摘 要

本课题主要完成了基于STM32F4嵌入式开发平台开发的四轴飞行器的硬件系统设计，设计实现了四轴飞行器的硬件平台，实现了各功能部件的驱动编写与机械部件的搭建，为下学期的操作系统构建，和未来基于四轴飞行器的应用设计提供基础。

本系统采用 STM32F4 嵌入式开发平台作为控制核心，通过小组设计制作的 PCB 转接板集成各个外设功能模块，主要有蓝牙、GY-86、电机电调、遥控器、OLED、蜂鸣器等模块。其中通过蓝牙模块实现飞行器与上位机的信息交流，GY-86 传感器模块采集飞行器的姿态信息，电机与电调调整飞行器桨叶转速，遥控器操控飞行器电机转动，OLED 模块实时显示调试数据。在四轴机架上集成了各部件，通过各个功能模块的交互，达到在硬件系统上实现裸机软件集成的目标。

关键词：四轴飞行器，STM32 嵌入式开发平台，软件集成

目 录

第一章 复杂工程问题归纳与实施方案可行性研究

1.1 需求分析与建模

1.1.1 应用场景

本课题中的四轴飞行器的主要应用场景有：可二次开发的飞行平台

1.1.2 需求分析

① 单片机能够控制电机以不同转速转动
② 单片机能够捕获到遥控器接收机的操作信号
③ 单片机能够读取 GY-86 传感器数据
④ 蓝牙模块能够发送字符串数据到上位机
⑤ OLED 可以显示字符、数字等

1.2 复杂工程问题归纳

1.2.1 使用遥控器控制电机

使用遥控器拨动摇杆，单片机接收遥控数据，根据遥控数据计算对应电机转速，根据转速要求控制电机转速。

1.2.2 OLED 显示 GY-86 与遥控数据

获取 GY-86 姿态数据和遥控器数据，在 OLED 对应位置显示相应数据。

1.2.3 蓝牙发送 GY-86 姿态数据至上位机

获取 GY-86 姿态数据，将数据转换为数据帧形式，通过蓝牙发送数据帧至上位机，实现单片机与上位机的数据交互。

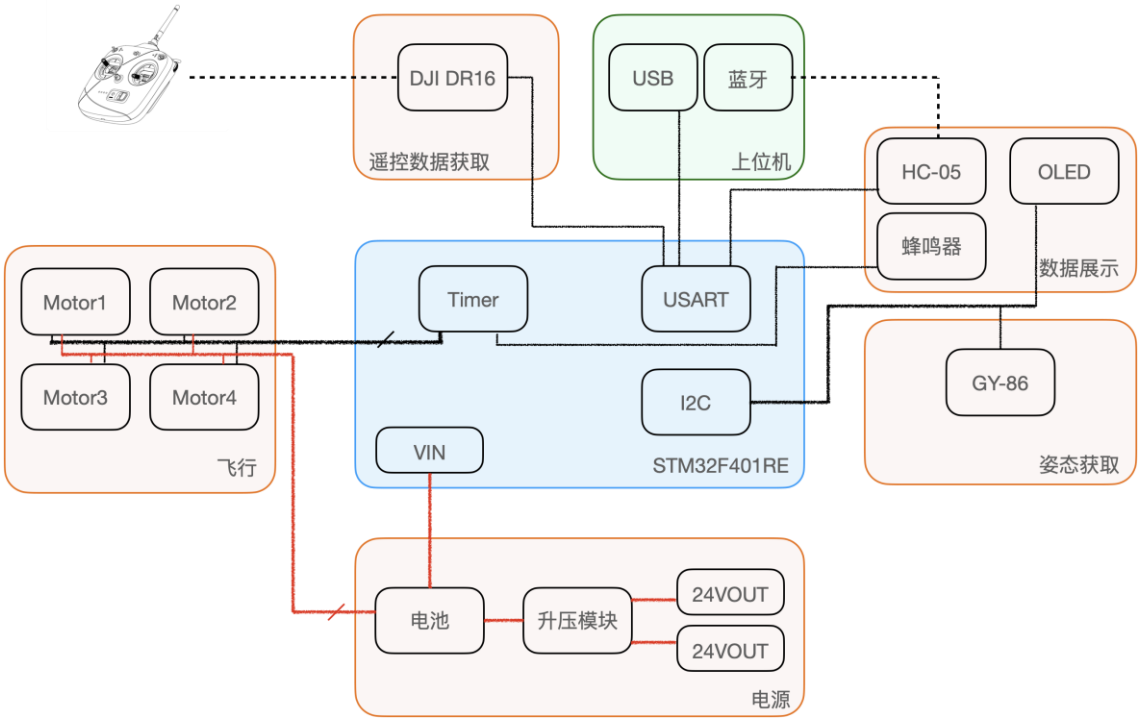
1.3 实施方案与可行性研究

1.3.1 总体架构

1.3.1.1 机械架构

选用标准的 F450 四轴机架，通过螺丝、铜柱等固定电池、电机、主控板等部件

1.3.1.2 硬件架构



图表 1-1 硬件模块逻辑连接图

1.3.2 模块功能分析

1.3.2.1 STM32F401RE 评估板

模块名称：STM32F401RE 评估板	通信方式：/
模块说明：该模块作为四轴飞行器的主控模块，负责采集各传感器数据，进行数据解析、运算，控制电机转动等工作，是四轴飞行器的“大脑”。	
优点：Cortex-M4, 512 KB Flash, 96 KB SRAM, 84MHz 主频, 具有丰富外设接口和较小体积，满足飞行器要求的数据处理能力。	
缺点：暂无	

表格 1-1 主控模块说明

1.3.2.2 转接板

模块名称：转接板	通信方式：/
模块说明：与评估板配套使用，将各模块与评估板相连接	
优点：便于各硬件模块与主控板的连接，提高连接便利性和稳定性。	
缺点：不易修改，新增或修改模块、接线时需重新设计和制板。	

表格 1-2 转接板模块说明

1.3.2.3 电机与电调

模块名称：朗宇电机、好盈电调	通信方式：PWM
模块说明：电机与电调配套使用，为四轴飞行器提供飞行动力。	
优点：该款电机使用 PWM 方式进行转速控制，控制方式简单。	
缺点：该款电机无转速反馈功能，无法进行转速闭环控制。	

表格 1-3 电机电调模块说明

1.3.2.4 姿态传感器

模块名称：GY-86	通信方式：I2C
模块说明：该模块集成了 MPU6050、HMC5883L 与 MS5611 三款芯片，为四轴飞行器提供 10DOF（线加速度、角加速度、角度与气压）姿态数据。	
优点：单一模块集成了三款芯片，满足四轴飞行器飞行所需的姿态数据要求。	
缺点：暂无	

表格 1-4 姿态传感器模块说明

1.3.2.5 遥控器与接收机

模块名称：DJI DT7 与 DR16	通信方式：DBUS
模块说明：该模块可远程发送与接收遥控器控制指令，为主控板提供操控指令。	
优点：提供 7 通道数据；操控距离远，可达 1km；具有摇杆回中功能，适合操控旋翼类无人机。	
缺点：暂无	

表格 1-5 遥控器与接收机模块说明

1.3.2.6 蓝牙

模块名称：HC-05 蓝牙模块	通信方式：USART
模块说明：使用该模块后，主控板可通过蓝牙向上位机发送调试数据并接收上位机的指令。	
优点：该模块使用广泛，开发简单，可进行无线通信。	
缺点：仅能进行一对一的通信。	

表格 1-6 蓝牙模块说明

1.3.2.7 OLED

模块名称：SSD1306-OLED	通信方式：I2C
模块说明：该模块为 128*64 像素显示屏，可直观展示飞行器姿态等相关数据。	
优点：模块小巧、性价比高、可直观展示调试数据。	
缺点：暂无	

表格 1-7 OLED 模块说明

1.3.2.8 RGB

模块名称：三色共阳 RGB 灯	通信方式：5V-GND
模块说明：该模块为三色 RGB 灯，通过调节 RGB 端的高低电平，控制红绿蓝灯珠的亮灭，从而发出不同的颜色	
优点：模块价格低廉，可发出多种颜色的光，作为无人机的航向灯和状态指示灯。	
缺点：占用引脚多，一个 RGB 灯需要占用四个引脚	

表格 1-8 RGB 模块说明

1.3.2.9 蜂鸣器

模块名称：TMB-12A05V-12095	通信方式：PWM
模块说明：该模块为有源蜂鸣器，通过 PWM 调整占空比控制发声音符，通过发声时间控制音符长度，从而播放所预期的一段音乐	
优点：可用于播放警报音提示模块离线或其他错误，更加直观。	
缺点：暂无	

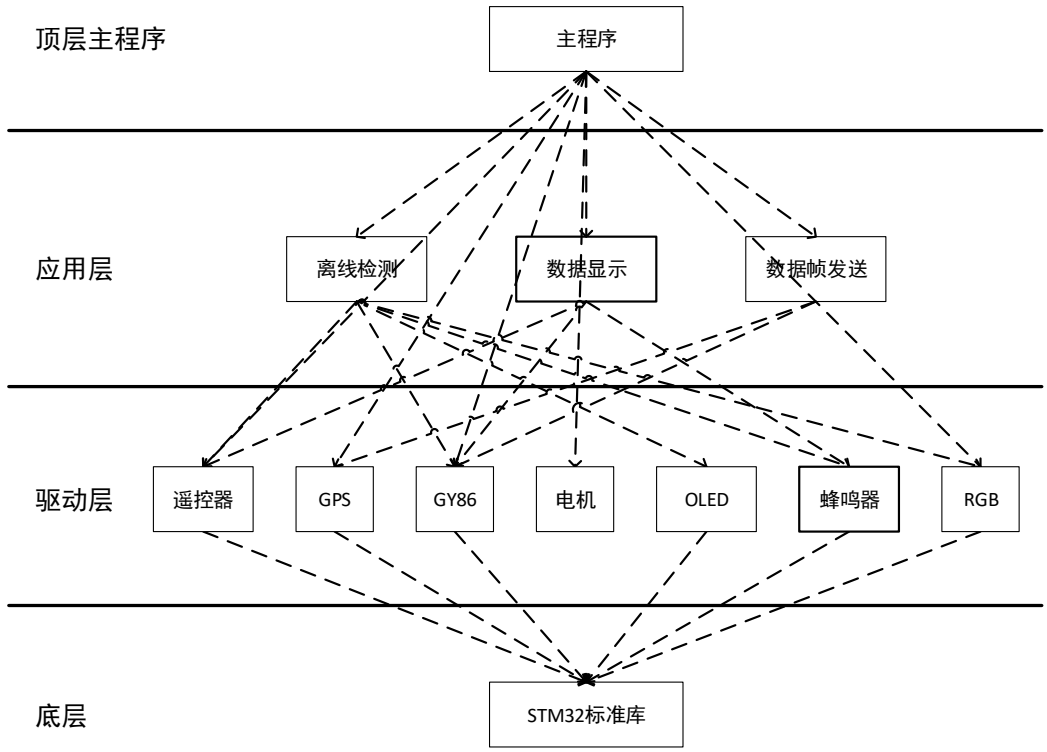
表格 1-9 蜂鸣器模块说明

1.3.2.10 GPS

模块名称：ATK-S1216-V1.1	通信方式：USART
模块说明：该模块为 GPS 定位模块，可获取四轴飞行器坐标	
优点：按照 NMEA 协议编写数据帧，易于解析	
缺点：暂无	

