



POWERCUBE VR

This short manual explains how to prepare the Cube flight computer for operation in the **PowerCube VR** power supply unit. It also describes the required parameter settings in ArduPilot Mission Planner to enable communication between the Cube flight computer and the **PowerCube VR**.

1. Connect the Cube Pilot to the **PowerCube VR** and secure it using the four screws included in the package.
2. Power up the system using at least one battery. The power input supports a voltage range from 6V to 35V.
3. Turn on the system by pressing and holding the SET button for one second. When the red LED lights up, press buttons I and II while continuing to hold the SET button.
4. Connect a USB cable between the Cube flight controller (FC) and your computer, then click **CONNECT** in the Mission Planner.
5. Open the **Full Parameter List** and change the settings in the recommended sequence:

a) **BATT and BATT2 Parameters**

Command: **BATT_MONITOR** - Option: **DroneCan-BatteryInfo**

Komando	Δ	Wert	Default	Einheiten	Optionen	Desc
BATT_MONITOR		8	0		DroneCAN-BatteryInfo	Controls enabling monitoring of the battery's voltage and current
Disabled Analog Voltage Only Analog Voltage and Current Solo Bebop SMBus-Generic DroneCAN-BatteryInfo ESC						

Command: **BATT2_MONITOR** - Option: **DroneCan-BatteryInfo**

Kommando	Δ	Wert	Default	Einheiten	Optionen	Desc
BATT2_MONITOR		8	0		DroneCAN-BatteryInfo	Controls enabling monitoring of the battery's voltage and current
					Disabled	
					Analog Voltage Only	
					Analog Voltage and Current	
					Solo	
					Bebop	
					SMBus-Generic	
					DroneCAN-BatteryInfo	
					ESC	

b) CAN Parameter

Command: **CAN_P1_DRIVER** - Option: **First driver**

CAN_P1_DRIVER	1	0		First driver	Enabling this option enables use of CAN buses.
				Disabled	
CAN_P2_DRIVER	0	0		First driver	Enabling this option enables use of CAN buses.
				Second driver	
				Third driver	
CAN_SLCAN_CPORT	0	0		0:Disabled	CAN Interface ID to be routed to SLCAN, 0 means no routing
				1:First interface	
				2:Second interface	

c) BRD Parameter

Command: **BRD_SAFETY_DEFLT** – Option **0: Disabled**


Command: **BRD_SAFETYOPTION** – Option **0**

Command: **BRD_SBUS_OUT** – Option **3: 100Hz**

BRD_IO_ENABLE			0:Disabled	This sets the IOMCU for the IOMCU to be enabled. Setting to 2 will enable the IOMCU but not attempt to update firmware on startup	<input type="checkbox"/>
BRD_OPTIONS				Options for safety button behavior	<input type="checkbox"/>
BRD_PWM_VOLT_SEL				This controls the activation of the safety button. It allows you to control if the safety button can be used for safety enable and/or disable, and whether the button is only active when disarmed	<input type="checkbox"/>
BRD_RTC_TYPES				<input type="checkbox"/> ActiveForSafetyDisable <input type="checkbox"/> ActiveForSafetyEnable <input type="checkbox"/> ActiveWhenArmed <input type="checkbox"/> Force safety on when the aircraft disarms	<input type="checkbox"/>
BRD_RTC_TZ_MIN					<input type="checkbox"/>
BRD_SAFETY_DEFLT	0	1	0:Disabled 1:Enabled	This controls the default state of the safety switch at startup. When set to 1 the safety switch will start in the safe state (flashing) at boot. When set to zero the safety switch will start in the unsafe state (solid) at startup. Note that if a safety switch is fitted the user can still control the safety state after startup using the switch. The safety state can also be controlled in software	<input type="checkbox"/>
BRD_SAFETY_MASK	0	0		A bitmask which controls what outputs can move while the safety switch has not been pressed	<input type="checkbox"/>
BRD_SAFETYOPTION	0	3	Set Bitmask	This controls the activation of the safety button. It allows you to control if the safety button can be used for safety enable and/or disable, and whether the button is only active when disarmed	<input type="checkbox"/>
BRD_SBUS_OUT	3	0	0:Disabled 1:50Hz 2:75Hz 3:100Hz	This sets the SBUS output frame rate in Hz	<input type="checkbox"/>

d) Write parameters

After writing the parameters, the Cube will restart and more options are available for the **BRD** and **CAN** parameters.

 Write Raw Params

Are you Sure?

☒ Show me again?

