

RCSIM MCS — illustrated user manual

Illustrated user guide: tabs, controls, settings and step-by-step procedures. The screenshots show the English interface.

Contents

1. What MCS does and how to read the illustrations	4
2. First start — from access to channel testing	4
Access from a phone or computer	4
Recommended preparation sequence	5
3. Operator workspace — header, profile and arming	5
3.1. What to check in the header	5
3.2. Selecting a profile and confirming it was saved	6
3.3. Arming and ending a test	6
4. Live Monitor — inputs, channels and driving modes	7
4.1. Output RC Channels	7
4.2. Physical devices and Matrix diagnostics	7
4.3. Test Signal Source and virtual sticks	8
4.4. Sport, Eco, Crawler and Wet	8
4.5. Cruise control	8
5. HUD / Cockpit — readable instruments	9
5.1. Operating the view	9
5.2. Understanding readings	11
6. Navigation & Telemetry — map, sensors and diagnosis	12
6.1. Telemetry header and sensors	12
6.2. Layers and map movement	12
6.3. Downloading offline maps	13
6.4. FFB monitor and effect settings	14
6.5. Radio, autopilot mode and MAVLink Forward	14
6.6. Logs and recognising demonstration data	15
7. Profiles — creation, selection, and persistent saving	15
Creating and copying	16
Selecting a profile and confirming the save	17
8. Control mapping, channel definitions, and calibration	17

Mapping table and Matrix diagnostics	17
Device assignment and automatic detection	18
Channel type — choose behavior before tuning	18
Raw input calibration	20
Reverse, Subtrim, Expo, and dead zone	20
Input failsafe	21
First setup: wheel and two pedals	21
9. Wheel, FFB, gearbox, and special buttons	22
Device settings in Channel Definitions	22
Telemetry FFB — separate from profile Autocenter	22
Virtual gearbox	23
Drive Mode, Cruise, and Arm	23
10. Hardware — RC output, port and station power	24
Output Protocol	24
UART Serial Port	25
Eight or sixteen channels?	25
UPS Sensor Type and I2C Address	25
11. Bluetooth — pairing a controller with MCS	26
12. Wi-Fi — client mode, hotspot and returning to the panel	27
Selecting a mode	27
Fields and change procedure	27
13. Coprocessor — compiling, flashing and logs	28
Selecting the target and file	29
Compiling RP2350 or STM32 firmware	29
Flashing RP2350 and STM32	29
Flashing ESP32 — preparation and RESET	29
Progress, console and UART monitor	30
14. Troubleshooting and sources	30

1. What MCS does and how to read the illustrations

MCS (Mobile Control Station) uses a Raspberry Pi as the operator station: it reads local USB/Bluetooth controllers, processes axes and buttons, and sends channels to a bridge or transmitter. A phone, tablet or computer opens the browser panel. Connect the controller to the MCS host; connecting it only to a phone does not automatically transmit its axes to MCS.

MCS and GCS are operator-station applications. Running MCS on a Raspberry Pi does not make that Pi the vehicle's Tier 4 autonomy computer. Choose a protocol that matches the actual bridge, transmitter and firmware, rather than relying on a Tier number alone.

Project classification:

- Tier 0: Sound Card PPM.
- Tier 1: Arduino/RP2350 Digital to PPM.
- Tier 2: ESP32-CAM.
- Tier 3: ELRS/mLRS, CRSF/MAVLink and EdgeTX/ExpressLRS modifications, such as ER5C V2.
- Tier 4: Raspberry Pi Robotic/Autonomous 5G Platform.

Illustration provenance. Screenshots were captured on 14 October 2026 from the locally running MCS source frontend, with English selected and no connected Raspberry Pi backend, controller or vehicle. They show the actual application interface, not generated mockups. Device names, profiles, voltages, RSSI, GPS position, logs and arming state in the illustrations may be examples or simulated. They are neither hardware test results nor ready-to-use settings for your model. Screenshots retain their original resolution; zoom into the PDF to read small fields.

This manual explains current behaviour and limitations checked in the code. No flights, drives, physical failsafe tests, firmware flashing or latency measurements were performed. A control's description does not guarantee support for the selected function on every device.

2. First start — from access to channel testing

Access from a phone or computer

Start the prepared MCS host and connect your phone to the same network or the station's hotspot. Open `http://MCS_ADDRESS:8080`. MCS_ADDRESS is the Raspberry Pi's actual address; 0.0.0.0 is the service's listening address, not an address to enter on your phone. Wi-Fi access alone does not prove that the MCS server is running.

Hotspot names and addresses depend on the deployment. The panel defaults show RCSIM-MCS and 192.168.4.1, while the AP installation script may use RCSIM_MCS and 10.42.0.1. Check the network actually running on your host. Keep port 8080 in the panel URL; the shortened address without a port in the Wi-Fi hint may not open the service.

If MCS is not installed yet, the repository's installation instructions target Linux on Raspberry Pi. From the module directory, `sudo bash scripts/install_services.sh` installs `usb_rc.service` under `/opt/usb_rc_converter`. Do not run the installation while operating a model. A particular system image may contain a different panel version from the sources used for this publication.

Recommended preparation sequence

1. Secure the drivetrain and keep the model disarmed. Connect the controller to the Raspberry Pi.
2. Open the panel. Check host access and distinguish its status from vehicle telemetry.
3. In Profile Editor, create or clone a profile. Check the battery and intended channel order.
4. In Channel Definitions, select each required function's type. In Control Mapping, assign the device and axis/button.
5. Calibrate physical axis endpoints and neutral position. Check direction, EPA, sub-trim and failsafe, then adjust dead zones and expo.
6. In Hardware Settings, select the correct protocol and physical port. Settings must match the transmitter or bridge firmware.
7. Wait for save confirmation, reload the page, and check the active profile and settings.
8. In Live Monitor, move each input separately. Compare the raw device reading, calculated channels and receiver response.
9. If using FFB, read chapter 9 first. The MCS test uses a different force level from the short GCS test.
10. Before driving, check the complete path's response to input loss, radio-link loss and station power loss. Account for the MAVLink disarming limitation explained in chapter 3.

If the page does not open: check Pi power, the phone's network, the actual address, port 8080 and the service. The host administrator can use `hostname -I, systemctl status usb_rc.service --no-pager` and `journalctl -u usb_rc.service -n 80 --no-pager`. Do not run a second application instance on the same serial port.

3. Operator workspace — header, profile and arming

The header remains your reference while working. Distinguish access to **MCS Host**, **RC Signal** status, and the actual receiver and vehicle state. A browser connection to the station does not prove the radio works or the vehicle controller is armed.

3.1. What to check in the header

- **Model** shows the profile selected in the browser and its pack description. Check it before every test: another model's profile may have different channel roles and throttle neutral.
- **MCS Host — Connected/Disconnected** indicates station data availability. Its button changes local interface state; it is not a command to disconnect the transmitter, disable the drivetrain or stop MCS. The next status refresh may show it connected again.
- **RC Signal — ARMED/DISARM** toggles an arming/disarming request. “DISARM” may also appear while the local state is disarmed; do not judge the vehicle solely by the word on the button.
- **Station/UPS power** concerns the MCS station. I2C readings, USB 5 V power and demonstration mode are not measurements of the same pack as the vehicle battery in telemetry.
- **RSSI/LQ** concern radio link data when available. A previously displayed reading does not confirm a current link.
- **PL/EN** changes interface language; the sun/moon icon switches light/dark appearance. The theme is remembered in this browser.
- **HUD Mode** opens the instrument view described in section 5.

Example: MCS Host is connected but navigation shows offline. The station responds to the browser, but vehicle telemetry is unconfirmed. Check the radio path and sensors instead of arming the model again.

3.2. Selecting a profile and confirming it was saved

1. Secure the drivetrain and open **Profile Editor**.
2. Select the correct profile card. On a narrow screen, you can also choose the profile from the header list.
3. Check **Model**, pack type, and steering, throttle, brake and arm-switch mappings.
4. Clicking a profile card or the header list alone changes the browser selection. Do not treat an “Active” highlight as proof that the station has saved the selection.
5. Complete the intended mapping configuration in the editor: mapping parameter changes are saved automatically together with the profile selection. Do not look for a separate “Save mapping” button if your view has none. Check the operation message, then refresh the page and confirm the correct profile remains selected.
6. In Live Monitor, check physical axes and their corresponding channels. A changed Model name alone does not confirm the active control path.

If the previous profile returns, do not continue driving. Repeat saving and the refresh check; check host availability. Do not use language changes as a way to save a profile, even though they may also trigger a settings save.

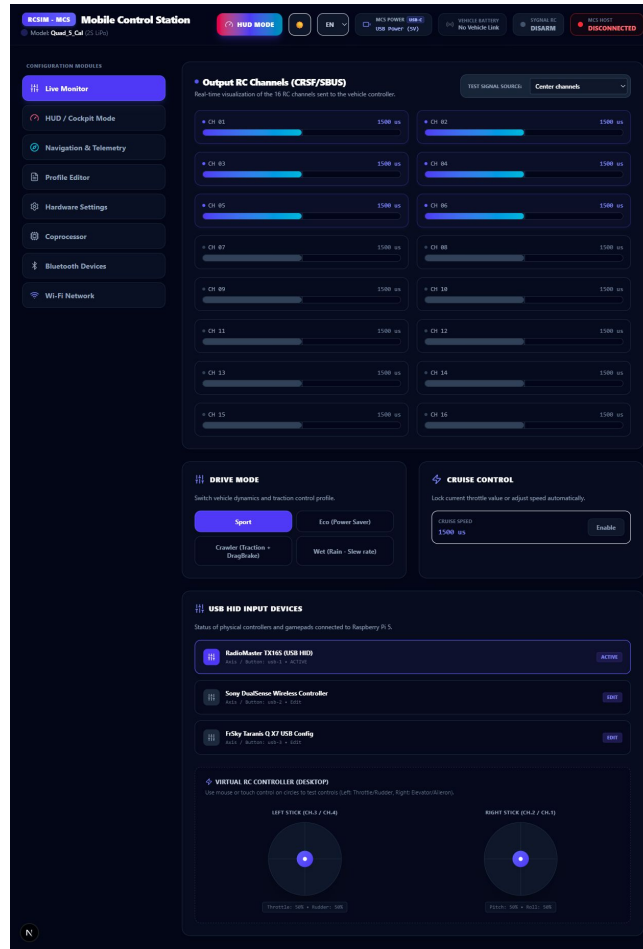
3.3. Arming and ending a test

1. Ensure the profile, protocol and output device match the model. Check neutral throttle with released pedals and that cruise control is off.
2. With the drivetrain secured, check the **RC Signal** button's effect on actual receiver outputs. Repeat for disarming.
3. If the path uses MAVLink and a vehicle controller, check the controller's own confirmed state. A colour change on the MCS button or a station response does not confirm the vehicle executed ARM/DISARM.
4. Only after this test, arm for a short low-speed drive. Do not change the profile, port or throttle mapping during it.
5. To finish, first stop the model and disable cruise control. Disarm and check the actual output state, then secure the drivetrain.

