

# 2025RoboMaster辽宁科技大学COD战队 电控通用控制系统(达妙MC02 STM32H723VGT6)

---

## 1 简介

---

开发工具：Keil V5.38 STM32CubeMX V6.12.0 VsCode

软件环境：Windows11

硬件环境：达妙MC-02开发板 (STM32H723VGT6) [购买链接](#)

编译工具：Arm Compiler V6.19, C/C++编译



本开源部分原理可参考以下文章：

[STM32H7系列FDCAN配置成经典CAN的经验教程和注意事项 - 知乎](#)

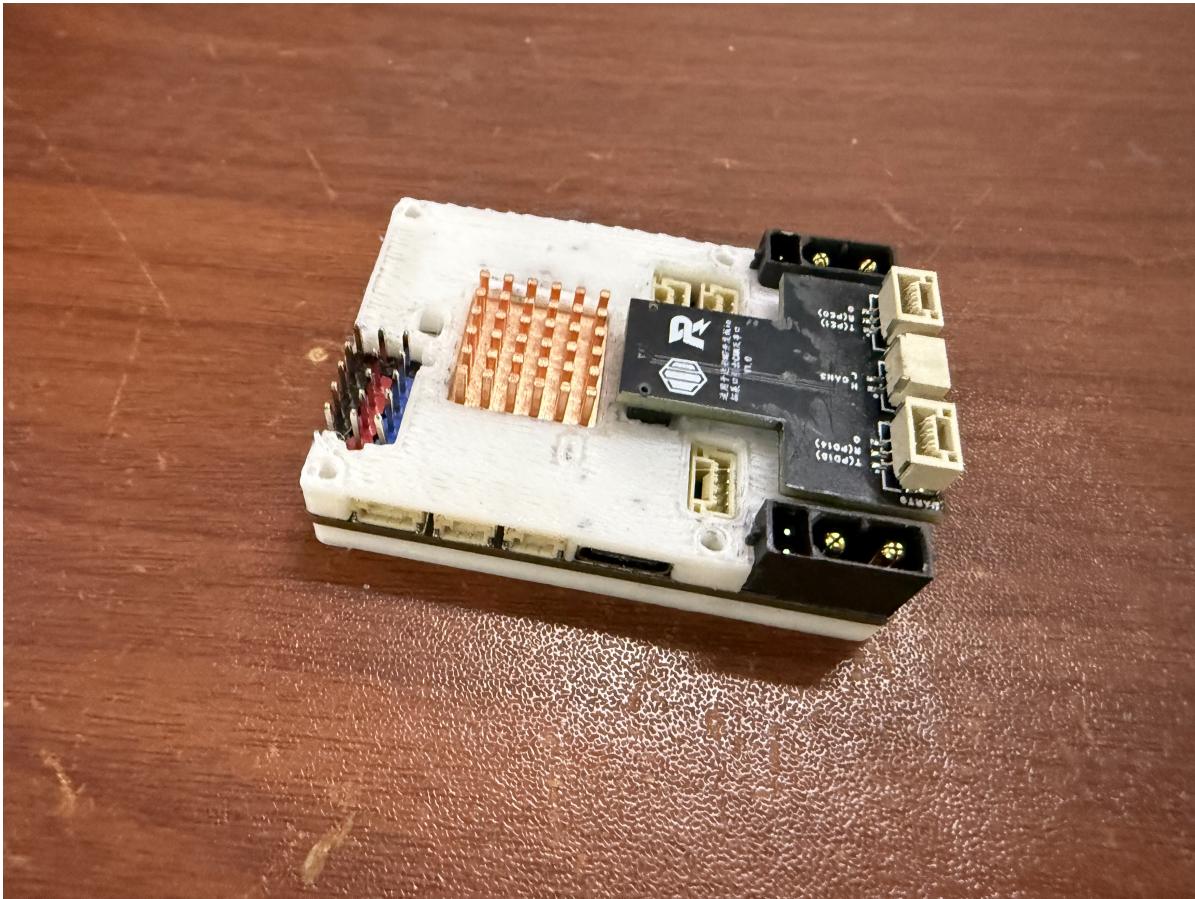
[STM32H7系列教程2 使用DMA双缓冲区接收大疆DT7遥控器数据\(SBUS/DBUS协议\) - 知乎](#)

[STM32H7系列教程3 SRAM区详解和DMA1、2无法访问TCM的解决方法 - 知乎](#)

STM32H7系列教程4 FDCAN的使用 (待更新)

STM32H7系列教程5 Cache和MPU (待更新)

开发板外壳可参考西南石油大学铁人战队开源[达妙DM-MC02 H7开发板TPU保护壳开源 - RoboMaster 社区](#)



