1. 实验名称及目的

1.1. 实验名称

外部控制开发数传通信实验

1.2. 实验目的

以如下连接方式为例介绍数传的通信方式：通过 USB-TTL 模块有线连接飞控和电脑，并实现硬件在环下的外部控制。

1.3. 关键知识点

关键知识点 1：飞控与电脑或机载板卡之间的连接方式

连接类型		用途
有线连接（常用于飞机与机载电脑的通信，连接飞控上 TELEM2；或地面调试阶段飞控与地面电脑的通信（此时连 TELEM1）。）	USB-TTL 模块有线连接飞控和电脑	实现硬件在环（HIL）模式下的外部控制，确保系统的实时性和可靠性。
	通过数传线（USB-TTL）直接连接飞控和机载板卡	启动 MAVROS，并实现数据监测，用于在飞行过程中监测和控制飞行状态。
无线连接（主要用于飞机与地面电脑的通信，通常连接飞控上的慢速口 TELEM1）	低速数传连接飞控真机	实现地面站的无线飞行调试，适用于较远距离但低速的数据传输。（使用 3DR X6 或微空科技 LR900P 等低速数传设备）
	高速数传连接飞控真机	通过自定义数据协议，实现外部 Python/Simulink 的实时真机控制，适用于高实时性、高速数据传输需求的应用场景。（使用雷迅 P9Radio 等高速数传设备）
	WIFI/4G 方式	通过 WIFI 网络连接，实现对飞机

	连接飞机	的外部控制，适用于短距离且需要较高数据速率的控制任务。

注：TELEM1 和 TELEM2 这两个接口的默认波特率不同，但在大多数情况下，可以在飞控的配置中通过修改参数来设置它们的波特率。两个 TELEM 接口可以为系统提供冗余。

关键知识点 2：配置串口波特率和传输速率

波特率决定了数传的传输速度，通常以“位每秒”（bps）为单位，通常选择 57600 低速无线，115200 中速（高速无线通信，或有线测试），或 921600 高速（硬件在环仿真，飞控连板卡）。

注意：波特率必须匹配双方设备（如飞控和地面站）的设置，确保连接稳定。

参考资料：https://docs.px4.io/main/en/peripherals/mavlink_peripherals.html

关键知识点 3：配置外部控制程序和通信串口的连接

QGC 与数传的连接

Python 与数传的连接

Simulink 与数传的连接

2. 实验效果

3. 文件目录

例程目录：安装目录\0.ApiExps\e19_Ros2CtrlDemo1.UE4StarterContent

文件夹/文件名称	说明

4. 运行环境

序号	软件要求	硬件要求	
		名称	数量(个)
1	Windows 10 及以上版本	笔记本/台式电脑①	1
2	RflySim 工具链	Pixhawk 6X 或其它飞控②	1
3	Visual Studio Code	USB_TTL 线	1

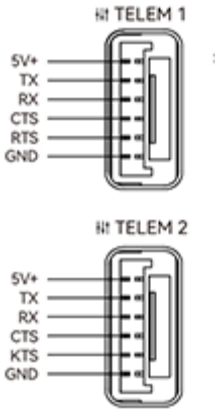
① ：推荐配置请见：https://rflysim.com/doc/zh/HowToInstall.pdf

② ：若使用 Pixhawk 6X 飞控，平台安装时的编译命令为：px4_fmu-v6x_default，推荐 PX4 固件版本为：1.13.2。配套飞控配置请见：https://rflysim.com/doc/zh/B/2.3Pixhawk6X.html

5. 实验步骤

5.1. 硬件连线情况概览

Step 1：飞控端接口情况



如上图，可以看到飞控数传口 TELEM1 和 TELEM2 的线序情况，我们需要注意的是电源（VCC,5V+）、发送（TX）、接收（RX）和地线（GND）的分布情况，分别位于 1 2 3 6 号口（从 1 开始计数）。

Step 2：USB_TTL

1.连接端口情况

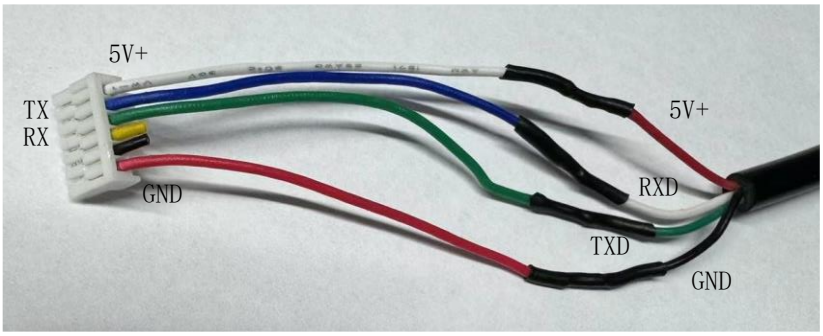
飞控数传口使用的是 GH1.25 6P 的端子，可以从网上购买端口线缆，并减去 4-5 号线，仅保留 1 2 3 6 号线缆进行备用。

2.电脑端接口情况：

从网上可以搜索 USB-TTL 的模块，从上图可以看到它的线序定义“红线+5V、黑线 GND、白线 RX、绿线 TX”。

3.线缆连接

将 USB-TTL 的红线(+5V)，通过 GH1.25-6P 端子，连接到飞控 TELEM1 端口的 +5V（1 号口）上；USB-TTL 黑线 GND 连到飞控 TELEM1 的 GND（6 号口）上；TTL 白线 RXD（接收线）连接到飞控 TELEM1 的 TX（发送线 2 号口）；USB-TTL 绿线 TXD（发送线）连接到飞控 TELEM1 的 RX（接收线 3 号口）。最终效果如下图：



注：GH1.25 6P 端口的线缆颜色不一定和 USB-TLL 的颜色定义一致，主要是参考飞控 TELEM 口的顺序。例如，上图中 GH1.25 6P 的白色线缆为 1 号口对应 5V+，而红色线缆为 6 号口对应 GND。

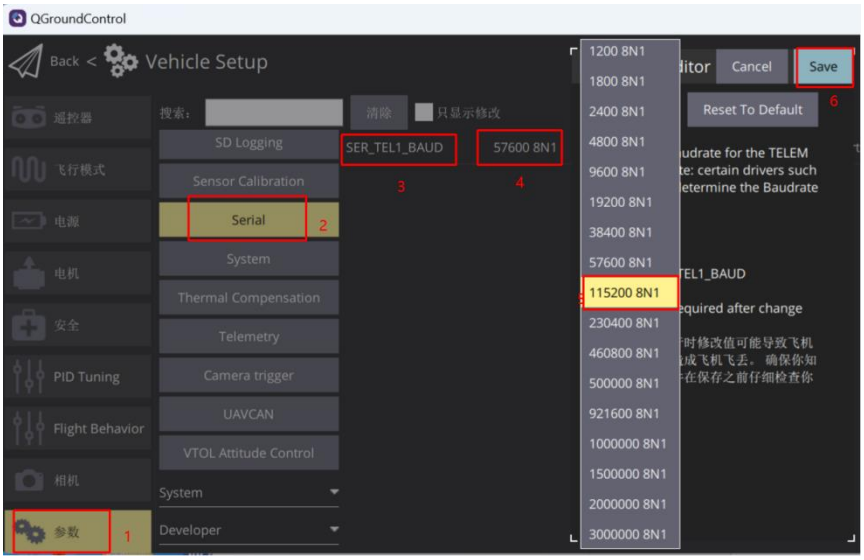
注：上述线缆可以自定购买后焊接，也可以直接购买飞思实验室的教具“硬件在环仿真套装”会自带连接好的线缆。

