# Bohunt School *(Wokingham)*

# Solar-Powered Weather Station

*Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

***I'm one of the students who has been attending the (after-school) IoT Computer Club at Bohunt Wokingham academy here in the UK.***

***This is a write-up of my personal project to design and build a solar-powered weather station.***

## *Objectives*

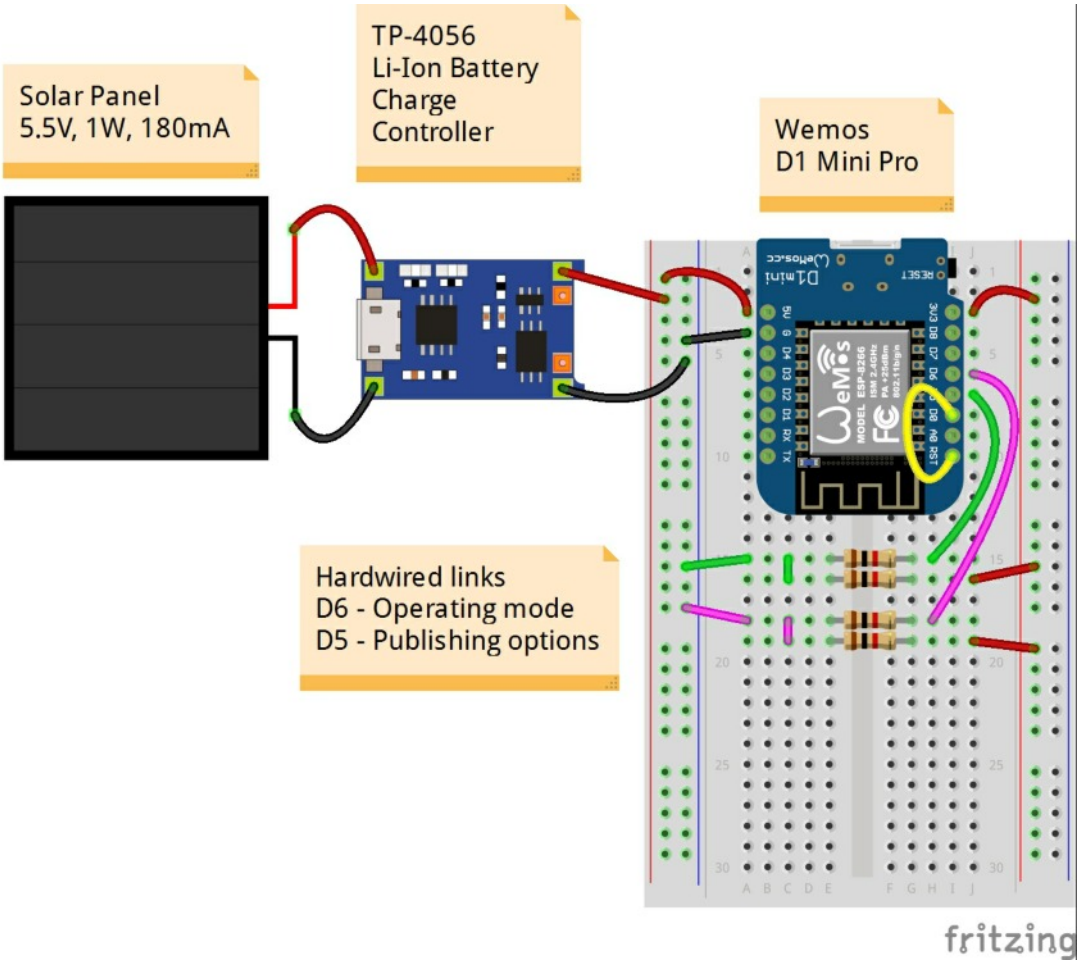
**My overall objective is to build a solar-powered weather station that will be located in one of the school's gardens to measure temperature, humidity and air pressure. The data will be sent via WiFi to the 'Cloud' and by writing some programs the information should be made available locally or remotely.**

**The task I'm going to describe in this particular document is:**

- ***Put the Wemos D1 Mini into 'deepsleep' to conserve the battery's charge***

## *Hardware layout and connections*

**Here's a pictorial view of the interconnections for the main hardware items.**




**Note: I couldn't find a 'body-shape' for a Wemos D1 Mini PRO (in Fritzing) so I had to use one for a Wemos D1 Mini instead.**

*Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

**I've also omitted the connections to the Li-Ion battery in order to keep the diagram simple and uncluttered.**

**So the outputs from the solar-panel are connected to the TP-4056 (Li-Ion battery charging controller) and the outputs from the TP-4056 are wired to the 5V and G pins on the Wemos.**

**There is a wire-link between D0 and RST, which is used to bring the Wemos out of 'deepsleep'. If you forget to make this connection then things won't work.**

**There are a couple of hard-wired connections for D6 and D5. These are formed by the two sets of 1K resistors that go from +3V3 to D6 and D5 respectively.**

**The diagram (shown on the previous page) have links (shown in green and purple) that take the mid-point of the resistor chain to ground. This means that D6 and D5 will be at 0V or logic '0'.**

**If you remove the links (remove the short green and/or purple wires) then D6 and/or D5 will rise towards +3V3 or logic '1'.**

**The logic value on D6 and D5 is sensed by the 'rule-set' in the Wemos firmware and used to carry out specific tasks (described below).**

## *Task settings (D6)*

**If D6 is a logic '1' - Node-RED controls the Wemos going into 'deepsleep'.**

**If D6 is a logic '0' - Wemos controls the Wemos going into 'deepsleep'**

## *Task settings (D5)*

**If D5 is a logic '1' - Publish readings to Node-RED**

**If D5 is a logic '0' - Publish readings to Node-RED and to ThingSpeak**

**The D6 settings are really useful as it means Node-RED can inhibit the normal *'sleep*, *wake-up*, *send information*, *go back to sleep'* cycle of the Wemos, so you have a chance to log-in to the web maintenance panel and make changes.**

## *Device Driver settings (Wemos D1 Mini)*

**On the next page is a screen-shot of the Devices tab available within the ESP Easy firmware.**

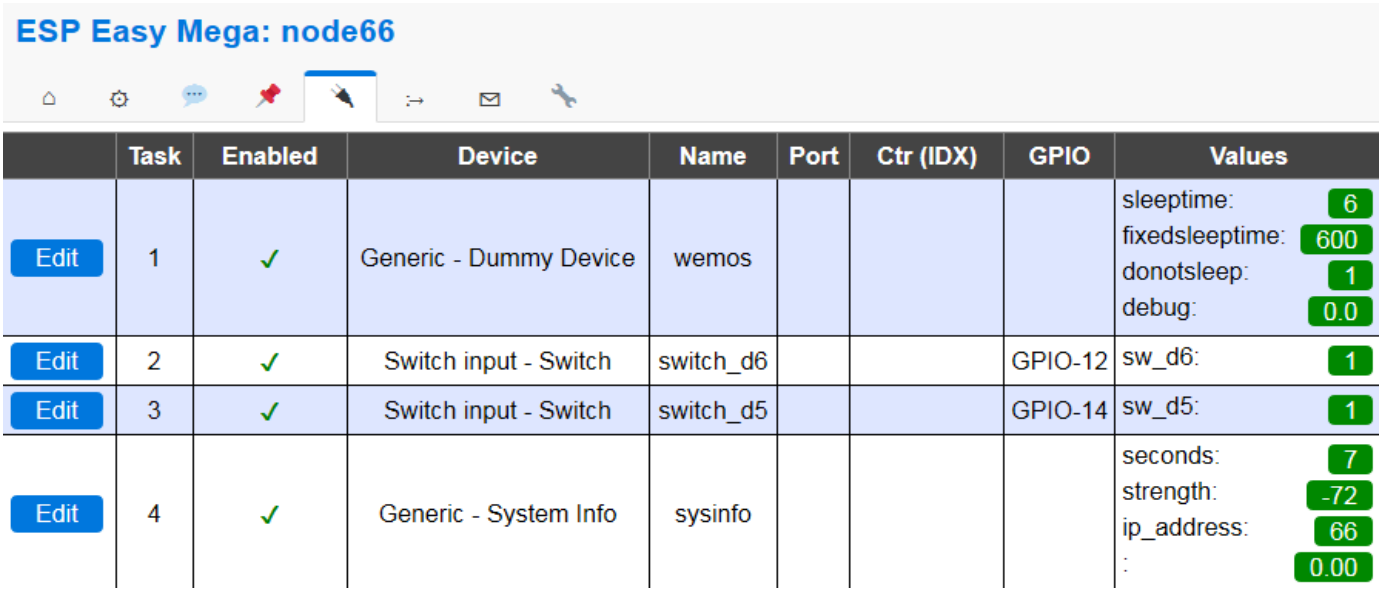
**There is a 'switch input' configured on D6 and D5 (Tasks 2 and 3), a 'generic dummy device' as Task 1 and a 'generic system information' as Task 4.**

