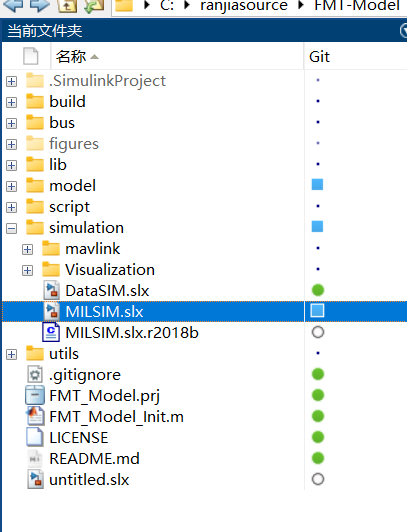
# FMT-model基础开发记录

## 板块介绍和核心开发区位

### 大板块介绍

进入FMT-Model中的simulation/MILSIM.slx



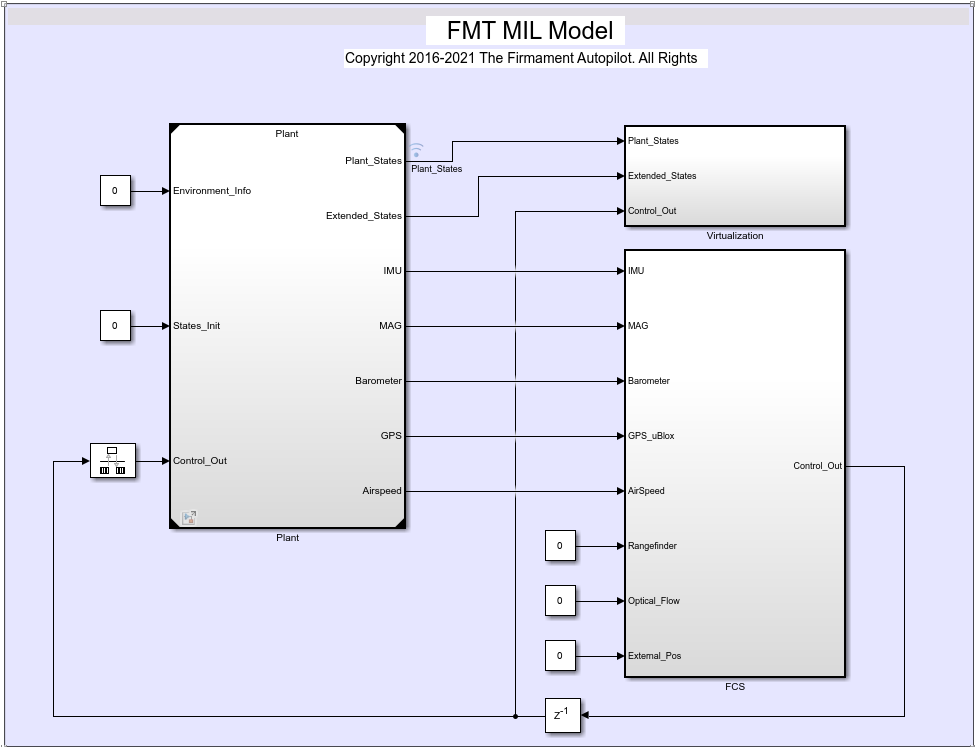
以下为FMT MIL Model的主体结构，其中

**1**.plant板块就相当于你的实体（船，无人机、小车），具体来说就是通过sumilink实现对你的实体进行构建，具体包括船体模型、环境状态、控制以及一些通过GPS等传感器对船的水平状态、位置、空速进行反映。

**2**.Virtualization板块是在matlab中做一些视觉仿真，目前不是重点

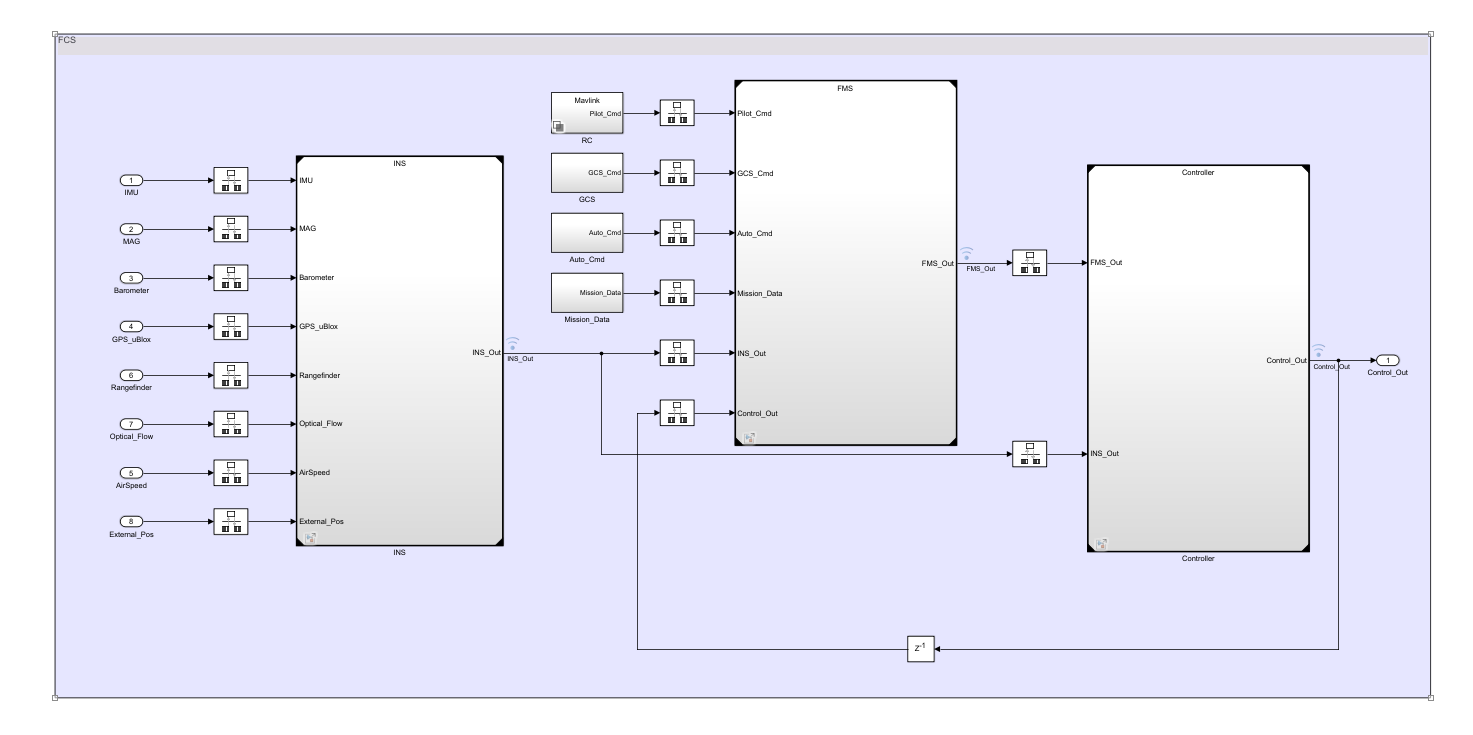
**3**.FCS分3大块：导航、飞控、控制器（其中，通过FMT-Faware烧录到飞控板上的就是FCS部分的代码）

