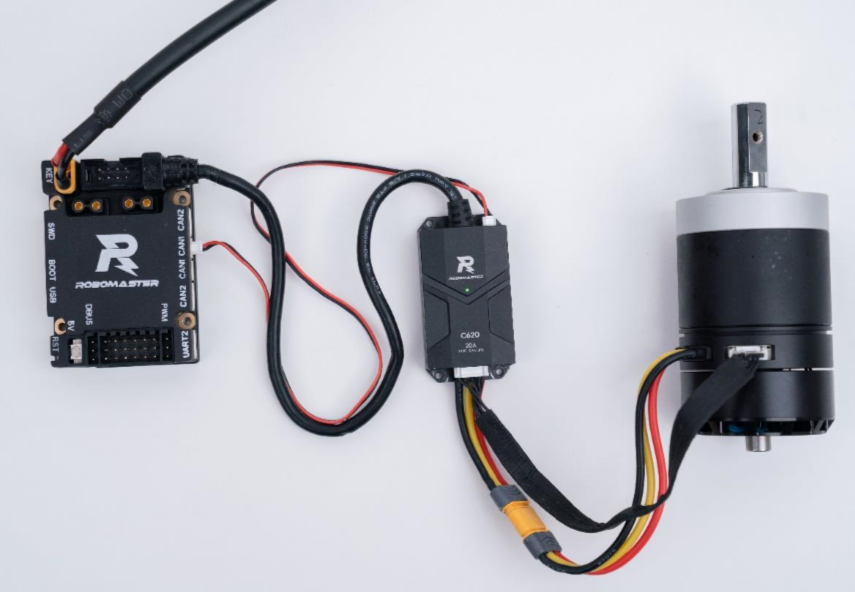
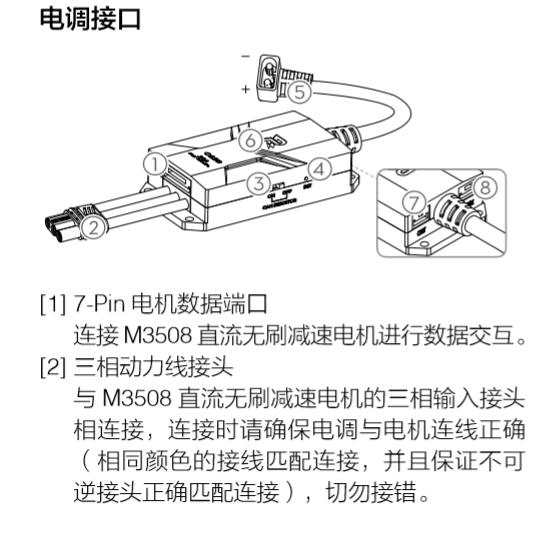
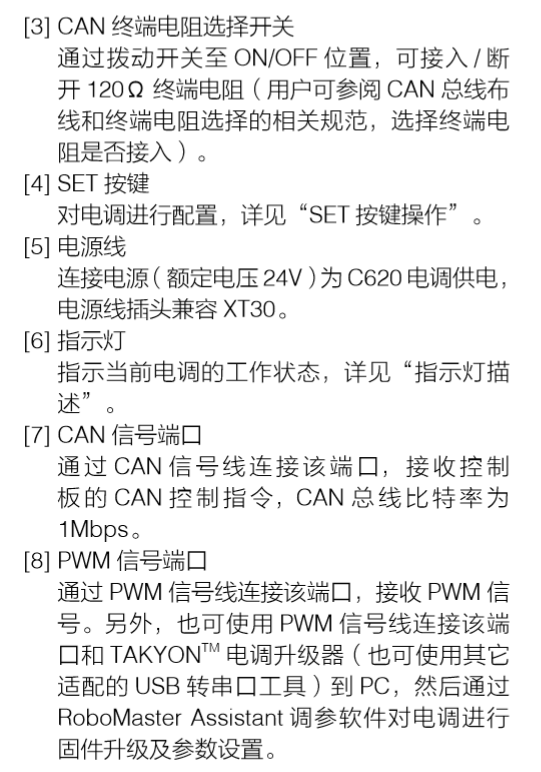
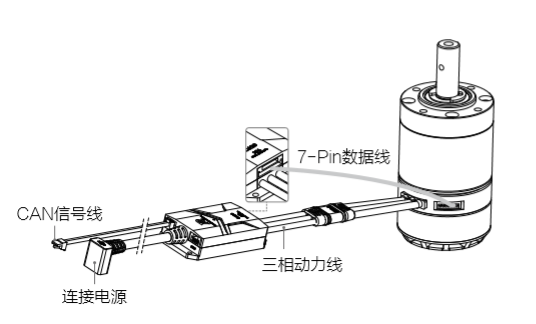
# M3508电机和C620电调



