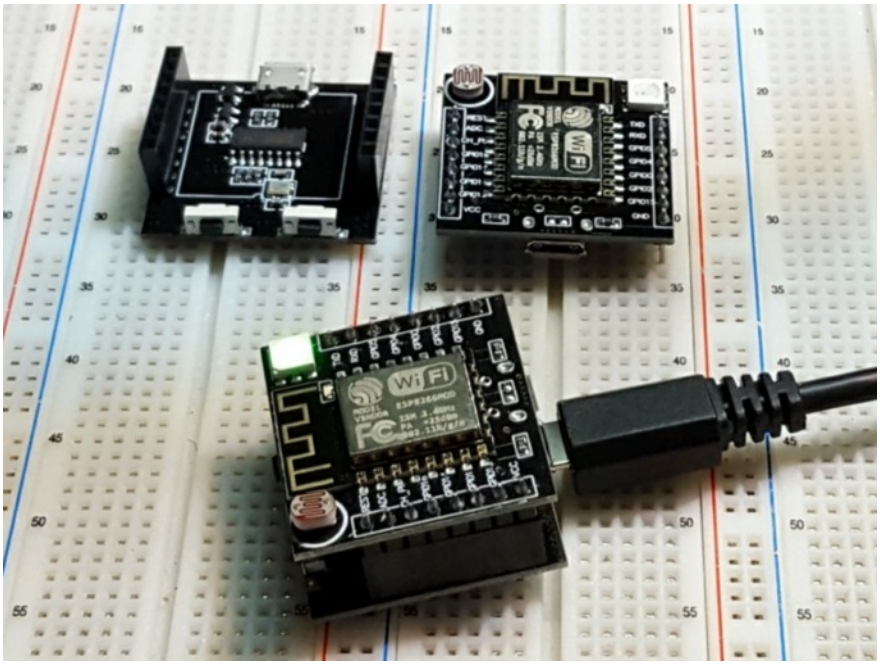
Bohunt School (Wokingham)

Having fun with a Witty node

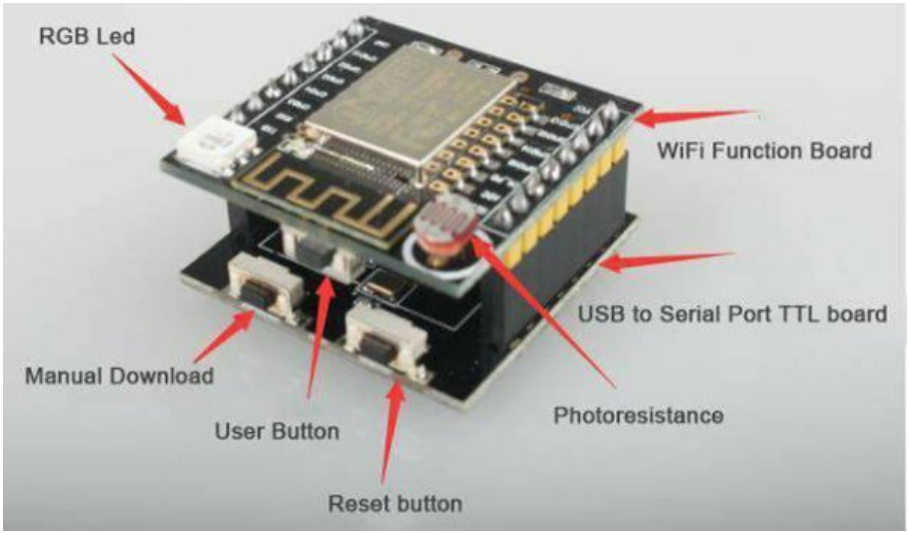
This step-by-step guide will help you find out how to use a Witty node and have a lot of fun discovering what you can do with it.

This is what a Witty node looks like



As you can see there are two parts to a Witty. On the bottom board is a power regulator and a CH430 programming chip.

Located on the top board is the ESP8266 microcontroller, WiFi antenna, a LDR (light dependant resistor), a tri-colour LED and a user button as shown below.



Take a moment or two to familiarise yourself with the position of the items.

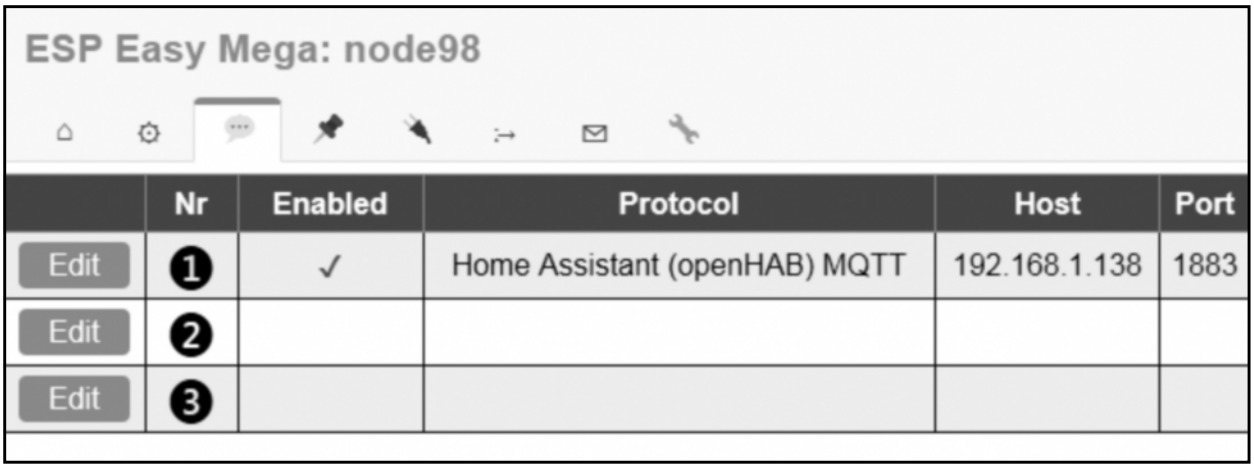
When you use the Witty node please power it with USB cable that has been plugged in to the micro-USB socket on the bottom board.