关系：现实操控中就是plant与FCS进行不断的交互



因为FCS为核心开发区域，我们点击进入开始探索

FCS内部也分三大区域：INS（导航等传感器，与外部进行数据交互） FMS（飞管系统） Controller（控制器，包含多种控制算法）

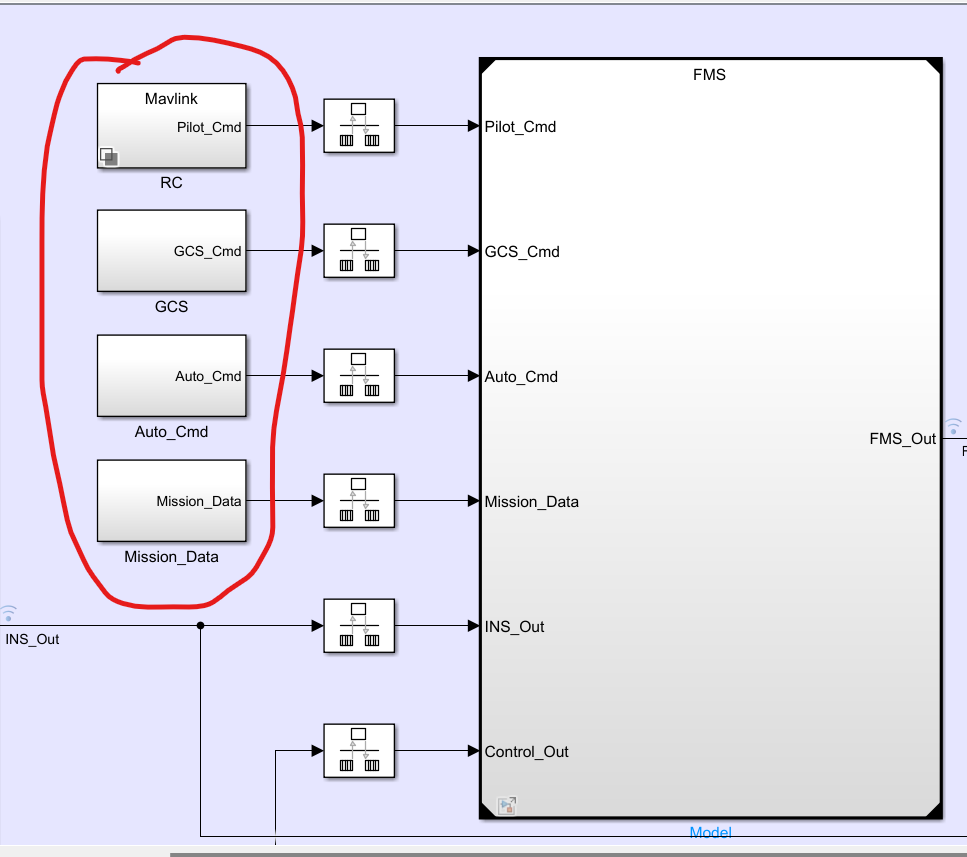


其中，核心修改区为FMS和Controller

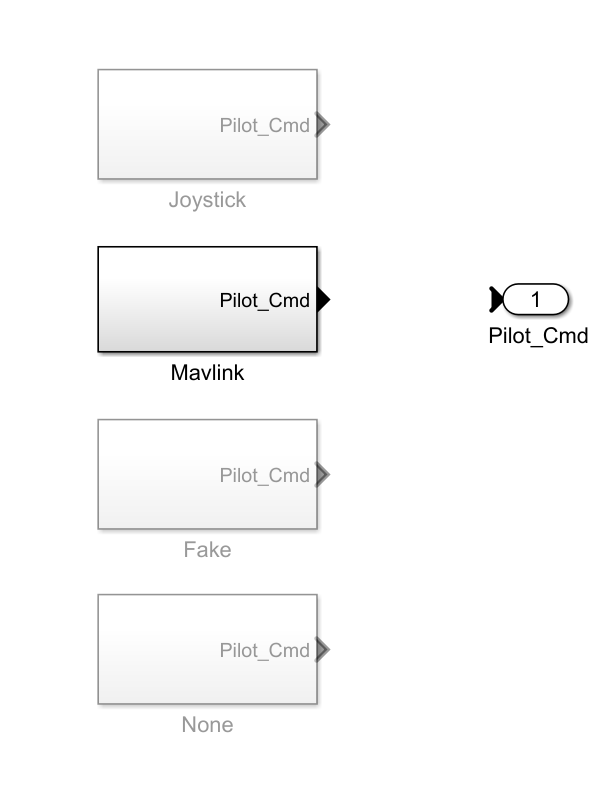
以下进行详细介绍

1. FMS（飞管（飞行管理））

它其实是一个大型的反馈转换系统，当我们期望实体达到某一速度、高度、偏角等等状态（多寡取决于你搭载和开发使用的传感器），它可以通过闭环反馈控制达到这一效果。具体我们看一下它的外围和内部，分析它的工作方式。