# Bohunt School *(Wokingham)*

# Solar-Powered Weather Station

## *Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

**Screen-shot of the 'Devices' tab in ESP Easy**



ESP Easy Mega: node66

| | Task | Enabled | Device | Name | Port | Ctr (IDX) | GPIO | Values |
|---|---|---|---|---|---|---|---|---|
| Edit | 1 | ✓ | Generic - Dummy Device | wemos | | | | sleeptime: 6<br>fixedsleeptime: 600<br>donotsleep: 1<br>debug: 0.0 |
| Edit | 2 | ✓ | Switch input - Switch | switch_d6 | | | GPIO-12 | sw_d6: 1 |
| Edit | 3 | ✓ | Switch input - Switch | switch_d5 | | | GPIO-14 | sw_d5: 1 |
| Edit | 4 | ✓ | Generic - System Info | sysinfo | | | | seconds: 7<br>strength: -72<br>ip_address: 66<br>: 0.00 |

## *Wemos D1 Mini - Rule-set 1*

**Here's the first part of the rule-set.**

```
// Link D0 and RST for wake-up at the end of DeepSleep
// D5 is used to select whether to publish to ThingSpeak or not
// D5=1 (pulled-hi) Publish to Node-RED
// D5=0 (grounded)  Publish to Node-RED and ThingSpeak
//
// D6 is used as an input to sense the operating mode
// D6=1 (pulled-hi)  Node-RED invokes DeepSleep
// D6=0 (grounded)   Wemos invokes DeepSleep
//
// D7 connects to the base of a BJT
//    Logic '0' turns BJT 'off'
//    Logic '1' turns BJT 'on'

on MQTT#Connected do
   TaskValueSet,1,3,0    //Clear the flag wemos#donotsleep=0
   TaskValueSet,1,2,600  //Set wemos#fixedsleeptime to 10 minutes
   gpio,13,1     //Set D7 to logic '1' to turn BJT 'on'
   timerSet,1,6  //Delay 6 secs to allow AUX devices to become active
                 //and take a couple of readings
endon
```

**When the Wemos wakes up (from 'deepsleep') and successfully connects to the MQTT broker it sets two TaskValues. One is a flag for 'wemos#donotsleep' to zero (false) and 'wemos#fixedsleeptime' to 600 seconds (i.e. 10-minutes).**

**The rule-set also sets D7 to a logic '1' (this is used to switch-on power to the auxiliary devices - more about this in the next document).**

# Bohunt School *(Wokingham)*

# Solar-Powered Weather Station

## *Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

**Timer1 is triggered for 6 seconds, which should be enough time for the auxiliary devices to power-up and takes some readings.**

**The next part of the Rule-Set is…**

```
on Rules#Timer=1 do
   event publishReadings    //Publish readings to Node-RED and ThingSpeak
   timerSet,2,3             //Allow 3 sec for Node-RED to send
                              'event,doNotSleep=1'
endon

on Rules#Timer=2 do
   if [switch_d6#sw_d6]=0 and [wemos#donotsleep]=0
      gpio,13,0            //Set D7 to logic '0' to turn BJT 'off'
      timerSet,3,1         //1 second delay then go in to DeepSleep
   endif
endon
```

**When 'timer1' expires an event to publish the transducer readings to Node-RED is called and then another timer is fired (i.e. timer2 for 3 seconds).**

**When 'timer2' expires the Rule-Set checks if D6=0 and the flag 'donotsleep' is zero (false). If both conditions are satisfied (signifying the Wemos is controlling if it goes into 'deepsleep') then 'timer3' is fired for 1-second**

**If the conditions are not satisfied (signifying Node-RED is controlling if the Wemos goes into 'deepsleep') then nothing happens (Wemos remains 'awake').**

**When 'timer3' expires the Wemos enters 'deepsleep' for a fixed time.**

```
on Rules#Timer=3 do
   deepsleep,[wemos#fixedsleeptime]
endon
```

