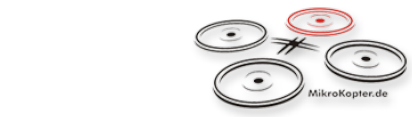
Mikrokopter VSM User Guide

UgCS 2.10.1123



Contents

1 Connecting MikroKopter autopilot to UgCS

1.1 First time vehicle connection

See Disclaimer.

Please follow these steps to connect an MikroKopter vehicle to the U[g]CS:

1. MikroKopter vehicle must be properly configured, calibrated and tested using tools and instruction from the official `MikroKopter web site` prior to using it with U[g]CS. U[g]CS does not support initial configuration, setup and calibration of MikroKopter vehicles.
2. Turn on the vehicle and plug in the radio modem paired with the vehicle or direct USB cable from the MikroKopter board to the computer where VSM is running. U[g]CS uses serial ports for communication with MikroKopter vehicles. Serial interface based communication devices like WI232 radio modems (and their analogs) and direct USB-serial connections are supported, as long as OS driver for virtual serial port is installed and serial port is successfully created. Please refer to your communication equipment manufacturer documentation about driver installation instructions.
3. Open *Vehicles* window in U[g]CS Client and wait until new vehicle appears there automatically. Both Uplink and Downlink connections should be available. Choose the corresponding vehicle for editing by clicking on the menu item and selecting *Edit* button. Now you can select the vehicle profile and change the default vehicle name to be convenient for you:

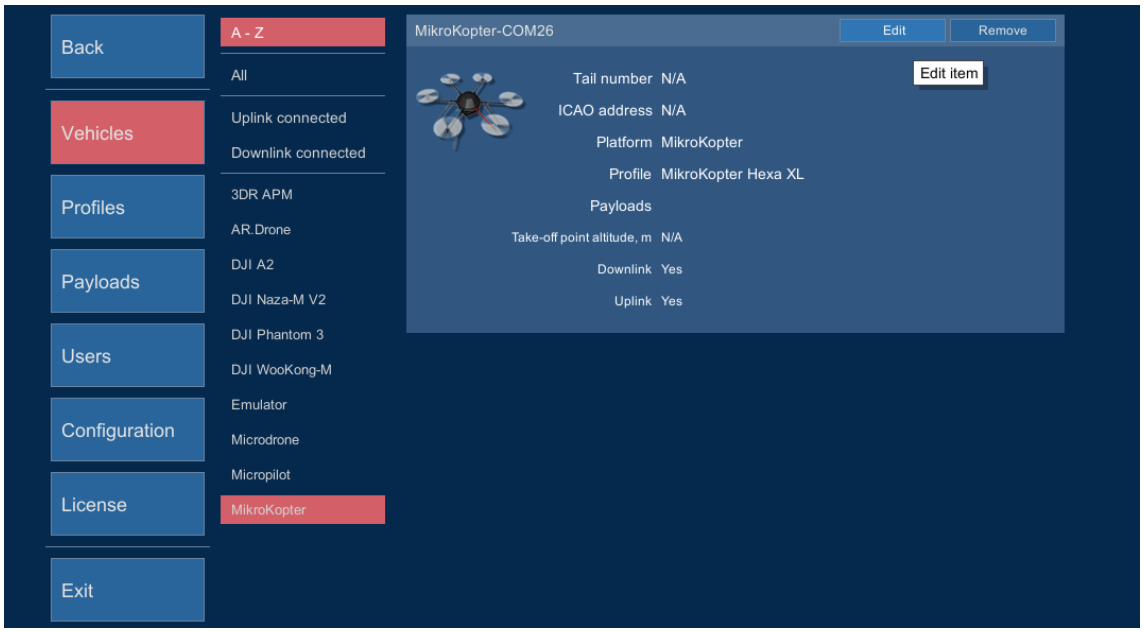


Figure 1: New MikroKopter vehicle

Vehicle profile needs to be assigned to allow mission planning with this vehicle. Vehicle avatar should be assigned in vehicle profile to properly see the vehicle location on map.

4. Repeat steps above for each your MikroKopter vehicle.

Please note that there is no technical possibility to distinguish between different MK drones if they are connected via the same port. The only way to identify drones is looking at their serial port instances. To be sure that you have permanent drones mapping, please ensure that you connect each drone via a dedicated serial port. This firstly means that you cannot share the same radio-modem or USB-serial cable between different drones. If you are using Windows OS it usually reserves unique serial port number for each USB-serial device. On Linux OS you will need to take additional measures - manually provide udev rules for permanent device mapping based on its serial number (if such exists. Some USB devices do not have serial number specified, you will not be able to use them to connect multiple MikroKopters). Udev rules creation example: http://noctis.-de/ramblings/linux/49-howto-fixed-name-for-a-udev-device.html

If you use your own serial device naming on Linux, do not forget to add the corresponding name to the VSM configuration:

```
vehicle.mikrokopter.serial_port.2.name = /dev/mikrokopter_1
vehicle.mikrokopter.serial_port.2.baud.1 = 57600
```

Supported MikroKopter firmware versions:

- 0.x
- 2.00 may be supported but not tested.

