# mavlink

无人机通讯协议

MAVLink 是一种二进制遥测协议，专为资源受限系统和带宽受限链路而设计。 MAVLink 部署在两个主要版本：v1.0 和 v2.0，这是向后兼容的（v2.0 实现可以解析和发送 v1.0 数据包）。遥测数据流在多播设计中发送，而改变系统配置并需要保证交付的协议方面（如任务协议或参数协议）是点对点的重传

# 2 特点

a效率高

MAVLink 是一个非常有效的协议。 MAVLink 1 每个数据包只有 8 个字节的开销，包括开始符号和丢包检测。 MAVLink 2 只有 14 个字节的开销（如果使用签名，则为 27 个字节），但现在是一个可扩展得多的协议。

b支持的设备多

MAVLink 支持多少设备？ 255 ，系统 ID 范围为 1 到 255（0 不是有效的设备 ID）。 注意：严格来说，MAVLink 支持 255 个并发系统，这些系统可以包括车辆、GCS、天线跟踪器和其他硬件的混合。

c支持多种控制器和系统

ARM7, ATMega, dsPic, STM32 and Windows, Linux, MacOS, Android and iOS.

加密和可靠

它提供了检测数据包丢失的方法，以及用于数据包损坏检测的完善的 ITU X.25 校验和

MAVLink 提供消息签名，允许系统验证消息来自受信任的来源。 MAVLink 不提供消息加密。

d V1和V2的区别

它修复了早期版本的一些限制

MAVLink 2 库还支持 MAVLink 1，因此您可以在需要时使用它们与旧系统进行通信。

e支持用户自定义消息类型

f MAVLink 如何检测和解码字节流中的消息？ MAVLink 等待数据包开始符号，然后读取数据包长度并匹配 n 个字节后的校验和。如果校验和匹配，则返回解码的数据包并再次等待开始符号。如果字节被更改或丢失，它将丢弃当前消息并继续下一次尝试以下消息。

系统和组件 ID 有什么用？ 系统 ID 表示特定 MAVLink 系统（车辆、GCS 等）的身份。 MAVLink 最多可同时用于 255 个系统。组件 ID 反映了作为更大系统一部分的组件 - 例如，系统可能包括自动驾驶仪、配套计算机和/或相机，它们可以单独寻址。因此，组件 ID 允许 MAVLink 用于板载和板外通信。

为什么需要 MAVLink 标头中的序列号？ MAVLink 是无人机系统安全关键组件的一部分。丢失许多数据包的不良通信链路可能会危及飞机的飞行安全，因此必须对其进行监控。在报头中包含序列允许 MAVLink 持续提供有关丢包率的反馈，从而允许飞机或地面控制站采取行动。

为什么数据包校验和中需要 CRC\_EXTRA？ CRC\_EXTRA CRC 用于验证发送方和接收方是否对特定消息的无线格式有共同的理解（这是必需的，因为作为轻量级协议，消息结构不包含在有效负载中）。 在 MAVLink 0.9 中没有使用 CRC（尽管有长度检查）。在少数情况下，描述消息的 XML 发生了更改而没有更改消息长度，从而在读取消息时导致字段严重损坏。

如何进一步减小生成的 C 库大小？ 在资源极度受限的系统上，您可以通过设置 MAVLINK\_COMM\_NUM\_BUFFERS=1 和 MAVLINK\_MAX\_PAYLOAD\_LEN="size of your maximum buffer" 来减小生成库的大小（假设只有一个通信链接并且您的有效负载小于MAV 链接）。您还应该确保用于传递给 MAVLink 的任何缓冲区也尽可能小（例如传递给 mavlink\_msg\_to\_send\_buffer() 的缓冲区）。 另一种选择是使用其他生成器之一。例如 fastMavlink 断言它比 mavgen 生成的库更小、更高效（这尚未得到 MAVLink 项目的验证）。

mavlink支持 串口(电脑连接飞控进行校准 设置) udp nrf24l01 等

# 支持不同的语言（C ROS Python。。。。。）

A number of higher level APIs have been written to simplify interacting with MAVLink autopilots, cameras, ground stations, etc. (MAVLink is a relatively low-level API). These wrappers typically provide implementations of the main [microservices](https://mavlink.io/en/services/) and simple/specific interfaces for sending commands and accessing vehicle information. The list here contains actively maintained implementations.