**If the Wemos remains 'awake' then it can be triggered by Node-RED sending an event called 'SampleThenSleep' to the Wemos (detected by the Rule-Set).**

```
on SampleThenSleep do
   TaskValueSet,1,1,%eventvalue1%
                              //Set wemos#sleeptime from %eventvalue1%
   event publishReadings      //Rule Set 2
   if [wemos#sleeptime]>0
      timerSet,4,3            //Delay for 3-sec then go in to DeepSleep
   endif
endon
```

**A parameter can be sent with the event to put the Wemos to sleep for a specified time or not.**
**e.g. SampleThenSleep=6  // Sends the Wemos to sleep for 6-seconds**

# Bohunt School *(Wokingham)*

# Solar-Powered Weather Station

## *Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

**There is an opportunity when the Wemos wakes up from 'deepsleep' and reports the transducer readings to Node-RED, for Node-RED to send an event to the Wemos to allow it to go back to sleep or keep it awake.**

**The event is named 'doNotSleep'. It basically sends a value which is a flag to tell the Wemos what to do.**

**For example, 'doNotSleep=0' sends the Wemos back to sleep.**

**And, 'doNotSleep=1' prevents the Wemos from going into 'deepsleep'.**

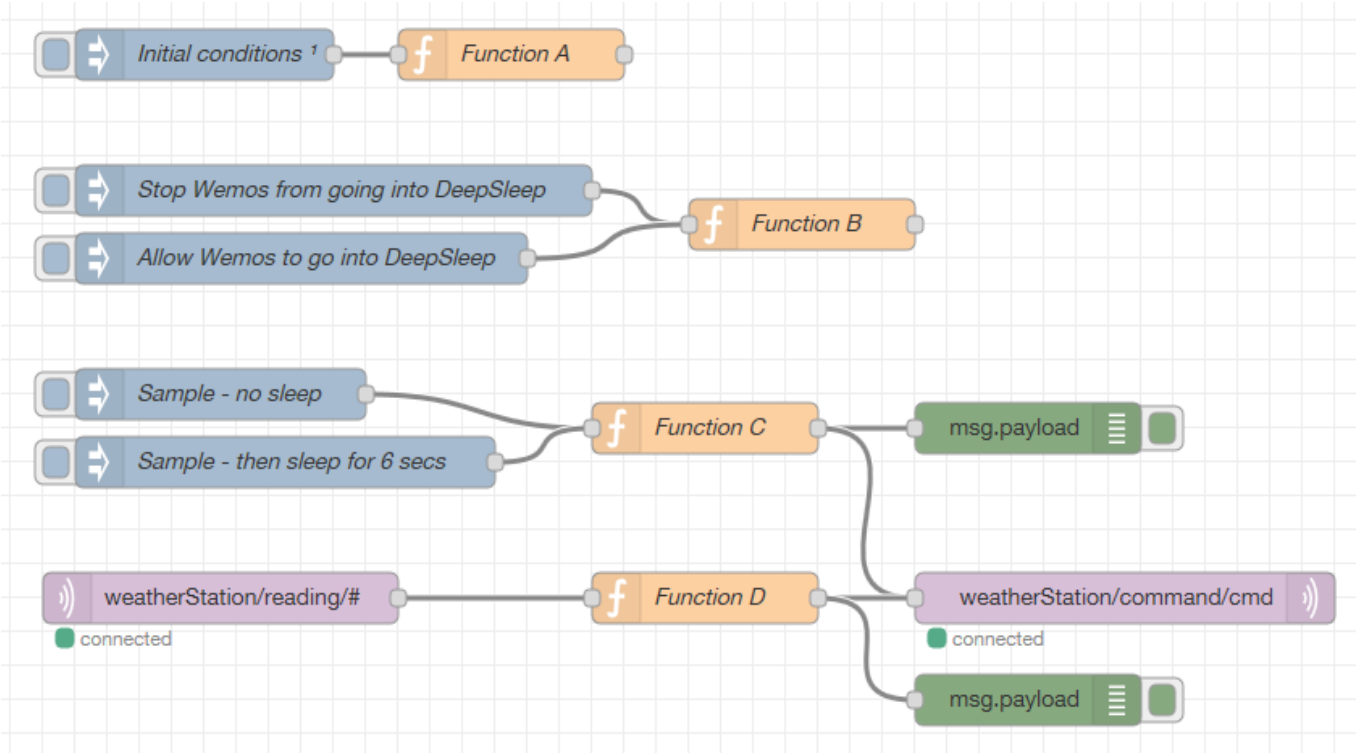
**Here's the rule-set in ESP Easy.**

```
on doNotSleep do
   TaskValueSet,1,3,%eventvalue1%
                     //Set wemos#donotsleep from %eventvalue1%
endon
```

