



WIBOTIC™

9706 4th Ave NE- Suite 208

Seattle, WA 98115

206-580-09007

<http://www.wibotic.com>

info@wibotic.com

Prepared by: WiBotic Inc

Date Updated: Revision 11.0 – July 14, 2020

Notice: This document is provided for informational purposes only. It represents WiBotic's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessments of the information in this document and any use of WiBotic's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from WiBotic.

© 2020, WiBotic Inc. All rights reserved.

Contents

Description.....	6
Key Features	6
Development Kit Configuration	7
Inside the Box	7
Specifications.....	8
Power Level.....	8
Initial Hardware Setup.....	10
Connecting the Components	11
Battery Power Cable	13
Auxiliary Power Input for Onboard Charger	14
Configuring Ethernet Connectivity.....	16
Windows 7/8/10.....	16
Linux (Temporary Connection)	18
Linux (Permanent Connection) or Another OS	19
Chromebook.....	19
Ethernet Configuration Reset	20
Web GUI Functions.....	22
WiBotic Control Panel	22
“Settings” Menu	24
Battery Settings	24
Charge Settings	28
Network Settings.....	30
CAN Settings	31
“Diagnostics” Menu	31
Live Logs	31
Version Information.....	32
Log Messages	33
“Update” Menu	33
Upload Button.....	34

Update Button.....	35
“Data Plots” Menu	38
“About” Menu	38
Wireless Power Transmission	39
About the Battery Charging Process.....	40
Distance Connect Mode	41
Power-Down Events.....	43
Troubleshooting GUI Connectivity.....	43
Suppliers Declaration of Conformity (SDoC)	44
APPENDIX A: NETWORK API.....	46
Getting Started	46
General Packet Format	47
Request Packet Types.....	47
Response Packet Types	47
API Requests	47
Building API Requests	48
Parsing API Responses	48
Parameters.....	49
Parameter Status Codes	52
Real-time ADC Packets.....	52
Device ID.....	53
APPENDIX B: PYTHON LIBRARIES	54
Introduction	54
Setup.....	54
Verify Sample Code Operation	54
Python Development	55
Provided Code	55
APPENDIX C: ONBOARD API	57
CAN Interface.....	57
USB to CAN	58

CAN API.....	58
Basic Setup	59
Talking over CAN.....	60
Provided Code	62
Serializing Packets	62
WiBotic DSDL Definitions	62
Parameters	63
Parameter Status Codes.....	65
UAVCAN GUI Tool	65

Description

WiBotic wireless power solutions enable autonomous wireless charging of mobile robots, aerial drones, unmanned aquatic vehicles, and other industrial automation devices across a wide range of applications. For simplicity and consistency, we refer to all of these vehicles, drones, and devices as “robots” for the remainder of this User Guide.

The WiBotic Development Kit is the first step in automating power delivery to robots or entire robot fleets. In addition to fully automating the battery charging function, remote monitoring of battery parameters and configurable charging profiles can extend the life of every battery. This configurability helps to ensure every robot is ready to take on its next mission.

The Dev Kit consists of standard WiBotic components selected to meet customer needs for size, weight, and power as well as maximum charging range. The WiBotic Web-Based Graphical User Interface (GUI) and Application Programming Interface (API) are also supplied for system configuration and monitoring purposes.

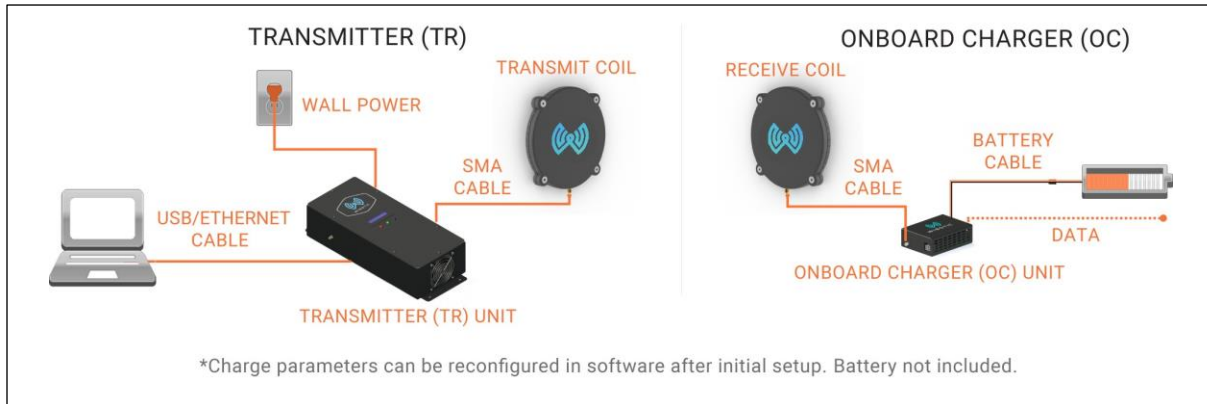
This User Guide describes setup and operation of the Dev Kit and is applicable to all additional production systems.

Key Features

- ✓ **100% autonomous wireless battery charging**
 - Automatically starts to charge when robot approaches transmitter
 - Wide transmit-receive coil range and alignment flexibility
 - Multiple configurations from 90W to 300W to provide the best combination of size, weight and power for each application
- ✓ **Flexible charging modes**
 - Supports most common robot battery chemistries including LiPo, LiFePO₄, and SLA
 - Constant current/constant voltage charging modes for lithium chemistries and a programmable mode for non-standard batteries
 - Fully programmable charging current and voltage via the GUI or API
- ✓ **Optional wired battery charging**
 - Multiple options up to 30A/300W delivered to the battery when used with a DC power supply for non-wireless operation
- ✓ **Lightweight Onboard Charger system for weight-sensitive installations**

Development Kit Configuration

The following diagram shows the operational configuration of the system as well as the individual components supplied with every WiBotic Development Kit:



Inside the Box

- | | |
|----------------------------|---------------------------------|
| 1. Ethernet Cable | 6. Receive Coil |
| 2. USB to Ethernet Adapter | 7. Transmit Coil |
| 3. Power Cable | 8. Coaxial SMA Cable (x2) |
| 4. Power Transmitter (TR) | 9. Battery/Load Connector Cable |
| 5. Onboard Charger (OC) | |



Specifications

All WiBotic Wireless Power systems share the following standard specifications:

PARAMETER	RANGE
Wireless Power Operating Frequency	6.78 ± 0.015 MHz
Communication Link Frequency	2.433 - 2.483 GHz
Ambient Operating Temperature	0° to 40° C
Transmitter AC Input Voltage*	90 - 264 VAC
Transmitter AC Input Frequency	50 – 60 Hz
Transmitter Data Port	Ethernet (RJ45)
Transmitter Data Communications	WiBotic API via WebSockets
Onboard Charger Auxiliary DC Input Voltage	22 - 50 VDC (48V preferred)
Onboard Charger Data Communications	WiBotic UAVCAN API over CAN-bus

*The standard WiBotic Development Kit includes an AC/DC power supply built into the transmitter. Direct DC input versions are also available. Contact WiBotic for details.

Power Level

Development Kits are available in configurations that combine Transmitters and Onboard Chargers to provide the best combination of size, weight and power for each application. The following table shows the available combinations for each desired power level. All systems utilize the same 200mm Transmitter and 100mm Receiver antenna coils unless custom antenna sizes/shapes are requested.

Up to:	90W	125W	250W	300W
Transmitter	TR-110	TR-110	TR-300	TR-300
Onboard Charger	OC-110	OC-210	OC-250	OC-300

Onboard Chargers (OCs) are paired with the correct Transmitter to achieve their maximum power level, so the OC ultimately determines the amount of power delivered to the battery. Each OC has a different maximum power level (watts), maximum current level (amps), and output voltage range as shown in chart below:

ONBOARD CHARGERS	OUTPUT VOLTAGE (TO BATTERY)*	MAX OUTPUT POWER	OUTPUT CURRENT
OC-110	7.92 - 30.1 VDC	90W	0.5 - 5A
OC-210	12.03 – 36.0 VDC	125W	0.5 - 10A
OC-250**	12.03 – 36.0 VDC	250W	0.5 - 10A
OC-300	0 - 58.4 VDC	300W	0.5 - 30A

*DC Output voltage and current is configurable within the specified range via WiBotic software or API.

**The OC-250 can be configured for higher output voltage up to 58.4V if required.

The total amount of power delivered to the battery is determined by both the maximum power output and the maximum current output, based upon the voltage of the battery being charged. Since there is a straightforward equation that relates voltage, amperage and wattage (*volts x amps = watts*) we can use simple calculations to determine which factor will ultimately determine your system’s power level and charge speed:

Example:

The user wishes to configure the OC-110 to charge a battery that reaches 12.6V when fully charged. In this case, the OC-110 will reach its 5A limit before the 90W limit is reached, and power delivery will not exceed 63W.

$$12.6V \times 5A = 63W$$

A different user wishes to configure the same OC-110 to charge a battery that reaches 25.2V when fully charged. In this case, the OC-110 will reach its maximum 90W limit before it can reach the 5A limit. Power delivery will not exceed 90W and amperage will be a maximum of 3.57A.

$$90W / 25.2V = 3.57A$$

Note that the above values are representative only, and other factors such as antenna coil position can also affect system efficiency and power delivery.

Finally, there is no risk in inadvertently setting the desired charge current above the maximum level. In that case, the system firmware will limit operation to the maximum power level that is safely achievable.

After fully reading this User Guide, please contact WiBotic if there are any questions about the maximum power level available for your specific system.

Initial Hardware Setup

For initial system familiarization, WiBotic highly recommends bench-top testing. This allows the system to be operated in open air and at various power levels and antenna positions – demonstrating the unique flexibility of WiBotic’s technology.



Note: WiBotic does not recommend immediate installation of the wireless power system on robots for testing purposes. Many robots consist of metallic frames or body parts that can temporarily de-tune wireless power antennas, resulting in degraded performance. To avoid de-tuning problems and to maximize system performance, WiBotic offers integration services for customers who are ready to test the system on their robot. Contact WiBotic for details.



For bench-top testing WiBotic provides an antenna stand to new customers. Shown at left, the stand includes mounts for both the transmit and receive antenna coils as well as a “T” shaped base. Sliding the antenna brackets on the base allows power transfer to be tested at different distances (both face-to-face and side-to-side) and angularities.

First, install the two brackets on the “T” shaped base with the thumb screws facing outward.

To install the transmit antenna, rotate it until the SMA antenna connector is facing sideways and slide it into the pocket on the bracket. The bottom two bosses on the antenna enclosure should seat firmly into the rounded openings on the bracket.

It is not necessary to screw the antenna in place, but for a more secure mounting you may substitute the bottom two antenna enclosure screws for longer M5 screws. Thread them through the antenna enclosure and into M5 nuts placed in the small capturing pockets on the back side of the bracket.

When finished, the vented back side of the transmit coil enclosure should be facing outward (on the same side as the thumb screw) and the embossed WiBotic log should be facing toward the receiver coil.



Repeat the process for the receiver antenna – again using longer M3 screws to more securely affix the antenna to the bracket. Note that the receiver antenna enclosure does not have a vented back side. When mounting this antenna, be sure that the side of the enclosure with the embossed WiBotic logo is facing the transmitter antenna.

To move the antennas relative to one another, simply loosen the thumb screws and slide both antennas within the T-Base channels. Now loosen the thumb screws and insert the head of each thumb screw bolt into the slots on the T-base.

Please do not remove the antennas from their 3D printed enclosures without consulting with WiBotic. **High voltage on the antenna coil PCBs can cause burns if touched during power transmission.** However, if removing the PCB antennas is required, be sure to mount them with the coil sides facing inward toward each other. The side of the coils with the exposed capacitors is the back side, and the back sides of both coils should be facing outward.

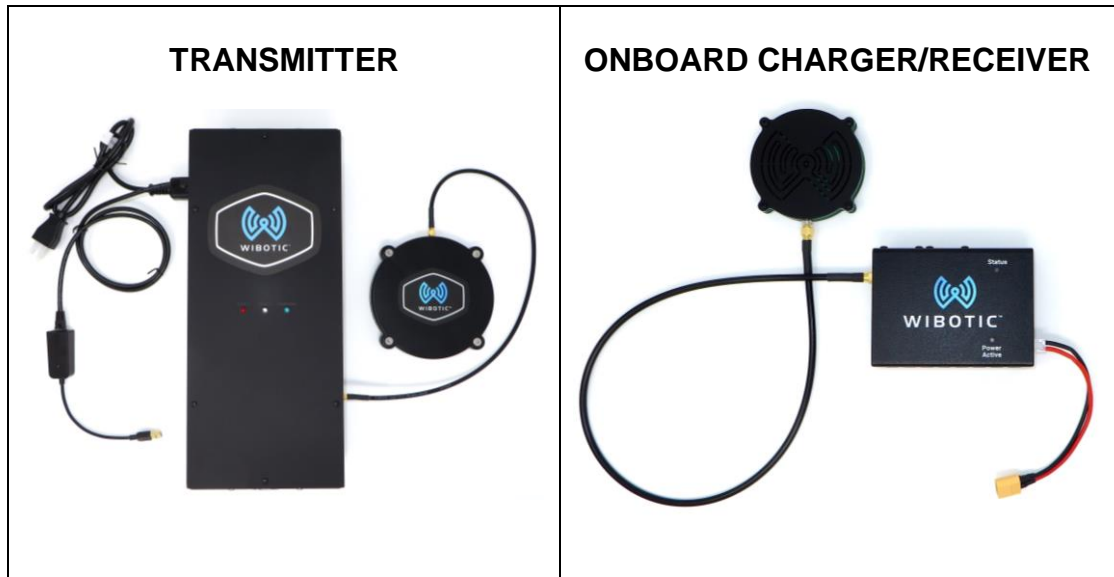
Once the antenna stand is set up, proceed with installing the antenna cables.

Connecting the Components

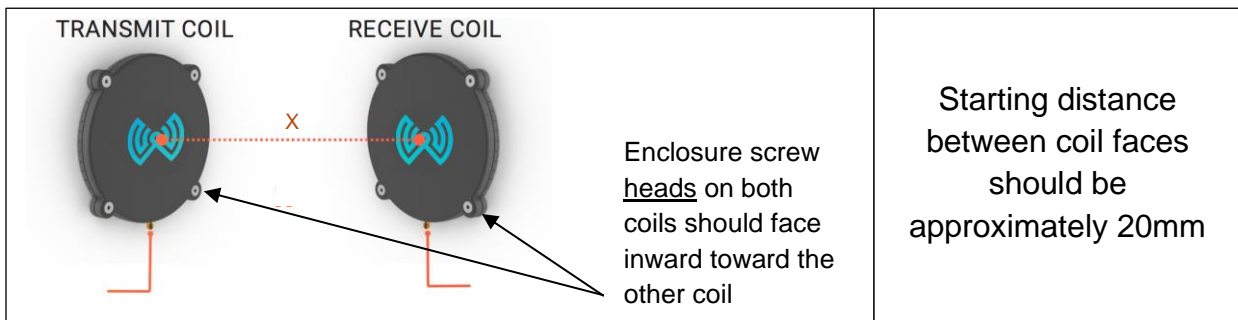
1. Connect one coaxial SMA cable between the Power Transmitter and the Transmit Coil.
 - a. For TR-110 systems, make sure the connection is tightened to 8 inch-lbs. on both ends.
 - b. For TR-300 based systems, hand tighten the larger type “N” connector on the transmitter itself, and tighten the SMA connector on the transmit antenna end to 8 inch-lbs.
2. Connect the second coaxial SMA cable between the Onboard Charger and the Receive Coil. Make sure the connection is tightened to 8 inch-lbs on both ends.
3. Connect the Onboard Charger to the target device to be powered (e.g. the battery) using the supplied battery connector cable. **NOTE:** *The development kit requires a direct connection to the battery for operation. The “heartbeat mode” algorithm is disabled if the Onboard Charger does not receive power from the battery before wireless power is enabled.*



CAUTION: If using a battery power cable other than the one supplied by WiBotic, be sure to follow the pin-out instructions in the following section.



4. With the antenna coils mounted on the coil stands, loosen the thumb screws and adjust the coil enclosures so they are concentric and approximately 20mm apart. This will be the starting position for testing, but you may then move the coils in and out and side to side during testing. You may also twist one or both coils to experience power transfer when the coils are not parallel. **NOTE:** *It may be possible for the coils to be placed too close together. Please reference your system's Certificate of Conformance (CoC) for optimal coil spacing (x in diagram below). The CoC provides the distance between plastic enclosure faces only. If you intend to remove the antenna coils from their enclosures, please contact WiBotic for the correct spacing.*





CAUTION: Do *not* place the Transmit Coil on a metallic surface, and make sure that all metal objects are at least 30cm away from the Transmit and Receive Coils during testing.



CAUTION: Removing the antenna coils from their enclosures is not recommended without guidance from WiBotic. High voltage exists on antenna coil components during charging. **DO NOT** touch any portion of a bare antenna coil during charging.