5.2. 数传线的配置与连接方法

本实验以 115200 波特率为例子，展示设置与连接方法。

Step 1：设置 MAVLINK 端口 TELEM1 的波特率

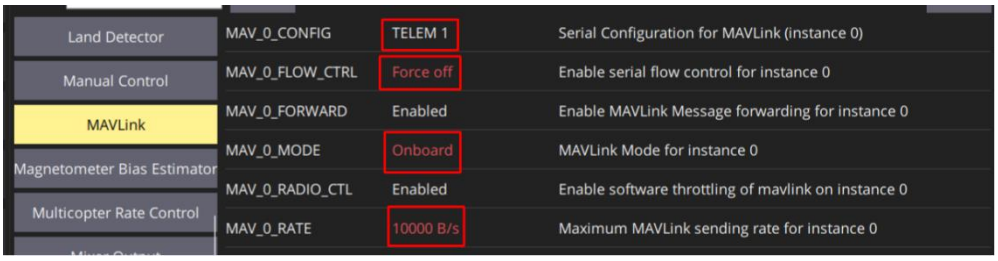
通过 USB 口连接飞控到电脑（注意，这里不是用数传的 USB 线，而是用硬件在环仿真的 USB-TypeC 的线），然后打开 QGC 等待连接成功后，进入参数设置页面。如下图所示：1.找到“参数/Parameters”页面，2. 找到 Serial 标签，3. 找到“SER_TEL1_BAUD”参数，4.点击当前波特率“57600 8N1”，5. 在弹出菜单中选择 115200



注意：上述参数修改后，需要重启飞控才能生效。请**重新插拔飞控**，来进行下面步骤。

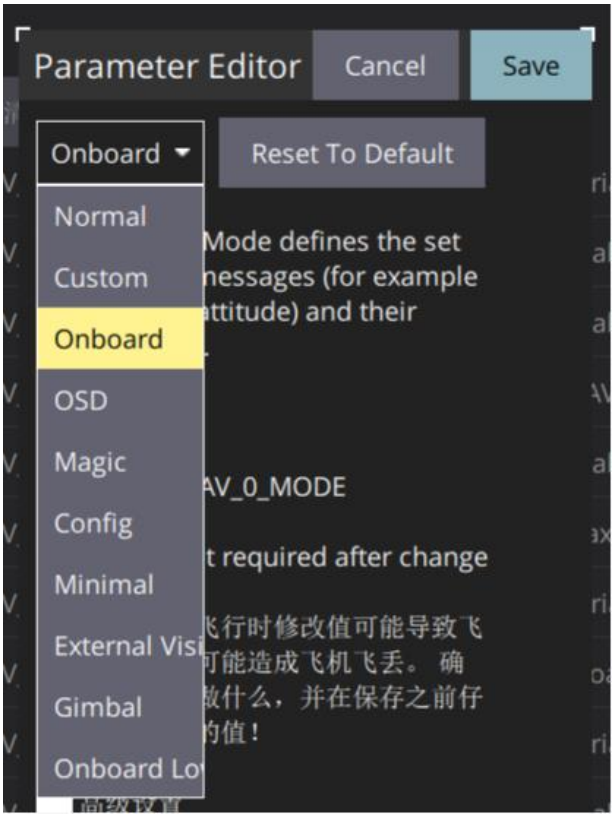
Step 2：配置与 TELEM1 所选波特率匹配的 MAVLINK 实例

重新连上飞控后，进入参数界面下的 MAVLINK 页面。本实验可以参考如下设置

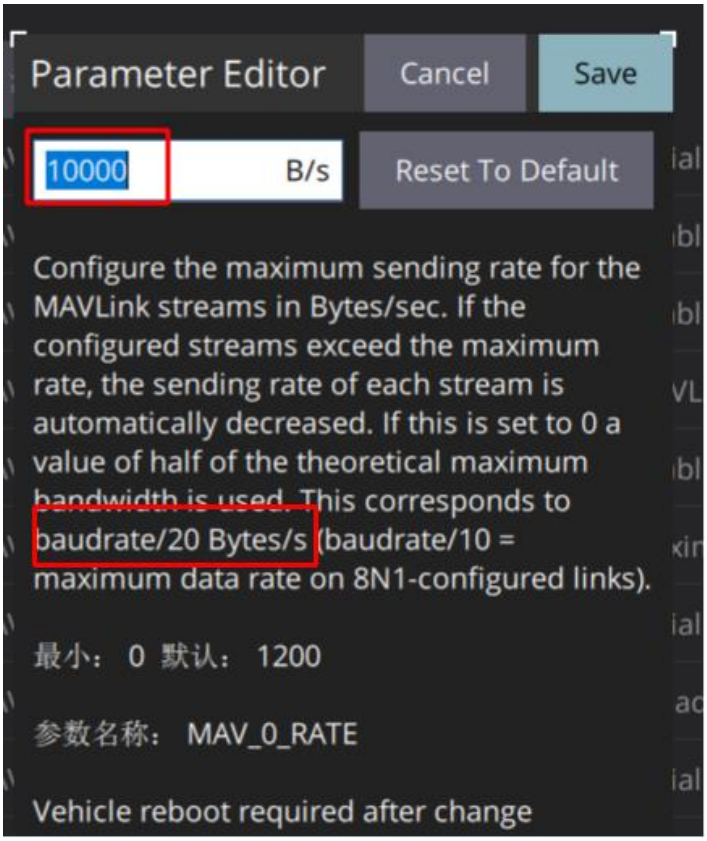


参数详解如下：

- MAV_0CONFIG：TELEM1。设定 MAVLink0 号实例映射到 TELEM1 口。
- MAV_0_FLOW_CTRL：Force off，设置为不启用流控制（1.13 版本后才有此选项）。注 1：这里默认可以设定为 Auto-detect，如果出现不能连接数传的情况，请设置为 Force off。注 2：如果 QGC 界面中，没有 MAV_0_FLOW_CTRL 参数，请重新还原固件后，重新尝试。
- MAV_0_FORWARD：是否启用消息转发，设置为 Enable 即可。
- MAV_0_RADIO_CTL：是否启用 mavlink 遥控器控制，设置为 Enable 即可。
- MAV_0_MODE：数据传输协议，本实验选择 Onboard 模式，对应了比较全的数据输出。飞控不可能把所有数据都传出来，根据不同的需求传输不同的数据量和频率，确保能够和链路的带宽（传输速率）相匹配。有如下协议可选：Normal（常规连接地面站调试）、Onboard（连接机载电脑，数据量大，最好配合 921600 波特率）、Custom（完全自定义数据和频率，需要修改源码）等。

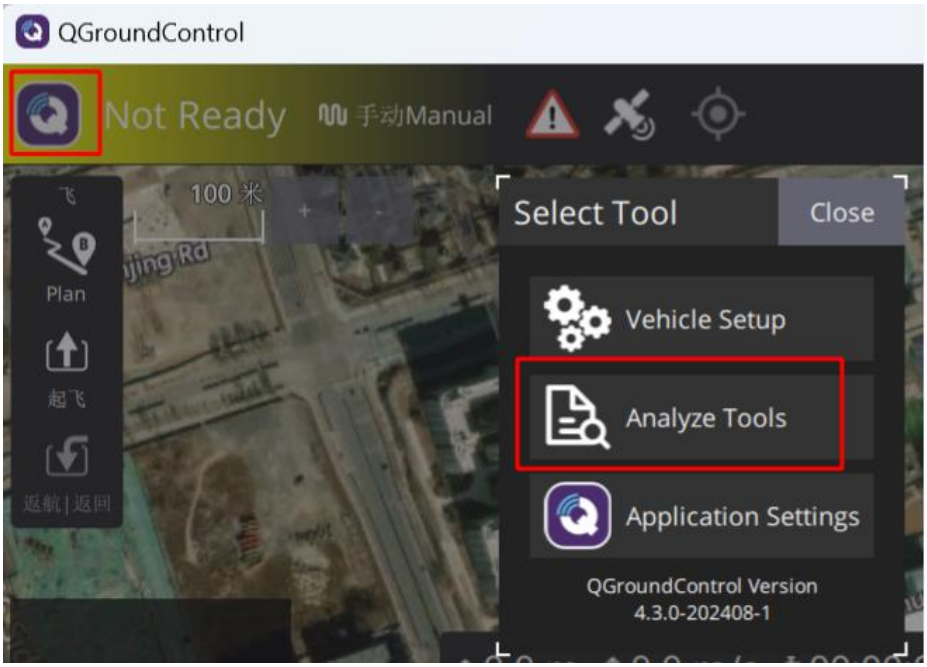


- MAV_0_RATE：数据传输速率，这里设置为 10000。注：如下图可知，传输速率通常设置为：波特率/20，最多设置为波特率/10（理论最大值）。 本实验使用最大值，即 波特率/10＝11520，取一定裕度，最终为 10000。