The meaning of disarming depends on the output. Some outputs pause transmission and close the port, direct CRSF may continue neutral frames, and the MAVLink path may keep sending RC overrides despite a local pause while separately sending a DISARM request. Neutral 1500 μ s is not a safe STOP for every ESC.

Emergency stopping: prepare and physically test the stop method appropriate to your receiver, ESC or autopilot, including a way to cut drivetrain power. A global Space shortcut stopping the vehicle was not confirmed in the checked interface; do not base a test on that assumption. The MCS Host toggle, closing the tab, leaving HUD or losing browser Wi-Fi are not emergency cutoff commands.

4. Live Monitor — inputs, channels and driving modes



MCS Live Monitor — RC channels, driving modes and input devices

Use this screen to check the full sequence: physical input → active profile mapping → channel. A channel preview is the station's calculated state, not a measurement of the pulse at the receiver connector.

4.1. Output RC Channels

Each **CH 01–CH 16** bar shows a value in μs . Highlighting identifies a channel assigned in the profile. The middle of the bar corresponds to around 1500 μs , not a universally safe position for every model.

1. Read the values with controls released.
2. Move steering, then throttle and brake separately. Note which channels respond.
3. Check both range endpoints and return to neutral. If the direction is wrong, correct the mapping/Reverse rather than compensating with another driving mode.
4. Check auxiliary channels and the arm button with the drivetrain secured.
5. Compare the result with the transmitter monitor or actual receiver. Defining 16 bars does not mean every protocol supports 16 channels; the current MAVLink path sends the first 8 RC override channels.

4.2. Physical devices and Matrix diagnostics

Select a controller in **USB HID Devices**. “Active” in this list identifies the diagnostic preview source, not an automatic assignment of all axes to the model.

In **Controller Diagnostics (Matrix)**, watch the axis name, code, raw value and Min/Max. Move one control and identify its axis. Then press a button and observe the highlighted grid entry.

The grid displays 32 positions corresponding to a particular button-code group. An unlit indicator does not prove that any button outside that group is unavailable; also check it through assignment in the editor. Real diagnostics require a device supplied by the station, not an example demonstration entry.

Example: a raw axis changes but CH1 stays still. The controller connection works; check the profile, output channel and axis assignment. If raw data is already missing, check the cable, Bluetooth pairing and device detection on MCS.

4.3. Test Signal Source and virtual sticks

The list contains **Center channels**, **Sinusoidal (Auto-movement)**, **Sawtooth (Linear)** and **Noise (Random/Turbulence)**. The left virtual stick is labelled Throttle/Rudder and the right Aileron/Elevator.

In the checked version, these controls feed the display simulation when the host is disconnected. They are not a remote joystick sending commands to the running MCS. With active host data, changing them does not replace the physical controller.

Demonstration test: keep the actual model secured, observe host status, select sine or sawtooth and compare bar movements in offline view. Select centring and move a virtual stick. A stick returning in the browser does not confirm the actual model stopped. Do not disconnect the host as a way to switch to touchscreen driving while moving.

4.4. Sport, Eco, Crawler and Wet

Drive Mode buttons select Sport, Eco, Crawler or Wet. These are MCS control-processing settings; they are not automatically autopilot modes.

In the supported input path, Eco limits positive throttle demand, Crawler reduces it further and may add behaviour near neutral, and Wet softens throttle. The name Wet does not mean tyre grip is measured or guarantee traction control.

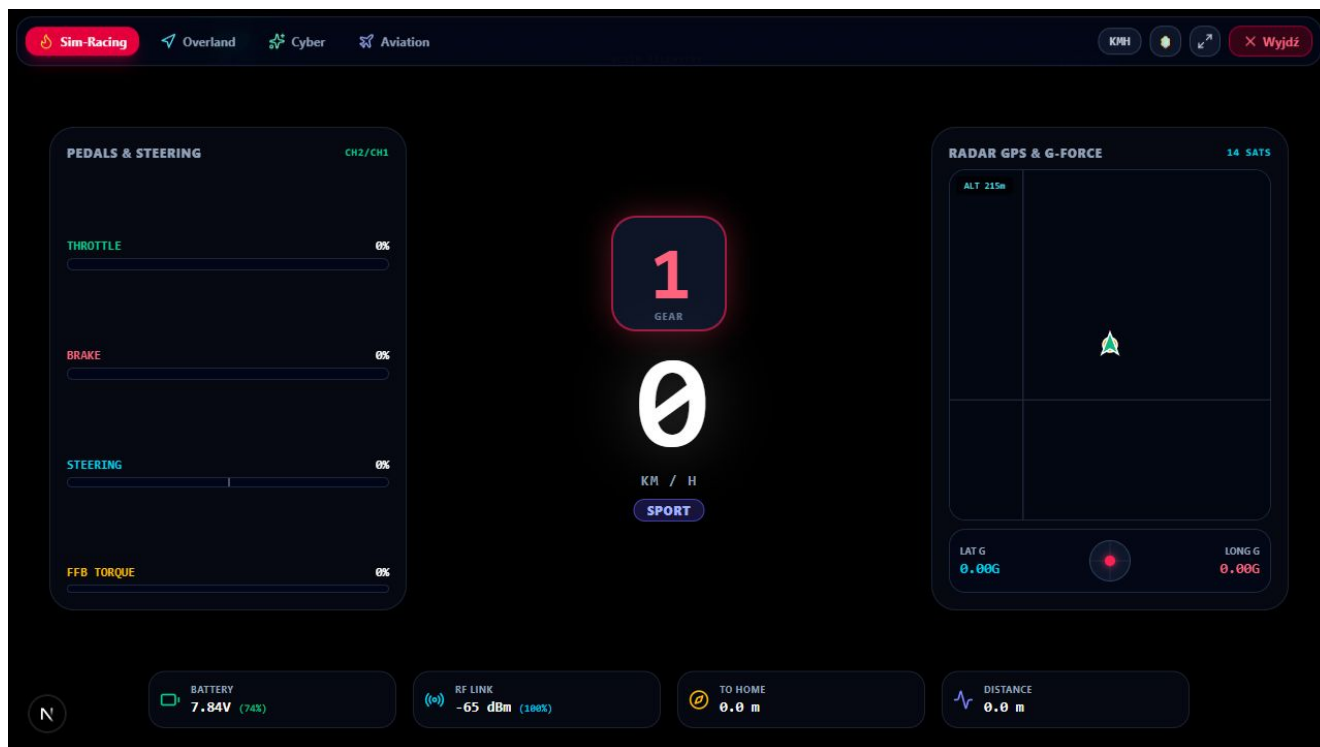
Procedure: at a small input, compare the throttle channel in Sport and then one other mode. Wait for the station status reading. Check neutral and brake behaviour before driving. If highlighting does not change, check the host and status data; a message appended to the terminal alone does not confirm execution.

4.5. Cruise control

Cruise Control displays status and a demand in μs . In this version, **Enable** requests **1650 μs** , not capture of the current pedal position or GPS speed regulation. The relevant output depends on profile mapping and the supported input reader.

1. Test enabling it with wheels raised or the drivetrain disconnected. Watch throttle channel and demand.
2. Press the active button again to disable it; check that control actually returns.
3. Before using it in motion, also confirm braking/manual takeover on your own profile. Do not assume the Cruise Control label implies the same logic as a road car.
4. Before ending a session, disable cruise and stop the vehicle. Station disarming clears the local cruise state but does not replace checking the drivetrain.

5. HUD / Cockpit — readable instruments



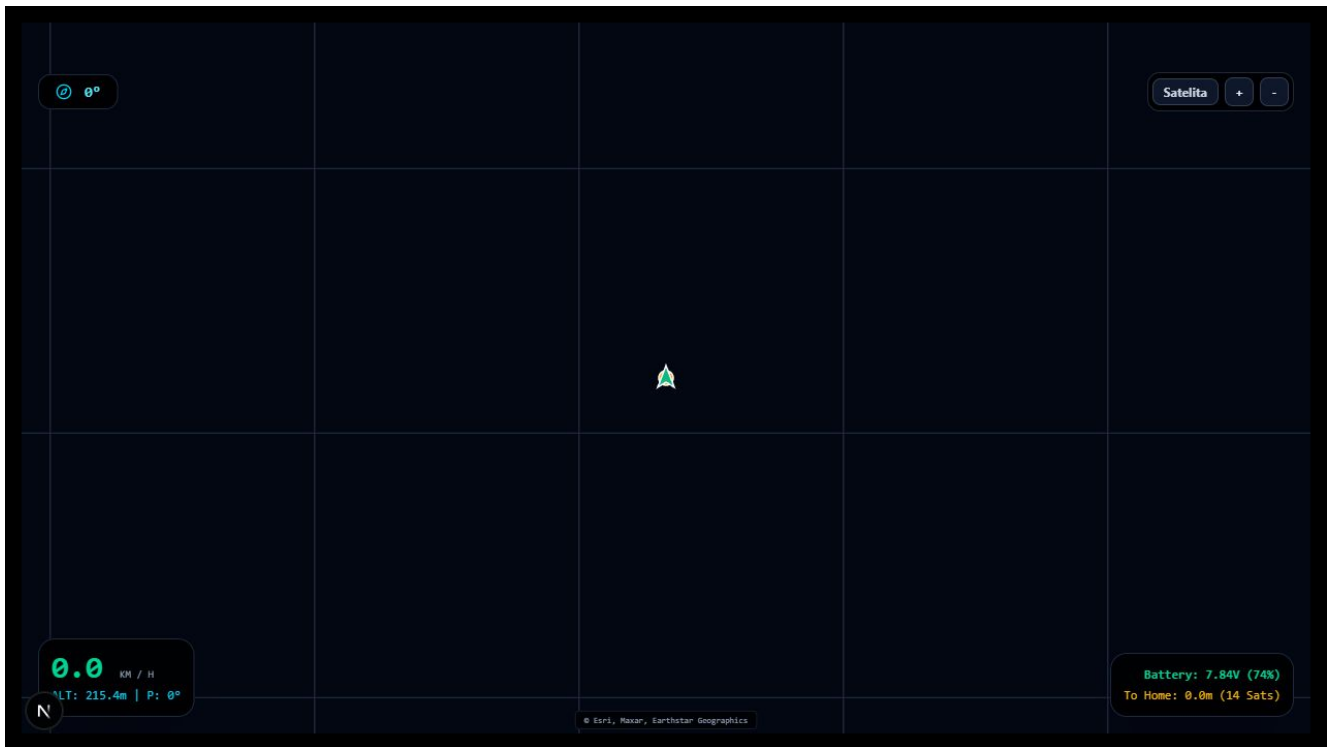
MCS HUD — instruments and layout selection

Open **HUD Mode** from the header or select the HUD tab. The view shows instruments and a map; it is not an FPV camera player.