Setting-up the MQTT Controller

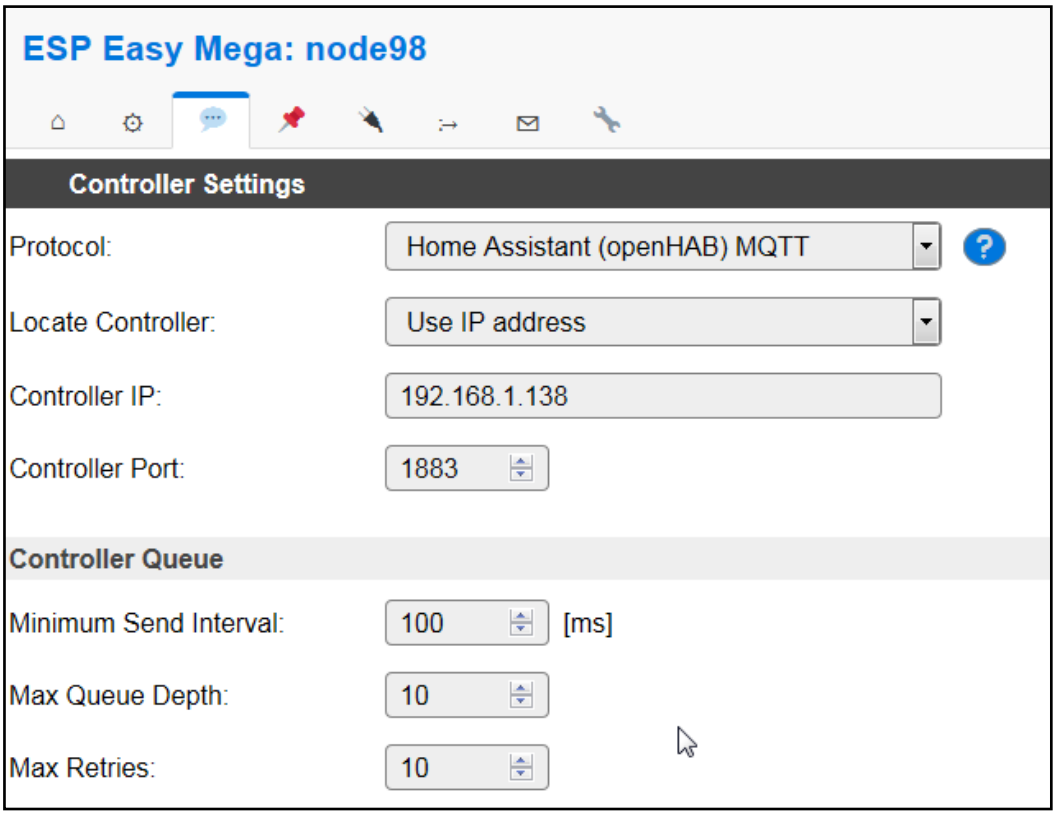
The first thing you need to do is set-up the MQTT Controller inside the Witty.

Login to the Witty by entering its node number (on the bottom of the board) as the URL in a browser. For example… 192.168.1.XX (where XX=node number)

Then select the controller tab (third from the left) as shown below.



Select '*Home Assistant (openHAB) MQTT*' from the drop-down options and then fill-in the following details.



Make sure the Controller IP address matches with the Raspberry Pi server you have been allocated - edit it if necessary and then press 'Submit' and reboot the Witty node using the 'Tool's menu. See Mr D for details.

