拆官方步兵电控框架、NUC（小电脑）通信检测

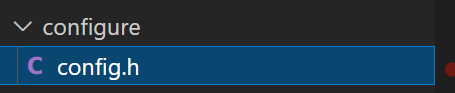
2022年2月3日 SAS电控组 陈卓勋

1. 背景

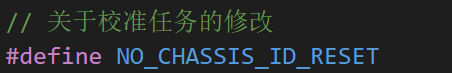
底层过于庞杂，且自己编写出了问题，把freertos弄崩溃了。刚刚成功将官方步兵的云台、底盘、设计控制程序全部删除了，且通过了编译，不影响其他OLED、usb、姿态解算掉线检测等功能的使用。因此希望迅速将“为比赛重新设计云台、底盘、射击”版本的程序（也就是第二版电控程序）分模块移植到此空白机器人程序中并测试，希望这样一点点验证的代码不会再出现崩溃的问题。

首先介绍一下新版本（第三版）的电控程序。这是从官方步兵上删除程序得到的。在删除的过程中，有许多有依赖关系的头文件和C源文件都需要修改。使用空白函数暂时替代删除的有依赖关系的函数，得到了我们的emptyRobot工程文件夹。

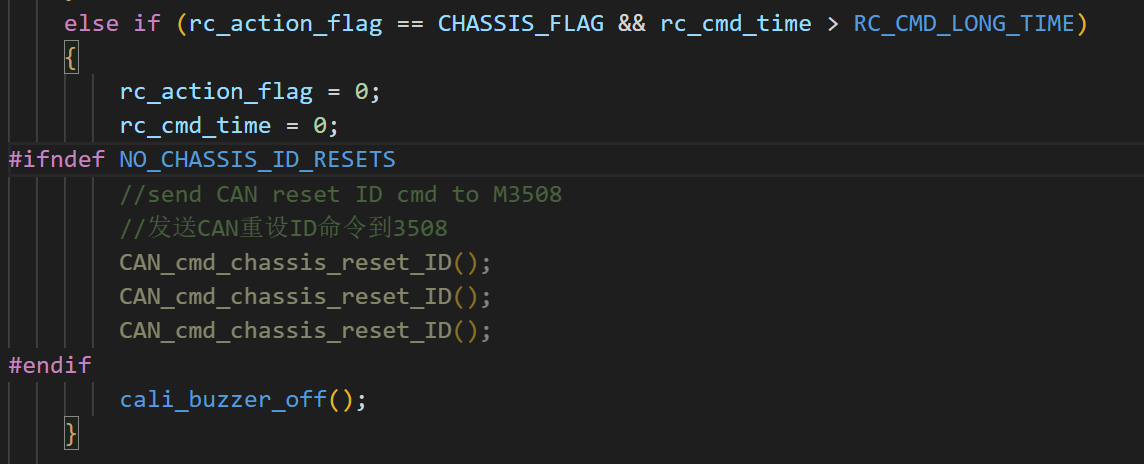
相比于原始的控制程序，增加了一个文件夹



保存一些配置。用宏定义的方法选择某些模式的开启与关闭。例如，不希望有遥控器重新设置电机底盘ID的情况出现，使用宏定义

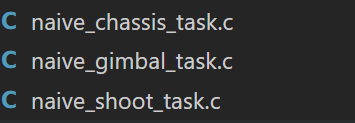


修改caliberate.c