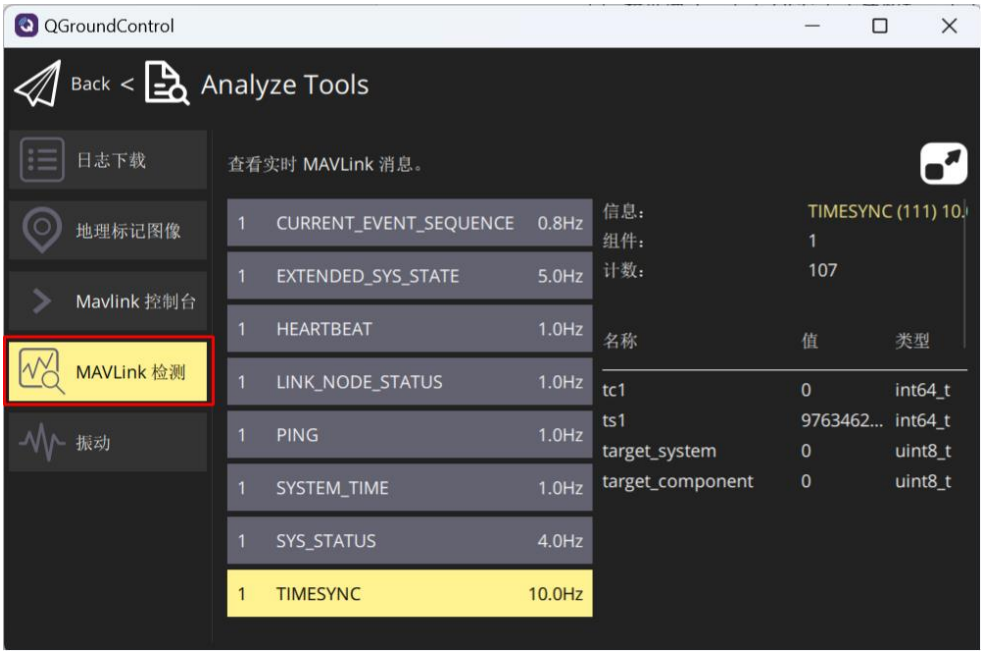


Step 3：数据确认

点击 QGC 左上角图标，点击进入“Analyze Tools”分析工具页面。

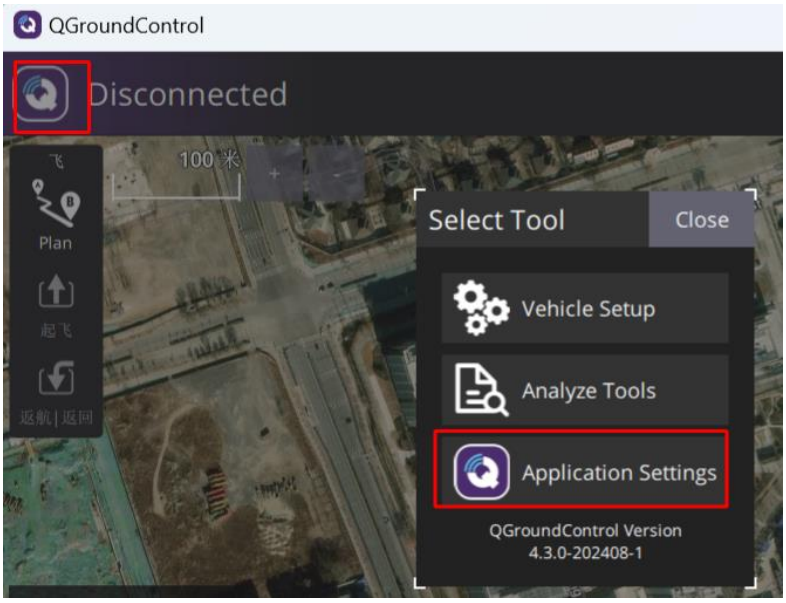


在 MAVLink 检测页面，可以看到当前所有接收到的 MAVLink 消息。



Step 4：数传连接准备

通过 USB_TTL 连接飞控，打开 QGroundControl，点击左上角图标，再点击“Application Settings”进入应用设置页面。

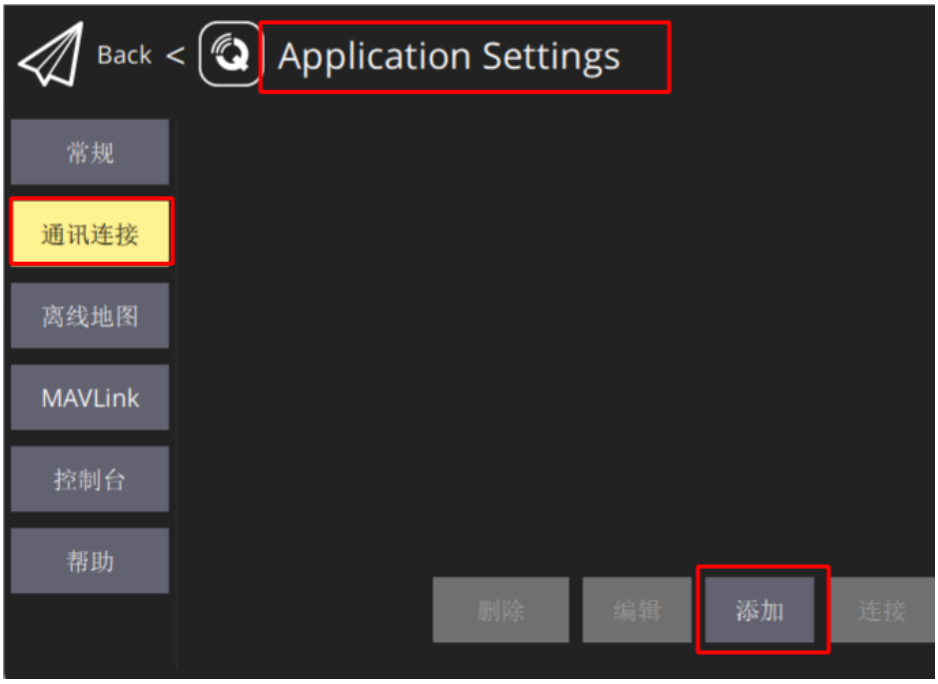


进入 常规 – 自动连接到下列设备 – 取消勾选 SiK 电台，确保自动连接数传串口功能处于关闭状态。

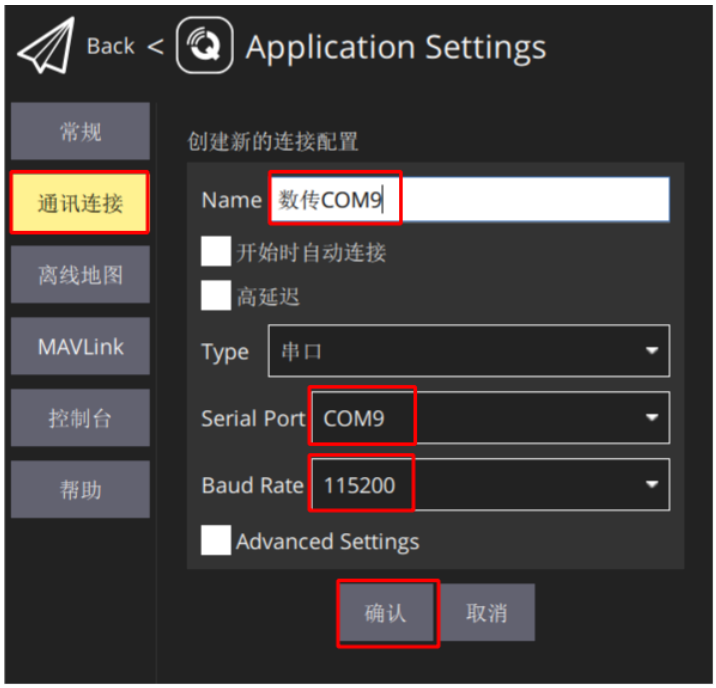
注：如果勾选“SiK 电台”，QGroundControl 会在后台一直尝试使用 57600 的默认波特率连接到数传模块，这种模式与我们设置的波特率不匹配，因此会导致数传串口一直处于错误占用状态，最终导致其他程序无法正常连接数传。

Step 5：QGC 数传连接配置

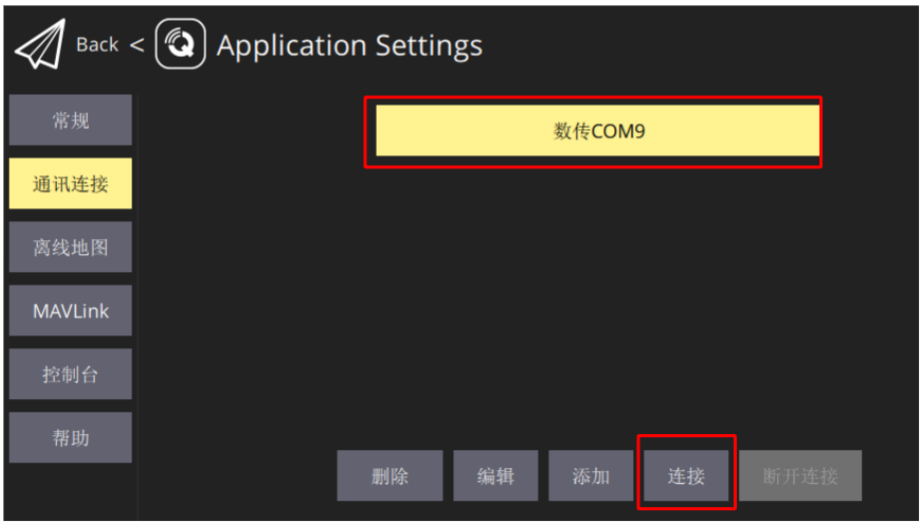
