WiFi Multimeter

GKE Labs R1: Jan 27, 2019

Table of Contents

What it does

Measures the usual electrical quantities and makes them available to WiFi Clients (laptops, mobile phones, tablets) near the multimeter or anywhere on the Internet.

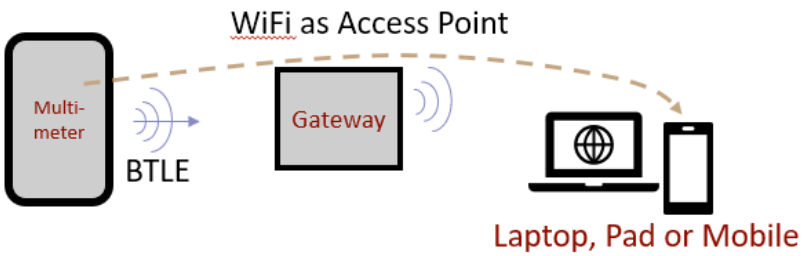
Use Cases & Applications

- Measuring high voltage or current away from hazardous circuits
- Using your tablet to see measurements more conveniently
- Home automation: measuring anything and watching it remotely
- Measuring soil resistance and access it via the Internet
- Measuring your car or your UPS battery voltage
- Measuring the amperage drain of any AC or DC circuit
- Logging battery discharge cycles – creating graphs
- Monitoring (and logging) illumination of your plans – photosensor is required
- Monitoring and logging ambient temperatures
- Logging data during long term experiments (hours, days) and have them available real-time
- … and myriad others

System diagram

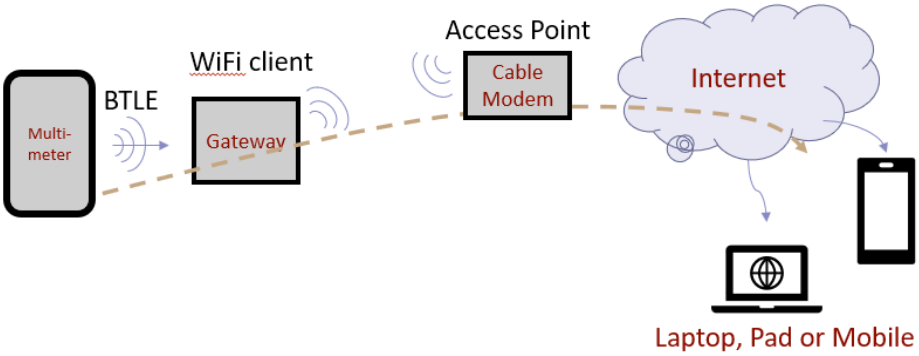
The above use cases can be divided to two classes depending on the distance of the multimeter to the display device. The first case is for distances “around the house or lab” with no Internet connection.

Use Case 1: Multimeter via an Access Point



The second case is for remote locations utilizing the Internet to connect to the multimeter