5. Connect the Power Cable to the Power Transmitter (before plugging in the Power Cable to an electrical outlet or other power source).
6. Plug the other end into a standard electrical outlet. The Green “READY” LED on the Power Transmitter will only turn on after the system has been activated via the software GUI (see later steps) but the LCD screen will become active.
7. Connect the Ethernet cable from the Power Transmitter to a computer. Either plug directly into the Ethernet port on the computer (if available) or use the provided Ethernet-to-USB Adapter. There are additional steps required to configure the IPv4 settings on Windows. Refer to the following section on Configuring Ethernet Connectivity to properly configure the Ethernet connection before using the system.

Battery Power Cable

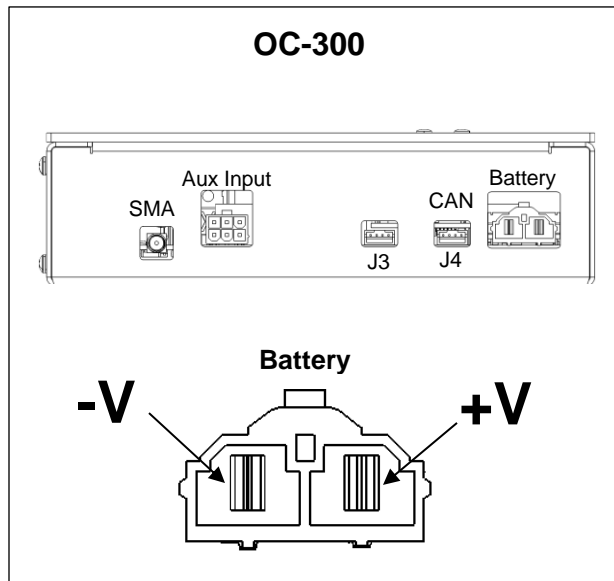
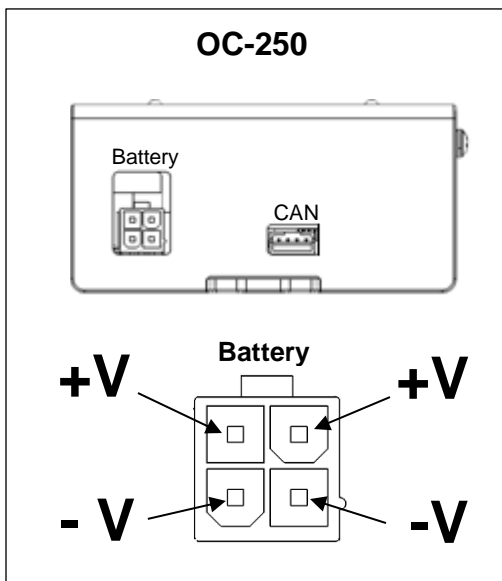
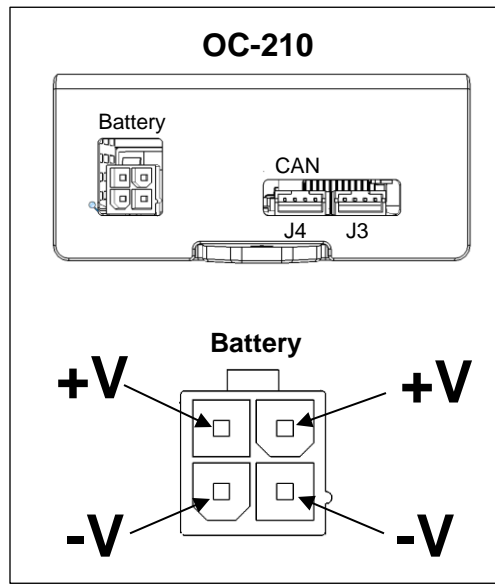
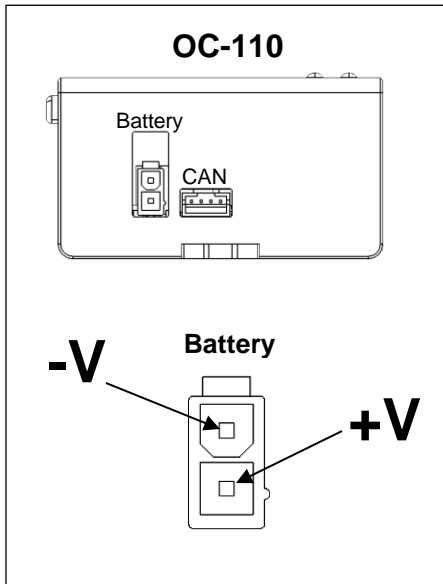
The battery output connectors and cables on WiBotic’s Onboard Chargers are sized for the OC’s maximum rated current. If not using WiBotic’s battery cable, the following information can be used to select the proper connector and wire gauge. Terminal pin-outs are also provided for reference.

Battery Cable Connectors

The following connectors, crimps and wire gauges will mate with the battery power output connectors on WiBotic OCs. See the following page for pin-out information.

OC Model	Connector Brand	Housing Part #	Crimp Part #	Wire Gauge
OC-110	Molex	39-01-2020	45750-3111	16AWG
OC-210	Molex	39-01-2040	45750-3111	16AWG
OC-250	Molex	39-01-2040	45750-3111	16AWG
OC-300	Molex	42816-0212	42815-0134	8 AWG (24A max)

The following diagrams show the various connectors on each OC and the pin-out of each male battery output connector. Use these diagrams to properly wire the corresponding female connectors.



Auxiliary Power Input for Onboard Charger

WiBotic Onboard Chargers are primarily designed as wireless power receivers. However, there may be times when it is desirable to use the OC as a wired (i.e. “plug in”) charger. For example, if a battery dies when the robot is a long distance

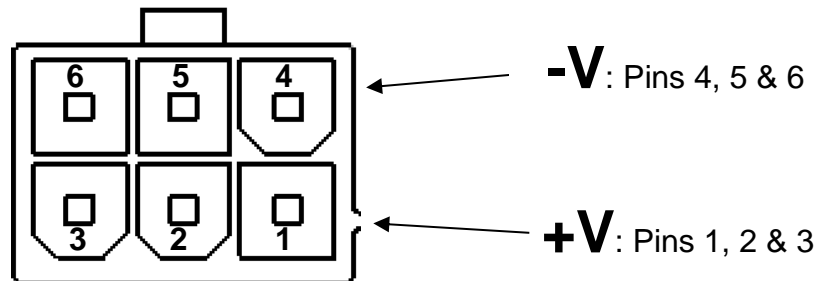
from the transmitter it may be easier to charge robot in place rather than manually pushing it back to the transmitter station.

Also, for some customers the WiBotic OC may be used as a programmable plug-in charger at first and then upgraded to wireless charging in the future by adding a receiver antenna and transmitter station.

For these reasons, the OC-210, OC-250 and OC-300 all have a six-pin Molex Mini-Fit Jr. auxiliary DC power input port. This port is typically located near the coaxial SMA input and is the only 6-pin input on the OCs. (Note that the OC-110 does not have this option.)

To power the OC outside of wireless power range, use this port to plug in any 18-50V DC power supply rated for at least 150% of the power level (watts) you require as output from the OC.

If adapting an existing DC power supply, the pinout for the male connector on the OC is shown below.



You will need a matching Molex Mini-Fit Jr female connector (Molex Part No: 0039012060) on the power supply.

Note that this input is current-limited at 8A for the OC-210 and OC-250 and 24A for the OC-300. This means, for example, that a 24V power supply will result in a maximum charging power of 192W when using the OC-250. For this reason, we recommend using a 48V supply with a sufficient power rating for the battery you wish to charge.



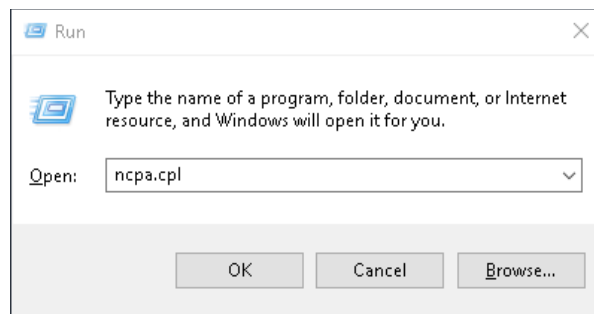
Note that the 6-pin Auxiliary Input connector is similar to the 4-pin (OC-210, OC-250) Molex connectors used for the battery cable. DO NOT plug the battery cable into this port. Damage to the Onboard Charger circuits may result if the aux port is connected to a battery.

Configuring Ethernet Connectivity

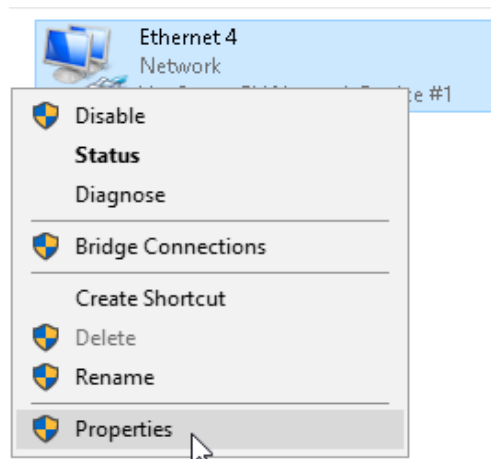
The WiBotic Web GUI allows users to control the WiBotic system, view its status, and make changes to settings without having to install separate software. The interface is accessible via modern versions of Firefox, Chrome, Edge, and Safari. A direct point-to-point Ethernet connection is recommended for initial setup although networking is possible through the Network Setting menu. The Power Transmitter serves the WiBotic Web Graphical User Interface (“Web GUI”) on a specific IP address configured on HTTP port 80.

Windows 7/8/10

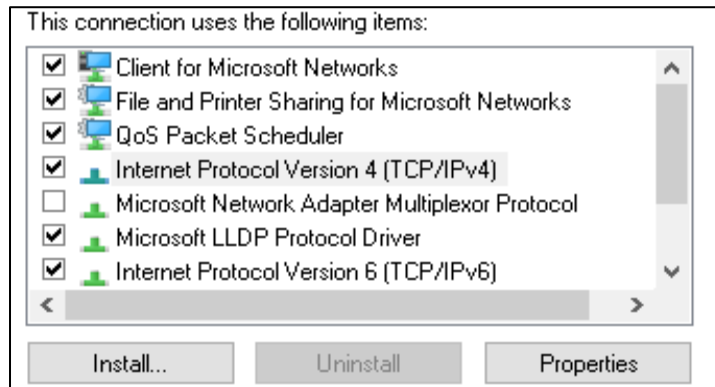
1. With the Power Transmitter powered on, confirm that the Ethernet Cable is connected to the computer either directly, or via the provided USB Adapter.
2. Open “Network Connections” by pressing **Windows Key + R**. Type in “ncpa.cpl”, and click OK:



3. Right click on the network adapter that is connected to the WiBotic Power Transmitter and select **Properties**:

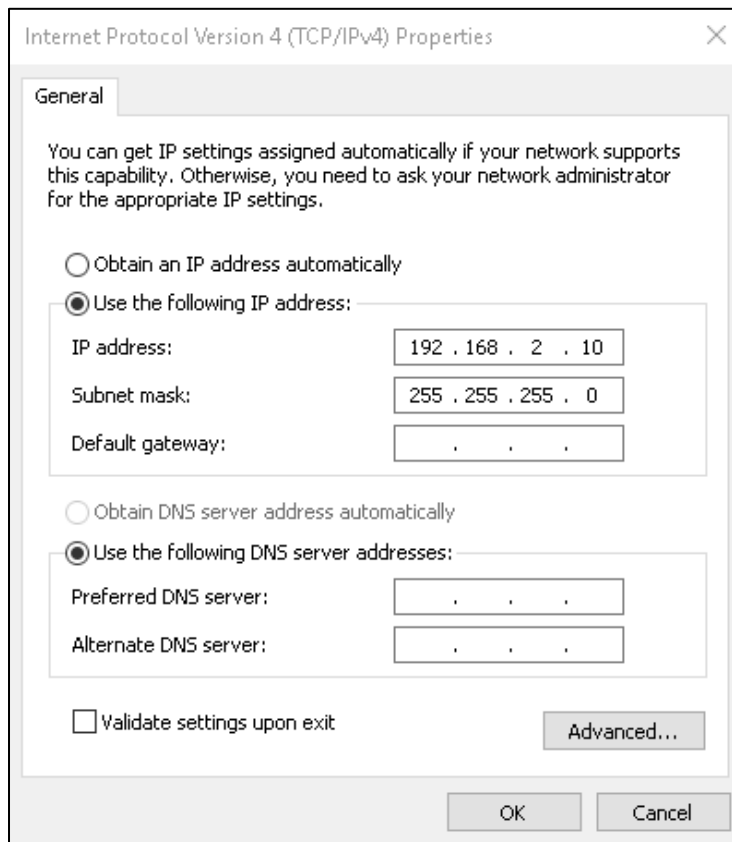


- Select **Internet Protocol Version 4 (TCP/IPv4)** and click the **Properties** button below it:



- Click the **Use the following IP** address button and specify the following settings. The DNS section may be left blank. Click OK:

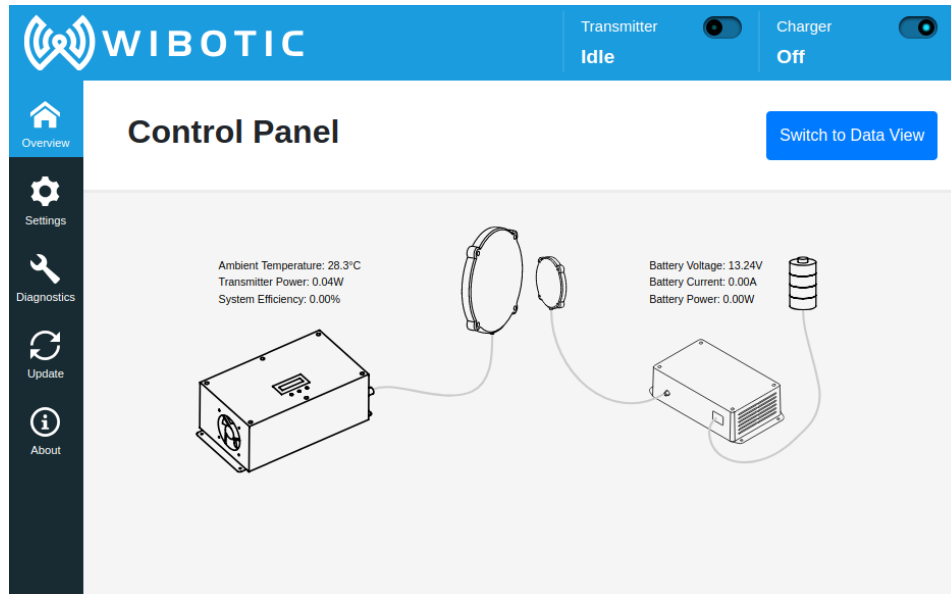
IP address:	192.168.2.10
Subnet mask:	255.255.255.0
Default gateway:	Leave Blank



6. After the network settings have been configured, **open a Web Browser** (Firefox, Chrome, or Edge) and go to the following URL:

<http://192.168.2.20>

7. Verify that the web application loaded and that the Transmitter status at the top of the screen shows “Idle”.



Linux (Temporary Connection)

1. Open a terminal window (many desktop environments have a keyboard shortcut CTRL+ALT+T to do this).
2. Type “ip address” to list all network interfaces on your computer and their respective IP addresses.
3. Locate the name of the network interface connected to the WiBotic system. This interface is usually near the bottom if using a USB dongle. It should be something like *enx0012345678*, *eno1*, or *eth0*.
4. Type “sudo ip addr add 192.168.2.10/24 dev *enx0012345678*” assuming you’re running a system that uses sudo to elevate permissions. Replace *enx0012345678* with the name of the network interface identified above.
5. Visit <http://192.168.2.20> in a web browser (Firefox, Chrome).
6. Verify that the web application (shown above) loaded and that the Transmitter status at the top of the screen shows “Idle”.