Bohunt School *(Wokingham)*

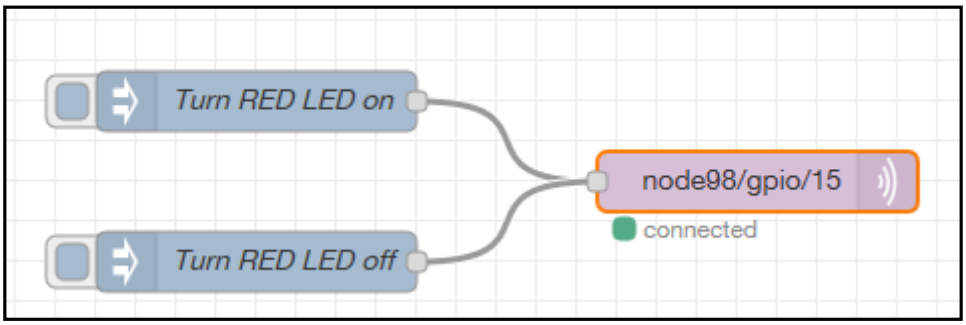
Having fun with a Witty node

Witty node (output connections)

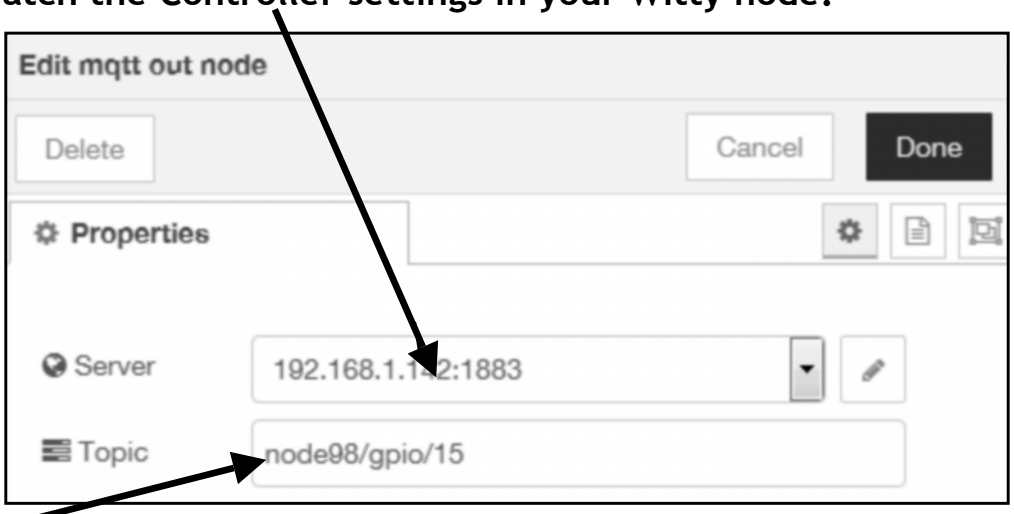
Here's a list of the output pins that operate the tri-colour LED.

- **Red LED GPIO-15 (D8)**
- **Green LED GPIO-12 (D6)**
- **Blue LED GPIO-13 (D7)**

Your first task is to write a simple flow (using Node-RED) to control the LEDs.



The flow consists of two 'inject' nodes (that output a '1' and a '0') that are connected to the 'MQTT-Out' node. The 'Server' setting in the MQTT node needs to match the Controller settings in your Witty node.



Also the 'Topic' setting needs to match your Witty node number. The rest of the 'Topic' contains the letters 'gpio' which indicates you are controlling the *general purpose input output* pins on the Witty node. Finally the two numbers (15 in this example) define the pin-number that will be controlled.

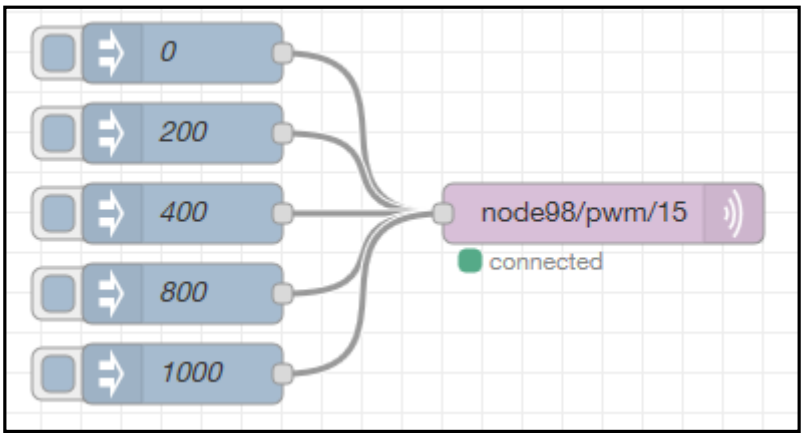
If you have followed these instructions carefully you should be able to control the red LED connected to pin-15 (D8). Have a word with Mr D if you encounter any problems or can't get the node to work.

Once you have the red LED working you can replicate the above flow and change the MQTT-Out settings for the green and blue LEDs.

Pulse Width Modulation (PWM)

The output pins, you have just been using to turn the LEDs 'on' and 'off', have the capability to be driven in PWM mode. Ask Mr D to explain this to you.

To take advantage of this mode requires a minor change to your flow.



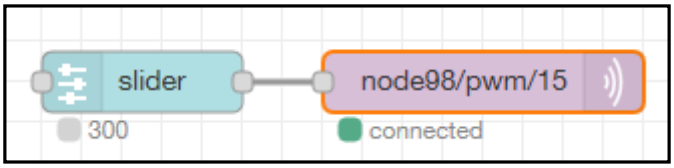
As you can see, there are a few more 'inject' nodes that output values between 0 and 1000 to the MQTT-Out node. These values control the amount of illumination or brightness for the red LED.