表格 1-10 GPS 模块说明

1.3.3 软件设计



图表 1-2 软件架构图

我们组将每个传感器模块编写成驱动程序，对外提供一系列接口，方便应用程序和主程序的调用，使主程序更为简洁直观。各传感器驱动都具有可移植性，可移植至其他 STM32F4 系列开发板使用。

1.3.3.1 电机模块驱动

我们组使用的电机由 PWM 脉冲控制，PWM 周期为 20ms，电调能识别的占空比范围为 5%-10%，即 1000us-2000us，我们组使用 STM32 计时器的 PWM 模式来输出相应的 PWM 脉冲。

本驱动完全根据 STM32F401 英文参考手册编写，未使用 STM32 的标准库。本驱动包含以下函数：

- bsp_Motor_Init()：初始化电机

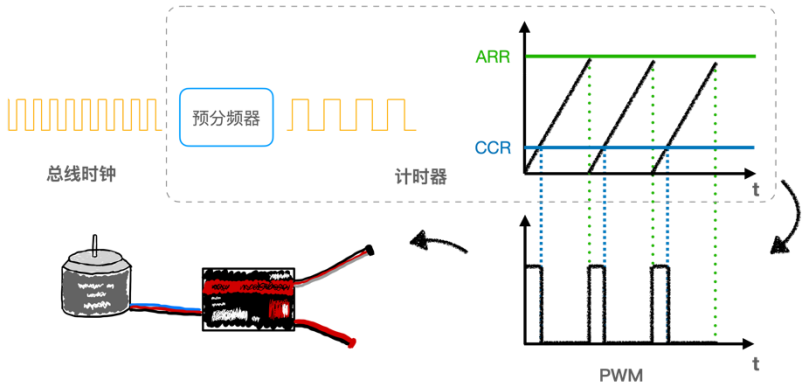
  用于打开计时器，开始输出 PWM 脉冲并解锁电机。

```
RCC->APB1ENR |= (1<<1); //将 bit1 置为 1，使能 TIM3 时钟
TIM3->PSC = 16-1; //设置 TIM3 的预分频器系数为 16-1
TIM3->ARR = 20000-1;  //设置 TIM3 的重装载值为 20000-1
TIM3->EGR |= 1; //产生更新事件

TIM3->CCMR1 |= 0x6060;//配置 TIM3_CH1,2 的输出比较模式为 PWM 模式 1,TIM3_CNT<TIM3_CCR，通道 1,2 便为有效状态
TIM3->CCMR2 |= 0x6060;//配置 TIM3_CH3,4 的输出比较模式为 PWM 模式 1,TIM3_CNT<TIM3_CCR，通道 3,4 便为有效状态

TIM3->CCR1 = 1000; //设置 TIM3_CCR1 的值
TIM3->CCR2 = 1000; //设置 TIM3_CCR2 的值
TIM3->CCR3 = 1000; //设置 TIM3_CCR3 的值
TIM3->CCR4 = 1000; //设置 TIM3_CCR4 的值

TIM3->CCER |= 0x1111;  //在相应的引脚下输出信号
TIM3->CCMR1 |= 1<<3;   //TIM3_CH1 预装载使能
TIM3->CCMR1 |= 1<<11;  //TIM3_CH2 预装载使能
TIM3->CCMR2 |= 1<<3;   //TIM3_CH3 预装载使能
TIM3->CCMR2 |= 1<<11;  //TIM3_CH4 预装载使能
//预装载：修改 CCR 时保留在“预加载”寄存器，在 CNT 溢出时才更新 CCR

TIM3->CR1 |= 1; //TIM3 定时器使能（TIM3_CNT 开始计数）

RCC->AHB1ENR |= 1<<0;  //将 bit1 置为 1，使能 GPIOA 时钟
RCC->AHB1ENR |= 1<<1;  //将 bit1 置为 1，使能 GPIOB 时钟
```

```
GPIOA->MODER |= 0X00002000;      //将 bit[13:12]置为 10，PA6 设置为复用功能模式
GPIOA->MODER |= 0X00008000;      //将 bit[15:14]置为 10，PA7 设置为复用功能模式
GPIOB->MODER |= 0X00000002;      //将 bit[1:0]置为 10，PB0 设置为复用功能模式
GPIOB->MODER |= 0X00000008;      //将 bit[3:2]置为 10，PB1 设置为复用功能模式

GPIOB->AFR[0] |= 2<<4; //将 bit[7:4]置为 0010，设置 PB1 复用功能为功能 2
GPIOB->AFR[0] |= 2;      //将 bit[3:0]置为 0010，设置 PB0 复用功能为功能 2
GPIOA->AFR[0] |= 2<<24;      //将 bit[27:24]置为 0010，设置 PA6 复用功能为功能 2
GPIOA->AFR[0] |= 2<<28;      //将 bit[31:28]置为 0010，设置 PA7 复用功能为功能 2
```

表格 1-11 电机初始化相关代码

本函数中，先使能 APB1 总线至 TIM3 的时钟信号，调节预分频器参数，使得计时器中计数器每 1us 计数一次。配置 ARR 预装载值为 20000-1，使得计数器大于 19999 归 0，即周期为 20000us=20ms，频率为 50Hz，与电机能识别的 PWM 频率对应。配置计时器相关计时器，使得四个通道均为 PWM 比较模式 1，开启 PWM 脉冲输出，设置 4 路通道的 CCR 的初始值为 1000，即电机解锁信号。当计数器值 CNT<CCRx 时，对应 OCxREF 寄存器的值为'1'，否则为'0'。随后，查阅数据手册的引脚功能对照表，配置 GPIO 相应引脚为引脚复用功能，复用为计时器模式，当 OCxREF='1'时，对应引脚输出高电平，否则输出低电平，实现 4 路 PWM 脉冲的输出。



图表 1-3 PWM 脉冲输出控制电机示意图

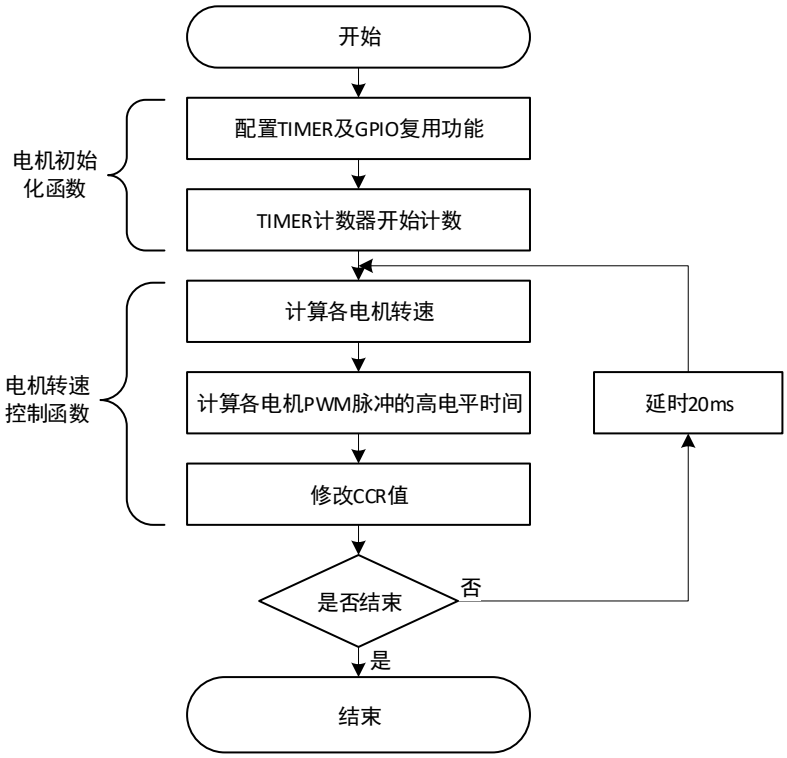
- bsp_motor_set_speed(int* speed)：设置电机转速

传入四个电机的角速度值，换算成对应的占空比，调节计时器对应通道的 CCR 值，PWM 脉冲的占空比随之改变，继而调节电机转速。

```
void bsp_motor_set_speed(int *speed) {
    MOTOR_TIMER.Instance->CCR1 = MIN_DUTY + (uint16_t)((MAX_DUTY - MIN_DUTY)
```

```
* (speed[0] * 1.0f / MOTOR_MAX_SPEED));
    MOTOR_TIMER.Instance->CCR2 = MIN_DUTY + (uint16_t)((MAX_DUTY - MIN_DUTY)
* (speed[1] * 1.0f / MOTOR_MAX_SPEED));
    MOTOR_TIMER.Instance->CCR3 = MIN_DUTY + (uint16_t)((MAX_DUTY - MIN_DUTY)
* (speed[2] * 1.0f / MOTOR_MAX_SPEED));
    MOTOR_TIMER.Instance->CCR4 = MIN_DUTY + (uint16_t)((MAX_DUTY - MIN_DUTY)
* (speed[3] * 1.0f / MOTOR_MAX_SPEED));
}
```

表格 1-12 电机转速控制函数代码



图表 1-4 裸机状态驱动电机程序流程图

1.3.3.2 GY-86 模块驱动

GY-86 模块集成了 MPU6050、HMC5883L、MS5611 芯片，使用 I2C 进行通信。本驱动主要包含以下函数：

- bsp_gy86_Init(init_device)：初始化 GY86

  调用传入编号所代表的设备的初始化函数

- bsp_gy86_IMU_Init(accelrange, gyrorange)：初始化 IMU

程序初始化 I2C 和 GPIO 复用功能配置，通过 I2C 配置 MPU6050 的测量精度、采样频率等信息，释放 MPU6050 对 I2C 从设备的控制（因 HMC5883L 挂载

在 MPU6050 的 I2C 总线上，需令 MPU6050 释放对 I2C 从设备的控制，主控板即可与 HMC5883L 进行通信）

- bsp_gy86_AXIS_Init()：初始化地磁仪

  类同 bsp_gy86_IMU_Init，初始化配置设备的测量精度、测量方式等相关信息

- bsp_gy86_APG_Init()：初始化气压计

  类同 bsp_gy86_IMU_Init，初始化配置设备的测量精度、测量方式等相关信息

- bsp_gy86_Refresh_Data(refresh_device)：刷新 GY-86 数据

  调用传入编号所代表的设备的刷新函数

- bsp_gy86_IMU_Refresh_Data()：刷新 IMU 数据

向 MPU6050 发送需被读取的数据寄存器号指令，MPU6050 芯片内的寄存器指针会指向对应数据寄存器，通过 I2C 连续读取 14 字节数据完成原始数据采集，根据参考手册上的计算公式对原始数据进行解析，将解析结果存入 imu_data 全局结构体变量中，即完成数据解析。

- bsp_gy86_AXIS_Refresh_Data()：刷新地磁仪数据

因 HMC5883L 在采集数据后，芯片内的寄存器指针会重新指向需被读取的数据寄存器，所以不再需要发送指令修改采集芯片内寄存器指针，可直接进行数据采集与解析，其余与刷新 IMU 类同，最后将解析结果存入 axis_t 全局结构体变量中。即完成数据解析。