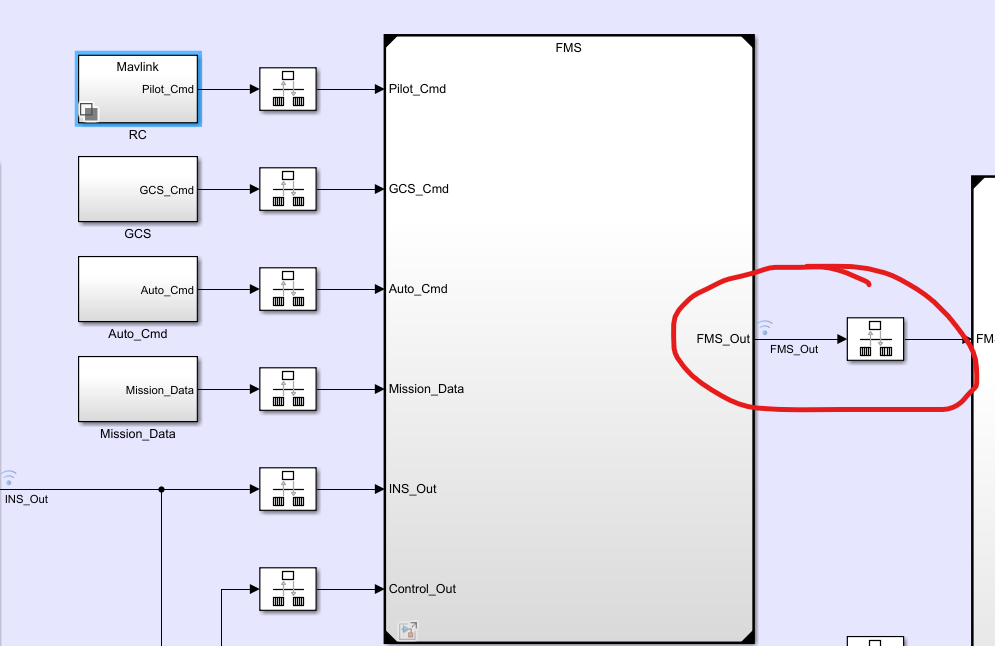


我们注意到飞管有四个输入，其中，第一个（RC）是通过遥控器进行输入



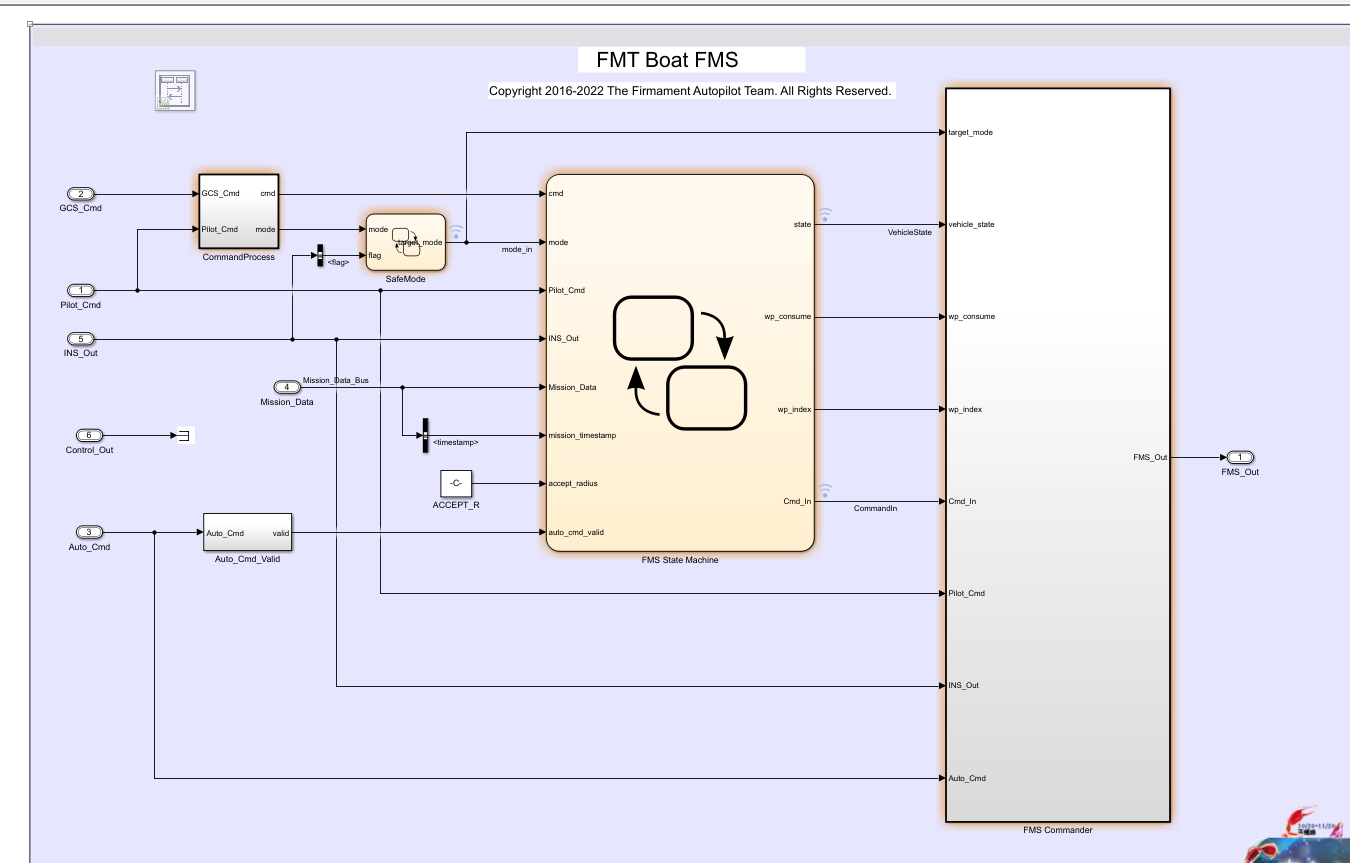
其中，从上到下分别为Joystick（游戏手柄控制）、Mavlink（通过电脑sumilink输入指令来控制）、Fake（通过虚拟摇杆来控制，与遥控器效果相同）None（不用遥控器）

FMS的作用体现在例如你通过摇杆来控制船体前进，你拨一定程度的摇杆其实对应了一定的速度，那么，如何将你拨摇杆的程度转换为对应的速度呢？



我们注意到FMS有一个大的输出，我们仍然以速度来举例，相当于船体通过将外部控制信号进行转换，转换成编程语言中的‘期望速度’，而这是后续PID控制中的重要参数“expected speed”。