The other thing that has been altered is the text within the 'Topic' /gpio/ has been replaced with /pwm/

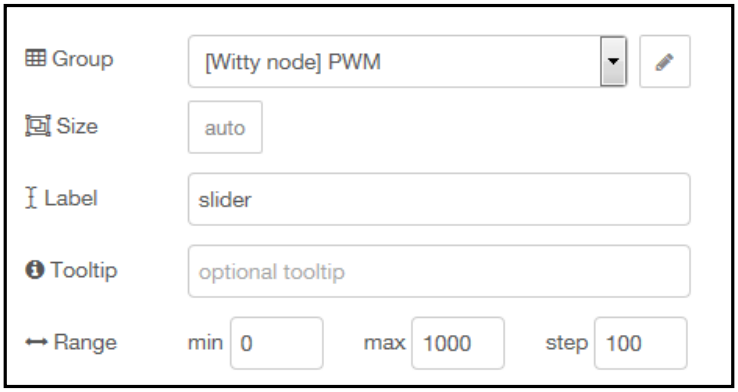
Make these changes and see if you can vary the brightness of the red LED ?

Using the UI_slider to control the LED

Modify your Node-RED flow so it matches the diagram below.



All the 'inject' nodes have been removed and replaced by a slider node (with the following settings). Have a word with Mr D about creating a Dashboard.



Bohunt School *(Wokingham)*

Having fun with a Witty node

Witty node (analog input)

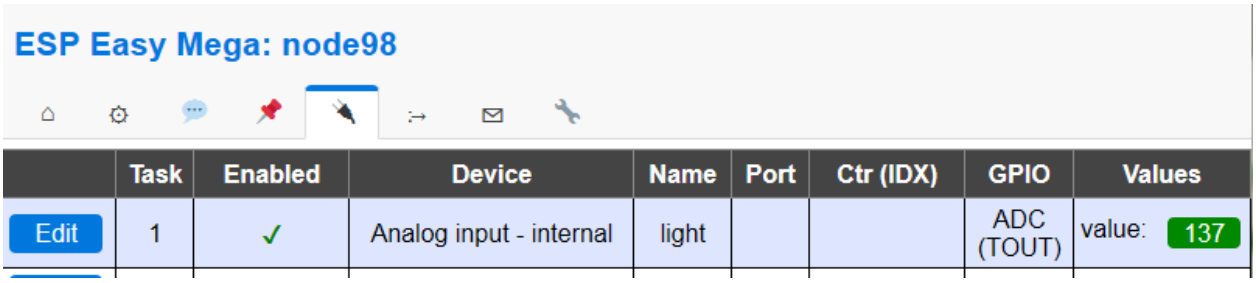
The light dependent resistor (LDR) is situated on the front edge of the top board next to the WiFi antenna. As the name implies, it measures the amount of ambient light falling on the device, and reports this as number between 0 and 1024 (where 0 is pitch blackness and 1024 is full sunlight).

The LDR is connected to A0 which is the analog input on the Witty.

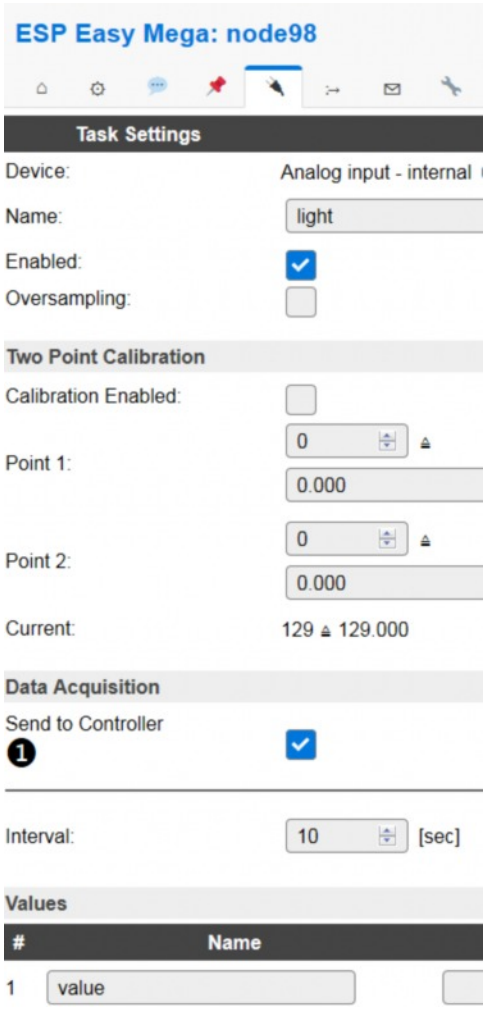
The first thing you need to do is to setup device drivers for the analog input.

Login to the Witty by entering its node number (on the bottom of the board) as the URL in a browser. For example… 192.168.1.XX (where XX=node number)

Then select the 'Devices' tab (fourth from the right) as shown below.



Select *'Analog input - internal'* from the drop-down options and then fill-in the details given here.