- bsp_gy86_APG_Refresh_Data()：刷新气压计数据

MS5611 的数据采集与解析步骤与 MPU6050 相同，最后将解析结果存入 apg_t 全局结构体变量中,即完成数据解析。不做赘述。

本驱动所使用的数据结构：

```
typedef struct {
    int16_t x, y, z;   //地磁数据
} axis_t;
```

表格 1-13 地磁仪数据结构体

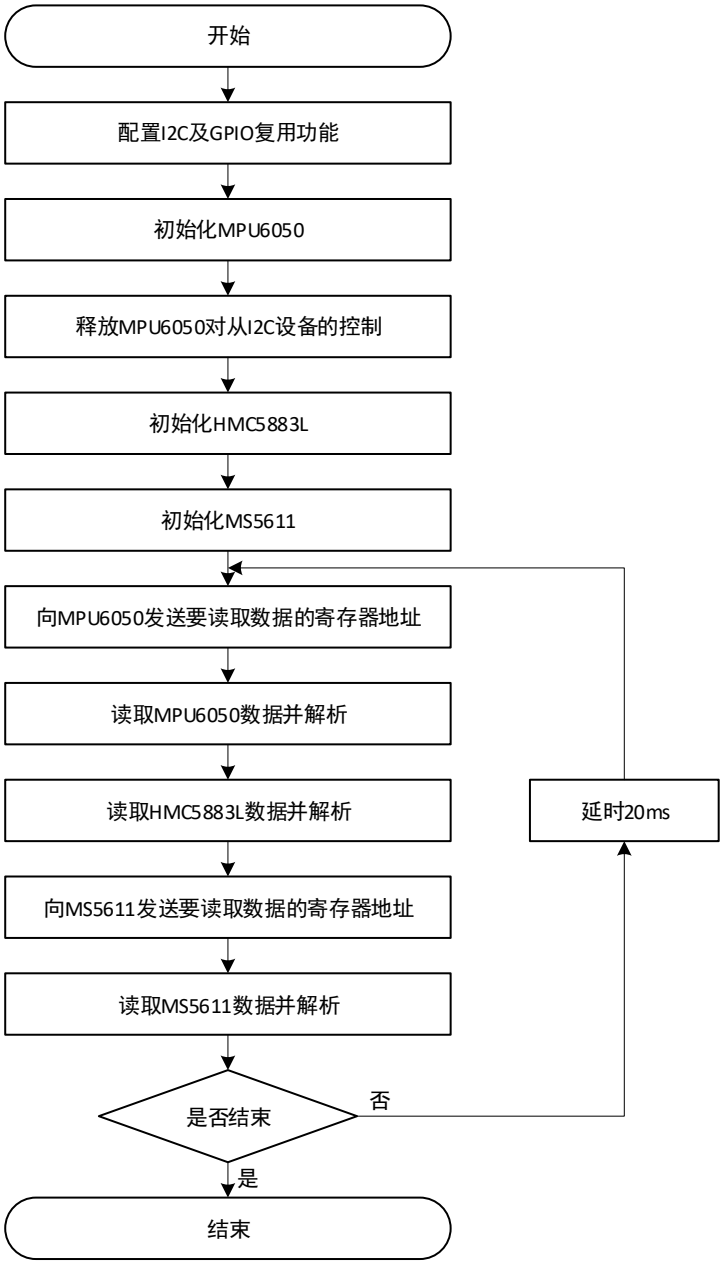
```
typedef struct {
    float x, y, z;   //线加速度
    float roll, pitch, yaw;   //角加速度
    float temp;   //温度
} imu_t;
```

表格 1-14 IMU 数据结构体

```
typedef struct {
```

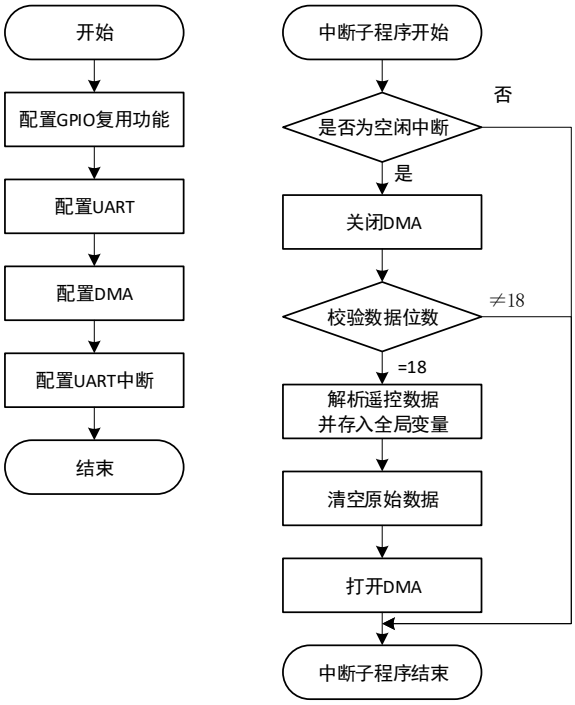
```
    uint32_t pressure;   //大气压，单位：0.01mbar，eg:100009=1000.09mbar
    uint32_t temp;       //温度，单位:0.01C，eg:2007=20.07C
} apg_t;
```

表格 1-15 气压计数据结构体



图表 1-5 裸机状态获取 GY86 数据程序流程图

1.3.3.3 遥控模块驱动



图表 1-6 遥控模块驱动程序流程图

我们的接收机输出信号为标准的 DBUS 协议数据，当遥控器与接收机建立连接后，接收机每隔 14ms 通过 DBUS 发送一帧 18 字节数据，经三极管取反电路后转换为 UART 协议数据。

本驱动包含以下主要函数：

- bsp_dbus_It(void)：Dbus 中断处理函数

当接收机输出数据时，触发 UART 中断，程序检查该中断是否为 DMA 空闲中断（DMA 空闲了一段时间为完成一帧遥控数据的接收），关闭 DMA 接收，防止下一帧遥控数据输入影响上一帧的数据解析，调用数据解析函数，数据解析完成后打开 DMA，准备接收处理下一帧遥控数据。

- bsp_dbus_Init(void)：Dbus 总线初始化

程序初始化 UART、GPIO 复用功能等配置，配置 DMA 由 USART1_RX 直接传输数据到内存中，开启 USART 与 DMA 中断。

- rc_datacheck(void)：Dbus 总线数据校验

检查目前接收到的数据是否为 18 位以及接收的各位数据的有效性和正确性。

- bsp_dbus_Analysis(void)：Dbus 数据解析

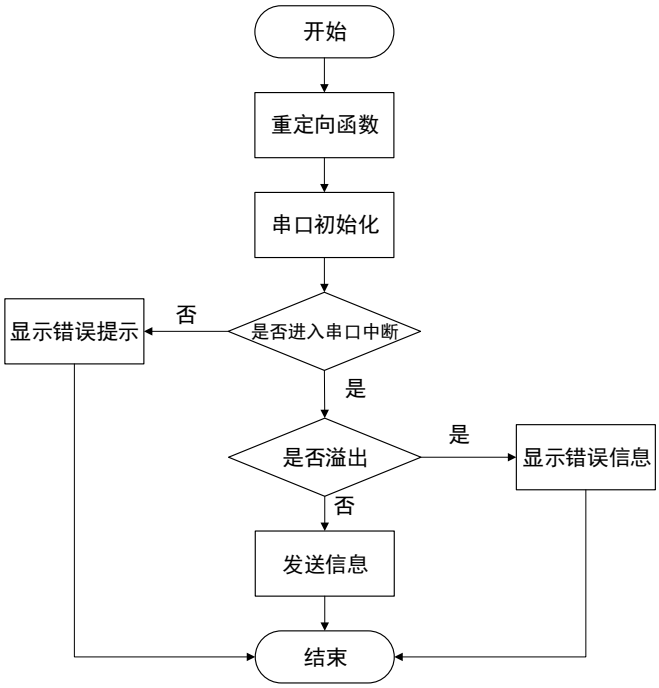
按照参考手册进行数据解析，将通道量等数据写入到全局结构体变量中，清空原始数据。

本驱动所使用的数据结构：

```
typedef struct rc_rec {
    int16_t ch0;          //通道 0
    int16_t ch1;          //通道 1
    int16_t ch2;          //通道 2
    int16_t ch3;          //通道 3
    uint8_t S1;           //左拨码开关
    uint8_t S2;           //右拨码开关
    int16_t Dial;         //拨码盘
    struct {
        int16_t X;          //X 轴
        int16_t Y;          //Y 轴
        int16_t Z;          //Z 轴
        uint8_t Leftkey;    //左键
        uint8_t Rightkey;   //右键
    } Mouse;              //鼠标信息   遥控支持鼠标操控,
} bsp_dbus_RC_Data;
```

表格 1-16 遥控器数据结构体

1.3.3.4 蓝牙模块驱动



图表 1-7 蓝牙模块驱动程序流程图

首先，进行 printf 函数的重定向，进行 UART 和 GPIO 的配置。然后判断是否进入串口中断，如否，显示错误提示码；如是则判断是否溢出。溢出则返还错误信息，未溢出就通过蓝牙发送四轴飞行器的姿态信息和 GPS 等信息转化成的信息帧。

1.3.3.5 OLED 模块驱动

OLED 使用 I2C 总线进行控制。控制其显示字符的主要原理是设置待修改像素的区域坐标，发送一系列的像素点亮灭二进制数据，OLED 寄存器会在指定区域中按照一定顺序修改寄存器的值，显示出的内容随之改变。

在店家附赠的例程和中期所写的驱动中，都是采用了将要显示的数字或字符串转换成一个个字符，随后调用字符显示函数，指定一个字符的区域坐标，发送一个字符的像素数据，再指定下一个字符的区域坐标，发送下一字符的像素数据。这种方法实现起来相对简单，但是字符串或数字已经是在一段连续的区域中显示，频繁的发送区域坐标会导致程序效率低下和 I2C 总线占用时间增多，为了提高程序效率，我们决定以空间换取时间，一次性处理出整个字符串或数字的像素数据，根据字符串长度和字体大小计算待显示的区域，随后将整块像素数据一起发送至 OLED 中。经实际测试，显示相同的数字，时间缩短了一半。在下学期学习了操作系统的相关知识后，可以将整块的像素数据发送修改为 DMA 传送的形式，CPU 的资源能被更高的利用。

本驱动实现了以下函数：

- bsp_oled_Init()：初始化 OLED
- bsp_oled_Display_On()：打开显示器
- bsp_oled_Display_Off()：关闭显示器
- bsp_oled_Clear()：显示器清屏
- bsp_oled_Clear_Part(x1, x2, y1, y2)：显示器部分清屏
- bsp_oled_Show_Char(x, y, chr, fontsize)：显示字符
- bsp_oled_Show_String(x, y, *chr, fontsize)：显示字符串
- bsp_oled_Show_Num(x, y, number, fontsize)：显示有符号整数
- bsp_oled_Show_Float(x, y, number, precision, fontsize)：显示小数

显示小数功能支持自定义显示精度。

程序先分离出整数部分和小数部分，将小数部分乘以 $10^{精度}$，四舍五入取整，判断四舍五入的过程是否会影响整数并对整数进行修改（例如 8.995 保留两位小数时，结果是 9.00，结果的整数部分相对于原数的整数部分包含了一个进位的 1）。