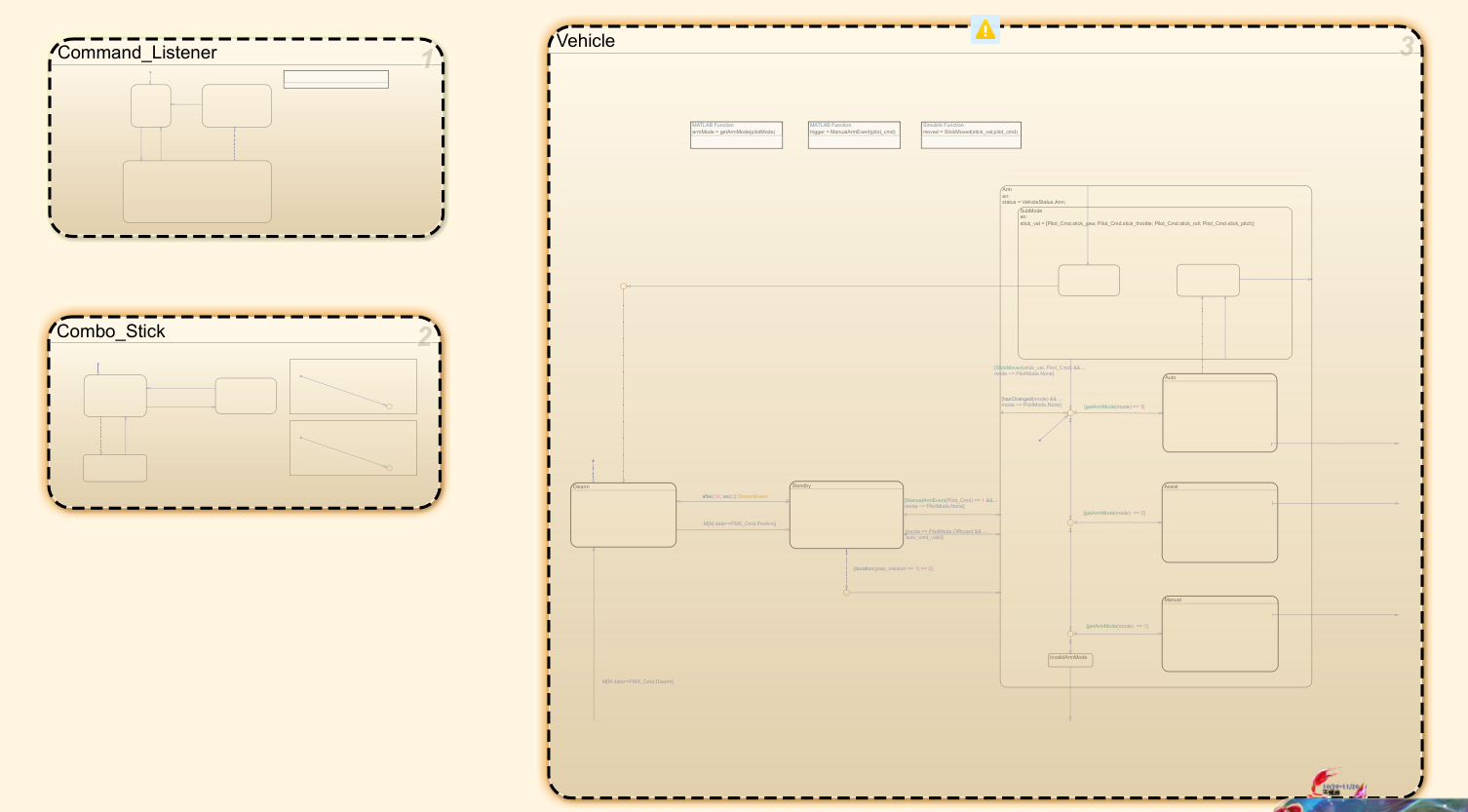
在了解FMS的内外输入输出后，我们探究一下它的内部



经常开飞机的小伙伴们都知道：飞机从起飞到降落的过程中有很多的状态（准备状态、加速状态、自稳状态等等）

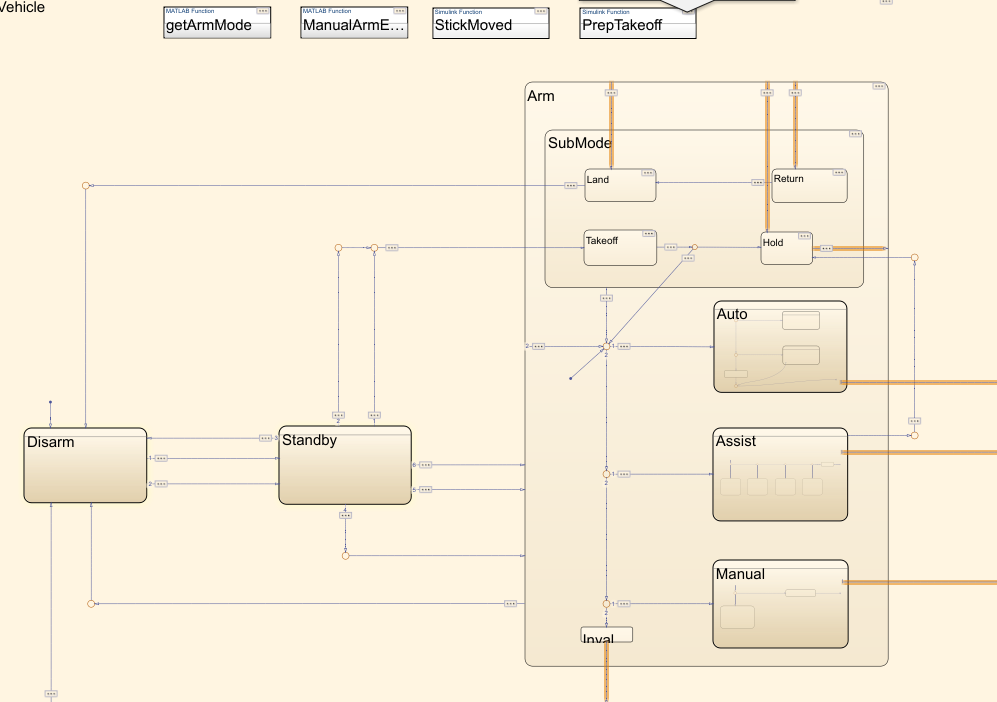
而飞管的真正作用体现在根据你输入的命令知道自己要处于什么模式。

让实体知道自己要处于什么模式的模块为上面的FMS State Machine（状态机）



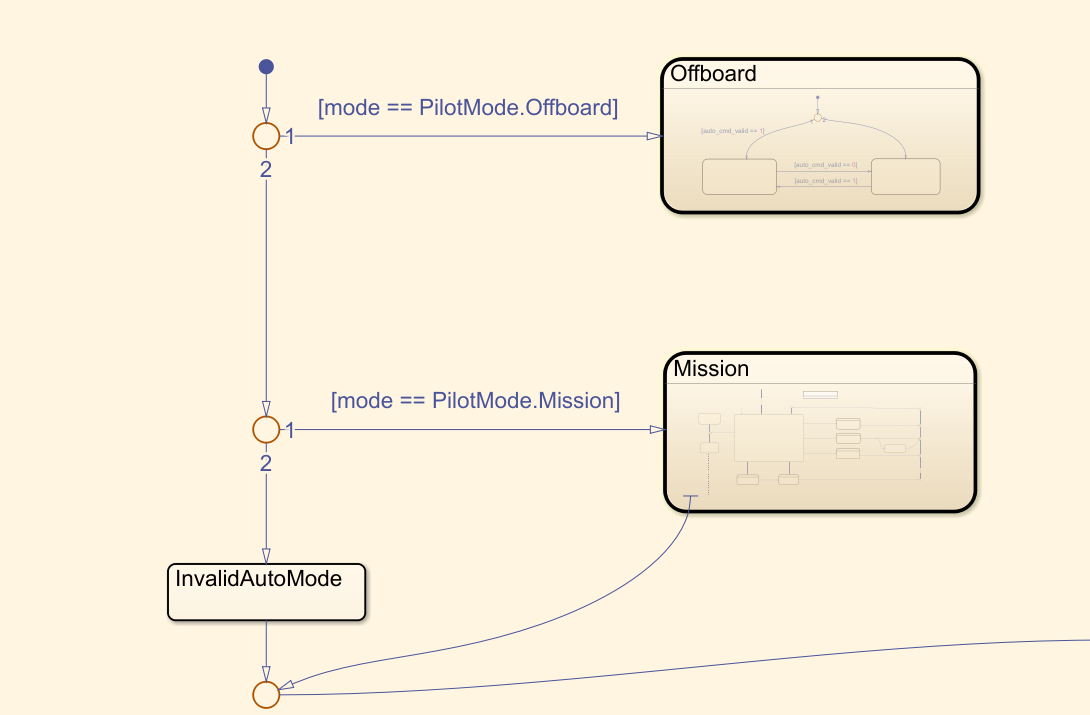
接下来我们详细介绍一下状态机：

我们点击右边这一块，进入后，首先右下角的Disarm为上锁模式



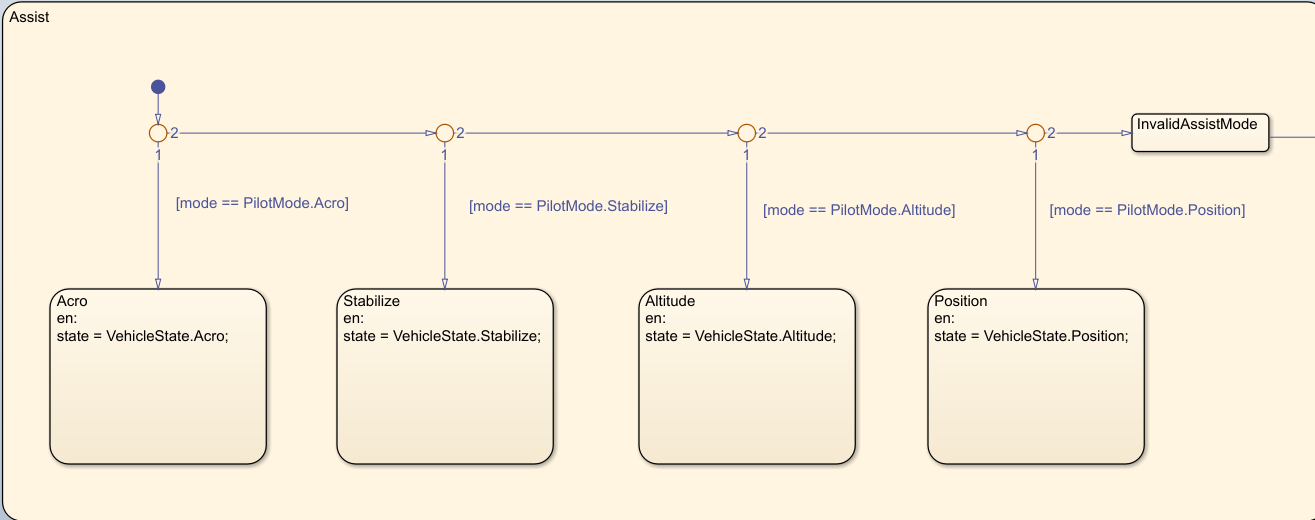
一般在遥控器输入后，会通过Disam向右走进入Standby进入一个准备阶段，然后再往后走进入飞机的某个状态

