Getting Started on Windows and Raspberry Pi with Electrodacus & Wifi Interface & MQTT

v.007 (for SBMS0 v03d, Wifi Firmware v0.3-dirty)

Sherry McCampbell

REVISION LOG

.007	23-Jan-2026	After updating, nodejs (25.3.0), node-red (4.1.5), and mosquito (2.0.22) versions to latest stable, had a problem getting mosquito to work as a service. Added info on that procedure. Also added info on the wifi misconfiguration issue I had in 2023 and some other notes about getting the wifi working. And info on Victron MPPT vs Morningstar. Still need to get a copy of the new version of the screen.
.005	25-Mar-2023	Updates based on new "clean" install on a new computer (Windows 11 Pro v22) and node.js v16.15.1 and node-red v3.0.2
.004		New versions of node.js and node-red
.003	25-May-2022	Added info on the Palette nodes I use in my flows. Removed json code in the Appendix and added a link to download the files from the website.
.002	16-Apr-2021	Updated instructions for installing new nodes to the palette, and updated JSON files. Added some notes on lack of success getting Mosquitto running as a Windows service.

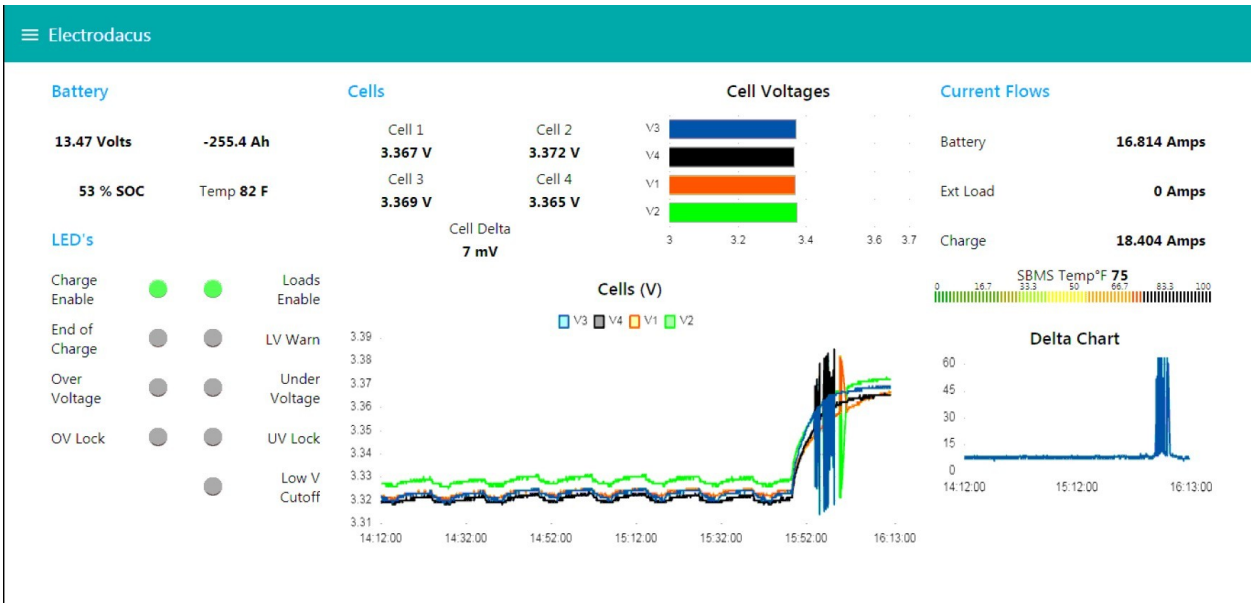
1 Using MQTT And Node Red to Monitor Your Electrodacus

1.1 What is MQTT and Why Do I Need It?

This is a monitoring setup I made using MQTT and Node-Red on a Windows 10 computer. It is also very easy to also do on a Raspberry Pi. It runs in a browser window.

This is VERY specific to an Electrodacus, and the MQTT data that the Electrodacus wifi module "publishes".

The visuals only show the current status and the last 2 hours of data, but I am logging in a CSV file midnight-to-midnight, every 10 seconds.



The squigglies in this graph show periods of balancing. After a little experience, you learn to read this graph. (Tech note: This snapshot is really at nearly 90% SOC but we haven't run the new setup up to 14.2 to reset the SOC yet. Once the Electrodacus SOC gets to 100%, the calculated Ah (upper left corner) resets to zero, and after that, it tracks pretty well with our onboard Ah monitor (Link 2000 Marine volts and amps monitor).

The Electrodacus does not output Amp Hours, this is a calculated value (ie it is an estimate, not perfect).

This setup is using a Mosquitto MQTT Broker (free software), and Node Red (free software) to receive the MQTT package (send from the Wifi module of the SBMS0) and break out the data into human readable, graphable, and loggable data.

It's also possible to stuff the received data into a database, for longer term data analysis, but I have not done that yet on my system.

1.2 How Hard is this Going to Be?

I was a computer programmer 15 years ago, in the Windows environment. I haven't programmed since (so my skills are pretty obsolete). A year ago I had never heard of node-red. A couple of months ago, I had never heard of MQTT, and I only vaguely understood "IOT" (the Internet of Things).

A person who is not fairly computer savvy will have trouble setting things up using my guide, as it is not detailed enough. But there are lots of online resources, including hold-your-hand how-to's to learn node-red and mosquitto. The documentation out there on the internet is truly amazing.

I did a LOT of Googling to figure things out.

Somebody recommended this Youtube playlist, called Node-Red Essentials, for people just getting started with Node-Red. But for a real newbie, this is too programmery.

https://www.youtube.com/playlist?list=PLyNBB9VCLmo1hyO-4fIZ08gqFcXBkHy-6

A really basic "Getting Started" for Raspberry Pi is here:

https://projects.raspberrypi.org/en/projects/getting-started-with-node-red

1.3 Hardware Setup

First, our solar/LifePO4 installation is on a 44-foot catamaran sailboat. We use this as our "house bank" to run our 12v system onboard.

We typically sail in warm waters (so there is no consideration in our efforts for cold weather monitoring). We are halfway through a round-the-world sailing trip, and have been living off-grid on our sailboat for 15 years. You can check out more details on our website: http://svsoggypaws.com

We have the following new battery setup:

- 1560 Watts of solar coming in at 36v providing approximately 70-100 Amps at peak sun
- Two Solar Controller which reduces the 36v to just above battery bank voltage at maximum amps. Though the Morningstar does not have a LifePO4 charge profile, per se, it is configurable enough to be able to set all the charge parameters necessary to ensure safe charging and long LifePO4 cell life.
- 8x272 Amp Hour 3.3v LifePO4 cells wired into one 12v 545 Ah bank

In addition to solar, our other charging sources are:

- A Sterling 110/220v “shore charger” that does not have a proper LifePO4 charge profile. We only hook this up when absolutely necessary and then only charge under close supervision.
- Twin engines, each fitted with with externally-regulated 60Amp alternators that DO have proper LifePO4 charge profile.

Our BMS hardware is a single Electrodacus SBMS0 v03d (late 2020) with a “stock” Wifi/USB module running firmware b1d8f5ec).

We have a small boat wifi access point (router) that is currently connected to the internet via a cell connection (though, a network connection is not required except to download install modules, or to access from off the boat).

Computer-wise, for monitoring the Electrodacus, here is a very crude block diagram of what’s going on.

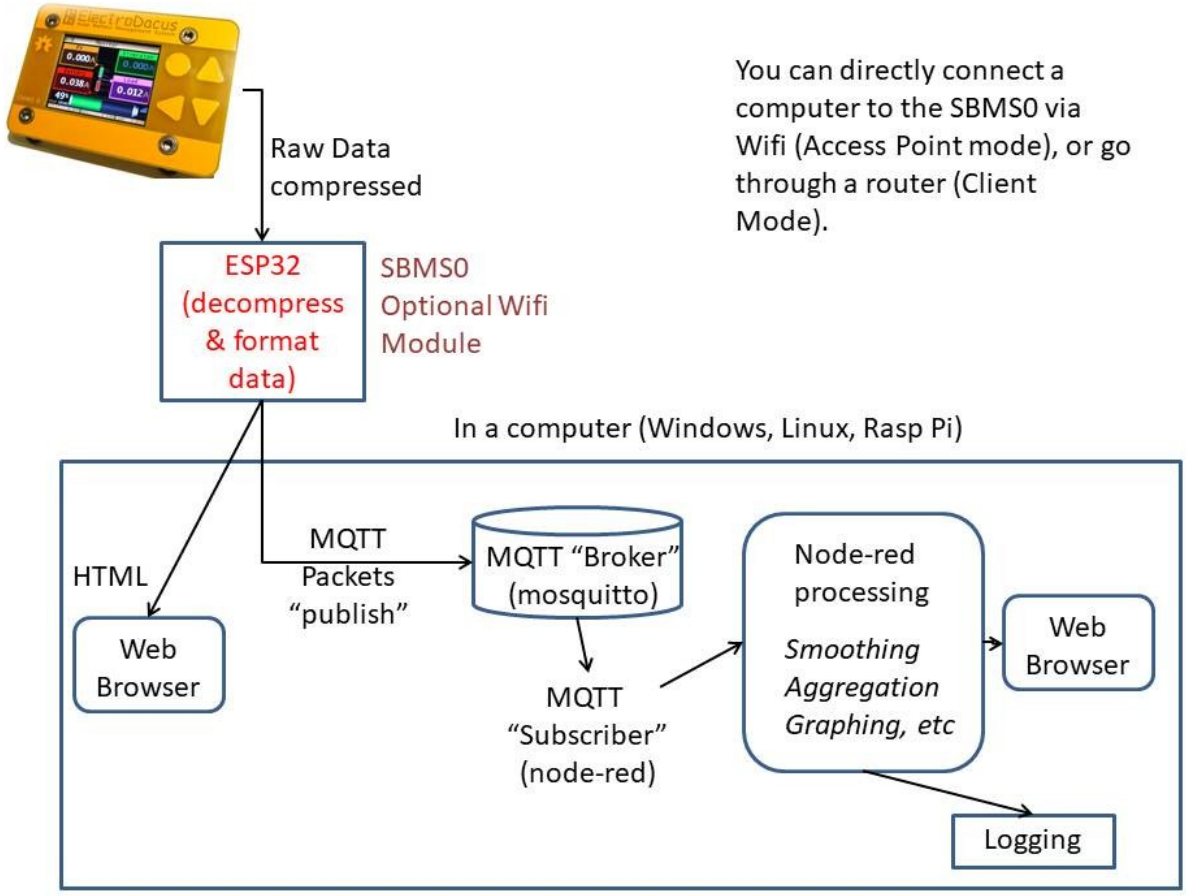


Figure 1-1 Electrodacus Monitoring with MQTT (Block Diagram)

1.4 Software Setup

Though the Electrodacus can actually be an access point itself, the better solution is to connect the Electrodacus as a client on a local wifi access point (via wifi). This allows any computer or wireless device on the network to be able to monitor the Electrodacus (however, the Electrodacus is designed to only be connected to by only one device at a time).

1.4.1 Changing the Electrodacus Settings to Connect to your Local Network

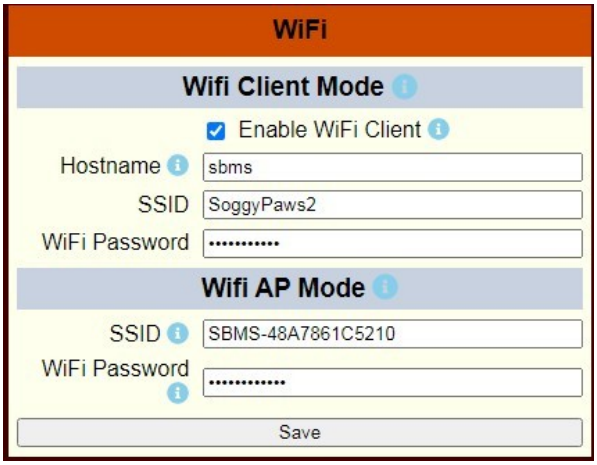
When it first boots up, the Electrodacus with the optional wifi module creates a little wifi hotspot, named SBMS-XXXXXXXX. You need to connect to this hotspot with a computer or mobile device. The default password is "electrodacus". Once connected, bring up a browser window and put this in for the URL:

http://192.168.4.1

The “Legacy” page looks like this:

Figure 1-2 SBMS0 Basic Monitoring Screen

If you wish to connect the Electrodacus to an existing network, go to the settings page, and check the “Enable Wifi Client” checkbox. You can leave the Hostname at the default setting of sbms. And you put your own access point’s information in the SSID and password. Click Save.