显示小数部分时，先判断小数的高位是否需要补 0，再将小数部分填充至像素数据中（例如 9.052 保留两位小数时，经过四舍五入处理后，整数部分为 9，小数部分转成整数后为 5，此时长度不足精度的两位，小数的高位需要补充 1 个 0，才能正确显示出 9.05）。

- bsp_oled_Draw_Bmp(x1, x2, y1, y2, bmp)：显示图片

1.3.3.6 RGB 模块驱动

本驱动用枚举变量列举了 RGB 灯的 8 种颜色选项，第 2 位至第 0 位分别控制红、绿、蓝三种颜色输出，对应位置 0 颜色输出有效。例如，PINK=$\bar{R}G\bar{B}$，RGB 灯亮起红色和蓝色，混合形成粉色的光。

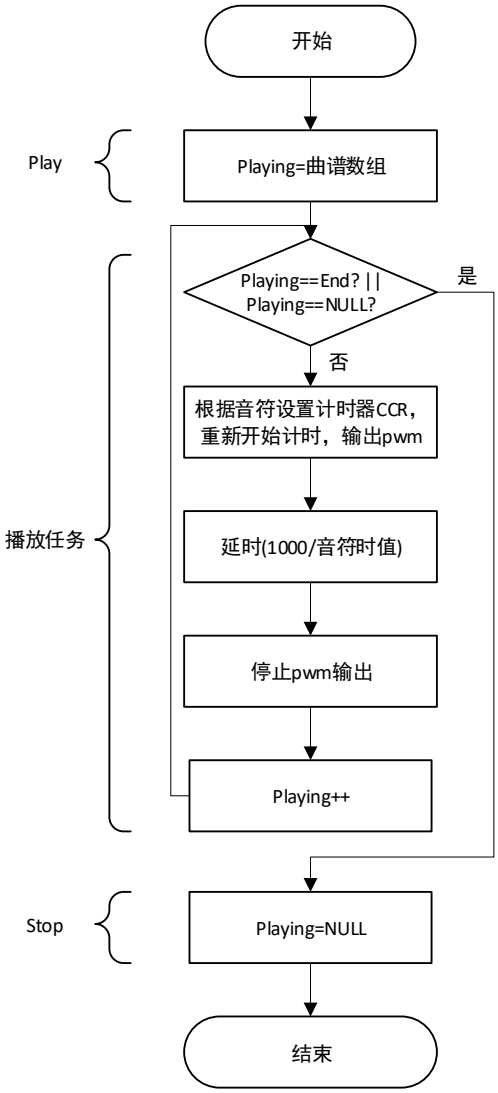
```
enum RGB_COLOR {
    WHITE = 0,
    YELLOW = 1,
    PINK = 2,
    RED = 3,
    CYAN = 4,
    GREEN = 5,
    BLUE = 6,
    OFF = 7
};
```

表格 1-17 颜色枚举变量

本驱动包含以下主要函数：

- bsp_rgb_On(pos, color)：控制指定位置的 RGB 灯显示指定颜色。
- bsp_rgb_Off(pos)：关闭指定位置的 RGB 灯。
- bsp_rgb_Toggle(pos)：切换指定位置 RGB 灯的亮灭状态（若切换为开默认亮起上一次指定颜色）。
- bsp_rgb_Toggle_Frequency(pos)：以指定频率切换 RGB 灯亮灭状态（若切换为开默认亮起上一次指定颜色）。
- bsp_rgb_set_frequency(f)：设置切换频率（搭配 Toggle_Frequency 函数使用以修改切换频率）。
- bsp_rgb_set_color(pos, color)：设置指定位置的 RGB 灯颜色（搭配 Toggle 函数使用以修改下一次亮起颜色）。

1.3.3.7 蜂鸣器模块驱动



图表 1-8 通过蜂鸣器播放音乐流程图

在驱动中预置了多种警报声、确认声、歌曲等常用音乐曲谱，仅可播放预设的音乐。

本驱动主要包含以下三个函数：

- bsp_beep_Music_Play(musicHead)：开始播放函数

开始播放指定的音乐

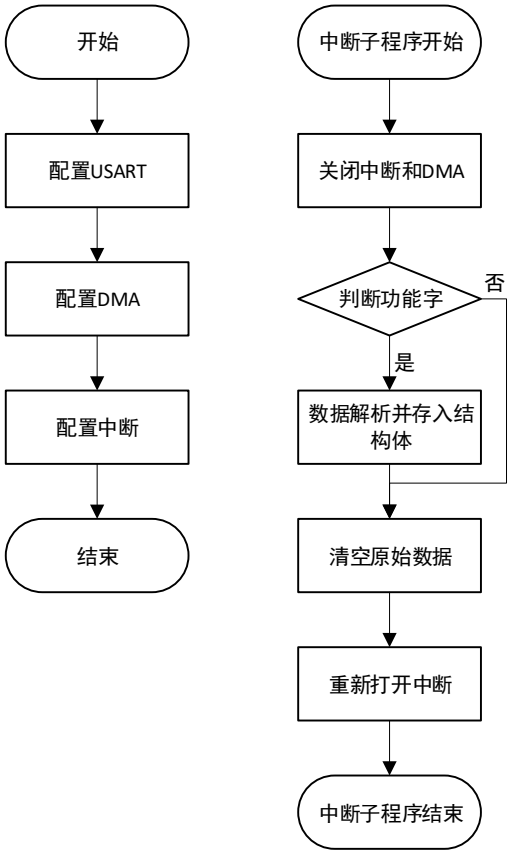
- bsp_beep_Music_Stop()：停止播放函数

结束播放函数可以立刻停止播放。

- bsp_beep_Music_Task()：播放控制任务函数

播放控制任务函数主要用于控制每个音符的播放，需要以 2Hz 的频率调用此函数，本学期在主程序中以轮询的方式调用，下学期学习操作系统后可以以中断或任务的方式进行调用，节约 CPU 的资源占用。通过音符与占空比对应表，根据当前需播放的音符调节 PWM 的占空比，开始 PWM 输出，蜂鸣器即可播放出对应的音符，根据当前音符的拍数（即播放长度），等待对应的时间，即为音符播放的时间，延时结束后停止 PWM 输出，蜂鸣器不再发声，完成一个音的播放。以此循环，即可完成一段音乐的播放。

1.3.3.8 GPS 模块驱动



图表 1-9 GPS 驱动程序流程图

首先，初始化 USART 等配置，通过 DMA 直接将接收到的 GPS 信息放入内存，开启 USART 中断。每当收到 GPS 数据时触发中断，关闭中断和 DMA 接收（防止影响上一帧数据解析）并通过功能字对收到的 GPS 数据帧进行判断，若是所需的 GPS 信息帧则将其解析并将其写入对应结构体。清空原始数据，重新开启中断，准备接收下一帧信息。

- bsp_gps_Init()：GPS 初始化函数

  清空中断标志位，打开中断，通过 DMA 将收到的数据放入内存。

- bsp_gps_It()：GPS 中断初始化函数

  关闭中断和 DMA（防止下一帧数据影响上一帧的解析），对收到的数据帧进行解析，将解析结果存入结构体，清空原始数据，并重新打开中断，准备下一帧数据的处理。

- GPS_Analysis(gps_data_t *gps, char *buf)：GPS 数据解析函数

  对所需的数据帧进行分析，如 GNGGA。

- GPS_GNGGA_Analysis(gps_data_t *gps, char *buf)：GNGGA 数据解析函数

  将收到信息与功能字进行比对，若相同，则将数据帧中对应信息写入结构体中。同理，若需要其他 GPS 信息，也可按照此种处理方法进行处理。

- GPS_Comma_Pos(char *buf, uint8_t cx)：数据帧内信息定位函数

  查看 GPS 数据帧的编写协议可知，各不同信息之间都用逗号隔开，查阅协议可知各数据所在位置（在第几个逗号之后），我们就可以根据逗号的位置对数据进行定位。

1.3.3.9 蓝牙数据帧发送应用

在 STM32 通过 HC05 与上位机进行通信时，为使上位机程序收到数据信息后进行二次处理，我们选择将要传送数据写成数据帧的形式进行传送，本程序中需要传送 GY86、GPS 的数据。当遇到过长的数据时，可分为多个数据帧进行传送。数据帧总长度 15 位，由帧首、功能字、数据、帧尾构成。

帧首	功能字	数据	帧尾
0XFF	1 位	12 位	0X0D

表格 1-18 数据帧结构表

1、IMU 加速度计 h，功能字 0x01

帧首	功能字	ax	ay	az	帧尾
0XFF	0x01	float	float	float	0X0D

表格 1-19 IMU 数据帧(h)结构表

2、IMU 加速度计 m，功能字 0x02

帧首	功能字	temp	pitch	yaw	帧尾
0XFF	0x02	float	float	float	0X0D

表格 1-20 IMU 数据帧(m)结构表

3、IMU 加速度计 l，功能字 0x03

帧首	功能字	roll	pitch	补零	帧尾
0XFF	0x03	float	float	4 位	0X0D

表格 1-21 IMU 数据帧(l)结构表

4、APG 气压计，功能字 0x04

帧首	功能字	temp	pressure	补零	帧尾
0XFF	0x04	float	float	4 位	0X0D

表格 1-22 APG 数据帧结构表

5、AXIS 地磁仪，功能字 0x05

帧首	功能字	X	Y	Z	帧尾
0XFF	0x05	float	float	float	0X0D

表格 1-23 AXIS 数据帧结构表

6、GPS 定位数据 h，功能字 0x06

帧首	功能字	latitude	nshemi	补零	帧尾
0XFF	0x06	double	char	3 位	0X0D

表格 1-24 GPS 数据帧(h)结构表

7、GPS 定位数据 l，功能字 0x07

帧首	功能字	longitude	ewhemi	补零	帧尾
0XFF	0x07	double	char	3 位	0X0D

表格 1-25 GPS 数据帧(l)结构表

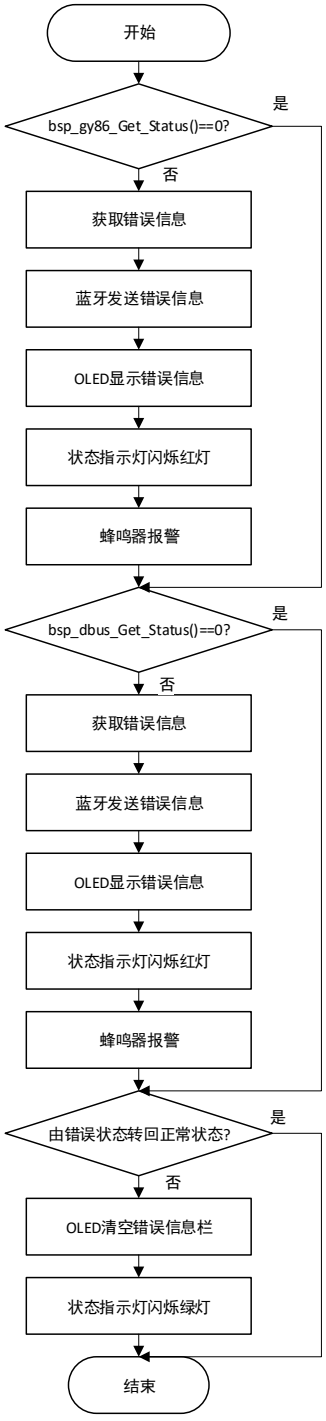
- app_hc05_SendData(uint8_t _Functionword)：发送指定数据帧