Linux (Permanent Connection) or Another OS

Please refer to the network setup guide for your operating system. You will need the following information:

Static IP (for the computer you are setting up):	192.168.2.10
Static IP (for the WiBotic transmitter):	192.168.2.20
Subnet Mask:	255.255.255.224 (/27) or larger

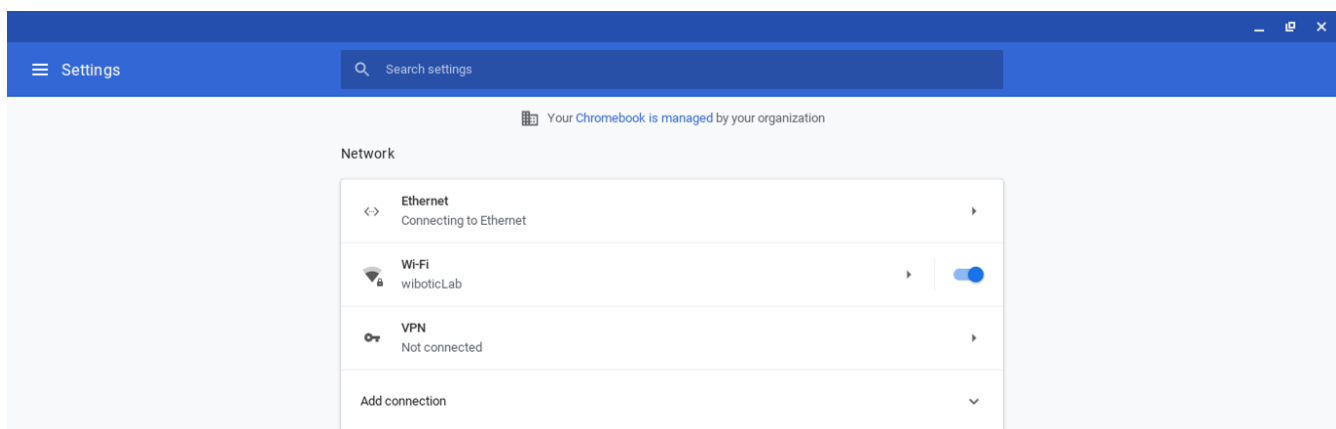
Additionally, if your OS has an option like “Use network only for resources on its own network”, this option should be checked for the network connection used for a direct Ethernet connection to the WiBotic hardware.



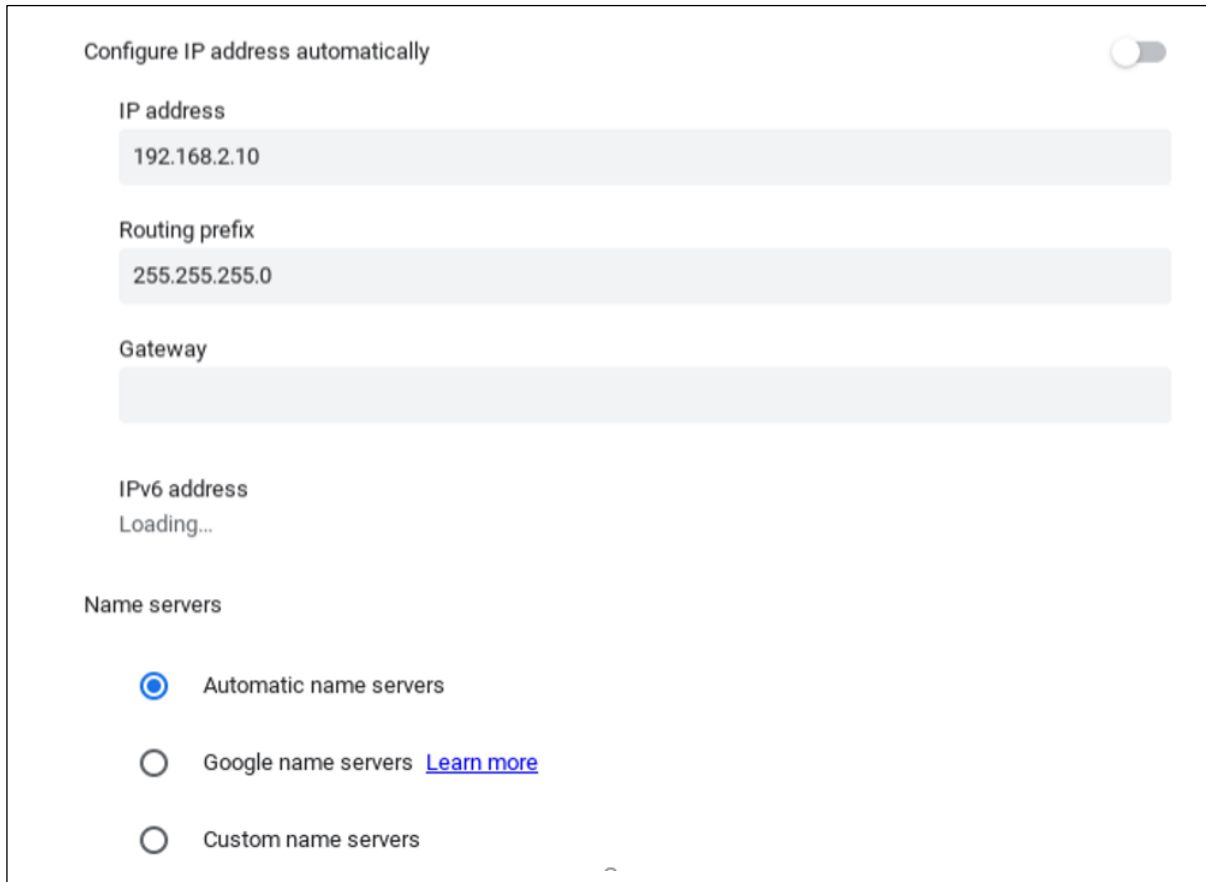
Note: All Ethernet connectivity instructions are intended for an ad-hoc point-to-point connection directly to the Development Kit. Changing the IP address, default gateway, and subnet mask of your default adapter may cause a loss of connectivity between your computer and your local network.

Chromebook

1. Plug in the Ethernet cable from the WiBotic transmitter.
2. Go to Settings and click “Ethernet”.



3. In the following screen, de-select the “Configure IP address automatically” toggle switch, and enter in the IP address, Routing prefix, and Gateway as shown below.



The screenshot shows a configuration interface for IP address settings. At the top, there is a toggle switch for "Configure IP address automatically" which is currently turned off. Below this, there are three input fields: "IP address" containing "192.168.2.10", "Routing prefix" containing "255.255.255.0", and "Gateway" which is empty. Underneath these is the "IPv6 address" section, which shows "Loading...". At the bottom, there is a "Name servers" section with three radio button options: "Automatic name servers" (selected), "Google name servers" (with a link to "Learn more"), and "Custom name servers".

4. Open a browser (Chrome) and navigate to <http://192.168.2.20> to access the WiBotic GUI.

Ethernet Configuration Reset

If the WiBotic system does not seem to be responding on the default IP address, you can attempt to reset the Ethernet configuration to the default factory settings.

1. Locate a thin rigid object such as a small hex key or an unfolded paperclip.
2. Gently insert the object into the hole located on the Transmitter just below the Ethernet jack.



3. Upon inserting the thin rigid object, you should feel a button that depresses about 1mm. Hold the object on that button for more than 5 seconds.
4. Upon releasing the button, the LCD screen (if equipped) on the front of the transmitter will turn magenta and display “Ethernet Config Reset”.



Note: If the LCD only turns white and displays network configuration settings after being released, the button did not detect a continuous 5 second press and the Ethernet settings have not been reset. Try again.

Power cycle the transmitter for the new network settings to take effect.

Web GUI Functions

Once the network connection has been established, all system cables have been connected, and the antenna coils are within the ideal operating range, you're ready to open the Web GUI to begin experiencing WiBotic wireless power!



Important Note: Once again, testing the system on a benchtop is highly recommended as a first step. This will ensure the system is operating correctly at the full range and power level it can provide. Because nearby conductive materials (like metal robot chassis panels) can de-tune the system's antennas, it is not recommended that you test the system on your robot without first consulting WiBotic. Detuned antennas may result in poor performance or damage to some wireless power components. Please contact WiBotic to discuss testing on your robot and the possible need for coil tuning services.

At this time, open the Web-GUI using the following IP address typed directly into your web browser: <http://192.168.2.20>

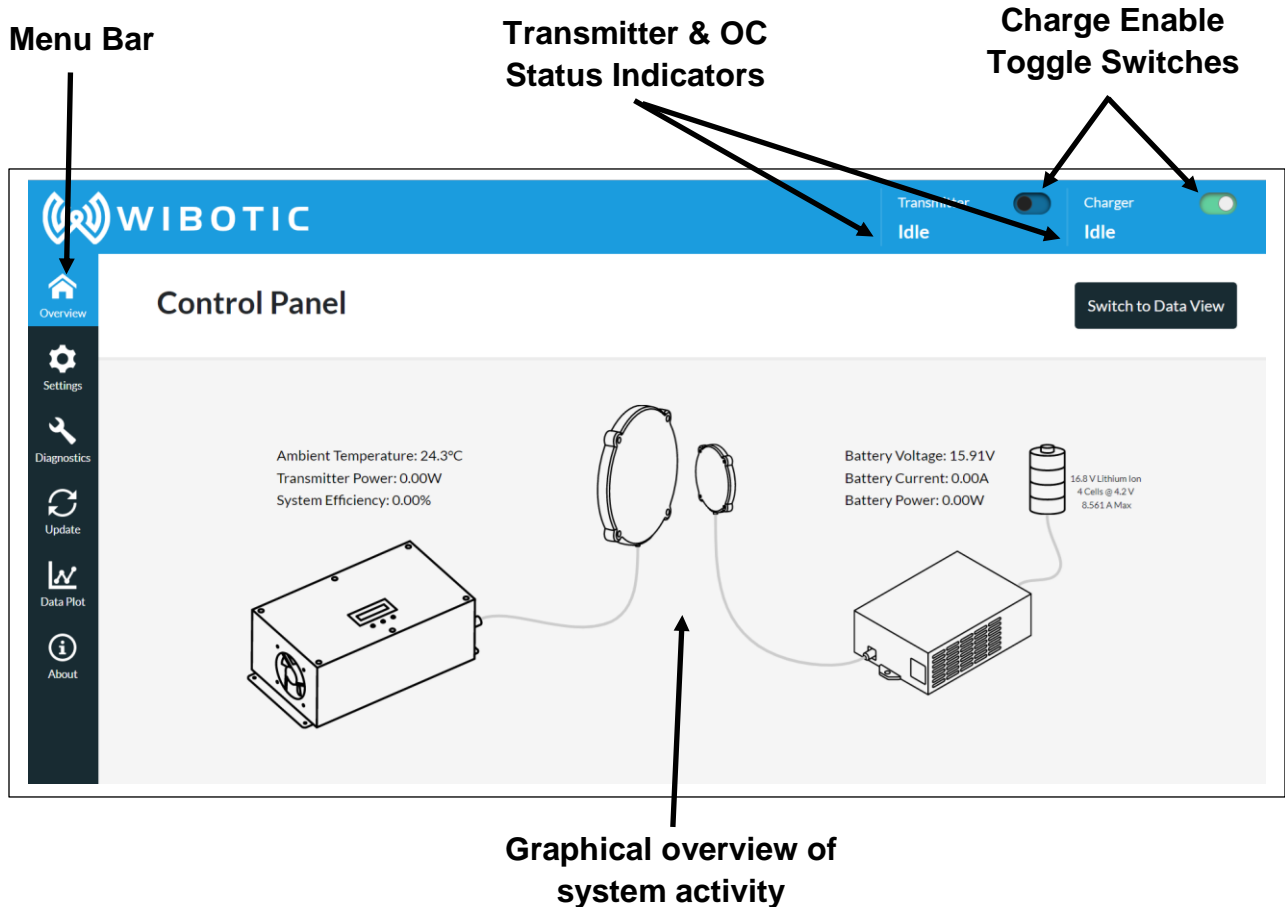
As the default operating mode, wireless power will be disabled (OFF) when the Transmitter is initially powered up. Therefore, the GUI must be opened, and the system must be manually turned ON to begin charging. Later, the default operating mode can be changed to have the system automatically turn ON whenever a robot is in range. See the "System Settings" menu section later in this document for details.

Before starting, please review the rest of this document to familiarize yourself with Web GUI application and overall system operation.

WiBotic Control Panel

The image below shows the primary Control Panel interface within the WiBotic Web GUI. This display provides a graphical representation of the Transmitter and Onboard Charger (if in range) and their current operational state. If no Onboard Charger is within range, "See Available" will appear in the Charger Status area at the top right. Clicking on this button allows operators to see all Onboard Chargers currently within radio range

of the Transmitter (about 30ft). While wireless power obviously cannot be sent to a Charger in this state, the Transmitter is able to see the status of the Charger and connected battery, change settings, and perform firmware updates. This is called “Distance Connect Mode” (see section later in document).



The Transmitter and Onboard Charger images in the center of the screen represent the specific models in use. If the GUI is used to log into a different Transmitter model, or if different Onboard Charger models approach the same transmitter, the images will change accordingly. The Onboard Charger image will only appear when the robot is within charging range and an antenna “handshake” has occurred to confirm it is ready for charging.

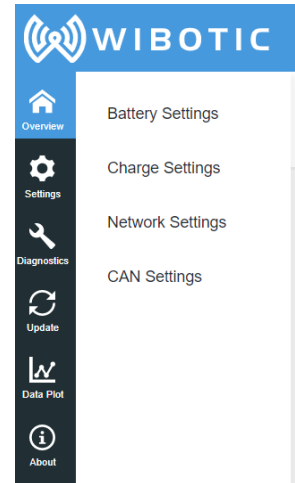
When power transmission begins, the wires and/or wireless “waves” between the components and antennas will becoming active. After further system checks, the link between the OC and battery will also become animated – signaling that charging has begun.

Live data related to each component is displayed on the Control Panel, and battery settings are shown just to the right of the battery image. A more detailed view of all parameters is accessible via the “Switch to Data View” button in the upper right corner of the main display.

“Settings” Menu

The Settings menu provides four sub-menus that allow for customized system operation:

1. **Battery Settings:** provides support for a wide range of battery chemistries, voltages, and current levels.
2. **Charge Settings:** allows for deeper configuration of autonomous functions.
3. **Network Settings:** allows for network connections to transmitters over Ethernet.
4. **CAN Settings:** Provides CAN-bus communications settings for the Onboard Charger’s optional CAN interface.



Battery Settings

Prior to shipping the system, WiBotic will pre-configure the battery chemistry, number of battery cells, per-cell charging voltage (which begins as a default based upon battery chemistry) and maximum charge current based upon the information you provide. Other settings will also be configured at factory default levels. Begin by opening the “Battery Settings” sub-menu within the main “Settings” menu to **confirm** that these values are correct for the battery you are using.

If you have switched batteries or the WiBotic system is installed on a different robot later on, these values can be adjusted at any time. Simply use the Battery Configuration screen to select the proper “**Battery Chemistry**” and “**Number of Cells**”. A default “**Volts per Cell**” will be displayed based upon the chosen chemistry, but this value can also be manually overridden. “**Float Voltage**” is a calculated value based upon the number of cells and per-cell voltage and displays the overall maximum charge voltage the system will target for the robot’s battery or battery pack. If the available options do not provide the correct voltage for your system, a “Custom” battery chemistry option is also available.

Charging stops when the float voltage for the pack is reached and the current being delivered to the battery drops to 1/10th of the maximum current setting.

It is normal for battery voltage to sag immediately after charging stops, and it would be impractical for the system to immediately resume charging to again reach the float voltage. Therefore, a default “**Restart Threshold**” is built into the system based upon known characteristics for various battery chemistries. Once battery voltage drops to this restart threshold (typically due to an active load on the battery or self-discharge) the WiBotic system will automatically resume charging until the float voltage is reached once again. The default restart threshold can be overridden if necessary using the available input box, but we recommend maintaining the default value unless you are extremely familiar with your battery’s performance characteristics.

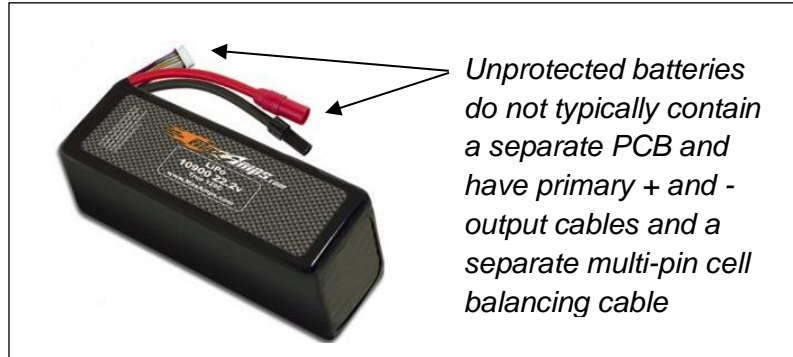
Battery Chemistry	Lithium Ion
Number of Cells in series	4
Volts per Cell	4.2 V
Float Voltage	16.8 V
Restart Threshold	16.3 V
Maximum Charge Current	8.561 A
Enable Recovery Charge <small>Charge slowly during undervolt conditions. Ensure battery support before enabling.</small>	<input checked="" type="checkbox"/>

Save

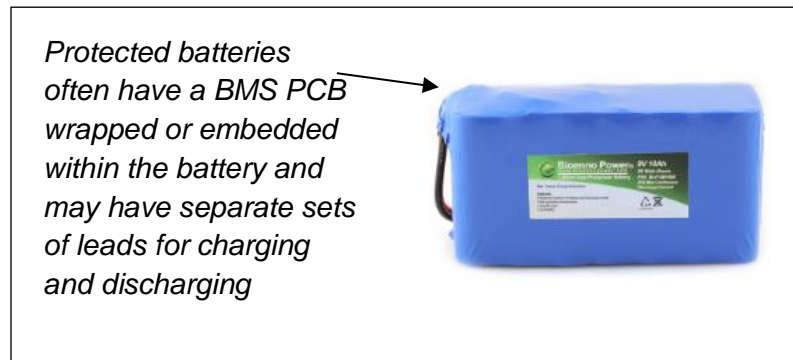
The “**Maximum Charge Current**” value is typically set at the WiBotic factory based upon customer input and the battery being used. However, it can be increased or decreased within the overall power range of the WiBotic system. For instance, the OC-250 has a maximum power rating of 250W. As an example, when charging an 8S LiFePO4 battery to a float voltage of 29.2V, the Maximum Charge Current can be calculated as $250W/29.2V = 8.56A$. If the user attempts to increase charge current above this value, an error message is displayed, and the value will reset back to the maximum. However, a *lower* battery current can be specified at any time for better management of battery lifespan. (With many lithium chemistries, charging batteries at the maximum allowable rate can significantly decrease the total number of available charge cycles. Therefore, this setting can be used to slow down battery charging when fast charging isn’t needed. Charge speed can then be increased again when the robot needs to quickly return to service. Using the WiBotic API to programmatically make

these adjustments based upon the robot's duty schedule lets users truly maximize the lifespan of entire fleets of batteries.

