DroneCAN Setup Guide for use with PX4

Garrett Asper
Virginia Polytechnic Institute and State University, Blacksburg, Virginia 24061

Updated: March 2024

I. Introduction

The term "UAVCAN" has historically referred to a modular, decentralized communication protocol designed for aerospace and robotic applications. However, to clarify and address some of the confusion around its versions, the protocol has been forked into two distinct projects: DroneCAN and OpenCyphal. DroneCAN, previously known as UAVCAN v0, also encompasses elements of UAVCAN v0.5, which did not achieve broad adoption. OpenCyphal, on the other hand, was previously recognized as UAVCAN v1. The choice of software and tooling for a given vehicle depends significantly on which of these two protocols is being implemented.

DroneCAN is compatible with PX4, particularly for GPS peripherals, and is currently supported within the PX4 developer community. OpenCyphal (formerly UAVCAN v1) is seeing a gradual increase in adoption within both ArduPilot and PX4 ecosystems, with expectations of wider support in future releases. For a detailed comparison of DroneCAN and OpenCyphal, refer to Ref. [1]. It is important to note that, throughout this documentation, "OpenCyphal" may be interchangeably referred to as "Cyphal."

This document describes the setup process for the Zubax Myxa ESC, detailing its integration with PX4 v1.14 and the MathWorks® UAV Toolbox for direct RPM command execution on uncrewed aerial vehicles (UAVs). Throughout this tutorial, the Hover Aircraft for Testing Controls and Handling (HATCH) quadcopter, shown in Fig. 1, was used as a hardware testbed.

Fig. 1 The HATCH vehicle at the Kentland experimental aerial systems (KEAS) laboratory.

II. DroneCAN

A. Hardware and Software

The hardware used in this tutorial is outlined in Table 1.

Table 1 Summary of hardware components.

Manufacturer	Model	Component Type
CubePilot	Cube Orange	Autopilot
Zubax	Myxa B1	ESC
Zubax	Babel	DroneCAN Network Sniffer
UCANPHY	Micro patch Cable	Cable
UCANPHY	Micro termination plug	Termination Plug
DJI	2312E 960KV Motor	Motor

Fig. 2 shows a wiring diagram for the DroneCAN network used on HATCH-003.

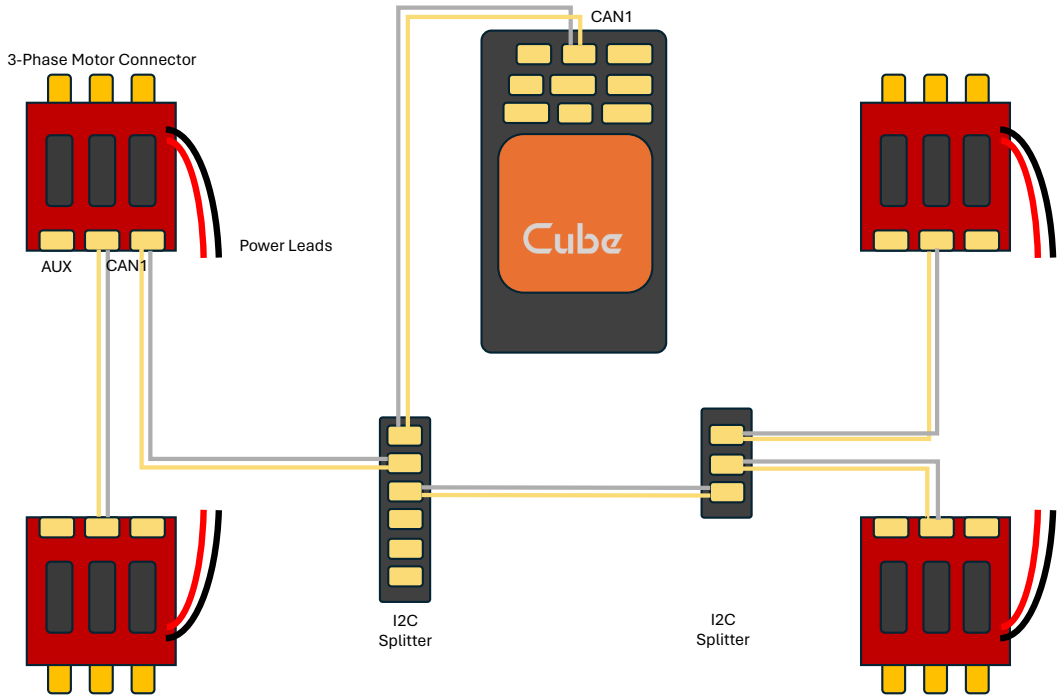

Fig. 2 Wiring diagram of the Zubax Myxa ESCs on the DroneCAN network for HATCH-003.