Use Case 2: Multimeter on the Net

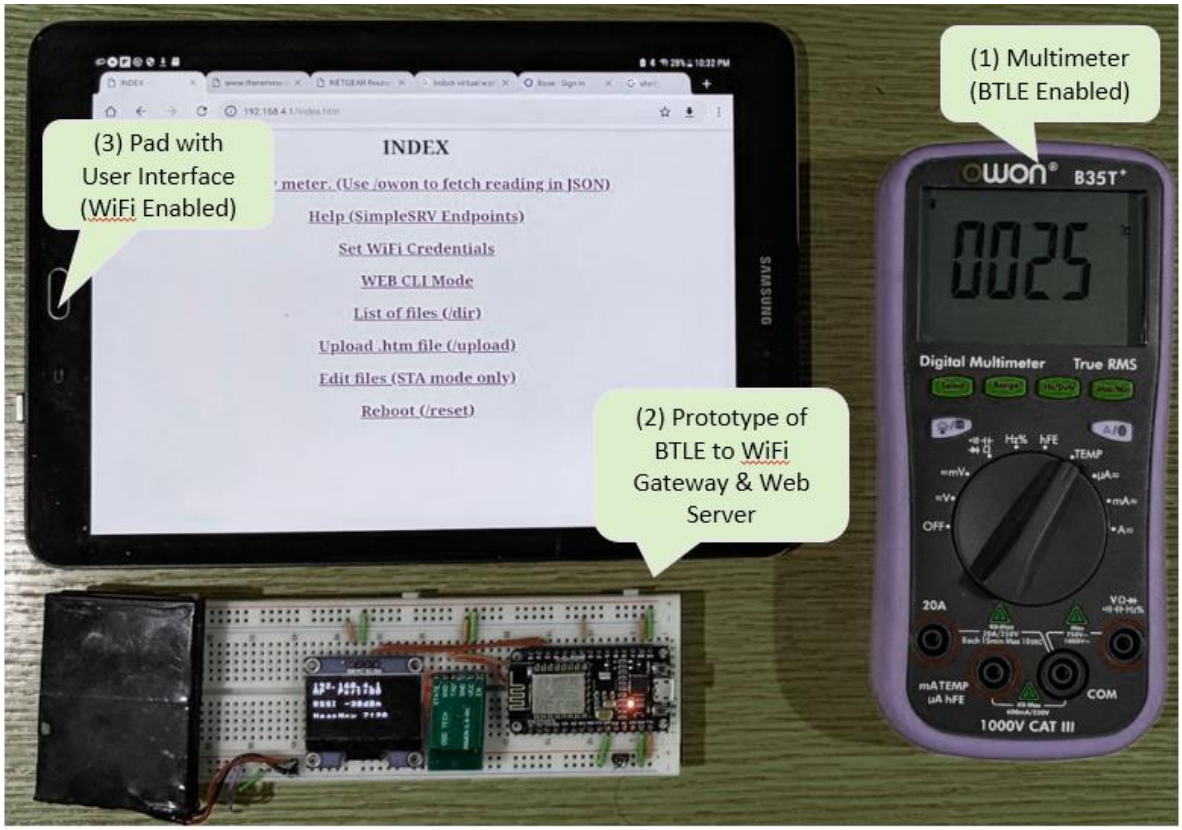


Features and Capabilities

Networking	• Access Point mode (Soft AP) • Station mode (STA) • In STA mode, IP addresses via DNS or fixed • Port 80 or user defined port • mDNS discoverable in the local LAN
Bluetooth and BTLE	• Low power interface; looks for advertisements and then connects • Specific to OWON B35T+ Multimeter of 2400 baud • Multiplexes in s/w Serial Port 1 of the NodeMCU between the meter (2,400 baud) and USB/computer interface (11,520 baud)
Data Display/Logging	• Web pages to display measurements every 1 or 2-seconds • Web page text area to cut-and-paste to Excel spreadsheets • Local USB/Serial connection to log data from the meter (galvanically isolated) • Data logging in Thinker.io Buckets • Data logging to Dropbox on private directory • Data Notifications via IFTTT