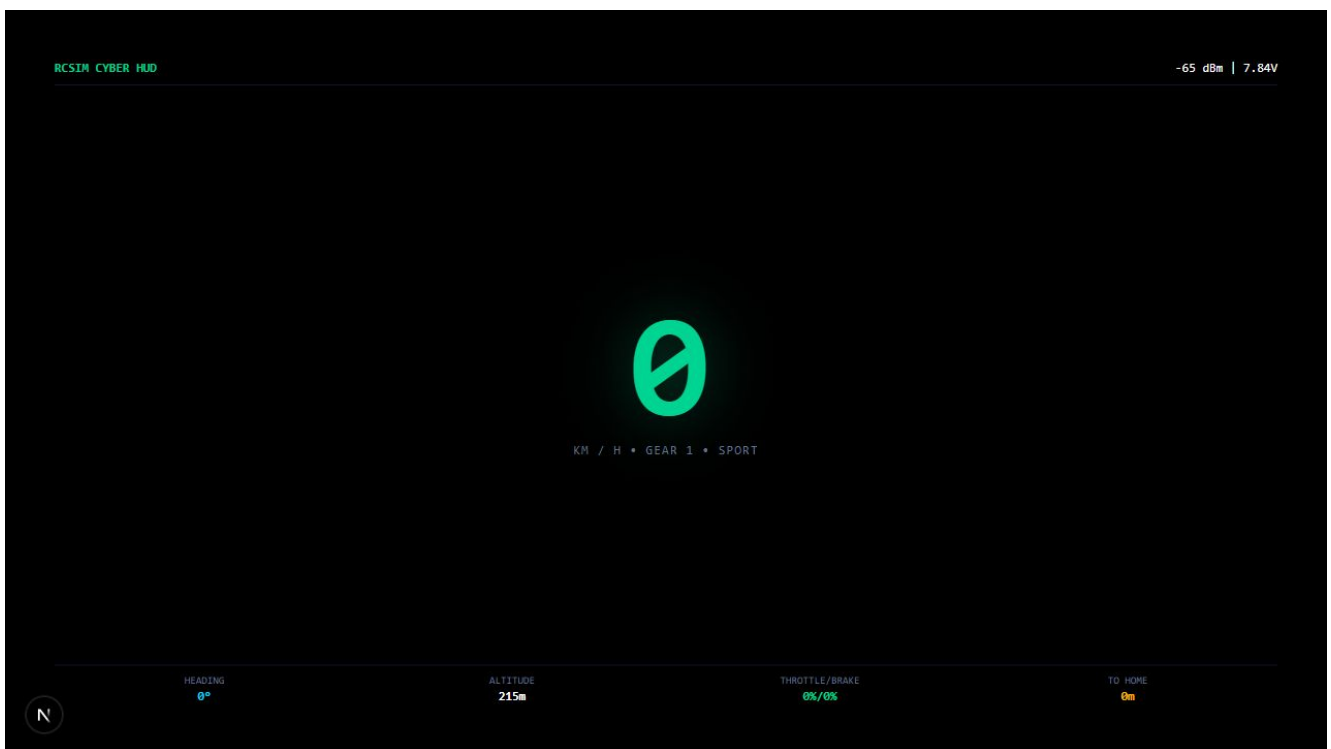
5.1. Operating the view

1. Touch the screen or move the pointer to reveal the top bar. It hides after about 4 seconds.
2. Select **Sim-Racing**, **Overland**, **Cyber** or **Aviation**. Presentation changes; the control profile, equipment and autopilot type do not change with the graphics.
3. Use **KMH/MPH** to change the HUD speed unit.
4. The  icon mirrors the view; it changes presentation, not model steering direction.
5. The fullscreen icon switches browser mode if supported by the device. It is not a driving start button.
6. In the map layout, select OSM or satellite and use zoom-in/zoom-out buttons.
7. **Wyjdź** returns to the monitor. Close this overlay before using another tab; clicking Navigation behind it may not hide it. In this version, the caption may remain Polish even with EN selected. Exiting does not disarm the model.

The browser may attempt to keep the screen awake, but the phone's operating system may restrict it. Check screen locking and visibility before a test; do not assume an open HUD overrides all sleep settings.



HUD Overland — map layout and journey indicators



HUD Cyber — simplified data presentation



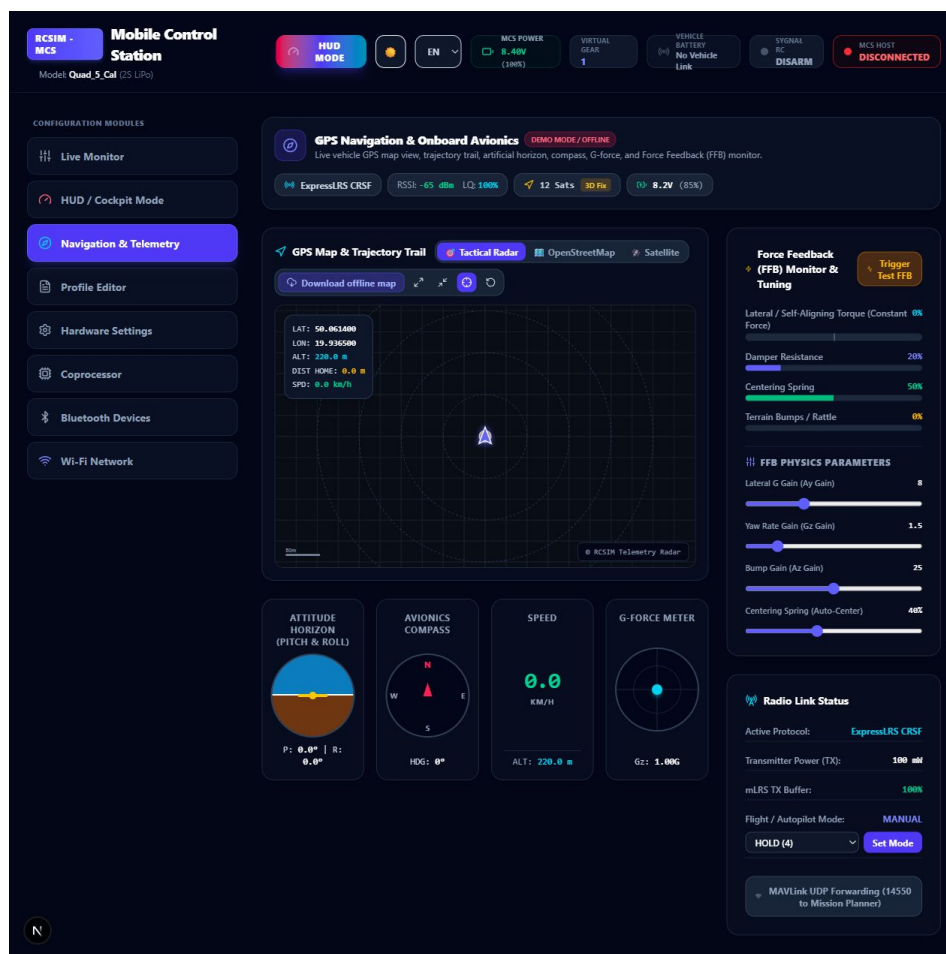
HUD Aviation — horizon and aviation instruments

5.2. Understanding readings

- Speed, position, altitude, heading, satellites, battery and link quality depend on vehicle telemetry.
- Gear, driving mode and cruise come from MCS state; they are not direct measurements of a mechanical gearbox.
- Steering and pedal bars use an assumed channel arrangement. With custom mapping, compare them with Live Monitor.
- The tachometer-like bar and shift suggestion are a calculated throttle/speed presentation, not measured engine RPM.
- G-force indicators depend on IMU data and orientation. A stationary horizon without current data does not confirm that the vehicle is level.

Without communication, HUD may retain previous or initial example data, including coordinates. Instrument appearance, satellite count or a 3D FIX graphic alone do not confirm current GPS data. First check connection and fresh data in Navigation & Telemetry.

6. Navigation & Telemetry — map, sensors and diagnosis



MCS Navigation & Telemetry — map, measurements and link controls

6.1. Telemetry header and sensors

Active/Offline in this view concerns telemetry, not just browser access to the host. Check protocol, RSSI in dBm, LQ in %, satellite count and **NO FIX/2D FIX/3D FIX**. Battery readings here concern the vehicle, separately from station power.

Horizon, compass, speed and G-force panels show orientation, heading, GPS speed and acceleration components respectively. Before interpreting them, make sure the vehicle actually supplies these sensors. Demonstration data or values retained after a fetch error can look plausible.

Test: with the drivetrain secured, rotate the vehicle slightly and compare horizon and heading changes. Check GPS where a valid fix is possible. Do not enable autonomous operation solely because a map is drawn.

6.2. Layers and map movement

- **Tactical Radar** is a local geometry and trail view; the name does not imply a physical obstacle radar.
- **OpenStreetMap** and **Satellite** select the map background.
- **+ / -** change scale. Drag the map with the mouse to inspect surroundings; use the recenter icon to return to the vehicle.
- The clear-trail icon clears the locally displayed trajectory. The trail may return after the next station update; this is not permanent deletion of device history.

- **Distance to Home** shows distance to the remembered home; **Distance traveled** shows travelled distance. Home is set automatically from the first suitable 3D GPS position; it is not a target set by clicking the map.

The view has no GCS-style waypoint mission editor. Clicking/panning the background does not send a drive-to-point command. Without valid coordinates, the map or download may use example position **50.0614, 19.9365**. Check the numbers before saving offline maps.

6.3. Downloading offline maps

1. With working internet and the correct GPS position, select **Download offline map**.
2. Check the displayed centre. Downloading uses the telemetry position, not an arbitrary place to which you pan the map.
3. Select OSM or satellite in **Map Layer**. Choose 1, 3, 5 or 10 km in **Area Radius**.
4. The view specifies zoom levels **13–17**. Tile counts and MB estimates are approximate; larger areas require more space and transfer.
5. Press **Start Map Download** and watch completed tiles, progress and messages. A partially failed download does not provide a complete offline map.
6. **Cancel download** interrupts the task. Closing the dialog with X only hides it; use the proper button to cancel.
7. Check **Local cache** and test the prepared area without internet access. Access to MCS Host is still required.

Clear cache deletes downloaded tiles after confirmation. Do not use it before a trip if you need to keep offline maps. A missing background with a correct position calls for checking layer, downloaded coverage and cache, not changing control channels.

6.4. FFB monitor and effect settings



MCS — force monitor and FFB effect settings

Constant Force, **Damper**, **Spring** and **Bump** bars display effect state. **Trigger Test FFB** requests a short **65% pulse for 0.6 s**. This differs from the gentle GCS test; prepare the wheel and keep hands clear of pinch points before pressing it.

The test requires a compatible device and working station FFB support. Button animation or a page response does not prove force was physically produced. The test also injects an artificial IMU sample; do not interpret it as vehicle movement.

Lateral Acceleration (ay), **Yaw Rate (gz)**, **Vertical Acceleration (az)** and **Centering Spring** sliders adjust lateral acceleration, rotation, vertical movement and centring contributions respectively. Each change is sent immediately and saved in the active profile; this panel has no additional Apply.

First check a small movement and correct sign in fresh telemetry, then increase one effect at a time. On reopening the view, do not assume initial slider positions faithfully read all previous settings; compare the saved profile and actual response.

6.5. Radio, autopilot mode and MAVLink Forward

The link panel shows protocol, TX power, transmit buffer and reported **Flight Mode**. Choosing a list item does not change the vehicle until you press **Set Mode**.