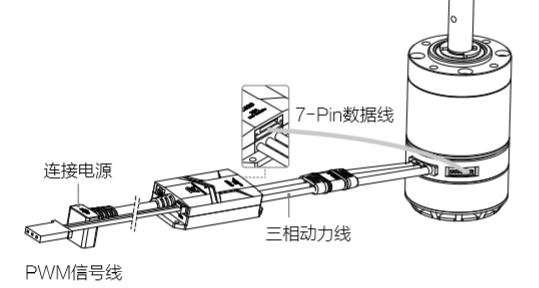




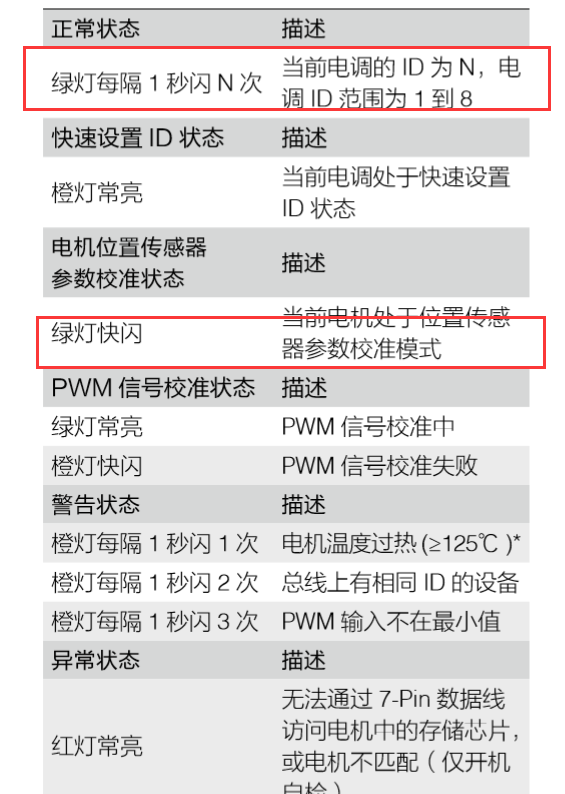
通过CAN信号线控制的连接方法



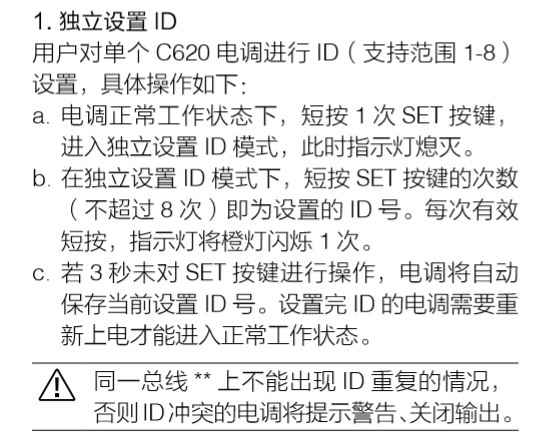
通过PWM信号线控制的方法

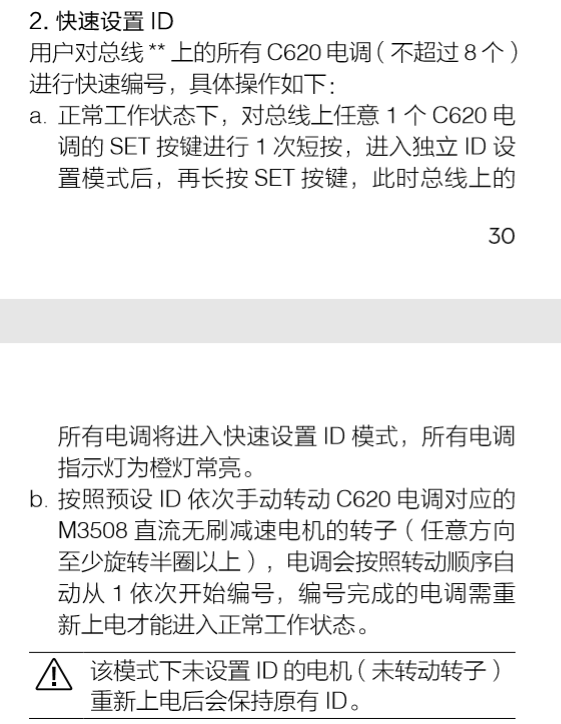


指示灯含义

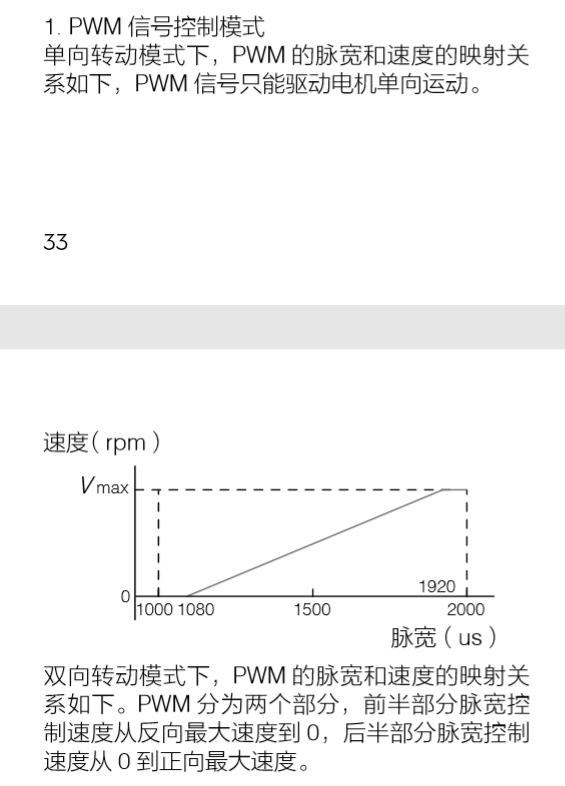


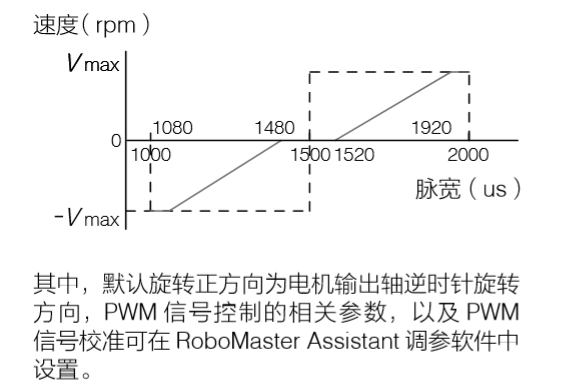
使用SET按键设置ID





PWM控制信号说明



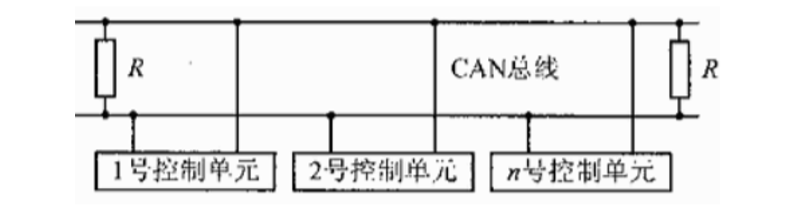


# CAN总线

CAN 是控制器域网 (Controller Area Network, CAN) 的简称，是由研发和生产汽车电子产 品著称的德国 BOACH 公司开发，并最终成为国际标准（ISO11898），CAN 是国际上应用

最广泛的现场总线之一。 在北美和西欧，CAN 总线协议已经成为汽车计算机控制系统和嵌 入式工业控制局域网的标准总线，并且拥有以 CAN 为底层协议专为大型货车和重工机械车 辆设计的 J1939 协议。

CAN 总线由 CAN\_H 和 CAN\_L 两根线构成，各个设备一起挂载在总线上。



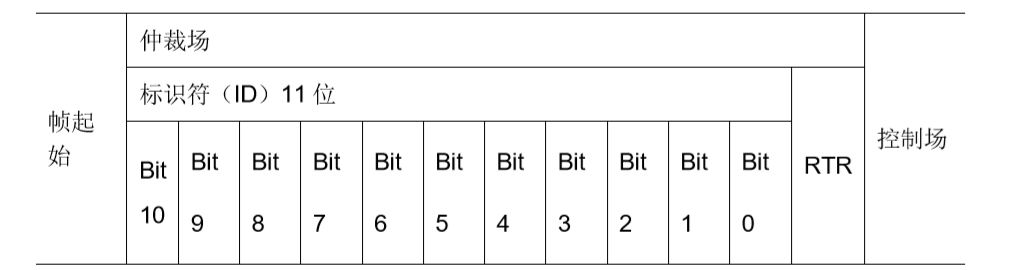
RoboMaster 系列电机也采用 CAN 协议进行通信，CAN 协议比较复杂，一个完整的数据帧

由下图中的各个部分组成：