1. Auto模块（内包含Offboard（板外模式）Mission（航点模式））

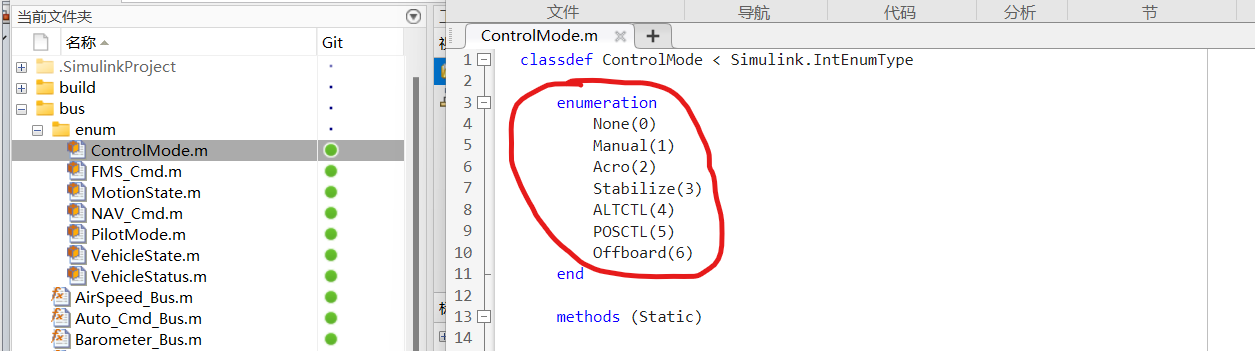


1. Assist模块（acro （特技模式）Stabilize（自稳模式）Position（定点模式））

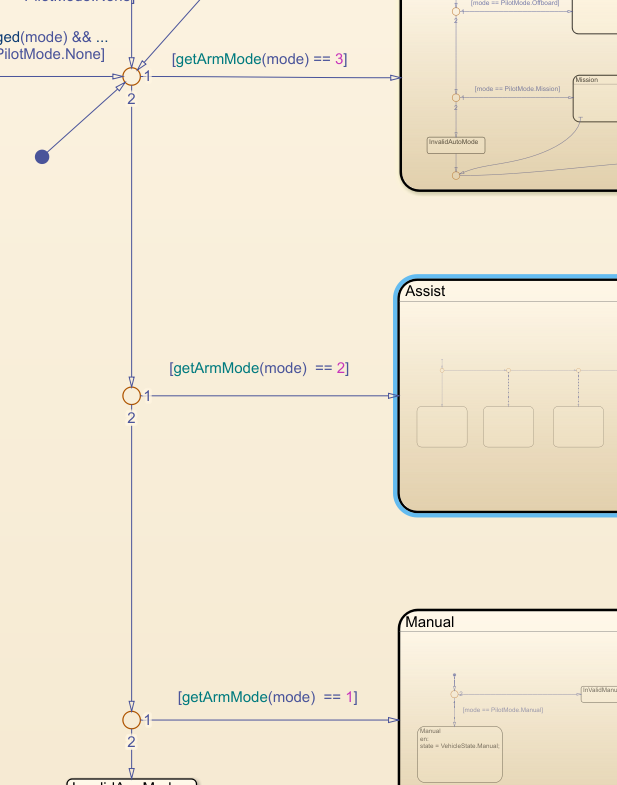
[offboard模式的开发及应用-CSDN博客](https://blog.csdn.net/msq19895070/article/details/78779401)



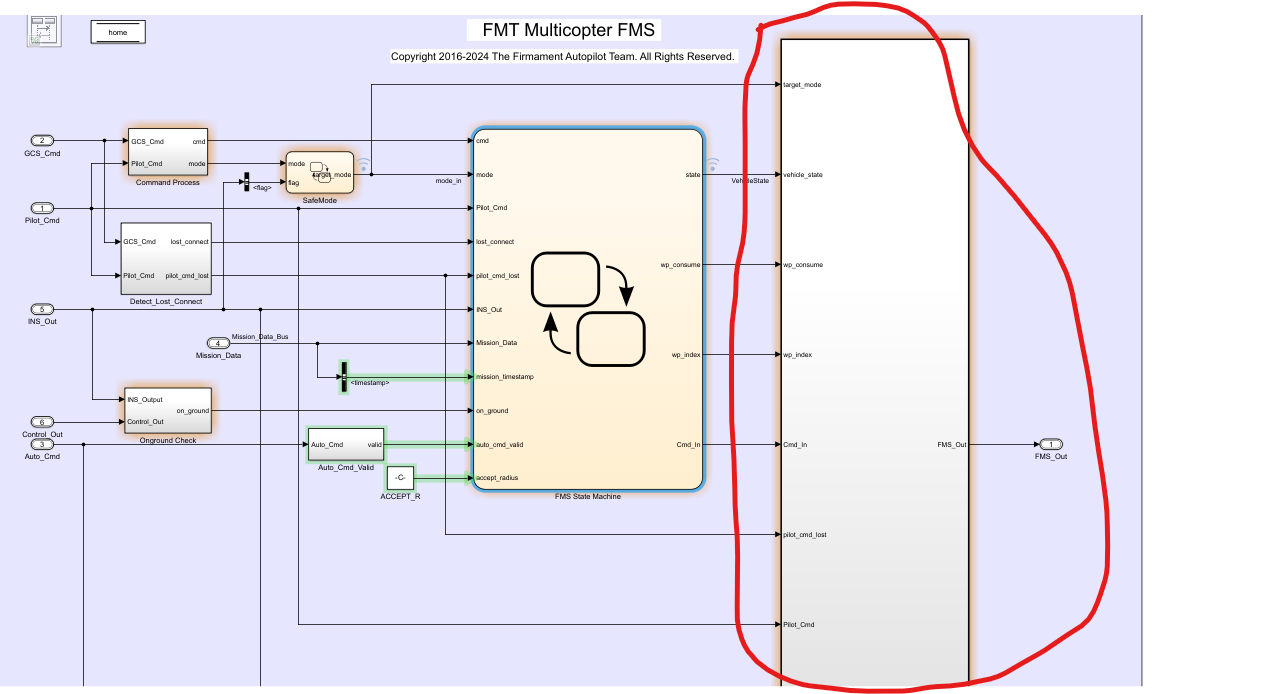
那么，模式是怎么对应上的呢，具体是通过一个数来对应，具体我们可以通过bus中的ControlMode中查看，具体如下



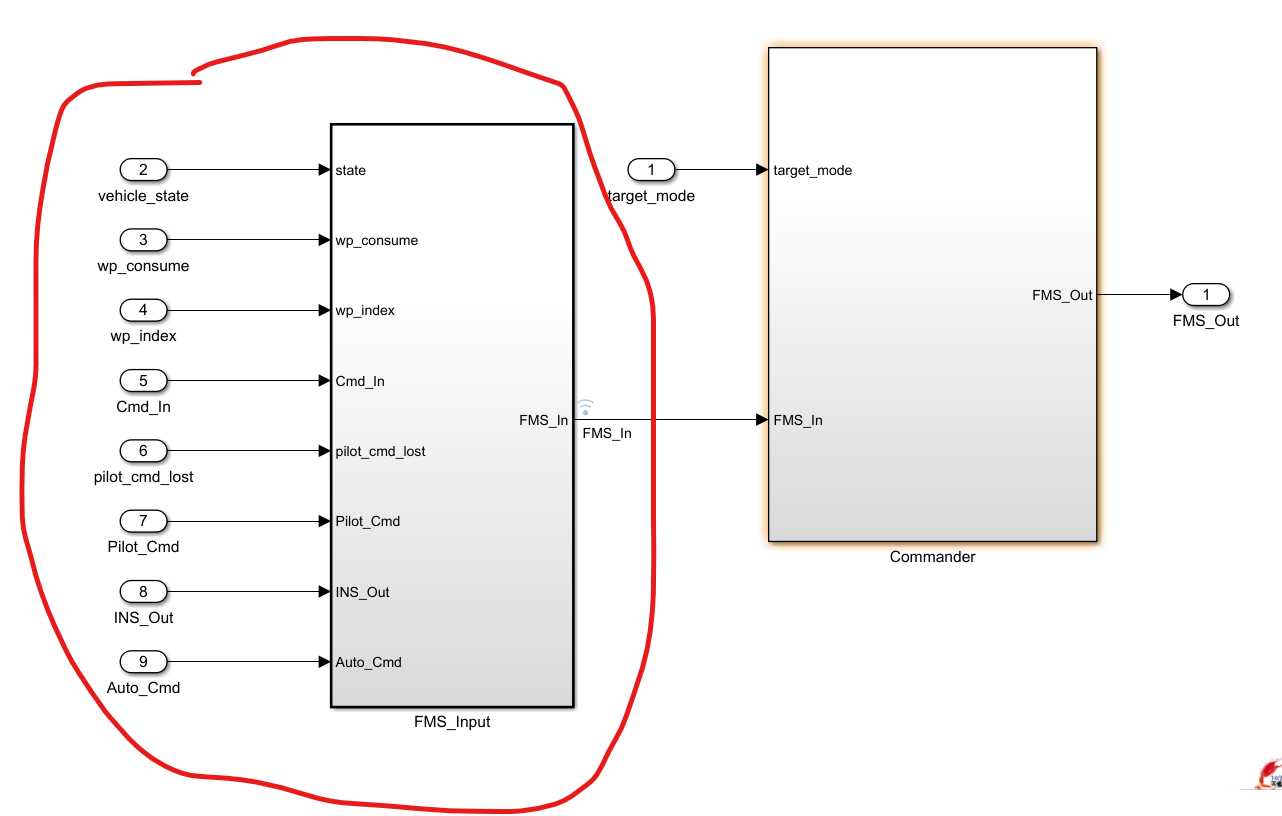
我们也可以看到它底层类似于C++的if-else判断通过轮询对应数字来进入相应状态框



