Trong main.c

*****Chương trình chính*****

1: Khởi tạo #define

```
35
36  /* Private macro -----*/
37  /* USER CODE BEGIN PM */
38  float Max_PWM = 1800;
39  float Max_THR = 1450;
40  float Min_PWM = 1050;
41  #define thr_start 1300
42  #define speed_up_down 0.1
43  float throttle = 1000;
44  float throttle_PID = 1000;
45  float pid_throttle = 0;
46  float acc_total_vector = 4250;
47  #define speed_LRFB 100
48  #define KP_xy 0.92 //0.88
49  #define KI_xy 0.022 //0.02
50  #define KD_xy 16 //15
51  #define KP_z 3.2 //3
52  #define KI_z 0.019 //0.018
53  #define KD_z 0
54  #define KP_dc 0.2
55  #define KI_dc 0.001//0.001
56  #define KD_dc 2//3
57  float Setting_dacao = 150.0;
58  /* USER CODE END PM */
```

```

68  /* USER CODE BEGIN PV */
69  uint16_t PWM[4]={1000,1000,1000,1000};
70  NRF24L01_config_TypeDef nrf_rx_cfg;
71  uint8_t data_rx_real = 0 ;
72  uint8_t button_press = 0;
73  uint8_t old_data_rx_real = 0;
74  uint16_t count_data_rx_real = 0;
75  MPU6050_t MPU6050_Data;
76  PIDSingle PID_ROLL;
77  PIDSingle PID_PITCH;
78  PIDSingle PID_YAW;
79  PIDSingle PID_HG;
80  uint32_t loop_time = 0;
81  uint32_t loop_time_hacanh = 0;
82  uint32_t timer_before = 0;
83  float ax_cal_tt=0,ay_cal_tt=0;
84  float gyro_roll_cal_tt=0,gyro_pitch_cal_tt= 0,gyro_yaw_cal_tt= 0;
85  float angle_roll_acc=0, angle_pitch_acc=0, angle_pitch=0, angle_roll=0;
86  float pitch_level_adjust = 0,roll_level_adjust = 0; //Set the pitch angle correction to zero
87  float pid_pitch_setpoint = 0,pid_roll_setpoint = 0,pid_yaw_setpoint = 0; //Set point
88  // INIT kh value
89  uint32_t IC_Val = 0;
90  float_t Distance = 0;
91  float_t Kalman_Distance = 5;
92  uint8_t ok = 0;
93  uint8_t first_read_hc = 0;
94  bool loi_xxx = false;
95  bool is_running = false;
96  bool is_giudocao = false;
97  bool hacanh = false;
98  bool firt_start = true;
99  /* USER CODE END PV */

```

Dùng để lưu giá trị nhận được từ cảm biến là xử lý giá trị đưa ra Drone

3 Hàm

HAL_GPIO_EXTI_Callback :Ngắt ngoài
 set_val_for_nfr24 : Set giá trị nfr24 giống nhau (truyền và xử lý)
 quad_up :Điều khiển Drone bay lên
 quad_down : Điều khiển Drone hạ
 quad_right :Drone sang phải
 quad_left: Drone sang trái
 quad_front: Drone bay về phía trước
 quad_behind:Drone bay về phía sau
 quad_giudocao :giữ cho drone ở 1 độ cao
 hacanh_quad:hạ cánh khẩn cấp
 quad_stop:Dừng lại đột ngột
 quad_start:Khởi động động cơ
 quad_reset:Reset lại các biến
 calibrate_gyro:trả về giá trị cân bằng cho IMU pmu6050
 correct_data_and_calibrate_3truc:Chỉnh 3 trục về đúng tọa độ và trừ về giá trị mặc định

calculate_agl_roll_pitch :Tính toán ra góc và độ trượt của 3 trục
Roll,Pitch,Yaw

calculate_setpoint_pid:Tính toán PID cho cả 3 trục

read_hc05: cân bằng xong thì giữa độ cao(chưa hoàn thiện-vì hc05 k đáp ứng đủ nhu cầu)

read_hc05_and_fillter: lọc giá trị đọc từ IMU mpu6050(LỌC THẤP VÀ KALMAN)
(2 CÁCH LỌC)