这里重点介绍 CAN 的仲裁场和数据场的内容。每一个挂载在 CAN 总线 上的 CAN 都有一个自己独属的 ID，每当一个设备发送一帧数据时，总线其他设备会检查这个 ID 是否是自己需要接收数据的对象，如果是则接收本帧数据，如果不是则忽略。

ID 存储在数据帧最前头的仲裁场内，CAN 的 ID 分为标准 ID 和拓展 ID 两类，标准 ID 长度 为 11 位



在通过 ID 判断本帧数据可以接收后，控制场中的 DLC 规定了本帧数据的长度，而数据场内 的数据的大小为 8 Byte，即 8 个 8 位数据。CAN 总线的一个数据帧中所需要传输的有效数 据实际上就是这 8Byte。



# C620电调的CAN通信

在配套的C620手册中找到CAN协议描述



这是电调接收报文格式，即如果要发送数据给 1 号到 4 号电调，控制电机的输出电流，从而 控制电机转速时，需要按照表中的内容，将发送的 CAN 数据帧的 ID 设置为 0x200，数据域 中的 8Byte 数据按照电调 1 到 4 的高八位和第八位的顺序装填，帧格式和 DLC 也按照表中内容进行设置，最后进行数据的发送

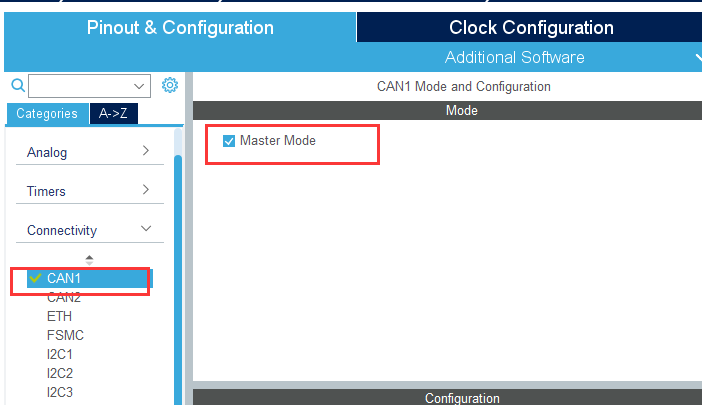
下表是C620向32发送的报文，32需要按下面的格式进行解析。

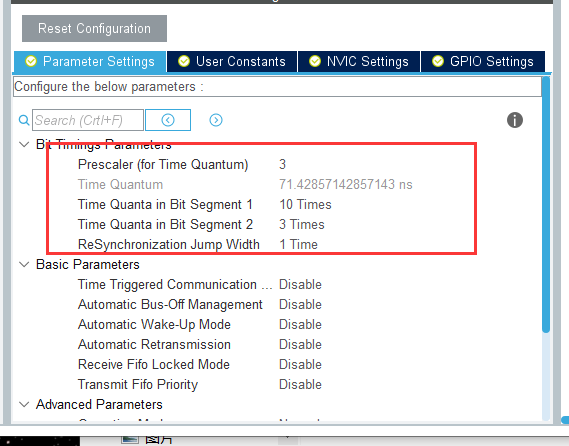


可以发现按这样的格式发送，一帧完整的CAN报文就完全可以将所需要的数据发送出去，不许将数据拆分成许多不同的数据帧，对比遥控器的串口每次只能发送一个字节的数据，一帧完整的数据需要发送18个字节，减少了数据拼接时丢帧的可能。

