**Serial Communications**

**To get Realterm, go here:**

[**https://sourceforge.net/projects/realterm/**](https://sourceforge.net/projects/realterm/)

**Then, to set it up, connect the board to the computer with the USB plug. Usually, green is data+, white is data-, red is Vcc, and black is ground.**

**Next, connect the board to the power supply (The Nissan leaf board uses an external 12v, the chevy volt board uses 12v, the control/driver board uses 24v)**

**Then, run realterm.**

**On the “Display” tab, set “Rows” to 40, and click the “Scrollback” box and set it to 2000 for the scrollback.**

**Now, go to the “Port” tab, and change the “Baud” to 115200. The “Port” should already have detected the right port. Click the “Change” box, so that it will now have the baud rate updated.**

**Now, click on the black open text box area at the top, and hit enter. You should see a message like “Invalid command. Hit ‘?’ to see a list of valid commands”.**

**You are now ready to communicate!**

**current-sensor-amps-per-volt (not needed for the Leaf board/Leaf motor)**

The way you use this is, if your current sensor has a feedback of 2.5 volts + 1 volt at “x” amps, then you would type:

*current-sensor-amps-per-volt x*

So, in the case of the LEM Hass 300-s, the datasheet says that at 0 amps, the current sensor feedback is 2.5v, and at 300 amps, the voltage is about 2.5v + 0.625v. So, the current sensor amps per volt is 300/0.625 amps = 480 amps. So, if you are using a LEM Hass 300-s, you would type:

*current-sensor-amps-per-volt 480*

If you are using one of the Nissan Leaf control boards with a stock Leaf controller, you would use 400. For the Chevy Volt stock controller, you would use 350.

**run-pi-test2 (not needed for the leaf board/leaf motor)**

This is the first test that must be done for the motor. It works on all motor types. Start with a battery pack voltage < 100vDC, but at least about 50vDC. If you have no idea about the motor’s characteristics, type

*run-pi-test2*

It will display a bunch of numbers that are from 0 to around 500. The goal is for the numbers to fall to zero quickly. So, the data might look like this (it always starts at zero. Ignore that one!):

0,500,498,425,402,356,322,300,288,233,188,130,89,63,22,-30,15,-25,44,-12,34,-55,27,…

That would be a good result. It quickly converged, in less than like 30 iterations to around zero. It’s OK that it bounces back and forth once it hits zero.

If you are using a Nissan Leaf motor and my Nissan Leaf control board, don’t worry about this.

**kp/ki (not needed for the Leaf board/Leaf motor)**

To change the proportional gain of the PI control loop for the torque control, type:

kp xxxxx

To change the integral gain of the PI control loop for torque control, type:

Ki xxxxx

where ‘xxxxx’ is a number from 1 up to 20000. Here’s an example of the usage:  
kp 6200

ki 100

For best results, kp should be about 50 or 100 times bigger than ki. Once you have changed kp and ki to a new value, if you want, you can run ‘run-pi-test2’ to see if the results are good.

**current-angle-offset**

for motor type 2 (permanent magnet with encoder, with one index pulse per mechanical revolution), and motor type 3 (1 index pulse per electrical revolution, which is common when using a resolver. I use this setting for the Nissan Leaf motor and the Remy). The angle offset is the difference between the magnetic north of the motor and the index pulse of the position sensor. You can change it to be in the range of 0 to 511, where the circle is divided up into 512 “degrees”.

This is one of the configurations for the motor that must be done once to find the ideal angle offset. Before doing this procedure, you should find good Kp and Ki values. To do this, follow this procedure:

angle-offset 0

attempt to turn the motor with throttle

angle-offset 20

attempt to turn motor with throttle

…

Keep going until the motor changes directions. Then, reduce the angle offset until the motor no longer turns. The motor may not turn for a range of angle offsets. For example, like this:

angle-offset 138 makes the motor turn clockwise

angle-offset 139 makes the motor not turn at all

angle-offset 140 makes the motor not turn at all

…

angle-offset 163 makes the motor not turn at all

angle-offset 164 makes the motor turn COUNTER-clockwise.