WiBotic charging systems can be used with a wide range of battery types – including a range of both protected and unprotected Lithium chemistries. The **“Enable Recovery Charge”** mode is particularly useful when Protected batteries are used.



Unprotected Lithium batteries have no onboard circuitry to avoid overvoltage or undervoltage conditions. This can be very dangerous if not properly managed. The WiBotic Onboard Charger inherently provides overcharge protection through the use of the “float voltage” setting. The OC constantly monitors battery voltage and will automatically stop charging once the safe float voltage is reached and the current being delivered to the battery drops to 1/10th of the Maximum Charge Current setting.



However, the WiBotic charger cannot control battery discharging.

Know the type of battery you are using!

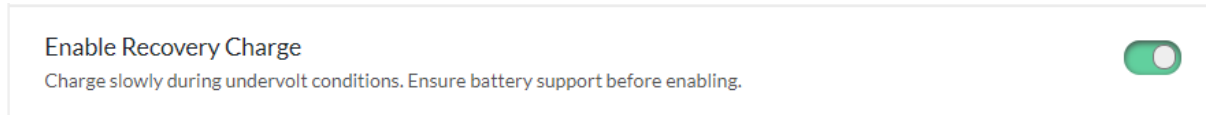
It is possible that overuse of the robot or drone could lead to a dangerous undervoltage condition when unprotected batteries are used. During undervoltage conditions the battery cells may become damaged. It is therefore not recommended that a normal high-current charge cycle be attempted after an undervoltage event.

The WiBotic OC is designed to avoid that possibility. By default, the OC will not attempt to charge a battery that reads below its minimum safe voltage. In fact, if battery voltage has dropped below 5V, then the OC will no longer have enough power to communicate with the transmitter and will turn off completely.

Protected batteries operate differently. They are designed with internal circuits to automatically stop charging or discharging when overvoltage or undervoltage thresholds

are met. This protects the battery cells, avoiding potentially dangerous failures and extending overall battery life.

In these cases, since permanent damage to the cells has been prevented, it is acceptable to “wake up” the battery so charging can resume. This mode of operation can be allowed by activating the Enable Recovery Charge (ERC) toggle switch on the Battery Settings page.



By enabling this feature, you are signaling to the WiBotic charger that it is acceptable to send a low-level current to the battery even when it senses no (or very low) voltage from the battery. This effectively resets the BMS circuitry and allows the battery voltage to slowly and safely climb to the minimum threshold, at which point full-current charging can resume.

If you choose to enable the ERC switch, however, two important concepts must be understood:

- 1) You must only enable this feature if you are sure that your battery can support a low level charge when in protected mode. This feature is **NOT** designed for recovering **unprotected** batteries. If an unprotected battery is over-discharged, attempting to recover it in this way may lead to permanent battery failure or even battery fire.
- 2) When the WiBotic OC is powered only by the battery itself, which is the most common case, then it will likely lose power completely when the protection circuitry engages. This means charging will not be possible even if the robot/drone is manually brought within range of an active transmitter. To enable charging through the WiBotic OC, a secondary source of power for the OC is required. This can be supplied by a DC power supply connected via the six-pin Auxiliary Power input on most OCs (see Hardware Setup Section). Or, in certain robots and drones, a 5V USB connection may also be used.

Due to the somewhat complex nature of protected vs. unprotected batteries and connecting alternate power sources for the OC, we recommend contacting WiBotic for guidance before the ERC function is used.



WARNING: *Please confirm the battery requirements to ensure the charge current and voltage are within permissible limits for your battery. Damage to the battery, charger or connected equipment may occur if incorrectly configured. A damaged battery could cause fire or personal injury.*

Charge Settings

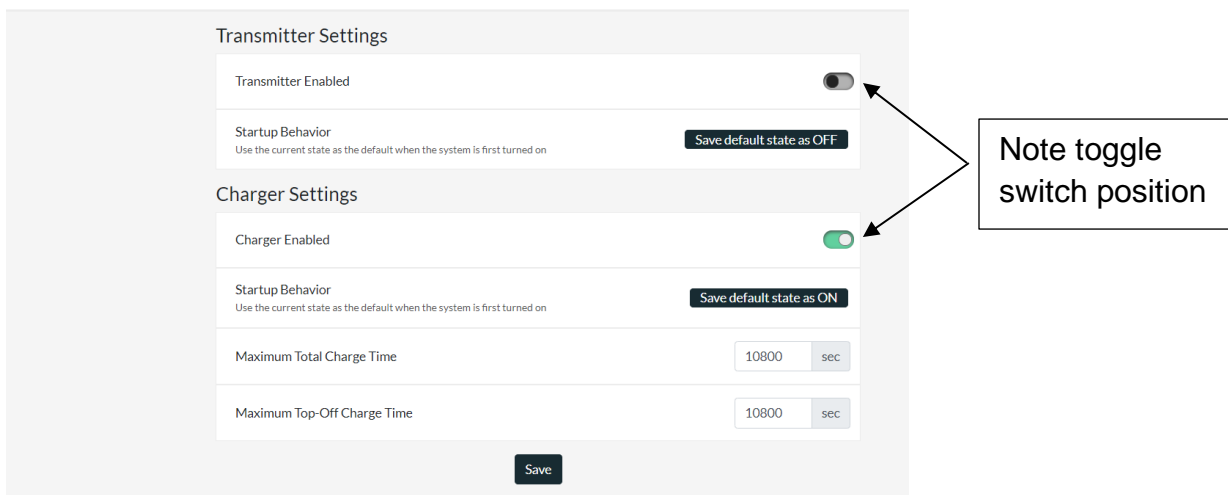
The Charge Settings sub-menu allows users to configure the system for autonomous operation. It also provides for “Charge Cycle” and “Top-off Period” time limits.

Autonomous Operation:

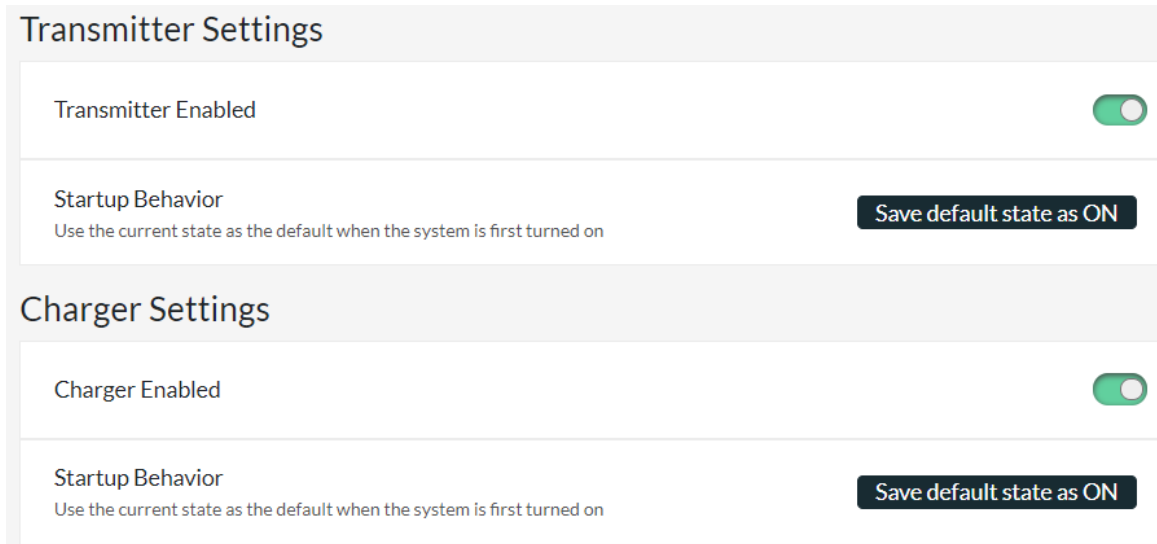
In the previous section we described how to start the charging process by manually clicking on the Transmitter and Onboard Charger software toggle switches in the upper right corner of the GUI. However, using the GUI to start charging may not be possible in all cases, or you may want the system to run fully autonomously without the need for a human to monitor and control it over the Ethernet connection. This capability is built into the system and is easy to enable.

To configure the system to start charging automatically when a robot approaches (and the battery is not already full), first navigate to the Charge Settings page and note the current status of the Transmitter Enabled and Charger Enabled toggle switches.

Charge Settings



In the above image, the transmitter is disabled/off so users will need to manually flip the toggle switch in the upper right corner of the Web GUI any time the system is repowered and they would like a robot to start charging. However, if fully automatic operation is desired, simply click on both the transmitter and charger toggle switches and then click the black “Save default state as...” button immediately below.



Transmitter Settings

Transmitter Enabled

Startup Behavior
Use the current state as the default when the system is first turned on **Save default state as ON**

Charger Settings

Charger Enabled

Startup Behavior
Use the current state as the default when the system is first turned on **Save default state as ON**

An alert will appear in the task bar confirming the change of settings. From this point forward, the transmitter and onboard charger will automatically turn ON and start charging a robot/drone as soon as it approaches a charging station. To disable this feature, simply toggle the Transmitter and Charger buttons back to the “OFF” position and click “Save default state as...” again.

Maximum Total Charge Time:

In some cases, it may be preferable to limit the total duration of a charge cycle so the system does not continue to transmit energy if the battery does not reach a “Battery Full” state. The Max Charge Time is set to 3hrs (10800s) by default, but users can adjust this value by entering a new value in the input box and clicking the “Save” button. The charge-time clock starts when the WiBotic transmitter first starts wireless charging and it will automatically turn off once the end of the time cycle is reached. If the battery is not fully charged after the system turns off, it will automatically turn back on and the timer will be reset.

Maximum Top-Off Charge Time:

Finally, the “Maximum Top Off Charge Time” is also a user-adjustable variable. This setting determines how long the charger will remain in “Constant Voltage” mode at the end of a charge cycle. Normally, current slowly drops during CV mode until it reaches 1/10th (C/10) of the Maximum Charge Current value. The charger then turns off and waits for battery voltage to drop to the “Restart Threshold” before starting to charge again.

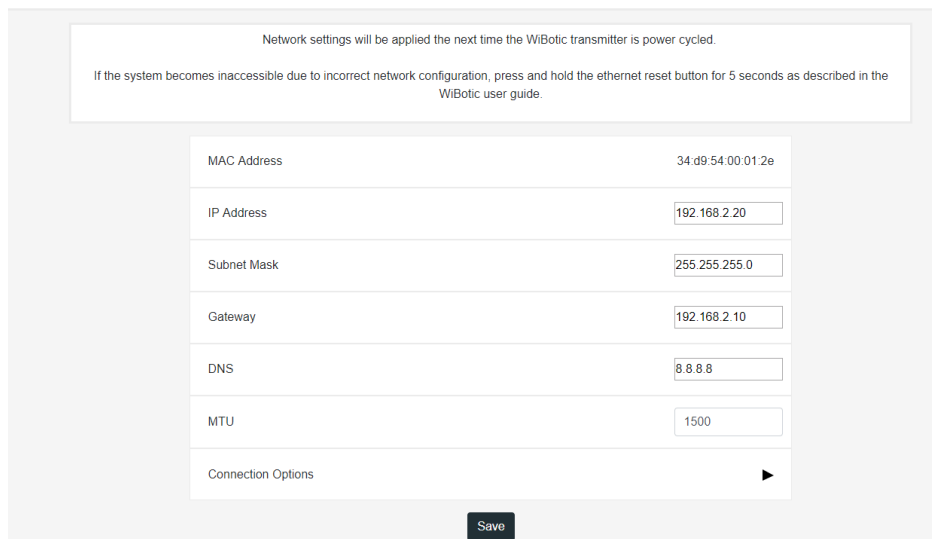
In some cases, the robot’s own load (if it remains powered on) may not allow the WiBotic system to reach the C/10 level where the battery is considered full and the system stops sending energy. This may be desirable if keeping the robot powered at all times is a key goal. However, if you do not want the WiBotic system to continue sending power indefinitely, you can limit the amount of time it will spend in CV mode. To do this, simply adjust the Maximum Top-off Charge Time level to the desired time (in seconds) and click “Save”.

When the system turns off at the end of this period, it will remain off as usual until the Restart Threshold voltage is met.

Network Settings

The Network Settings sub-menu provides configuration parameters for adding the WiBotic transmitter to a Local Area Network. The options available via this menu should be familiar to networking/IT professionals, but please contact WiBotic if questions arise.

Network Settings



Network settings will be applied the next time the WiBotic transmitter is power cycled.

If the system becomes inaccessible due to incorrect network configuration, press and hold the ethernet reset button for 5 seconds as described in the WiBotic user guide.

MAC Address	34.d9:54:00:01:2e
IP Address	<input type="text" value="192.168.2.20"/>
Subnet Mask	<input type="text" value="255.255.255.0"/>
Gateway	<input type="text" value="192.168.2.10"/>
DNS	<input type="text" value="8.8.8.8"/>
MTU	<input type="text" value="1500"/>
Connection Options	<input type="button" value="▶"/>

CAN Settings

The CAN Settings sub-menu provides configuration parameters for CAN-bus communications between the Onboard Charger and the robot/drone controller. See Appendix C for a complete discussion of WiBotic’s Onboard API and related CAN-bus setup and functionality.

“Diagnostics” Menu

The WiBotic Development Kit is intentionally configured for in-air operation using the supplied antenna coil mounting brackets. As you test various coil positions and power levels, and particularly if you move the coils to your robot or another location where de-tuning may occur, you may experience an increase in alerts or other system messages. Apart from appearing on the main overview page, these events can be tracked and logged using the Diagnostics menu.

Live Logs

The Live Log is a running list of system events that have occurred since the Web GUI was opened and connected to the Transmitter. The most recent event is listed at the bottom of the log file, so some scrolling may be required.

Events that appear in **black** text are normal and for informational purposes only. No operator action is required. Events that occur in **orange** text are alerts that generally represent a change in system condition. These may or may not require operator intervention (see Alert message list below). Events shown in **red** text are typically more severe and are unusual. If a red error message occurs and the system does not automatically return to normal operation after powering down and repowering, contact WiBotic for assistance.

Table 1: GUI Alert Messages

ALERT NAME	ALERT DESCRIPTION
TX Alarm: Power Supply Over Current	If the power supply exceeds the maximum permitted current, this alarm will occur. The system will restart and enter heartbeat mode. If the problem persists, disconnect the transmitter power cable and contact WiBotic before continuing.
TX Alarm: Power Supply Over Power	If the power supply exceeds the maximum power limit, this alarm will occur. The system will restart and enter heartbeat mode. If the problem persists, disconnect the transmitter power cable and contact WiBotic before continuing.
TX Info: Radio Communication Lost	If the radio link is not maintained between the transmitter and Onboard Charger, the transmitter will turn off. Try moving the transmitter and/or

	Onboard Charger closer to each other. The system will shut down and enter heartbeat mode.
TX Alarm: Power transfer too low	This usually occurs when the coils are positioned too far apart or in a non-optimal orientation. Transmitter will immediately shut down. Try repositioning the coils if this alert persists.
TX Info: Internal communication error	An internal error might occur that will cause the system to restart. The system will restart and enter heartbeat mode. Contact WiBotic if this problem persists.
TX Alarm: RX is in alarm state	The Onboard Charger has encountered a problem. The system will attempt to restart and resume operation.
TX Alarm: Power transfer too inefficient. Try moving coils apart.	Wireless power transfer is presently too inefficient. Try moving coils farther apart to resolve issue.
TX Alarm: System over temperature. Restarting when cooled.	The transmitter has detected an unsafe operating temperature. This could be due to sub-optimal coil positioning or de-tuning of the coils
TX Alarm: Hardware Protection	An environmental factor is causing coil detuning or another hardware issue
TX Alarm: Coil Check Over Current	An environmental factor is causing coil detuning and excess current during coil check procedure
RX Alarm: Battery voltage incorrect	The battery parameters (number of cells, chemistry, volts per cell) are inconsistent with the current battery voltage detected by the charger. Correct the battery parameters or change batteries and try again.
RX Alarm: Rectified voltage crashed	The input voltage seen by the Onboard Charger has changed abruptly. This may be due to poor coil position or other hardware error. If this problem is not resolved by repositioning coils, contact WiBotic.
RX Alarm: Charger shorted	The battery charger on the Onboard Charger has detected a short circuit on the battery output. Try disconnecting the battery to power cycle the Onboard Charger and see if the problem persists. If this problem is not resolved, contact WiBotic as there may be a hardware issue with the Onboard Charger.
RX Alarm: Internal event alarm	The Onboard Charger has had an internal error and has attempted recovery. Try disconnecting the battery to power cycle the Onboard Charger and see if the problem persists. If this problem is not resolved, contact WiBotic.
RX Alarm: Low battery current	Something is limiting the amount of current that can be used to charge the battery. This caused charging to stop. Allow the system to resume, but if the problem persists across multiple charging attempts, contact WiBotic.