Table 2 summarizes the essential software utilized in the setup process, along with considerations by use case.

Table 2 Summary of software tools.

Software	Version	Purpose and Notes
Kucher	v1.1.0	Used for motor parameter identification. Interfaces with the ESC over USB [2]. Note: Disable Bluetooth on Windows to prevent freezing.
DroneCAN GUI Tool	v1.2.25	Enables communication with multiple ESCs by accessing all network nodes [3].
QGroundControl	v4.2.8	Interfaces with PX4 to configure DroneCAN node IDs with the PX4 airframe [4].
UAV Toolbox	R2024a	Facilitates the building of a custom flight controller and direct RPM command sending. Optional unless developing a custom flight controller [5, 6].

Finally, the firmware versions used are listed in Table 3.

Table 3 Summary of Firmware Versions Used

Component	Version	Notes
CubePilot Cube Orange	PX4 v1.14	The firmware version used on the autopilot system.
Zubax Myxa ESC	Telega v0.6	Firmware version used for the ESCs, facilitating efficient motor control.

B. Motor Identification

Each motor must be identified before using it with the Zubax Myxa. If the motor being used is present on the sheet shared in Ref. [7], the manufacturer given parameters should be used as they are more accurate. In the case of the DJI motor, motor identification was completed in the lab using Kucher. Kucher is only compatible with Telega v0. If the Zubax Myxa shipped with Telega v1, the COM port will not be accessible. Refer to Appendix A for instructions on rolling the firmware back to Telega v0 before completing motor identification.

The setup procedure below was derived from Ref. [8].

1) Connect the motor to a power supply.
2) Connect the motor to a computer using USB.
3) Open Kucher, select the COM port of the motor and click connect.
4) Define the number of motor poles using the `m.num_poles` parameter.
5) Define the maximum current using the `m.max_current` parameter.
6) Navigate to the "Motor Identification" subpanel and select "Resistance, Inductance, and Flux Linkage."
7) Launch the motor identification process. This step takes between three to five minutes and should be completed on an unloaded motor.
8) After identification is complete, highlight all parameters under the "m" list in the registers panel.
9) Right-click and select the "Read Selected Registers" option.

The user has the option to use the same parameters for each motor on the vehicle or to run this procedure for each motor separately.

C. Motor Enumeration

With all the motors connected together properly, as discussed in Ref. [9], connect the network to the CAN1 port of the Pixhawk. The PX4 parameter `UAVCAN_ENABLE` must be set to three. It is important that the Pixhawk and ESCs are powered as they will be in flight in order to access the network for motor enumeration.

The Zubax Babel-Babel can be used to connect to a pre-existing DroneCAN network by connecting a CAN cable from the Babel 1 port to the network. For the HATCH, this was done using an I2C breakout board.

The following steps were completed to enumerate the Zubax Myxa ESCs for identification with PX4:

1) Power the vehicle including the Pixhawk and the motors.
2) Connect the Babel-Babel to the DroneCAN network and to a computer via USB.
3) Launch the DroneCAN GUI Tool.
4) Select the USB Serial device listed in the drop-down menu, and initialize connection.
5) Click the checkmark next to local node configuration to view information about the devices on the network.
6) Double click the first node and click "Fetch Parameters" to view the list of parameters.
7) Define the parameter for `uavcan.esc_index` as $n-1$ for each ESC on the network where n is the motor number.

After each ESC has been enumerated using the DroneCAN GUI Tool, the Actuator Setup Panel in QGroundControl can be used to define ESC 1 = Motor 1, etc.

D. Parameter Description

Table 4 shows the key parameters defined on the Zubax Myxa ESCs before flight.