The list groups **ArduPilot Rover** modes: MANUAL (0), ACRO (1), STEERING (3), HOLD (4), LOITER (5), AUTO (10), RTL (11), SMART_RTL (12), GUIDED (15), TURTLE (23), and **ArduPilot Copter** modes:

STABILIZE (0), ALT_HOLD (2), POSHOLD (16), BRAKE (17). These are requests for a compatible autopilot, not functions created by the MCS panel alone.

Mode-change procedure: check autopilot type and its prerequisites, select the appropriate mode, press Set Mode and check the vehicle's subsequent Flight Mode reading. A finished button animation or list selection does not confirm command acceptance. Do not expect a MAVLink mode change on an ordinary CRSF path without a supported autopilot.

MAVLink UDP Forward enables forwarding telemetry to a prepared receiver, such as an external station. It is not a separate transmitter protocol choice or arming action. After enabling it, check data in the receiving program; a green button may precede confirmation from the station. Forwarding address and port must be prepared in hardware configuration.

6.6. Logs and recognising demonstration data

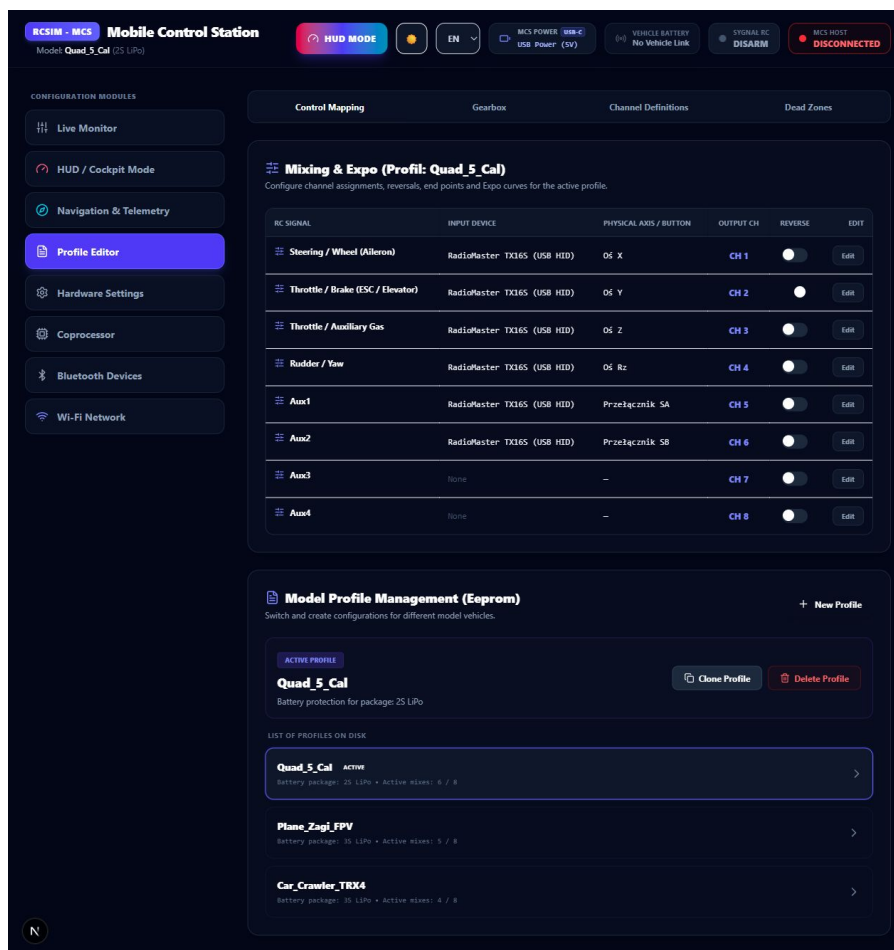
In **Coprocessor**, distinguish the firmware update log from the system log terminal. When an error occurs, record the time, selected profile/protocol, actions taken and last messages. Do not start an update just to view telemetry.

Some initial port-open and coprocessor connection messages are created by the page after its first status response, while offline mode generates example messages. These are not independent proof of a physical UART connection. The real update log and the device's current response mean something different from such screen messages.

During diagnosis, check in order: host access, real controller input in Matrix, the active profile's channel, physical output response and fresh telemetry. If only the screen shows movement or a correct status, keep the model secured until the complete path is confirmed.

7. Profiles — creation, selection, and persistent saving

A profile combines device assignments, channel definitions, calibration, dead zones, wheel settings, and the virtual gearbox. USB/Bluetooth controllers must be visible on the MCS host, usually Raspberry Pi. A phone displaying the panel does not automatically become the controller host.



MCS profile editor

The illustrations show the actual frontend running locally without a backend. They do not confirm a vehicle connection, configuration saving, or device measurements. Some Polish labels may remain when the interface language is English.

Creating and copying

1. Disarm the model and secure the drivetrain. Open the profile editor, then the profile management section at the bottom.
2. Select **New Profile**, enter a recognizable name such as Rover_2S_Wheel1, and choose the LiPo battery: **2S (7.4 V)**, **3S (11.1 V)**, **4S (14.8 V)**, or **6S (22.2 V)**.
3. Confirm creation. The new profile is selected and submitted for saving. **Cancel** closes the form without creating it.
4. Assign devices; a new profile has eight prepared functions: Aileron/CH1, Elevator/CH2, Throttle/CH3, Rudder/CH4, and Aux1–Aux4/CH5–CH8. No devices are initially assigned.
5. Before changing a working setup, use **Clone**. The copy retains mappings and settings, receives the _Kopia name suffix, and becomes the selected profile.

Names cannot be empty. The application simplifies them to an identifier containing lowercase Latin letters, digits, and hyphens; different name spellings can therefore collide. If an existing-profile message appears, change the name. **Delete** requires confirmation and cannot remove the only profile.

Battery type affects presentation and profile-related thresholds; it does not detect the physical pack's chemistry or cell count. The “battery protection” label alone does not confirm drivetrain cutoff.

Selecting a profile and confirming the save

Clicking a card in the list changes the profile displayed in the browser. In the verified version, that click alone does not submit the change to the backend. Before driving, make an actual, intended setting change in the selected profile, wait for a successful-save message from the RPi, refresh the page, and check that the correct profile remains selected.

Mapping, definition, gearbox, and profile changes are submitted automatically. There is no separate “edit, then confirm everything” session. Saving may reload input and output threads. Edit with the model disarmed and secured; do not assume that closing a form undoes a change.

Configuration is stored on the MCS host in `config/config.json`, not solely on the phone. After losing the Internet/local connection, the state visible in the form may not mean a successful save. Check the message and readback after refreshing. Avoid concurrent edits from several browsers: the form submits the complete set of profiles and hardware settings.

8. Control mapping, channel definitions, and calibration

Mapping table and Matrix diagnostics

Control Mapping displays the function, device, axis/button, output channel, and direction settings. Rows are sorted by channel. Expanding a row's editor provides device selection, axis identifier, channel, subtrim, failsafe, calibration, and expo. A split axis has separate throttle and brake rows.

Matrix diagnostics are not a separate sidebar entry. In the monitor, click a controller card in the USB/HID list; once data is available, the diagnostic panel appears below the list. It shows the selected device's actual inputs: axis name and code, current raw value, minimum and maximum, and a position bar. The button grid highlights the expected codes 288–319. A button with another code may not light up without being faulty; check detection and the code returned by the device.

Matrix is a monitor, not a matrix of arbitrary channel connections. Merely selecting a device does not change assignments. Data comes from the MCS host, not USB devices connected only to the phone.

Device assignment and automatic detection

The screenshot shows a software interface for configuring an axis mapping. It is divided into several sections:

- HID CONTROLLER:** A dropdown menu showing "RadioMaster TX16S (USB)" and a blue "Assign" button with a target icon.
- AXIS / SWITCH:** A dropdown menu showing "Oś X".
- OUTPUT CHANNEL (1-16):** A dropdown menu showing "1".
- SUB-TRIM: 50us:** A slider control ranging from -100us to +100us, currently centered at 50us.
- FAILSAFE MODE:** A dropdown menu showing "Center (1500us)".
- PHYSICAL AXIS CALIBRATION:** A section with the instruction "Configure physical endpoint limits and the center (neutral) point." It contains three input fields: "Min (Raw):", "Środek (Raw):", and "Max (Raw):", each with a "Domyślne" (Default) button. Below these is a "Click Start to measure" label and a "Start Calibration" button.
- EXPO CURVE:** A section with the instruction "Adjusts sensitivity around center (stick neutral)." It features a graph showing a curve between -100% and +100% on the x-axis. Below the graph is a slider control ranging from "LINEAR (0)" to "MAX (100)", currently set at 35%.

Expanded axis mapping editor — frontend preview without a backend

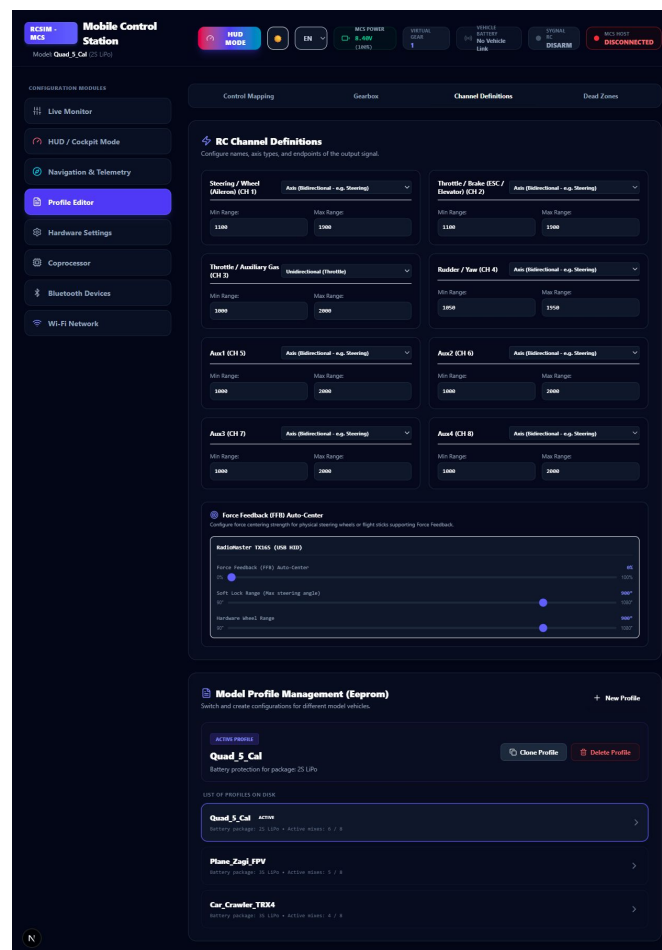
1. Connect the wheel/pedals to the MCS host and check their presence in the monitor.
2. Open the appropriate row's editor. Choose the device or **Start detection**.
3. Within **5 s**, move only the desired axis or press one button. Do not operate the wheel and pedals simultaneously.
4. Check the detected device name and identifier, such as ABS_X. Repeat if detection times out.
5. Select **output channel 1–16** to match the receiver and control path. An eight-channel bridge does not become sixteen-channel simply because the form accepts 16.