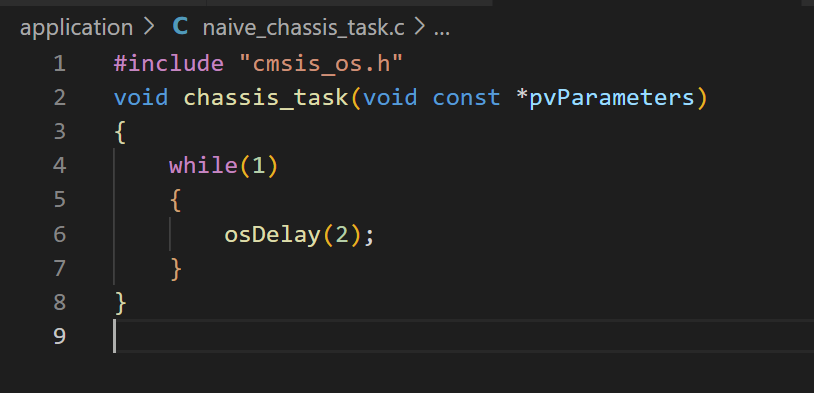


避免误操作。

删除了官方的云台、底盘控制代码。使用新的空白代码

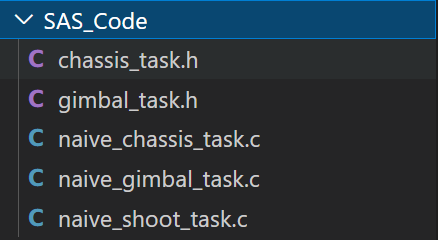


例如



我们可以在这里编写自己的控制程序。

自己编写的程序放在这里面吧。



另外，我们还有一些电机和官方电机不一样。因此，在CAN\_receive.c中也有许多需要修改的。可以直接使用我们之前第一个版本的canMotor源文件替换这个文件。那已经在机器人身上被证明成功了。

未来，我们需要一个模块一个模块地编写，编写完成后立即测试，全部功能正常后再去组装大的功能。

1. 移植机器人总模式控制的过程

曾经在第二版电控程序中，为了机器人模式控制的一致性，增加了一个总模式控制任务。可以在freertos中手动开启。

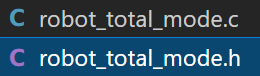
现在以此为例，展示如何编写和测试。

复制一个文件夹，重命名，加上日期

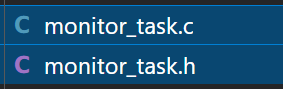


打开并编写程序。

打开我们第20版的程序，将其内总模式控制程序复制到空白机器人中。



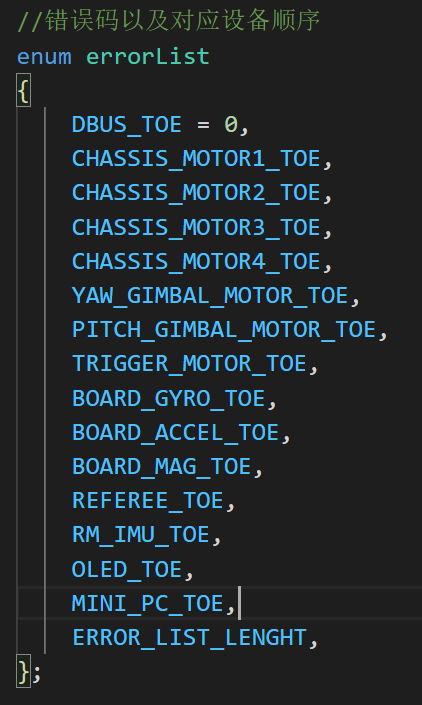
在复制之后，发现这其实需要之前的一个离线检测任务，那么，我们先移植离线检测任务。