Version Information

The Version Information section contains the code number for the Web GUI and firmware versions that are currently installed on the system. MAC addresses and the currently selected radio channel are also supplied. In future releases, additional hardware information such as assigned common names for each component will be supplied.

Log Messages

The Log Messages section allows users to view and download separate log files for the Transmitter and Onboard Charger – typically for troubleshooting with assistance from WiBotic.

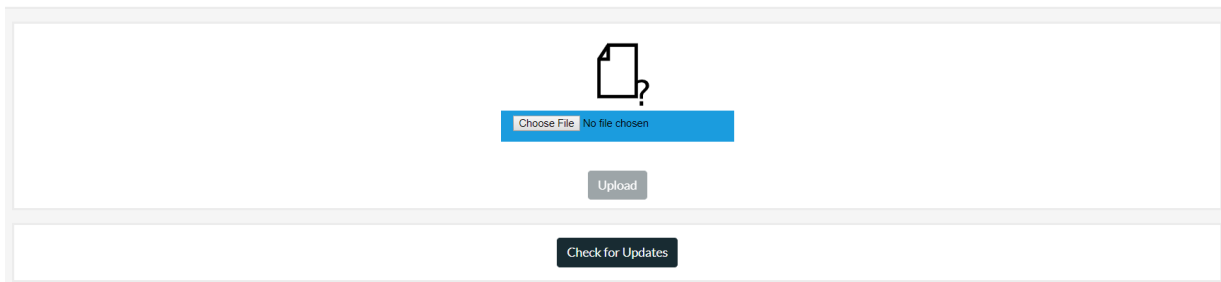
The “System Log” file contains a chronological record of all system events since the Transmitter was last powered on. It is similar to the Live Log above, but the System Log file is not date/time stamped and is not affected by closing the browser. The most recent event is listed at the bottom of the log file.

The numbered log files relate to each charging session for the Transmitter and Onboard Charger. There is currently a limit of 101 (0-100) individually recorded cycles with numbering starting at 0 again after the 100th record is saved. Therefore the “Log 0” file may not be the most current set of system data. The most current logs are shown at the top of the list. If detailed analysis of the log files is required, all of the available logs can be downloaded at once using the “Download All Log Files” link.

“Update” Menu

WiBotic will occasionally release new firmware for the Transmitter or Onboard Charger to improve system performance. Beginning with Rev 8 of WiBotic firmware, users can check to see if a more recent version is available. To do this, simply click on the black “Check for Updates” button near the top of this page.

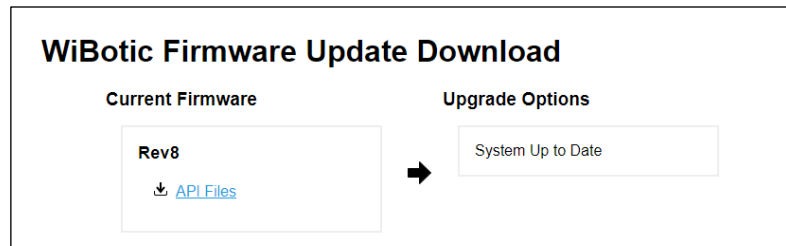
Update



The following screen will appear to advise you of your current firmware status. The left side shows your current firmware version and related files (API Docs, User Guide, etc.) are available for download in case you haven’t already received them.

The right side will show all versions of firmware that are more current than the version currently running on your system. If there is a direct upgrade path to the most current

version, you are encouraged to install it directly. If you must step through the installation of more than one version, notes will be available advising you of the proper sequence.



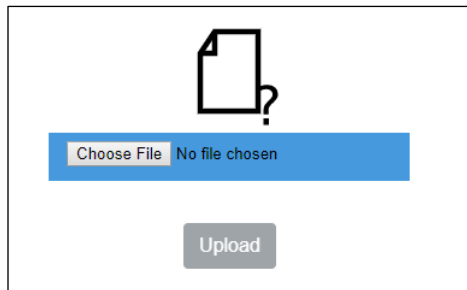
Under the heading for each firmware version will be a series of files for download. A new version of the Web GUI interface may be made available with enhanced features as well as new .bin files for the transmitter and receiver. Other available files may include the User Guide or special version-specific documentation.

When updating firmware, the first step is to download the .rfs and .bin files to a computer that is connected to the WiBotic transmitter via an Ethernet cable. The filenames will be similar to the following:

- **WEB.RFS** – the Web GUI update that is loaded onto the Transmitter
- **rx-m4-ota.bin** – the “rx” indicates this is the firmware file for the Onboard Charger/Receiver
- **tx-m7-ota.bin** – the “tx” indicates this is the firmware file for the Transmitter

Upload Button

Once the files reside on the connected computer, the next step is to upload them to the Transmitter. To begin this process, first navigate to the “Update” menu page.



The top of the Update page contains a section for choosing and uploading files to the transmitter. Use the “Choose File” button to individually select each of the three firmware files above and then click “Upload” to load them onto the Transmitter’s SD card. A yellow status bar and green check mark will indicate upload progress and completion.

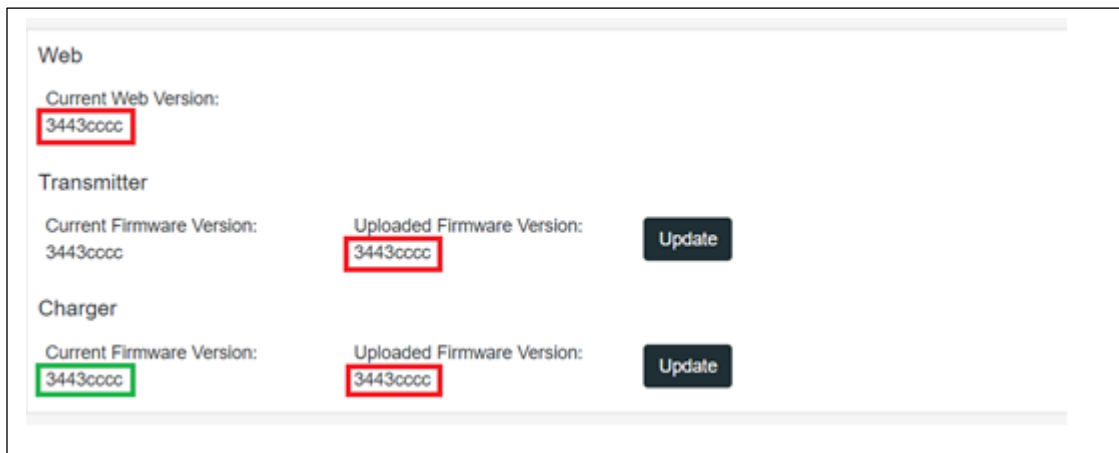


The recommended order of upload from your computer to the transmitter is 1) WEB.RFS, 2) Transmitter Firmware, and 3) Onboard Charger Firmware. You will notice a slight flash and reload of the Web GUI when the WEB.RFS file is uploaded.

Update Button

After completing the above steps, you will use the lower portion of the Update page to actually install the new firmware onto the Transmitter and Onboard Charger circuits.

Begin by confirming that the “Current Web Version” and both “Uploaded Firmware Versions” (highlighted in the red boxes below) have matching version code numbers. If they do not, repeat the steps in the previous section until all three code numbers match. Contact WiBotic for assistance if unable to complete this step.



Updating Web GUI

By uploading the Web.RFS file in the previous step, you have automatically overwritten the previous file and the new GUI is already installed.

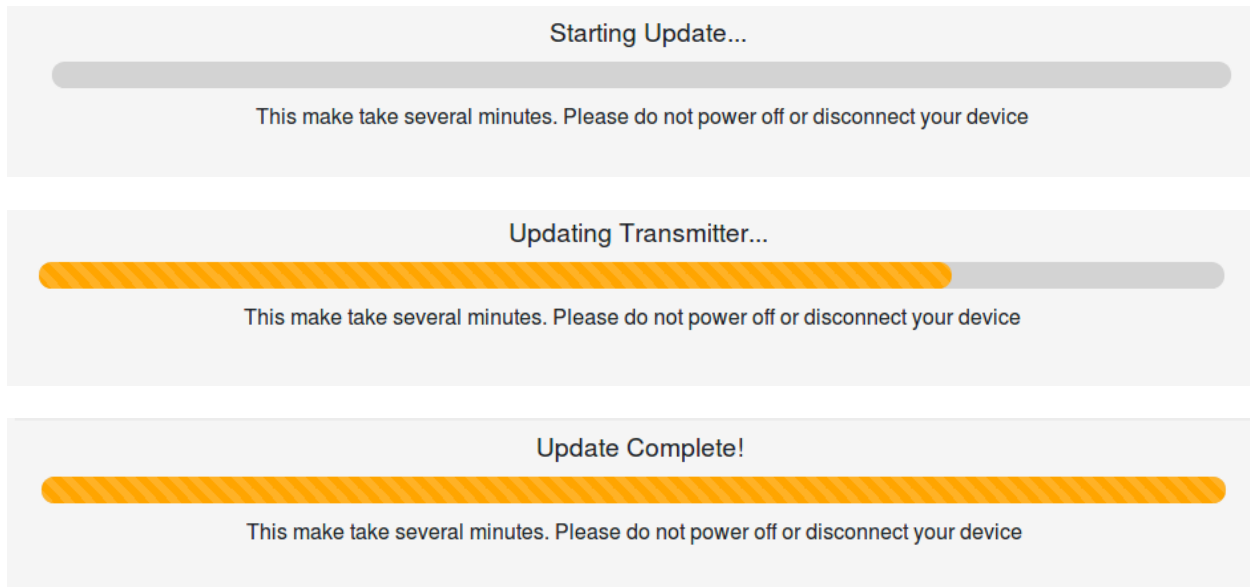
After the new Web GUI has been installed, you will update the transmitter firmware first and Onboard Charger firmware second – unless otherwise instructed by WiBotic.

Updating Transmitter Firmware

If the Current Firmware Version under the Transmitter section of the update page is different than the Uploaded Firmware Version, then proceed with the update by pressing the black “Update” button to the right of the uploaded firmware version code.

After the update is complete, the Current Firmware Version shown in the page under the Transmitter section should match the firmware version under Uploaded Firmware Version as shown in the red and green boxes above.

If the system takes more than 5 minutes to update, first try refreshing the web page to see if the firmware may have updated, but just not refreshed the page. If it appears that the transmitter still hasn't been updated, try power cycling the transmitter and try again.



Starting Update...

This make take several minutes. Please do not power off or disconnect your device

Updating Transmitter...

This make take several minutes. Please do not power off or disconnect your device

Update Complete!

This make take several minutes. Please do not power off or disconnect your device

Updating Onboard Charger Firmware

Next, ensure that the onboard charger hardware that you want to update is connected to the transmitter. If you can see a “Current Firmware Version” (highlighted in green on previous page) then the Onboard Charger is connected and is communicating with the GUI.

If the Current Firmware Version under the Charger section of the update page is different than the Uploaded Firmware Version, then proceed with the update by pressing the “Update” button to the right of the uploaded firmware version section.

The update will proceed with status bars informing of its progress. Because the Onboard Charger is being updated over a radio link, it typically takes slightly longer to complete its update as compared to the transmitter.

Starting Update...

This make take several minutes. Please do not power off or disconnect your device

Updating Charger...

This make take several minutes. Please do not power off or disconnect your device

Finishing up...

This make take several minutes. Please do not power off or disconnect your device

Update Complete!

This make take several minutes. Please do not power off or disconnect your device

After the update is complete, the “Current Firmware Version” and “Uploaded Firmware Version” for the Onboard Charger should match. Note that it may take several seconds for the web page to update after the status bar indicates “Update Complete”. During this time, the Current Firmware Version may be shown as “Unavailable”.

If the system takes more than 5 minutes to update, restart both the Onboard Charger and the Transmitter and try again.

If you are unable to finish the firmware update and your Onboard Charger appears to be “stuck” in a non-functional mode, a recovery process is available. Use the area at the bottom of the update page to enter the last six digits of the MAC Address for the OC and press “Finish Update”. This will reset the OC and should allow completion of the firmware update process. Contact WiBotic if you are unable to complete the update.

Continue Interrupted Update

If a charger was disconnected in the middle of an update and does not reconnect normally, enter the last six hexadecimal digits of its MAC address (check [system log](#)) below and press finish update.

Charger MAC

At this point your system’s firmware has been updated and you can begin to enjoy the new features and performance.

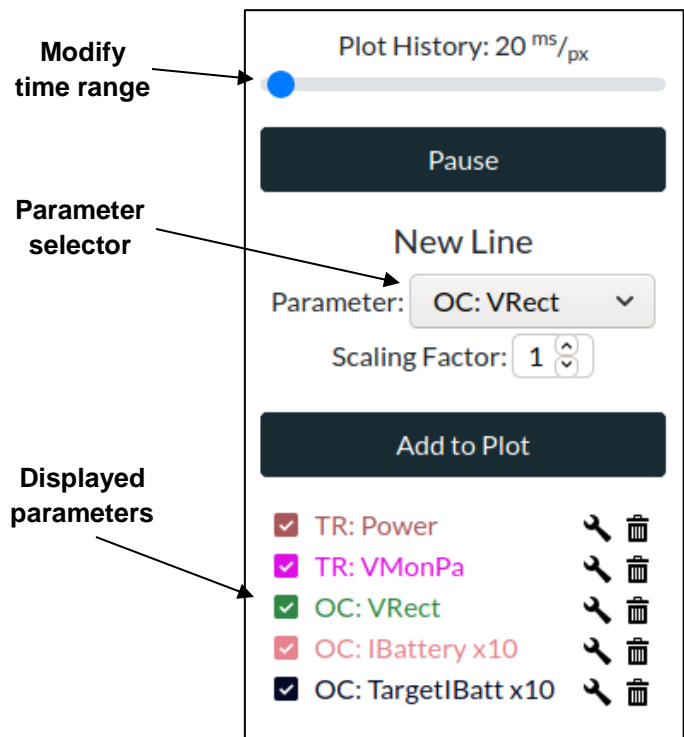
“Data Plots” Menu

The Data Plots menu provides a graphical representation of various system parameters over time. To choose the displayed parameters, click on the Parameter pull down menu on the right side of the screen. You may also reset the default scale for that parameter by typing a new value in the Scaling Factor box immediately below. Click “Add to Graph” to add the parameter to the list of displayed parameters. It will now appear on the plot.

To momentarily deselect a parameter from the plot, simply click on the check box. Click again to re-display the parameter.

To eliminate the parameter from the list of current options, click on the “Trash” icon. Parameters can always be added again using the directions above.

To extend the time range shown in the plot, slide the “Packet Count” bar to the right.



“About” Menu

The About menu provides WiBotic contact information as well as detailed information on the software and firmware licenses used within the WiBotic platform.

Wireless Power Transmission

To begin testing wireless power, first confirm that the System Status “Charger” toggle switch in the upper right corner is in the default “On” position and the “Transmitter” toggle switch is in the default “Off” position.

Next, turn the Transmitter on by clicking the Transmitter toggle switch. If there is an Onboard Charger ready to accept power, and it is within range of the transmit coil, the system will now enter its charging sequence. In a few seconds, charging will begin and the graphical indicators on the center of the screen will show active charging.



Once wireless power transmission begins, the system will stop only if the Transmitter toggle is clicked again to turn it off, if the battery is fully charged, or if a shutdown event occurs. (See the “Power Down Events” section for details). WiBotic recommends disconnecting power to both the Transmitter and Onboard Charger when not in use for extended periods of time.

The table below enumerates the transmitter and/or charger status indicators that will appear in the upper right portion of the GUI as the system is operating.