Names such as ABS_X, ABS_Z, and BTN_... are identifiers, not universal function descriptions. The same pedal on another device may use a different code. Use detection results, especially for special buttons; a manually entered descriptive name may not match the reader's format.

Do not assign several independent functions to one channel without a deliberate purpose. The last event may overwrite the value; the form is not an arbitrary mixer summing system. Assignments with device **None** do not create an active input in the saved configuration.

Channel type — choose behavior before tuning

In **Channel Definitions**, select a type for each function:



Channel type definitions and Autocenter settings — frontend preview without a backend

- **Bidirectional Axis:** an axis with a center, such as a wheel. Below center decreases the pulse; above center increases it.
- **Unidirectional:** one pedal or trigger. The physical range is normalized to 0–1; this does not automatically mean a 1500 μ s neutral. Throttle behavior also depends on a function named Throttle, the gearbox, and drive mode.
- **Split Axis (Throttle/Brake):** two independent axes combine into one bidirectional channel. Throttle and brake may come from different devices. Brake takes priority in ordinary driving mixing.
- **Incremental Axis (Rate Mode):** deflection determines how quickly the channel value moves, not its direct position. Returning the axis to neutral stops the change and retains the value reached. **Speed Scale 50–5000**, in steps of 50, sets the rate scale in μ s/s at full deflection; the default is 500.
- **Switch ON/OFF:** successive presses toggle a remembered state between **OFF** and **ON**. Each form value accepts **800–2200 μ s**.
- **Momentary Switch:** **Default** when released and **Active** while pressed; both **800–2200 μ s**.
- **Multi-position Switch (Carousel):** successive presses cycle through a comma-separated pulse list, for example 1000, 1500, 2000. Check the actual starting position and first step instead of assuming that the first press selects the first number.
- **Multi-position Switch (Separate Buttons):** the form label describes more than the current implementation. For an axis, the reader divides its range into positions from the list; for one assigned button, it selects the list's last value. This form does not assign a separate button to every position.

EPA Min accepts **800–1500 µs** in the form, and **EPA Max 1500–2200 µs**. Minimum must remain below maximum. However, the backend's current channel state clamps signals to **1000–2000 µs**. Entering 800 or 2200 does not guarantee that pulse at the output.

For **Split Axis**, the form shows brake minimum, center **1000–2000 µs**, and throttle maximum. In the verified evdev reader, the actual mixing center is **1500 + throttle Subtrim**; the separate center field is saved but not used in this path. Check the monitor and physical output rather than relying on the field alone.

Raw input calibration

Raw Min / Raw Center / Raw Max calibration defines the controller's physical reading. It is not ESC calibration and is not measured in µs: the range may be 0–65535 or another value reported by the driver.

1. Assign the device and axis, then select **Start calibration**. The live reading must respond to that particular axis.
2. Move to the first physical endpoint and select **Set Min**.
3. For a wheel, return to its actual center and select **Set Center**.
4. Move to the other endpoint and select **Set Max**, then **Finish**.
5. Calibrate throttle and brake pedal minimum and maximum separately; their midpoint does not serve the same purpose as a wheel's center.
6. Check the complete movement. Min and Max must not be identical. If direction is reversed, use inversion instead of changing several parameters at random.

Values can be entered manually. **Reset** removes custom calibration points and allows the driver range to be used; it does not reset every profile setting. A disconnected device cannot supply an actual reading. Check endpoints again after changing the cable, driver, or device mode.

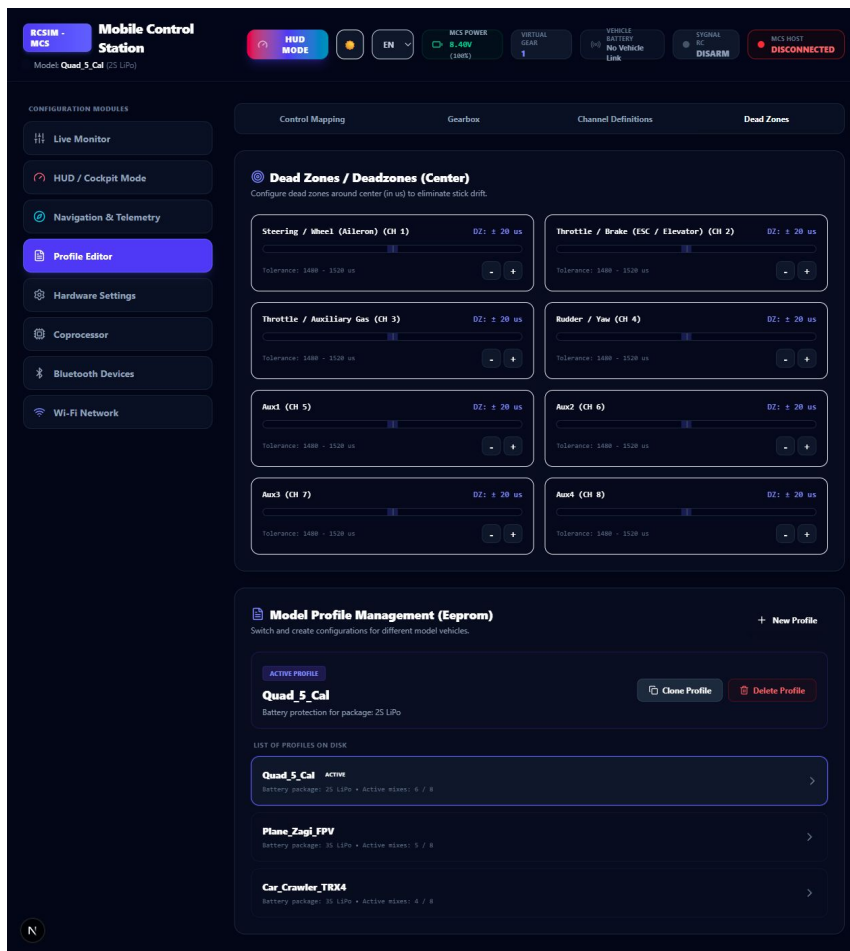
Reverse, Subtrim, Expo, and dead zone

Reverse changes assignment direction. A split axis has separate Reverse and Expo settings for throttle and brake. With both pedals released, check that both readings represent zero demand before fully pressing a pedal.

Subtrim: –100 to +100 µs shifts the neutral point. It corrects the final signal, not the physical Raw Center. A separate brake Subtrim is visible and saved, but the verified evdev mixer does not use it for the final split-axis channel; do not expect independent offsetting of both halves.

Expo: 0–100% softens response at small deflections while retaining the extreme range. At 0 the curve is linear; increasing it makes the response more cubic. The graph previews the curve, not servo movement. Set calibration and direction first, then add Expo.

Dead Zones reduce responses to noise near neutral. **–/+** changes the setting by **5 µs**, within a nominal **0–100 µs** range; the bar shows tolerance around 1500. In the current form and save process, 0 is replaced by the default **20 µs**, so do not treat reaching zero as successfully disabling the zone. For a pedal, the zone acts near release, not halfway through its travel.



Channel dead zones — frontend preview without a backend

Input failsafe

Center sets **1500 + Subtrim**, limited by EPA. **Custom** uses the selected **1000–2000 μ s** value, also clamped. **Hold** retains the last value. The reader blocks Hold for a function named exactly Throttle (case-insensitive), but do not assume identical protection for every differently named throttle function.

This is a response to loss of an input device, not the complete radio or coprocessor failsafe. Disarming and output loss have separate, protocol-dependent behavior. Test controller disconnection, MCS shutdown, transmitter USB loss, and RF loss separately. For some ESCs, 1500 does not mean a safe stop.

First setup: wheel and two pedals

1. Create a profile, disarm the model, and check actual inputs in Matrix.
2. For the wheel, select the steering function, bidirectional type, and receiver-compatible channel, for example CH1. Detect the axis, set Raw Min/Center/Max, and check left–center–right.
3. For propulsion, select the Throttle function, change its type to Split Axis, and set the correct drive channel, for example CH3. Do not use CH2 merely because it is named Elevator.
4. Assign and calibrate throttle and brake separately. With both released, expect neutral; throttle should change the output in one direction and brake in the other. Check both pedals pressed simultaneously.
5. Leave the gearbox disabled and use Sport mode for the first check; set conservative EPA, modest Expo, and a verified failsafe. ESC settings must match this control method.

6. Wait for saving, refresh the page, and verify the profile and mappings. Check the physical output with the drivetrain secured.
7. Test device disconnection. Only then add FFB, the gearbox, and button functions.

If nothing moves, check the device and axis identifier. If the input monitor works but the channel does not, check type, assignment, and the backend's active profile. If the channel works but the model does not, check port/protocol/arming and the rest of the control path. If a pedal constantly requests motion, check Reverse and calibration first rather than blindly increasing the dead zone.

9. Wheel, FFB, gearbox, and special buttons

Device settings in Channel Definitions

Assigned devices expose **FFB Autocenter 0–100%**, **Steering Range 90–1080°**, and **Hardware Wheel Range 90–1080°**, both angle ranges in 10° increments. The first describes the useful steering range; the second describes the base's physical range. Match them to the base, then check input endpoints.

Autocenter requires driver support. For selected Logitech wheels, the application attempts to set the system hardware range; this is not a guarantee for every brand. If Steering Range is below Hardware Range and the device supports a spring effect, a resisting soft-lock may be available. Do not treat it as a mechanical stop.