数据帧发送函数负责将指定的数据帧通过蓝牙发送出去。通过输入的功能字判断要发送哪些数据帧，将对应数据从源数据结构体中找到并写入数据发送结构体中进行发送。

- app_hc05_SendAllData()：发送所有数据帧

1.3.3.10 离线检测应用

为了提高四轴飞行器程序的健壮性，我们组为四轴飞行器提供了模块在线状态检测功能，根据模块重要性程度设置优先级，按优先级从高到低的顺序对模块的在线状态进行轮询，若模块离线，状态指示灯闪烁红灯、蜂鸣器发出警报声。



图表 1-10 离线检测程序流程图

图表 1-11 OLED 显示遥控器离线信息

1.3.3.11 OLED 显示数据应用

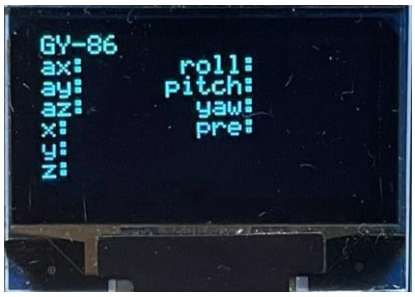
本应用用于在 OLED 显示屏上的前 7 行显示当前的 GY86、遥控器数据，最后一行保留用于离线检测应用显示错误信息。本应用包含一个 oled_function_id，用于区分显示 GY86 数据、显示遥控器数据功能。

本应用包含以下函数：

- app_oled_Show_Template()：显示模版

  根据待显示的数据功能，显示标题，数据名称等信息。

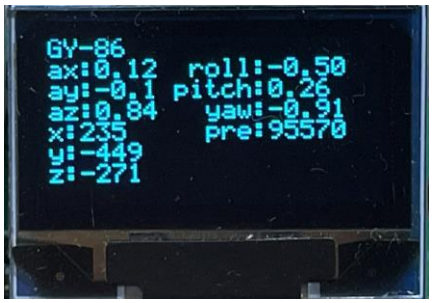
  此函数由主程序初始化和切换显示功能时调用。



图表 1-12 GY-86 数据模版

- app_oled_Show_Data()：显示数据

  此函数负责在上述模版中的对应位置填充目前的传感器数据。

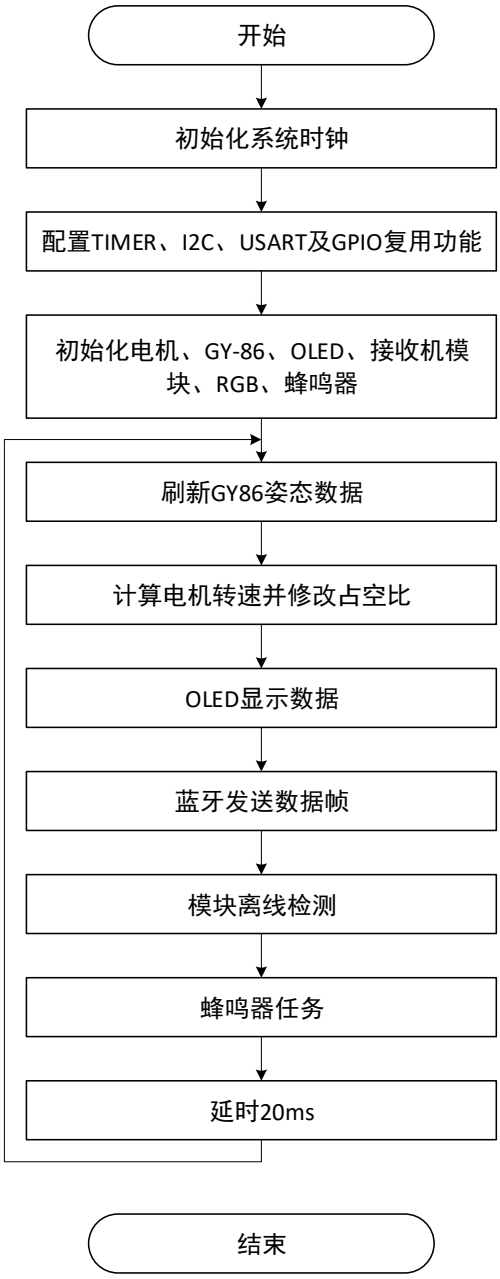


图表 1-13 OLED 实时显示 GY-86 数据

- app_oled_Switch_Function()：切换显示功能

在检测到按钮按压的中断后调用此函数，可以切换显示功能，实现按压按钮在GY-86和遥控数据显示间切换的功能。

1.3.3.12 主程序

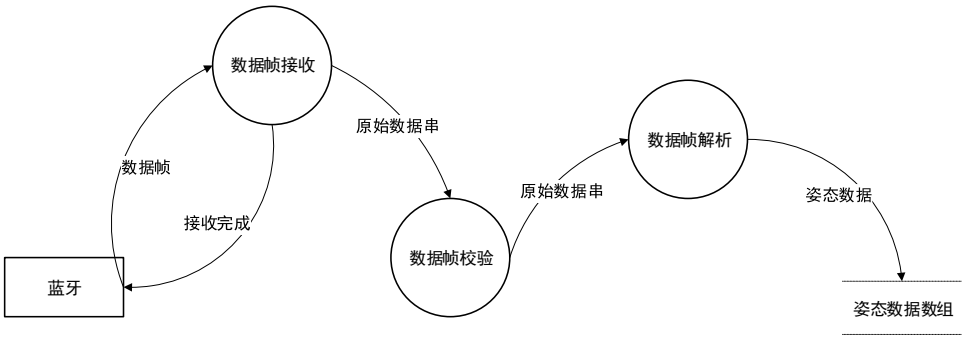


图表 1-14 主程序流程图

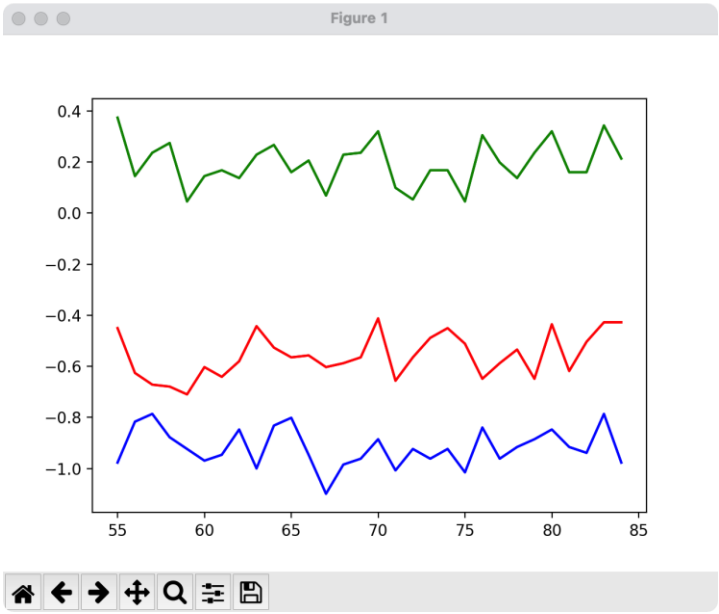
主程序先初始化系统时钟，初始化计时器、I2C、UART等功能，初始化各模块，然后以大约50Hz的频率轮询调用应用或模块驱动，实现整机各模块协同运行。

1.3.3.13 上位机软件

我们组使用 python+matlabplot 编写了上位机软件，通过笔记本电脑上的蓝牙连接四轴飞行器的蓝牙模块，实现了接收数据帧，解析数据并动态绘制折线图的功能。



图表 1-15 上位机软件处理数据流图



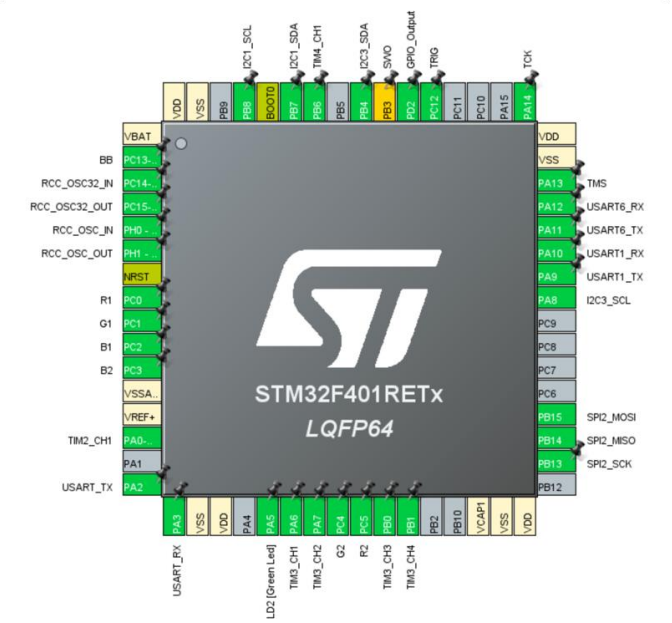
图表 1-16 上位机使用 matlabplot 绘制的动态折线图

1.3.4 转接板绘制

为方便各模块的连接，我们使用 Altium Designer 设计转接板，并交由专业公司进行制板。

1.3.4.1 引脚规划

根据四轴飞行器的需求，我们需要使用 GY-86(I2C), HC-05(UART), OLED(I2C), DR16(UART), 电机(PWM)等模块，利用 STM32CubeMX 软件进行引脚规划，确定各引脚需要复用的功能。

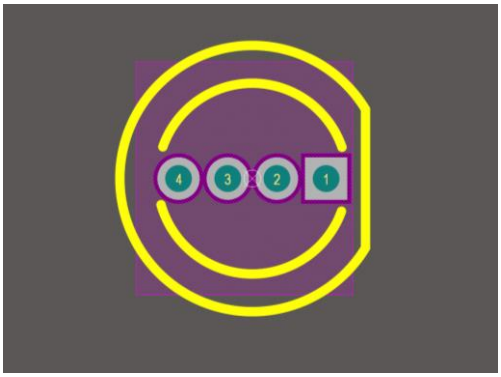


图表 1-17 CubeMX 引脚规划图

1.3.4.2 元件库绘制

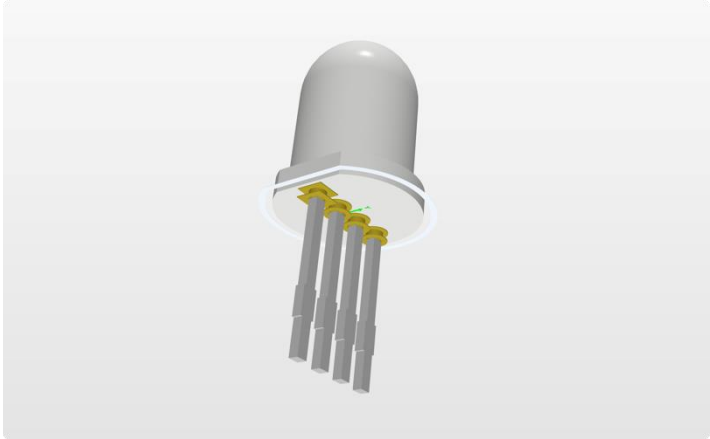
为提高原理图和 PCB 图的准确性，我们小组根据所需模块，绘制了元件库，根据模块或元器件的类型