接着上面的步骤，在 Application Settings 的应用设置页面，进入“通信连接”页面，点击“添加”按钮。



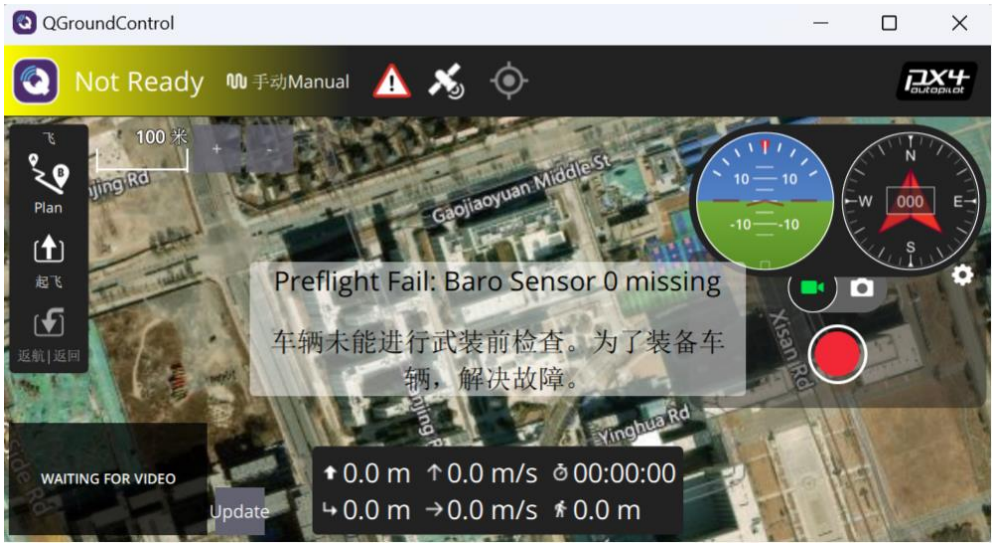
输入串口号（图中是 COM9，根据实际串口确定），波特率设置为设定值（本例程为 115200），设定连接名（任意设置即可，便于区分不同串口），最后点“确认”按钮。



然后，继续在“通讯连接”页面，选中刚才创建的串口名，然后点击“连接”



回到 QGroundControl 可以看到已经正常连上飞控了。注：下图的警告属于正常现象，因为飞控处于硬件在环仿真模式，却未连接 CopterSim 接收传感器数据，会导致飞控校验不通过。

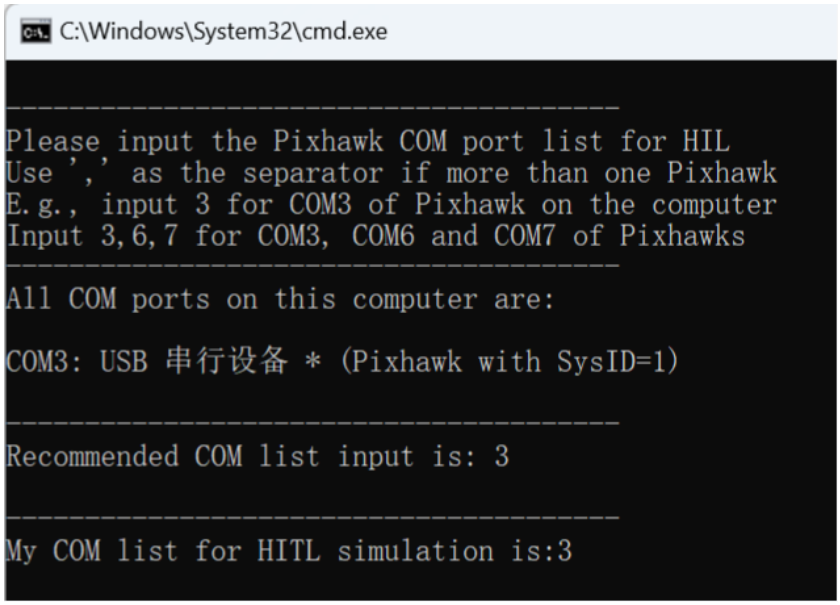


5.3. 地面站通过数传连接飞控，进行起飞、降落、航路飞行。

注：本实验需要两台电脑，避免 QGroundContol 的同时连 HIL 仿真和数传串口时数据冲突。

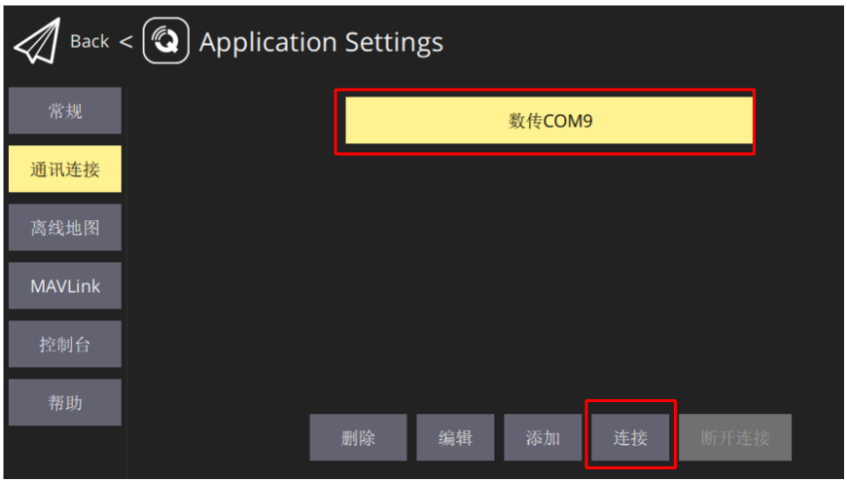
Step 1：USB 串口连接硬件在环仿真

通过 USB-TypeC 将飞控的 USB 口（注：这里是硬件在环仿真线，不是 USB-TTL 串口线）连接到电脑上，并运行桌面 RflyTools\HITLRun 快捷方式，输入串口号并回车，开始硬件在环仿真。

