

Communication Raspberry et Flight Controller

Prepared by

Bournousouzis Christos

May 2021

Objectif

L'objectif de ce rapport est de vous introduire au protocole (MAVLink) permettant de créer une communication entre la Raspberry Pi Zéro et un Flight Controller. Ce protocole permettra de configurer le Flight Controller à distance via les lignes de commandes MavLink ou encore via un script tournant sur la Raspberry Pi Zéro.

Introduction au protocole MAVLink

MAVLink (Micro Air Vehicle Link) est un protocole de communication pour des petits véhicules sans pilotes. Ce protocole est principalement utilisé pour faire la communication entre une station de contrôle au sol (Ground Control Station) et un véhicule. Cela peut donc être utilisé pour contrôler le véhicule en position, en vitesse, en orientation, etc. MAVLink peut être également utilisé pour faire tourner des scripts sur un Flight Controller.

Il n'est pas très intéressant pour ce rapport d'aller dans les détails du protocole mais il est tout de même important d'en connaître la base.

MAVLink suit un modèle de conception hybride moderne de publication-abonnement (mode de topic) et point à point: les flux de données sont envoyés et/ou publiés en tant que sujets tandis que les sous-protocoles de configuration tels que le protocole de mission ou le protocole de paramètres sont retransmis en point à point.

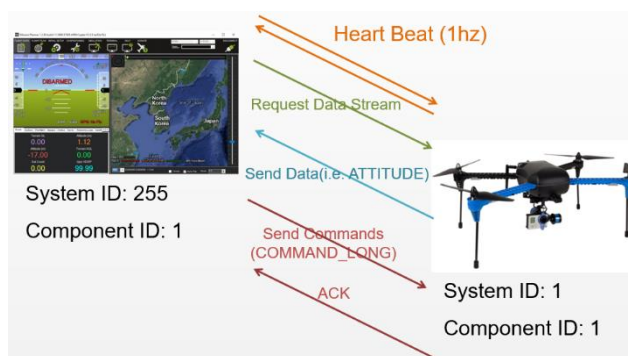


Figure 1 - MAVLink Communication

Tous les langages de programmation ne supportent pas les protocoles et ses différentes versions. Il est donc vivement recommandé de s'informer sur la [compatibilité](#).

Pour en savoir plus dans les détails, n'hésitez pas à aller consulter une documentation plus [détaillée](#).

MAVLink sur la Raspberry Pi Zéro

Le langage de programmation le plus intéressant et le plus facile d'accès sur la Raspberry Pi Zéro reste le Python. Python supporte le MAVLink le plus récent et possède également une librairie (pymavlink) permettant de créer des scripts simples pour analyser les logs de télémétrie à partir de pilotes automatique tel que ArduPilot.

La configuration de MAVLink sur une Raspberry se fait en installant le module MAVProxy. MAVProxy est une station de contrôle entièrement fonctionnel pour les drones, conçu comme une station de contrôle minimaliste, portable et extensible pour tout système autonome prenant en charge le protocole MAVLink (comme celui utilisant ArduPilot). MAVProxy est un puissant software en ligne de commande pour du développement.

L'installation pour un Python 3 se fait comme suit :

```
sudo apt-get install python3-dev python3-opencv python3-wxgtk4.0 python3-pip python3-
matplotlib python3-lxml python3-pygame

pip3 install PyYAML mavproxy --user

echo "export PATH=$PATH:$HOME/.local/bin" >> ~/.bashrc
```

L'installation pour un Python 2 se fait comme suit :

```
sudo apt-get install python-dev python-opencv python-wxgtk4.0 python-pip python-matplotlib
python-lxml python-pygame

pip install PyYAML mavproxy --user

echo "export PATH=$PATH:$HOME/.local/bin" >> ~/.bashrc
```

Etant donné que nous souhaitons communication en UART, il faut activer les ports séries de la Raspberry. Cela se fait via la commande suivante :

```
sudo raspi-config
```