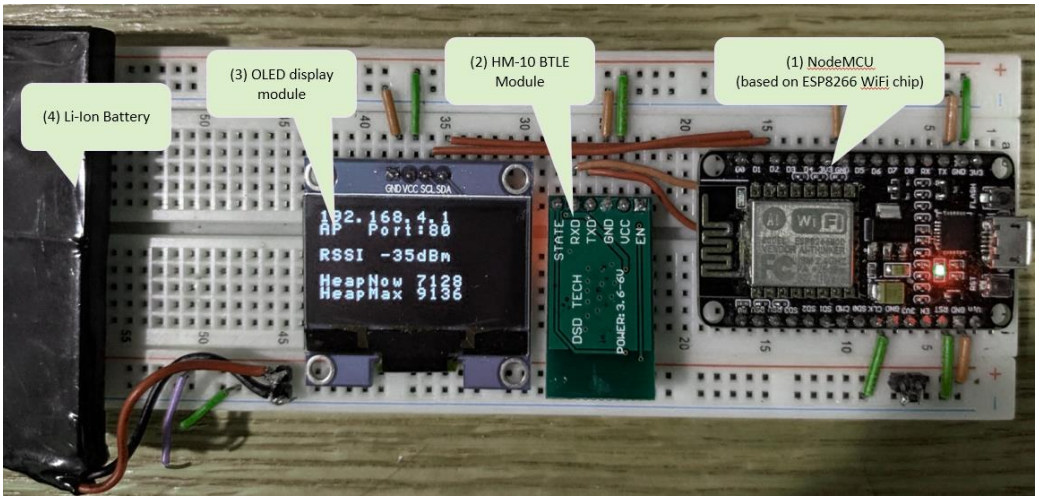
System Modules

The following pictures illustrate the three major modules of the system



The multimeter (1) takes an electrical measurement – in this case, the multimeter reads it's own temperature. The reading is transmitted to the gateway (2) which is also running a web server; the web server is configured as an Access Point (AP) so it is discoverable by any WiFi client – in this case the Pad (3). The landing page of the Pad is an index of various functions available.

The Gateway consists of the sub-modules shown below.



The “main brain” of the gateway is the NodeMCU (1) based on the ESP8266 WiFi processor; this module includes a 32-bit ARM processor, full WiFi antenna, MAC and IP stacks, 0.5MB of memory, USB interface circuit and utilizes the Arduino IDE for all code development.

The NodeMCU connects via a serial port to a BTLE module (2) which is programmed to be receiving transparently the BTLE Multimeter readings.

The prototype is enhanced with a small OLED display (3) which is used to display local status information and optionally the multimeter readings. Last, the prototype is powered by a small Li-Ion battery (4).

User Modes of Operation

There are three modes of operation of the Gateway:

- ***CLI Mode***
- ***AP Mode***
- ***STA Mode***

In ***CLI mode***, the user interfaces with the Gateway via a USB connection to the NodeMCU Serial Port via a generic terminal. This mode is useful for configuration and setup of the unit; for example, the user can test various I/O pins, the EEPROM, the display, WiFi parameters, etc.

The ***AP Mode*** is used for connecting the Multimeter to a WiFi Client directly, i.e. not using an infrastructure Access Point. The gateway itself behaves as an Access Point discoverable by the WiFi Clients with SSID “GKE_LABs” and no password. After connecting the WiFi Client, the browser can be directed to 192.168.4.1 which is the main web server published by the Gateway.

The ***STA Mode*** is used to connect the Multimeter to the Internet via an infrastructure Access Point. The WiFi Credentials are stored in the EEPROM and, if correct, connection is straightforward. If not the user can use the AP Mode to define the infrastructure SSID, the password and optionally the port number for the web server. Futhermore, instead of using the default DNS for obtaining the IP address of this WiFi Client, the user can specify a fixed IP address.