check_looptime :SET THỜI GIAN MỖI LẦN XỬ LÝ LÀ 4 MS

main :XỬ LÝ TRUNG TÂM(SETTING GIÁ TRỊ BAN ĐẦU CHO hệ THỐNG)

```
332 int main(void)
333 {
334     /* USER CODE BEGIN 1 */
335     Int_PID_Integrator(&PID_ROLL, KP_xy , KI_xy , KD_xy );
336     Int_PID_Integrator(&PID_PITCH, KP_xy , KI_xy , KD_xy );
337     Int_PID_Integrator(&PID_YAW, KP_z , KI_z , KD_z );
338     Int_PID_Integrator(&PID_HG, KP_dc , KI_dc , KD_dc );
339     /* USER CODE END 1 */
340
341     /* MCU Configuration-----*/
342
343     /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
344     HAL_Init();
345
346     /* USER CODE BEGIN Init */
347
348     /* USER CODE END Init */
349
350     /* Configure the system clock */
351     SystemClock_Config();
352
353     /* USER CODE BEGIN SysInit */
354
355     /* USER CODE END SysInit */
356
357     /* Initialize all configured peripherals */
358     MX_GPIO_Init();
359     MX_I2C2_Init();
360     MX_SPI1_Init();
361     MX_TIM1_Init();
362     MX_TIM2_Init();
363     /* USER CODE BEGIN 2 */
364     HAL_GPIO_WritePin(GPTOC, GPTO_PIN_13, GPIO_PIN_RESET);
```

4 while (1):VÒNG LẶP XỬ LÝ

1. Đọc giá trị từ điều khiển

```
-----READ DATA-----
a. mbal_NRF24L01_GetData(&nrf_rx_cfg, &data_rx_real);
```

2. Đọc giá trị cảm biến

```
a. MPU6050_Read_All(&hi2c2, &MPU6050_Data); //read data
```

3. Chỉnh lại 3 trục về đúng quỹ đạo

```
a. correct_data_and_calibrate_3truc();
```

4. Tính toán góc 3 trục

```
a. calculate_agl_roll_pitch();
```

5. Đưa giá trị thiết lập của góc vào Drone(muốn cân bằng thì góc của 3 trục sẽ sắp xỉ 0)

a. `calculate_setpoint_pid();`

6. Đọc cảm biến độ cao

a. `read_hc05_and_fillter();`

7. Lọc xem người điều khiển muốn làm gì(xem trong báo cáo)

```

/*-----Check right left front behind-----*/
/* USER CODE END WHILE */

/* USER CODE BEGIN 3 */
button_press = 0;
for(button_press = 0 ;button_press < 5 ; button_press ++){
    if((data_rx_real>>button_press)&1){
        break;
    }
    if(button_press == 4){
        if(!((data_rx_real>>button_press)&1)){
            button_press = 10;
            break;
        }
    }
}

if(button_press==3) { quad_stop(); }
else if(button_press==2){ if (hacanh == false){quad_start(); }}
else if(button_press==0){ quad_down(); }
else if(button_press==1){ if (hacanh == false){quad_up(); }}
else {}

```

a.

8. Tính toán PID

```

/*-----Operation Run Motor -----*/
PID_Calculation(&PID_ROLL,pid_roll_setpoint,MPU6050_Data.Gx_roll );
PID_Calculation(&PID_PITCH,pid_pitch_setpoint,MPU6050_Data.Gy_pitch );
PID_Calculation(&PID_YAW,pid_yaw_setpoint,MPU6050_Data.Gz_yaw );
PID_Calculation_thr(&PID_HG,Setting_docao,Kalman_Distance);

```

a.

9. Chạy trường trình động cơ theo PID

```

if((is_running== true)){
    PWM[0] = (uint16_t)(throttle_PID + PID_ROLL.pid_result + PID_PITCH.pid_result + PID_YAW.pid_result);
    PWM[1] = (uint16_t)(throttle_PID + PID_ROLL.pid_result - PID_PITCH.pid_result - PID_YAW.pid_result);
    PWM[2] = (uint16_t)(throttle_PID - PID_ROLL.pid_result + PID_PITCH.pid_result - PID_YAW.pid_result);
    PWM[3] = (uint16_t)(throttle_PID - PID_ROLL.pid_result - PID_PITCH.pid_result + PID_YAW.pid_result);
    for(int i=0 ; i <4 ; i++){
        if (PWM[i] > Max_PWM) {PWM[i] = Max_PWM;}
        if (PWM[i] < Min_PWM) {if(hacanh == false){PWM[i] = Min_PWM;}}
    }
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_1,PWM[0]);
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_2,PWM[1]);
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_3,PWM[2]);
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_4,PWM[3]);
}
else{
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_1,1000);
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_2,1000);
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_3,1000);
    __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_4,1000);
}
}

```

a.