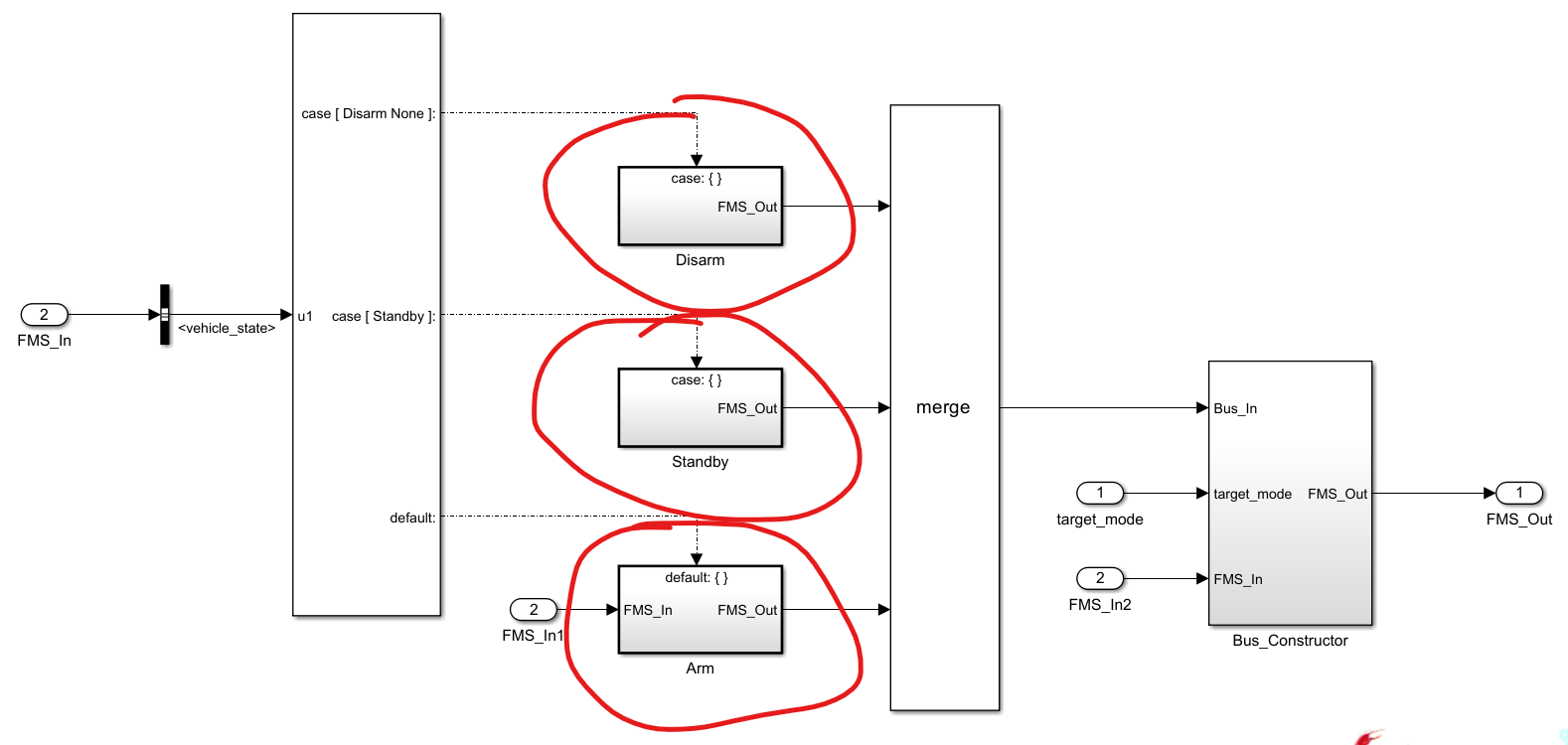
在获取自己的状态后，接着要通过摇杆来输出一个期望的速度，右边这一块为通过摇杆状态来计算输出一个相应的速度值



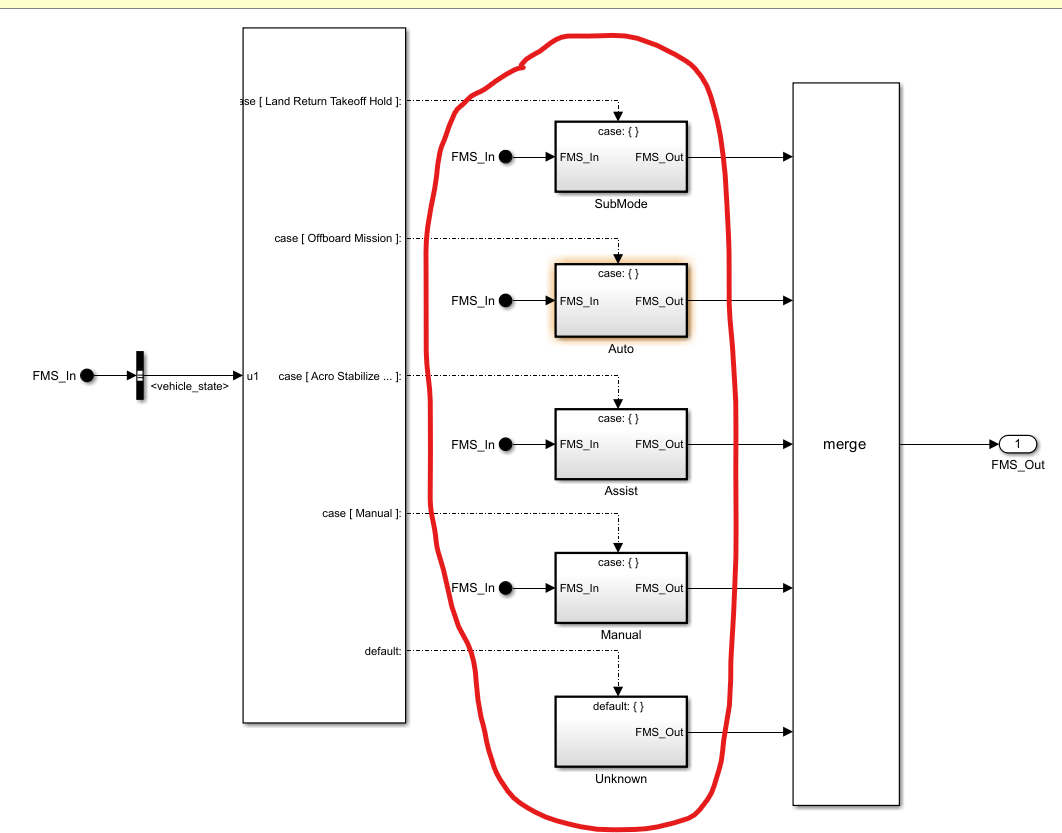
我们点击进入以后，看到左边这一块为在确认状态后通过FMT\_Input来汇总数据，输出一个总线数据