## 2 目录结构

---

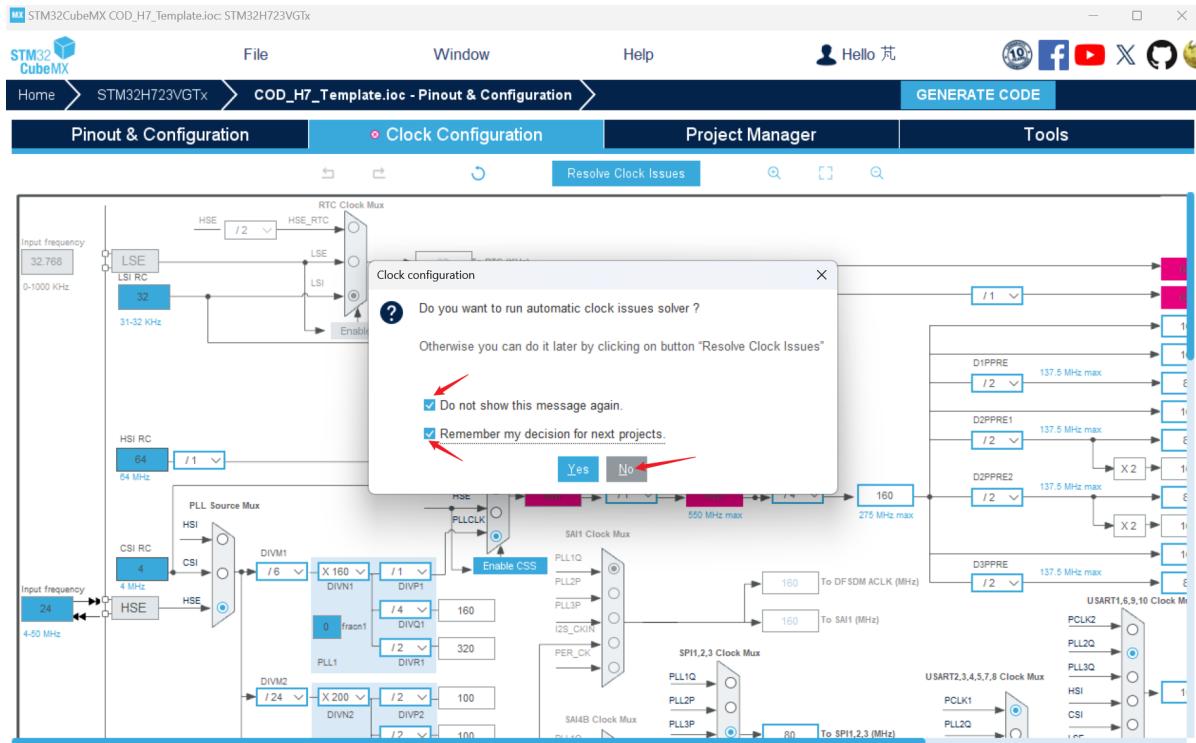
COD-H7-Template

- |——Application/Task
- |——Bsp
- |——Components/Algorithm
- |——Components/Controller
- |——Components/Device
- |——Core
- |——Drivers
- |——MDK-ARM
- |——Middlewares
- |——USB\_DEVICE
- |——Picture
- |——Document

# 3 功能模块说明

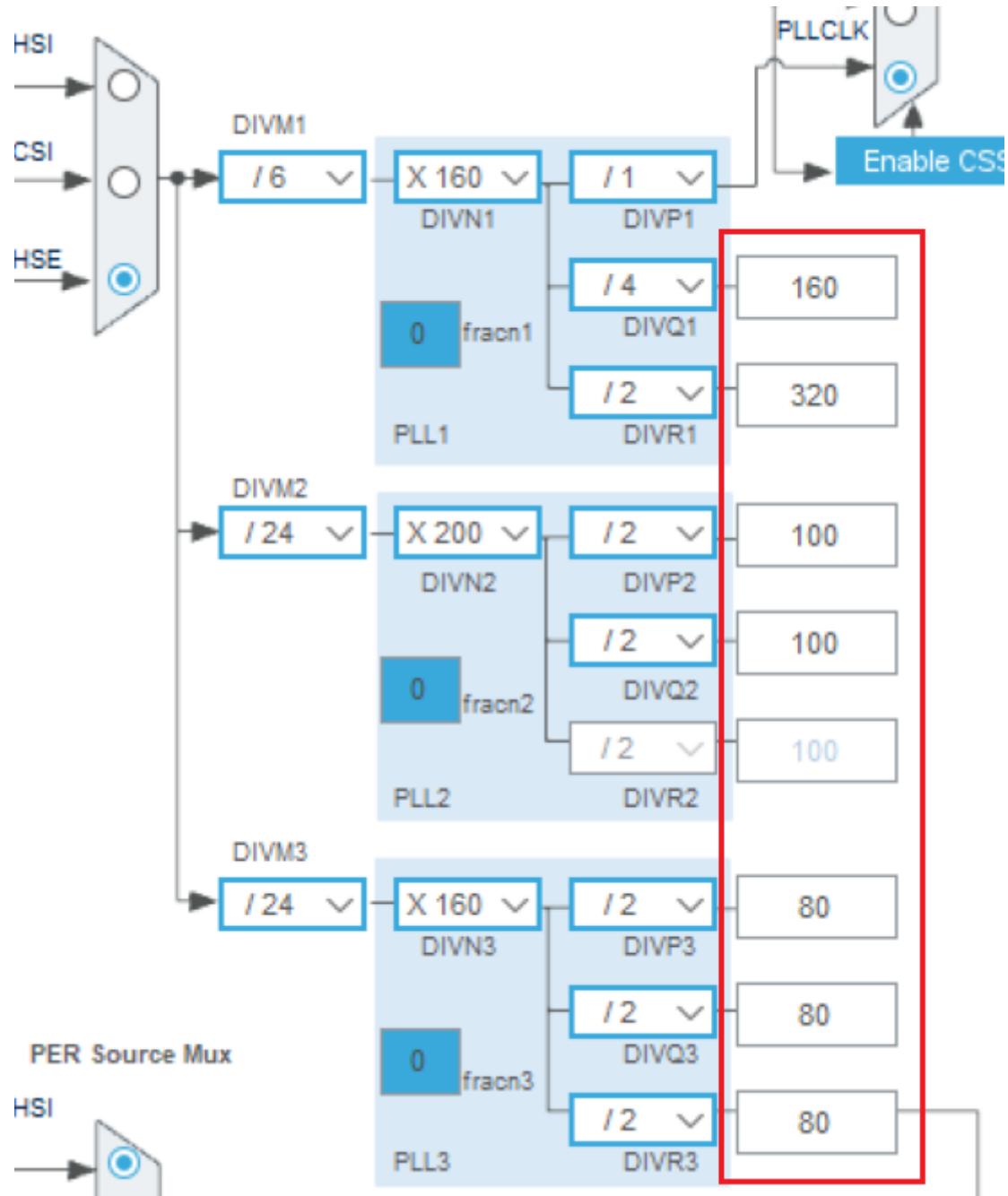
## 3.1 时钟

本模板将STM32H7VGT6主频设置为640Mhz（超频），大于理论最大值550Mhz，在半年的使用中未出现问题。在STM32CubeMX中时钟树中会出现超频的提示



可勾选不要提示此信息和记住我的选择在下一个文档（其实记不住 每次打开需要重新遇到这个提示）点击NO

若需要修改时钟 修改后请保证PLL1、2、3时钟与本模模板保持一致 否则一些设计时钟计算的外设（FDCAN、PWM、ADC等）的参数会发生改变 导致无法正常工作。