**If you look back to the previous page you can see where the value of 'wemos#donotsleep' is checked (in the `on Rules#Timer=2 do` section).**

## *Node-RED flow*

**Here's the flow to control the 'deepsleep' operation inside the Wemos.**




**Note: The function nodes have been labelled with a letter and are described in detail on the next page.**

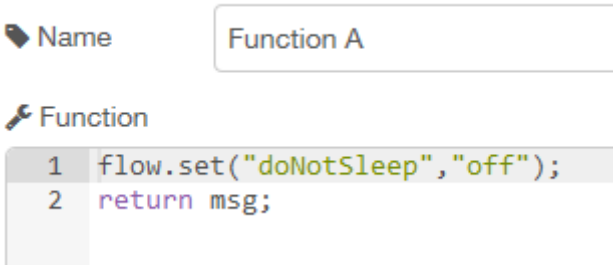
# Bohunt School *(Wokingham)*

# Solar-Powered Weather Station

*Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

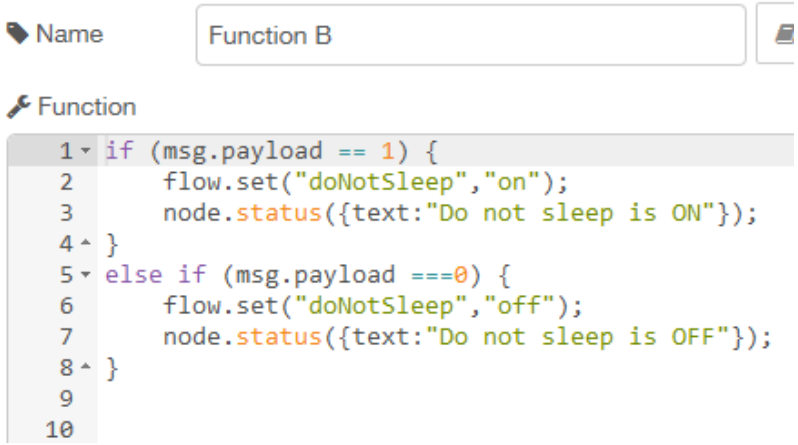
## Function A

**This function is executed when Node-RED start running the flow. It sets a named variable ("doNotSleep" to "off"). This variable is tested, at various places in the flow, to determine whether theWemos goes to sleep or not.**




## Function B

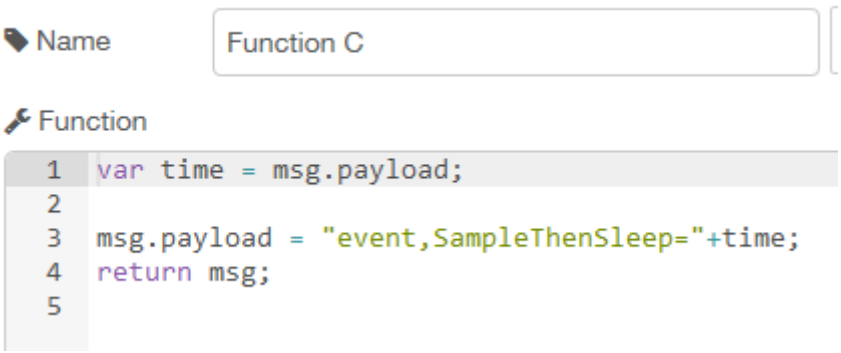
**This function is triggered whenever one of the 'inject' nodes are pressed.**




**It just changes the state of the named variable from the 'inject' node.**

## Function C

**This function sends an event command to the Wemos via the MQTT-Out node.**



## Function D

**This function is triggered whenever the Wemos publishes a set of readings to Node-RED via the MQTT-In node.**


**The function gets the value of the named variable ("doNotSleep") and depending on its state sends a specific event command to the Wemos.**

**This can be used to override the D6 switch setting in the Wemos, so Node-RED can take control over whether the Wemos goes to sleep or not.**

## BeeBotte (MQTT portal)

**As I wanted to be able to see the various readings from the Wemos at my home as well as school, I registered for an account with…** https://beebotte.com

**This means the MQTT-Out and MQTT-In nodes (shown on the previous page) send and receive data with 'beebotte'.**

**Here are the settings for the MQTT-Out node.**

Here are the settings for the MQTT-In node.