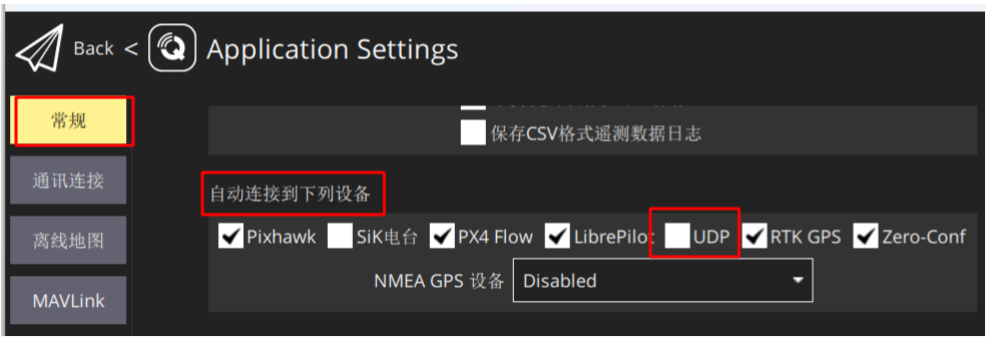


Step 2：数传串口连接地面站

将 USB-TTL 串口线（或自己的数传）连接到另一台电脑上，打开 QGroundControl 中，进入 Application Settings – 通讯连接页面，创建并连接自己的数传串口。



注意：如果没有两台电脑，只有一台电脑。则进入 QGC 应用设置 – 常规 – 自动连接到下列设备 中 取消勾选 “UDP” 选项，重新打开 QGroundControl，我们发现飞控没有自动连接硬件在环仿真飞控，这时候可以继续下列流程，使用数传串口连接飞控。

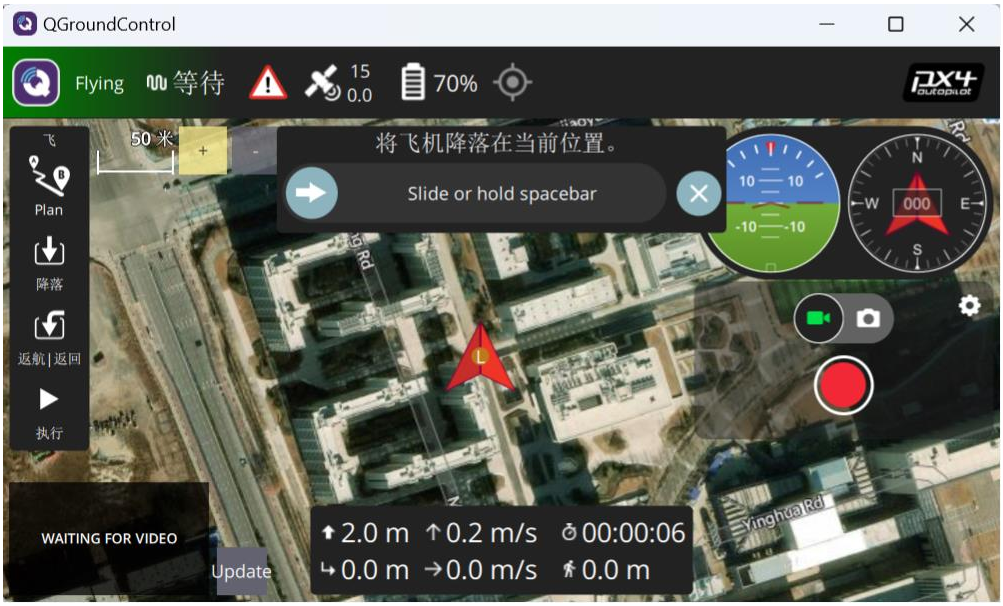


注意：做完本实验，一定要重新勾选“UDP”选项，不然硬件在环仿真时，QGC 不会

连接到 CopterSim 和飞控。

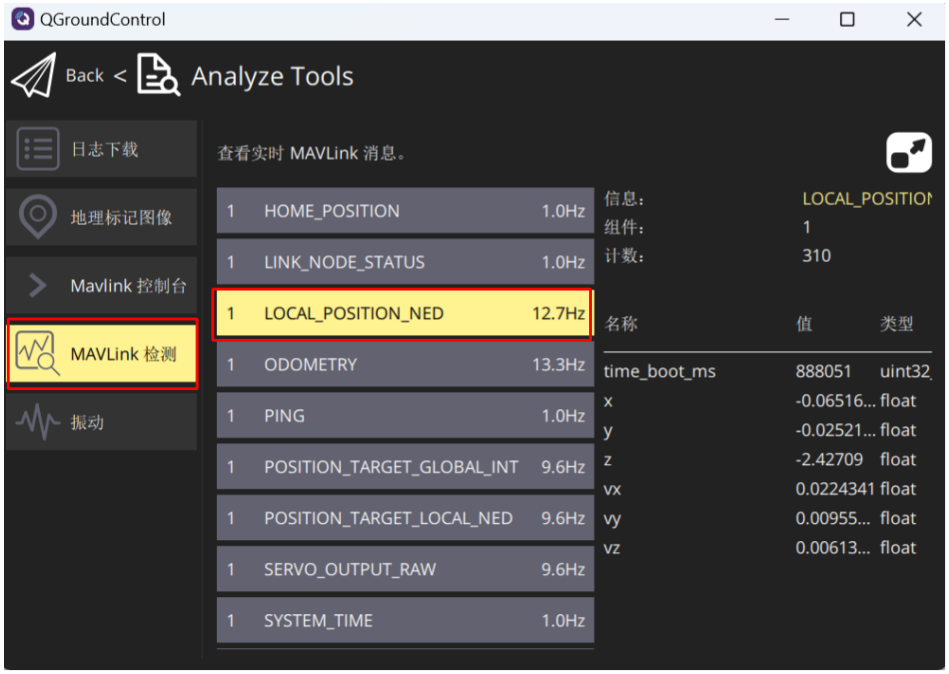
Step 3：执行外部控制任务

在 QGroundControl 中可以正常控制无人机起飞、降落以及画航线等，这和用地面站控制真实飞机的状态一致。



Step 4：数据确认

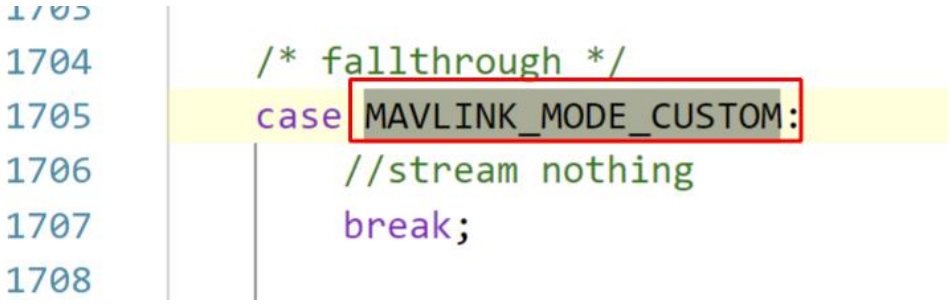
进入 MAVLink 检测页面，可以发现返回的数据非常多，但是数据频率不够稳定。这是因为 Onboard 的数据发送频率非常大，115200 波特率无法满足传输带宽要求。