Table 4 Configuration Parameters and Descriptions

Parameter	Description	Defined by
`uavcan.node_id`	UAVCAN node ID	Default
`uavcan.esc_index`	Index of ESC on vehicle	User
`uavcan.esc_rcm`	Ratiometric Control Mode	User
`m.num_poles`	Number of magnetic poles on the rotor	User
`m.max_current`	Rated phase current in amperes	User
`m.min_current`	Minimum stable operating current	User
`m.flux_linkage`	Magnetic flux linkage in weber	Motor ID
`m.resistance`	Phase resistance in ohm	Motor ID
`m.induct_direct`	Direct axis phase inductance in henry	Motor ID
`m.induct_quad`	Quadrature axis phase inductance in henry	Motor ID
`m.min_eangvel`	Minimum electrical angular velocity	User
`m.max_eangvel`	Maximum electrical angular velocity	User
`m.max_id_current`	Maximum ID current in amperes	User
`m.min_current`	Minimum stable operating current	User
`m.eangvel_ctl_kp`	P gain of the RPM PID controller	Motor Tuning Spreadsheet
`m.eangvel_ctl_ki`	I gain of the RPM PID controller	Motor Tuning Spreadsheet
`m.eangvel_ctl_kd`	D gain of the RPM PID controller	Motor Tuning Spreadsheet
`bec.can_pwr_on`	BEC power output enable/disable	User
`ctl.spinup_durat`	Spinup duration in seconds	User
`ctl.field_weaken`	Enables or disables field weakening	User

E. Configuring RPM Tracking

Configuring the RPM tracking is a crucial step in ensuring that your UAV operates efficiently and responds accurately to control inputs. This section will outline the determination of minimum and maximum motor RPMs, which are essential for properly configuring the ESCs to track RPM setpoints throughout the flight envelope with its internal control loop. The process involves a series of steps, from conducting initial tests to programming the ESC with the derived parameters.

1. Initial Testing for Maximum RPM

The first step involves conducting a test to determine the maximum RPM the motor can achieve at the minimum voltage per cell at which the UAV will be operated. In this example, the testing was conducted at 3.6 volts per cell. All RPM measurement information was taken from saving the `esc_status` to a ulog file. Once a maximum RPM has been established, select a desired minimum RPM for your system. It is necessary to choose a non-zero minimum RPM, around 500, for example, to avoid stalling the motor and to ensure smooth operation across the entire speed range.

2. Script Use and Parameter Programming

With the maximum and minimum RPM defined, along with the number of motor poles, the next step is to calculate the actuator outputs that correspond to these RPM values. This is where the MATLAB script provided comes into play. By running the script with the applicable parameters, it calculates the minimum and maximum actuator output values that need to be programmed into QGroundControl (QGC). The script below accounts for the ratio between the min/max actuator outputs matching the min/max electrical RPM (eRPM) set on the Zubax Myxa ESCs. This ensures that the linear mapping between commanded and actual RPM aligns as expected.

The script calculates the necessary actuator output values and provides the actual mechanical minimum and maximum RPMs. These values should be programmed into the UAV Toolbox for accurate RPM control outlined in Section II.F.

Moreover, the script calculates the equivalent eRPM values for the minimum and maximum mechanical RPMs. These eRPM values are then used to program the `m.max_eangvel` and `m.min_eangvel` parameters on the Zubax Myxa ESCs. Programming these parameters correctly is essential for the ESC's internal control loop to track the RPM setpoints accurately.

```
%%%%%%%%%%%%%%%%%%%%%%% USER DEFINED PARAMETERS %%%%%%%%%%%%%%%%%%%%%%%
m_RPM_max = 10900; % maximum mechanical RPM from testing at 3.6 v/cell
m_RPM_min_proposed = 500; % initial/proposed minimum RPM (this will change)
p = 12; % number of poles in the motor

%%%%%%%%%%%%%%%%%%%%%%%%%%%% CALCULATIONS %%%%%%%%%%%%%%%%%%%%%%%%%%%%
max_act_output = 8191; % standard maximum actuator_output for CAN ESC

% Calculate initial min_act_output based on proposed minimum RPM
min_act_output_initial = (m_RPM_min_proposed * max_act_output) / m_RPM_max;

% Round min_act_output to nearest integer
min_act_output_rounded = round(min_act_output_initial);

% Calculate the actual minimum RPM based on the rounded min_act_output
actual_m_RPM_min = (min_act_output_rounded * m_RPM_max) / max_act_output;

% Calculate the eRPM for the zubax parameters
eRPM_max = vpa((1/60)*pi*p*m_RPM_max);
eRPM_min = vpa((1/60)*pi*p*actual_m_RPM_min);

%%%%%%%%%%%%%%%%%%%%%%%%%%% DISPLAY RESULTS %%%%%%%%%%%%%%%%%%%%%%%%%%%
disp(' ')
disp(' ************************************************')
disp('Program this into QGC: (where # is ESC 1,2,3, etc.)')
fprintf('UAVCAN_EC_MIN#: %d\n', min_act_output_rounded);
```

```
fprintf('UAVCAN_EC_MAX#: %d\n', max_act_output);
disp(' ')
disp(' ***************************************************')
disp(['Program this into the Simulink diagram as the new minimum RPM ' ...
    'that is being sent: '])
fprintf('Actual Minimum RPM: %.10f\n', actual_m_RPM_min);
fprintf('Actual Maximum RPM: %.10f\n', m_RPM_max);
disp(' ')
disp(' ***************************************************')
disp('Program these parameters on the Zubax Myxa ESCs: ')
fprintf('m.max_eangvel: %.10f\n', eRPM_max);
fprintf('m.min_eangvel: %.10f\n', eRPM_min);
```