OK

e) BATT and BATT2 Parameters

Command: **BATT_SERIAL_NUM** - Option: **0**

BATT_MONITOR	8	0		0:Disabled 3:Analog Voltage Only 4:Analog Voltage and Current	Controls enabling monitoring of the battery's voltage and current
BATT_OPTIONS	0	0			This sets options to change the behaviour of the battery monitor
BATT_SERIAL_NUM	0	-1			Battery serial number, automatically filled in for SMBus batteries, otherwise will be -1. With DroneCan it is the battery_id.

Command: **BATT2_SERIAL_NUM** - Option: **1**

BATT2_MONITOR	8	0		0:Disabled 3:Analog Voltage Only 4:Analog Voltage and Current	Controls enabling monitoring of the battery's voltage and current
BATT2_OPTIONS	0	0			This sets options to change the behaviour of the battery monitor
BATT2_SERIAL_NUM	1	-1			Battery serial number, automatically filled in for SMBus batteries, otherwise will be -1. With DroneCan it is the battery_id.

f) CAN Parameters

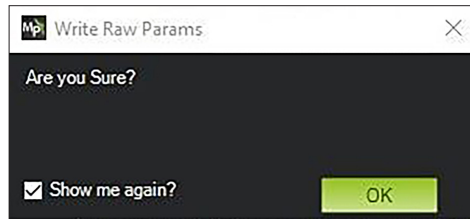
Command: **CAN_P1_FDBITRATE** – Option **1: 1M**

CAN_P1_DRIVER	1	0		0:Disabled 1:First driver 2:Second driver	Enabling this option enables use of CAN buses.
CAN_P1_FDBITRATE	1	8		1M 1M 2M 4M 5M	Bit rate can be set up to from 1000000 to 8000000
CAN_P2_DRIVER	0	0			Enabling this option enables use of CAN buses.

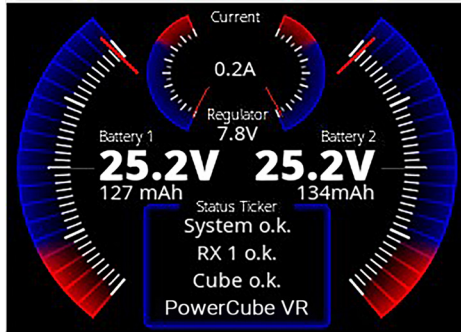
Command: **CAN_LOGLEVEL** – Option 1: **Log Error**

CAN_LOGLEVEL	1	0	Log Error	Loglevel for recording initialisation and debug information from CAN Interface
			Log None	
			Log Error	
CAN_P1_BITRATE	1000000	1000000	Log Warning and below	Bit rate can be set up to from 10000 to 1000000
			Log Info and below	
			Log Everything	
CAN_P1_DRIVER	1	0	1:First driver	Enabling this option enables use of CAN buses.
			2:Second driver	

g) Safe settings and restart the system



When the Cube has booted up again you will see the status message: **Cube o.k.** in the PowerBox monitor.
This means that the servo data from the Cube is correctly received in the **PowerCube VR**.



In the Mission Planner, the battery data from the **PowerCube VR** is found in the left lower corner.