<- Wifi Setup Page in your Browser Window

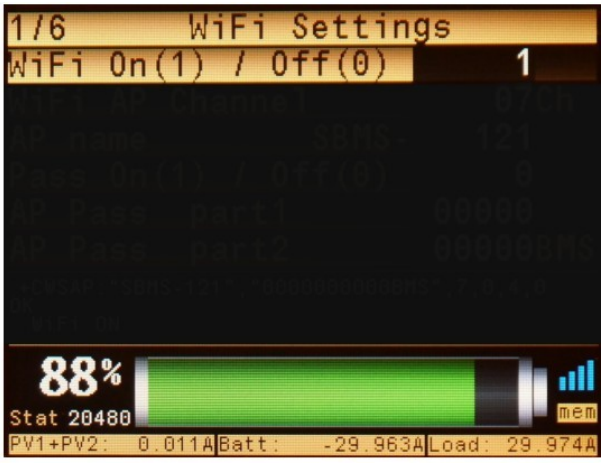
http://192.168.4.1/index.html#/settings

NOTE that SBMS can only connect to a 2.4Ghz router, not a 5GHz router. Make sure you put in the correct SSID if you have a dual band router!

IMPORTANT: Then you need to cycle power on the wifi board. This can be done by cycling power on the Electrodacus, but it’s better to go to the Device

Settings page on the Electrodacus, and set the Wifi first to zero, Save Device Settings, and then back to 1, and again Save Device Settings.

Electrodacus Device Settings / Wifi Page -> (on the Electrodacus)



This is the page to cycle power on the wifi board without having to cycle power on the whole SBMS0

Once the power cycles, when the wifi module boots up, it will attach itself as a Client on your router.

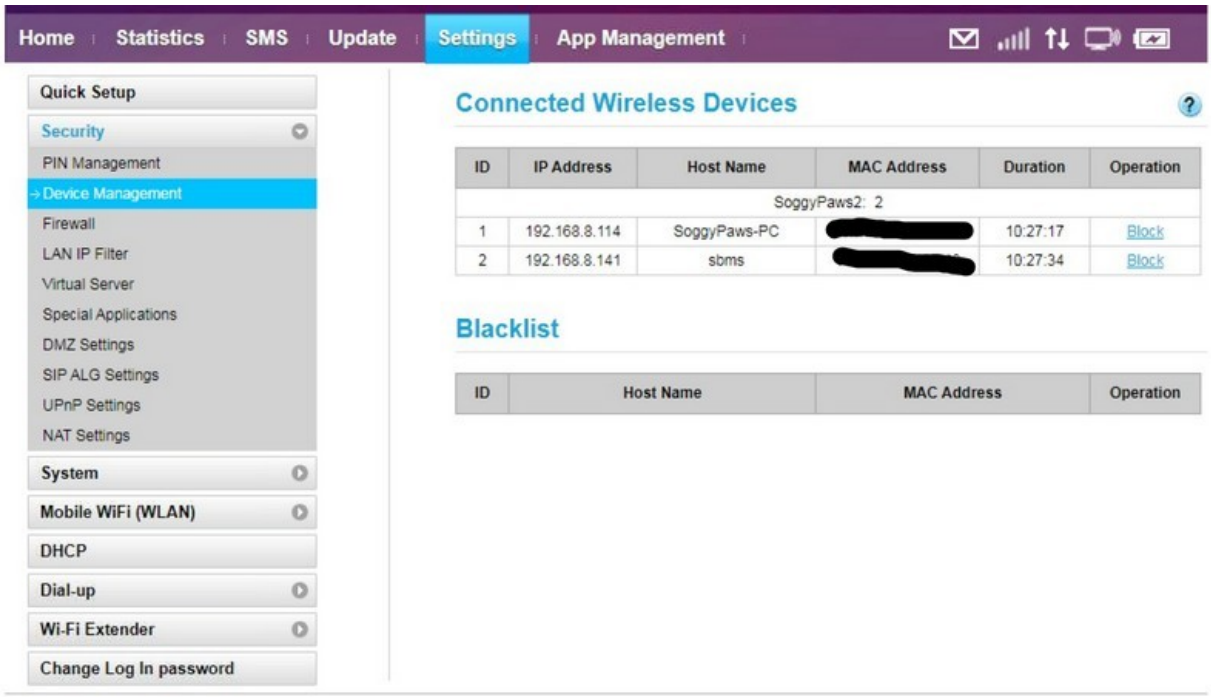
If everything is working, the Electrodacus should now be connected to your router.

You need to then determine what IP address was assigned to the Electrodacus when it connected.

It won't be 192.168.4.1 anymore.

1.4.2 Finding the Electrodacus & Node-Red Computer's IP Address

If you know how to access your router's configuration pages, the easiest way to find the IP address of the SBMS0 and your own computer is to look on the Admin Devices page, which shows you all the devices connected to the router. Look for one with a "hostname" of "sbms" and that is the IP address of the SBMS0.



In this case, my computer IP address is 192.168.8.114 and the SBMS0 address is 192.168.141

You will then use the SBMS0 IP address to access the "Legacy" and "Settings" screens of the Electrodacus Wifi module, similarly to the mechanism above in Section 1.3.1, but using the new IP address of the SBMS0 that it was assigned by your router when the sbms0 connected to your router. Example:

http://192.168.8.141/index.html#/settings

If you don't know how to access your router configuration pages, on your computer, open a command prompt (type CMD in the search window and click "Open CMD Prompt App") and type in "ipconfig".

This will give you the network configuration of your own computer. Mine has a bunch of entries, but what you are looking for is something like this (this assumes you are connecting your computer to your router via Wifi):

```
Wireless LAN adapter Wi-Fi:
   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::3504:ba5d:6442:8036%8
   IPv4 Address. . . . . . . . . . . : 192.168.8.114
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.8.1
```

The IPv4 Address is YOUR computer's IP address. You'll need to enter this into the MQTT configuration window on the Electrodacus later. In this example it is this: 192.168.8.114.

Then enter "arp -a"

The easiest way to tell your own computer's IP address is to bring up a cmd prompt and type in "arp –a"

```
Interface: 192.168.8.114 --- 0xb     <--- My computer IP Address
  Internet Address      Physical Address      Type
  192.168.8.1           1c-15-1f-88-8a-08     dynamic
  192.168.8.141         10-52-1c-86-a7-48     dynamic    <--- SBMS IP Address
  192.168.8.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.1.60            01-00-5e-00-01-3c     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

The entry in blue is YOUR computer's address and the entry in red is the address of the SBMS0 (by a process of elimination). (x.8.1, and x.8.255 are router-specific addresses). If you have a lot of things attached to your router, this list may be long and it may not be easy to figure out which one is the SBMS.

1.4.3 Confirming It Works

Putting http://192.168.8.141/ in my browser window now connects me to the Electrodacus wifi system, with the same access as when directly connected to the Electrodacus as described in Section 1.3.1.

So now I can access the Electrodacus screen from any device connected to my local network.

But there are a few things not provided on this screen. So I wanted to try to build my own interface. One way would be to modify the provided HTML screen, and then re-flash the wifi module with the new HTML screen. This seemed risky to me.

The second way is to take advantage of the new feature built into the wifi module… MQTT data. In short, a MQTT client/publisher attached to a server/broker and sends a specific set of data. The server/broker then provides the ability for other clients/subscribers to display data available from the broker.

1.4.4 Recovering from Wifi Problems

IP Address Issues: Originally I did not hard assign the IP address of the Electrodacus in my router. It mostly always connected to the right IP address. I found that sometimes another device (cell phone) would connect first and grab the IP address that the Electrodacus normally connected via. It doesn't affect the MQTT stuff, but does affect being able to pull up the Electrodacus' own screen. I finally figured out how to hard-assign a specific port to the Electrodacus and that solved that problem.

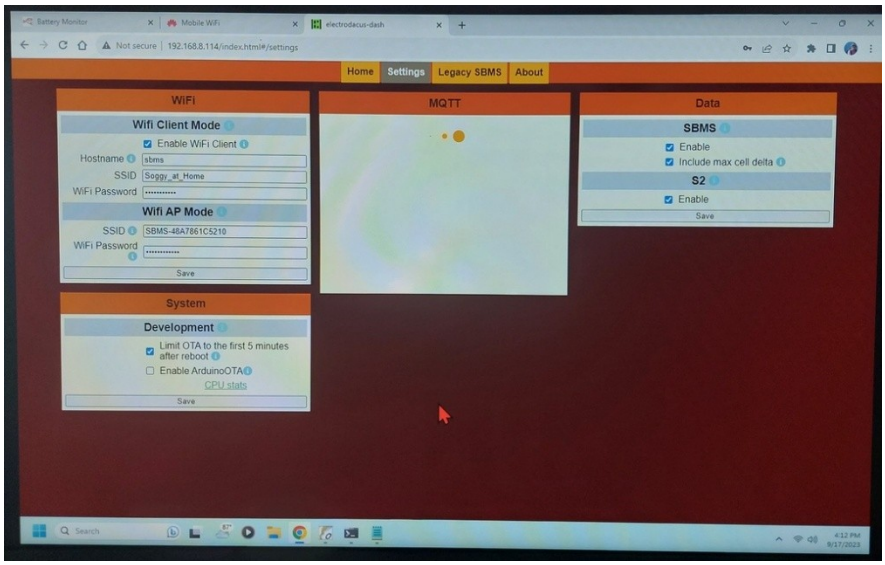
Electrodacus not connecting: Several times if I've taken down the router for various reasons, once the router is back up, the Electrodacus refuses to re-connect. (can see by looking for the Electrodacus SSID in your computer's wifi list). In all cases, going into the Electrodacus setup, Device Settings Page, and toggling wifi off and then back on, the Electrodacus re-establishes the connection.

2023 – Bad Configuration: I recently changed from one router to another, and successfully changed the settings in the sbms0 wifi module. The two routers have 2 different base ip addresses, so had to change both the wifi connection part and the MQTT part. Everything was working!

I found the new router keeps hanging, so I wanted to change back to my old router. When I changed the settings again in the wifi module configuration, rather than using the IP address for the computer in the MQTT settings section, I put in the computer's name: DESKTOP-LA5C070.

Maybe I mispelled it, or maybe it's too long, or maybe it really only will accept a hard IP address vs the "host name". It didn't complain when I entered that name and clicked the save button.

After cycling power on the wifi module, it connects to the computer via the new wifi SSID, but the MQTT module isn't working, and the MQTT configuration page is not working. See screenshot.



Answer: Flashing firmware is quite easy now that binary releases are available. No need to maintain a build environment and compile source yourself any longer.

Instructions follow, but if you do not want to do a full reflash, you can also attempt my alternate fix below

You can download the binary release from:

https://github.com/armageddon421/electrodacus-esp32/releases/tag/0.4.1

Get the firmware.bin and the spiffs.bin file

To upgrade simply

- Reboot the wifi module
- upload firmware.bin to the web interface at (In your case) http://192.168.8.114/update
- upload spiffs.bin to the web interface at (In your case) http://192.168.8.114/update
- Module should reboot automatically, if not reboot.

Please note this will completely wipe your settings so you will start all over again, meaning you have to connect to the wifi hotspot at 192.168.4.1 and enter your own wifi credentials etc.

An alternative fix in your specific case would be to just overwrite the /cfg/mqtt file. The following is untested as I'm on a mac but should work for windows:

Open powershell and issue the following commands:

$mqttsettings = Invoke-WebRequest
https://raw.githubusercontent.com/armageddon421/electrodacus-esp32/master/data/cfg/mqtt

Invoke-RestMethod -Uri "http://192.168.8.114/cfg/mqtt" -Method 'Post' -Body $mqttsettings

Then reboot the wifi module.

Your 2nd procedure, explained very clearly, was very easy to follow and allowed me to recover and re-set-up everything properly. I didn't try the first method. Thank you so much for taking the time to explain it clearly, step-by-step.

Actually, it turns out that what I had recorded to do was your first procedure, not the second.

That was easy and worked fine, and apparently upgraded my Wifi module software.

2 Getting MQTT running on Windows

2.1 Installing MQTT (mosquitto)

Mosquitto is just ONE of a number of possibilities for a “broker” (server) for MQTT on Windows, Linux, and Mac. There are also “Cloud” broker systems. The advantage of a cloud-based system is you can access from anywhere you have an internet connection, not just when connected to your local router.

However, I am setting up a system that will not always be connected to the internet, so I chose to start with an MQTT server set up on my local computer. I chose Mosquitto specifically because several people had mentioned it as a good open source solution for a locally hosted MQTT broker.

Download Mosquitto from here:

https://mosquitto.org/download/ I downloaded the latest version (2.0.8) Approx 1.5Mb

Install default install. I used x64 version as I have a 64-bit Windows 7 machines (also later installed it on a Windows 10 Home system with no problems).

I used this tutorial to guide my install and configuring, but it’s not quite up to date

http://www.steves-internet-guide.com/install-mosquitto-broker/

When I installed on an old Windows 7 computer, there were no dependencies noted, but when I tried to start up the server/broker, I got the error message: “Cannot proceed
because VCRUNTIME140.dll was not found”

So I went to here:

https://www.microsoft.com/en-ph/download/details.aspx?id=48145

And downloaded and installed the vc_redist.x64 package. (approx. 14.5 Mb)

When I did the same install on a new Windows 10 computer, I don’t think I had to install this.

I also downloaded the SSL installer as described in the Steve's Internet Guide document linked above (Win64OpenSSL_Light-1_1_1j.exe), but it must now be included in the basic mosquito install, as I did not need to run this install.

The mosquitto install installed into this directory on both my Windows 7 and Windows 10 computers:

C:\Program Files\mosquitto

including the data files (which would normally be put into c:\ProgramData on a proper Windows install.

2.1.1 Windows 11 Pro

On a new computer with Windows 11 Pro, the first time

2.2 Minimum Configuration Required

2.2.1 mosquitto.conf

The mosquitto configuration (mosquitto.conf) file is located in the main mosquitto directory (mine ended up in C:\Program Files\mosquitto). It is about 800 lines long (a lot of that is explanatory text, but it is still intimidating the number of options). Pretty much everything in the file as installed is commented out, with explanations for what the default setting is. Here are the only things I added or changed in this file:

```
# listener port-number [ip address/host name/unix socket path]
listener 1883

allow_anonymous false

password_file passwdfile.pwd
```

Note that in the latest version of mosquitto, logging to the console is not turned on by default. You have to look through the mosquito.conf file and figure out how to log informational messages to the console (so you can see what's going on, as you get things set up). Once you have your setup working, you can turn off logging to the console, if you like.

To turn on logging to the console, add this to your mosquito.conf file:

TBD

I used a simple text editor to edit the file (something like Notepad will work, right click and say "open with").

However, Windows would not let me save the file once I finished editing it. I had to save it somewhere else, and then copy it to the mosquitto directory using Admin priviledges.

2.2.2 passwdfile.pwd

The password file, I also created with Notepad. It is formatted like this:

```
user1:user1password
user2:user2password
```

I created 3 users in my password file…one for the electrodacus message, one for my node-red instance (configured in the mqtt in node in node-red) and one for myself, to be able to connect and play.

Save it somewhere else as text, and rename it passwdfile.pwd. Then copy that file to your mosquitto folder (you'll again need Administrator privs).

IMPORTANT: Note that if your version of Windows does not have "show file extensions" turned on, the file you create with Notepad will likely be named passwdfile.pwd.txt, which won't work.

Then you need to encrypt the password file in the mosquitto folder, as leaving it in clear text is not very secure!

```
mosquitto_passwd -U passwdfile.pwd
```

Windows 11 Pro: With the latest version of W11 (v23), I had a LOT of trouble getting the password file recognized by Mosquitto. I finally had to set the file priviledges to everyone can do everything. There is likely a more specific way to do this, but I tried for an hour or two and couldn't figure it out.

Now when you start up the broker, it will read the conf file and the pwd file, and you have BASIC password protection for your simple MQTT system. This is fine if you are only ever going to use your system on a local network. If you will be allowing access from elsewhere, outside your local network, you need to research better access control.

2.3 Testing Your Basic Mosquitto Install On One Computer

I used this tutorial to confirm that my broker is running and I can subscribe and publish something from both users.

https://medium.com/jungletronics/mosquitto-user-access-configurations-setups-2f95dc593adf

(basically get 3 cmd windows running, one is the broker, one is the subscriber, and one is the publisher)

Broker startup:

```
c:\Program Files\mosquitto>mosquitto -c mosquitto.conf -v
```

Subscriber startup:

```
c:\Program Files\mosquitto>mosquitto_sub -h localhost -p 1883 -u user1 -P
user1password -t temperature
```

this subscribes user "user1" to the "temperature" item

When the subscriber starts up, you should see something like this on the Broker screen:

```
1614753884: New client connected from ::1:62269 as auto-2017FC54-7D68-5B0B-
E026-
E3FFF1B7DC43 (p2, c1, k60, u'user1').
1614753884: No will message specified.
1614753884: Sending CONNACK to auto-2017FC54-7D68-5B0B-E026-E3FFF1B7DC43 (0,
0)
1614753884: Received SUBSCRIBE from auto-2017FC54-7D68-5B0B-E026-E3FFF1B7DC43
```

Publisher startup:

```
c:\Program Files\mosquitto>mosquitto_pub -h localhost -p 1883 -u user2 -P
user2password -t temperature -m 45
```

This "publishes" a temperature item at 45 degrees.

When the publisher starts up, you should see something like this on the Broker screen:

```
1614749883: New connection from ::1:62077 on port 1883.
1614749883: New client connected from ::1:62077 as auto-F21A65D8-8CF8-C0D3-
A016-55ED43D3EFD1 (p2, c1, k60, u'user2').
1614749883: No will message specified.
1614749883: Sending CONNACK to auto-F21A65D8-8CF8-C0D3-A016-55ED43D3EFD1 (0,
0)
```

```
1614749883: Received PUBLISH from auto-F21A65D8-8CF8-C0D3-A016-55ED43D3EFD1
(d0, q0, r0, m0, 'temperature', ... (2 bytes))
1614749883: Sending PUBLISH to auto-E357B5BA-300D-7413-6C16-A42E522F2440 (d0,
q0, r0, m0, 'temperature', ... (2 bytes))
```

Now, every time you re-send the publisher command,

```
mosquitto_pub -h localhost -p 1883 -u user2 -P user2password -t temperature -m 45
```

you should see something like this on the subscriber screen:

```
45
45
```

And this on the Broker Screen:

```
1614749883: Received PUBLISH from auto-F21A65D8-8CF8-C0D3-A016-55ED43D3EFD1
(d0, q0, r0, m0, 'temperature', ... (2 bytes))
```

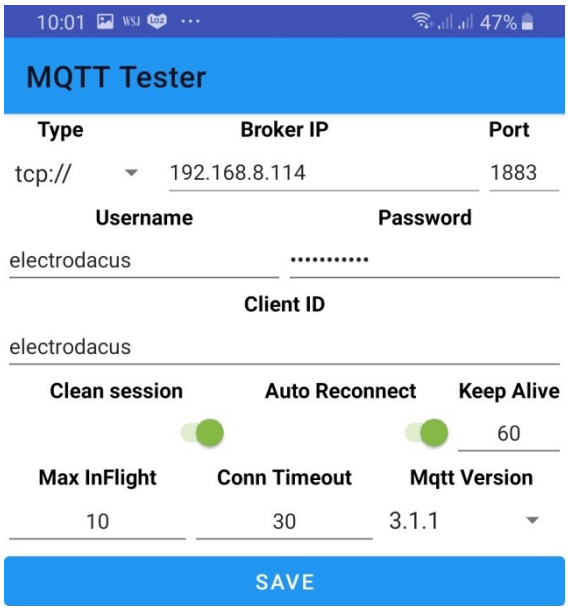
2.3.1 Using an Android to Check from Outside the Computer

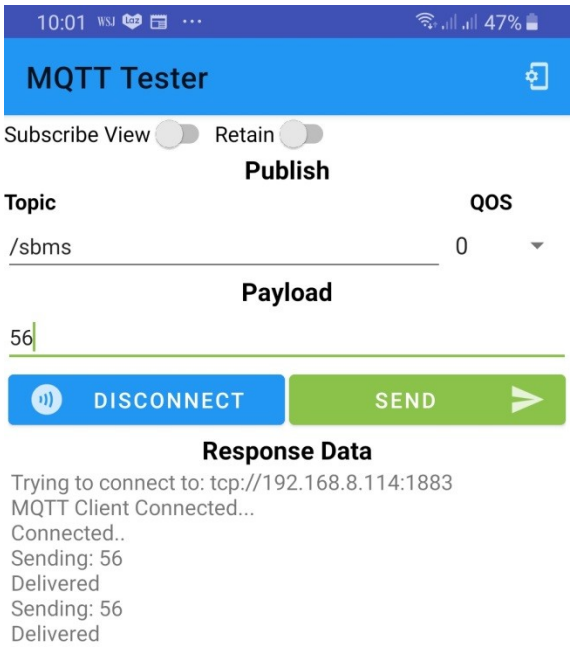
The next step I tried was configuring the Electrodacus wifi module to send the MQTT data from the Electrodacus. I did that, but saw nothing on my Broker.

So I looked for an alternative to test my setup, to make sure it wasn't a Windows firewall issue. One way would be to use another computer on the same network using the mosquitto publish function, and instead of "localhost" in the 2nd computer, the mosquitto_pub should look like this:

```
mosquitto_pub -h 192.168.8.114 -p 1883 -u user2 -P user2password -t temperature -m 45
```

But I didn't want to go to the trouble to configure another computer with mosquitto. So I searched for an Android option. I found a simple free MQTT Tester app on the Google Play store, and in 5 minutes had it sending (publishing) something from my phone to my broker on the computer.





2.4 Installing the Mosquitto User Interface (aka Management Center)

I stumbled on something while I was googling around that looked interesting, but I haven't taken the time to investigate whether I need or want this. Just leaving the link here for now.

Haven't done this yet! Not sure it's necessary, but it might make it easier to manage and view what's happening with your broker.

https://docs.cedalo.com/latest/docs/management-center/mc-overview

Maybe version .003 of this guide will cover this.

2.5 Setting the Mosquitto Broker up to Operate as a Windows Service

I don't remember exactly the details, of which is the default when you install mosquitto on Windows. But it can be run either manually from the command line, as I am showing above, or automatically as a Windows Service. Once you've got your system running well, you might want to run the broker automatically as a Windows Service.

To bring up the Windows Services screen, type in Services in the Windows Command window.

Scroll down to the "M"'s and you should see Mosquitto

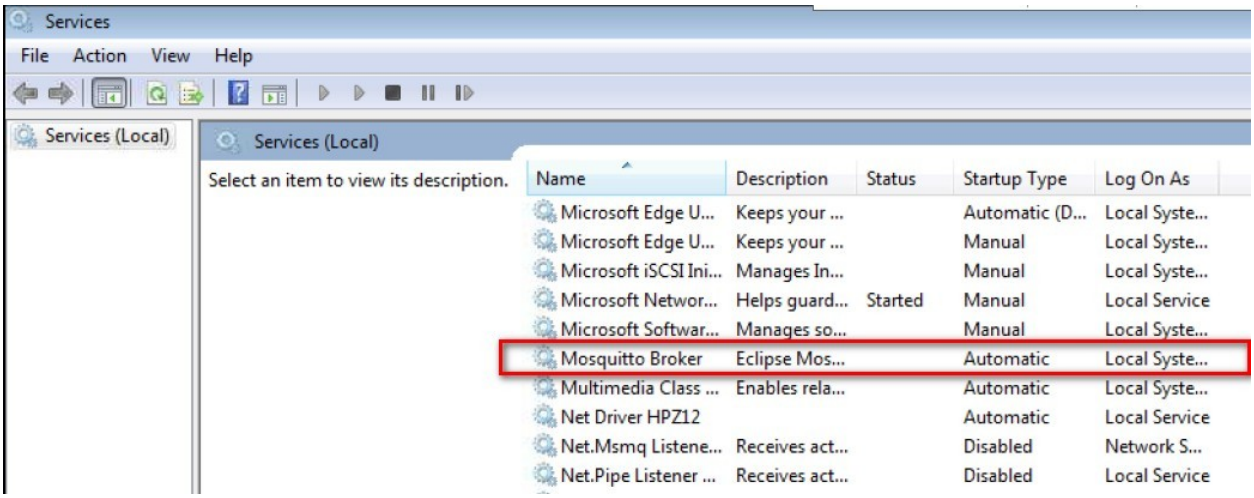


Figure 2-1 Setting Up Mosquitto Broker as a Windows Service

From here you can set the Startup Type to Manual or Automatic. With Automatic, the mosquitto broker service will start up automatically when the computer is booted. There will be no user interface screen like the one shown in the example above in 2.3.

v.002 Update: I moved my system to a Windows 10 computer, and have yet to get the Mosquitto Broker Service to run. When I manually start it in the Services window, it just stops, but I don't know why. I tried to get it to log to a log file, but that didn't work either. I tried to look in the Windows System Error log, but I don't know much about Windows System Error Logging, so I just gave up for the time being.

Instead, I made a small bat file on my desktop and just launch Mosquitto from that. Then the "connect" and "disconnect" messages log to the cmd window.

v.006 Update: After updating mosquitto to latest version (2.0.22), it overwrote my mosquitto.conf file, and I had trouble getting the service to run again. Here are my notes about the settings to make it work:

First, the reference site I finally used to get it working (though, "nsmm" link doesn't work anymore).

https://cedalo.com/blog/how-to-install-mosquitto-mqtt-broker-on-windows/

OK, that page I linked was very helpful. But it still took me an hour of fiddling around to get it working as a service.

The service is installed by the installer, but no mosquitto.conf file is put in the startup parameter for the service. I added this: -c "C:\Program Files\mosquitto\mosquitto.conf" as a startup parameter, and then started the service. (if the service doesn't start, it doesn't remember that parameter, but I hope it remembers it once you get it started successfully).

The bat file I was using to "prove" it was configured correctly first set the directory to C:\Program Files\mosquitto and then started with parameter -c mosquitto.conf, but apparently the service doesn't know where it is, so you need to specify the full path to the mosquitto.conf when starting the service, and if it is in Program Files, you need to use quotes around the file name (as above).

The "as working" (I thought) backup of my saved mosquitto.conf file was apparently not the latest (after I got it working again as a service 2 years ago). So I needed to edit the file and change (what I had listed above) to use fully qualified file names WITHOUT quotes in the conf file. ie log_dest file C:\Program Files\mosquitto\log\mosquitto.log and password_file C:\Program Files\mosquitto\passwdfile.pwd

Make sure your cmd-started instance of mosquitto is not running when you try to start the service again!

Here are the correct active conf file entries (everything else is commented out).

```
listener 1883
log_dest file C:\Program Files\mosquitto\log\mosquitto.log
log_type error
log_type warning
log_type notice
log_type information
connection_messages true
log_timestamp true
log_timestamp_format %Y-%m-%dT%H:%M:%S
allow_anonymous false
password_file C:\Program Files\mosquitto\passwdfile.pwd
```

The only thing I had to change in the service setup was the startup parameter:

-c "C:\Program Files\mosquitto\mosquitto.conf" (and then start the service).

2.6 What's in the Electrodacus MQTT Object

Once you get mosquitto and node-red setup, you can see exactly what's in the MQTT object that is sent from the Electrodacus Wifi module. (note that the exact contents may vary based on what version of the wifi software you are running, and what optional features of the SBMS0 you are using).

object
time: *object*
    year: 0
    month: 1
    day: 1
    hour: 4
    minute: 2
    second: 55
soc: 49
cellsMV: *array[8]*
    0: 2905

1: 2916
2: 0
3: 0
4: 0
5: 0
6: 2924
7: 2935
tempInt: 28.6
tempExt: -45
currentMA: *object*
battery:-1100
pv1:0
pv2:0
extLoad:1040
ad2: 0
ad3: 0
ad4: 0
heat1: 0
heat2: 10774
flags: *object*
OV: false
OVLK: false
UV: false
UVLK: false
IOT: false
COC: false
DOC: false
DSC: false
CELF: false
OPEN: false
LVC: false
ECCF: false
CFET: true
EOC: false
DFET: true
delta:8);

3 Using Node-Red to Monitor Your SBMS0

3.1 What is Node Red and Why Do I Need It

Node-RED is a programming tool for non-programmers to provide a way to do programmable things.

Once installed, you can interactively build a monitoring system for your Electrodacus, something like this:

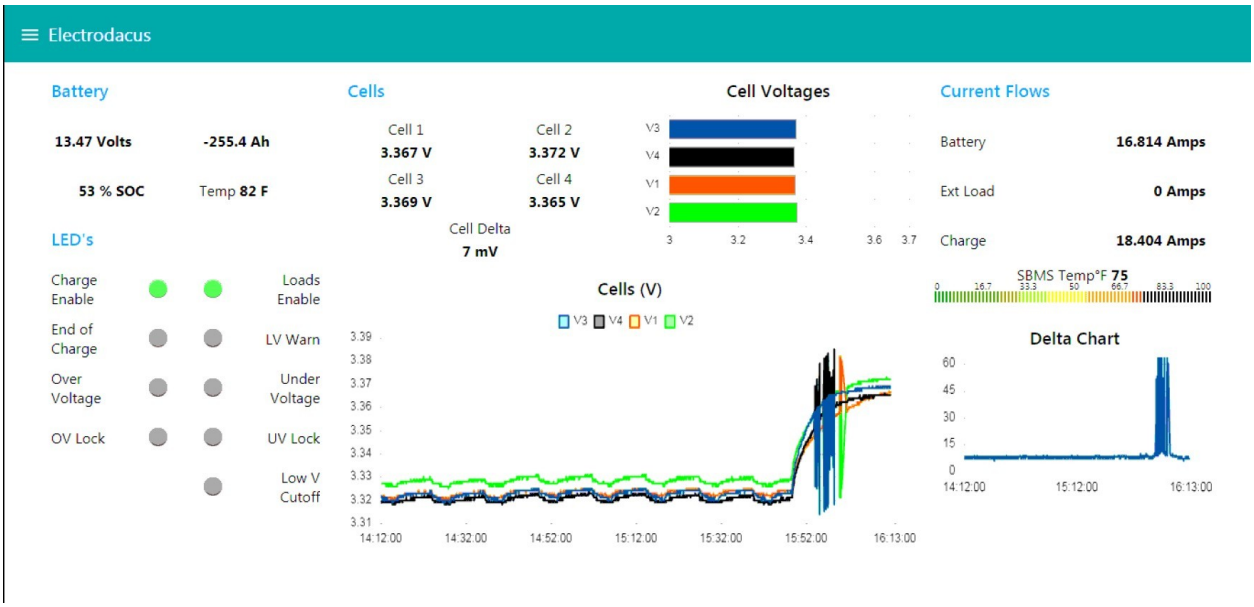


Figure 3-1 A Node-Red Dashboard for Electrodacus Monitoring

The "code" to display your SBMS data as above, in a browser window, looks like this:

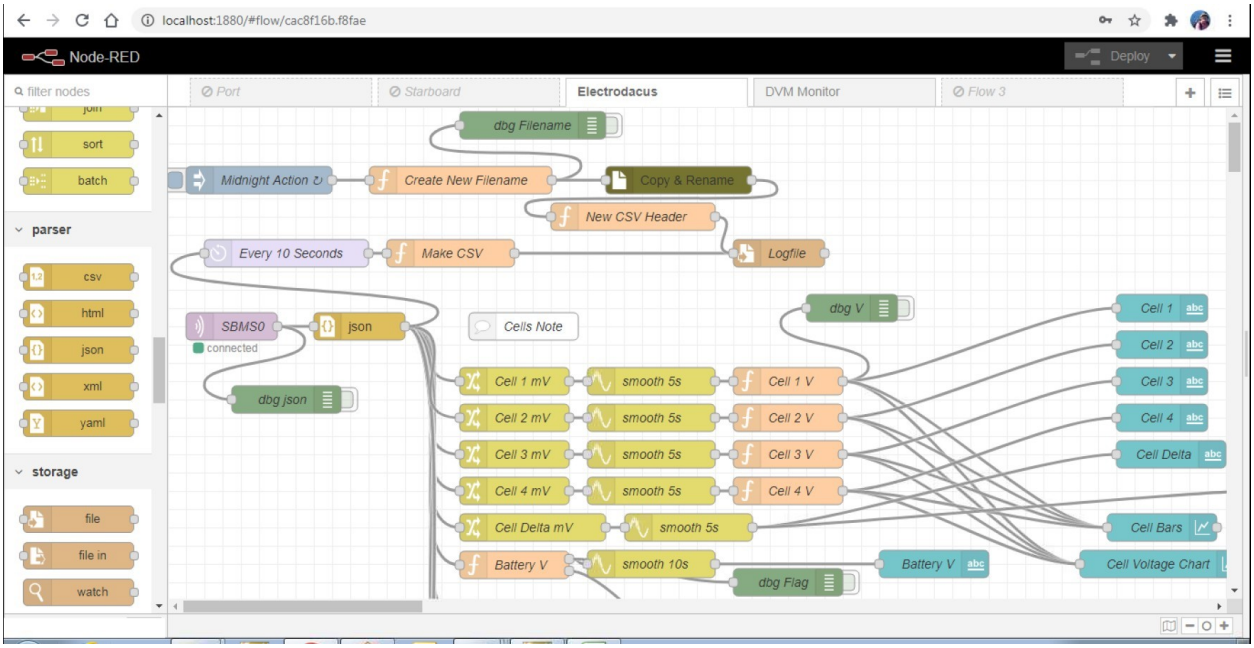


Figure 3-2 Example Node-Red Flow

It's easy to start with someone else's "flows" file and then configure things that are specific to your computer installation. See section on Installing a Trial json File for how to do this, after you get your node-red installation up and running.

3.2 Installing Node-Red on a Windows Computer

Note, on my first attempt I found this Youtube video helpful:

https://www.youtube.com/watch?v=hEpeobDyj8k

3.2.1 Installing Node.js

The first step in installing node-red on Windows is to install the Node.js programming environment.

Go to http://nodejs.org

And download the version that says: Use this one. (currently it is node-v10.15.1-x64.msi, for my 64bit computer). I did download the newer one, but when I installed it, it was missing some modules.

I don't remember being given the option to download a 32-bit one, but if you are not sure whether your computer is 64 bit or 32 bit, usually picking the 32 bit is the safe bet.

Double click the file you downloaded, to install, and accept all the defaults.

June 2022: The current node.js version is node-v16.15.1-x64.msi, and it wanted to install in C:\Program Files\nodejs\

It also wanted to install Chocolatey:

- https://chocolatey.org/packages/chocolatey

- https://chocolatey.org/packages/python

- https://chocolatey.org/packages/visualstudio2019-workload-vctools

It downloaded and installed a bunch of stuff, then seemed to hang. I waited a long time to see if something else would happen.

This is where it stopped:

[4cf8:0012][2022-06-20T15:18:01] Download of 'https://go.microsoft.com/fwlink/?linkid=2066144' succeeded using engine 'WebClient'

This page had news about Visual Studio 2022

3.2.2 Installing node-red

Assuming node.js installed OK, you then use the node.js environment to install node-red

Then open up a Command Prompt. On Windows 10, on the lower left of the screen where the search bar is (ie Cortana's window), where it says "Type here to search", just enter "CMD". You should see an old DOS-style screen.

In the Command Prompt mode, enter these commands, one by one (letting each finish before doing the next one). You MUST be connected to the internet for this step.