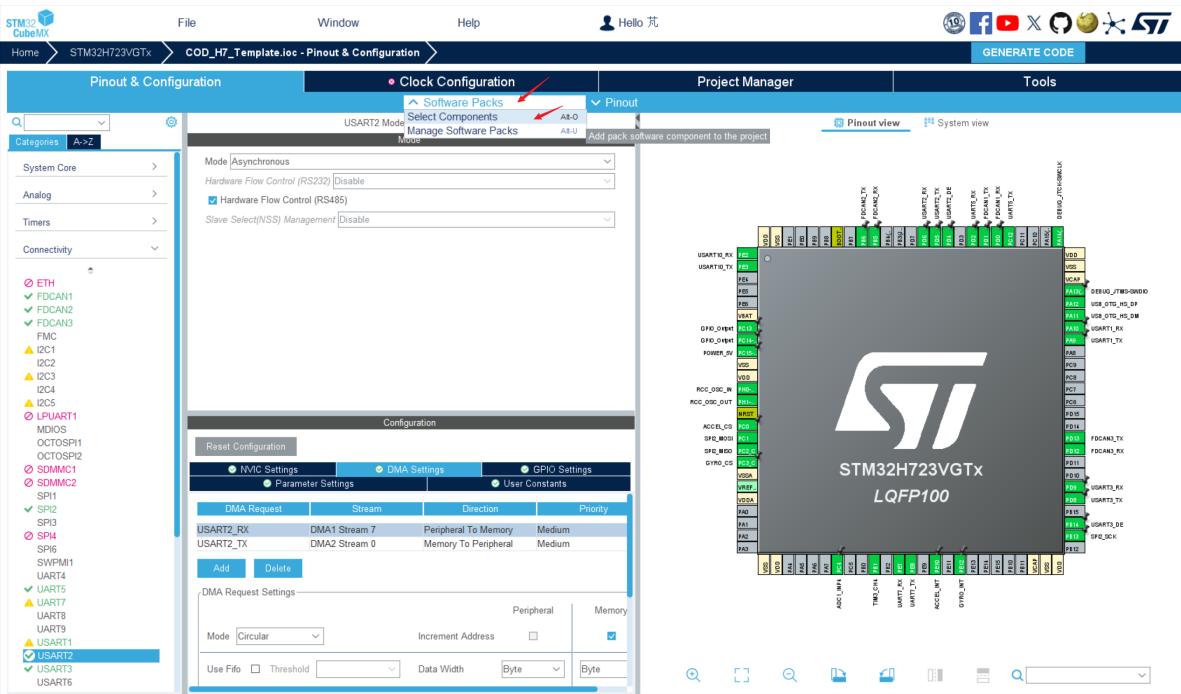
## 3.2 IMU惯性测量单元

模块参考[哈尔滨工程大学创梦之翼战队惯导姿态解算项目](#)

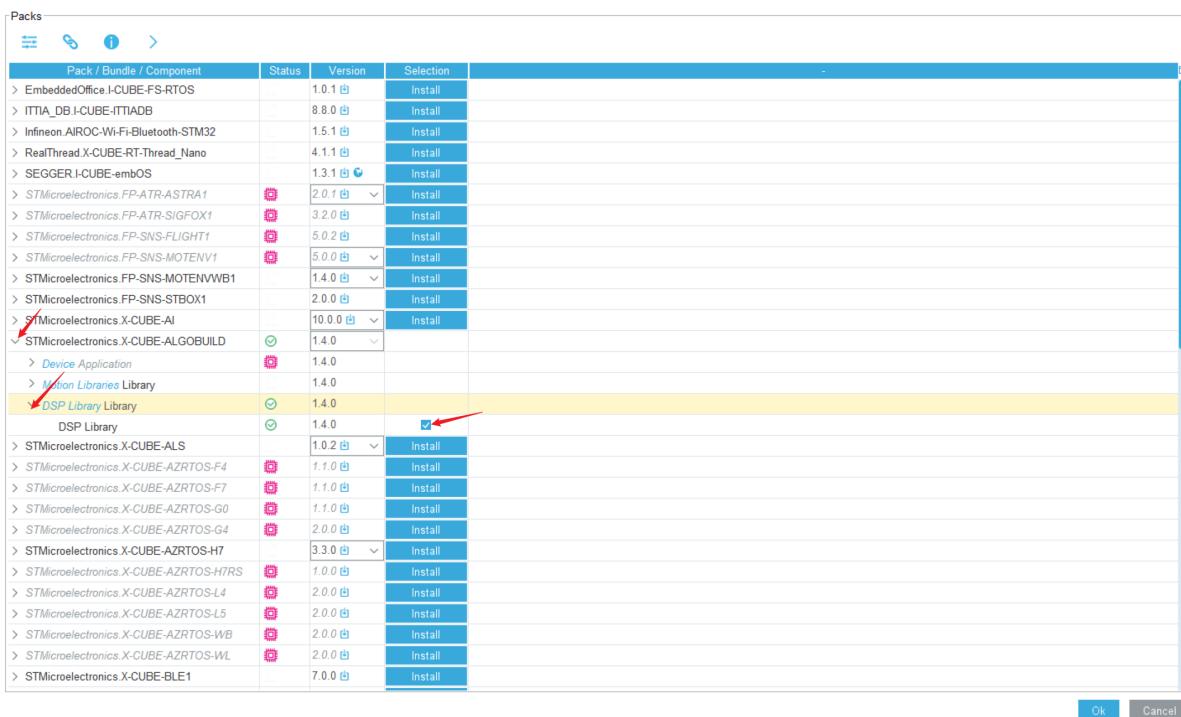
详情见[Quaternion](#)