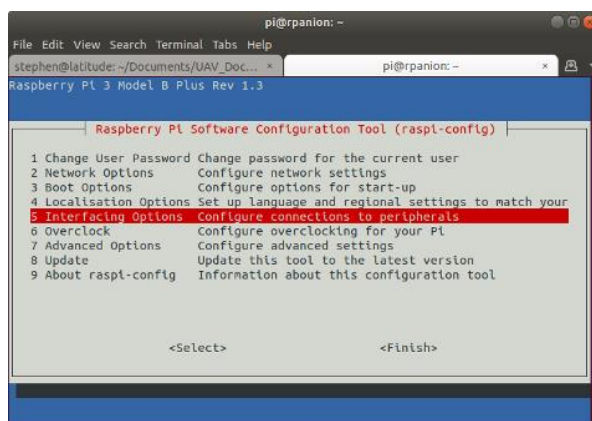


Figure 2 - Screen 1 : Etape 1

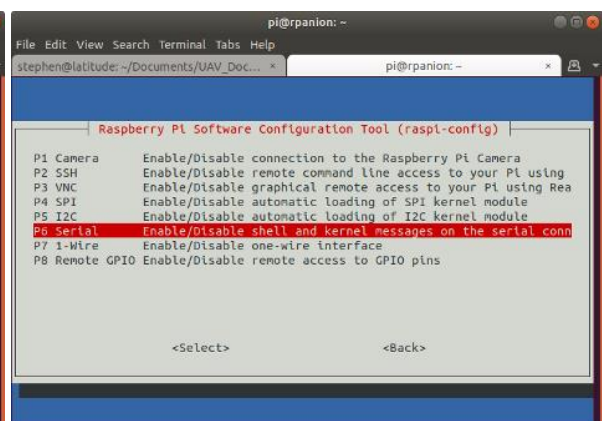


Figure 3 - Screen 2 : Etape 2

La démarche est la suivante :

1. Aller dans les options d'interfaces
2. Sélectionner Série
3. Sélectionner no pour la demande "Would you like a login shell to be accessible over serial?"
4. Sélectionner yes pour la demande "Would you like the serial port hardware to be enabled?"
5. Redémarrer la Raspberry

Pour [information](#), il est tout à fait possible de faire en sorte que la Raspberry puisse communiquer avec le véhicule et l'ordinateur à distance via un télémètre, sans utiliser aucune connexion physique.

MAVLink sur le Flight Controller

Il faut également configurer certains paramètres du Flight Controller pour qu'il puisse accepter les communications séries. Nous allons utiliser dans notre cas le SERIAL0 puisque notre Raspberry est connecté au Flight Controller via USB mais libre à vous de vous connecter en UART par d'autres ports.

Attention que les paramètres qui suivent ne sont valables que pour un Firmware de Ardupilot !

Les paramètres à configurer sont les suivants :

1. SERIAL0_PROTOCOL = 2
 - Active MAVLink 2 sur le port SERIAL0
2. SERIAL0_BAUD = 57600
 - Active un baudrate de 57600 sur le port SERIAL0. Le baudrate est libre de choix, celui-ci n'est qu'un exemple.
3. LOG_BACKEND_TYPE = 3
 - Active les logs pour MAVLink 2 et APSync

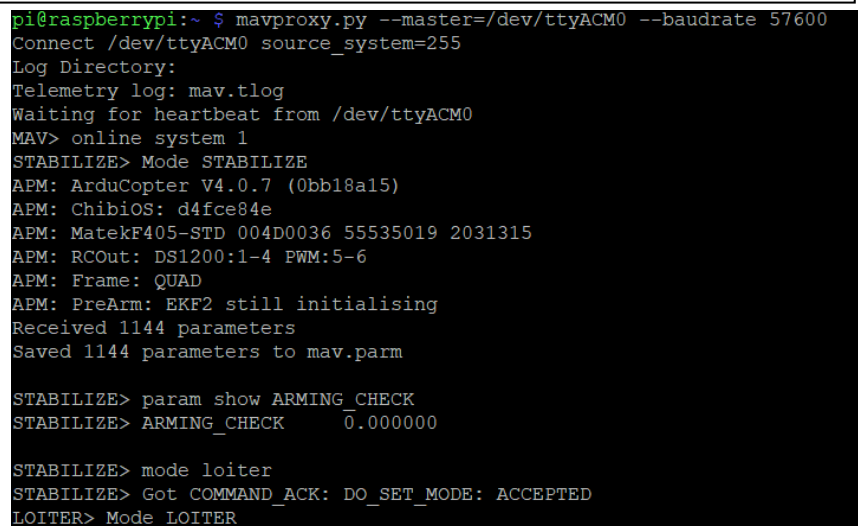
Démonstration

Nous pouvons maintenant réaliser une connexion entre notre Raspberry Pi et notre Flight Controller. Il ne faut pas oublier d'également réaliser la connexion physique permettant la communication série.

Dans un premier temps connectez-vous à la Raspberry pour avoir accès à son terminal.

