认真读完整篇文档有利于您更好的理解整个平衡小车程序。

开发平台:MDK5.1

1、我们的代码使用 MPU6050 的 INT 的引脚每 5ms 触发的中断作为控制的时间基准，严格保证系统的时序！

2.根据不同阶层的学习者，我们提供了复杂程度不同的代码：

1.针对普通用户，提供了以下三个代码：

MiniBalanceV3.5 平衡小车源码（DMP 版）

MiniBalanceV3.5 平衡小车源码（互补滤波版）

MiniBalanceV3.5 平衡小车源码（卡尔曼滤波版）

以上代码除了使用 DMP、卡尔曼滤波、互补滤波分别获取姿态角外，还提供了超声波避障代码。

2.针对入门用户，提供以下代码：

MiniBalanceV3.5 平衡小车源码（精简入门版）

去除所有附加的代码，使用最少的代码量实现小车直立。

3.针对大神用户，提供以下代码：

MiniBalanceV3.5 平衡小车源码（顶配版含无线模块和线性 CCD）

普通版的基础上增加无线模块的驱动和线性 CCD 的采集代码，大家可以拓展体感控制和小车巡线。

3.整个程序应用了 STM32 大量的资源：

ADC 模块：采集电阻分压后的电池电压,采集模拟 CCD 摄像头数据
TIM1：初始化为 PWM 输出，CH1，CH4 输出双路 10KHZ 的 PWM 控制电机
TIM2：初始化为正交编码器模式，硬件采集编码器 1 数据
TIM3：CH3 初始化为超声波的回波采集接口。
TIM4：初始化为正交编码器模式，硬件采集编码器 2 数据
USART1：通过串口 1 把数据发到串口调试助手
USART3：通过串口 3 接收蓝牙遥控的数据，接收方式为中断接收。并发送数据给 app。
IIC：利用 IO 模拟 IIC 去读取 MPU6050 的数据，原理图上 MPU6050 链接的是 STM32 的硬件 IIC 接口，但是因为 STM32 硬件 IIC 不稳定，所以默认使用模拟 IIC，大家可以自行拓展。
SPI：利用 IO 模拟 SPI 去驱动 OLED 显示屏,硬件 SPI 驱动 NRF24L01
GPIO：读取按键输入，控制 LED，控制电机使能和正反转
SWD：提供用于在线调试的 SWD 接口
EXTI:由 MPU6050 的 INT 引脚每 5ms 触发一次中断，作为控制的时间基准

4.程序主要用户文件说明如下：

- Source Group1
  - Startup_stm32f10x_md.s :stm32 的启动文件
- User
  - Minibalance.c： 放置主函数，并把超声波读取和人机交互等工作放在死循环里面。
- SYSTEM
  - Delay.c：提供系统延时初始化函数及相关的函数
  - Sys.c:提供时钟、中断、系统初始化函数
  - Usart1.c：提供串口 1 初始化函数及相关的函数
- HARDWARE
  - Led.c：提供 LED 初始化函数及相关的函数
  - Key.c：提供按键初始化函数及相关的函数，如单击、双击、长按检测。
  - Oled.c：提供 OLED 初始化函数及相关的函数
  - adc.c：提供 ADC 初始化函数及相关的函数，如电池电压检测,线性 CCD 摄像头初始化和驱动程序。
  - Timer.c：提供超声波的初始化代码和采集代码。
  - Motor.c 提供电机控制初始化函数
  - Encoder.c 提供编码器采集相关函数
  - Ioi2c.c：提供模拟 IIC 初始化函数及相关的函数
  - Usart3.c：提供串口 3 初始化函数及相关的函数，其中蓝牙遥控使用的串口 3 接收中断函数在这里面。
  - Exti.c：提供外部中断初始化代码。
- Balance
  - Control.c 提供全部的控制函数，并放在由 MPU6050 触发的外部中断里面执行
  - Inv_mpu.c：MPU6050 内置 DMP 的相关库文件
  - Inv_mpu_dmp_motion_driver.c：MPU6050 内置 DMP 的相关库文件
  - Fitler.c：提供平衡小车常用的滤波算法，如卡尔曼滤波，互补滤波
  - Mpu6050.c：提供 MPU6050 初始化函数及相关的函数
  - Show.c：提供用于数显和 APP 使用的相关函数

5.滤波算法

平衡小车获取姿态角的滤波算法一般为卡尔曼滤波和一阶互补滤波。但是 MPU6050 内置的 DMP 可以直接输出和姿态相关的四元数。所以，常用的有三种方法可以获取角度。

本程序内置了多种滤波算法，获取四元数的算法放在 DMP 相关的库文件里面，卡尔曼滤波和互补滤波放在 filter.c 里面。除了精简版代码外，其他每个代码都提供了 Way_Angle 变量控制滤波算法。

Way_Angle=1：通过 DMP 获取四元数，并算出角度

Way_Angle=2：通过卡尔曼滤波对陀螺仪和重加进行数据融合

Way_Angle=3：通过互补滤波对陀螺仪和重加进行数据融合

根据长时间的实践，以 DMP 输出的四元数表示的角度和卡尔曼滤波最为稳定。互补滤波的效果稍差，但也是很不错的。

6.控制算法

- 直立控制：PD 控制，这是最核心的控制，其他的控制都是相对直立控制而言都是干扰。
- 速度控制：PI 控制 对编码器信息进行低通滤波可以削弱电机控制的比重，提高系统稳定性。
- 转向控制：PD 控制 结合了 Z 轴陀螺仪 PD 控制;