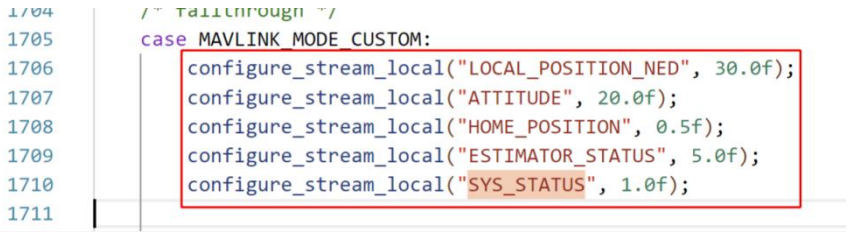
5.4. 自定义通信数据和频率设置实验

Step 1：添加 MAVLINK 消息

进入【平台安装目录】\Firmware\src\modules\mavlink 目录，用 VS Code 打开 mavlink_main.cpp 文件，并搜索“MAVLINK_MODE_CUSTOM”，定位到下图所示位置。



在此处添加想要发送的数据消息和频率，如下



注：这里我们主要添加了五条常用的消息，包括定位数据 LOCAL_POSITION_NED、姿态数据 ATTITUDE、起飞点坐标 HOME_POSITION（用于多机原点坐标解析）、滤波器状态 ESTIMATOR_STATUS（用于健康监测）、系统状态 SYS_STATUS（用于识别飞行状态），其频率设定如下

```
configure_stream_local("LOCAL_POSITION_NED", 30.0f);
configure_stream_local("ATTITUDE", 20.0f);
configure_stream_local("HOME_POSITION", 0.5f);
configure_stream_local("ESTIMATOR_STATUS", 5.0f);
configure_stream_local("SYS_STATUS", 1.0f);
```

注：可以从 Onboard 的配置页面中，直接复制粘贴感兴趣的消息即可。如果有其他的需求，例如做健康故障诊断，需要 IMU 数据，可以增加订阅 HIGHRES_IMU 消息，不过要考虑数传的通信带宽支持。

```
configure_stream_local("HIGHRES_IMU", 50.0f);
```

Step 2：重新编译固件

以 Pixhawk 6x 为例，对应编译命令为 px4_fmu-v6x_default。在 MATLAB 中输入 PX4Official px4_fmu-v6x_default

```
>> PX4Official px4_fmu-v6x_default
Modifed Data: 20231026
成功找到px4_fmu-v6x_default的cmake文件
在C:\PX4PSP\Firmware\boards\px4\fmu-v6x\default.cmake文件中添加px4_simulink_app模块
rcS中找到px4_simulink_app自启动项目，已配置正确，没有修改
删除旧的Simulink生成代码文件
成功删除旧的px4_simulink_app文件夹
成功新建px4_simulink_app\empty_file.c文件
```

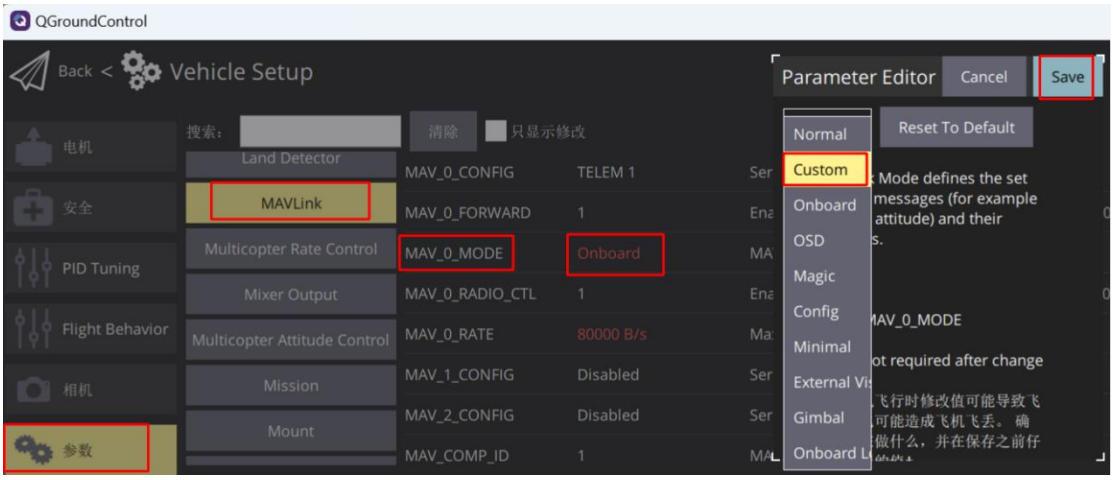
等待编译完成，即可得到修改好的固件 C:\PX4PSP\Firmware\build\px4_fmu-v6x_default \ px4_fmu-v6x_default.px4。

Step 3：烧录固件

在 MATLAB 中输入 PX4Upload，然后插入飞控，即可完成固件的覆盖。注：如果上述方法无法烧录固件，可使用 QGC 的手动烧写固件功能。

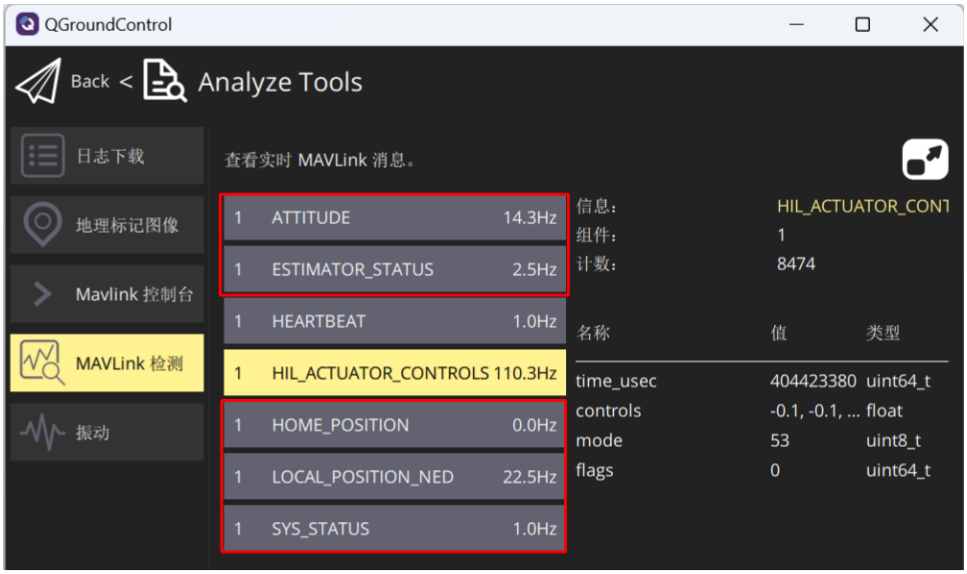
Step 4：配置 MAVLINK 实例

打开 QGC，在载具设置“Vehicle Setup”–参数–MAVLink–MAV_0_MODE 选项，将消息格式从 Onboard 改成 Custom，就会切换到我们刚才修改的自定义消息模式。



Step 5：仿真验证

重复实验 5.3，在一台电脑上运行 HITLRun 开启硬件在环仿真，在另一台电脑上，打开 QGC 并连上数传串口，并进入“MAVLink 检测”页面，可以看到目前收到的消息就是我们设置的五条消息，以及 HEARTBEAT（心跳包，强制发送）和 HIL_ACTUATOR_CONTROLS（执行器消息，硬件在环仿真时强制发送）。