- 对于立创商城购买的元器件，首先从立创商城下载模块的原理图和 PCB 图,转换为原理图库和 PCB 图库，并对元器件进行封装。



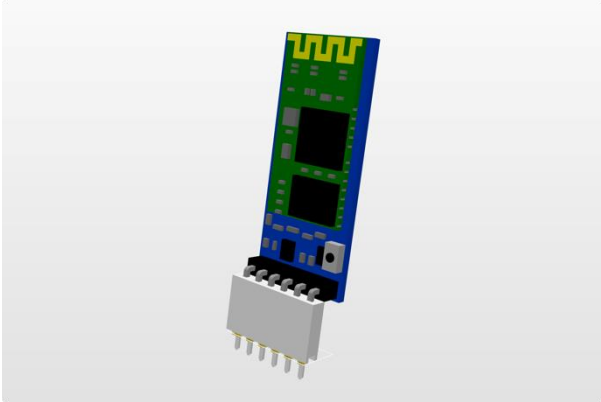
图表 1-18 RGB LED 的 PCB 库

从 3dcontentcentral 与 traceparts 网站下载元器件的开源 3D 模型，为 PCB 添加 3D 元件体，即完成一个元器件的绘制。



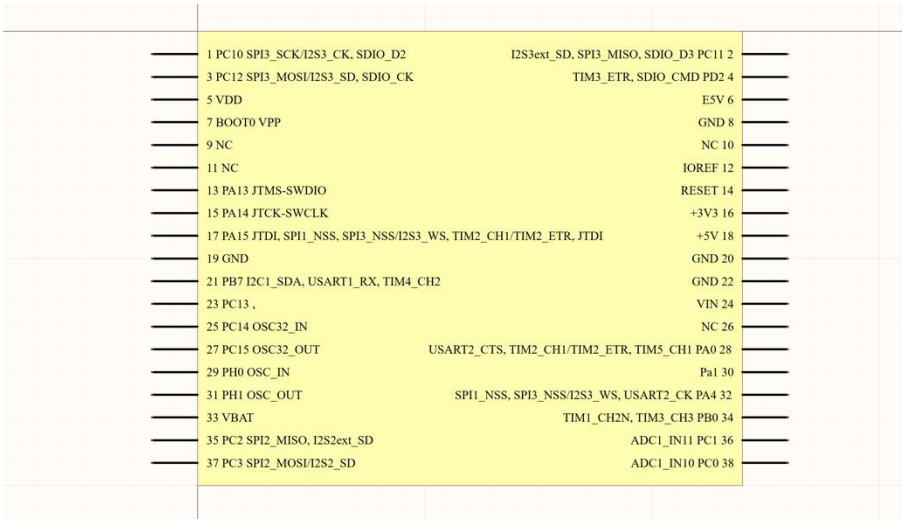
图表 1-19 RGB LED 的 PCB 3D 视图

- 对于 GY-86、HC-05 等模块，从官方库导入对应的排针组件，测量模块的长宽数据，在丝印层绘制模块边框，从 3dcontentcentral 网站下载元器件的开源 3D 模型，添加 3D 元件体，完成一个模块的绘制。



图表 1-20 HC-05 模块的 PCB 3D 视图

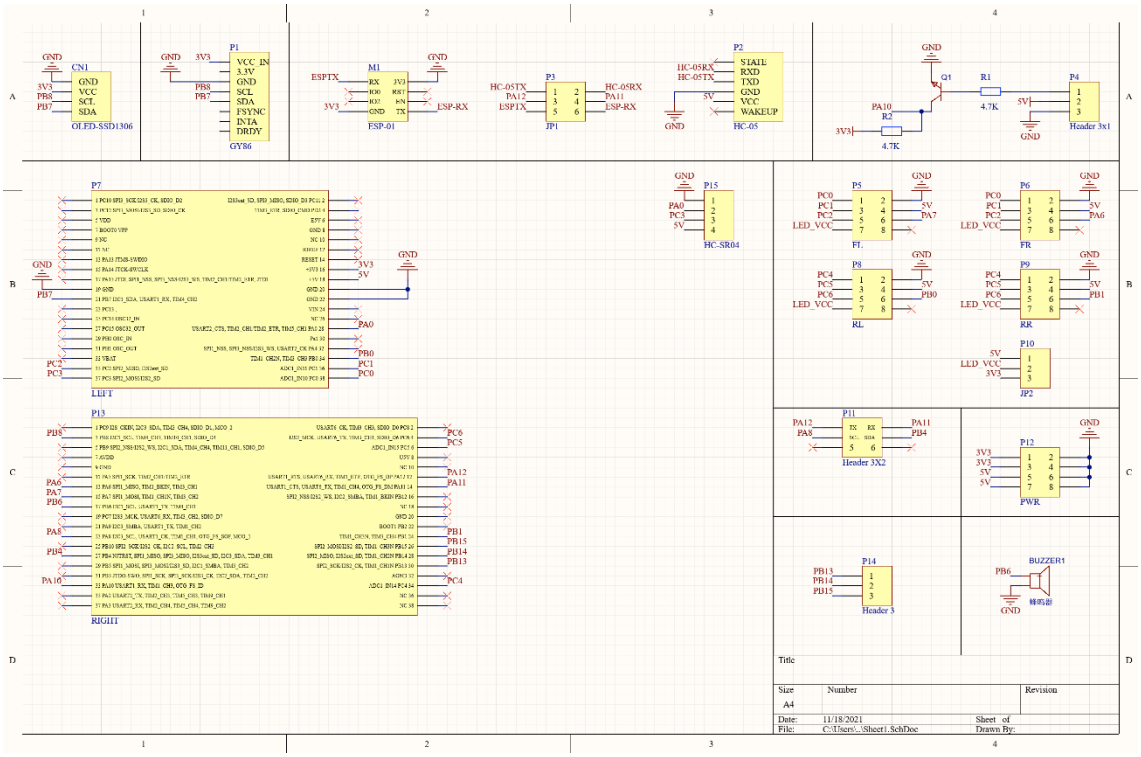
- 对于与 STM32F401RE 的左右两排排针，我们小组将针脚名称修改为“编号 GPIO 编号 复用功能”的形式，方便后续原理图的逻辑连线与引脚功能检查。



图表 1-21 STM32F401RE 评估板左排针的原理图库

1.3.4.3 原理图绘制

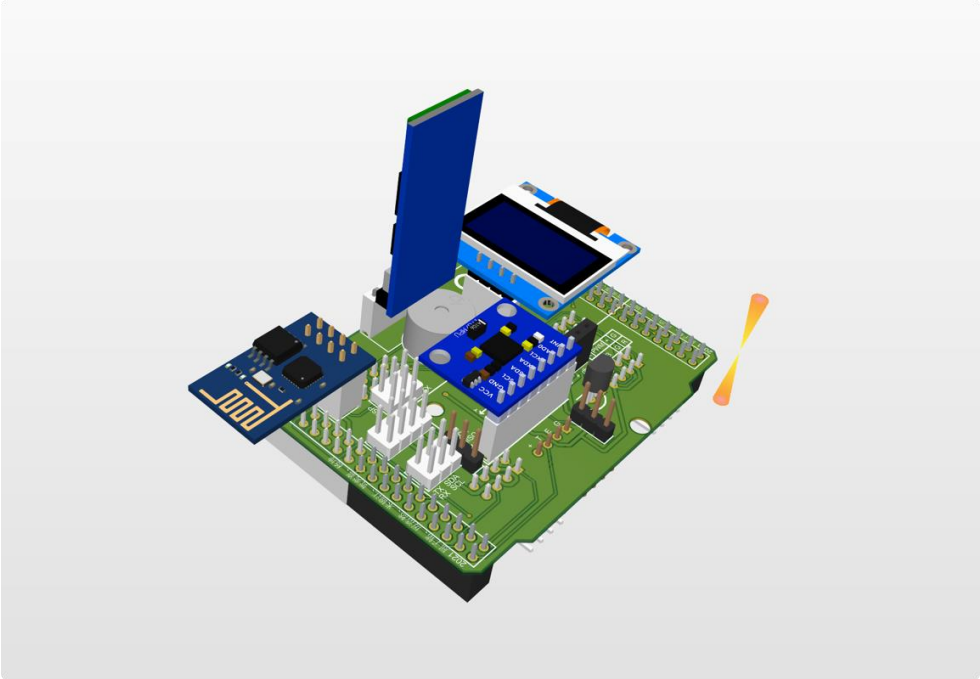
根据所需元器件类型与数量，将小组库中的元器件导入到原理图中。根据引脚规划图，为导线打上网络标签，并预留未使用的 UART、SPI、I2C 功能对应的引脚与电源。



图表 1-22 转接板原理图

1.3.4.4 PCB 绘制

根据 STM32F401RE 评估板规格手册，绘制固定孔与排布排针位置，然后优先将控制电机的 PWM 连接排针放置在四个角，缩短接线距离，防止实际接线时四个电机与排针对应错误，之后将剩余部件合理布局在转接板中，因为自建元件库中已有 3D 模型，可直观查看当前 PCB 的 3D 布局，尽最大可能避免元件反插，三维空间交错等情况发生。



图表 1-23 转接板的 PCB 3D 布局

第二章 存在问题与解决方案

2.1 原理图少连线

在第一版原理图中，各部件均打上了 5V 或 3V3 的网络标签，而在 STM32F401RE 的引脚上漏打了 5V 与 3V3 网络标签，导致各模块无法供电。

在之后的 PCB 检查中，逐条检查导线进行检查，查看连线是否均有器件连接到左右的 STM32F401RE 排针上，防止此类错误再次发生。

2.2 PCB 少过孔

在第二版转接板中，焊接元器件后发现模块无法使用，经排查发现 3V3 的连线缺少了一个过孔，回到 Altium Designer 软件中检查发现，因自动布线的原因，3V3 连线有一处上层铜和下层铜走线已走到同一位置，但因此处距离其他导线较近，无法打过孔，板信息显示连线率未达到 100%而 PCB 中已无走线引导线（走线引导线已重合成一个点）

本次失败为后续走线积累了经验，在板信息未达到 100%时可检查是否是缺少过孔导致的。

2.3 元器件布局混乱

在第四、五版转接板中，部分模块预期位置与实际位置旋转了 180 度，出现了需要反接情况，导致空间上交错。

因初期使用官方组件库，每次布局 PCB 时对于模块插入排母的方向需拿着模块进行判断，效率低下且容易发生错误，为防止此类错误再次发生，故建立小组的元件库并绘制各模块，仔细检查模块方向，并添加了 3D 元件体，将 3D 元件体与 PCB 图的焊盘孔位、固定孔位、布局方向、边框大小进行相互对照，互相校验，提高了元件布局的正确率。