首先根据接收到的 ID 判断究竟接收到的是哪个电调发送来的数据，手册中规定 1 号电调 ID 为 0x201，2 号为 0x202，3 号为 0x203，4 号为 0x204。判断完数据来源之后，就可以按照手册中的数据格式进行解码，通过高八位和第八位拼接的方式，得到电机的转子机械角度，转子转速，转矩电流，电机温度等数据。

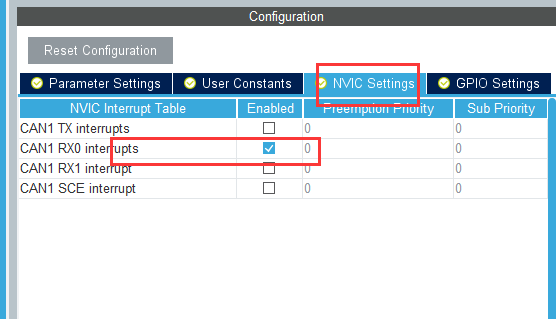
# CAN 在 cubeMX 中的配置





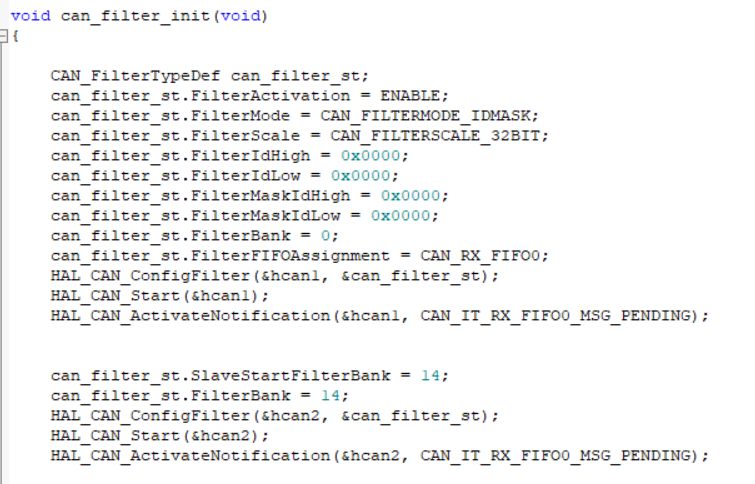
CAN的通信速率按照C620手册要求的1Mbps配置

71.42857142857143𝑛𝑠 ∗(10 +3+1) = 1000𝑛𝑠 = 1𝑢𝑠=1mbps



开启对应的CAN接收中断

# CAN的初始化

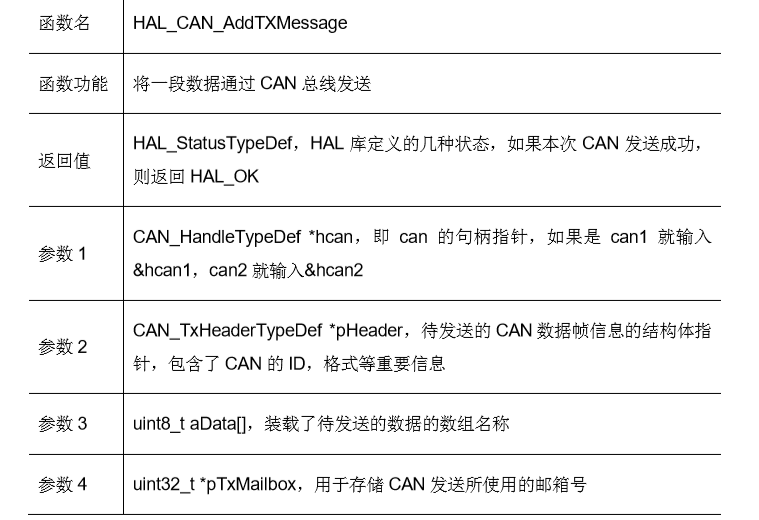


因为在有些设备比较多的场合，单片机只需要接收特定设备的CAN报文，不要用将所有数据全都接收，所有CAN总线有报文屏蔽的功能。在32中，因为设备比较少，而且都是32在控制，所有将上面的代码是将屏蔽功能关闭，接收所有报文。