F. Commanding Direct RPM with the UAV Toolbox

The UAV Toolbox can be used to deploy a custom flight controller from Simulink® to a PX4 autopilot. Assuming a flight controller has been built, the diagram shown in Fig. 3 shows how commands can be mapped from the UAV Toolbox, sent to the `actuator_motors` uORB topic. These commands are then mapped linearly to the min/max value programmed through QGroundControl to the `actuator_outputs` uORB topic. Finally, these commands are sent as raw DroneCAN commands to the Zubax ESCs, and interpreted linearly between the `m.min_eangvel` and `m.max_eangvel` programmed on the Zubax Myxa ESCs. The blue parameter is programmed as an input to the previously discussed MATLAB script, and the orange parameters are output from the script.

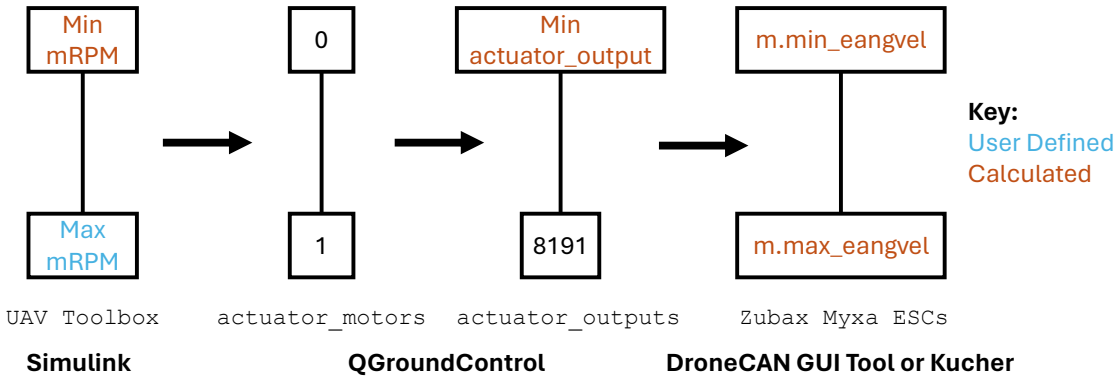

Fig. 3 Diagram of linear mapping to ensure proper RPM commands.

Fig. 4 shows how the RPM commands from the flight controller are mapped to a unit value between 0 and 1 to be sent to the `actuator_motors` uORB topic where `mapRPMtoUnit` is a function that maps the RPM command to a number between 0 to 1. This is necessary for compatibility with the `actuator_motors` uORB topic.