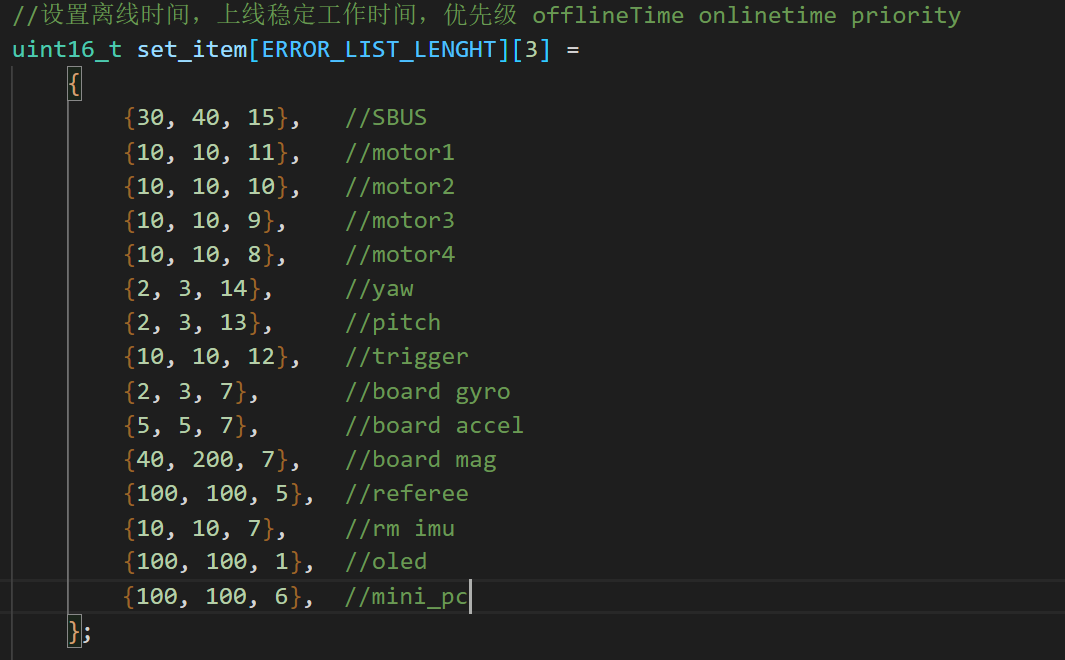


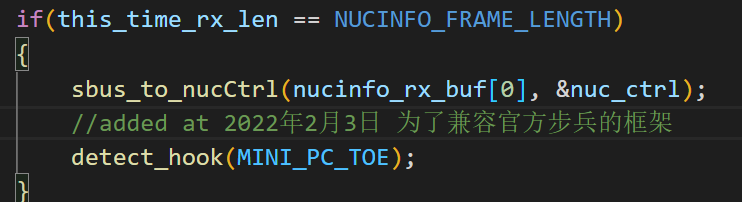
里面使用了修改后的遥控器数据解析函数，包含了系统时钟获取。因此需要修改官方的遥控器数据解析。

但是我们能否直接使用官方的框架，探测设备是否掉线？官方的遥控器掉线是怎么做的？那么，我们目前不妨实现一个NUC掉线检测吧。

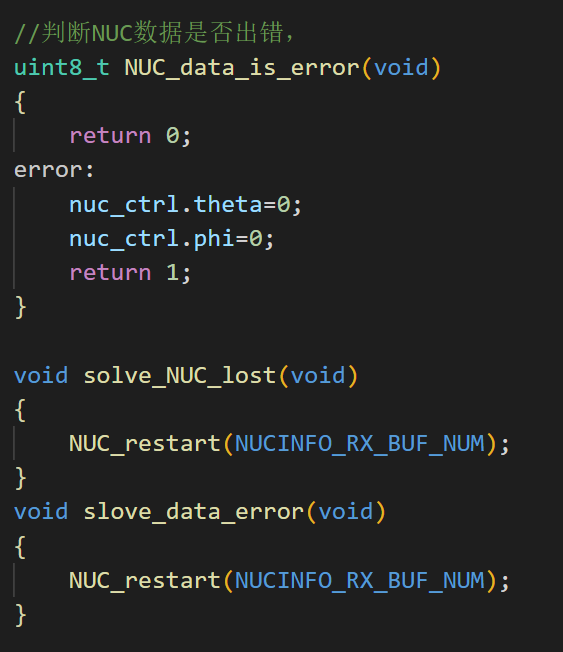
打开detect\_task.c按照前面的提示增加各个部分，仿照遥控器数据解析的代码，增加相关函数。







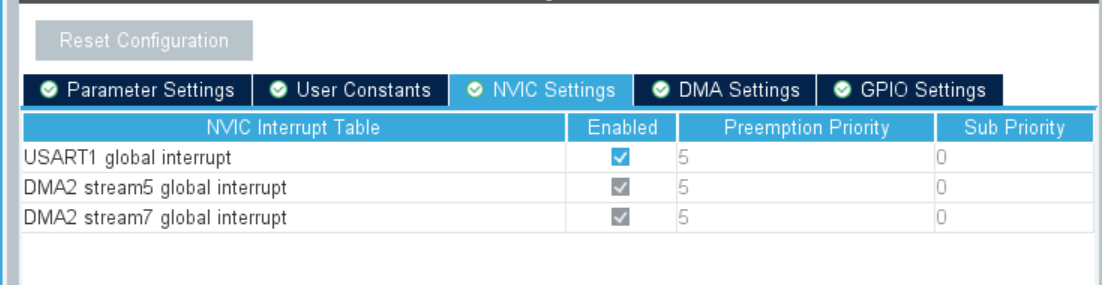
所以现在nuc解析代码支持两种离线检测方式。一种是开源的框架内的，一种是自己写的naïve的。