1.2 Mission execution specifics

Before executing a mission, Mikrokopter must be set to Altitude hold mode and throttle must be in mid position!

Flight plan element / action	Support	Notes
Camera control	Partial	Only pitch control is supported.
Camera trigger	Partial	Only photo shot is supported.
Panorama	Partial	Panorama is always counted from North or last set heading
Take-off	No	
Landing	No	
Set camera by time	No	
Set camera by distance	Yes	
POI	Yes	
Heading	Yes	
Wait	Yes	

Take-off and landing is not supported in scope of automatic mission execution. You should take-off or land the drone either manually or using auto-start/landing feature of the MikroKopter which can be activated by switch on RC. See MikroKopter vendor documentation.

CareFree feature must be enabled in MikroKopter for all actions which automatically control copter heading (like POI, yaw set etc.) See http://www.mikrokopter.de/ucwiki/en/CareFree

None of fail-safe mode adjustments is supported in the U[g]CS mission parameters (as well as emergency return altitude value). Use MikroKopter original software instead to set up fail-safe parameters.

Home position can not be changed from software. It is controlled by the autopilot hardware and always corresponds to launch position.

1.2.1 Camera trigger action

Can be used to trigger payload-specific action during mission execution. It triggers WP-event in the Mikro-Kopter. See http://www.mikrokopter.de/ucwiki/en/WaypointEvent and http://www.-mikrokopter.de/ucwiki/en/WpEvent for more detailed description of this MK feature. As described there, before you can use this feature you should set up pattern for the flight controller output OUT1 or SRV3

using KopterTool software. You may want to use `shutter cable` or `IR-controller` provided by the drone manufacturer.

When the "take photo" action is fired the waypoint is created with wp-event-channel field set to the value specified in the VSM configuration (see WP-event value section). It is duration in deciseconds of one unit in output signal pattern. Also waypoint duration is set to time which is enough to execute at least one cycle of the pattern. Due to granularity difference in MK protocol the pattern may start the second cycle of execution before the waypoint is finished.

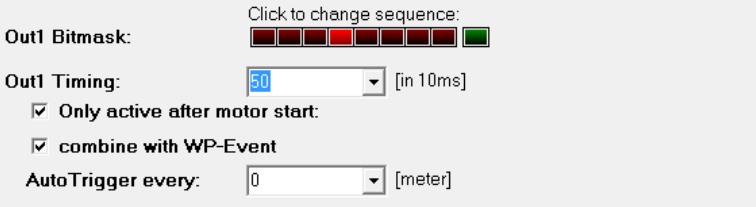


Figure 2: Example of output configuration

In KopterTool output configuration ensure you have set "Combine with WP-event" option.

1.3 Command execution specifics

Command	Support	Notes
ARM	No	
DISARM	No	
AUTOMODE	No	
MANUALMODE	No	
CLICK & GO	No	
JOYSTICK	No	
HOLD	No	
CONTINUE	No	
RETURN HOME	Yes	Current mission is erased in the device. It should be uploaded again in order to run it. Works only when the drone is in AUTO mode.
TAKEOFF	No	
LAND	No	
EMERGENCYLAND	No	
CAMERA_TRIGGER	No	

Use your RC to execute commands and switch between flight modes according to the manufacturer instructions.

1.4 Telemetry information specifics

Nothing specific.

1.5 Fail-safe actions

Fail-safe actions can be set only in KopterTool.

1.6 Configuration file

Default configuration file of the MikroKopter VSM suits most needs and it is generally not necessary to modify it.

Configuration file location:

- **On Microsoft Windows:**

```
C:\Program Files (x86)\UgCS\bin\vsm-mikrokopter.conf
```

- **On GNU/Linux:**

```
/etc/opt/ugcs/vsm-mikrokopter.conf
```

- **On Apple OS X:**

```
/Users/[user name]/Library/Application Support/UGCS/configuration/vsm-mikrokopter.conf
```

1.6.1 WP-event value

This parameter defines which value will be sent in wp-event-value field of the waypoint with "take photo" action. See Camera trigger action section.

```
vehicle.mikrokopter.wp_event_value = 50
```

1.6.2 Common parameters

All VSMs share a common set of configuration file parameters described in Common configuration file parameters. MikroKopter VSM configuration file prefix is:

```
vehicle.mikrokopter
```

1.6.3 Serial port configuration

Mandatory. At least one serial port should be configured, otherwise VSM will not try to connect to the vehicle.