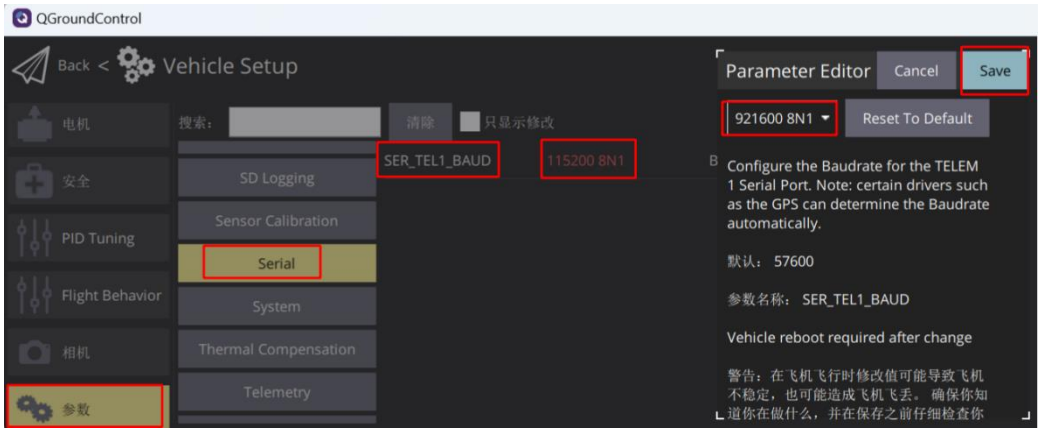


注：硬件在环仿真时 HIL_ACTUATOR_CONTROLS 消息会占用带宽，导致 LOCAL_POSITION_NED 和 ATTITUDE 消息的速率达比预期的 30Hz 和 20Hz 稍小。在真机飞行时，没有 HIL_ACTUATOR_CONTROLS 高频消息的影响，LOCAL_POSITION_NED 和 ATTITUDE 能够达到预期。

5.5. 波特率修改实验（仅限有线连接方式）

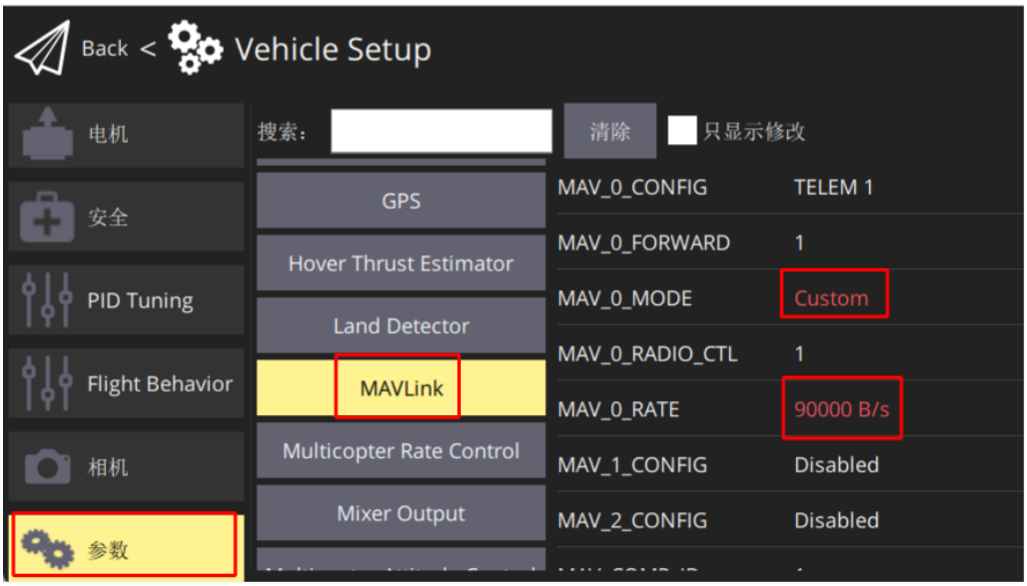
Step 1：修改飞控 TELEM1 口的波特率

打开 QGC，通过 USB（硬件在环仿真口，非数传串口）插入飞控，如下图所示，修改 TELEM1 的波特率为 921600.



Step 2：对应波特率设置传输速率

设置速度 MAV_0_RATE 为 90000（921600/10 再减去一些取整）

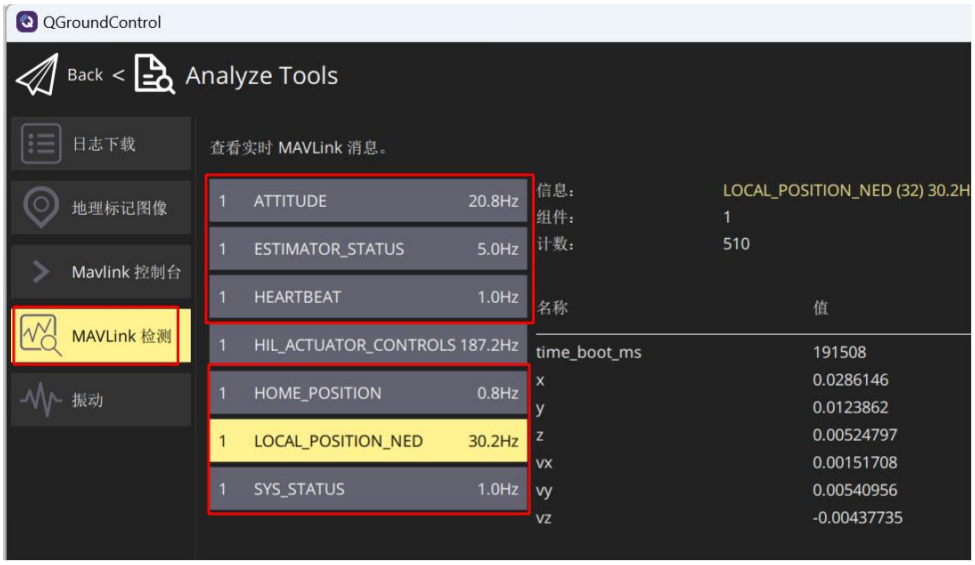


Step 3：修改 QGC 通信接口波特率

重复 5.3 中的实验，运行 HITLRun 开启硬件在环仿真。在 QGC 中，修改数传串口波特率为 921600.

Step 4：数据确认

再去 MAVLink 检测页面，可以看到 ATTITUDE 消息稳定在 20Hz，LOCAL_POSITION_NED 稳定在 30Hz 为我们设定的频率。其他消息也都稳定在对应频率上，说明提升波特率和通信频率，才能够确保数据频率与期望频率保持一致。