10.Kiểm tra xem đã đủ 1 vòng lặp 4ms chưa(chưa thì chờ)

`check_looptime();`

}

Nguyên văn vòng while xem trong code đính kèm

```

/* USER CODE BEGIN WHILE */
while (1)
{
/*-----READ DATA-----*/
    mbal_NRF24L01_GetData(&nrf_rx_cfg, &data_rx_real);
    MPU6050_Read_All(&hi2c2, &MPU6050_Data); //read data
    correct_data_and_calibrate_3truc();
    calculate_agl_roll_pitch();
    calculate_setpoint_pid();
    read_hc05_and_fillter();
}
/*-----Check right left front behind-----*/
/* USER CODE END WHILE */


/* USER CODE BEGIN 3 */
button_press = 0;
for(button_press = 0 ; button_press < 5 ; button_press++){
    if((data_rx_real >> button_press) & 1){
        break;
    }
    if(button_press == 4){
        if(!((data_rx_real >> button_press) & 1)){
            button_press = 10;
            break;
        }
    }
}

if(button_press == 3) { quad_stop(); }
else if(button_press == 2){ if (hacanh == false){quad_start(); }}
else if(button_press == 0){ quad_down(); }
else if(button_press == 1){ if (hacanh == false){quad_up(); }}
else {}

/*-----Operation Run Motor -----*/
PID_Calculation(&PID_ROLL, pid_roll_setpoint, MPU6050_Data.Gx_roll );
PID_Calculation(&PID_PITCH, pid_pitch_setpoint, MPU6050_Data.Gy_pitch );
PID_Calculation(&PID_YAW, pid_yaw_setpoint, MPU6050_Data.Gz_yaw );
PID_Calculation_thr(&PID_HG, Setting_docao, Kalman_Distance);

```

5 Setting phần cứng:

 main_quad.ioc

Mở cubeIDE và chọn mở file này :

6 Kalman và MPU6050

6.1 Soure :mpu6050.c

- Kalman_getAngle(Kalman_t*, double, double, double) : double
- MPU6050_Init(I2C_HandleTypeDef*) : uint8_t
- MPU6050_Read_Accel(I2C_HandleTypeDef*, MPU6050_t*) : void
- MPU6050_Read_All(I2C_HandleTypeDef*, MPU6050_t*) : void
- MPU6050_Read_Gyro(I2C_HandleTypeDef*, MPU6050_t*) : void
- MPU6050_Read_Temp(I2C_HandleTypeDef*, MPU6050_t*) : void

6.2 Header: mpu6050.h (giá trị của cảm biến sử dụng)

```
5 // MPU6050 structure
7 typedef struct
8 {
9
10     int16_t Accel_X_RAW;
11     int16_t Accel_Y_RAW;
12     int16_t Accel_Z_RAW;
13     int16_t Gyro_X_RAW;
14     int16_t Gyro_Y_RAW;
15     int16_t Gyro_Z_RAW;
16     float A_roll;
17     float A_pitch;
18     float A_yaw;
19     float G_roll;
20     float G_pitch;
21     float G_yaw;
22
23     float KalmanAngleX;
24     float KalmanAngleY;
25     float KalmanAngleZ;
26     float Gx_roll;
27     float Gy_pitch;
28     float Gz_yaw;
29     uint8_t MPU6050_address;
30     float Temperature;
31     float time_run;
32 } MPU6050_t;
33
34 // Kalman structure
35 typedef struct
36 {
37     double Q_angle;
38     double Q_bias;
39     double R_measure;
40     double angle;
41     double bias;
42     double P[2][2];
43 } Kalman_t;
```

6.3 Cánh dùng :

1:khai báo trước while để khởi tạo giá trị:xem chỗ mục 2 vòng main

```
364 HAL_GPIO_WritePin(GPIOC, GPIO_PIN_13,GPIO_PIN_RESET);
365 while(MPU6050_Data.MPU6050_address != 0x68){
366     MPU6050_Data.MPU6050_address =MPU6050_Init(&hi2c2);
367     HAL_Delay(50);
368 }
369 set_val_for_nrf24(&nrf_rx_cfg);
370 mbal_NRF24L01_Init(&nrf_rx_cfg );
371 HAL_Delay(100);
372 // Innit value for NRF24
373 MPU6050_Data.A_roll = 0;
```

```

371     HAL_Delay(100);
372     // Innit value for NRF24
373     MPU6050_Data.A_roll = 0;
374     MPU6050_Data.A_pitch = 0;
375     MPU6050_Data.A_yaw = 0;
376     MPU6050_Data.KalmanAngleX = 0;
377     MPU6050_Data.KalmanAngleY = 0;
378     MPU6050_Data.KalmanAngleZ = 0;
379     MPU6050_Data.Gx_roll = 0;
380     MPU6050_Data.Gy_pitch = 0;
381     MPU6050_Data.Gz_yaw = 0;
382     HAL_Delay(100);