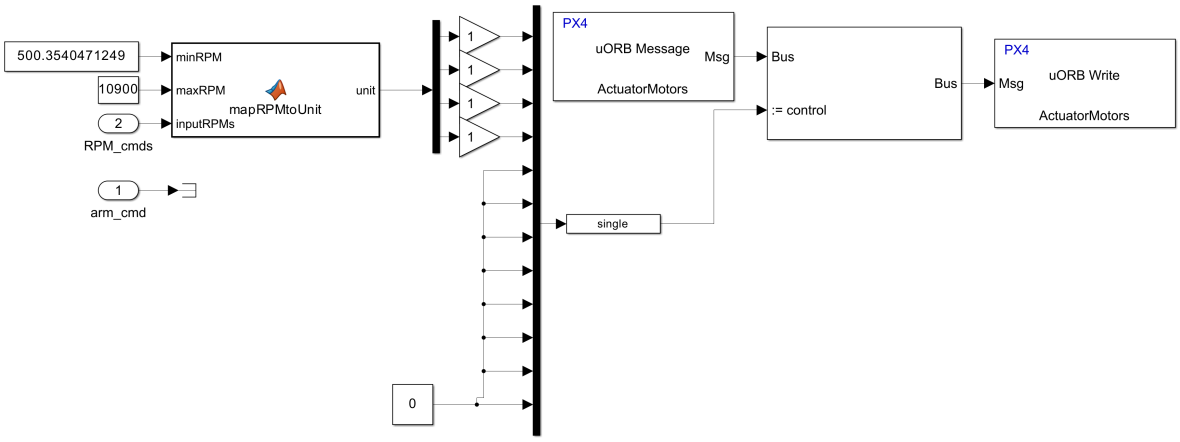

Fig. 4 Simulink® diagram showing direct RPM commands from a flight controller to hardware.

```
function unit = mapRPMtoUnit(minRPM, maxRPM, inputRPMs)
    % Initialize units vector with the same size as inputRPMs
    unit = zeros(size(inputRPMs));

    % Check and map each element in inputRPMs
    for i = 1:length(inputRPMs)
        if inputRPMs(i) < minRPM
            unit(i) = 0;
        elseif inputRPMs(i) > maxRPM
            unit(i) = 1;
        else
            % Map the input RPM to a value between 0 and 1
            unit(i) = (inputRPMs(i) - minRPM) / (maxRPM - minRPM);
        end
    end
end
```

References

[1] "Cyphal vs. DroneCAN," https://forum.opencyphal.org/t/cyphal-vs-dronecan/1814, Accessed 17 March 2024.

[2] "Kucher," https://files.zubax.com/products/com.zubax.kucher/, Accessed 17 March 2024.

[3] "DroneCAN GUI Tool," https://dronecan.github.io/GUI_Tool/Overview/, Accessed 17 March 2024.

[4] "QGroundControl," http://qgroundcontrol.com/, Accessed 17 March 2024.

[5] "UAV Toolbox," https://www.mathworks.com/products/uav.html, Accessed 17 March 2024.

[6] Asper, G. D., and Simmons, B. M., "Rapid Flight Control Law Deployment and Testing Framework for Subscale VTOL Aircraft," Tech. Rep. NASA/TM-20220011570, Sep. 2022.

[7] "PMSM Parameters," https://docs.google.com/spreadsheets/u/1/d/1Is3rCVp-Ibe1MELjCmrP3yA8ozJ6n9Sme-N9w2nRsi0/htmlview?usp=gmail#, Accessed 17 March 2024.

[8] "Telega v0 Quick-Start Guide," https://forum.zubax.com/t/telega-v0-quick-start-guide/2049, Accessed 17 March 2024.

[9] "CAN | PX4 User Guide," https://docs.px4.io/main/en/can/, Accessed 17 March 2024.

[10] "Yukon," https://files.zubax.com/products/org.opencyphal.yukon/releases/, Accessed 18 March 2024.

[11] "Firmware," https://files.zubax.com/products/com.zubax.telega/firmware/, Accessed 18 March 2024.

A. Rolling Back Firmware from Telega v1 to v0

As of March 2024, all Zubax Myxa ESCs ship with Telega v1 unless a special request is sent to have them ship with v0. Rolling the firmware back from v1 to v0 is relatively easy and requires the installation of Yukon v2023.3.45 [10].

The ESC and Pixhawk must be powered to create the DroneCAN network. Next, attach the Babel-Babel to the network, launch Yukon, and navigate to Transports > CAN > SLCAN. Select the COM port of the Babel-Babel, and wait for the ESC node to appear under the "Monitor" menu. Next, download the binary file of the desired Telega v0 release. An application and compound binary version are present on the Zubax firmware repository [11]. For the HATCH, the application version of Telega v0.6 was selected. In Yukon, select the "firmware" button, and navigate to the `com.zubax.telega-1-0.6.cd758ab7.application.bin` file. This update will take several minutes to complete. If successful, the ESC will now appear under the DroneCAN tab of Yukon and can be accessed through Kucher or the DroneCAN GUI Tool. Note that firmware updates can only be conducted over the CAN1 port of the ESC.