The ideal angle offset for the motor NOT MOVING in this case would be (139 + 163)/2 = 151. What you have found is the absolute WORST angle offset for the motor. Now, to find the ideal angle offset, either add or subtract 128 (the scale is 512 “virtual degrees” in a circle, so 128 “virtual degrees” is the same as 90 real degrees). So, you would type:

angle-offset 279

Or

angle-offset 23

**max-regen-position**

**min-regen-position**

**min-throttle-position**

**max-throttle-position**

The throttle feedback to the controller will be a signal in 0 to 5v. The controller interprets that as a number from 0 to 1023. The throttle is similar to how they do it on a Tesla and the Nissan Leaf. For some range for the first few % of throttle, it is variable regenerative braking. Somewhere in the middle is zero torque command, and the last bit of throttle is variable positive torque command. So, it might be something like this:

*max-regen-position 59*

*min-regen-position 150*

*min-throttle-position 220*

*max-throttle-position 942*

100% throttle would be around 942 for the raw feedback, and 0% throttle would have a raw feedback of around 59. Zero torque command would be any raw value in 150 up to 220 in this case.

**max-motor-amps**

**max-motor-amps-regen**

This basically sets the maximum torque producing current (Iq) for a permanent magnet motor, and it sets the maximum Id and Iq for an induction motor. This is not battery current, but just current in the motor. You would use it like this:

max-motor-amps 300

max-motor-amps-regen 75

With those settings, you would see 300 amps for Iq in the motor when the throttle is at max-throttle-position. And at zero throttle, you would see -75amps for Iq. So, if you want your car to have very strong regen braking at zero throttle, set the max-motor-amps-regen to something very high. If you want to be thrown back in your seat at takeoff, set max-motor-amps to something very high.

**max-battery-amps**

**max-battery-amps-regen**

This sets the maximum current allowed to go into/out of the BATTERIES. You would use it like this:

max-battery-amps 200

max-battery-amps-regen 50

**pole-pairs (not needed for the Leaf board/Leaf motor)**

this is the number of pole pairs of the motor. One way to find out the number of pole pairs for a permanent magnet motor is to turn the motor shaft at 1 revolution per second, and then measure the Hz on 2 of the phases with a volt meter. It is used for finding the mechanical RPM. Usage:

*pole-pairs 4*

If you are using a Nissan Leaf motor, the number of pole pairs is 4. But this will already be set if you are using my Leaf control board replacement.

**max-rpm**

This sets the maximum RPM of the motor shaft. It will hit this, and then bounce against it like in a traditional car (“rev rev rev rev”). Usage:

*max-rpm 4500*

That would set the max RPM to 4500 RPM.

**config**

This just shows how the controller is currently configured. Pole pairs, throttle settings, etc… Usage:

*config*

**precharge-time**

This sets the time for the precharge delay, in tenths of a second. Example usage:

*precharge-time 59*

That would set the delay to 5.9 seconds from the time of closing the precharge relay to closing the main contactor. A very safe time delay is 5 RC time constants. The C is the capacitance of your bus capacitor. The R is the resistance of your precharge resistor. So, let’s say you have a bus capacitance of about 1200 microfarad. That means C = 0.0012 Farad. Also, let’s say you are using a 470 Ohm precharge resistor. Then, you would set the precharge time to 5\*470\*0.0012 seconds = 2.82 seconds. So, you would type:

*precharge-time 29*

*save*

One other thing to consider is the energy dissipated in the precharge resistor. You can’t just use any resistor. The total energy during precharge is 0.5 \* C \* V\*V, where C is the bus capacitance, and V is the final bus voltage. Let’s say the bus voltage is 400 volts. So, in our example, the total energy through the resistor is 0.5 \* 0.0012 \* 400 \* 400 Joules = 96 Joules. So, you would need to choose a resistor with a Joule rating greater than 96 Joules. Most resistors don’t list their joule rating, so I just use something from this family of resistors:

<https://www.mouser.com/ProductDetail/Ohmite/AY821KE?qs=sGAEpiMZZMvmQ%252bOLa8n%2fMxoZ6mTh6DRODpV4zT6BJZY%3d>

**data-stream-period**

This starts the datastream. You tell it the delay, in milliseconds. Example of usage:

data-stream-period 100

That will start streaming the data that has been enabled for streaming, and will stream it at 100ms intervals. To stop the datastream, type “off”.

**stream-raw-throttle**

This is one example of enabling something for streaming. This adds the raw throttle to the datastream. Usage:

stream-raw-throttle 1

If you want to remove raw throttle from the datastream, type:

stream-raw-throttle 0

There are lots of things that can be added to the stream. To see a list, type “?”.