将所需文件添加进工程

测试NUC通信支持。

打开CubeMX，打开usart1 DMA传输

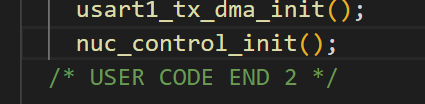


生成工程。

打开uvision并增加工程文件配制

增加OLED显示NUC上线状态

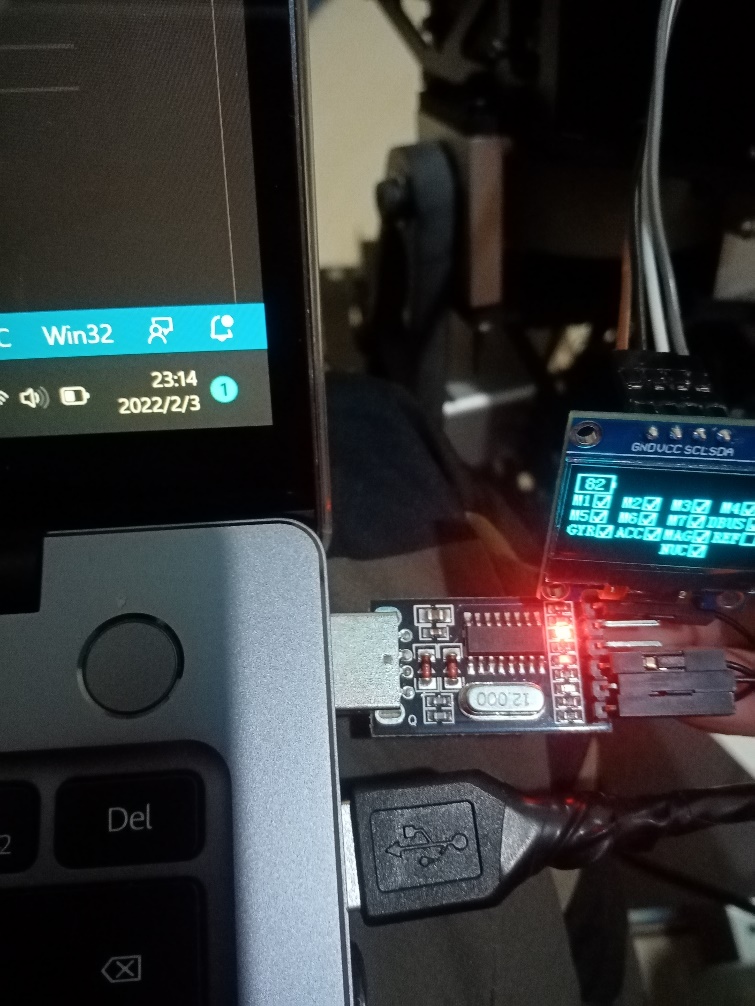
为了启动NUC串口数据接收，还需要在main函数里增加初始化代码。



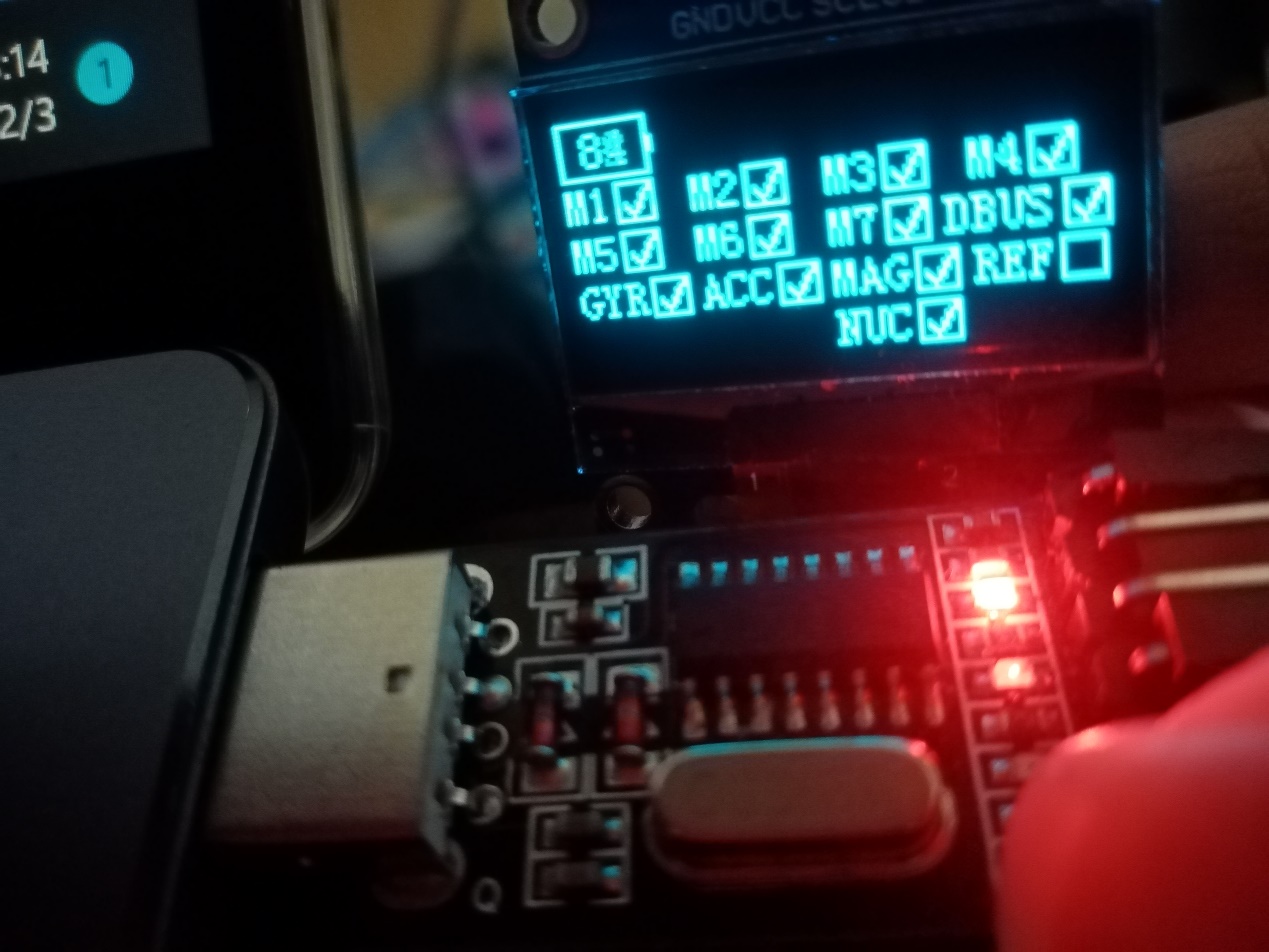
三、测试结果