Note: I took advantage of this nodes ability to output a 'parsed JSON object'.

If the ability to remotely monitor the weather station is not needed then you could make use of a MQTT broker running on your Raspberry Pi

## Basic experiments

Stage 2 of my project had two key objectives:

- To work out how to put the Wemos into 'deepsleep' for a certain amount of time
- To monitor the output voltage of the solar panel and the state of charge of the Li-Ion battery

To measure the output voltage of the solar panel I made a simple voltage divider consisting of two 10K resistors wired in series across the battery. The mid-point was taken to an input on an ADS1115 (4 channel analog multiplexer and analogue to digital converter).

I used the same arrangement to measure the output voltage of the Li-Ion battery (which is wired to the 5V pin on the Wemos board) and the voltage on the 3V3 pin of the Wemos.

Whenever the Wemos wakes up from 'deepsleep' it publishes these values together with some system parameters (WiFi name, WiFi signal strength and IP address) to Node-RED where they are plotted on a graph.

This means I could see what was happening to the solar panel and the important voltages on the Wemos.

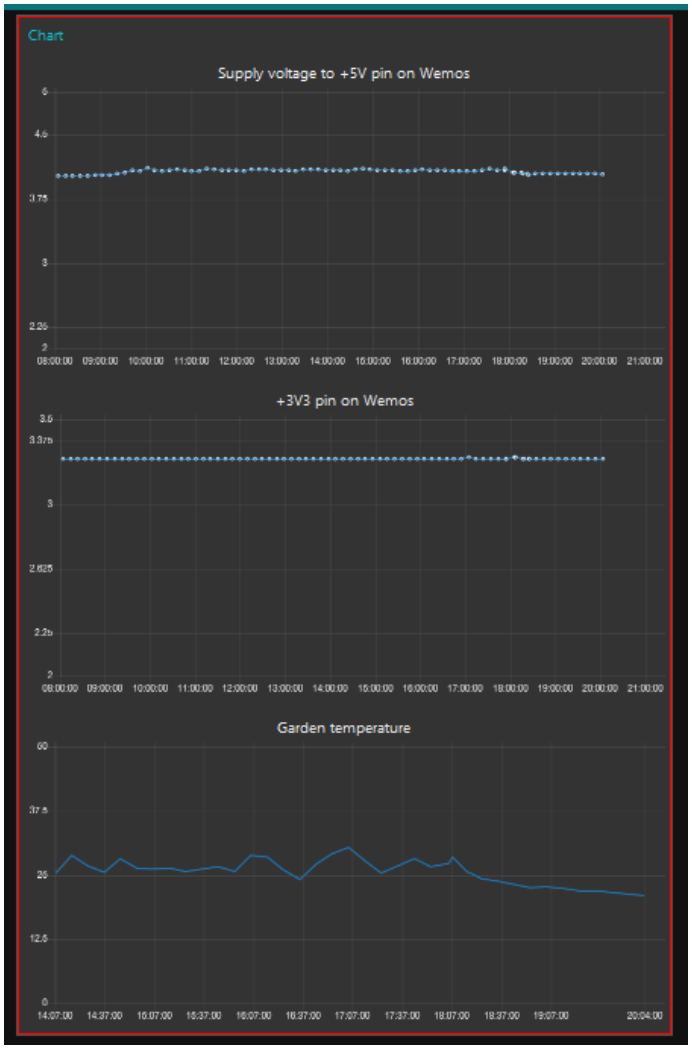
# Bohunt School *(Wokingham)*

# Solar-Powered Weather Station

*Stage 2: Putting the Wemos D1 into 'deepsleep' to conserve the battery's charge*

## Dashboard

**Here's a screen-shot of the simple dashboard I used for these experiments.**




## Resources

**Here's a link to the Rule-Set I used in the Wemos D1 Mini.**

**Here's a link to the Node-RED flow.**

**I need to thank Mr D for writing-up some of the descriptions about the Rule-Sets used in the Wemos and Node-RED (as they were a bit complicated for me).**

**The next thing you need to do is read the write-up for Stage-3.**
***"Switching on/off ancillary devices (e.g. BME280 and ADS1115 modules) to conserve the battery's enercharge"***

*I need to thank Mr D for helping and encouraging me with this project*