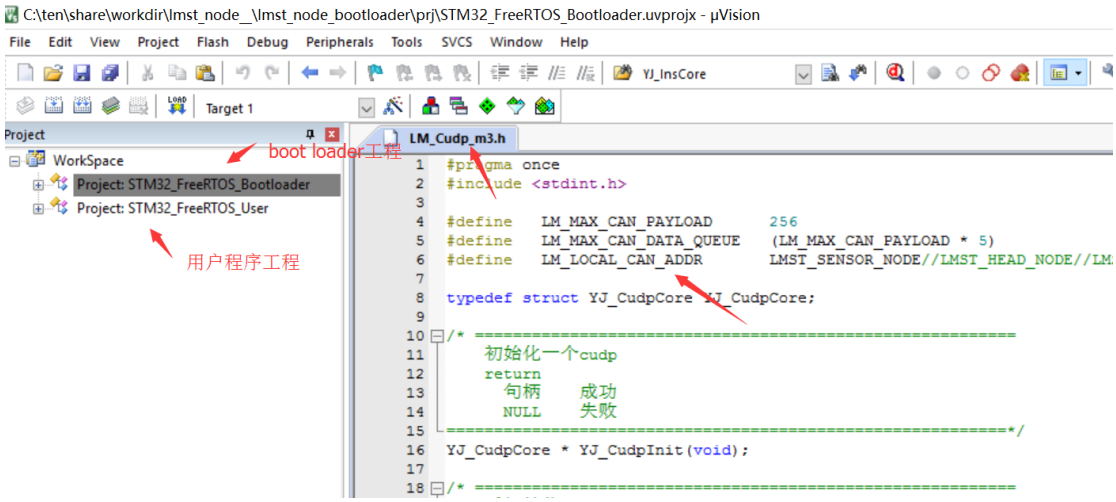
SmartTuna stm32节点程序开发指南

一、程序烧写

确保电脑已经正确安装好keil uVison5 和jlink驱动； 用jlink连接电脑和单关节stm32核心板；找到开发源码文件夹，并进入文件夹workspace，打开keil工作空间LMST_Node.uvmpw；空间已经包含了bootloader和用户程序两个工程，工程通过宏LM_LOCAL_CAN_ADDR来定义不同节点（bootloader和用户程序共享同一个宏），修改这个宏就可以选择对应节点的程序进行编译：

LMST_SENSOR_NODE	传感器舱
LMST_HEAD_NODE	头舱
LMST_SERVO_MOTOR_NODE	螺旋桨推进舱
LMST_DIVING_NODE	浮力舱
LMST_TAIL_NODE	摆动推进舱
LMST_HEAD_LOCAL_CTRL_NODE	头舱自动控制demo



程序下载步骤请观看视频《stm32程序烧写视频教程.wmv》；

PS：用户程序可以通过PC控制平台在线下载，具体请参考《SmartTuna PC控制平台使用说明书v1.0.pdf》

二、程序开发

工程已经提供了一个通过头舱节点对SmartTuna进行自动控制的demo，把宏LM_LOCAL_CAN_ADDR定义为LMST_HEAD_LOCAL_CTRL_NODE 然后编译用户程序工程即可；接口部分代码放在文件LMST_HeadNode_LocalCtrl.c中；

```
void YJ_HeadNodeTask_LocalCtrl( void *parameters )
{
        int32_t         ret = 1;
        char        * tmp = pvPortMalloc(LM_MAX_CAN_PAYLOAD);
```

```
int32_t       SrcAddr;
  YJ_MpuParam p;
  const portTickType xDelay = pdMS_TO_TICKS(3);
  const portTickType xDelay_5s = pdMS_TO_TICKS(5000);
  const portTickType xDelay_1s = pdMS_TO_TICKS(1000);
  mDEBUG("boot ----> user");
  while(1)
  {
      //尾舱控制
      TailCtrl(15, 7);
      vTaskDelay( xDelay_5s );
      TailCtrl(0, 7);
      vTaskDelay( xDelay_1s );

      //螺旋桨推进舱控制
      ServoMotorCtrl(8, 10, 8, 10);
      vTaskDelay( xDelay_5s );
      ServoMotorCtrl(7, 7, 7, 7);
      vTaskDelay( xDelay_1s );

      //mpu数据获取
      if(!GetMpuData(&p))
          continue;

      //打印调试信息到pc端
mDEBUG("mpu data -> Accel_X: %d Accel_Y: %d Accel_Z: %d Gyro_X: %d
Gyro_Y: %d Gyro_Z: %d Temp: %d", p.Accel_X, p.Accel_Y, p.Accel_Z,
p.Gyro_X, p.Gyro_Y, p.Gyro_Z, p.Temp);
  }
}
```