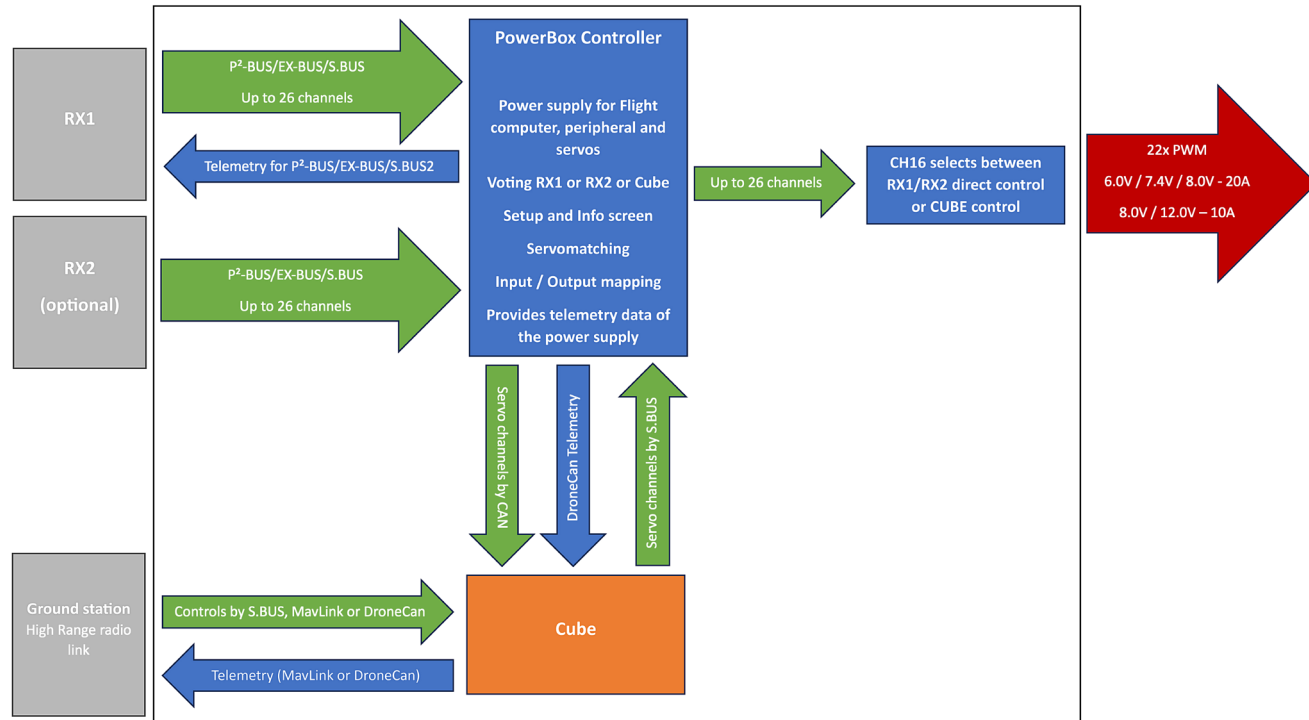
Multiple input architectures are supported for commanding the aircraft. **RX1** and **RX2** serve as the primary redundant control inputs when operating with **P² BUS**, **EX BUS**, or **S.BUS protocols**. The **PowerCube VR** performs input arbitration by continuously comparing both receiver streams and forwarding the validated command signal either to the Cube Flight Controller (FC) or directly to the servo output stage.

The selection between FC controlled mode and direct receiver passthrough is assigned to a dedicated switching channel; **by default, this is channel 16**, but it can be reassigned as required by the integration.

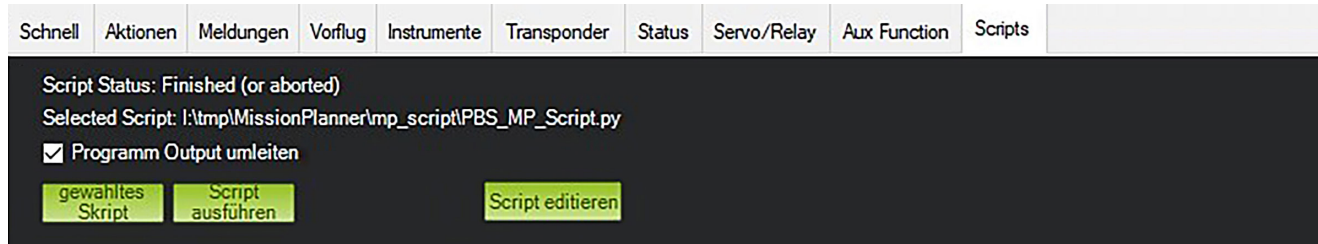
In addition to manual mode switching, the system incorporates automatic failover logic. If the Cube FC becomes unavailable or fails, control authority is immediately transferred to one of the two receivers. Conversely, if both receiver inputs are lost or invalid, the Cube FC assumes control and maintains command output based on its configured fallback behavior.

The system can also be commanded via **MavLink** or **DroneCAN** interfaces connected directly to the Cube, providing an alternative control path when RX1/RX2 inputs are not used.

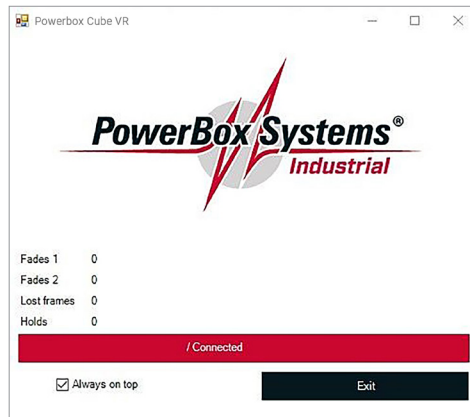
You can refer to the **PowerCube VR's internal signal path architecture** to analyze how control inputs are acquired, validated, prioritized, and routed through the system:



For detailed receiver data like Fades, Lost frames and Holds you can use our Script. Open the Scripts tab:




Select the Script from our download section **PBS_MP_Script** and execute it. You will see following screen:




For technical questions you can contact us here:
industrialsupport@powerbox-systems.com

PowerBox-Systems GmbH

Dr.-Friedrich-Drechsler-Straße 35
86609 Donauwörth
Germany

 +49 906 99999-200

 sales@powerbox-systems.com

www.powerbox-systems.com