- **Name:** vehicle.mikrokopter.serial_port
- **Description:** Serial port configuration, for more details see Serial port configuration. Default MikroKopter serial port communication speed is 57600 baud.
- **Example:**

```
vehicle.mikrokopter.serial_port.1.name = com1
vehicle.mikrokopter.serial_port.1.baud = 57600
```

1.7 Common configuration file parameters

VSM configuration file is a text file specified via command line argument - *-config* of the VSM application. Example:

```
--config /etc/opt/ugcs/vsm-ardupilot.conf
```

Each configuration parameter is defined as a line in the configuration file with the following structure:

```
name1.name2....nameX = value
```

where name1, name2 ... nameX are arbitrary names separated by dots to divide a variable into logical blocks and a value which can be a number value or a text string depending on the context. See below the description about common VSM configuration parameters.

1.7.1 UgCS server configuration

1.7.1.1 Listening address

Mandatory.

- **Name:** ucs.local_listening_address = [IP address]
- **Description:** Local TCP address to listen for incoming connections from UgCS server. Specify *0.0.0.0* to listen from all local addresses.
- **Example:** ucs.local_listening_address = 0.0.0.0

1.7.1.2 Listening port

Mandatory.

- **Name:** ucs.local_listening_port = [port number]
- **Description:** Local TCP port to listen for incoming connections from UgCS server. Default is 5556.
- **Example:** ucs.local_listening_port = 5556

1.7.2 Logging configuration

1.7.2.1 Level

Optional.

- **Name:** log.level = [error|warning|info|debug]
- **Description:** Logging level.
- **Default:** info
- **Example:** log.level = debug

1.7.2.2 File path

Optional.

- **Name:** log.file_path = [path to a file]
- **Description:** Absolute or relative (to the current directory) path to a logging file. Logging is disabled if logging file is not defined. File should be writable. Backslash should be escaped with a backslash.
- **Example:** log.file = /var/opt/ugcs/log/vsm-ardupilot/vsm-ardupilot.log
- **Example:** log.file = C:\\Users\\John\\AppData\\Local\\UGCS\\logs\\vsm-ardupilot\\vsm-ardupilot.log

1.7.2.3 Maximum single file size

Optional.

- **Name:** log.single_max_size = [size]
- **Description:** Maximum size of a single log file. When maximum size is exceeded, existing file is renamed by adding a time stamp and logging is continued into the empty file. [size] should be defined as a number postfixed by a case insensitive multiplier:

  - Gb, G, Gbyte, Gbytes: for Giga-bytes
  - Mb, M, Mbyte, Mbytes: for Mega-bytes
  - Kb, K, Kbyte, Kbytes: for Kilo-bytes
  - no postfix: for bytes
- **Default:** 100 Mb
- **Example:** log.single_max_size = 500 Mb

1.7.2.4 Maximum number of old log files

Optional.

- **Name:** log.max_file_count = [number]
- **Description:** Log rotation feature. Maximum number of old log files to keep. After reaching single_max_size of current log file VSM will rename it with current time in extension and start new one. VSM will delete older logs so the number of old logs does not exceed the max_file_count.
- **Default:** 1
- **Example:** log.max_file_count = 5

1.7.3 Mission dump path

Optional.

- **Name:** [prefix].mission_dump_path = [path to a file]
- **Description:** File to dump all generated missions to. Timestamp is appended to the name. Delete the entry to disable mission dumping. All directories in the path to a file should be already created.
- **Example:** vehicle.ardupilot.mission_dump_path = C:\\tmp\\ardupilot_dump

1.7.4 Automatic service discovery

VSM can respond to automatic service discovery requests form UgCS server.

When this parameter is not configured, service discovery is disabled.

Optional.

- **Name:** service_discovery.vsm_name = [Service name]
- **Description:** Human readable service name.
- **Example:** service_discovery.vsm_name = Ardupilot VSM

1.8 Communication with vehicle

VSM can communicate with Vehicle over different communication channels

Currently supported channels:

- Serial port, see Serial port configuration for details.
- TCP link, see TCP connection configuration for details.
- UDP link, see UDP connection configuration for details.
- vsm-proxy (XBee), see Proxy configuration for details.

1.8.1 Serial port configuration

Optional. VSM which communicates with vehicles via serial ports should define at least one serial port, otherwise VSM will not try to connect to the vehicles. Port name and baud rate should be both defined. [prefix] is unique for each VSM.

1.8.1.1 Port name

Optional.

- **Name:** [prefix].[port index].name = [regular expression]
- **Description:** Ports which should be used to connect to the vehicles by given VSM. Port names are defined by a [regular expression] which can be used to define just a single port or create a port filtering regular expression. Expression is case insensitive on Windows. [port index] is a arbitrary port indexing name.
- **Example:** vehicle.ardupilot.serial_port.1.name = /dev/ttyUSB[0-9]+|com[0-9]+
- **Example:** vehicle.ardupilot.serial_port.2.name = com42

1.8.1.2 Port baud rate

Optional.