Table 2: GUI Status Indicators

STATUS NAME	STATUS DESCRIPTION
Off	The Device is off.
Idle	The Power Transmitter will periodically check for Onboard Chargers when they are in range of the 2.4GHz low power radio.
Stabilizing / Ramp Up	Wireless power is enabled, and power is ramping up to pre-set target levels.
Charging	The transmitter is sending power to the OC for battery charging.
Constant Current	The OC is charging the battery in constant-current mode.
Constant Voltage	The OC is charging the battery in constant-voltage mode.
Battery Full / Done	The battery float voltage has been reached and charge current has dropped to C/10, so the system has completed charging.
Alarm	An unexpected event occurred. See log message for details. The system should restart once conditions return to normal.

About the Battery Charging Process

WiBotic Onboard Chargers are configured by default to act as Constant Current/Constant Voltage (CC/CV) battery chargers.

If the Transmitter is physically powered, the Transmitter toggle switch is turned on, and an Onboard Charger is in range, the system will begin charging if the battery is not already considered fully charged. The charger regulates output voltage to the battery at the pre-defined voltage limit in the “Battery Settings” menu. The initial voltage limit is set at the WiBotic factory based upon the customer’s reported battery specs and is listed on the Certificate of Conformance. The voltage limit can be changed and will be stored in memory even if the Onboard Charger is powered off and/or disconnected from the battery.

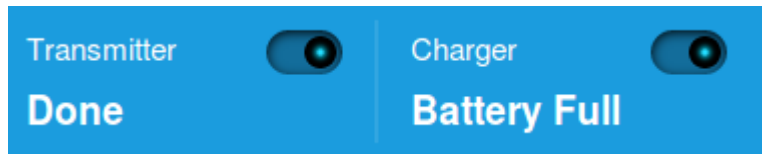
During startup, the charging current will ramp up to the maximum constant-current charge rate as configured in the WiBotic software. The Onboard Charger will always try to ramp-up to the maximum charge current. However, in some cases, depending on either sub-optimal coil position, environmental conditions or high temperatures of the electronics, the Onboard Charger may not be able to achieve the maximum charge current. If unable to reach the maximum charge current, the Onboard Charger will not increase beyond a charge current that the system can reliably and safely sustain.

After the constant current (CC) portion of the charge cycle is complete, the Onboard Charger will enter constant-voltage (CV) charging mode. During CV mode, the charge current decreases gradually as the voltage of the battery approaches the charge termination voltage.

There are two ways for the system to terminate charging once it is in constant-voltage mode:

- 1. CV Mode Timeout:** Once CV Mode has started, the Maximum Top-Off Charge Time period begins. Even if the battery is not fully charged at the end of this period, the charge cycle will terminate and the Power Transmitter will momentarily return to standby. If the system senses battery voltage is below the fully charged level, the system will resume charging until the battery is fully charged and charger enters CV Mode.
- 2. CV Mode C/10 “Charge Complete”:** If the charge current drops below one-tenth of the maximum charge current rate (C/10), then charging is considered complete and the Power Transmitter will return to standby.

The GUI will indicate that the battery is fully charged by displaying “Battery Full” as the charger’s status.

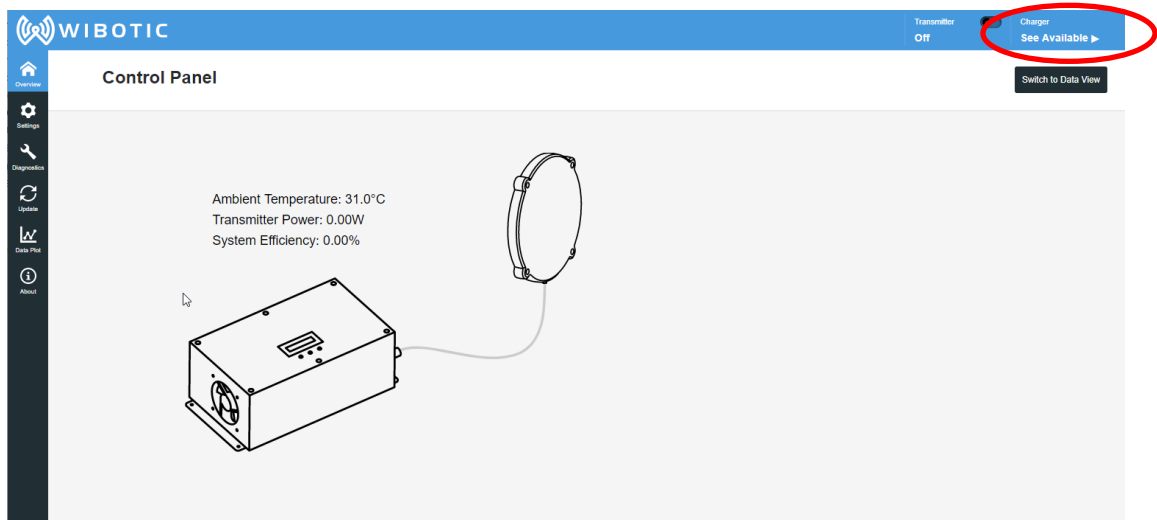


Distance Connect Mode

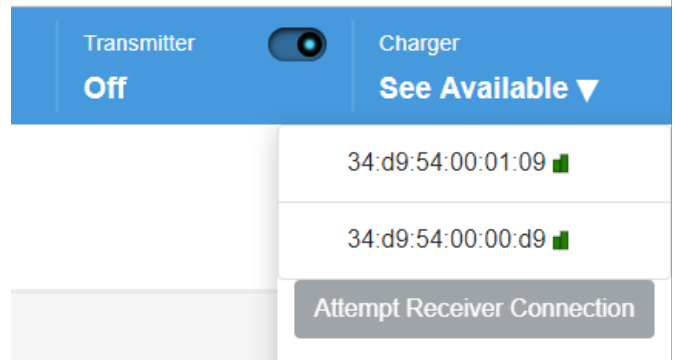
When there are no Onboard Chargers within charging range of the Transmitter, it is still possible to determine which, if any, OCs are within radio range. This may be helpful when monitoring various aspects of a fleet of robots, or for updating firmware across a fleet without asking each robot to approach the charging station.

Instructions for using Distance Connect

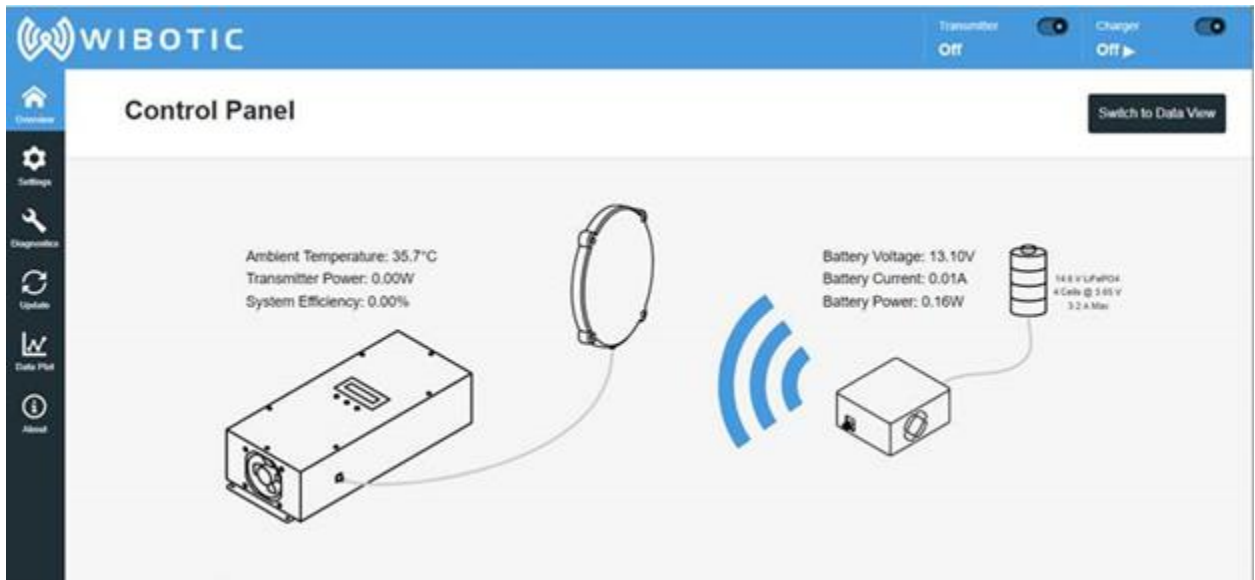
- 1) Confirm that all OCs in the fleet are connected to a battery and powered up.
- 2) Make sure there is no Onboard Charger antenna within charging range of the transmitter antenna. Usually a separation of 12in is sufficient. If the two antenna coils are allowed to pair, then the transmitter will only be able to “see” the paired Onboard Charger.
- 3) Open the WiBotic GUI.
- 4) Click on the “See Available” link in the upper right corner (see below). This will provide a list of all of the Onboard Chargers within radio range of the Transmitter (roughly 30ft).



- 5) If only one OC is connected to a battery, then only one will appear. However, if you have a fleet of OCs that are active on robots, you will see a list of all of them within radio range.
- 6) To connect to a particular OC, click on its MAC address (a feature to provide common names is coming soon) and select “Attempt Receiver Connection”.



- 7) After a few seconds, the Transmitter radio will connect to the OC and you will see a different animation on the Overview page showing a radio connection between the two devices instead of a wireless power connection (see below).



- 8) Once connected, you can proceed with monitoring of the remote system or with firmware updates following the normal update procedure. Note that the transmitter can only be connected to one Onboard Charger at a time, so if you're Distance Connected to an OC, the Transmitter will not connect to another OC even if it is brought within antenna coil range. You will first need to manually disconnect the transmitter from the first OC by clicking the “Disconnect” button.

Power-Down Events

There are several mechanisms that can cause the Power Transmitter to power-down during charging:

COMMUNICATION ERROR / RX TIMEOUT: The Power Transmitter and Onboard Charger communicate over a low-power 2.4GHz radio link. If the Onboard Charger is out of range, obstructed by metal, or is in a particularly electrically noisy environment, the GUI may display “RX Disconnected”. The Power Transmitter will power-down until the radio link is restored. If the radio link is restored when the battery’s voltage is above the restart threshold voltage, then the system will not turn back on until the restart threshold is met.

GUI INFORMATIONAL MESSAGES: There are informational messages that appear in the log of the GUI. These messages indicate either a status update or an alert – some of which may cause a precautionary shut down. If an alert has been triggered and power transfer stops, the message will provide a recommendation to ideally solve the problem. If the GUI application itself stops responding, it may be necessary to try refreshing the web page. If these problems persist, contact WiBotic for support. See the section on GUI Alert Messages for more details.

Troubleshooting GUI Connectivity

If the Power Transmitter is powered on and connected to the computer via Ethernet, but the web app is inaccessible, proceed through the following steps:

1. Make sure you have setup the IPv4 settings as outlined in the Configuring Ethernet Connectivity Section as in previous section.
2. Try resetting the Ethernet Connection settings as outlined in the Ethernet Configuration Reset section as in previous section.
3. If you still cannot connect to the GUI, contact WiBotic at info@wibotic.com for assistance. It is helpful if you include the results of pinging the device with the command “ping 192.168.2.20” in a terminal or command prompt window. If able, running a packet capture tool such as Wireshark on the interface where the WiBotic system is plugged in, and sending the capture to WiBotic, may also help us assist you in getting the system up and running.

Suppliers Declaration of Conformity (SDoC)

This device complies with Part 18 of the FCC Rules.

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

Changes or modifications not expressly approved by WiBotic Inc could void the user's authority to operate the equipment.

Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Name and address of the US Responsible Party

Company Name:	WiBotic Inc.	Telephone Number:	1-206-580-0900
Contact Name:	Benjamin Waters	Facsimile No:	N/A
Contact Title:	CEO	Email Address:	Info@wibotic.com
Company Address:	9706 4 th Ave NE Suite 208 Seattle, WA 98115	Web Address:	www.wibotic.com

Identification of Equipment

Product Names:	Wireless Power System
Model Numbers:	TR-301-A-ST, OC-251-12-ST, OC-301-30-ST

Note to robot manufacturers:

If a WiBotic Onboard Charger is installed inside the robot, or where the FCC approval label cannot otherwise be seen, a separate label must be affixed to the outside of the robot. The label should read as follows depending upon the OC model used:

OC-301: "Contains FCC ID: 2AVQUOC30130ST"

OC-251: "Contains FCC ID: 2AVQUOC25112ST"

APPENDICES:

A: Network API

B: Python Helper Library

C: Onboard API

APPENDIX A: NETWORK API

The WiBotic Network API allows users to programmatically access the WiBotic Transmitter over an Ethernet network for system monitoring and control. All functionality available via the WiBotic Web GUI is also programmable via this interface. The following pages provide an overview of the API functionality. WiBotic also provides a Python library and sample code to help users get started. If you have not received the sample files from WiBotic, please contact us at sales@wibotic.com for access.

Getting Started

The WiBotic Network API is accessible over a WebSocket at the address `ws://192.168.2.20/ws` (where 192.168.2.20 is the IP address of the WiBotic Transmitter) using the “*wibotic*” WebSocket subprotocol. You can use any language or environment that can communicate over WebSockets to send and receive API data to the WiBotic system.

A helper library and demo application are available in Python to assist in development (see Appendix B).

NOTE: The WebSocket server does not currently incorporate any authentication. Do not connect the WiBotic system to a public network without considering the security implications.

NOTE: The WiBotic WebSocket server could be susceptible to an attack where a malicious webpage opens a socket and sends data to the WiBotic system in another isolated network segment. This will be addressed in future versions of the system.

General Packet Format

The WiBotic Network API uses binary WebSocket frames. The frames begin with a byte that indicates the frame type (see Response Packet Types and Request Packet Types below). This byte determines how the rest of the frame should be interpreted.

Additionally, note that not every WiBotic system uses or reports every parameter or ADC value.

Request Packet Types

Name	Code	Description
Read Parameter	0x01	Request a parameter to be read
Write Parameter	0x03	Request a parameter to be written
Stage Parameter	0x04	Request a parameter to be staged
Commit Parameter	0x05	Commit all staged parameter to non-volatile memory
Request Connected Devices	0x06	Requests an updated set of currently connected devices

Response Packet Types

Name	Code	Description
Parameter Update	0x80	A parameter was updated
Parameter Response	0x81	Response to a request for data from a parameter
ADC Update	0x82	New real-time ADC packet
Stage Parameter Response	0x83	Response to a request to stage a parameter for commit
Commit Parameters Response	0x84	Response to committing staged parameters
Connected Devices	0x85	What devices are currently connected to the system
Message	0x86	Log messages in ASCII format with severity level
Charger Association	0x87	A charger is in radio range and is checking to see if it's in coil range

API Requests

The following diagrams illustrate the position of the bytes that should make up the binary WebSocket frames that contain the API requests and responses. Multiple successive bytes with the same name indicate that the value is to be split over those bytes. The bytes illustrated continue to the next line if there are too many bytes to show on one line.

Building API Requests

Read Parameter

0x01	Device ID	Parameter ID	Parameter ID	Parameter ID	Parameter ID
------	-----------	--------------	--------------	--------------	--------------

Write Parameter

0x03	Device ID	Parameter ID	Parameter ID	Parameter ID	Parameter ID
New Data	New Data	New Data	New Data		

Stage Parameter

0x04	Device ID	Parameter ID	Parameter ID	Parameter ID	Parameter ID
------	-----------	--------------	--------------	--------------	--------------

Commit Parameters

0x05	Device ID
------	-----------

Request Connected Devices

0x06

Parsing API Responses

Parameter Update

0x80	Device ID	Parameter ID	Parameter ID	Parameter ID	Parameter ID
Status					

Parameter Response

0x81	Device ID	Parameter ID	Parameter ID	Parameter ID	Parameter ID
Param Data	Param Data	Param Data	Param Data		

ADC Update

0x82	Device ID	ADC ID 1	ADC ID 1	ADC Data 1	ADC Data 1
ADC ID 2	ADC ID 2	ADC Data 2	ADC Data 2	...	ADC ID <i>n</i>
ADC ID <i>n</i>	ADC Data <i>n</i>	ADC Data <i>n</i>			

The number of ADC values in a packet can be determined by the following equation:

$$\frac{(\text{NumberOfBytes}) - 2}{4}$$

Stage Parameter Response

0x83	Device ID	Parameter ID	Parameter ID	Parameter ID	Parameter ID
Status					

Commit Parameters Response

0x84	Device ID	Status
------	-----------	--------

Connected Devices

0x85	Device Flags	Device Flags
------	--------------	--------------

Bit 1: Transmitter Connected

Bit 2: Charger Connected

Message

0x86	Device ID	Level	ASCII String	ASCII String	ASCII String
ASCII String	ASCII String	ASCII String	...		