2.4 UART2 无法使用

在进行手机蓝牙控制 STM32 实验时，我们起先选用了 USART2 进行实验，但却不能成功实验。于是我们对 MxCube 设定和代码进行了多次检查，确认无误。然后使用万用表对 STM32 进行了检查，也没有发现问题。于是我们选择使用 USART6 进行一次实验来进行最后的确认。使用 USART6 进行的试验没有发生任何问题，所以我们确定是 USART2 的问题。在查询许多资料后，我们了解到 NucleoF401RE 评估板是默认与板上集成的 ST-LINK 相连的，并不能直接使用。

5.7 USART communication

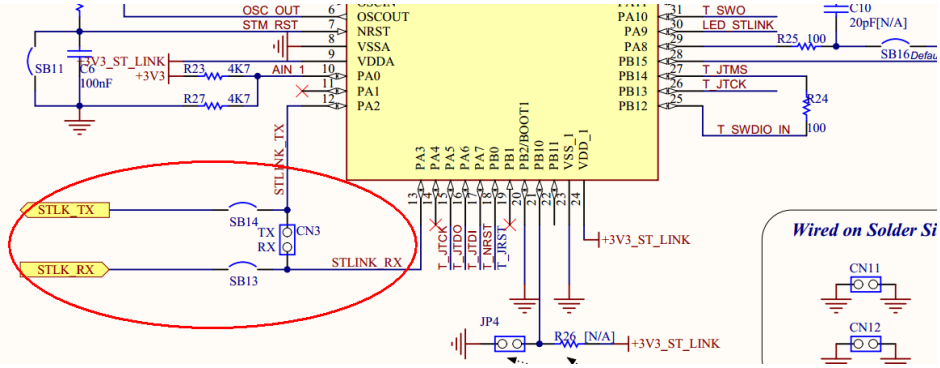
The USART2 interface available on PA2 and PA3 of the STM32 microcontroller can be connected to ST-LINK MCU, STMicroelectronics Morpho connector or to Arduino connector. The choice can be changed by setting the related solder bridges. By default the USART2 communication between the target MCU and ST-LINK MCU is enabled in order to support Virtual Com Port for mbed (SB13 and SB14 ON, SB62 and SB63 OFF). If the communication between the target MCU PA2 (D1) or PA3 (D0) and shield or extension board is required, SB62 and SB63 should be ON, SB13 and SB14 should be OFF. In such case it possible to connect another USART to ST-LINK MCU using flying wires between Morpho connector and CN3. For instance on NUCLEO-F103RB it is possible to use USART3 available on PC10 (TX) & PC11 (RX). Two flying wires need to be connected as follow:

- PC10 (USART3_TX) available on CN7 pin 1 to CN3 pin RX
- PC11 (USART3_RX) available on CN7 pin 2 to CN3 pin TX

图表 2-1 STM32F401RE 评估板用户手册

如上用户手册中有一段文字描述，说明使用 USART2 的时候需要做一些改动：焊接 SB62，SB63，拆掉 SB13，SB14。从原理图中我们也能看出其中的关系：

于是我们最终选择 USART6 来进行蓝牙数据传输。



图表 2-2 STM32F401RE 评估板原理图

2.5 I2C2 无法使用

使用 I2C2 时使用了 PB3，查阅评估板用户手册后发现，PB3 被预定义为 SWO 调试功能，在评估板的连线上接入了调试引脚，故无法将 PB3 再使用为 I2C 功能。

2.6 PWM 无法驱动电机

在初期配置 TIM 后，输出的 PWM 脉冲无法控制电机转动，电机发出“滴滴”警报声，因身边没有示波器，为确认 TIM 和 GPIO 功能配置正确，PWM 脉冲的频率和占空比为预期值，我们使用 PWM 脉冲控制发光二极管进行调试：将 PWM 频率调制 5Hz 以下，占空比调为 50%，观察发光二极管的闪烁频率；将 PWM 频率调为 50Hz，修改占空比，观察发光二极管的亮度，以此成功配置 PWM，继而能成功控制电机旋转。

2.7 无法与 HMC5883L 通信

初期，调试 GY-86 时成功与另外两个测量芯片通信，但是无法与 HMC5883L 通信。在课堂上和钟宇晨讨论，查阅相关资料后，了解到 HMC5883L 是接入 MPU6050 的旁路 I2C 总线上的，需要操作 MPU6050 对 HMC5883L 进行通信或令 MPU6050 放开对 HMC5883L 的“控制权”，通过对 MPU6050 进行相关配置，成功与 HMC5883L 通信，获取到 HMC5883L 的数据。

2.8 中断内使用延时函数

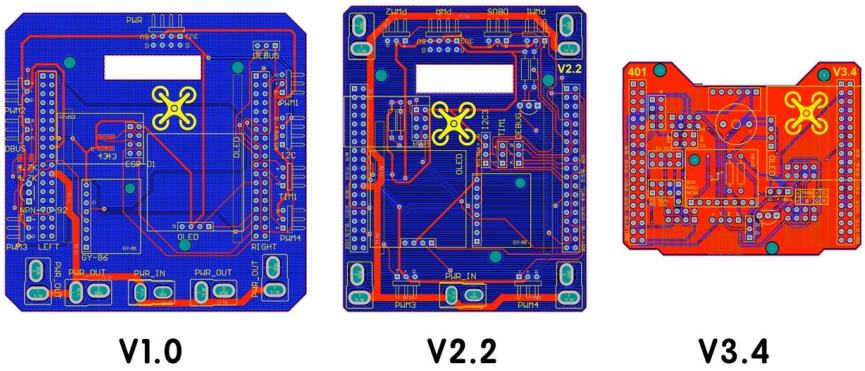
在中断程序中使用了延时函数后，程序会阻塞在延时函数中无法继续运行，经查阅资料得知，延时函数中使用的时间戳也是由中断进行计数的，时间戳更新中断的优先级是 15（即最低优先级），进入中断后无法再执行最低级中断，时间戳不会更新，导致延时函数阻塞，可以在主函数中以一定频率调用此函数，待下学期系统学习操作系统后再使用中断的方式。

第三章 执行情况与完成度

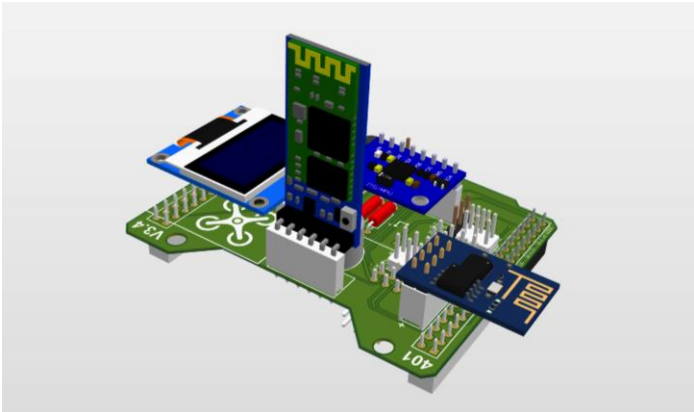
我们小组完成了老师布置的所有任务，并在任务的基础上，根据自己对四轴的理解和为了一年后四轴飞行器能稳定飞行，新增了 OLED、蜂鸣器、RGB 灯、GPS 等模块，额外编写了 OLED 实时查看数据、模块离线检测警报、上位机图表可视化监测等相关代码，整机代码能流畅运行，稳定性满足预期。

3.1 转接板制作

在两个月时间内我们小组共绘制了 8 版转接板，从第一版至第六版，体积缩小 50%，可连接元器件数量增加 50%，包含接收机、GY-86、蓝牙、四通道 PWM 输出，并预留蜂鸣器、ESP8266、超声波测距、四个 RGB 的接口，所有接口均能正常使用。



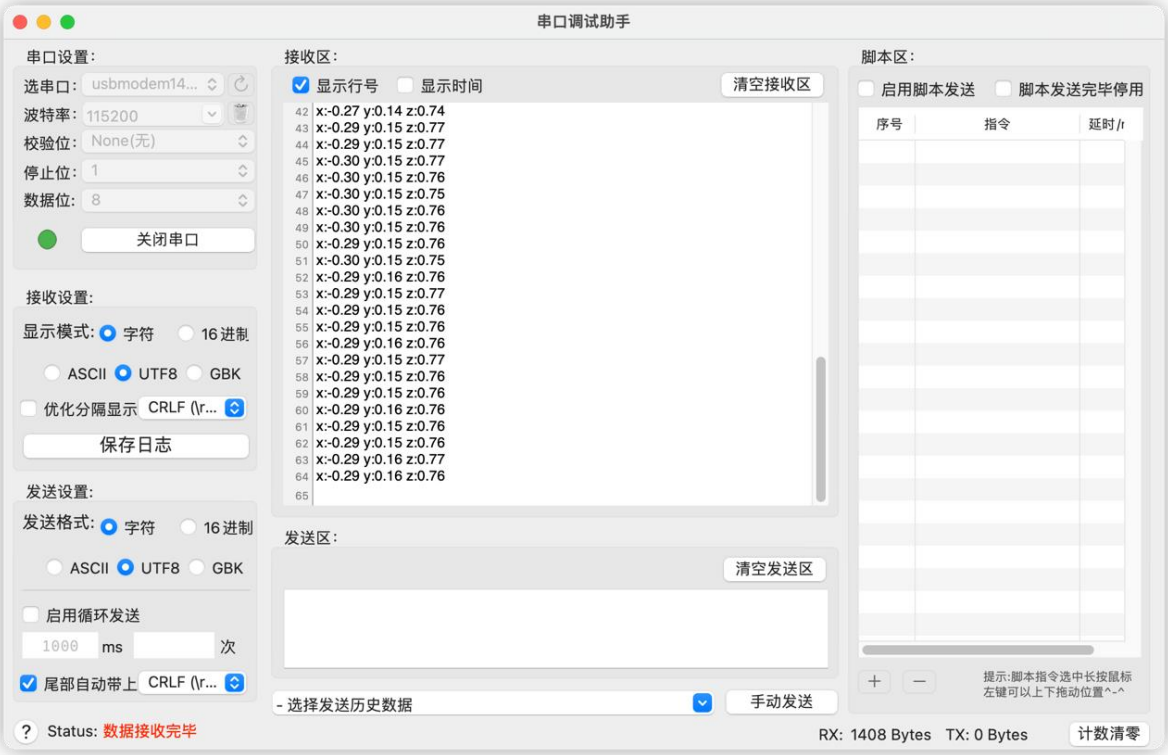
图表 3-1 转接板版本迭代图



图表 3-2 转接板 3D 视图

3.2 姿态传感器数据获取

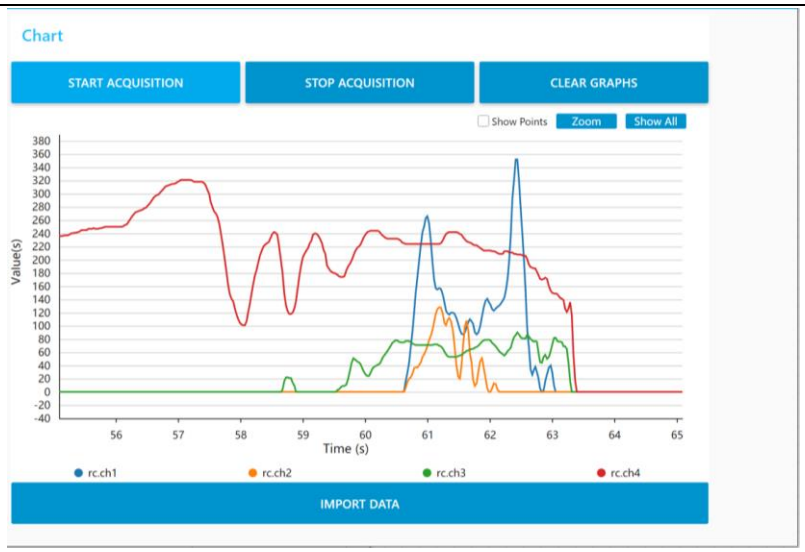
成功通过 I2C 与 GY-86 进行通信，获取原始数据，并按照参考手册的公式进行数据解析，将解析数据存入全局结构体变量，通过调试串口将解析数据发送至上位机中。