Telemetry FFB — separate from profile Autocenter

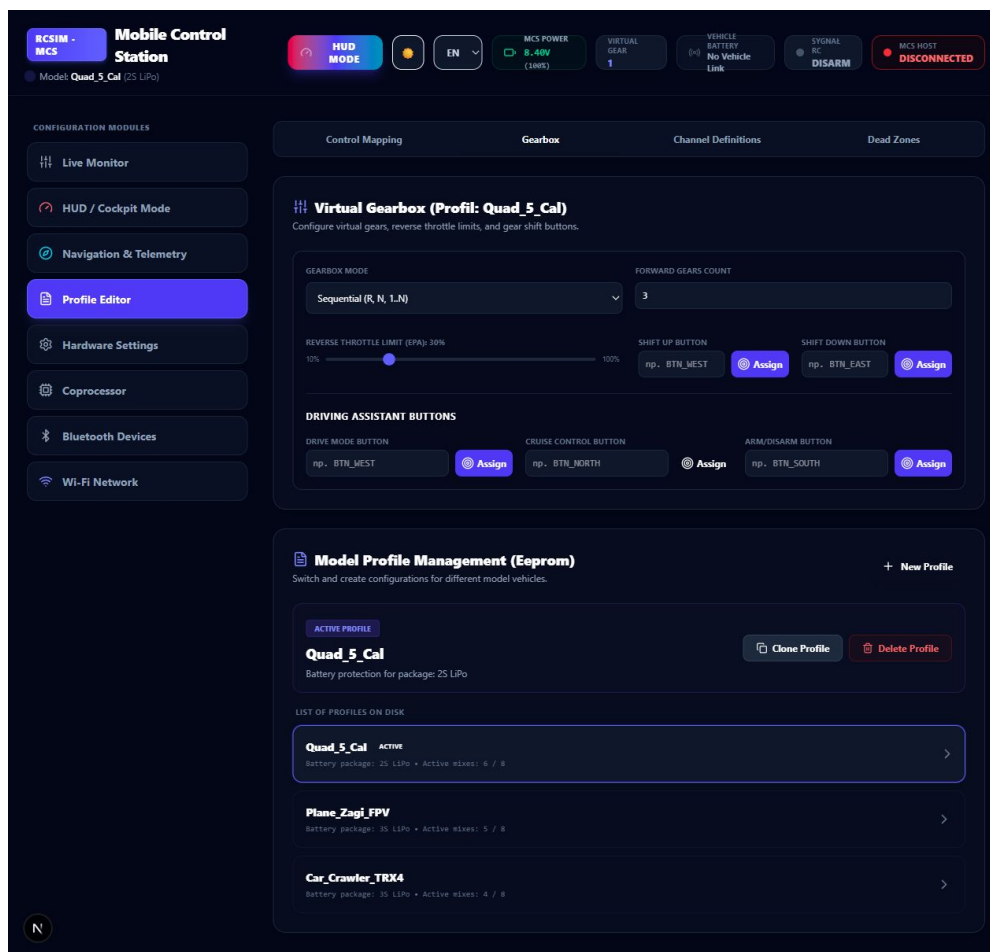
The telemetry/navigation view provides monitors for directional force, damping, spring, and bumps, plus sliders: **ay 0–25** in 0.5 steps, **gz 0–10** in 0.2 steps, **az 0–50** in steps of 1, and **centering 0–100%** in steps of 5. ay amplifies lateral motion, gz rotation, and az bumps; these are multipliers, not Nm measurements.

Support depends on device capabilities. In the verified evdev path, a wheel receives constant force, while a gamepad receives rumble: the strong motor represents bumps and the weak motor the lateral component. Spring and Damper values shown in the monitor do not prove that separate dynamic spring and damper effects are sent to the device. Input Autocenter is a separate function.

The FFB test button in this panel requests **65% for 0.6 s**. This is much stronger than the short 5% test in PC GCS. Limit force in the base first, mount it securely, and keep the rim safe. Do not press this test assuming it matches the GCS test.

FFB parameters are saved in the active profile through separate telemetry settings. Other profile-editor operations reconstruct configuration without explicitly carrying `ffb_config` across; after changing a mapping, check FFB sliders and their saved values again. Do not rely solely on the form's previous appearance.

Virtual gearbox



Sequential gearbox settings — frontend preview without a backend

In the **Gearbox** subtab, choose **none** or **sequential**. Forward gear count: **1–10**. Reverse limit: **10–100%**. Assign gear-up and gear-down buttons through detection; one press changes gear by one, with a 0.25 s repeat guard.

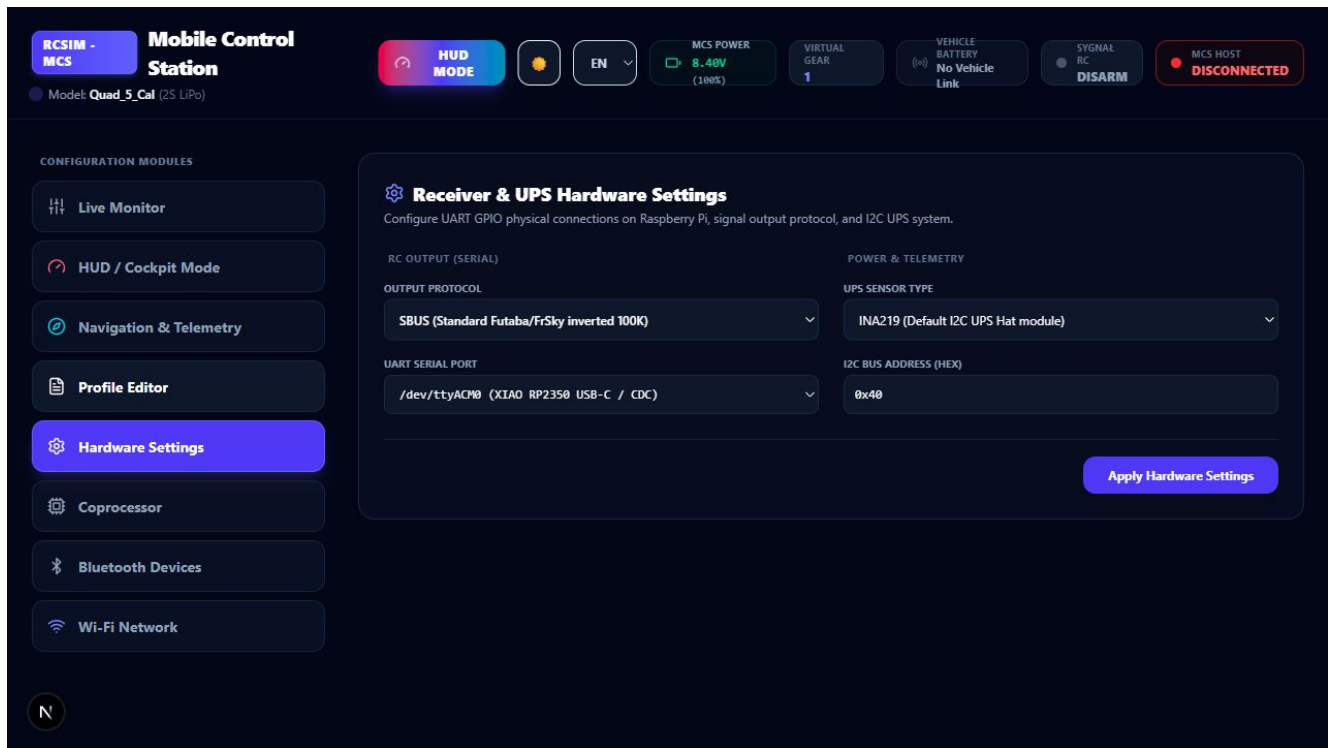
This limits and directs the throttle channel; it does not physically shift a transmission. Positions are **R (–1)**, **N (0)**, and **1...N**. Neutral zeros throttle demand; successive forward gears allow a power fraction of gear number / gear count. For example, with three gears, first gear gives a limit of about one third. The reverse limit applies in R; brake and throttle change directional roles according to the virtual gear.

Drive Mode, Cruise, and Arm

- **Drive Mode button** cycles **Sport** → **Eco** → **Crawler** → **Wet**. Sport keeps full scale; Eco limits the throttle component to 50%, Crawler to 20%, and Wet to 65%. These are demand multipliers, not guaranteed speeds. Crawler may add slight braking at neutral throttle.
- **Cruise button** toggles a held throttle pulse. It is not GPS-based speed regulation. On activation, the reader takes **CH3**; if its value is below **1520 μs**, it substitutes **1650 μs**. Enabling it while stopped can therefore request movement. Do not use it without checking the channel and drivetrain response. On a split axis, brake above about 5% cancels the hold; gear changes and disarming cancel it too.
- **Arm/Disarm button** toggles local pause and may send an arming command through the telemetry manager. This does not mean identical stopping behavior for every protocol; for MAVLink, separately verify command acceptance and the vehicle's actual response.

Special buttons are saved as codes without an independent device assignment. Matching codes on another controller in the profile may invoke the same function. Use different buttons for gears, cruise, arming, and AUX channels; otherwise one physical action may perform several operations at once.

10. Hardware — RC output, port and station power



Hardware panel preview without a connected backend or devices

The **Hardware** tab selects the output device of the Raspberry Pi running MCS. These are operator-station settings, not automatic configuration of the receiver in the model. Disarm the system and secure propulsion first, then identify the protocol required by the transmitter or coprocessor.

Output Protocol

The list contains seven choices:

- **Tier 3: mLRS / MAVLink RF 2.0 (Nomad / FC Serial)** — direct MAVLink output for a compatible link and flight/vehicle controller.
- **Tier 3: ExpressLRS Direct (CRSF 150–250Hz / Nomad)** — direct CRSF frames to the transmitter, bypassing the coprocessor.
- **CRSF (Crossfire — Fast 150Hz / 420kbps)** — also direct CRSF output in the current application.
- **Tier 1: RP2350 USB Bridge (PPM/SBUS/i-BUS/CRSF)** — generic bridge selection; currently starts the default PPM mode.
- **SBUS (Standard Futaba/FrSky inverted 100K)** — SBUS output from the coprocessor.
- **i-BUS (FlySky digital 115.2K)** — iBUS output from the coprocessor.
- **PPM (Analog via coprocessor)** — coprocessor-generated PPM for a compatible trainer input.

Frequency and baud-rate figures in these labels do not set the port parameters. Direct CRSF has a target interval of 10 ms and a default baud rate of 115200 bps in the application. Do not assume 420000 bps or

150–250 Hz from a label alone. SBUS and iBUS describe the signal leaving the coprocessor; the MCS-to-bridge USB/serial connection still operates at 115200 bps.

This form has no baud-rate control. Direct CRSF uses `output_baudrate`; MAVLink uses `mavlink_baudrate` or, if absent, `output_baudrate`, with a final fallback of 115200 bps. An example configuration may contain different values, such as 57600 bps for MAVLink. If the transmitter requires a different rate, compatible configuration outside this form is necessary. Saving from the panel replaces the configuration with the form and profile fields: it may remove additional baud-rate and Wi-Fi settings. Recheck those settings after saving, particularly before restarting the station.

UART Serial Port

The list offers `/dev/ttyACM0`–`/dev/ttyACM5`, `/dev/ttyAMA0`, `/dev/ttyUSB0` and `/dev/ttyS0`. ACM0 is described as XIAO RP2350 USB-C/CDC, USB0 as an external USB–UART converter, and AMA0 and S0 as Raspberry Pi UART variants. These are listed names, not automatic detection of the device actually connected. The port number may change after connecting another device or restarting. Connect the coprocessor or transmitter to the MCS host, not to the phone displaying the panel.

1. Select the protocol matching the device and actual wiring.
2. Select its port. If uncertain, check serial devices in the host operating system.
3. For SBUS or iBUS output, select that explicit protocol instead of generic RP2350.
4. Click **Apply Hardware Settings**.
5. Check the configuration-save and output-restart logs, then channel readings and receiver response.
6. Confirm operation on the bench with propulsion secured before arming for driving.

The expected result is correct channel reception by the actual device. The form's success message does not confirm transmission or the model's correct response.

Eight or sixteen channels?

MCS maintains 16 channels. Direct CRSF transmits 16 channels. The USB-to-coprocessor bridge passes only CH1–CH8 in a 19-byte frame. It cannot provide control of CH9–CH16 merely because SBUS itself can carry more channels. MAVLink output sends eight values using RC override. Before extending a profile, check the channel count of the entire path, not just the receiver's capacity.