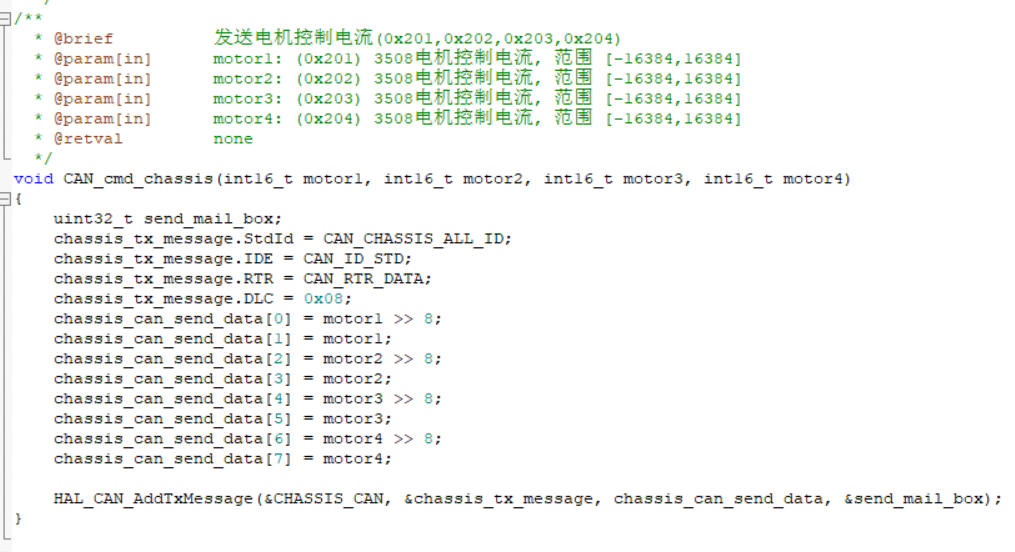
# CAN函数发送

HAL\_StatusTypeDef HAL\_CAN\_AddTxMessage(CAN\_HandleTypeDef \*hcan,

CAN\_TxHeaderTypeDef \*pHeader, uint8\_t aData[], uint32\_t \*pTxMailbox)



大疆提供的底盘电机CAN发送函数

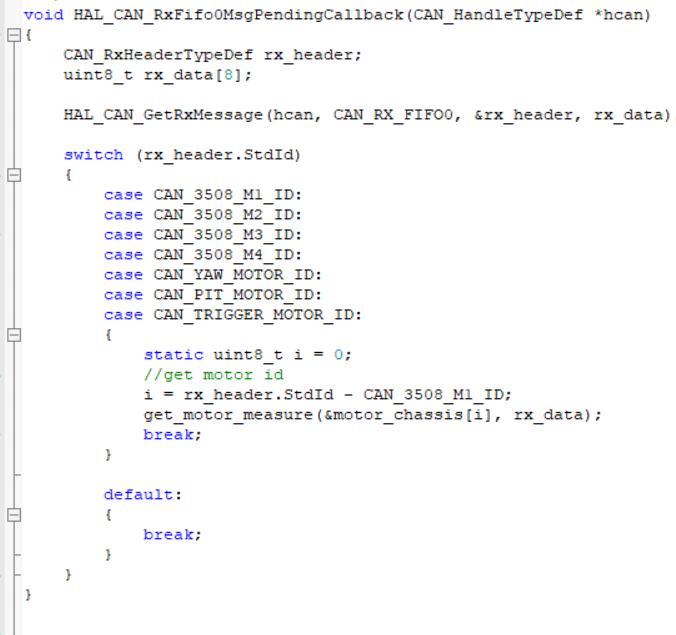


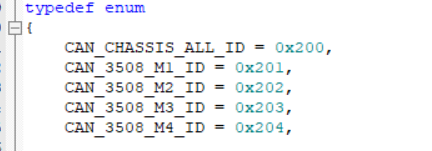
可以看到本质上最后还是调用了HAL库提供的CAN发送函数，不过加入了将各个电机的电流拆解成C620手册中规定的发送协议，然后定义了发送的CAN\_CHASSIS\_ALL\_ID = 0x200,就实现了控制底盘四个电机的目标

# CAN总线的数据接收流程

数据接收和串口一样，也是用到了接收完成中断，但是这里不需要考虑数据的完整性，HAL 库提供了 CAN 的接收中断回调函数

HAL\_CAN\_RxFifo0MsgPendingCallback(CAN\_HandleTypeDef \*hcan)，每当 CAN 完成一 帧数据的接收时，就会触发一次 CAN 接收中断处理函数。





在本次的程序中，在中断回调函数中首先判断接收对象的 ID，是否是需要的接收的电调发

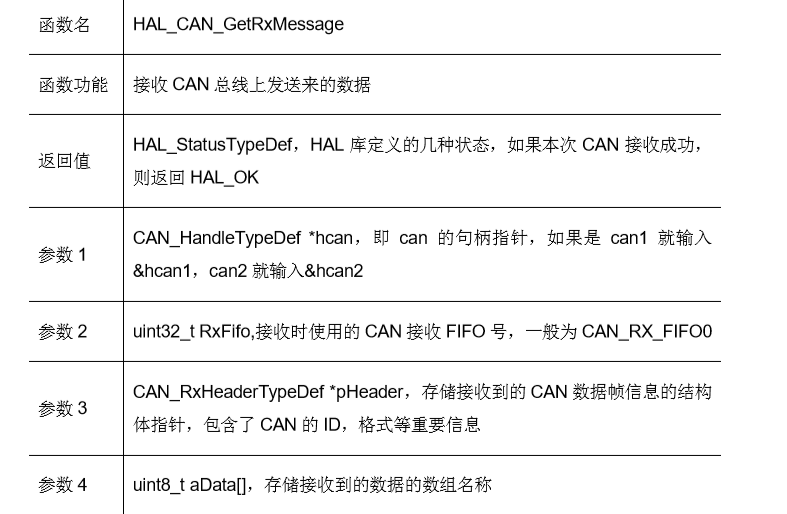
来的数据。完成判断之后，进行解码，将对应的电机的数据装入电机信息数组 motor\_chassis