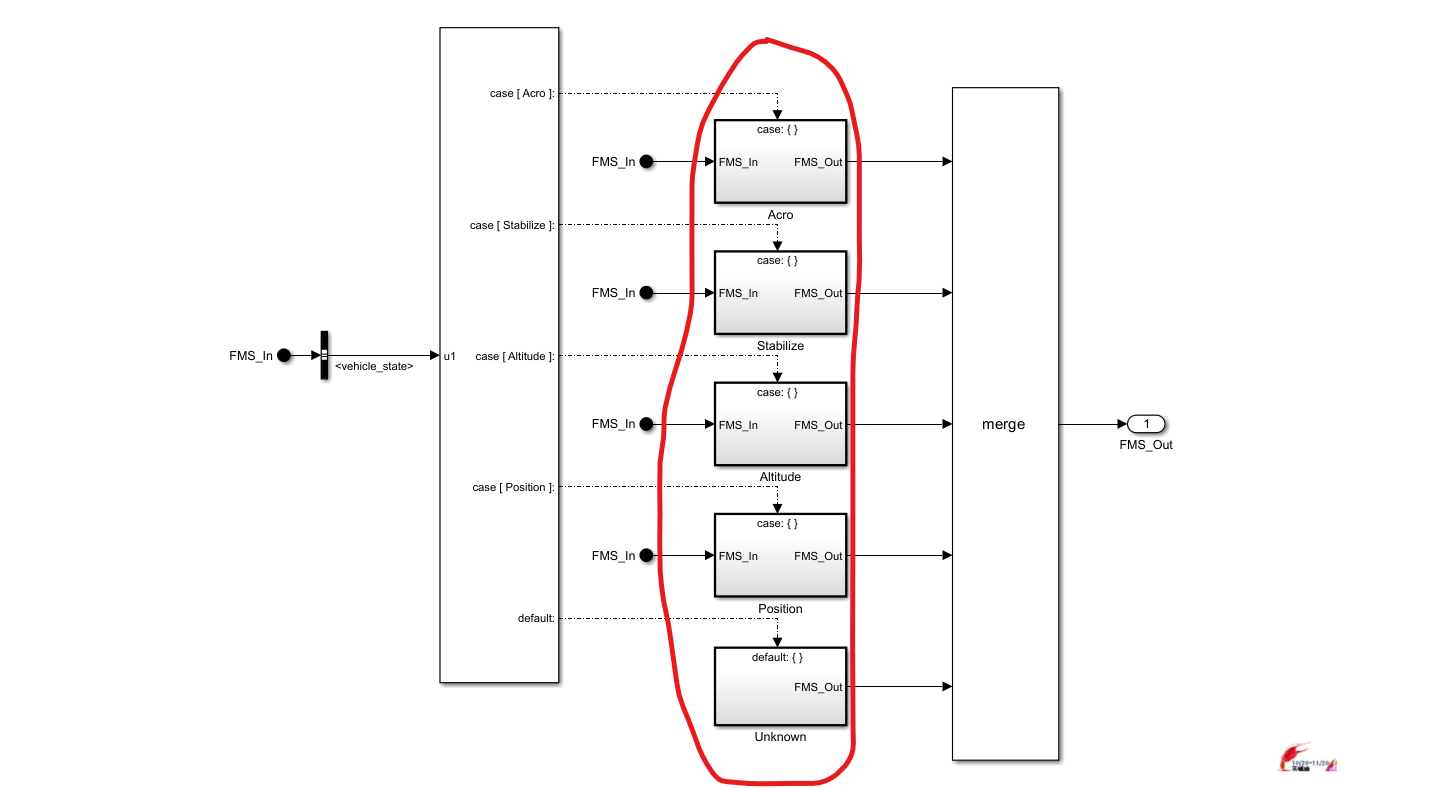
我们进入右边的Commander以后，中间三个对应状态从上到下分别为Disarm（上锁）、Standby（准备）、Arm（解锁后的模式）



进入Arm以后，我们看到中间从上到下分别为SubMode（下级模式）、Auto（自动模式）、Assist（辅助模式）、Manual（手动模式）



进入到Assist中我们就可以看到其中的一些如定点、自稳模式，其实是在状态机知道自己的状态后进入到这里的对应状态框，不同模式算法不同，会输出不同的期望速度和力矩

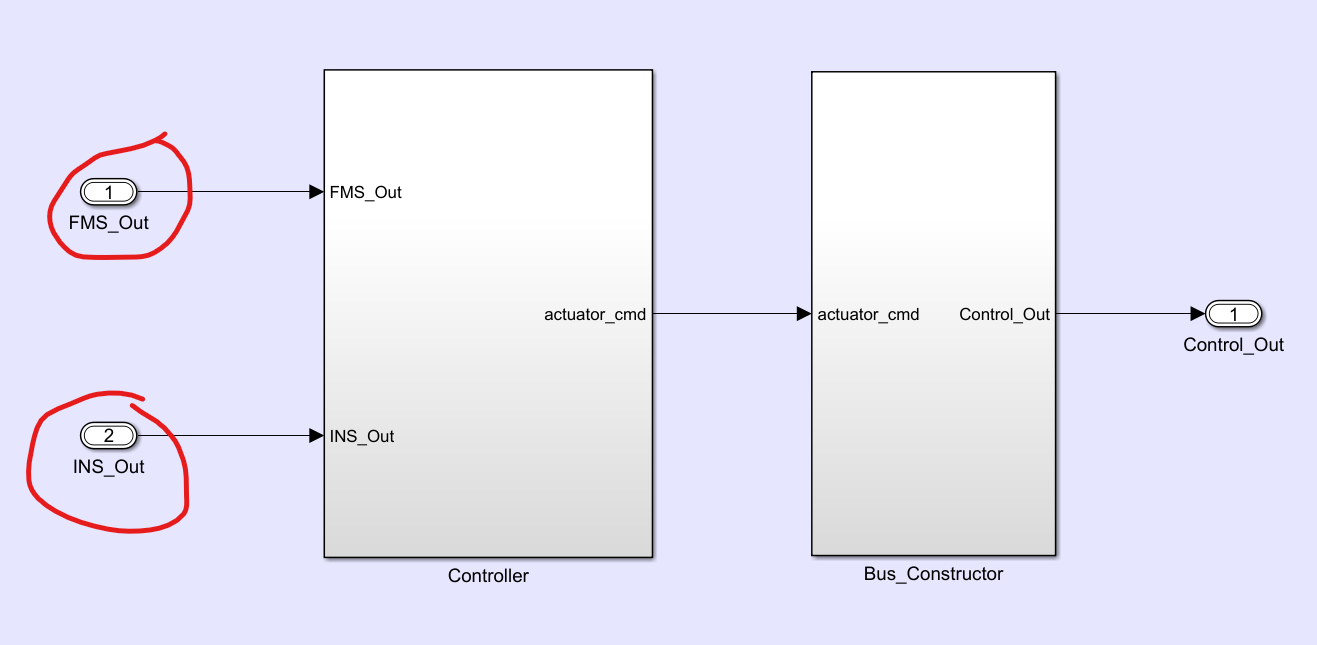


最后，期望数据会通过总线输出出来。

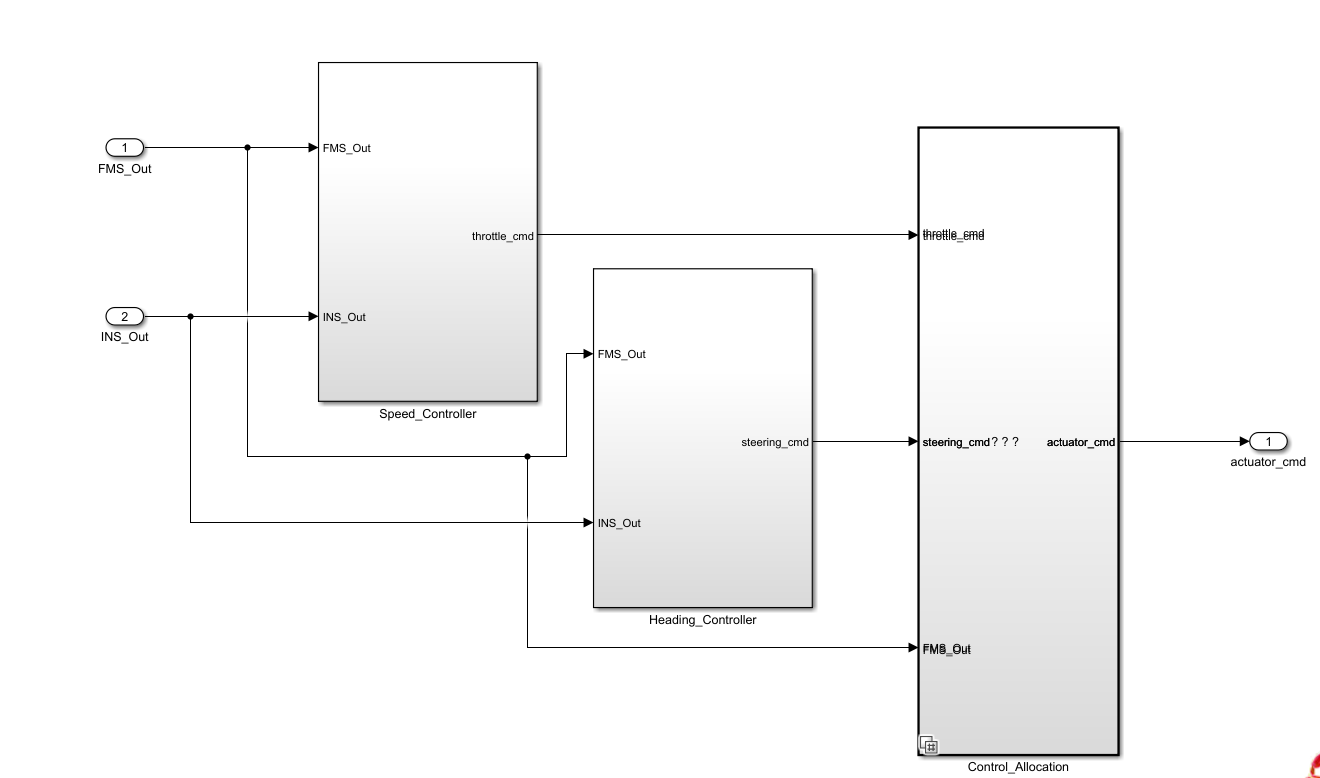
总之，飞管系统是通过接收到模式和遥控信号后，通过相应模式来走状态机来确认你的状态，在确定了你的状态以后，通过你的状态来通过对应算法来算出期望速度（数据），然后期望速度进入控制器中。