- **Name:** [prefix].[port index].baud.[baud index] = [baud]
- **Description:** Baud rate for port opening. [baud index] is an optional arbitrary name used when it is necessary to open the same serial port using multiple baud rates. [port index] is an arbitrary port indexing name.
- **Example:** vehicle.ardupilot.serial_port.1.baud.1 = 9600
- **Example:** vehicle.ardupilot.serial_port.1.baud.2 = 57600
- **Example:** vehicle.ardupilot.serial_port.2.baud = 38400

1.8.1.3 Excluded port name

Optional.

- **Name:** [prefix].exclude.[exclude index] = [regular expression]
- **Description:** Ports which should not be used for vehicle access by this VSM. Port names are defined by a [regular expression] which can be used to define just a single port or create a port filtering regular expression. Filter is case insensitive on Windows. [exclude index] is a arbitrary indexing name used when more than one exclude names are defined.
- **Example:** vehicle.ardupilot.serial_port.exclude.1 = /dev/ttyS.∗
- **Example:** vehicle.ardupilot.serial_port.exclude = com1

1.8.1.4 Serial port arbiter

Optional.

- **Name:** [prefix].use_serial_arbiter = [yes|no]
- **Description:** Enable (yes) or disable (no) serial port access arbitration between VSMs running on the same machine. It is recommended to have it enabled to avoid situation when multiple VSMs try to open the same port simultaneously.
- **Default:** yes
- **Example:** vehicle.ardupilot.serial_port.use_serial_arbiter = no

1.8.2 TCP connection configuration

Optional. VSM which communicates with vehicles over TCP should define at least one network connection, otherwise VSM will not try to connect to vehicles. [prefix] is unique for each VSM.

1.8.2.1 IP-address for outgoing TCP connection

Optional.

- **Name:** [prefix].detector.[con index].address = [IP-address]
- **Description:** IP-address of vehicle to connect to. Typically used for vehicle simulators.
- **Example:** vehicle.ardupilot.detector.1.address = 10.0.0.111

1.8.2.2 remote TCP port

Optional.

- **Name:** [prefix].detector.[con index].tcp_port = [port number]
- **Description:** Remote port to connect to.
- **Example:** vehicle.ardupilot.detector.1.tcp_port = 5762

1.8.3 UDP connection configuration

Optional. VSM which communicates with vehicles via network should define at least one network connection, otherwise VSM will not try to connect to vehicles. [prefix] is unique for each VSM.

1.8.3.1 Local IP-address for UDP

Optional.

- **Name:** [prefix].detector.[con index].udp_local_address = [IP-address]
- **Description:** Local IP-address to listen for incoming UDP packets on. Specify 0.0.0.0 if you want to listen on all local addresses.
- **Example:** vehicle.ardrone.detector.1.udp_local_address = 0.0.0.0

1.8.3.2 Local UDP port

Optional.

- **Name:** [prefix].detector.[con index].udp_local_port = [port number]
- **Description:** Local UDP port to listen for incoming packets on.
- **Example:** vehicle.ardrone.detector.1.udp_local_port = 14550

1.8.3.3 Remote IP-address for UDP

Optional.

- **Name:** [prefix].detector.[con index].udp_address = [IP-address]
- **Description:** Remote IP-address to send outgoing UDP packets to.
- **Example:** vehicle.ardrone.detector.1.udp_address = 192.168.1.1

1.8.3.4 Remote UDP port

Optional.

- **Name:** [prefix].detector.[con index].udp_port = [port number]
- **Description:** Remote UDP port to send outgoing packets to.
- **Example:** vehicle.ardrone.detector.1.udp_port = 14551

1.8.4 Proxy configuration

Optional. VSM is able to communicate with vehicle via proxy service which redirects dataflow received from vehicle through TCP connection to VSM and vice versa using specific protocol. In other words proxy service appears as a router between vehicle and VSM. At the moment there is one implementation of proxy in UgCS called XBee Connector which retranslates data from ZigBee network to respective VSM.

1.8.4.1 IP-address for proxy

Optional.

- **Name:** [prefix].tcp.[con index].proxy = [IP-address]
- **Description:** IP-address to connect proxy to. Specify local or remote address.
- **Example:** vehicle.ardupilot.tcp.1.proxy = 127.0.0.1

1.8.4.2 TCP port for proxy

Optional.

- **Name:** [prefix].tcp.[con index].port = [port number]
- **Description:** TCP port to be connected with proxy through. Should be the same as in congiguration on proxy side.
- **Example:** vehicle.ardupilot.tcp.1.port = 5566

2 Disclaimer