### 3.2.1 STM32CubeMX添加DSP库

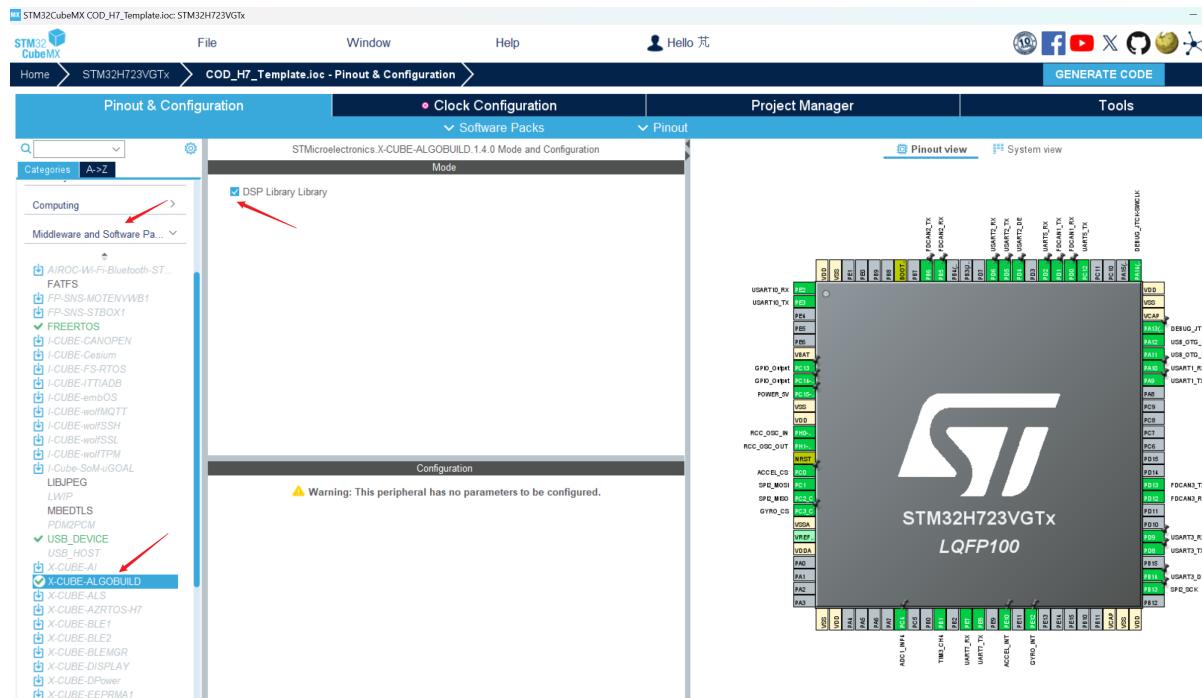
点击[Software Packs]/[Select Components]



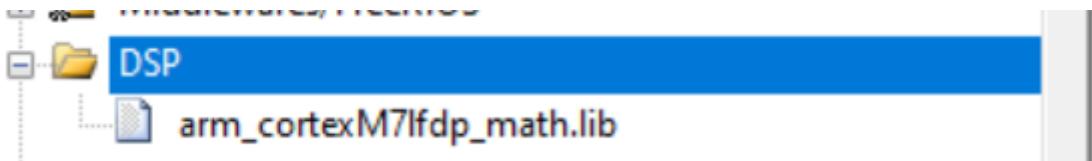
在弹出的[Software Packs Component Selector]窗口中，勾选[STMicroelectronics.X-CUBE-ALGOBUILD]/[DSP Library Library]/[DSP Library 1.4.0];



关闭[Software Packs Component Selector]窗口，在[Middle and Software Packs]/[X-CUBE-ALGOBUILD]栏勾选[DSP Library Library]