各个对应的位中。

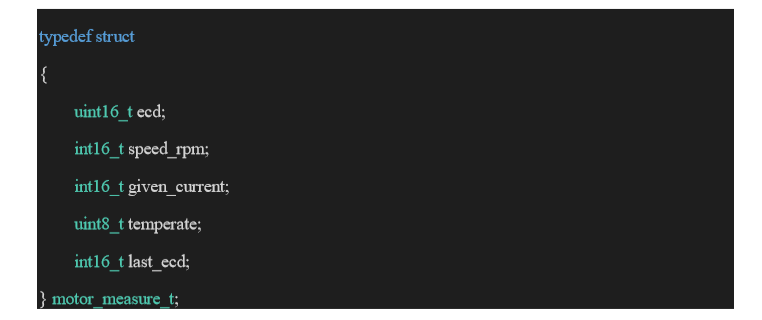
HAL 库提供的接收函数

HAL\_StatusTypeDef HAL\_CAN\_GetRxMessage(CAN\_HandleTypeDef \*hcan, uint32\_t RxFifo,

CAN\_RxHeaderTypeDef \*pHeader, uint8\_t aData[])

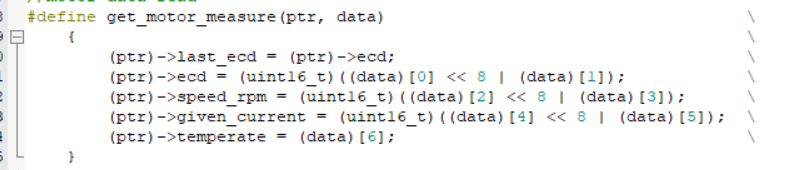


motor\_chassis 为 motor\_measure\_t 类型的数组，其中装有电机转子角度，电机转子转速， 控制电流，温度等信息。



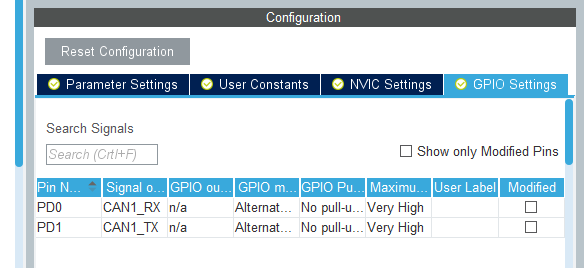
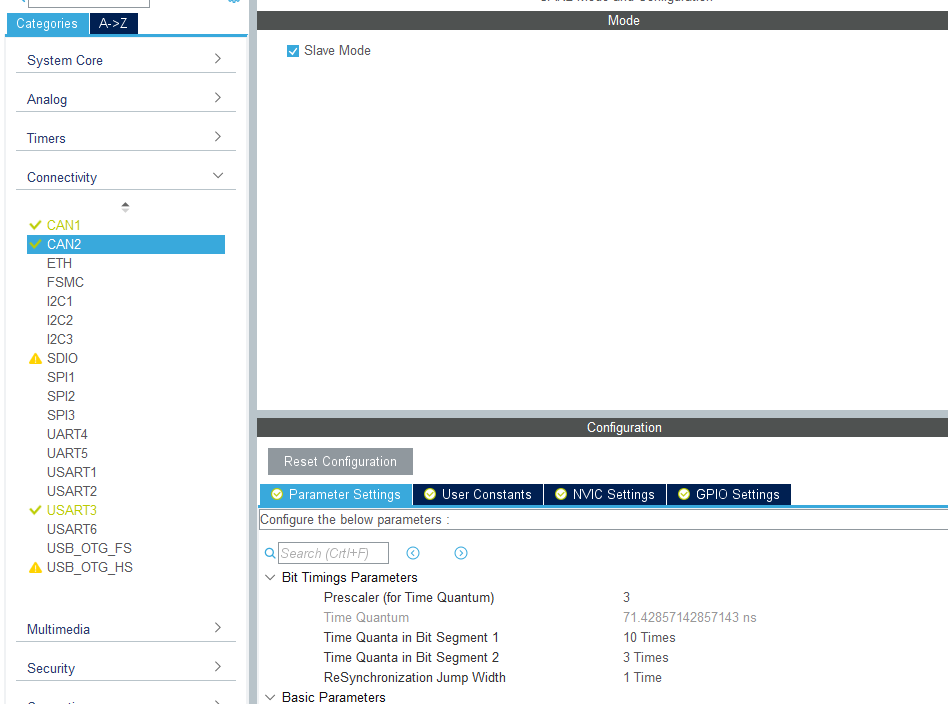
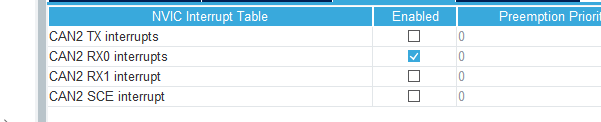