La commande permettant de lancer le protocole MAVLink sur la Raspberry est la suivante :

```
mavproxy.py --master=/dev/ttyACM0 --baudrate 57600
```



```
pi@raspberrypi:~ $ mavproxy.py --master=/dev/ttyACM0 --baudrate 57600
Connect /dev/ttyACM0 source_system=255
Log Directory:
Telemetry log: mav.tlog
Waiting for heartbeat from /dev/ttyACM0
MAV> online system 1
STABILIZE> Mode STABILIZE
APM: ArduCopter V4.0.7 (0bb18a15)
APM: ChibiOS: d4fce84e
APM: MatekF405-STD 004D0036 55535019 2031315
APM: RCOut: DS1200:1-4 PWM:5-6
APM: Frame: QUAD
APM: PreArm: EKF2 still initialising
Received 1144 parameters
Saved 1144 parameters to mav.parm

STABILIZE> param show ARMING_CHECK
STABILIZE> ARMING_CHECK 0.000000

STABILIZE> mode loiter
STABILIZE> Got COMMAND_ACK: DO_SET_MODE: ACCEPTED
LOITER> Mode LOITER
```

Figure 4 – Communication Basique

On a tout d'abord fait une demande pour nous montrer le paramètre ARMING_CHECK, ce qu'il a fait en nous donnant la valeur 0. On a ensuite demandé de passer en mode Loiter, ce qu'il a également exécuter sans soucis. On observe donc que notre Flight Controller répond bien à nos requêtes.

ttyACM0 correspond au port série, dans notre cas la sortie USB. Il existe plusieurs variantes de tty suivant la connexion série (physique) que vous avez établie. Pour trouver la bonne variante, lancer

dans un premier temps la commande suivante :

```
dmesg | grep tty
```

Cette commande permet de lister quel tty sont actifs dans le microcontrôleur. Une fois que vous avez la liste, n'hésitez pas à vous documentez pour trouver celui qui conviendra à votre configuration physique.

Il est également possible de réaliser une connexion MAVLink entre la Raspberry et le Flight Controller pour contrôler le véhicule via un script Python et avoir un aperçu sur une station de contrôle (Mission Planner dans notre cas).

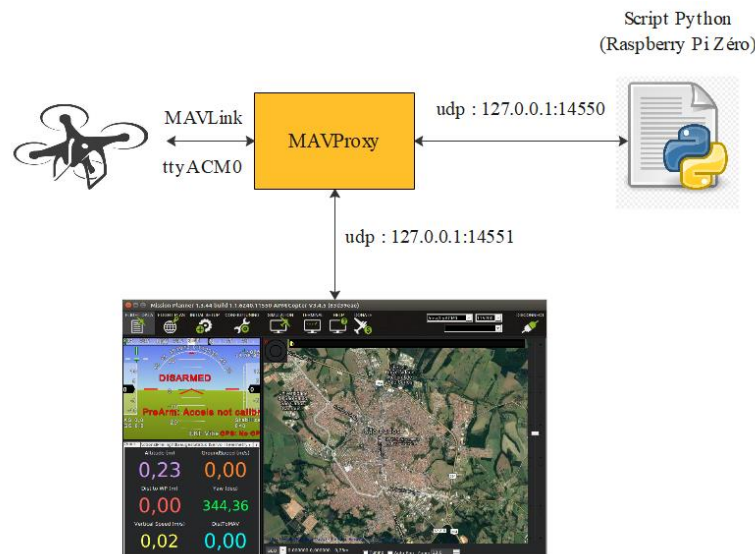


Figure 5 - Schéma de la communication Drone-Script-Station de Contrôle

Pour créer cette communication, il suffit de lancer la commande suivante sur la Raspberry :

```
mavproxy.py --master=/dev/ttyACM0 --baudrate 57600 --out=udp:laptop_ip:14551  
--out=udp:127.0.0.1:14550
```

« laptop_ip » correspond à l'adresse de l'ordinateur sur lequel vous souhaitez avoir la station de contrôle. 127.0.0.1 correspond au localhost (soit la Raspberry dans notre cas) où le script Python se connectera. Pour que cela fonctionne correctement, il faut bien évidemment que la Raspberry et l'ordinateur soient sur le même réseau.

N'hésitez pas à aller consulter la vidéo de démonstration de cette communication.

Références

- <https://mavlink.io/en/>
- <https://ardupilot.org/dev/docs/mavlink-basics.html>
- <https://ardupilot.org/dev/docs/raspberry-pi-via-mavlink.html>
- https://ardupilot.org/mavproxy/docs/getting_started/download_and_installation.html
- <https://ardupilot.org/mavproxy/>