Step 5：恢复 QGC 通信配置

恢复 QGC 的自动连接设置。如下图所示，保证“Pixhawk”处于勾选，“SiK 电台”处于未勾选，“UDP”处于勾选状态。

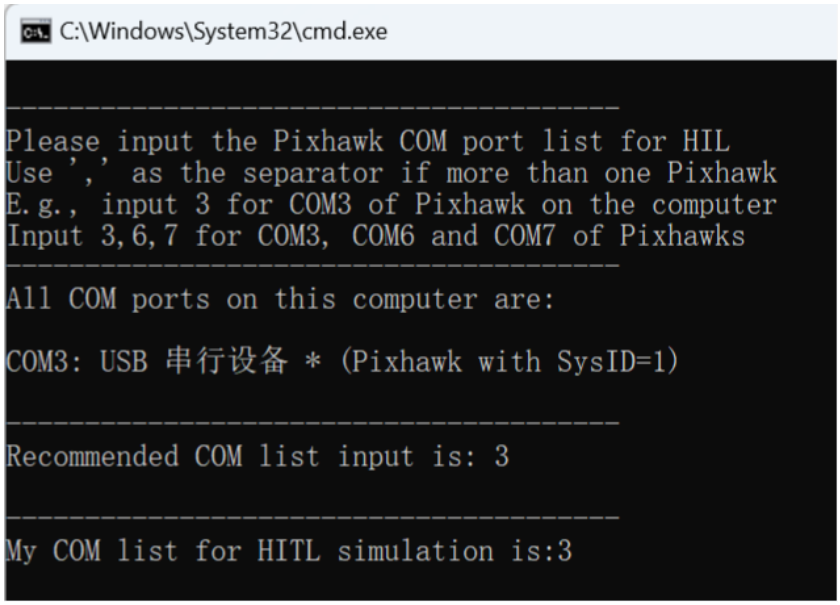
关闭所有硬件在环仿真软件，拔出飞控和数传备用。

5.6. Python 控制无人机仿真实验（波特率必须为 921600，传输速率为 90000，端口号为数传端口）。

注：在进行步实验之前请将飞控状态进行还原，具体可见：Recv_State1\飞控状态还原.pdf

Step 1：启动硬件在环

进入“6.RflySimExtCtrl\0.ApiExps\e2_PX4ComAPITest”文件夹，插入飞控 USB 仿真口，双击“PX4ComAPITest.bat”，输出串口号并回车，开始硬件在环仿真。



Step 2：配置 Python 的通信接口

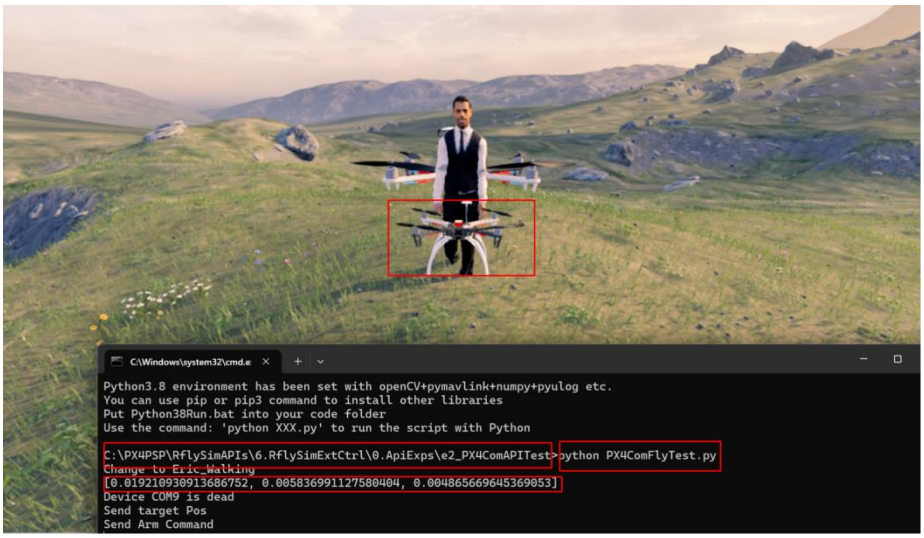
用 VS Code 打开“PX4ComFlyTest.py”，并修改 PX4MavCtrler 函数的串口号和波特率为期望值，例如本例是 COM9+921600。注：COM9 要改成电脑实际串口号。

```
mav = PX4MavCtrl.PX4MavCtrler(1,'127.0.0.1','COM9',921600)
```

```
5   import PX4MavCtrlV4 as PX4MavCtrl
6   import UE4CtrlAPI
7   ue = UE4CtrlAPI.UE4CtrlAPI()
8
9   #Create a new MAVLink communication instance, UDP sending p
10  mav = PX4MavCtrl.PX4MavCtrler(1,'127.0.0.1','COM9',921600)
11
```

Step 3：运行 python 程序通过数传串口进行外部控制

连上数传串口，再双击“Python38Run.bat”在其中输入“python PX4ComFlyTest.py”（或直接从 VS Code 中运行本 Python 程序）。



Step 4：验证结果

可以看到飞机正起飞，且读取到飞控状态数据为非 0 值。注：请忽略“Device COM9 is dead”的警告，为正常现象。

注：本程序也可以将数传串口连到另一台电脑上，再在此电脑运行 Python 程序，模仿真实控制情形。

5.7. Simulink 数传 MAVLink 外部控制飞机仿真实验

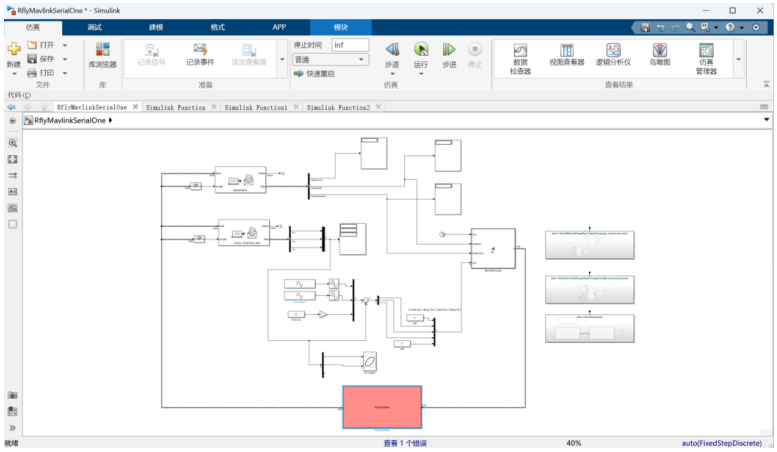
注：在进行步实验之前请将飞控状态进行还原，具体可见：Recv_State1\飞控状态还原.pdf

Step 1：启动硬件在环

进入 6.RflySimExtCtrl\2.AdvExps\e3_MavlinkSerial 文件夹，插入飞控和数传串口，点击 RflyMavlinkSerialOne.bat，自动开始硬件在环仿真。

Step 2：配置 Simulink 的通信接口

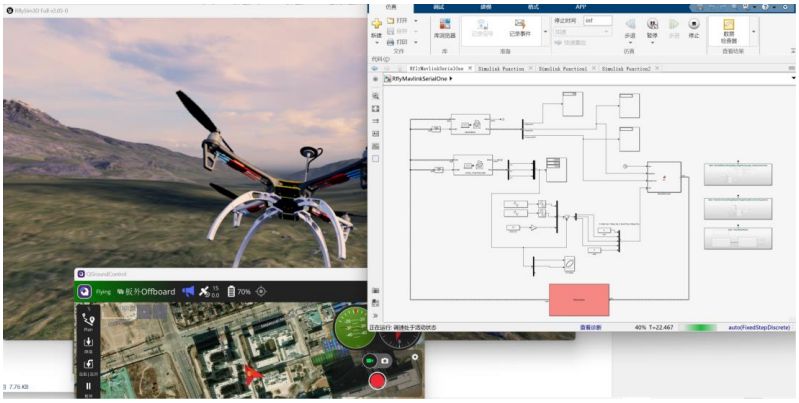
用 MATLAB 打开 RflyMavlinkSerialOne.slx，如下图所示，本例程通过 RflySerialRaw 模块收发 uint8 的字节流 buffer，直接给到无人机工具箱 UAV Toolbox 进行解析，并在其中实现了一个 Offboard 控制逻辑。注意：由于 UAV Toolbox 是在 MATLAB 2022 版本才引入的，因此本例程必须用 MATLAB 2022b 以上版本才能打开。



点击“RflySerialRaw”配置其中的串口和波特率。例如，本例为 COM9 和 921600.

Step 4：运行 simulink 程序通过数传串口进行外部控制

运行 Simulink 程序，可以看到飞机起飞，并绕圈飞行，说明配置正确。



注：本程序也可以将串口连到另一台电脑上，再另一台电脑上运行 Simulink 控制程序，模仿真实控制情形。

6. 参考资料

[1].

7. 常见问题

Q1：***

A1：***