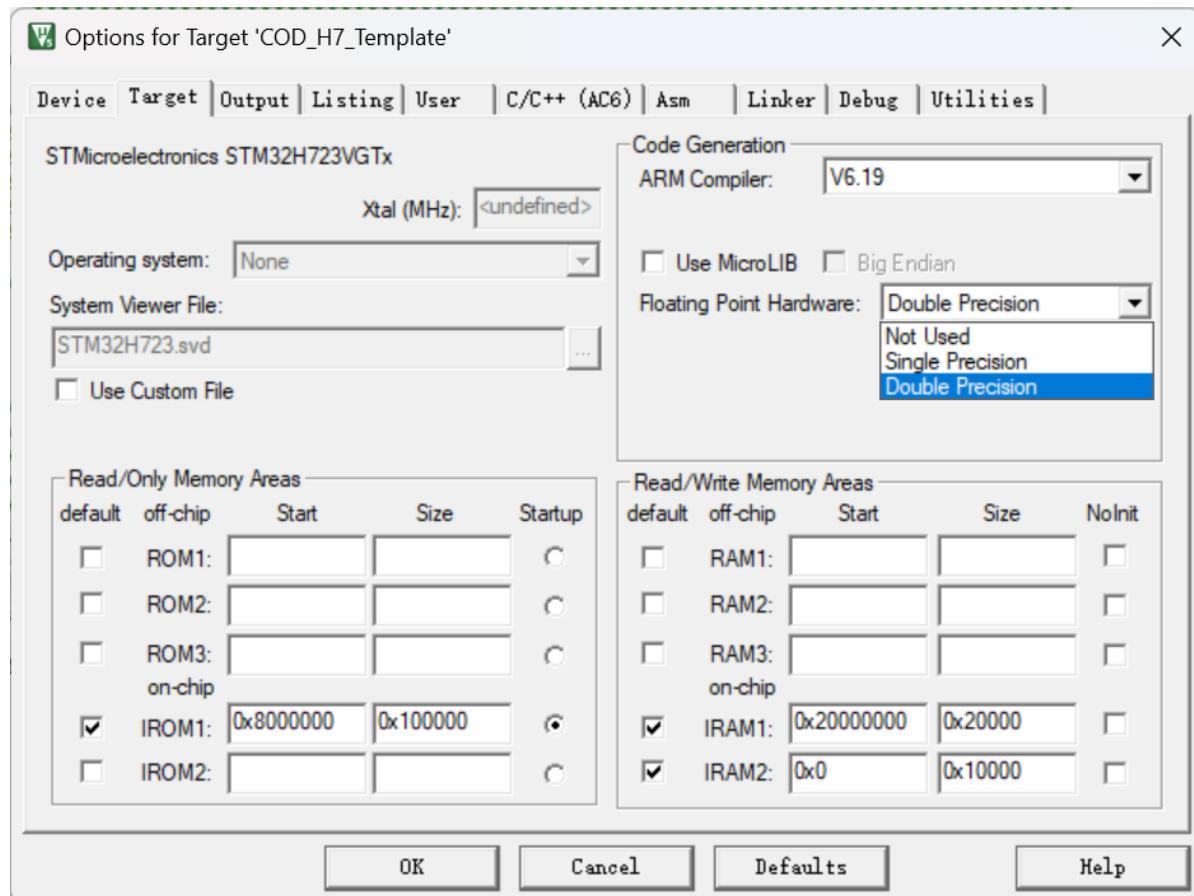


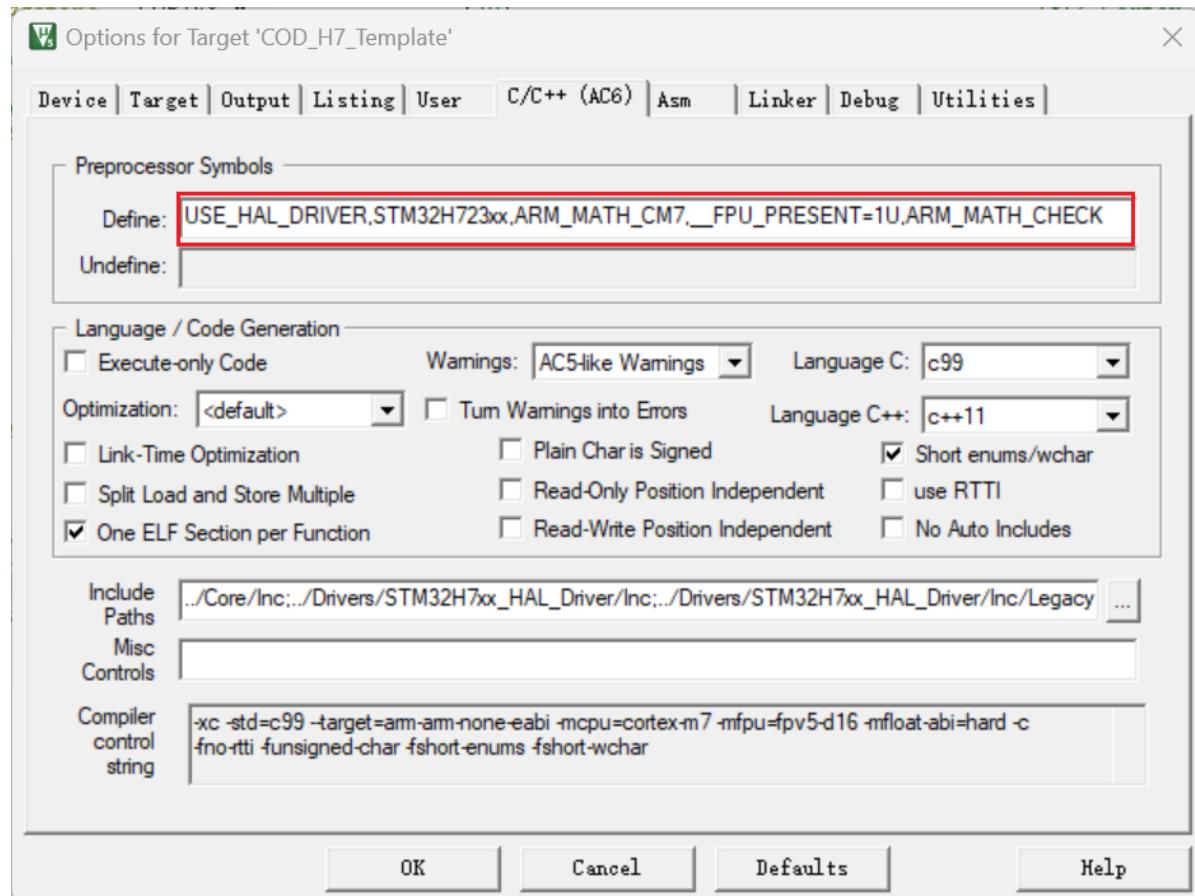
此时在工程中默认添加的LIB文件为arm\_cortexM7lfdp\_math.lib



STM32H7支持双精度浮点加速运算，请在Keil魔术棒中打开并添加宏定义

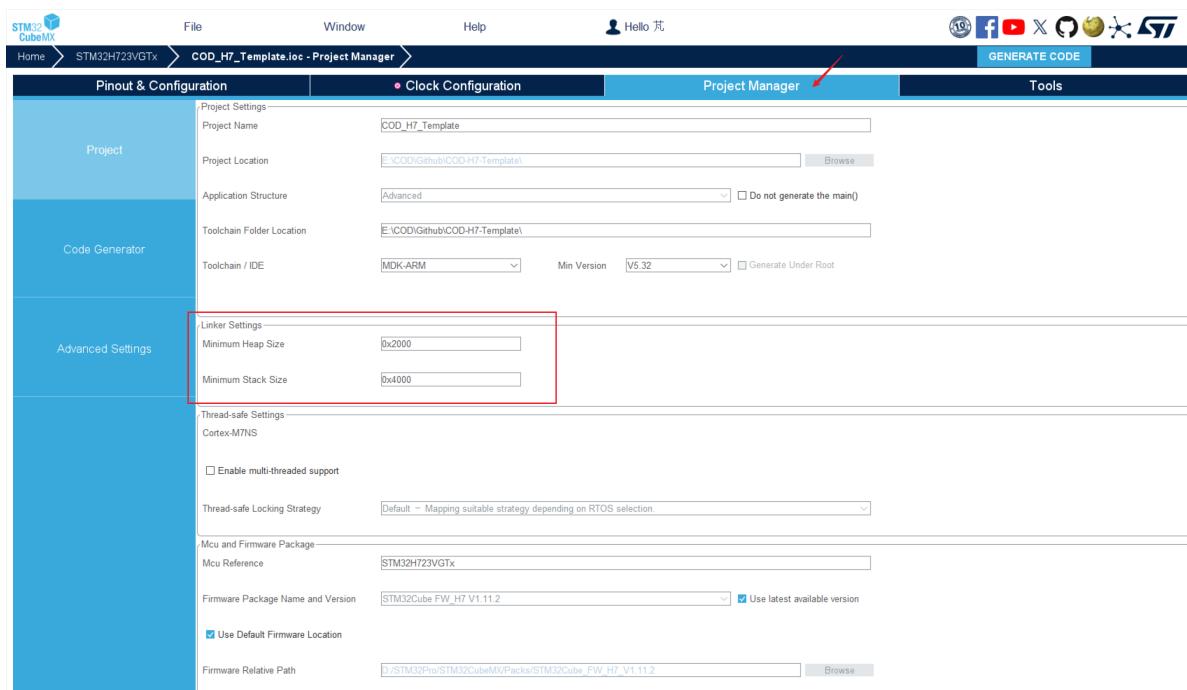
**ARM\_MATH\_CM7,\_FPU\_PRESENT=1U,ARM\_MATH\_CHECK**