Take note of the following settings…

- **The Name is labelled as *light***
- **The Enabled option is ticked [√]**
- **Send to Controller (1) is ticked [√]**
- **The Interval is set to 10 seconds**
  You can change this if you wish.
- **The first name-value is called *value***

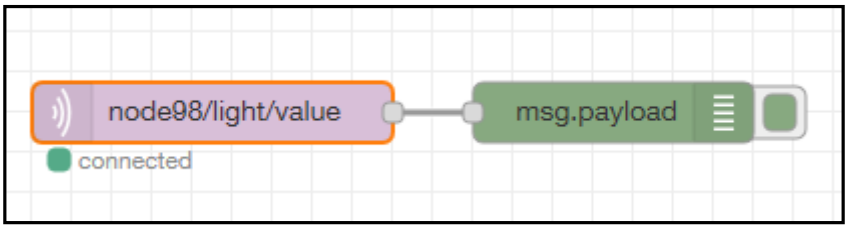
Click the 'Submit button when you have entered these settings.

So what will happen is every 10-seconds the light value will be sent, via MQTT, to Node-RED.

The next step is create a Node-RED flow.

Node-RED flow to receive the LDR value

Using this screen-shot as a guide create this flow.



Pay attention to the setting for the 'Topic' in the 'MQTT-In' node.

Once the flow is complete, click the 'Deploy' button (top-right of Node-RED).

After a few seconds you should see results appearing in the 'Debug' node.

Please note the result is a string of numerical characters. (e.g. "130") This means the first thing you need to do is convert the string to an integer if you want to compare it to a numerical value. There is an built-in function in Javascript to do this for you, it is called 'parseInt' and will be described below.

Using the LDR to control the RGB LEDs

This exercise brings together the two things you have just done, namely turning the LEDs on and off and reading the value of the LDR.

The objective is to read the value of the LDR and turn the red LED on if the light level is above 600; turn the green LED on if the light level is above 399 and less than 601; turn the blue LED on if the light level is less than 400.

In mathematical terms, all you need to do is:

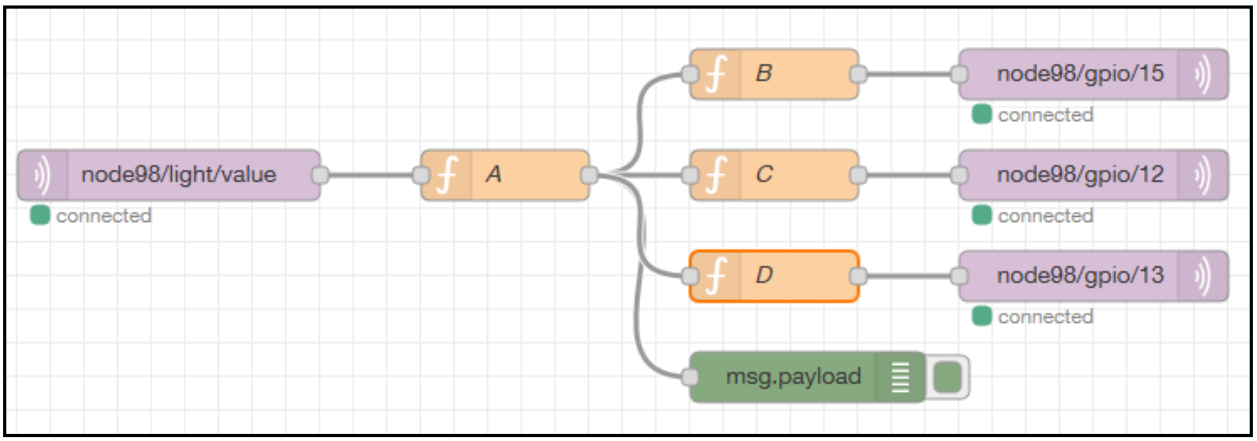
- **Convert the LDR reading from a string to an integer**
- **Turn the red LED if the light level is > 600**
- **Turn the green LED if the light level is > 399 and < 601**
- **Turn the blue LED on if the light level is < 400**