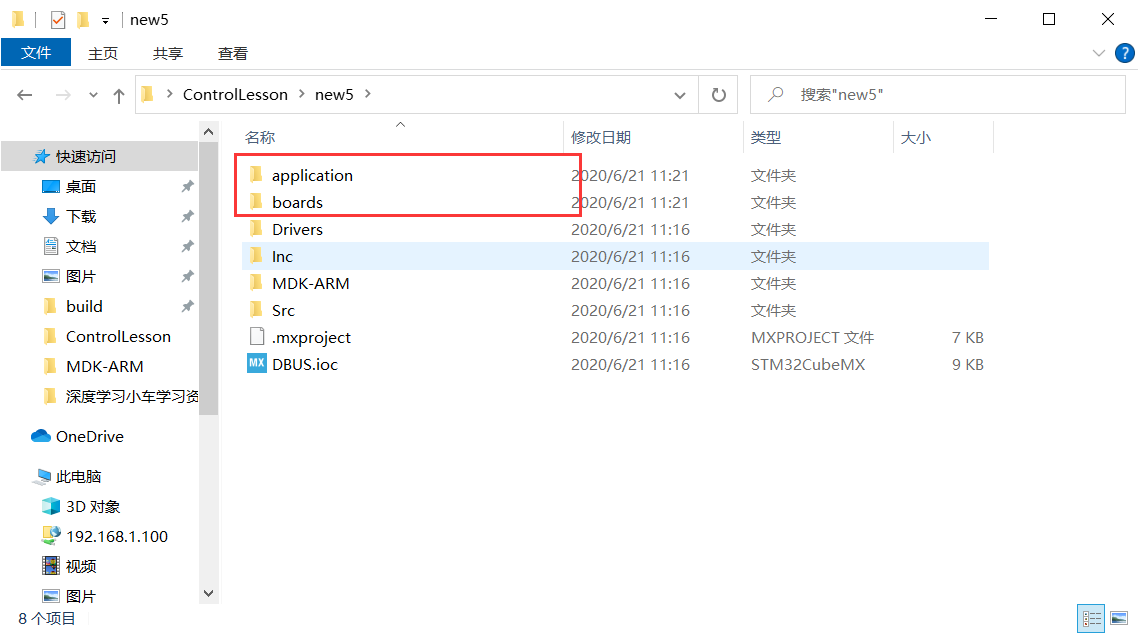
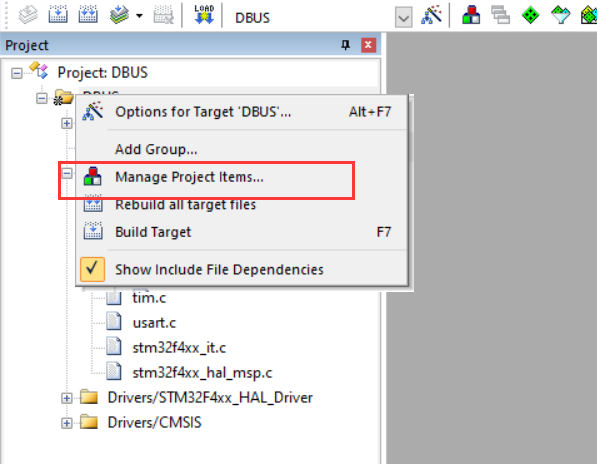
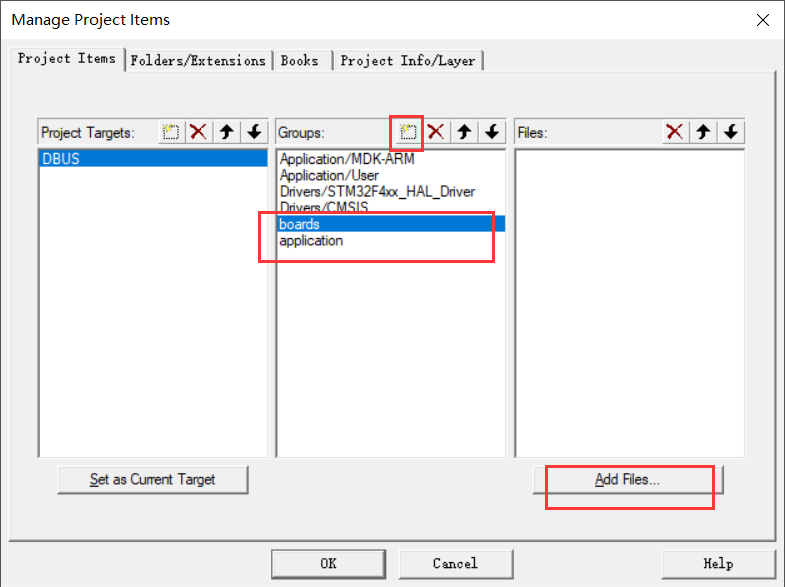
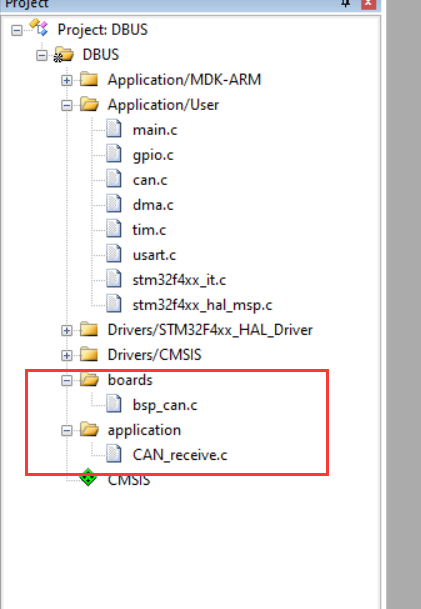
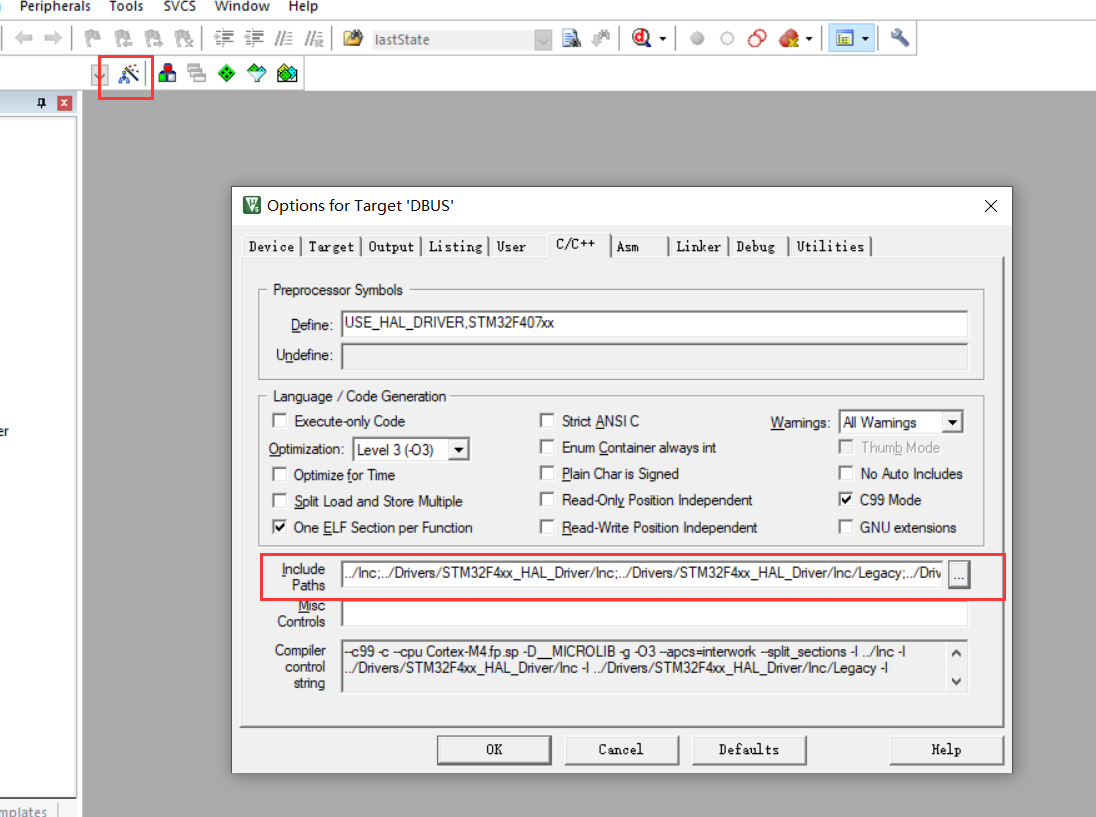
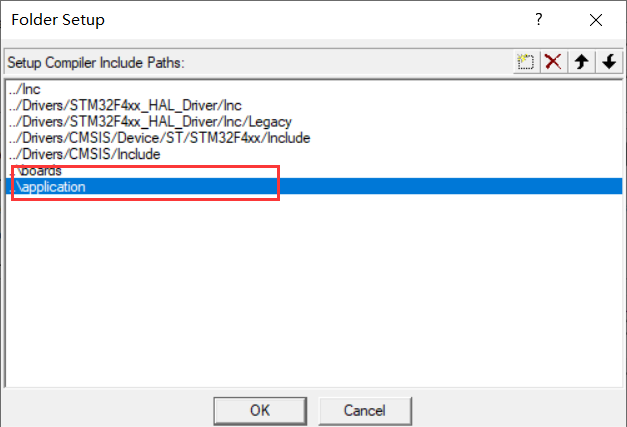
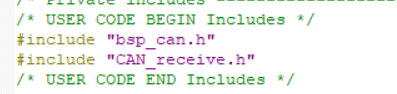
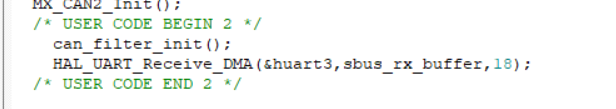
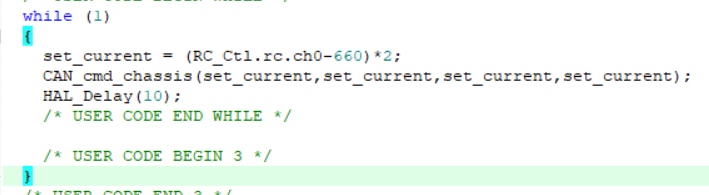
解码功能实际上完成的工作是将接收到的数据按照高八位和第八位的方式进行拼接，从而得