```
npm install -g --unsafe-perm node-red
```

(npm install node-red <- I think this is a duplicate of the line above, with some different options)

I got some Deprecation warnings, but everything installed. The end line was a Success with the version of node-red that was installed, and where it was installed.

```
npm install node-red-dashboard
npm install node-red-node-serialport
npm install node-red-contrib-aggregator
```

3.2.3 Adding Additional Items to the node-red Palette

If you share or get a shared json file, it is likely that there will be new node-red items used that you don't have in your node-red install.

For example, a flow I got from someone else for the Electrodacus used a new type of bar graph, so when I imported that flow, I got an error "unknown: ui_level"

So I googled "node-red ui-level", and found this location:

https://flows.nodered.org/node/node-red-contrib-ui-level

Here are the extra palette items I have installed on my node-red system. (if you use my flows, you will need to install most of them).

node-red-contrib-aggregator	not using this at the moment
node-red-contrib-fs	File manipulation
node-red-contrib-fs-ops	More file manipulation
node-red-contrib-modbus	Only For the MPPT/solar stuff
node-red-contrib-modbus-tcp-ip	Only For the MPPT/solar stuff
node-red-contrib-ui-led	For the LED's
node-red-contrib-ui-level	Only needed for fancy SBMS Temp display
node-red-dashboard	Lets you organize your user interface into groups and control ordering
node-red-node-random	Not used?
node-red-node-rbe	Report by Exception Handling
node-red-node-serialport	For the DVM interface
node-red-node-smooth	Smoothing of input data
node-red-node-tail	A file thing, may not be used
utf-8-validate	Required by another node

3.2.3.1 Install Using "Manage Palette" Menu Item

Once you have Node Red installed and running, the proper way to install new palette items is to, from the Node Red ui, click on the menu button on the far right of the Node Red toolbar, and select "Manage Palette". A window will pop up with 2 tabs. One shows the node red nodes you already have installed (and versions), and the other allows you to search for and install new node red notes.

This also allows you to see which nodes require updating, and you can do the update with one click.

3.2.3.2 Install Manually

I don't know why you would do it this way, unless maybe you are installing a bunch of stuff from a batch file. I didn't know any better when I first wrote this document. The first method is preferred.

Open up a cmd window with Admin privileges, set my directory to

C:\Users\<user>\.node-red directory, and ran this command:

```
c:\Users\sherr\.node-red>npm install node-red-contrib-ui-level
```

The response was:

```
npm WARN node-red-project@0.0.1 No repository field.
npm WARN node-red-project@0.0.1 No license field.
+ node-red-contrib-ui-level@0.1.40
added 1 package from 1 contributor and audited 5 packages in 7.234s
found 0 vulnerabilities
c:\Users\sherr\.node-red>
```

You must then stop and restart node-red (the cmd line interface) to get the newly added node activated on your node-red palette.

If you encounter an error like this:

```
npm WARN ws@7.4.4 requires a peer of utf-8-validate@^5.0.2 but none is
installed
```

It means that the node you just installed is dependent on another node that you don't have installed (or that the right version is not installed). So then you need to npm install "utf-8-validate".

3.2.4 Troubleshooting Installation Issues

Sorry, so far, my installation seems to have gone flawlessly, so I don't have any troubleshooting tips.

The "node" program installed here:

```
C:\Program Files\nodejs
```

The node.js data directory, by default, on Windows 10, installs here:

```
C:\Users\sherr\AppData\Roaming\npm
```

The node-red data directory, by default, on Windows 10, installs here:

```
C:\Users\<username>\.node-red
```

However, I don't see any log files in here that would be of use in determining the source of problems.

3.3 Upgrading Node.js and node-red

I just downloaded the latest node.js install, ran it and then tried to run my existing node-red. It gave an error that my node-red version was too old. So I ran through the npm install steps in 3.2.2 above, and it seems to have updated everything.

It is also possible, I just discovered, to use the Node Red Manage Palette menu item to review your node red versions and update it all from there. THIS IS THE PREFERRED METHOD!

Log of installation from the cmd window…

```
C:\Users\sherr>npm install -g --unsafe-perm node-red
npm WARN deprecated nodemailer@1.11.0: All versions below 4.0.1 of Nodemailer
are deprecated. See https://nodemailer.com/status/
C:\Users\sherr\AppData\Roaming\npm\node-red ->
C:\Users\sherr\AppData\Roaming\npm\node_modules\node-red\red.js
C:\Users\sherr\AppData\Roaming\npm\node-red-pi ->
C:\Users\sherr\AppData\Roaming\npm\node_modules\node-red\bin\node-red-pi
> bcrypt@2.0.1 install C:\Users\sherr\AppData\Roaming\npm\node_modules\node-
red\node_modules\bcrypt
> node-pre-gyp install --fallback-to-build
```

```
[bcrypt] Success: "C:\Users\sherr\AppData\Roaming\npm\node_modules\node-
red\node_modules\bcrypt\lib\binding\bcrypt_lib.node" is installed via remote
+ node-red@0.19.5
added 122 packages from 40 contributors, removed 92 packages and updated 114
packages in 80.799s

C:\Users\sherr>npm install node-red-dashboard
npm WARN saveError ENOENT: no such file or directory, open
'C:\Users\sherr\package.json'
npm WARN enoent ENOENT: no such file or directory, open
'C:\Users\sherr\package.json'
npm WARN sherr No description
npm WARN sherr No repository field.
npm WARN sherr No README data
npm WARN sherr No license field.

+ node-red-dashboard@2.13.2
added 30 packages from 23 contributors, updated 1 package and audited 961
packages in 12.002s
found 25 vulnerabilities (21 low, 4 high)
  run `npm audit fix` to fix them, or `npm audit` for details

C:\Users\sherr>npm audit fix
npm ERR! code EAUDITNOPJSON
npm ERR! audit No package.json found: Cannot audit a project without a
package.json

npm ERR! A complete log of this run can be found in:
npm ERR!     C:\Users\sherr\AppData\Roaming\npm-cache\_logs\2019-02-
23T08_03_04_261Z-debug.log

C:\Users\sherr>npm install node-red-node-serialport

> @serialport/bindings@2.0.7 install
C:\Users\sherr\node_modules\@serialport\bindings
> prebuild-install --tag-prefix @serialport/bindings@ || node-gyp rebuild

npm WARN saveError ENOENT: no such file or directory, open
'C:\Users\sherr\package.json'
npm WARN enoent ENOENT: no such file or directory, open
'C:\Users\sherr\package.json'
npm WARN sherr No description
npm WARN sherr No repository field.
npm WARN sherr No README data
npm WARN sherr No license field.

+ node-red-node-serialport@0.7.0
added 35 packages from 32 contributors, updated 1 package and audited 994
packages in 18.448s
found 24 vulnerabilities (20 low, 4 high)
  run `npm audit fix` to fix them, or `npm audit` for details

C:\Users\sherr>npm install node-red-contrib-aggregator
npm WARN saveError ENOENT: no such file or directory, open
'C:\Users\sherr\package.json'
npm WARN enoent ENOENT: no such file or directory, open
'C:\Users\sherr\package.json'
```

```
npm WARN sherr No description
npm WARN sherr No repository field.
npm WARN sherr No README data
npm WARN sherr No license field.

+ node-red-contrib-aggregator@1.4.0
updated 1 package and audited 1137 packages in 6.54s
found 24 vulnerabilities (20 low, 4 high)
  run `npm audit fix` to fix them, or `npm audit` for details
```

I don't know what all that means, but it did it all automatically and my new version of node-red runs fine.

3.4 Running Node-Red

3.4.1 Starting up the Node-Red server

Assuming everything installed OK… I got some warnings, but never any errors, and it seems to have installed OK.

To start node-red, in the CMD: window, (or the Windows Search Bar on W10) type in "node-red"

```
C:\Users\sherr>node-red
```

Here is approximately the response you should get

```
22 Apr 14:21:03 - [info]
Welcome to Node-RED
===================
22 Apr 14:21:03 - [info] Node-RED version: v1.3.2
22 Apr 14:21:03 - [info] Node.js  version: v14.16.1
22 Apr 14:21:03 - [info] Windows_NT 10.0.19041 x64 LE
<sometimes long delay, as much as a minute or two>
22 Apr 14:21:05 - [info] Loading palette nodes
22 Apr 14:21:06 - [info] Dashboard version 2.28.2 started at /ui
22 Apr 14:21:07 - [info] Settings file  : C:\Users\soggy\.node-
red\settings.js
22 Apr 14:21:07 - [info] Context store  : 'default' [module=memory]
22 Apr 14:21:07 - [info] User directory : \Users\soggy\.node-red
22 Apr 14:21:07 - [warn] Projects disabled :
editorTheme.projects.enabled=false
22 Apr 14:21:07 - [info] Flows file     : \Users\soggy\.node-
red\flows_SoggyPaws-Nav.json
22 Apr 14:21:07 - [info] Server now running at http://127.0.0.1:1880/
22 Apr 14:21:07 - [info] Starting flows
22 Apr 14:21:07 - [info] [function:Amp Hours] Set maxAh to Zero
22 Apr 14:21:07 - [info] Started flows
22 Apr 14:21:07 - [info] [mqtt-broker:MQTT Broker] Connected to broker:
nodered@mqtt://localhost:1883
```

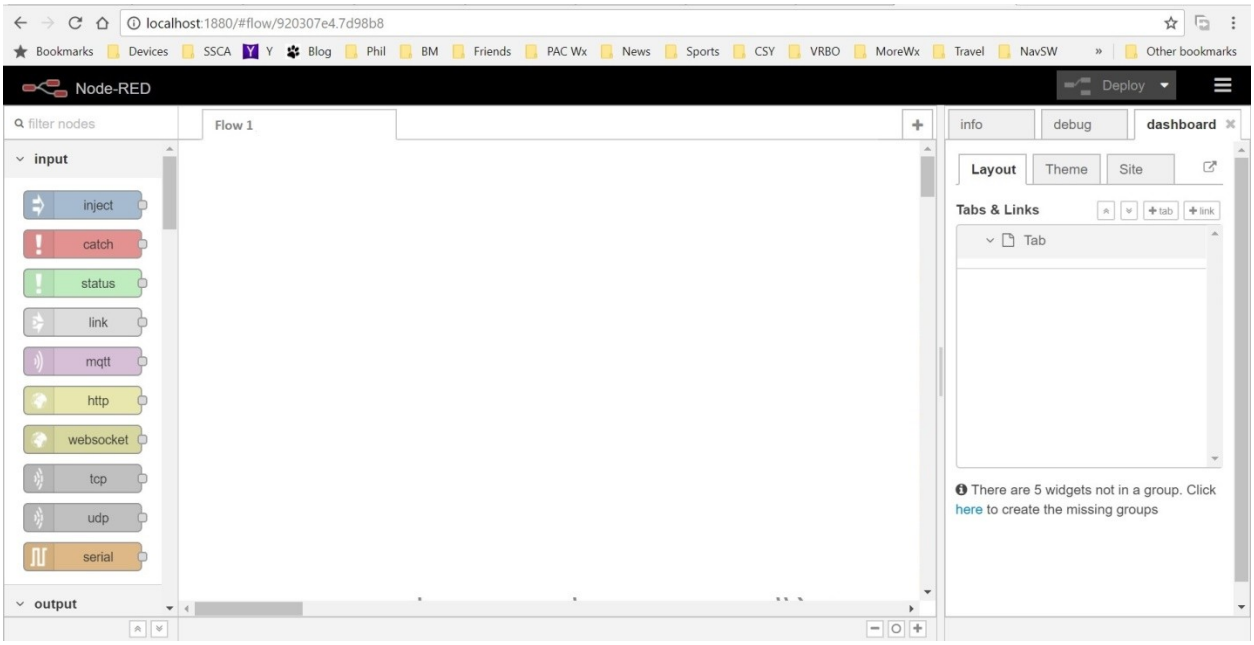
At some point in this sequence, probably the first time I ran node-red, Windows threw up a Security window, asking if I wanted node-red to run and whether to let it run in the local network, or on the public network. For now I said "local only".

3.4.2 In the Browser Window

Now go to your browser and open up a new window/tab and type in the browser bar:

```
localhost:1880
```

Voila, you will/should see the “node-red” environment in your browser window. It should look something like this on a new install:



3.4.3 Installing a Trial json file

My current Electrodacus flows file is included at the end of this document. Or you may see an “xxxx.json” file posted on the Electrodacus forum (search for node-red or nodered).

There are 3 ways to install (“import” in node-red terms) someone else’s json file. A json file is just a text configuration file that tells node-red what to do.

1. Import using the Menu (the best way, for a full flows file)

   Locate the node-red menu bar. It should be a typical menu indicator (3 horizontal bars) in the upper right corner of the window. Click on that and find the Import menu item. Apparently if you have copied a json file from somewhere else (and it’s just sitting in your “clipboard”) you can Import / From Clipboard. Or you can import the file you downloaded.

2. Copy and Paste

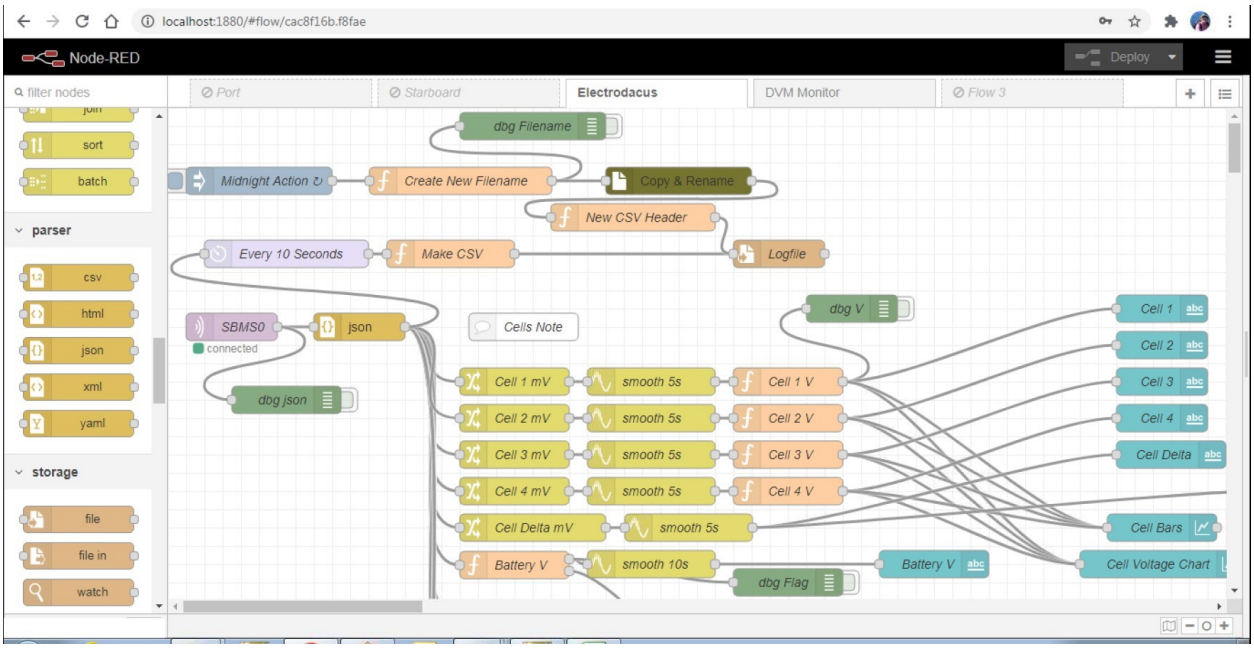
   Copy (ctrl-c) the contents of a node-red flows file (you can try with the one in the appendix A or B).

   Click in the middle of the blank “Flow1” page above and paste (ctrl-v), and voila, you should see all the nodes.

3. Drag and Drop.

   Get the File Explorer window where you downloaded (or saved from cut and paste into notepad) the json file laid on top of the node-red window in your browser, and drag the json file to the middle of the Node Red window.

After importing, your node-red browser window should look something like this:



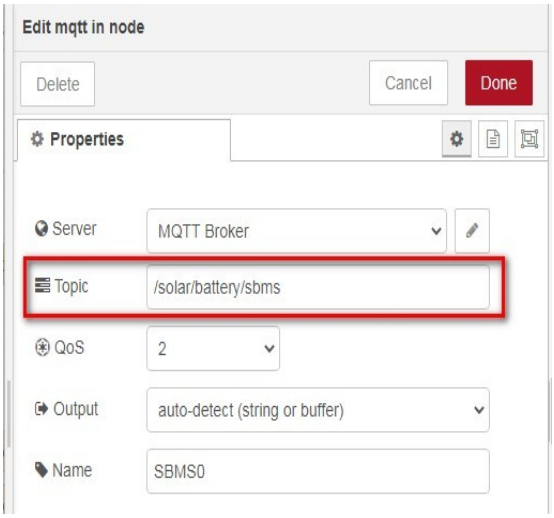
Terminology: A set of nodes that work together is called a flow (one flow per tab on my screen above). Note that the little items on the screen connected by lines are called “nodes”. The nodes with red dots on them have a configuration problem.

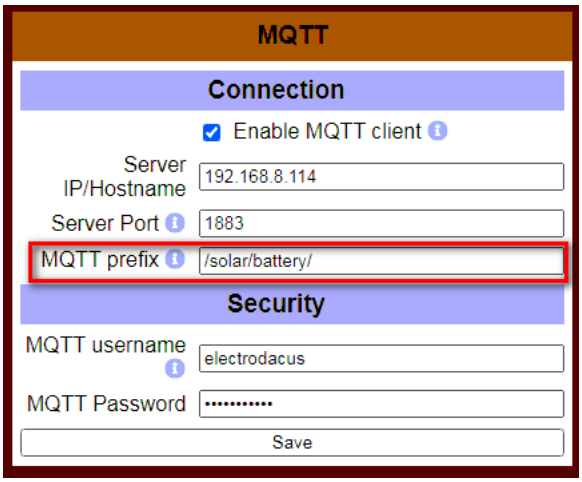
3.4.4 Configuring Your Electrodacus flow

There are several things you need to do first to configure the sample file for your own environment:

1. Double click on the Left most node “SBMS0”, which is the MQTT node, and configure it for your MQTT setup.

   Should match your Electrodacus wifi settings





2. Double click on any node that does file operations (Create New Filename, Logfile, Copy & Rename) node, and check that the file path is appropriate for your system. The daily log currently goes to a text log file on the desktop. Once a day just after midnight, the just completed days’ log gets copied and renamed to a date-stamped filename, and a header line is

logged as the first line of the Electrodacus.log file. The CSV files can be opened in Excel and you can use Excel graphing tools to make graphs and analyze your data.

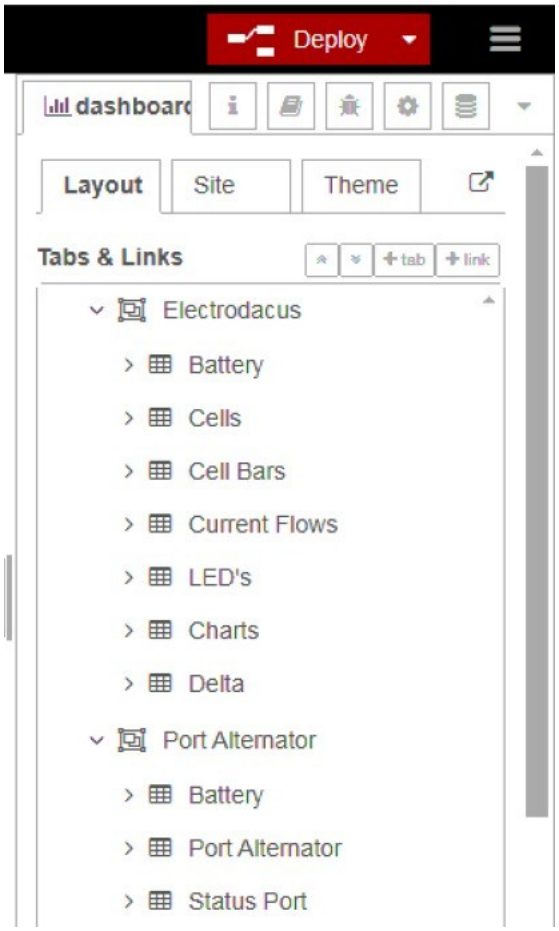
3. If you have 8 cells instead of 4, you'll need to add 4 more cell lines. Note that in a 4 cell configuration, cells 3 and 4 are actually placed as 7 and 8 in the incoming data stream.

3.4.5 Deploying

If you make any changes to your node-red flow, every node that has changed has a little blueish dot in the upper right corner, indicating it is not deployed.

Clicking Deploy deploys your design to an active state (whatever that means).

Then refresh the browser tab with your UI in it.



3.4.6 Displaying the User Interface

Create a new tab in your browser and put this in it: localhost:1880/ui.

I also did this to get the tab automatically created. In the panel on the right side of the node-red screen, which looks like this:

Click on the dashboard tab, and then click the little icon to the right of the Layout / Theme / Site tabs that looks like this:

This should open up a new tab on your browser that has your User Interface (ui) in it.

3.4.7 Saving Your Work

I am not exactly sure when / if your changes to the configuration get saved. At "Deploy", I think.

With my node-red user directory set to C:**\Users\sherr\.node-red**

The "flows" file was found in there.

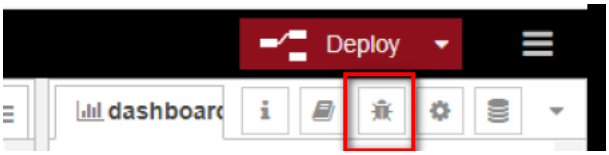
My computer name is SoggyPaws-PC and the flows file is automatically named:

`flows_SoggyPaws-PC.json`

3.4.8 Debugging

Trying to get an initial installation working can be frustrating.

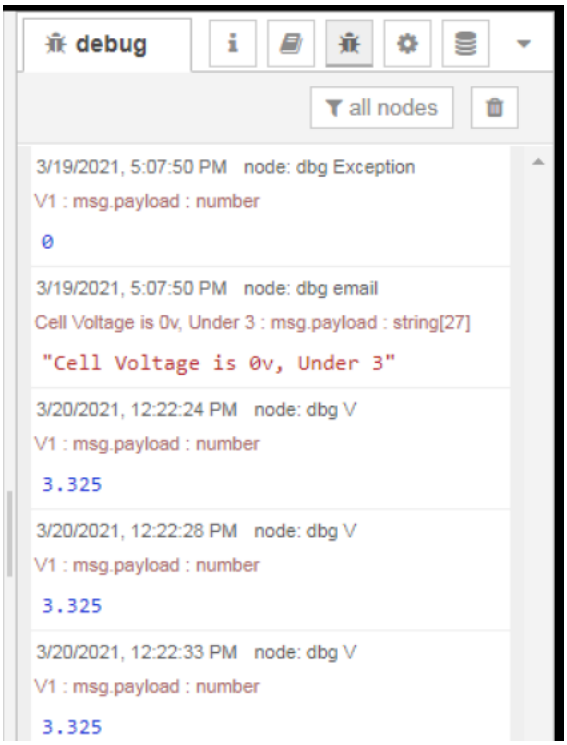
Node-red has built-in debugging capabilities that let you see the output of each node. The green nodes are debug nodes. They will display what gets sent to them by the output of the previous node on the debug panel on the right side of the screen.



You can turn on and off debug output to the debug panel by clicking in the square at the right end of the debug node. Filled in green means “debug on” and white means “debug off”.

Here is sample debug in the debug panel for Cell Voltage 1 output.

Right, a sample of the debug output on node-red



3.5 Shutting Down Node Red

To stop input, in the CMD window, CTRL-C, or close the CMD window.

Be sure to deploy any changes you want to save, before shutting down the node-red in the cmd line.

In your node-red browser tabs, it will say then “can’t find the node-red server, trying to reconnect”. Close the tabs.

3.6 Sharing Your Node-Red flow

Once you’ve got something useful to share, it’s easy to export a flow (or even just a part of a flow) that others can import into their node-red environment to give them a head start on creating their own.

This is also useful if you are asking for help on the node-red forum. They will want to see your actual flows file.

First, go to the tab (flow) you want to export. If you only want to export a subset of your flow, select the nodes you want to export. Then click on the menu button in the upper right of the node-red screen and select Export.

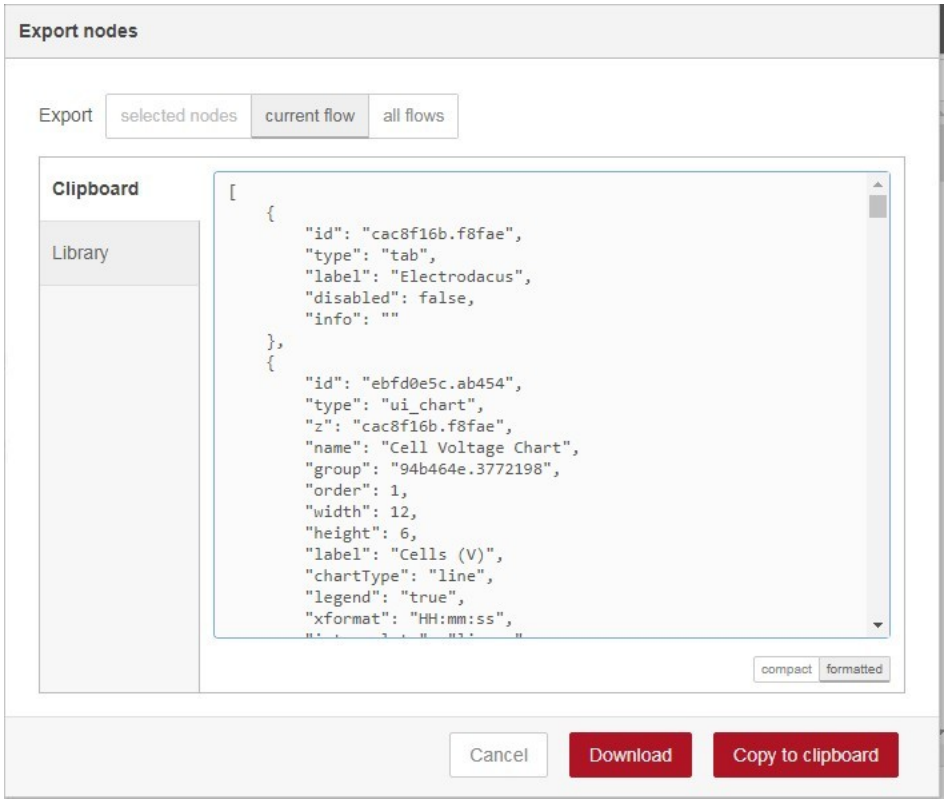


Figure 3-3 Exporting Your Flow to Share (or Backup)

Choose at the top of the window whether you want to export all flows (all the tabs in your Node Red UI), or just the current flow (tab), or just the nodes you have selected.

Clicking Download will open a file download window so you can save your flows file to a file. Clicking "Copy to Clipboard" makes it easy to paste into an email, for example.

3.7 Backing up Everything