Bohunt School *(Wokingham)*

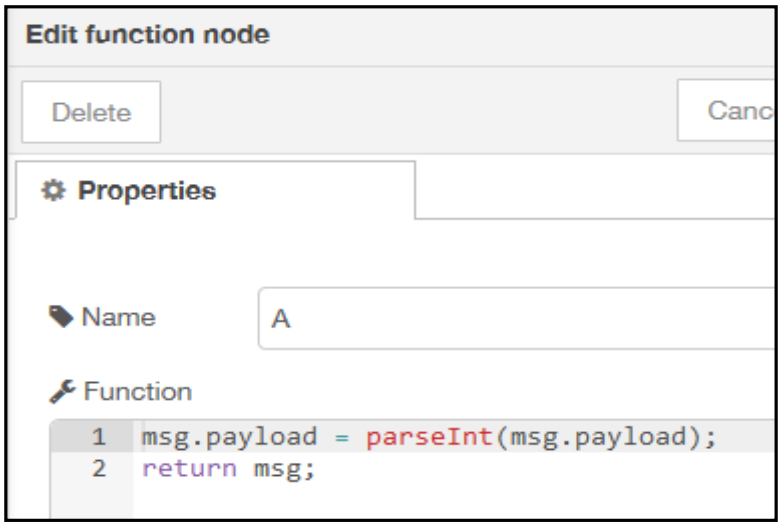
Having fun with a Witty node

Node-RED flow to control the LEDs from the LDR

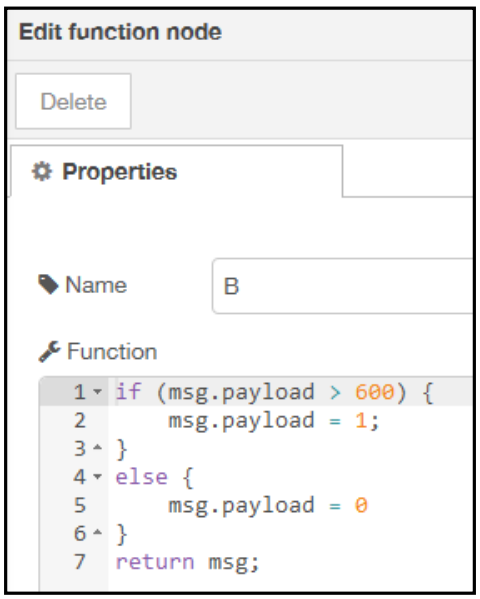
Using this screen-shot as a guide create this flow.



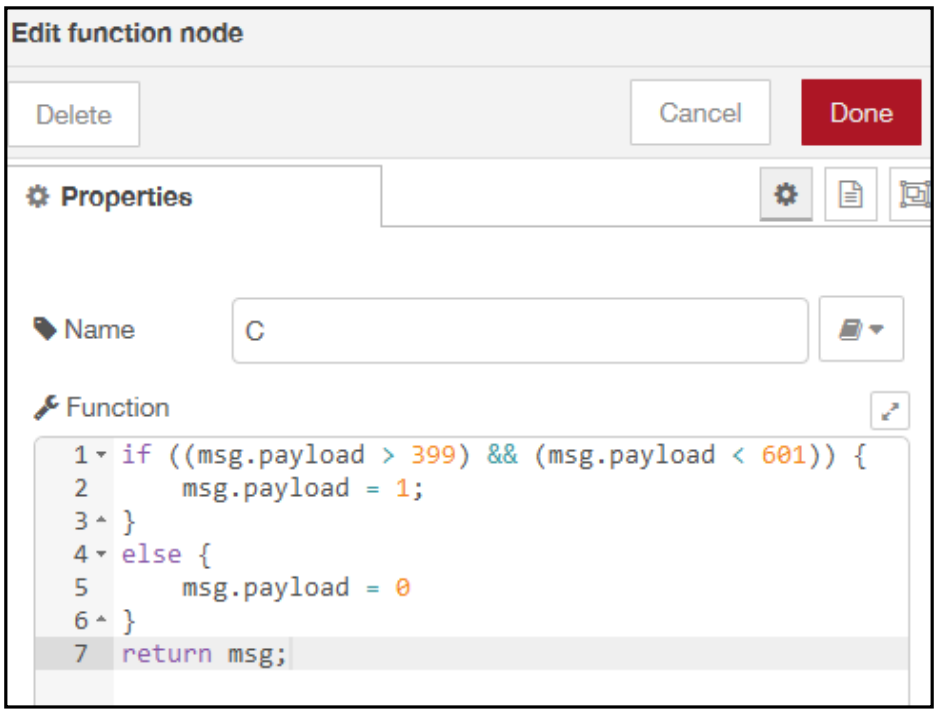
The contents of the function node labelled A is…



The contents of the function node labelled B is…

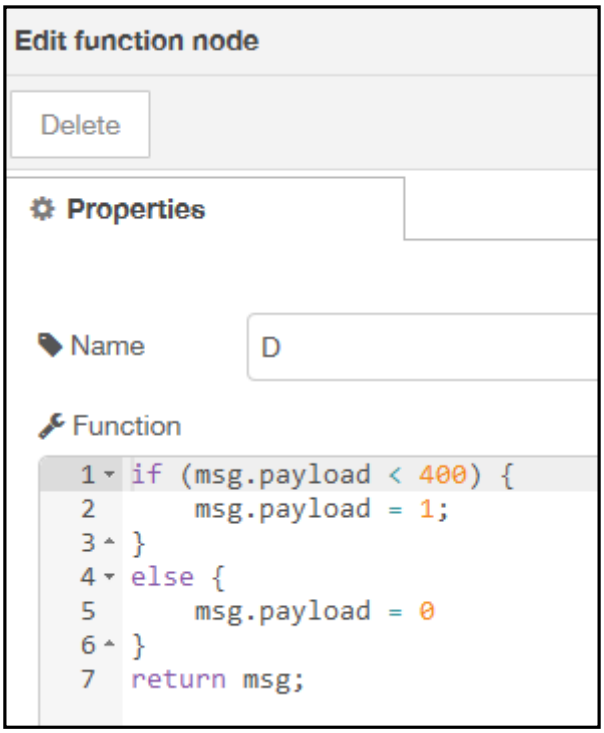


The contents of the function labelled C is…



Pay attention to how the compound IF statement is written. There are two tests that are joined together by the AND operator (&&). These two tests are surrounded by open and close brackets. See Mr D for more details.