### 3.2.2 mallo函数内存申请失败

STM32CubeMX默认生成startup\_stm32h723xx.s中分配的堆空间Heap\_Size只有0x0200个字节，而在初始化扩展卡尔曼时所申请的空间超过了0x0200，需要在STM32CubeMX中的[Project Manager]/[Project]/[Linker Settings]栏修改Minimum Heap Size的值以达到使用需求，修改后可在startup\_stm32h723xx.s文件中的Heap\_Size体现



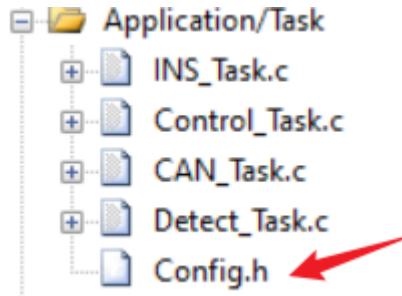
### 3.2.3 陀螺仪零飘校准（重要）

IMU数据在静止状态下不收敛或机器人纬度变化时（去比赛）请进行校准

校准步骤：

1 请自行查阅当前地区的重力加速度值 如鞍山通常为9.794 g/m^2

将Task目录下的config.h中的GravityAccel修改为当前地区重力加速度



```
28  /**  
29   * @brief the value of local gravity acceleration  
30   */  
31 #define GravityAccel 9.794f
```

2 打开Components/Device目录下的bmi088.c

找到BMI088\_Offset\_Update函数 将其上方IMU\_Calibration\_ENABLE 修改为1 (1表示开启校准 0表示不校准)

```
217  
218 #define IMU_Calibration_ENABLE 1U  
219  
220 static void BMI088_Offset_Update(BMI088_Info_TypeDef *BMI088_Info)  
221 {  
222 #if IMU_Calibration_ENABLE /* ENABLE the BMI088 Calibration */  
223     uint8_t buf[8] = {0,};  
224  
225     for(uint16_t i = 0; i < 5000; i++)  
226     {  
227         /* read the accelerator multi data */  
228         BMI088_Accel_Read_Multi_Reg(BMI088_ACCEL_XOUT_L, buf, 6);  
229         BMI088_Info->MPU_Info.Accel_X = (int16_t)((buf[1]) << 8) | buf[0];  
230         BMI088_Info->MPU_Info.Accel_Y = (int16_t)((buf[3]) << 8) | buf[2];  
231         BMI088_Info->MPU_Info.Accel_Z = (int16_t)((buf[5]) << 8) | buf[4];  
232  
233         /* read the gyro multi data */  
234         BMI088_Gyro_Read_Multi_Reg(BMI088_GYRO_CHIP_ID, buf, 8);  
235         /* check the ID */  
236         if(buf[0] == BMI088_GYRO_CHIP_ID_VALUE)  
237         {  
238             BMI088_Info->MPU_Info.Gyro_X = (int16_t)((buf[3]) << 8) | buf[2];  
239             BMI088_Info->MPU_Info.Gyro_Y = (int16_t)((buf[5]) << 8) | buf[4];  
240             BMI088_Info->MPU_Info.Gyro_Z = (int16_t)((buf[7]) << 8) | buf[6];  
241  
242             /* update the gyro offsets */  
243             BMI088_Info->Offsets_Gyro_X += BMI088_GYRO_SEN * BMI088_Info->MPU_Info.Gyro_X;  
244             BMI088_Info->Offsets_Gyro_Y += BMI088_GYRO_SEN * BMI088_Info->MPU_Info.Gyro_Y;  
245             BMI088_Info->Offsets_Gyro_Z += BMI088_GYRO_SEN * BMI088_Info->MPU_Info.Gyro_Z;  
246         }  
247         /* waiting lms */  
248         Delay_ms(1);  
249     }  
250 }
```

3 IMU\_Calibration\_ENABLE 修改为1之后 灰色部分的代码会变为黑色

进入debug 将BMI088\_Info放入watch窗口

BMI088_Info	0x20000458 &BMI088_...	struct <untagged>
Offsets_Init	0x00	char
Accel	0x2000045C	float[3]
Gyro	0x20000468	float[3]
[0]	0	float
[1]	0	float
[2]	0	float
Temperature	0	float
MPU_Info	0x20000478	struct <untagged>
Offsets_Gyro_X	0	float
Offsets_Gyro_Y	0	float
Offsets_Gyro_Z	0	float
<Enter expression>		

运行代码前 请保持陀螺仪静止且水平！！！

运行代码，等待温度Temperature上升至40度，温度保持至40度左右时，Reset，重新运行代码

BMI088_Info	0x20000458 &BMI088_...	struct <untagged>
Offsets_Init	0x00	char
Accel	0x2000045C	float[3]
Gyro	0x20000468	float[3]
[0]	0	float
[1]	0	float
[2]	0	float
Temperature	0 40	float
MPU_Info	0x20000478	struct <untagged>
Offsets_Gyro_X	0	float
Offsets_Gyro_Y	0	float
Offsets_Gyro_Z	0	float
<Enter expression>		

重新运行代码 Offsets\_Gyro\_X、Offsets\_Gyro\_Y、Offsets\_Gyro\_Z会不断累加，最后收敛为一个定值（Offsets\_Init变为0x01）（期间保持陀螺仪静止且水平！！！）

BMI088_Info	0x20000458 &BMI088_...	struct <untagged>
Offsets_Init	0x00 0x01	char
Accel	0x2000045C	float[3]
Gyro	0x20000468	float[3]
[0]	0	float
[1]	0	float
[2]	0	float
Temperature	0	float
MPU_Info	0x20000478	struct <untagged>
Offsets_Gyro_X	0 value1	float
Offsets_Gyro_Y	0 value2	float
Offsets_Gyro_Z	0 value3	float
<Enter expression>		

将得到的 `Offsets_Gyro_X`、`Offsets_Gyro_Y`、`Offsets_Gyro_Z` 的值填入 `BMI088_Offset_Update` 函数 `#else` 下对应的变量中

```
#else /* DISABLE the BMI088 Calibration */
    /* store the previous offsets */
    BMI088_Info->Offsets_Gyro_X =value1;
    BMI088_Info->Offsets_Gyro_Y =value2;
    BMI088_Info->Offsets_Gyro_Z =value3;
#endif
```