Charger Association

0x87	Device ID	RSSI	Charger MAC	Charger MAC	Charger MAC
Charger MAC	Charger MAC	Charger MAC			

Parameters

Name	ID	Description	Read	Write	TR/OC Availability
Address	3	Address of the device on the internal point to point wireless link	Yes	No	Both
RadioChannel	4	Current device radio channel	Yes	No	Both
DigitalBoardVersion	26	Version of the digital board that is running the system	Yes	No	Both
BatteryCurrentMax	34	Maximum current that should be delivered to the battery in milliamps	Yes	Yes	OC
ChargerCurrentLimit	35	Current limit that the system has decided can be delivered to the battery in milliamps	Yes	No	OC

MobileRxVoltageLimit	36	Calculated maximum battery voltage (mV) based on chemistry, number of cells, and voltage per cell	Yes	No	OC
RxBatteryVoltageMin	37	Calculated minimum battery voltage (mV) based on chemistry, number of cells, and voltage per cell	Yes	No	OC
BuildHash	38	Version hash of the firmware that is loaded on the device	Yes	No	Both
TargetFirmwareId	39	Type of firmware image that is running on the device	Yes	No	Both
OtaMode	41	Current firmware update state	Yes	No	Both
RxBatteryVoltage	42	Last seen value of the battery's voltage (millivolts)	Yes	No	OC
RxBatteryCurrent	43	Last seen value of the battery's current (milliamps)	Yes	No	OC
RxTemperature	44	Last seen value of the power board's temperature (Celsius)	Yes	No	OC
EthIPAddr	45	Network Static IP Address if no DHCP	Yes	Yes	TR
EthNetMask	46	Network Subnet Mask if no DHCP	Yes	Yes	TR
EthGateway	47	Network Gateway if no DHCP	Yes	Yes	TR
EthDNS	48	Network DNS Server	Yes	Yes	TR
EthUseDHCP	49	Device should use DHCP on the network	Yes	Yes	TR
EthUseLLA	50	Device should fallback to Link-Local Addressing if DHCP fails	Yes	Yes	TR
DevMACOUI	51	MAC Address Organizationally Unique Identifier	Yes	No	Both
DevMACSpecific	52	MAC Address Specific Identifier	Yes	No	Both
EthMTU	54	Ethernet MTU	Yes	Yes	TR
EthICMPReply	55	Reply to Pings	Yes	Yes	TR
EthTCPTTL	56	TCP Time to Live	Yes	Yes	TR
EthUDPTTL	57	UDP Time to Live	Yes	Yes	TR
EthUseDNS	58	Use DNS	Yes	Yes	TR
EthTCPKeepAlive	59	Keep TCP Connections Alive	Yes	Yes	TR
ChargeEnable	60	Enable or disable the ability for this device to transfer power or charge a battery	Yes	Yes	Both
I2cAddress	61	Address of this device on a I2C bus	Yes	Yes	OC

RxBatteryNumCells	62	Number of cells in the battery this charger is configured to charge	Yes	Yes	OC
RxBatteryMVPPerCell	63	Voltage of a battery cell (in millivolts) that this charger is configured to charge	Yes	Yes	OC
LogEnable	67	Enable logging battery charge data	Yes	Yes	TR
RxBatteryChemistry	68	Chemistry of the attached battery	Yes	Yes	OC
IgnoreBatteryCondition	70	Ignore battery condition when charging. Potentially Dangerous.	Yes	Yes	OC
PowerBoardVersion	71	Version of the power board that is running the system	Yes	No	Both
UpdaterMode	75	Mode for updating another device	Yes	Yes	TR
AccessLevel	78	Current level of access that external entities have to the system.	Yes	Yes	Both
ConnectedDevices	79	Devices connected to this device	Yes	No	TR
LcdVersion	80	Version of the LCD Display being used	Yes	No	TR
RadioConnectionRequest	81	Partial MAC of a device to request a connection to via radio. Uses DevMACOUI for OUI portion of MAC address. 0 to disable.	Yes	Yes	TR
CANMessageConfig	82	Configuration for what type of format to send over CAN	Yes	Yes	OC
CANID	83	Configuration for CAN bus id (requires reboot)	Yes	Yes	OC
OtaCtrl	84	Command parameter to control the firmware update process	Yes	Yes	TR
CoilCheckBaseStation	85	Partial MAC of the device that is connected to when in coil range	Yes	No	OC
RecoveryChargeEnable	86	Allow slow charging of a battery that has been over discharged	Yes	Yes	OC
CANBitRate	87	Bit rate of the CAN device (kbit/s)	Yes	Yes	OC
BatteryRestartPerCell	88	Number of millivolts per cell to allow the battery to drop before initiating charge	Yes	Yes	OC
MaxCVChargeTime	89	Maximum charge time in seconds while battery is in constant voltage mode	Yes	Yes	OC

QuietFans	90	Enable temperature based PWM on applicable fans	Yes	Yes	TR
-----------	----	---	-----	-----	----

Parameter Status Codes

Name	Code	Description
Failure	0	The parameter was not set due to a general failure
Hardware Failure	1	Some hardware did not respond as expected. The parameter was not set.
Invalid Input	2	The data that was to be written to the parameter was not valid for the parameter.
Non-critical Fail	3	The data was not written to the parameter, but this should be anticipated for the given parameter
Read only	4	The parameter is currently in a read only state and should only be read
Success	5	The data to be written to the parameter was written successfully
Not Authorized	6	The current session was not authorized to change the selected parameter
Pending	7	The parameter is being processed
Clamped	8	The parameter was set, but it was limited in value

Real-time ADC Packets

Name	ID	Data Type	Description
PacketCount	0	uint16_t	Packet identifier within second of time
Timestamp	1	uint32_t	Timestamp of the packet in seconds
ChargeState	2	uint8_t	Current state of system
Flags	3	uint16_t	Flags
PowerLevel	4	uint16_t	Output power level
VMon3v3	5	float	3.3v voltage
VMon5v	6	float	5v voltage
IMon5v	7	float	5v current
VMon12v	8	float	12v voltage
IMon12v	9	float	12v current
VMonGateDriver	10	float	Gate Driver voltage
IMonGateDriver	11	float	Gate Driver current
VMonPa	12	float	Power Amplifier voltage
IMonPa	13	float	Power Amplifier current
TMonPa	14	float	Power Amplifier temperature
VMonBatt	15	float	Battery Voltage
VMonBattProg	16	float	Charger Voltage

VRect	17	float	Rectified Voltage
TBoard	18	float	Board Temperature
ICharger	19	float	Charger current
IBattery	20	float	Battery current
TargetIBatt	21	float	Target battery current
IMaster	22	float	Charger 1 current
ISlave1	23	float	Charger 2 current
ISlave2	24	float	Charger 3 current
RfSense	25	float	Measured RF Power output
Vmon48v	26	float	48v input voltage
Imon48v	27	float	48v input current
TmonAmb	28	float	Ambient Temperature

Device ID

Device	Address
Transmitter	1
Charger	2

APPENDIX B: PYTHON LIBRARIES

Introduction

A Python helper library and sample code is provided to help the user integrate faster with the WiBotic system and to demonstrate usage patterns. Both a lower-level interface and higher-level interface are provided.

Setup

The libraries and sample code require Python 3.6 (or greater) to be installed and the sample code was developed to run on Windows or Linux, though may run on macOS with small modifications. Beyond the basic installation of Python, an additional library will be required: websockets..

- Install Python 3.6 (or greater). <https://www.python.org/downloads/>
- Ensure Python is correctly configured in your PATH environment variable for ease of access.
- Install the websockets package. Note that the *pip* tool referenced is included with Python 3.6. <https://websockets.readthedocs.io/en/stable/intro.html> If not familiar with *asyncio* or Python coroutines, it is recommended that the user briefly read the following information. <http://cheat.readthedocs.io/en/latest/python/asyncio.html>

If not already done, the user should install the WiBotic UI as included and documented elsewhere in this User Guide and ensure that the Windows laptop is correctly configured to communicate with the transmitter via an Ethernet connection. If the WiBotic UI is not able to communicate with the transmitter, the Python scripts will also fail to operate.

Verify Sample Code Operation

At a command prompt window, navigate to the directory that contains all the wibotic sample code and other python files and execute the following: `python wibotic_highlevel_sample.py`

You should see output similar to the following:

```
Transmitter is now enabled
=====
Transmitter
```

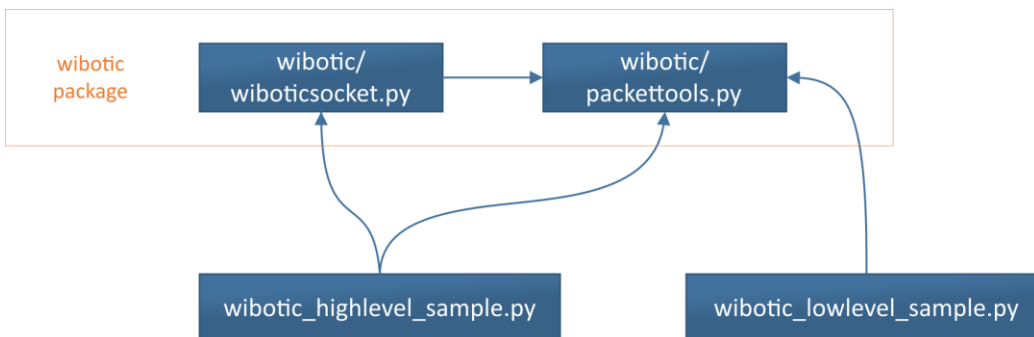
```

State:          RAMP_UP
Timestamp:      1 day, 1:34:16.093000
Set Power Level: 64000
PA Temperature: 28.0
=====
Charger
State:          IDLE
Timestamp:      0:16:33.045000
Battery Voltage: 12.43
Battery Current: 0.0
Temperature:    24.3
=====
Transmitter
State:          RAMP_UP
Timestamp:      1 day, 1:34:16.194000
Set Power Level: 50725
PA Temperature: 28.0
    
```

If you do not see this output, it is possible that either your TCP/IP settings are not configured correctly (is the WiBotic UI working?) or that the Transmitter is not configured to be at the default IP address of 192.168.2.20. In the latter case, you can modify `wibotic_highlevel_sample.py` to change the IP address to the address you have configured.

Python Development

The following diagram shows the dependency relationships between the various python files.



Provided Code

- `wibotic/packettools.py`: low level library code to help create and parse packets used in communication with the Transmitter over the websocket interface.
- `wibotic/wiboticsocket.py` higher level code to interact with the WiBotic system in a simpler blocking manner without having to consider asynchronous operations.
- `wibotic_lowlevel_sample.py`: Example code showing how to open the transmitter websocket directly, and communicate to the transmitter over that socket, using

the low level packettools library to parse and create data packets. Because this sample connects directly to the asynchronous websocket interface it is the method of choice when requiring high frequency (10Hz) ADC data packets to be received from the Transmitter and any connected Onboard Charger/Receiver.

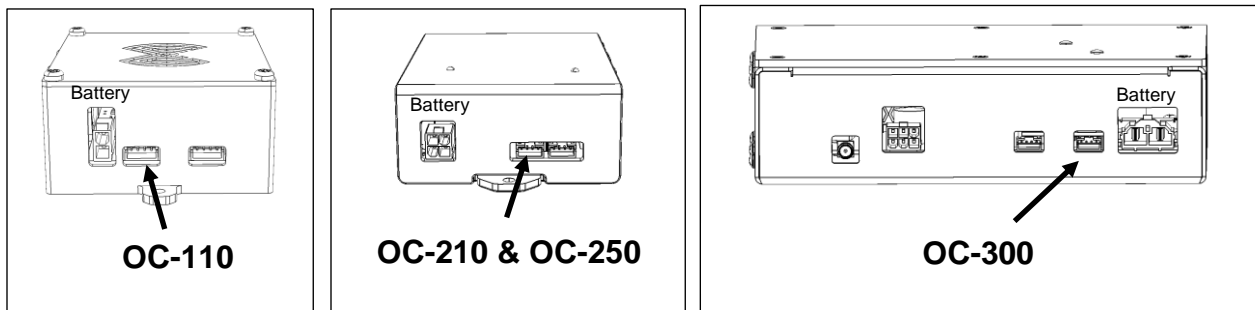
- `wibotic_highlevel_sample.py`: Example code showing how to use the higher level library code referenced above and implements some example commands. It sets the ChargeEnable setting to True, then it enters a loop to show the most recently received Transmitter and Onboard Charger/Receiver ADC packets.

APPENDIX C: ONBOARD API

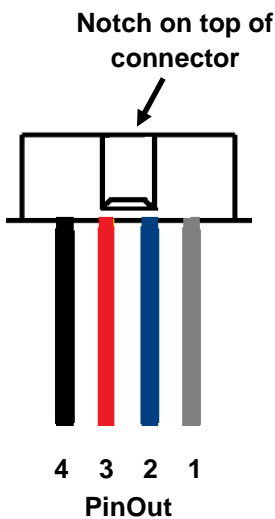
Similar in functionality to the Network API, the Onboard API allows for direct communication between your robot/drone and the WiBotic Onboard Charger. Rather than using Ethernet, however, the Onboard API uses a Control Area Network (CAN) interface as a means of communication - specifically “UAVCAN”, a lightweight protocol used for communication in robotic and aerospace applications (<https://uavcan.org/>). With this interface, users have complete access to the WiBotic Onboard Charger for monitoring and configuring charge settings directly from CAN enabled flight or robot controllers.

CAN Interface

The CAN-Bus connector is always the closest JST connector to the battery cable port on WiBotic Onboard Chargers. See below for the location on each OC model:



The following diagram describes the pin-out for the CAN connection. Note that wire colors may be different depending upon the brand of connector cable. Use the raised notch on the JST connector to determine orientation as shown.



As seen on OC-210/250

Pin	Function
1	NC
2	CAN Low
3	CAN High
4	GND

Connector and Cable Type:

JST PHR-04 AWG 24

USB to CAN

Pin	Signal	Description
1	-	No Connection
2	CANL	CANL bus line (dominant low)
3	CAN_GND	Can Ground
4	-	No Connection
5	CAN_SHLD	Connected to CAN_GND via 100Ω/0.1uF
6	CAN_GND	Can Ground
7	CANH	CANH bus line (dominant high)
8	-	No Connection
9	-	No Connection

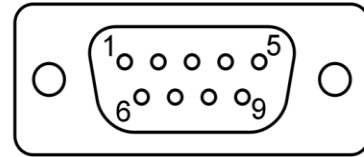


Figure 5: USB2CAN Pinout



NOTE: A 120 Ω terminator resistor must be placed between CAN High and CAN Low as per ISO-11898 standard.

It is possible to communicate with an Onboard Charger via CAN through any device that has a USB port by using a USB2CAN converter (or similar) and adapter cable. One option can be found at the link below, but any good USB-to-CAN adapter should do the job.



https://shop.8devices.com/index.php?route=product/product&path=67&product_id=95

The adapter cable must be plugged into the USB2CAN device with the wires connecting the JST connector for the CAN bus on the on-board charger.

CAN API

WiBotic's CAN API uses the UAVCAN protocol (<https://uavcan.org/>). UAVCAN handles the packet serialization and communication when using the API through the CAN bus. How information is sent through UAVCAN is determined by its DSDL (data structure description language) definition.

Adjustments to the data sent out from the on-board charger and the ID of the CAN bus can be changed by modifying the *CANMessageConfig* and *CANID* NVM parameters on the on-board charger. For access to all data accessible via the CAN API use custom WiBotic provided DSDL definitions.

Basic Setup

Python is a programming language that allows for quick setup and execution of commands.

Detailed documentation of the programming language itself can be found at:

<https://www.python.org/doc/>. When using the CAN API via Python it is strongly recommended that Python 3 is used.

Windows

1. Navigate to the official Python website and download the latest version of Python 3: <https://www.python.org/downloads>. For this guide we chose to download the executable installer.
2. Run the downloaded executable.
3. Make sure the “Add Python 3.x to PATH” box is checked and click “Install Now.”
4. Install a CAN driver for Windows (See vendors best practice). Use the vendor specified CAN tools to set the CAN interfaces bitrate to 500000.
5. Open a Command Prompt.
6. Run: `pip install uavcan`

Now you can either select the Python executable or type `python` into the command prompt to use Python.

Linux (Debian based)

1. Make sure the systems package information is up to date by running: `sudo apt-get update`.
2. Install Python 3 by running: `sudo apt install python3`.
3. Install PIP by running: `sudo apt install python3-pip`.
4. Install the UAVCAN Python library by running: `sudo -H pip3 install uavcan`.
5. Install a native CAN interface package (e.g. SocketCAN or pyserial). For this example, we install SocketCAN by running: `sudo apt install can-utils`.
6. If necessary for your CAN hardware, load the CAN kernel modules. Most CAN adapters do not need these modules; however, built in CAN interfaces like on a BeagleBone generally do:

```
sudo modprobe can
sudo modprobe can-dev
sudo modprobe can-raw
```

7. Bring up the CAN interface by running: `sudo ip link set up can0 type can bitrate 500000`.
8. Confirm CAN interface is running by checking: `ip addr`.
9. (Optional) For debugging purposes when using SocketCAN running the command `candump can0` will display the raw communications happening on the CAN bus.