The contents of function node D should be obvious, but is shown for clarity.



Try shining the flashlight on your mobile phone at the LDR and see if you can make the different LEDs come on/go off. You may need to turn room lights off.

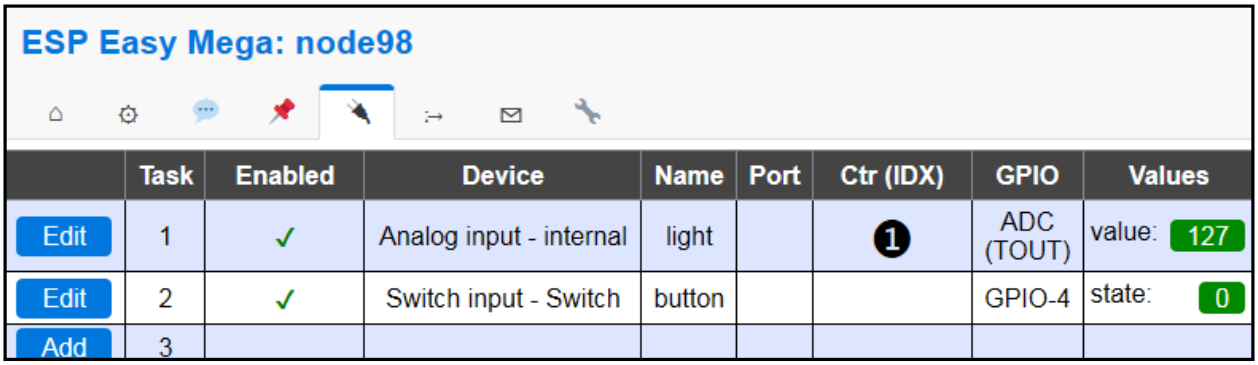
Witty node (push-button input)

Situated along the front-edge on the top board of the Witty is a push-button. It is connected to GPIO-4 (D2) and behaves just like a normal on/off switch.

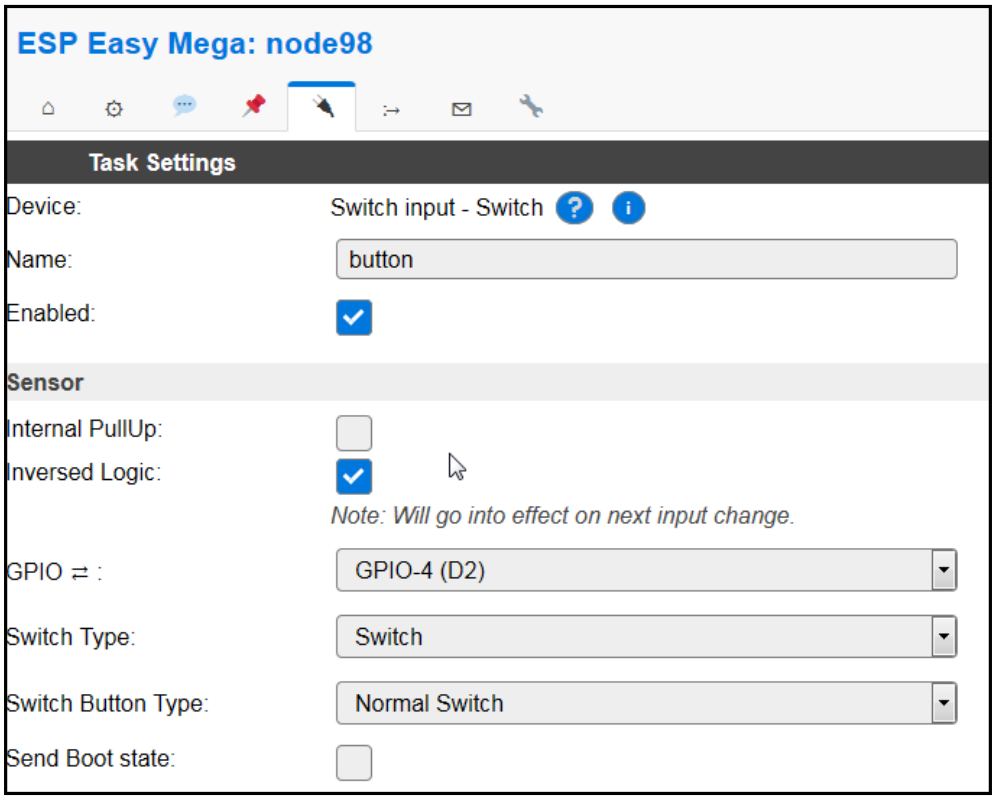
The first thing to do is to set-up the device driver for the switch.