2.Controller（控制器）

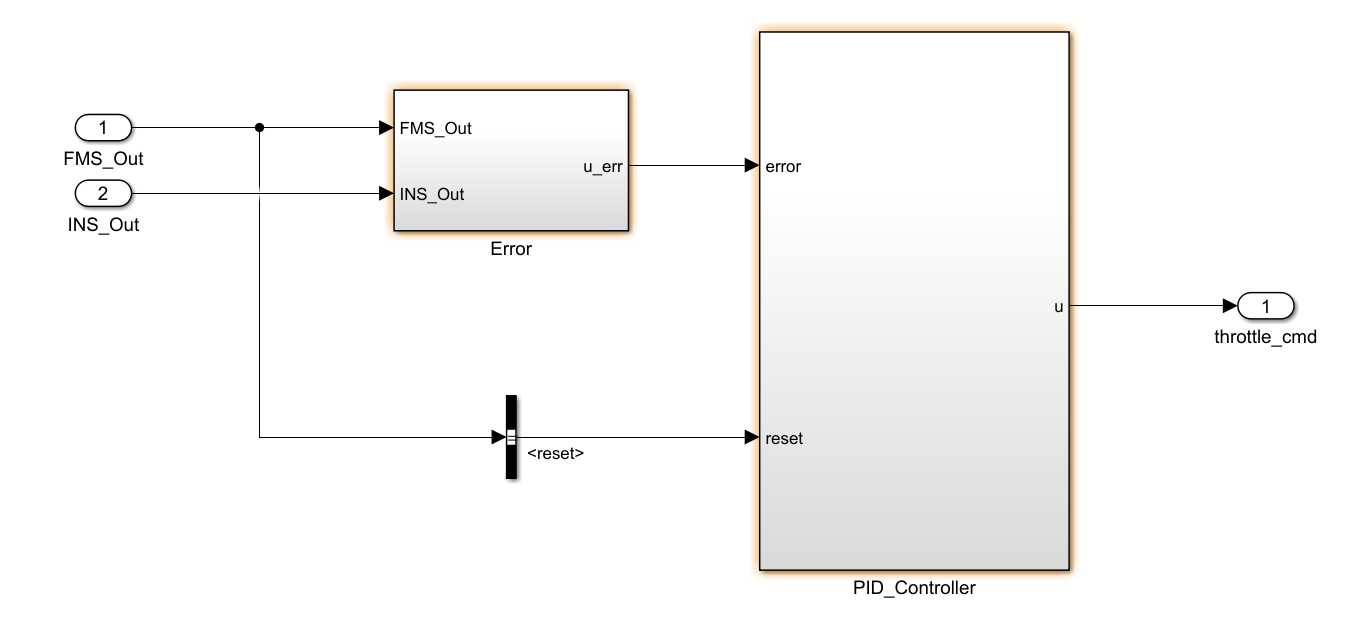
我们进入Controller模块后，Controller模块有两个输入，分别为FMS\_Out（飞管输出的期望速度），INS\_Out（导航输出的数据，即船当前的速度）

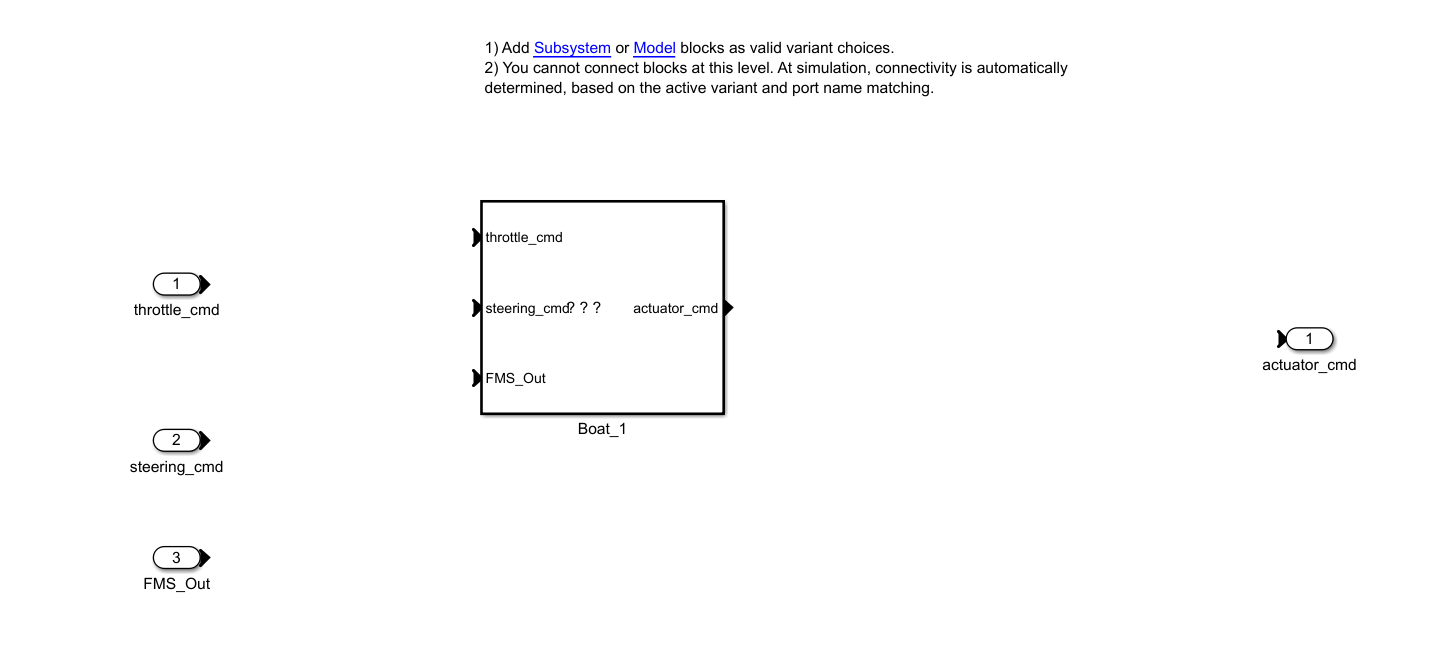


我们进入Controller后（以Boat Controller为例），看到了Speed\_Controller（速度控制）、Heading\_Controller（首向控制）



进入Speed\_Controller后，我们发现它其实是通过对FMS\_Out和INS\_Out进行作差，得到一个稳定补偿量。而这里是得到一个稳定的油门命令去控制船，但是这里相应数据还是一个信号，应当转换为对应的PWM值输出





如何实现上图的数据转换，让我们进入Boat\_1去了解