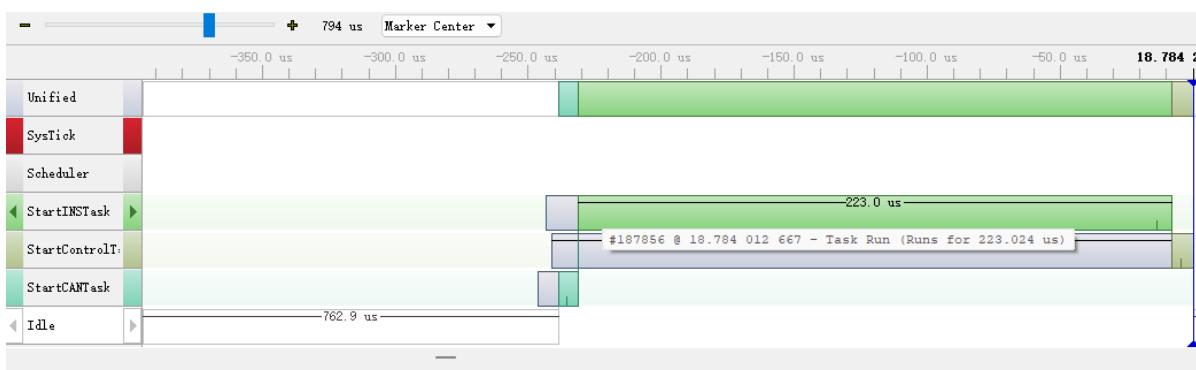
最后将 `IMU_Calibration_ENABLE` 修改为 0 (关闭校准) 完成校准

```
#define IMU_Calibration_ENABLE 0U
```

### 3.2.4 陀螺仪解算速度

使用 SEGGER SystemView 观测陀螺仪解算任务 `INS_Task`

RoboMaster C 型开发板板 STM32F407IGH6 主频 168MHz 解算时间 220-230us



达妙 MC-02 STM32H723VGT6 主频超频至 640MHz 开启 Cache 加速和将数据存放进 DTCM 解算时间 93-95us



### 3.3 MiniPC通信

使用Type-C连接STM32和上位机

在./Device/Src/MiniPC中封装了发送和接收函数

其中接收函数

```
void MiniPC_Recvive_Info(uint8_t* Buff, const uint32_t *Len)
```

在Application/User/USB\_DEVICE/App/usbd\_cdc\_if.c 的函数中调用实现接收数据

```
static int8_t CDC_Receive_HS(uint8_t* Buf, uint32_t *Len){
    MiniPC_Recvive_Info(Buf, Len)
    .....
}
```

### 3.4 裁判系统

本模板封装了最新的裁判系统协议[RoboMaster 裁判系统串口协议附录 V1.7.0 \(20241225\) .pdf](#)

使用USART1的DMA双缓冲接收 因暂未测试 可能存在BUG 回学校后第一时间测试。

本模板只实现了裁判系统数据的接收

UI绘制部分可参考南京航空航天大学开源可视化UI设计器 [【RM2024赛季-UI设计器开源】南京航空航天大学-RoboMaster 社区](#) 后续也会将UI更新至模板中

### 3.5 RS485

两路RS485为USART2和USART3 使用方法和普通串口一致 如接收可使用双缓冲接收，发送使用HAL库函数

### 3.6 FDCAN

FDCAN1、FDCAN3设置为了经典CAN 波特率为1M

FDCAN2设置为了FDCAN 仲裁域波特率为1M 数据域波特率为5M

若不需要使用FDCAN 可将FDCAN2中的FrameMat修改为Classic mod Data Prescaler修改为5即可 并在bsp\_can.c中的BSP\_CAN\_Init()中关闭发送延迟补偿

## Configuration

[Reset Configuration](#)

[Parameter Settings](#) [User Constants](#) [NVIC Settings](#) [GPIO Settings](#)

Configure the below parameters :

Search (Ctrl+F)		↶	↷	i
<b>Basic Parameters</b>				
Frame Format	FD mode with BitRate Switching			↶
Mode	Classic mode			↷
Auto Retransmission	FD mode without BitRate Switching			
Transmit Pause	FD mode with BitRate Switching			
Protocol Exception	Enable			
Nominal Sync Jump Width	5			
Data Prescaler	1 5			
Data Sync Jump Width	5			
Data Time Seg1	14			
Data Time Seg2	5			
Message Ram Offset	853			
Std Filters Nbr	1			
Ext Filters Nbr	0			
Rx Fifo0 Elmts Nbr	0			
Rx Fifo0 Elmt Size	8 bytes data field			
Rx Fifo1 Elmts Nbr	8			
Rx Fifo1 Elmt Size	8 bytes data field			
Rx Buffers Nbr	0			
Rx Buffer Size	8 bytes data field			
Tx Events Nbr	0			
Tx Buffers Nbr	0			
Tx Fifo0 Elmt Nbr	0			

```

FDCAN_FilterTypeDef FDCAN2_FilterConfig;
FDCAN2_FilterConfig.IdType = FDCAN_STANDARD_ID;
FDCAN2_FilterConfig.FilterIndex = 0;
FDCAN2_FilterConfig.FilterType = FDCAN_FILTER_MASK;
FDCAN2_FilterConfig.FilterConfig = FDCAN_FILTER_TO_RXFIFO1;
FDCAN2_FilterConfig.FilterID1 = 0x00000000;
FDCAN2_FilterConfig.FilterID2 = 0x00000000;
HAL_FDCAN_ConfigFilter(&hfcan2, &FDCAN2_FilterConfig);
HAL_FDCAN_ConfigGlobalFilter(&hfcan2, FDCAN_REJECT, FDCAN_REJECT, FDCAN_FILTER_REMOTE, FDCAN_FILTER_RI
HAL_FDCAN_ActivateNotification(&hfcan2, FDCAN_IT_RX_FIFO1_NEW_MESSAGE, 0);
HAL_FDCAN_EnableTxDelayCompensation(&hfcan2); //开启FDCAN的发送延迟补偿
HAL_FDCAN_ConfigTxDelayCompensation(&hfcan2, 14, 14); //设置补偿时间 参数2和参数3都为TimeSeg1的值
HAL_FDCAN_Start(&hfcan2);                                注释