Login to the Witty by entering its node number (on the bottom of the board) as the URL in a browser. For example… 192.168.1.XX (where XX=node number)

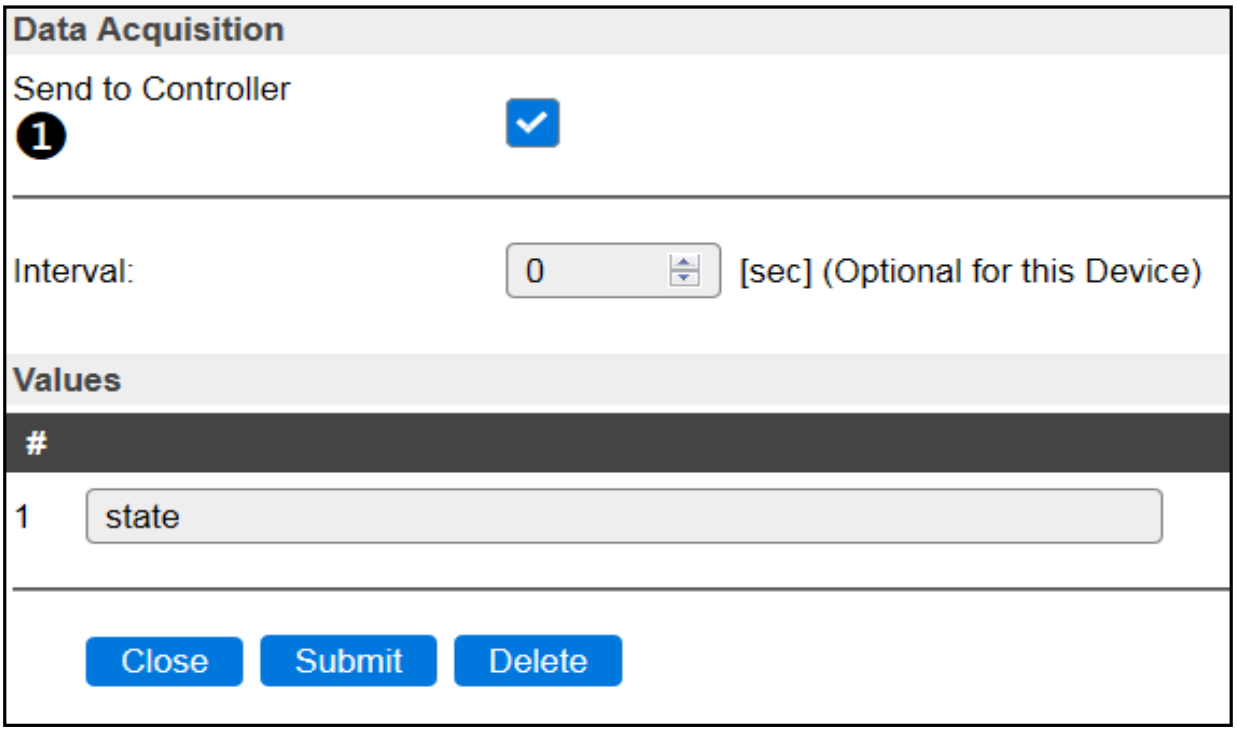
Then select the devices tab (fourth from the right) as shown below.



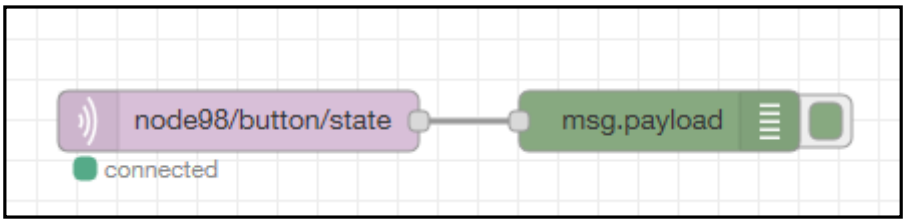
Select the '*Switch input - switch*' from the drop-down options and then fill-in the following details.



The lower half of the Task Settings page is shown on the next page.



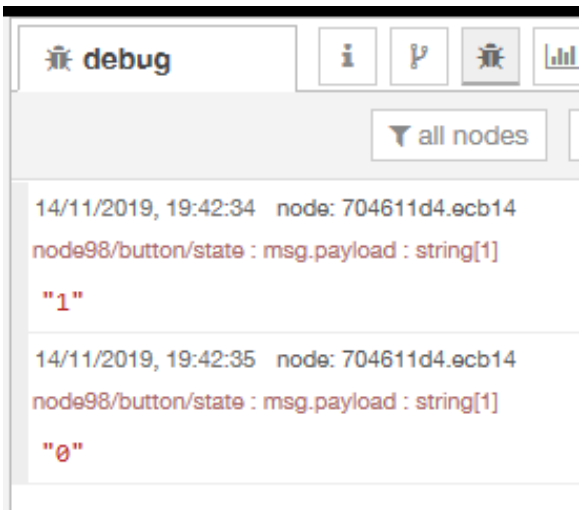
The last step to perform is to create a Node-RED flow to read the switch status.



This is a very simple flow that uses a MQTT-In node to read the switch.

Pay attention to the contents of the 'Topic' as it has to match the name and value you used in the Task Settings for the switch (push-button).

If you have got everything correctly put together, then you should see values appearing in the 'Debug' node when you press the push-button.



If you encounter any problems or your flow doesn't work, then have a word with Mr D.

If you have time you could try operating an LED remotely on one of your other nodes from the push-button on this node.

This is the end of the tutorial.