`candump` should result in a stream of data that looks similar to the image.

```
can0 012 [8] DA EB B1 68 03 4A 00 80
can0 012 [8] 00 00 00 00 00 00 00 20
can0 012 [8] 00 00 00 00 00 00 00 00
can0 012 [8] 00 00 00 00 57 69 42 20
can0 012 [6] 6F 74 69 63 00 40
can0 012 [8] D8 03 03 4A 00 00 C4 81
can0 012 [8] 2C 00 00 B1 68 00 00 21
can0 012 [7] 00 00 00 00 00 00 41
```

Talking over CAN

In order to talk to an Onboard Charger across CAN, the UAVCAN protocol must be setup. For ease of use it is recommended that `pyuavcan` (which was installed as a part of setup) is used. More descriptions of how to use `pyuavcan` can be found here:

https://uavcan.org/Implementations/Pyuavcan/Tutorials/2._Basic_usage/.

Setting up a UAVCAN Node in Python

1. Open a Python 3 terminal.
2. Import UAVCAN by typing: `import uavcan`
3. Setup node information (NOTE: lines starting with “#” are interpreted as comments in Python):

```
# Creates a Node Info object
node_info = uavcan.protocol.GetNodeInfo.Response()

# Set the name of the object
node_info.name = 'org.uavcan.pyuavcan_demo'

# Set the version of the node
node_info.software_version.major = 1

# Gives the node a unique ID
node_info.hardware_version.unique_id = b'12345'
```

4. Initialize a node to your device (make sure to replace `can0` with your device name if it differs) and start it spinning in another thread:

```
node = uavcan.make_node(
    'can0',
    node_id=123,
    node_info=node_info,
    mode=uavcan.protocol.NodeStatus().MODE_OPERATIONAL)
```

```
# Start the UAVCAN node in its own thread
import threading
threading.Thread(target=node.spin, daemon=True).start()
```

After this the node variable is all set and ready to use.

Including WiBotic's DSDL definitions

The function `uavcan.load_dsd1(PATH)` can be used to load WiBotic's DSDL definitions as a third party definition. This will add the `thirdparty` path to the definition (e.g.

```
uavcan.thirdparty.wibotic.*).
```

Sending a Message Over the CAN Bus with a UAVCAN Node

For this example, we will send a command to change the Onboard Charger's maximum top-off charge time to 36000 seconds.

1. Make sure to have already setup your node. We will be using the variable name "node" for this example.
2. Create a UAVCAN GetSet request:

```
# Note - the default node ID for WiBotic systems is 18. This can be
# read or changed from the WiBotic web GUI under "CAN Settings"
target_node_id = 18

# Creates the UAVCAN GetSet request for the parameter
# VTIM - the max top off charge time for the Onboard Charger
request = uavcan.protocol.param.GetSet.Request(
    name='VTIM',
    value=uavcan.protocol.param.Value(integer_value=36000))

# Send the request to the WiBotic system on the bus
node.request(
    request,
    dest_node_id=target_node_id,
    callback=lambda event: print(uavcan.to_yaml(event)))
```

3. Create and send a parameter read message:

```
# Creates the GetSet request
request = uavcan.protocol.param.GetSet.Request(name='VTIM')

# Broadcasts the message through the CAN bus
node.request(
    request,
    dest_node_id=target_node_id,
    callback=lambda event: print(uavcan.to_yaml(event)))
```

Within a second or two you should see the message from the callback function appear with data from the Onboard Charger on the CAN bus. The value should match what was sent in the initial write request.

Provided Code

- `wibotic_can_sample.py`: Example code that shows how to read, write, and save parameters over CAN. The example sets the maximum CV charge timeout to 36000 seconds, reads the value back, and saves the change.
- `wibotic_can_battery_setting_demo.py`: Opens an interactive terminal that can be used to configure a battery. Type “help” to see the list of available commands, or “help <command>” to see detailed information on a specific command.

Serializing Packets

General Concepts

DSDL (data structure description language) is a UAVCAN provided format to define data structures that can be read and written by the UAVCAN protocol. A DSDL definition is a description of how the data will be serialized and deserialized during communication.

Just setting up UAVCAN on a system will enable communication with any of UAVCAN's default DSDL definitions. WiBotic's CAN API has support for the BatteryInfo definition; however, only the temperature, voltage, and current will have values filled in.

For more in-depth information WiBotic's custom DSDL definitions can be used: `WiBoticInfo` and `RadioBaseStation`. Full details about WiBotic's custom DSDL definitions can be found under *WiBotic DSDL Definitions*.

WiBotic DSDL Definitions

WiBoticInfo

WiBoticInfo sends the information available from the on-board charger's ADC packets across the CAN bus.

Parameter	Description
VMonBatt	Battery Voltage
IBattery	Battery Current
VRect	Rectified Voltage
VMonCharger	Charger Voltage
TBoard	Board Temperature
TargetIBatt	Battery Target Current
ICharger	Charger Current
ISingleCharger2	Charger 2 Current
ISingleCharger3	Charger 3 Current

RadioBaseStation

RadioBaseStation sends information on a transmitter that is in radio range.

Parameter	Description
unique_id	MAC address of in range TR
rsssi	Received Signal Strength Indicator of the TR radio

Parameters

Name	ID	CAN Name	Description	Read	Write
Address	3	ADDR	Address of the device on the internal point to point wireless link	Yes	No
RadioChannel	4	RADC	Current device radio channel	Yes	No
DigitalBoardVersion	26	DBRD	Version of the digital board that is running the system	Yes	No
BatteryCurrentMax	34	IMAX	Maximum current that should be delivered to the battery in milliamps	Yes	Yes
ChargerCurrentLimit	35	ILIM	Current limit that the system has decided can be delivered to the battery (in milliamps)	Yes	No
MobileRxVoltageLimit	36	VLIM	Calculated maximum battery voltage (mV) based on chemistry, number of cells, and voltage per cell	Yes	No
RxBatteryVoltageMin	37	VMIN	Calculated minimum battery voltage (mV) based on chemistry, number of cells, and voltage per cell	Yes	No
BuildHash	38	HASH	Version hash of the firmware that is loaded on the device	Yes	No
TargetFirmwareId	39	FWID	Type of firmware image that is running on the device	Yes	No
RxBatteryVoltage	42	VBAT	Last seen value of the battery's voltage (millivolts)	Yes	No
RxBatteryCurrent	43	IBAT	Last seen value of the battery's current (milliamps)	Yes	No
RxTemperature	44	TEMP	Last seen value of the power board's temperature (Celsius)	Yes	No
DevMACOUI	51	MACO	MAC Address Organizationally Unique Identifier	Yes	No

DevMACSpecific	52	MACS	MAC Address Specific Identifier	Yes	No
ChargeEnable	60	ENBL	Enable or disable the ability for this device to transfer power or charge a battery	Yes	Yes
I2cAddress	61	I2C	Address of this device on a I2C bus	Yes	Yes
RxBatteryNumCells	62	CELL	Number of cells in the battery this charger is configured to charge	Yes	Yes
RxBatterymVPerCell	63	CLMV	Voltage of a battery cell (in millivolts) that this charger is configured to charge	Yes	Yes
RxBatteryChemistry	68	CHEM	Chemistry of the attached battery	Yes	Yes
IgnoreBatteryCondition	70	IGNR	Ignore battery condition when charging. Potentially Dangerous.	Yes	Yes
PowerBoardVersion	71	PBRD	Version of the power board that is running the system	Yes	No
AccessLevel	78	ACCS	Current level of access that external entities have to the system	Yes	Yes
CANMessageConfig	82	CNMG	Bitset for which DSDL definitions should be used to send ADC packet information. 0: None 1: BatteryInfo 2: WiBoticInfo 3: Both	Yes	Yes
CANID	83	CNID	ID of the CAN node on the Onboard Charger	Yes	Yes
CoilCheckBaseStation	85	CCHK	Partial MAC of the device that is connected to when in coil range	Yes	No
RecoveryChargeEnable	86	RCOV	Allow slow charging of a battery that has been over discharged	Yes	Yes
CANBitRate	87	CNBR	Bit rate of the CAN device (kbit/s)	Yes	Yes
BatteryRestartPerCell	88	BRPC	Number of millivolts per cell to allow the battery to drop before initiating charge	Yes	Yes

MaxCVChargeTime	89	VTIM	Maximum charge time in seconds while a battery is in constant voltage mode	Yes	Yes
-----------------	----	------	--	-----	-----

Parameter Status Codes

Name	Code	Description
Failure	0	The parameter was not set due to a general failure
Hardware Failure	1	Some hardware did not respond as expected. The parameter was not set.
Invalid Input	2	The data that was to be written to the parameter was not valid for the parameter.
Non-critical Fail	3	The data was not written to the parameter, but this should be anticipated for the given parameter
Read only	4	The parameter is currently in a read only state and should only be read
Success	5	The data to be written to the parameter was written successfully
Not Authorized	6	The current session was not authorized to change the selected parameter
Pending	7	Item is waiting to be processed
Value Clamped	8	The set value has been limited into a safer range for the given parameter

UAVCAN GUI Tool

The UAVCAN GUI tool (https://uavcan.org/GUI_Tool/Overview) can be used for interacting with WiBotic OC's.

Windows

Download and install the latest MSI package from https://files.zubax.com/products/org.uavcan.gui_tool

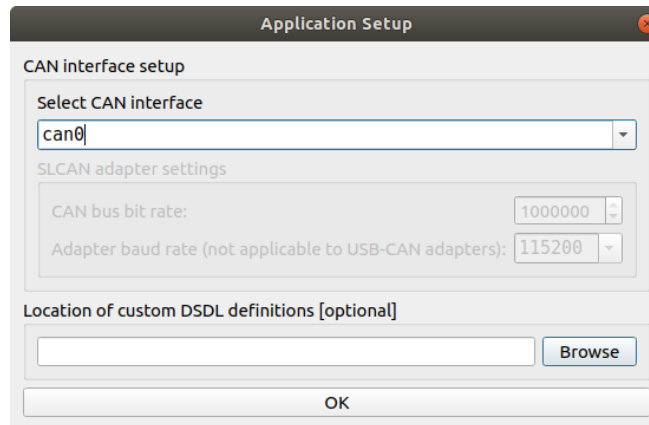
Linux (Debian based)

Make sure the systems package information is up to date by running:

```
sudo apt-get update
sudo apt-get install -y python3-pip python3-setuptools python3-wheel
sudo apt-get install -y python3-numpy python3-pyqt5 python3-pyqt4.qtsvg
git-core
sudo apt-get install git+https://github.com/UAVCAN/gui_tool@master
```

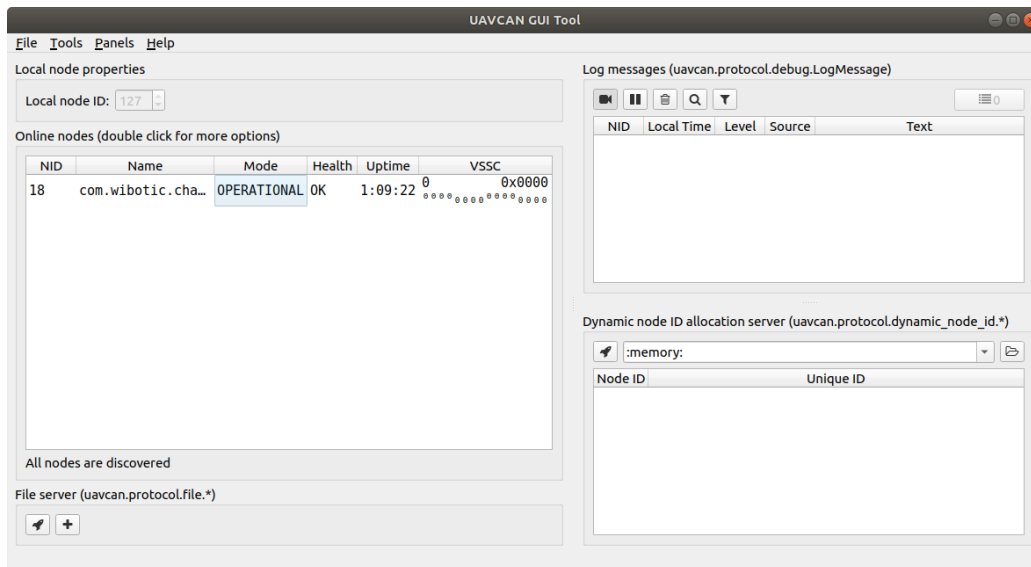
Using the UAVCAN GUI Tool

With the GUI tool installed, it can be launched from anywhere on your computer. For Linux users, open a terminal window and run `uavcan_gui_tool`. For Windows users, search for "UAVCAN GUI Tool" and run the application that is found. A setup window will appear and prompt you to select a CAN interface. If the instructions above for setting up the CAN device have been completed, the device should appear in the list. Select it and click OK to proceed.



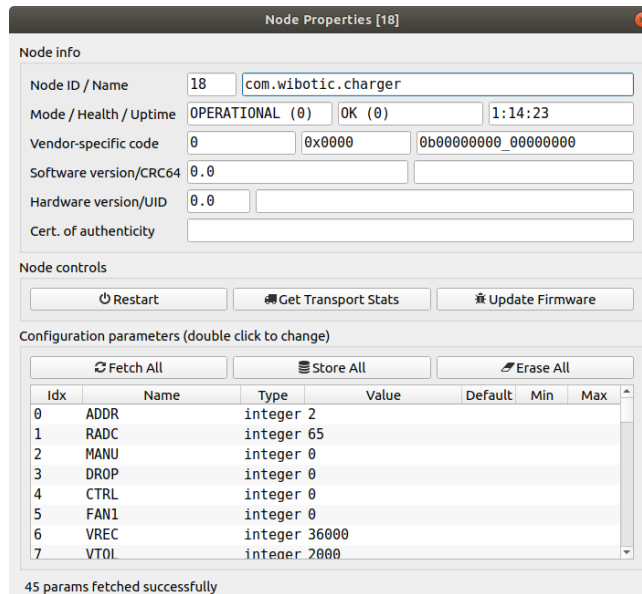
The UAVCAN GUI Tool will appear. At the top of the window, there is an option labeled “Set local node ID”. The default ID is 127 and is fine for this demonstration. This should be changed if there is already a UAVCAN device on your bus using node ID 127.

Click the check box next to the ID to assign it to your device. If your WiBotic OC has been plugged in and powered on, it should show up in the online nodes within a couple of seconds.

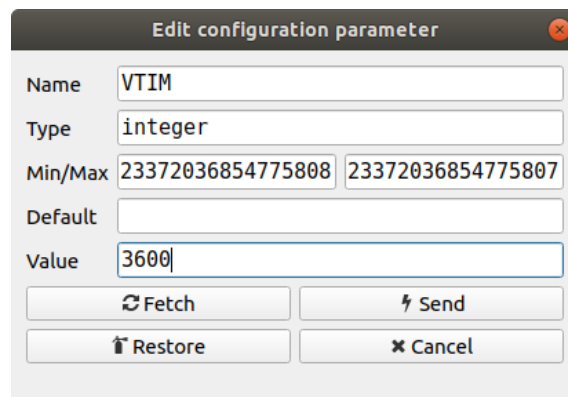


Configuring System Parameters

Double click on the online WiBotic charger to view detailed information about the UAVCAN node. From here, parameters can be inspected and manipulated, and firmware can be updated. To view system parameters, click “Fetch All” under the Configuration parameters subsection. The list will populate with parameter names and values.



Double clicking on a parameter opens up a window that will allow you to set a new value. For this example, scroll to the bottom of the list and double click the parameter named “VTIM”. From the parameter description above, we can see that VTIM corresponds to the parameter MaxCVChargeTime, which sets the maximum time that an OC will charge a battery in CV mode. In the value box, change the number to 3600, meaning 3600 seconds (or 1 hour), and click “Send”.



The parameter will now be changed on your system. Close the window and you will see that the parameter value has been updated. To save the parameter to memory so that it will remain 3600 across reboots, click the “Store All” button. **NOTE:** This will save all parameters that have been changed. If you only wish to save certain parameters, set and save those first before manipulating other parameters.

Updating System Firmware

Firmware can also be updated using the UAVCAN GUI Tool. From the same Node Properties window, click the button labeled “Update Firmware”. A dialog box will pop up indicating that a local dynamic node ID allocator is not configured. This is not necessary for upgrading firmware on WiBotic OC’s, so click Yes to continue.

A file selector will appear. Select a WiBotic OC firmware file and the update will proceed automatically. You will see the mode of your WiBotic OC will change to “Software_Update” as the update is written to your device. When the update completes, the OC will reboot and the mode will change back to “Operational”.

WiBotic Inc.

9706 4th Ave NE – Suite 208

Seattle, WA 98115

206-580-0900

www.wibotic.com

info@wibotic.com