* [MAVSDK](https://mavsdk.mavlink.io/develop/en/) - MAVLink API Library (C++, Python, Swift (iOS), Java, JS) that aims to be fully standards-compliant with MAVLink common microservices.
* [Dronecode Camera Manager](https://camera-manager.dronecode.org/en/) - Adds [Camera Protocol](https://mavlink.io/en/services/camera.html) interface for cameras connected to Linux computer
* [Rosetta Drone](https://www.youtube.com/watch?v=rBqEQoVGuzQ) - MAVLink wrapper around DJI SDK (fly a DJI drone with a Mavlink-based GCS, code: <https://github.com/diux-dev/rosettadrone>).
* [pymavlink](https://github.com/ArduPilot/pymavlink) - MAVLink python bindings.
* [MAVROS](https://github.com/mavlink/mavros) - ROS to MAVLink bridge.
* [DroneKit](http://dronekit.io/) - MAVLink API Library (Python, Android) and Log analysis tool (optimised for ArduPilot).

# ****Microservices**** 高层协议站（满足某种特定功能的命令）

<https://mavlink.io/en/services/>

# 心跳包

用来表示Mavlink系统一个系统的存在，包含系统ID 和组件 ID、车辆类型、飞行堆栈、组件类型和飞行模式。

## 心跳包的作用：

1 发现没有连接到Mavlink网络的系统和接口，如果定期收到一个组件的 HEARTBEAT 消息，则认为该组件已连接到网络，如果未收到许多预期消息（丢包），则认为该组件已断开连接。

2 根据组件类型和其他属性适当地处理来自组件的其他消息（例如，根据车辆类型布局 GCS 界面）。

3 将消息路由到不同接口上的系统。

## 心跳包的广播频率

组件必须经常广播自己的心跳包，并监听来自其他组件或者系统的心跳包。

心跳包发送的频率，以及没有收到多少个心跳包之后认为某个系统已经断开连接，取决于你的频道（不是由Mavlink定义）。无线遥测链接为例，组件通常以 1 Hz 的频率发布它们的心跳，如果没有收到四到五条消息，则认为另一个系统已断开连接。

一个组件可能不会在一个频道发送广播消息（除了心跳包）。如果一个组件收到了一个系统的心跳包，但是并没有检测到这个系统，他会一直给这个系统发送消息。所以对一个系统来说以下是重要的：

1 即使没有命令要发送，也要广播心跳包给远端的系统

2 系统发生故障时不要发送心跳包（不要在一个单独的，无法感知其他组件状态的线程里广播心跳包）

连接到一个GCS或者MAVLink API

GCS（或开发人员 API）也可以使用 HEARTBEAT 来确定它是否可以连接到车辆以收集遥测数据并发送任务/命令。

例如，QGroundControl 将仅连接到车辆系统（即不是另一个 GCS、云台或车载控制器），并且在显示车辆连接消息之前还会检查它是否具有非零系统 ID。 QGC 还使用特定类型的车辆和其他心跳信息来控制 GUI 的布局。

Component Identity组件标识

组件的类型从其 [HEARTBEAT.type](<https://mavlink.io/en/services/heartbeat.html#HEARTBEAT>)([MAV\_TYPE](<https://mavlink.io/en/services/heartbeat.html#MAV\_TYPE>)) 和 [HEARTBEAT.autopilot](<https://mavlink.io/en/services/heartbeat.html#HEARTBEAT>) ([MAV\_AUTOPILOT](<https://mavlink.io/en/services/heartbeat.html#MAV\_AUTOPILOT>)) 字段中获取：

1 飞行控制组件的组件ID必须和机器类型相符（e.g.MAV\_TYPE\_FIXED\_WING,MAV\_TYPE\_QUADROTOR etc）and set HEARTBEAT.autopilot to a valid flight stack.

2 所有组件的MAV\_TYPE必须与真实的情况相符，（e.g.: MAV\_TYPE\_GIMBAL, MAV\_TYPE\_BATTERY , etc.), and should set HEARTBEAT.autopilotto MAV\_AUTOPILOT\_INVALID NOTE：

The recommended way to recognise an autopilot component is to check that HEARTBEAT.type is not MAV\_AUTOPILOT\_INVALID 每个组件都必须有一个系统唯一的组件 ID，用于路由和识别特定组件类型的多个实例。

MAVLink 建议默认情况下组件使用类型适当的组件 id 来自[MAV\_COMPONENT](https://mavlink.io/en/messages/common.html" \l "MAV_COMPONENT),提供了一个接口来改变这个组件ID如果你需要的话。例如，相机组件可能使用任何 MAV\_COMP\_ID\_CAMERA n id，并且不应使用 MAV\_COMP\_ID\_GPS2。

Using type-specific component ids:

makes id clashes less likely "out of the box" (unless two components of the same type are present on the same system).

reduces the impact on legacy code that determines component type from the id.

## 地面端软件（控制飞机，显示图传，接受飞机信息）qgroundcontrol

官网：<https://docs.qgroundcontrol.com/master/en/index.html>

http://qgroundcontrol.com/

常用的地面端软件