图表 3-3 上位机接收 GY-86 解析数据

3.3 遥控器命令解析

学习了 UART 通信，DMA，中断等相关知识后，参考官方驱动代码，实现了遥控器数据的获取与解析，将解析数据存入全局结构体变量中，通过 Cube Monitor 软件监测变量值的变化情况。



图表 3-4 使用 Cube Monitor 监测解析后的遥控器 4 个通道变量值

3.4 蓝牙数据收发

3.4.1 消息收发

将 HC-05 模块通过 USB 转 TTL 模块连接到电脑上，此时 HC-05 处于自动连接工作模式。手机打开蓝牙串口助手，通过蓝牙与 HC-05 连接。在电脑的调试助手中设定好波特率等数据后打开串口，通过电脑上的调试助手和手机上的蓝牙串口助手进行数据传输，实现手机通过蓝牙，向 STM32 单片机发送消息，STM32 接收到消息之后原封不动的返回给手机。


```
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart)
{
  Uart6_Rx_Cnt = 0;
  if(Uart6_Rx_Cnt >=255)
  {
    memset(RxBuffer,0x00,sizeof(RxBuffer));
    HAL_UART_Transmit(&huart6,(uint8_t*)&"data overflow\r\n",15,0xffff);
  }
  else
  {
    RxBuffer[Uart6_Rx_Cnt++] = aRxBuffer;
    HAL_UART_Transmit(&huart6,(uint8_t*)&RxBuffer,1,0xffff);
    while(HAL_UART_GetState(&huart6)==HAL_UART_STATE_BUSY_TX);
    Uart6_Rx_Cnt = 0;
    memset(RxBuffer,0x00,sizeof(RxBuffer));
    }
  HAL_UART_Receive_IT(&huart6,(uint8_t*)&aRxBuffer,1);
}
```

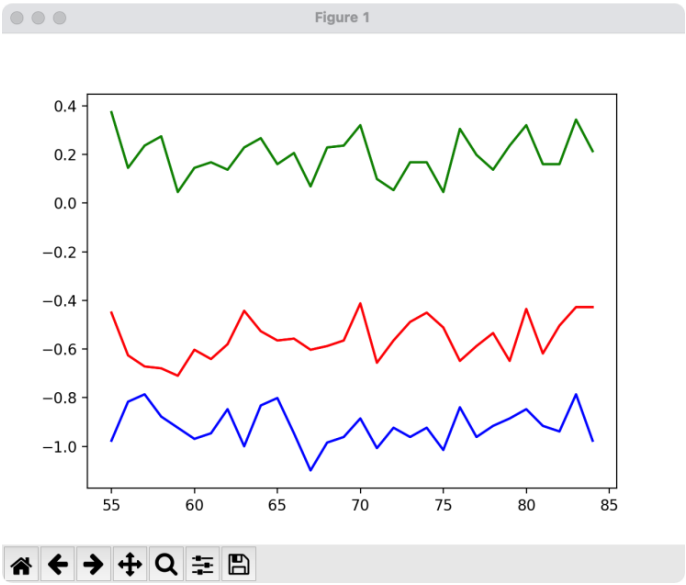
图表 3-5 使用蓝牙模块进行数据收发实验截图

3.4.2 上位机成功接收蓝牙数据帧并动态绘制折线图

成功将数据转换成数据帧并发送，上位机通过串口线或蓝牙能正常接收数据帧。

图表 3-6 上位机接收的数据帧（FF 为帧头，0D 为帧尾）

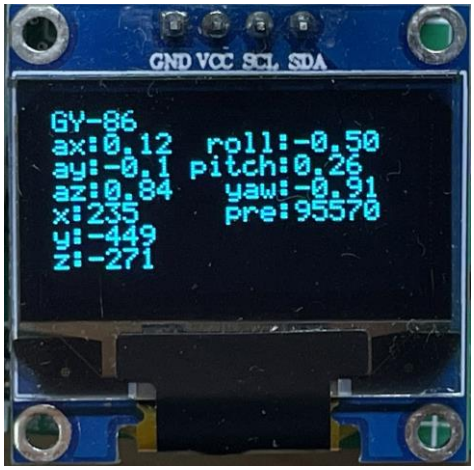
能使用 python 程序能接收数据帧并进行识别并解析，使用 matlabplot 动态绘制折线图。



图表 3-7 上位机绘制的折线图

3.5 OLED 数据显示

编写的 OLED 驱动能正确显示字符、字符串、整数、浮点数。应用程序能根据获取到的数据在 OLED 上显示。



图表 3-8 OLED 动态显示 GY-86 数据

3.6 遥控操控电机

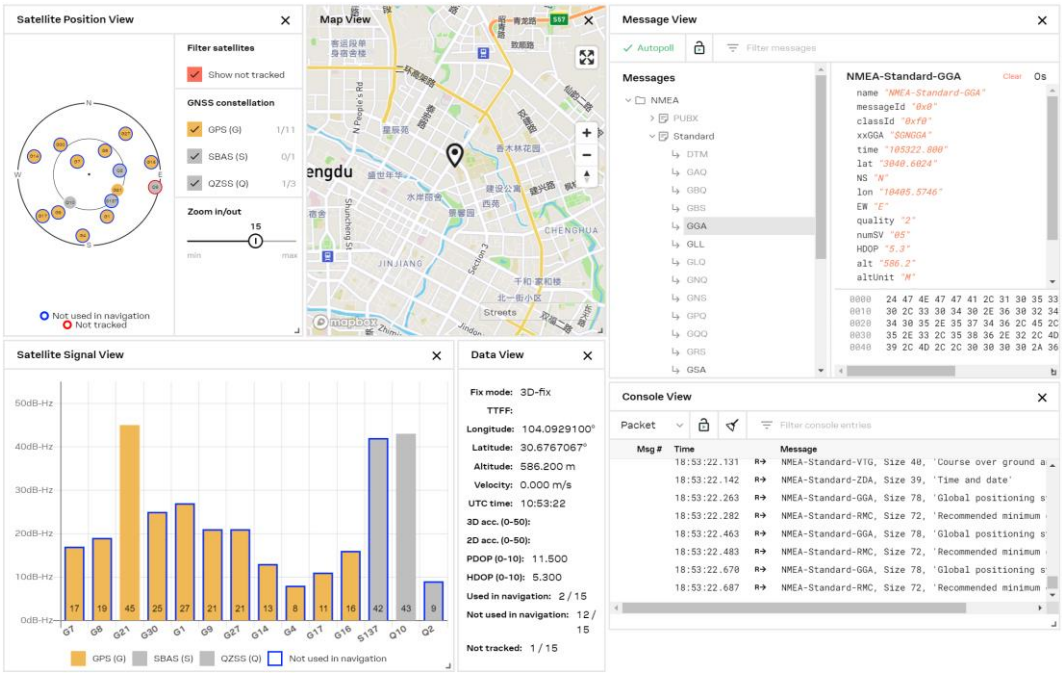
能通过遥控器控制飞机的飞行方向，电机根据飞行方向做出响应。

图表 3-9 遥控控制电机转动

3.7 GPS 获取定位数据

编写 GPS 相关驱动代码，能在裸机上成功获取 GPS 定位信息，时间等数据，

因为 GPS 需在开阔环境使用且主控板 UART 有限，目前暂未集成到四轴飞行器上。



图表 3-10 GPS 获取定位数据

第四章 分工协作与交流情况

4.1 小组分工情况

姓名	主要工作
屈子铭	电机、OLED、RGB、蜂鸣器驱动编写；离线检测应用编写；上位机可视化代码编写；整体进度统筹
米锦江	遥控器驱动编写；GY86MPU6050 驱动编写
厉浩然	GPS 驱动编写、蓝牙发送数据帧应用编写
杨坤	转接板绘制
贺奕嘉	GY86HMC5883L、MS5611 驱动编写

表格 4-1 小组分工表

根据廖勇老师四轴飞行器课程的挑战任务要求，每位成员主攻一至两个传感器模块的驱动编写，同时学习其他传感器的通信原理。软件集成时由组长与组员进行联调。

4.2 团队间交流情况

听了范宇小组对 GY-86 的讲解后，成功通过 I2C 与 GY-86 的 MPU6050 进行通信，获取原始数据，并按照参考手册的公式进行数据解析，将解析数据存入全局结构体变量，通过调试串口将解析数据发送至上位机中。

编写 GY-86 时无法获取 HMC5883L 的传感器数据，在课堂上和钟宇晨讨论，查阅相关资料后，了解到 HMC5883L 是接入 MPU6050 的旁路 I2C 总线上的，需要操作 MPU6050 对 HMC5883L 进行通信或令 MPU6050 放开对 HMC5883L 的“控制权”，通过对 MPU6050 进行相关配置，成功与 HMC5883L 通信，获取到 HMC5883L 的数据。

和任峙皓讨论了遥控数据解析的问题和 dbus 信号与 PPM 信号的区别。

和郭鑫泓讨论调试了使用寄存器输出 PWM 脉冲控制 LED 亮度的相关代码。

参考文献

[1] STM32F401xB/C and STM32F401xD/E advanced Arm®-based 32-bit MCUs Reference manual[M],Italy and France: STMicroelectronics.2018.

[2] stm32f401re Datasheet - production data[M], Italy and France: STMicroelectronics.2015.

[3] STM32 Nucleo-64 boards (MB1136) User manual[M] , Italy and France: STMicroelectronics.2020.

致谢

首先，我们要特别感谢我们的指导老师廖勇老师。本报告的工作是在我们的指导教师廖勇老师的悉心指导下完成的。进阶式挑战性综合项目的过程是艰难的，特别是在开始阶段显得尤为坎坷，但在廖勇老师每周两次的集中讨论指导下，我们顺利度过了开头难。而在课程学习中，我们也学会了对本专业所学理论知识的应用，学会了在应用中发现问题和解决问题的方法，加深了对专业知识的掌握和理解，所取得的这些进步与廖老师仔细、耐心的指导和开放性的思维引导是分不开的。在此对廖老师表示深深感谢。

其次，我们也要非常感谢所有其他参与本进阶式挑战性综合项目的 20 级同学们。在学习过程中，不同组的同学分别讲解不同部分的理论知识以及分享各自在实践应用过程中所遇到的难题，为我们在自己设计与实现时提供了思路与弥足宝贵的帮助。最后感谢我们小组的每一位成员的辛勤努力与认真参与，让我们共同完成了本课程设计。

再次衷心感谢在百忙之中评阅总结报告和参加答辩的各位专家、教授！