每隔10ms发送一个28字节的数据帧。

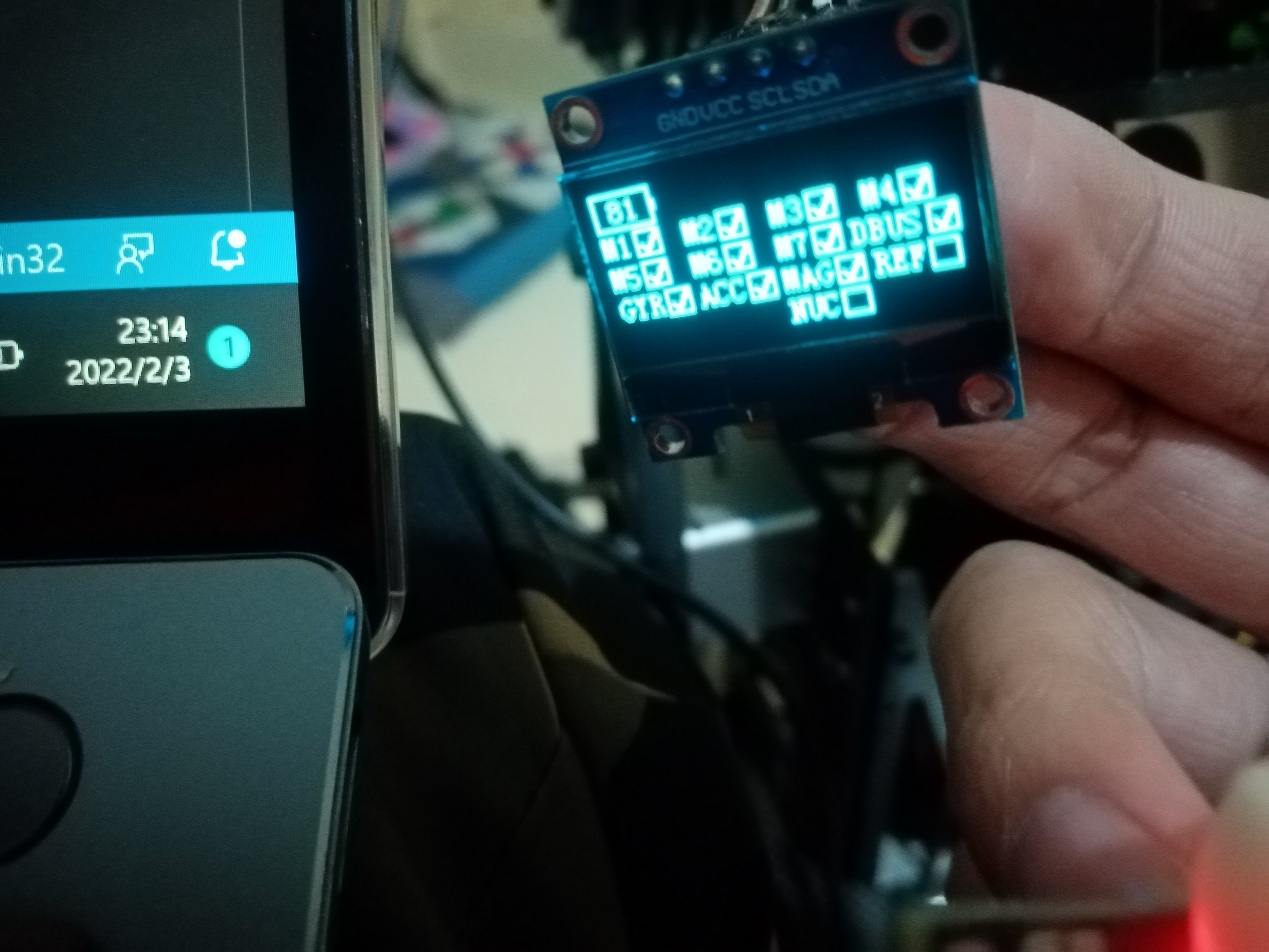


串口转USB模块的发送灯亮起，OLED显示屏上，最下面一排的NUC后面的框里有勾勾，表示NUC上线了。



特写

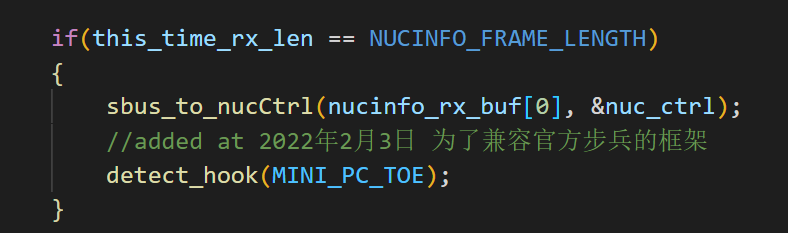
拔掉转换模块，单片机检测到NUC掉线了。



重新插上去，NUC再次上线。

将数据帧改为27个字节，提示NUC掉线。这是因为在数据转换钩子函数那里，是先检测了数据帧长度，长度正确才更新检测到的时间。





热插拔能够获得的现象很好。

结果可喜。算是一年多以来不断尝试，第一次成功吧。Orz

1. 总结

电控的东西真的太多了。第三个版本的电控代码统一使用v30.x作为编号。不再写v31是因为继续下去数字太大了。最基础的v30.是一个拆除云台、底盘、发射机构控制程序模块的“空机器人”。而v30.0是进行了初步移植实验的机器人。将之前两个版本里的NUC通信程序移植到了第3个版本里，增加了再开源框架里的掉线检测功能，利用OLED模块显示了NUC的上线与掉线，用27和28字节的数据帧实验，结果为能够识别正确长度的数据。好。