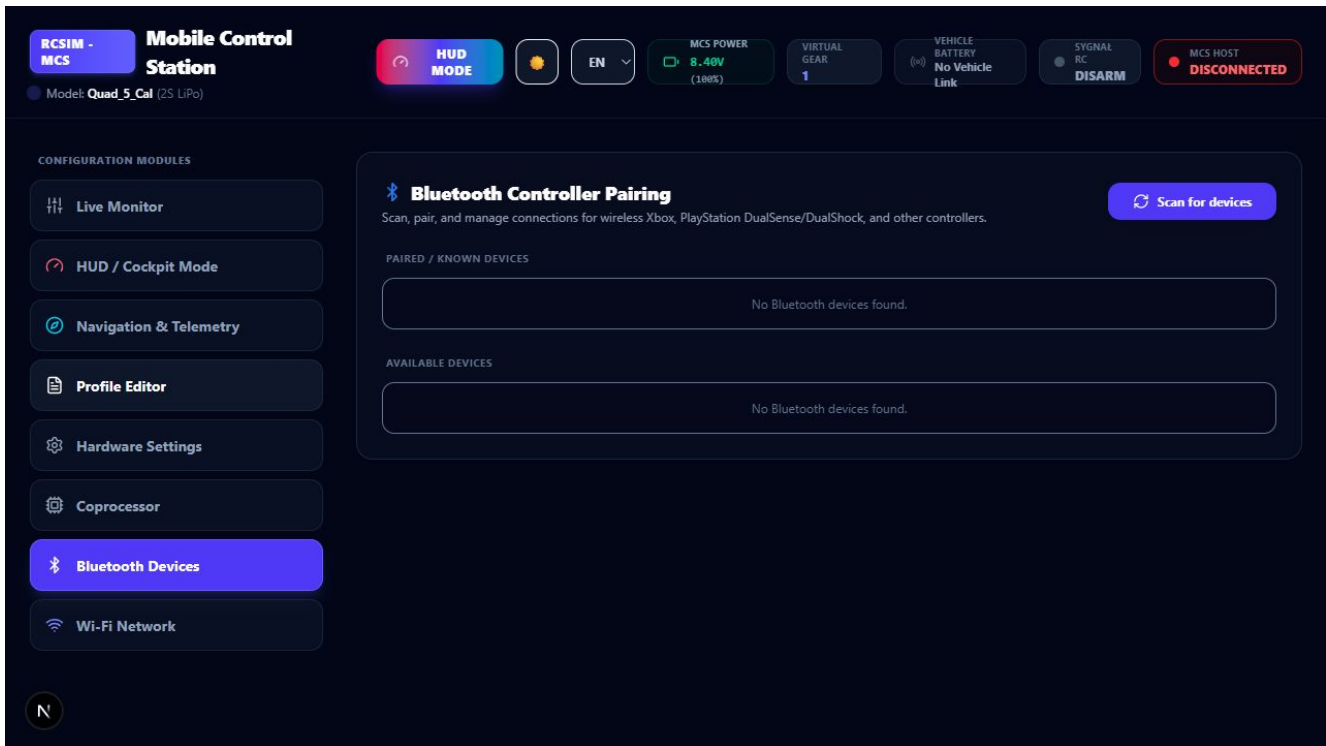
When disarmed, direct CRSF sends neutral channels corresponding to 1500 μ s. The bridge stops transmitting, after which behavior depends on the firmware watchdog. **The current MAVLink output does not check MCS pause and may continue sending live channel values.** The panel's disarm button therefore does not provide a verified stop for a MAVLink model. Neutral 1500 μ s is not a universal STOP command for every ESC either.

UPS Sensor Type and I2C Address

UPS Sensor Type offers **INA219**, **ADS1115** and **No power monitor**. Enter **I2C Address** as 0x40 or 0x42: the prefix 0x followed by exactly two hexadecimal digits. The address field is disabled when no sensor is selected. Save validation still requires a correctly formatted address, so leave a valid value in it.

The current power-monitoring path uses an INA219 driver and updates its address. Selecting ADS1115 does not switch to a verified ADS1115 driver; selecting no monitor is not a verified way to disable backend readings. Do not treat these list entries as proof of hardware support. The station UPS measures MCS power; vehicle battery data comes from separate telemetry. Missing readings or demonstration data do not confirm the physical battery's condition. This tab does not adjust FFB strength; configure the wheel in its appropriate profile/settings.

11. Bluetooth — pairing a controller with MCS



Bluetooth panel preview without a connected backend or devices

The panel manages Bluetooth on the Raspberry Pi. It does not pair the controller with the operator's phone. The phone or tablet is the panel display; the controller must be visible to the MCS host. Bluetooth connection and axis/button mapping in a profile are separate steps.

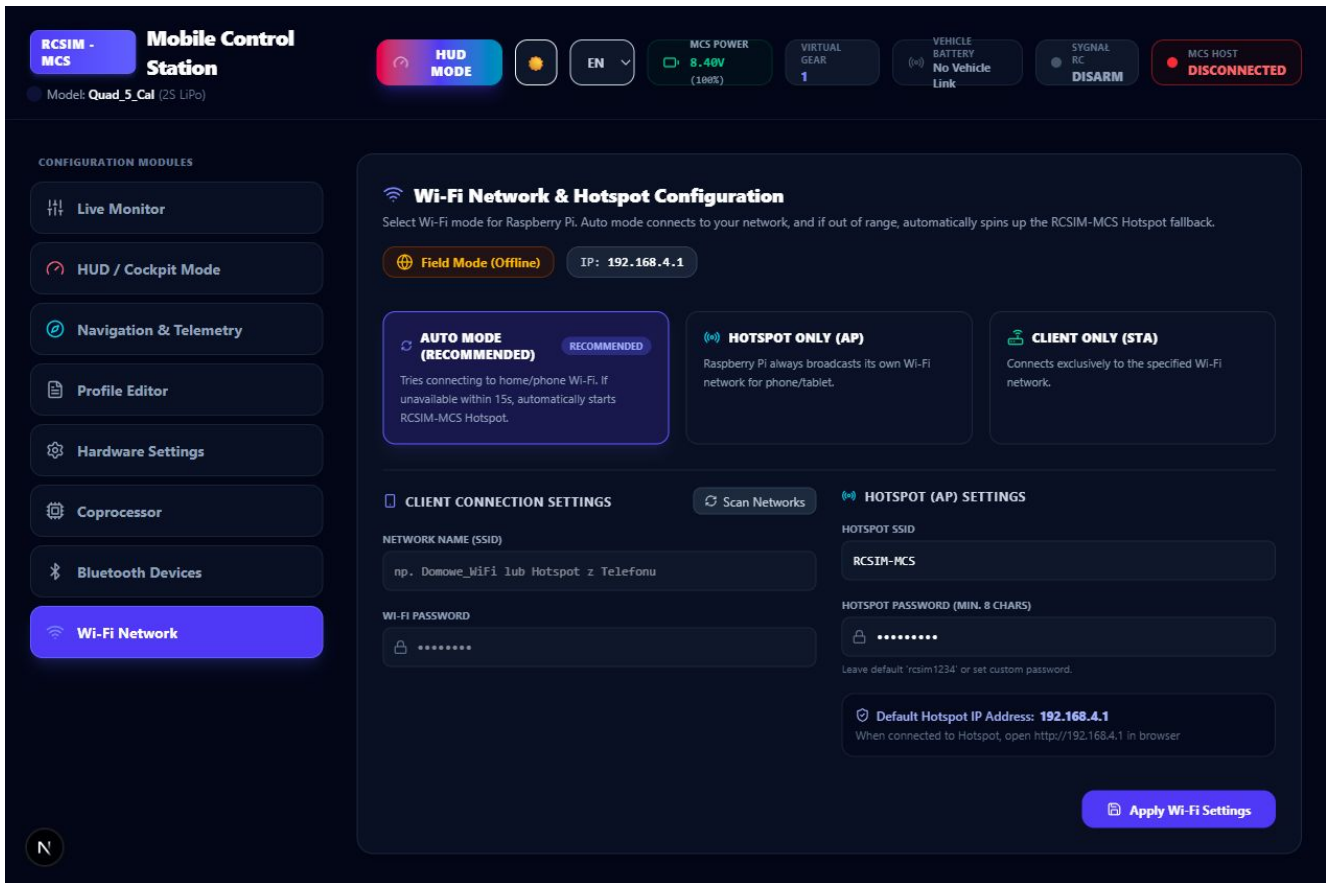
1. Put the controller in its manufacturer-specified pairing mode and keep it within range of the station.
2. Click **Scan for devices** and wait for scanning to finish.
3. Under **Available Devices**, check the intended controller's name and MAC address.
4. Click **Pair**. The application attempts to pair, trust and automatically connect the device.
5. Check the log and the **Connected** status under **Paired / Known Devices**.
6. Open the inputs/profile controls and check axes, buttons and their mapping.

The expected result is a connected device with inputs that actually respond. A name appearing in the list alone does not confirm readiness to control the model. The list refreshes approximately every three seconds; device buttons may be disabled while an operation runs.

For a known disconnected device, the chain-link **Connect** icon reconnects without removing pairing. The **Disconnect** icon ends the current connection while keeping the known device. The **Forget** trash icon removes pairing after confirmation; using it again requires pairing again. There is no separate Trust button: trusting is part of the pairing attempt.

Real Linux operation requires a working adapter, Bluetooth service and bluetoothctl. On other operating systems, the application returns demonstration devices and simulates operations. A successful attempt on a development computer does not confirm pairing of a physical gamepad. Before driving, test the response to controller disconnection and reconnection.

12. Wi-Fi — client mode, hotspot and returning to the panel



Wi-Fi panel preview without a connected backend or devices

The top section shows Internet status and the host's **IP** address. Internet access is not required for the local panel if the phone and MCS have IP connectivity. Open the panel at the station's actual address on port 8080, for example `http://MCS_ADDRESS:8080`. The `192.168.4.1` shown on the card is a hint, not a universal address for every installation. The panel also starts with example values; confirm that it is receiving a response from the host.

Selecting a mode

- **Auto Mode (Recommended)**: the application attempts to connect MCS to the selected network as a client, with a 15 s attempt timeout. It then attempts to start a hotspot whether the client connection succeeds or fails.
- **Hotspot Only (AP)**: MCS creates a network for the phone or tablet to join.
- **Client Only (STA)**: MCS connects to the specified router or phone hotspot.

Simultaneous client and AP operation requires Wi-Fi interface support. The application attempts to create the `uap0` AP interface, falling back to `wlan0` if unsuccessful. That fallback may change the existing client connection. The Auto description is not a guarantee of uninterrupted access or evidence of continuous monitoring for network loss.