```

2: đọc và lọc nhiễu:

```

400     while (1)
401     {
402     /*-----READ DATA-----*/
403     mba1_NRF24L01_GetData(&nrf_rx_cfg, &data_rx_real);
404     MPU6050_Read_All(&hi2c2, &MPU6050_Data); //read data
405     correct_data_and_calibrate_3truc();
406     calculate_agl_roll_pitch();
407     calculate_setpoint_pid();
408     read_hc05_and_filtter();

```

```

163
164 void MPU6050_Read_All(I2C_HandleTypeDef *I2Cx, MPU6050_t *DataStruct)
165 {
166     uint8_t Rec_Data[14];
167     // uint8_t set_gyro_angles = 0;
168     // Read 14 BYTES of data starting from ACCEL_XOUT_H register
169     HAL_I2C_Mem_Read(I2Cx, MPU6050_ADDR, ACCEL_XOUT_H_REG, 1, Rec_Data, 14, I2C_timeout);
170
171     DataStruct->Accel_X_RAW= (int16_t)(Rec_Data[0] << 8 | Rec_Data[1]);
172     DataStruct->Accel_Y_RAW= (int16_t)(Rec_Data[2] << 8 | Rec_Data[3]);
173     DataStruct->Accel_Z_RAW= (int16_t)(Rec_Data[4] << 8 | Rec_Data[5]);
174     DataStruct->Gyro_X_RAW = (int16_t)(Rec_Data[8] << 8 | Rec_Data[9]);
175     DataStruct->Gyro_Y_RAW = (int16_t)(Rec_Data[10] << 8 | Rec_Data[11]);
176     DataStruct->Gyro_Z_RAW = (int16_t)(Rec_Data[12] << 8 | Rec_Data[13]);
177
178 //
179 // DataStruct->Ax = (float)DataStruct->Accel_X_RAW / 1.0;
180 // DataStruct->Ay = (float)DataStruct->Accel_Y_RAW / 1.0;
181 // DataStruct->Az = (float)DataStruct->Accel_Z_RAW / 1.0;
182 // DataStruct->Gx = (float)DataStruct->Gyro_X_RAW / 1.0;
183 // DataStruct->Gy = (float)DataStruct->Gyro_Y_RAW / 1.0;
184 // DataStruct->Gz = (float)DataStruct->Gyro_Z_RAW / 1.0;
185 //
186 DataStruct->KalmanAngleX = Kalman_getAngle(&KalmanX, (float)DataStruct->Accel_X_RAW, (float)DataStruct->Accel_X_RAW/4200, 0.01);
187 DataStruct->KalmanAngleY = Kalman_getAngle(&KalmanY, (float)DataStruct->Accel_Y_RAW, (float)DataStruct->Accel_Y_RAW/4200, 0.01);
188 DataStruct->KalmanAngleZ = Kalman_getAngle(&KalmanZ, (float)DataStruct->Accel_Z_RAW, (float)DataStruct->Accel_Z_RAW/4200, 0.01);
189 }

```

Đọc và sử dụng Kalman để lọc nhiễu (đưa giá trị này thay cho giá trị gốc là có thể lọc được kalman).

7 Động cơ

7.1 Khai báo:

```

__HAL_TIM_SetCompare(&htim1, TIM_CHANNEL_1, 1000);
__HAL_TIM_SetCompare(&htim1, TIM_CHANNEL_2, 1000);
__HAL_TIM_SetCompare(&htim1, TIM_CHANNEL_3, 1000);
__HAL_TIM_SetCompare(&htim1, TIM_CHANNEL_4, 1000);
//set first value

```

7.2 Đưa giá trị:

```

450     PWM[0] = (uint16_t)(throttle_PID + PID_ROLL.pid_result + PID_PITCH.pid_result + PID_YAW.pid_result);
451     PWM[1] = (uint16_t)(throttle_PID + PID_ROLL.pid_result - PID_PITCH.pid_result - PID_YAW.pid_result);
452     PWM[2] = (uint16_t)(throttle_PID - PID_ROLL.pid_result + PID_PITCH.pid_result - PID_YAW.pid_result);
453     PWM[3] = (uint16_t)(throttle_PID - PID_ROLL.pid_result - PID_PITCH.pid_result + PID_YAW.pid_result);

```


7.3 Chạy động cơ

```
458     __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_1,PWM[0]);
459     __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_2,PWM[1]);
460     __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_3,PWM[2]);
461     __HAL_TIM_SetCompare(&htim1,TIM_CHANNEL_4,PWM[3]);
462 }
```