到电机的各个参数



# 实验一

遥控器开环控制电机的电流

1. 将第四课的cubeMX工程文件复制到课程5，可以将加上keil的代码（整个工程文件）一起复制到一个新文件夹，这样就不需要再keil中移植遥控器的代码
2. 打开cubeMX配置can1和can2的参数配置成一样，为了兼容大疆的驱动文件
3. 注意要将CAN1的引脚改为说明书上的配置，默认的引脚没有和实际的CAN1的接口连接
   1. 
   2. 
   3. 
4. 移植大疆给的CAN驱动库文件
   1. 
   2. 将源代码复制到工程目录下
   3. 
   4. 
   5. 新建两个文件夹然后将对应的.c文件添加进去
   6. 
   7. 之后将.h文件添加进包含的头文件
   8. 
   9. 点击右边的省略号在弹出的框中添加对应的文件夹
   10. 
5. 在main.c中包含驱动的头文件和遥控器的变量，在.h中声明RC\_Ctl\_t结构体
   1. 
   2. 
6. 初始化CAN总线的屏蔽滤波器为接收所有报文和遥控器DMA
   1. 
7. 将上节课遥控器的中断回调函数移植
8. 在while循环中，遥控器通道0的值是364-1684，假如向映射的目标是-1000到1000，那么就可以先减去1024，得到范围-660到660，在乘上2，得到范围-1320到1320，接近目标范围
9. 
10. 之后调用大疆给的can的发送函数将电流发给四个电调
11. 在Debug中查看对应电调返回的数据（转速，实际电流，温度等）