As always, you should regularly run backups on your computer, just in case the computer or hard drive crashes. If you are like me, and only back up specific directories, here are the ones you should probably add to your backup list:

C:\Users\<username>\.node-red

C:\Program Files\mosquitto

4 Node Red on Raspberry Pi

4.1 Installing Node-Red on Openplotter/SignalK

Here are the basics of getting node-red installed on a Raspberry Pi that is already running "openplotter" (OpenPlotter is a Raspeberry Pi distro that is configured for cruising boats and includes modules for navigation, and Signal-K, etc).

https://openmarine.net/openplotter

SignalK is a component of OpenPlotter. https://signalk.org/

1. Node Red on the RPi is a "plugin" to Signal K. (it can be installed separately, but if you are running OpenPlotter they recommend you install it as a Signal K plugin)
2. Start up Signal K, with an internet connection.
3. Create an Admin log-in for Signal K, if you have not already. Log in as Admin
4. Click on Webapps, and if Node Red is not there as one of the Webapps…
5. Click on Appstore, and Available… wait (it's contacting the internet to see what's there)
6. Eventually there should be a long list of available "apps", Node-Red should be one of them
7. There's a little cloud icon on the right end of the bar for each app. Click on the cloud icon to install Node Red. This took awhile (15 minutes).
8. I think you need to reboot before Node Red shows up as a Webapp under SignalK
9. Once you see Node-Red as a webapp under SignalK, click on it, and Walla.

4.2 Installing Node-Red Without OpenPlotter/SignalK

If you are not using OpenPlotter/SignalK, the Windows node-red install procedure provided above will likely be close to what you need to do for a Raspberry Pi install.

Here are two other resources that may be helpful:

https://projects.raspberrypi.org/en/projects/getting-started-with-node-red

https://nodered.org/docs/getting-started/raspberrypi

4.3 Installing Other Components

The node-red install does not include the Dashboard needed for gauges and stuff.

To install the dashboard…

1. Click on the hamburger symbol on the right side of the node-red toolbar. The dropdown should include "Manage Palette". Click on that.
2. There are two tabs there. Click on the Install tab and in the search box, put "dashboard". That should bring up node-red-dashboard as an installable module. Click on the install button.

Installing other components on the palette works the same.

4.4 Using the Windows Version Node-Red Json Files

I just copied the json file from C:\Users\<user>\.node-red to RPi, renamed it to something more generic, and then dragged and dropped it onto the node-red "workspace".

I think I had to install a few more nodes, and do file path changes appropriate to the differences between Windows and Raspberry Pi.

This ended up bringing over 85% of what I had running in the Windows version. What didn't get brought over properly were the tab and group assignments. After manually fixing this (by going in to all ui nodes and re-assigning the "group" value, I discovered the original groups and tabs (in the Unused Nodes) view of the dashboard.

Update: If I had used the proper "export" from my Windows version and "import" to my Raspeberry Pi, it may have worked better. See Sharing Your Node-Red flow for how to export and import.

Once imported, there were at least 2 changes I needed to make:

The file path for files:

Windows	Raspberry Pi
C:\Users\<user>\Desktop\Node-Red\	/home/pi/Documents/Node-Red/

Serial Port configuration (if you are using a serial port for any reason)

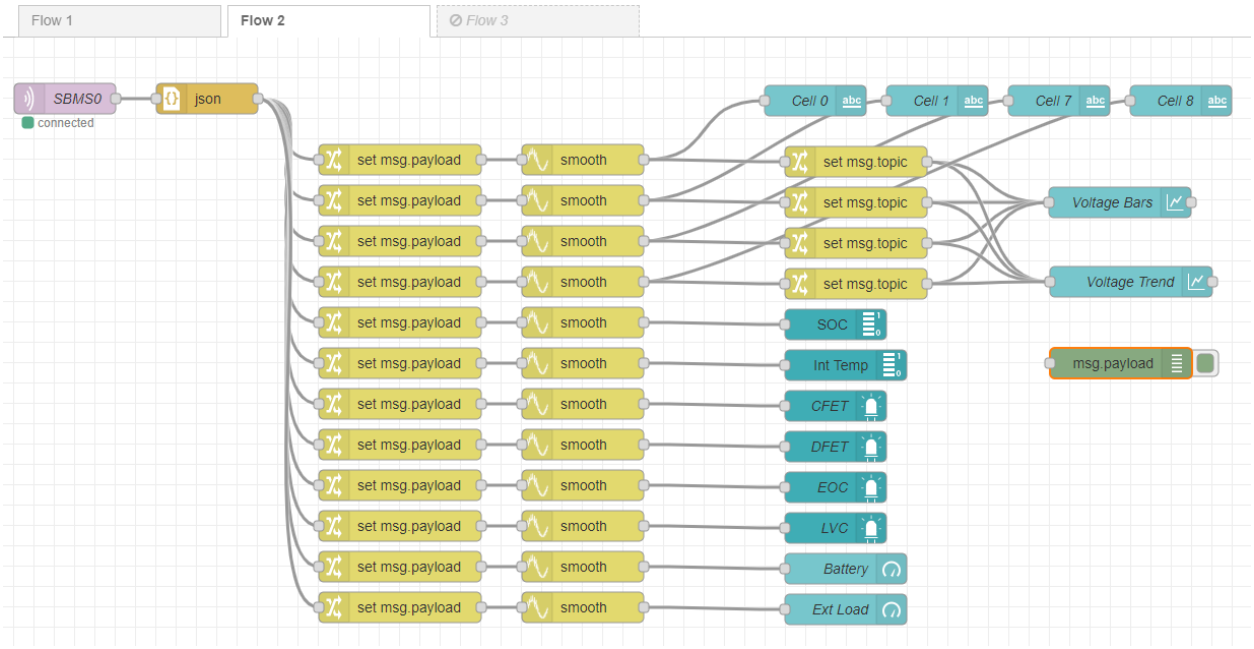
Windows	Raspberry Pi
Com6	/dev/ttyUSB0

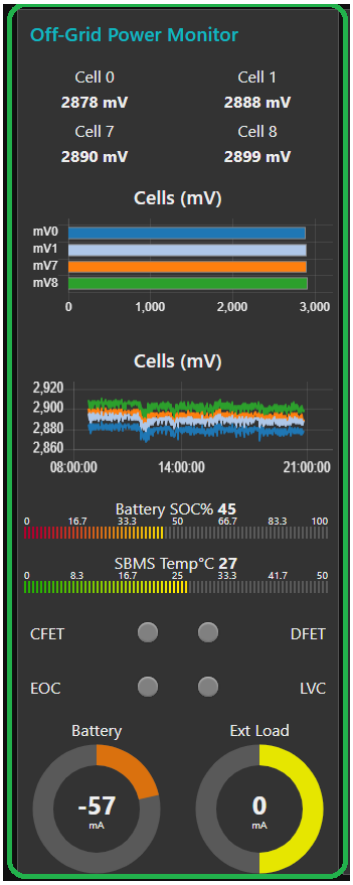
And the mosquitto configuration may need to change as well.

5 Appendix A – The Node Red json file I started with

Thanks to Kohala Jim for sharing his flows file! It was invaluable in getting me started.

Here is what the flows "editor" looks like in node-red, when you've imported the flows file (below).





And here is the resulting dashboard after the flows have been "deployed"

6 Appendix B – My Current JSON file

Here is my current dashboard (there's a bigger version of this on our website):

Update 2025: We changed from a Morningstar MPTT (Modbus) to 2 Victron MPPT's using Victron Connect USB cables. So the latest json files include Victron logic and rearranges things. I have left the Morningstar MPPT version on my website. This picture is from the old version.

Currently I am running on Windows 11 Pro with MQTT v2.0.22 and Node-RED version: v4.1.5 and Node.js version: v25.3.0. I have had no issues upgrading to the latest versions, but I am still using the old v1 Dashboard.

Here is a link to my current json files:

https://svsoggypaws.com/files/lifepo4/Soggy%20Paws%20Node%20Red%20Flows%20Electrodacus%20BMS%20and%20Victron%20MPPT.zip

http://svsoggypaws.com/files/lifepo4/Node_Red_Flows_Windows.zip (pretty old)

http://svsoggypaws.com/files/lifepo4/Node_Red_Flows_Pi.zip (pretty old)

Currently running on Windows 11 with MQTT v2.0.22 and Node-RED version: v4.1.3 and Node.js version: v25.3.0. I haven't noticed any problems when upgrading, but I suggest you make backup copies of the folders (once you get everything set up to your satisfaction), found here:

C:\Program Files\mosquitto and

C:\Users\<username>\.node-red

The json files are just text files. It looks like greek, but if you import it into a node-red flow, it will start to make a little more sense.

You WILL likely have to add nodes into your node-red configuration that are not already configured in your node-red setup. This is easy. The import process will tell you what nodes you are missing, then add them one by one until the import doesn't complain. See Adding Additional Nodes.

You will also likely need to change some configuration inside a few nodes (mainly file path settings, in the logging, and the Mosquitto broker settings).

IMPORTANT: Also, my latest version uses a node.js function that is external to node-red. You need to locate your "settings.js" file (normally located in C:\Users\<user>\.node-red) and find the "functionGlobalContext" line(s) and add this:

```
functionGlobalContext: { // enables and pre-populates the context.global variable
    // -- Pass in Libraries for convenience in function nodes -- //
    'fs'   : require('fs')
},
```

6.1 Components of our system

The Electrodacus is the main piece, but I have included the other flows to monitor the other parts of our charging system. It has gotten fairly complicated as I've added pieces. Hopefully not too complicated.

We have 2 alternators, each with a WS500 smart regulator that sends a text string through a serial port with information. They are not on all the time. The json files for this monitoring is fairly crude, as it was the first Node Red project I did. Though I have updated it recently to log less data.

~~We have a Morningstar TS-60 MPPT solar controller that also sends data through a serial port using serial Modbus protocol. This is implemented in a separate node-red flow.~~

Mid 2025: We now have 2 Victron 100/50 Smart Solar MPPT controllers that are connected through Victron USB adapters to 2 serial ports. This is implemented in a separate node-red flow. Changing settings in the Victrons is done using Bluetooth using the Victron mobile app on Android. You can also use the USB cable and the Victron Windows app, but this requires temporarily stopping the node-red flow.

I have the Electrodacus as my main page, but the WS500 alternators and the solar controllers on a separate monitoring page. Those separate pages feed a little bit of detail into the Electrodacus page, so I can see most of what is going on from the Electrodacus page, but if I want details, switch to the detail page for that device. These are not necessary, but it's useful to me to see the state of the other controllers (MPPT, Bulk, Absorb, Float, etc).

I also have a scary looking "hidden" component that puts the date and time in the upper right corner of the window. This is useful, as my dashboard doesn't update if no one is looking at the screen, and I can tell from the time shown whether the info is recent or not (compare with current computer time).

I included a text file on the things you need to change either to move to a Raspberry Pi, or to just change my file directory listings to yours. I was bouncing back and forth between Windows and the Pi often enough that it was easiest to do a search and replace via text editor vs going and finding the components that needed to be changed in the Node Red editor.

If you have problems/questions about getting set up with Electrodacus, I'd prefer you post your question on the Electrodacus Google Group, so everyone benefits from the exchange.

I am constantly tweaking the node-red setup. A few things I recently added:

- Tracking daily highs and lows (volts, amps, amp hours, etc)
- Daily log file copied as a CSV file at midnight every night
- Daily summary stats logged in a DailyStats.log file
- Adding the Port and Starboard Alternator "state" to the Electrodacus dashboard.
- Log any "exception" LED values in the log

- Get something working that will poll our Morningstar Tristar MPPT Solar Controller for its info, and add it's "state" to the Electrodacus dashboard.

Things I'm still working on adding in the future:

- Email notification at end of charge with the logged info from the last 5-10 minutes of charging (only activated when we are off the boat)
- Email notification at high and low voltage alarms (I have played with this in the DVM monitor, but need to incorporate it into the main Electrodacus so it can be enabled when we are off the boat)

7 Appendix C - Logging Voltage from a DVM with Serial Output

Another node-red (non-MQTT) application we found useful while playing with our new LifePO4 batteries is to monitor charging and discharging on a single cell (or several cells in parallel). This is useful for top balancing batteries. For this we use an inexpensive but quite accurate Digital Voltmeter that has the ability to output whatever it is set on (in this case, Volts) on a serial port.

While there are Windows programs that are set up to do this, we preferred to do it via Node Red (partially as a learning tool, partially so we could add alarm limits that would email us as cells were approaching a high or low limit).

7.1 Hardware Setup

We are using a Uni-T 61E Digital Volt Meter purchased off Amazon for about $60. Here is a link. Make sure you buy one with the RS-232 Serial Port connection.

We compared the voltage readings from this DVM to a very expensive DVM that a friend had, and ours was only 2mV off in the voltage range of 3-14 volts DC. The “B” through “E” models of this DVM comes with an isolated Serial Port output.

Unless you have a real 9-pin serial port on your computer, you will also need a serial-usb adapter.

7.2 User Dashboard Screen

This doesn’t look very exciting because currently my DVM is not on. But it shows the ability to change settings on the fly (for warning levels) on the left side, and get a snapshot of the last X hours of charging/discharging.

The idea being to generate an email as the battery is approaching either a high or low level, in time for us to get to where the batteries are physically located to shut down the load or the charge.

It also logs data to a DVM log file in CSV format.

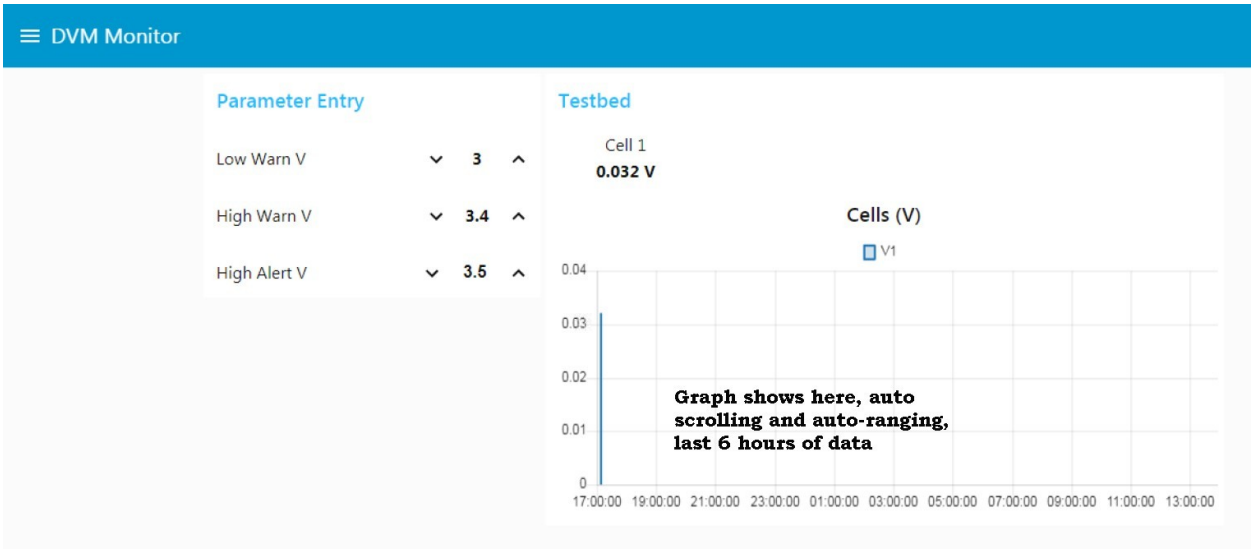


Figure 7-1 DVM Monitoring Screen

7.3 DVM Flow

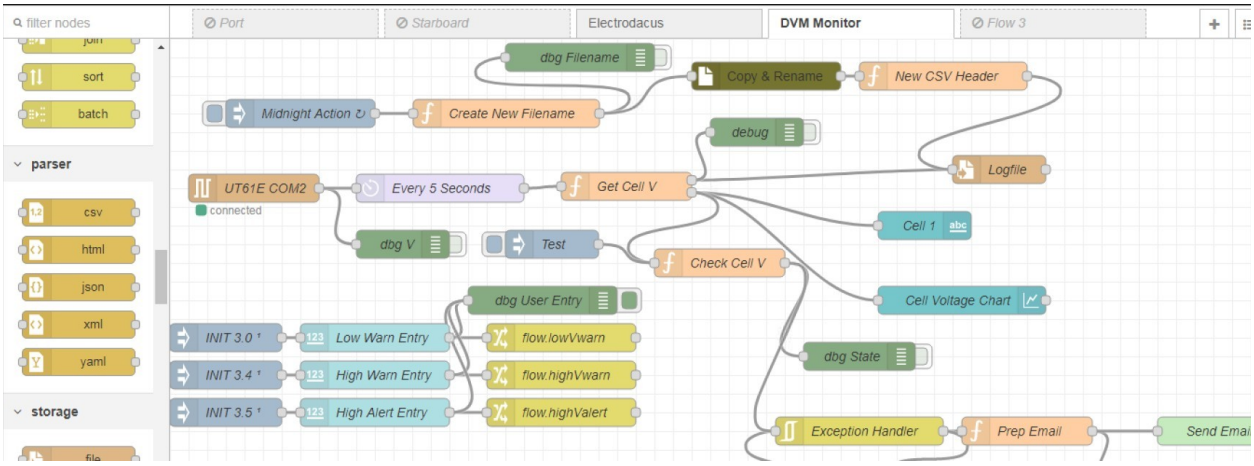


Figure 7-2 DVM File (top half)

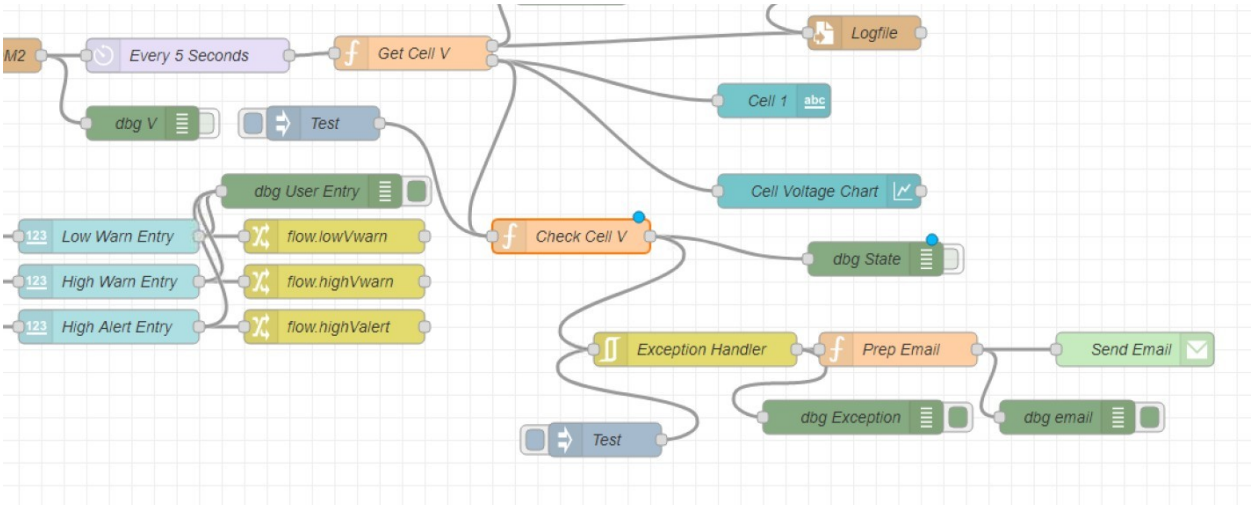


Figure 7-3 DVM Flow (bottom right)

7.4 DVM json File

The DVM json file is included in the Windows version of the json file linked above.