```

将FDCAN1和FDCAN3修改为FDCAN模式 反之操作即可

## 3.7 电机

本模板封装了两类电机 DJI系列电机和达妙系列电机 可参考CAN\_Task和bsp\_can.c进行发送和接收

经典CAN1000hz一路CAN最多只可带三个达妙电机（一发一收模式）且保证发送顺序与发送ID保持一致和接收ID大于发送ID

如电机1：发送0x01 接收0x11电机2：0x02 0x12电机3：0x03 0x13

则

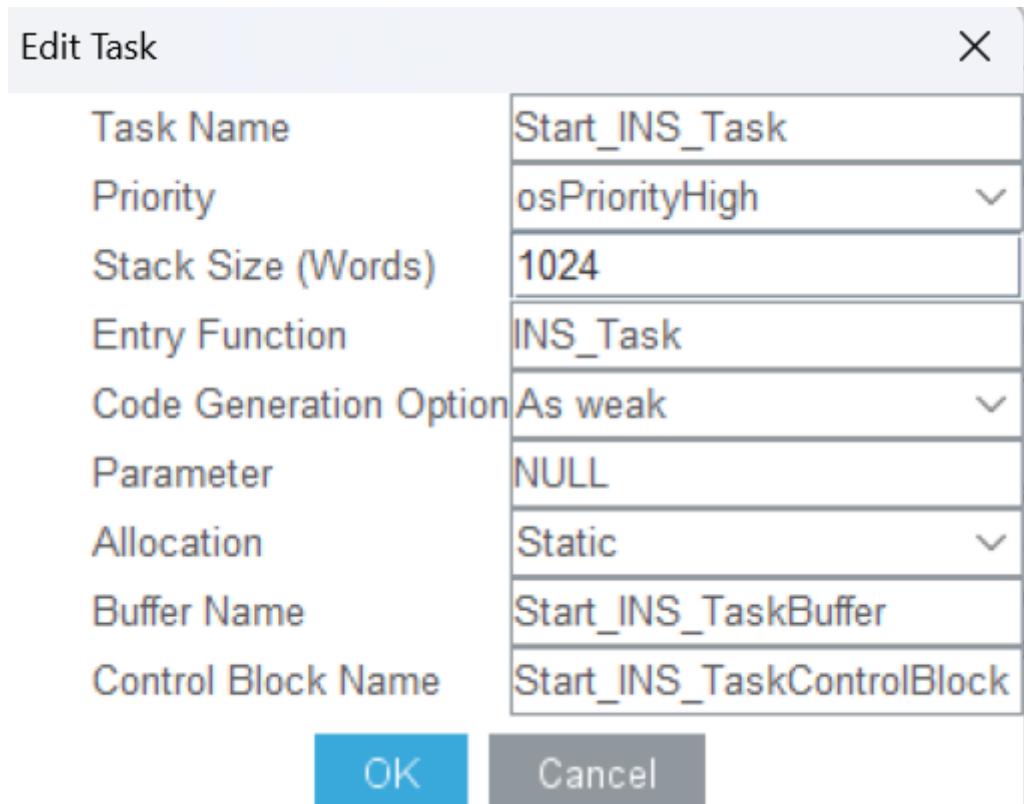
```
DM_Motor_CAN_TxMessage(&FDCAN2_TxFrame,&DM_8009_Motor[0],0,0,0,0,0);
DM_Motor_CAN_TxMessage(&FDCAN2_TxFrame,&DM_8009_Motor[1],0,0,0,0,0);
DM_Motor_CAN_TxMessage(&FDCAN2_TxFrame,&DM_8009_Motor[2],0,0,0,0,0);
osDelay(1)
```

这样可以保证三个电机的控制和数据都能接收到

CAN-FD作者一路1000hz最多带过6个达妙电机 理论上可带9个

### 3.8 FreeRTOS

本模板使用RTOS任务格式如下



优先级根据实现应用保持一致 Allocation使用静态static

Config parameters		Include parameters		Advanced settings				
Tasks								
Task Na...	Priority	Stac...	Entry Fu...	...	Parameter	Allocation	Buffer Na...	Control B...
Start_IN...	osPriorityHigh	1024	INS_Task	...	NULL	Static	Start_IN...	Start_IN...
Start_Co...	osPriorityAboveNormal	1024	Control_...	...	NULL	Static	Start_Co...	Start_Co...
Start_C...	osPriorityNormal	1024	CAN_Task	...	NULL	Static	Start_CA...	Start_CA...
Start_De...	osPriorityBelowNormal	1024	Detect_T...	...	NULL	Static	Start_Det...	Start_De...

任务中请不要出现多余的osDelay或osDelayUntil 请保证所有任务以1000Hz运行

若需要对部分功能进行降频和延时 请调用该任务的累计数systick进行取余运算 如

```
//500Hz  
if(Systick % 2 == 0){  
  
}  
osdelay(1);
```

## 4 联系方式

若遇到问题和BUG 请及时与作者本人联系 欢迎提交Issues和Pull Requests帮助我们改进

王草凡 GrassFan Wang

QQ：1985483641

邮箱：[1985483641@qq.com](mailto:1985483641@qq.com)

## 5 致谢

首先感谢我的学长，严远斌，杨涛，王锦璟，董文硕、鲍天龙。是他们留下来的C板通用控制系统让我能在此基础上修改完善。尤其是严远斌学长，我许多的代码都是对他的拙劣模仿。

感谢深圳市达妙科技有限公司，感谢喵总、刘工、苏工、徐工、周姐等在暑假实习时的帮助和支持，那是一段令我怀念和难忘的时光。

感谢我的队友涂仁杰、姜春洋。感谢涂仁杰同学为开发板设计扩展板，感谢姜春洋同学为H7版本的轮腿设计机械结构。

感谢我的学弟们，你们是此项目的一个用户。

感谢西南石油大学铁人战队、哈尔滨工程大学创梦之翼战队、东北大学T-DT战队、南京航空航天大学长空御风战队、大连理工大学凌Bug战队（排名不分先后）的开源或交流，使我可以更好的完善此项目。

最后祝愿所有队伍在即将到来的联盟赛取得好成绩

承蒙关照

写于2025年2月9日 江苏 无锡 清华大学无锡智能  
创业创新中心