Fields and change procedure

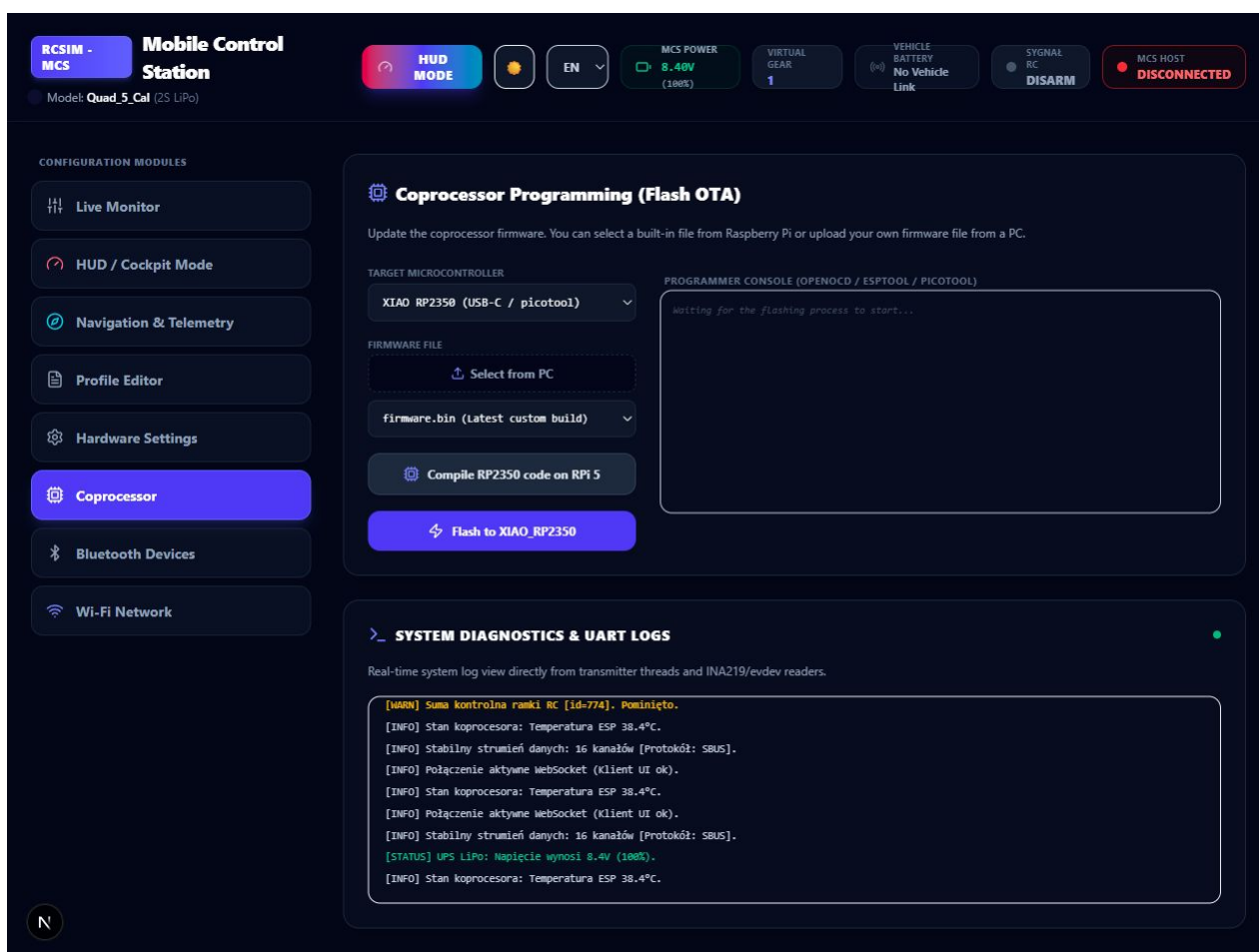
Client Connection Settings contains **Scan Networks**, results showing security and signal strength, **Network Name (SSID)** and **Wi-Fi Password**. Clicking a detected network fills in its SSID; it does not connect immediately. The client section is disabled in AP-only mode.

Own Hotspot Settings (AP) contains **Hotspot Name (SSID)** and **Hotspot Password (min. 8 characters)**. These fields are disabled in STA-only mode. The `rcsim1234` default-password hint does not mean that every prepared image uses that password. Set a known password of your own and record the new connection details before switching. An empty field is not an unambiguous command to delete a saved client password.

1. Disarm the system and prepare a way to regain host access.
2. Select the mode. For client operation, choose the SSID and enter the correct password.
3. For AP/Auto, set the hotspot name and a password of at least eight characters.
4. Click **Apply Wi-Fi Settings**. The browser connection may be interrupted.
5. Connect the phone to the selected network and determine the current MCS address.
6. Reopen the panel on port 8080 and check its status and channel readings.
7. After a controlled restart, verify that client settings and the password actually persist.

A saved response means the request was accepted; network switching happens in the background and may fail. The current save workflow has persistence limitations, particularly for the client password; do not assume successful persistence from a green message alone. On a system without Linux and `nmcli`, the manager operates in demonstration mode. Example networks or IP addresses on that computer do not confirm actual wireless access.

13. Coprocessor — compiling, flashing and logs



Coprocessor panel preview without a connected backend or devices

The **Flash OTA** label describes an operation initiated from the network panel. The browser-to-MCS connection may be wireless, but MCS programs the board over USB/UART or a programmer. It does not imply wireless updates for an arbitrary coprocessor. Perform these operations on a bench with propulsion physically secured. MCS pause alone does not protect the MAVLink variant.

Selecting the target and file

Target Microcontroller offers:

- **XIAO RP2350 (USB-C / picotool)** — .uf2 firmware and a USB connection to MCS.
- **STM32F103 (Blue Pill — OpenOCD)** — .bin or .hex firmware and a compatible programmer/OpenOCD configuration.
- **ESP32 (Co-Processor — esptool)** — an application .bin file for the compatible chip and flashing method.

Firmware File provides **Select from PC**, which selects a file on the device running the browser, including a phone. The trash icon beside a selected file removes it from the current selection. Without an uploaded file, the list offers `firmware.bin`, `Stable`, `Beta` and `Recovery` entries. These are filenames expected on the host, not a download catalog or proof of their presence and compatibility. For RP2350, choose a UF2 file; do not flash an ESP32 binary merely because it appears in the list. An incorrect extension may cause a fallback file on the host to be used instead of the expected upload. Before proceeding, therefore, check that the target, extension and file identified in the logs match.

Compiling RP2350 or STM32 firmware

1. Select the correct microcontroller and make sure its source project is present on the MCS host.
2. Click **Compile RP2350 code on RPi 5** or **Compile code on RPi 5**.
3. Wait for the result in the **Programmer Console**. The first build may download PlatformIO and dependencies.
4. Check the final result and generated file before starting flashing as a separate operation.

The compile button is available for RP2350 and STM32, not ESP32. Compilation is not flashing and requires available tools, storage and possibly Internet access. A successful build does not confirm correct operation of the physical board.

Flashing RP2350 and STM32

1. Check the port in Hardware and the correct target and firmware in Coprocessor.
2. For RP2350, ensure BOOTSEL mode is accessible; the automatic attempt opens the port at 1200 bps.
3. Disconnect unnecessary CDC boards beforehand: the application also checks discovered ttyACM ports, not just the selected one.
4. For STM32, check the compatible programmer connection and OpenOCD configuration.
5. Click **Flash to** with the target name and watch the console.
6. Wait for the tool's final result; then recheck the port and channel behavior.

RP2350 requires picotool and STM32 requires OpenOCD; a button's presence does not provide the entire physical interface. If manual BOOTSEL entry is needed, follow the instructions for the particular board. The application attempts to resume transmission when the operation ends.

Flashing ESP32 — preparation and RESET

1. Select ESP32, a compatible .bin and the correct port.
2. Click **Flash to ESP32**. The panel requests pause and activates the intended bootloader-entry line.

3. Read the instructions in the **Physical Action Required** dialog.
4. Briefly press the board's RESET button only when requested, then click **RESET pressed — Continue**.
5. Wait for the esptool result in the console.
6. After success, perform the final RESET as instructed to start the normal program.
7. Check the port, channel monitor, coprocessor response and disarming before returning to driving.

Automatic preparation requires compatible wiring between the host GPIO and the ESP32 bootloader input. Without it, an on-screen message does not confirm the pin's physical state. This workflow writes the ESP32 application at address 0x10000; do not assume compatibility with an arbitrary merged image. **Cancel** in the RESET dialog releases the programming line and resumes RC transmission. It is not a command to stop the model. The backend also sets the state to resumed after a handled flashing attempt, including an error. After cancellation or an error, recheck arming status and keep propulsion physically secured.

Progress, console and UART monitor

The **Programmer Console** shows stages and final picotool, OpenOCD or esptool logs. The percentage indicator represents interface stages, not an exact count of bytes written. Do not disconnect the board merely because the bar has reached a high percentage. **Monitor UART** after success and **Clear** after an error return to the ordinary log view. They do not restore earlier firmware or undo memory writes. **System Diagnostics & UART Logs** shows system messages and the available coprocessor monitor. No active monitor is a different condition from a physically disconnected board.

14. Troubleshooting and sources

- **The panel works, but the model does not respond:** check protocol, port, baud-rate/firmware compatibility, arming status and the receiver.
- **Channels move on screen, but AUX is missing:** check whether the function is outside the bridge's or MAVLink's eight channels.
- **RP2350 outputs PPM instead of SBUS:** explicitly choose SBUS; the generic RP2350 selection currently selects PPM.
- **Disarming does not stop MAVLink:** do not rely on local pause; secure propulsion and verify receiver/autopilot mechanisms.
- **UPS readings are incorrect:** check the actual INA219, address and bus, and distinguish the station battery from the vehicle battery.
- **Bluetooth cannot find the controller:** check pairing mode, the host adapter and service; disconnect the gamepad from its previous host.
- **A paired gamepad does not control anything:** connect it, check inputs and the active profile; pairing does not create mappings.
- **The phone loses the panel after a Wi-Fi change:** join the correct network and determine the new MCS address; use port 8080.
- **The Wi-Fi client does not reconnect after a restart:** check SSID/password persistence and the effect of subsequent hardware-configuration saves.
- **A firmware file is missing:** check the actual host file or select a compatible file of your own.
- **Flashing will not start:** check the tool, port, bootloader/programmer and final logs, not just the progress percentage.
- **Flashing fails or is canceled:** check arming again; transmission may resume automatically.

This documentation is based on code; panel illustrations do not confirm testing of physical devices. Sources relative to the project, read on 14 October 2026:

- `RCSIM_MCS/components/HardwareConfig.tsx`: protocol, port and UPS choices.
- `RCSIM_MCS/app/page.tsx`, around lines 463–495 and 911–1159: hardware saves and programming workflow.
- `RCSIM_MCS/src/main.py`, lines 195–227: output selection and baud-rate settings.
- `RCSIM_MCS/src/output/bridge.py`, lines 13–51; `crsf.py`, lines 95–155; `mavlink_rf.py`, lines 320–374: channels and pause behavior.
- `RCSIM_MCS/src/api/server.py`, around lines 160–195, 333–345, 594–604, 621–1007 and 1202–1248: networking, saves, firmware and Bluetooth.
- `RCSIM_MCS/src/hardware/battery.py`: INA219 monitor.
- `RCSIM_MCS/components/BluetoothConfig.tsx` and `src/hardware/bluetooth_manager.py`: scanning, pairing, trusting and connecting.
- `RCSIM_MCS/components/WiFiConfig.tsx` and `src/hardware/wifi_manager.py`, around lines 224–380: modes, switching and save limitations.
- `RCSIM_MCS/components/CoprocessorTab.tsx`: target/file fields, RESET dialog and consoles.
- `RCSIM_MCS/lib/translations.ts`: Polish/English button labels.