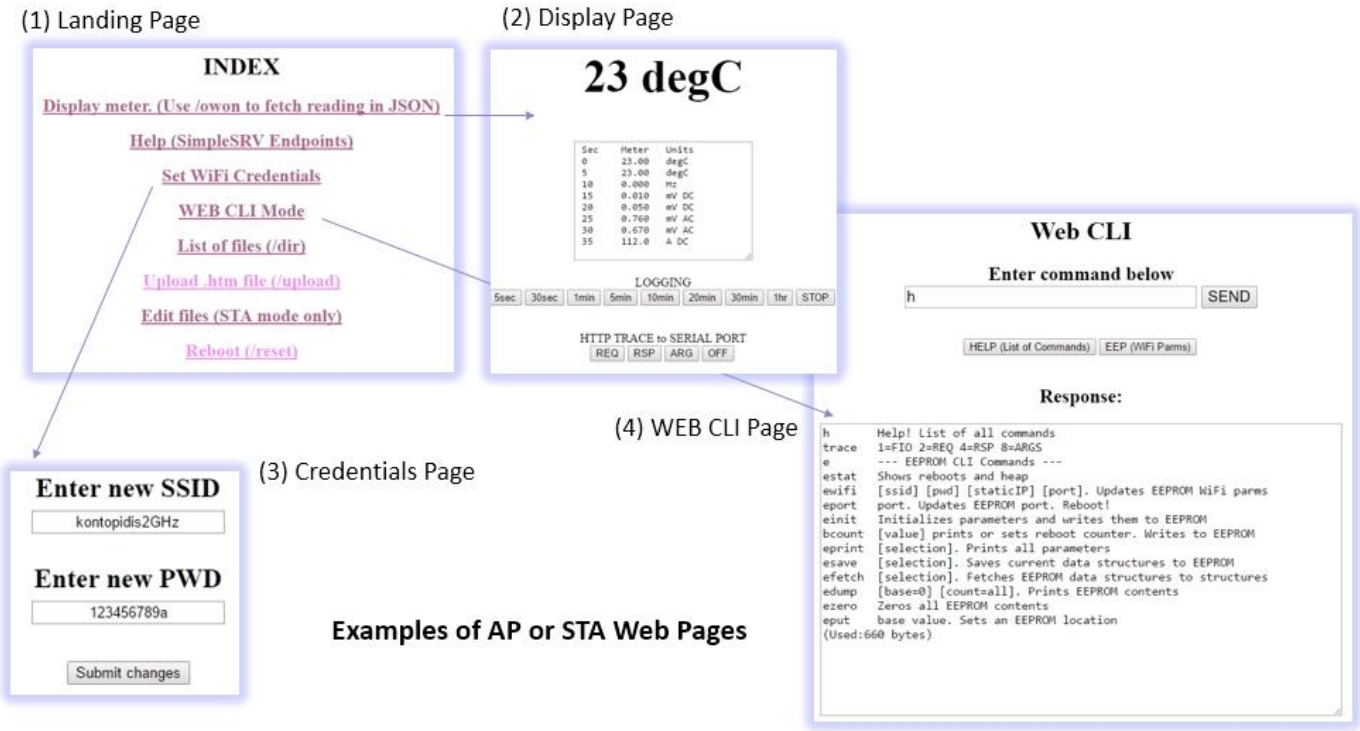
Selection of Modes of Operation

Selection of the mode of operation only occurs during “boot-up”, i.e. after cold RESET. The sequence is as follows:

1. User presses RESET and releases
2. NodeMCU starts booting; this is indicated by the LED blinking 3-times and the OLED display prompting the user to either press USB/Serial RETURN or press the FLASH button.
3. If the USB/Serial interface is active a RETURN is detected within 10-seconds, the ***CLI Mode*** is selected; this is indicated by the LED turning ON permanently and the terminal displaying the prompt “cmd:” and waiting for user commands. Press “h” for help and directory of commands.
4. If the FLASH button is pressed within 10-seconds, the ***AP Mode*** is selected; the LED turns ON and the OLED display indicates the SSID and IP Address the WiFi Client can connect to. Every time a client connects to the Gateway, the LED blinks OFF and remains ON thereafter.
5. If there is no user interaction after pressing RESET for 10-seconds, the Gateway uses the EEPROM parameters to connect to an Infrastructure Access Point in ***STA Mode***, i.e. as an WiFi Client. The user may
    a. access the Gateway locally using the specified IP Address (shown in the OLED display), or
    b. use the Thinger.io IoT Service to access the OWON-B35T Dashboard, or
    c. forward the port to the Infrastructure AP for full access over the Internet – more details on this follow in the next section.
6. To exit either the CLI mode or the AP Mode, the user must press RESET to reboot the processor.

Description of the WEB Pages

In ***STA Mode*** (or AP Mode) the following WEB pages can be used for the associated function or information:



More specifically, the Credentials Page (3) can be used in ***AP Mode*** to set the credentials of the subsequent (after RESET) ***STA Mode***.

The Display Page (2) is the user interface area where:

- Measurements are displayed in large letters, including the units
- Optionally you can start logging in a scrolling area measurement every 5-sec, 30-sec, …, 1hr. At any point you can cut-and-paste this area to an XLS spreadsheet for graphing or further analysis – the format is TAB separated columns and imports directly to XLS.
- Optionally you can enable USB/Serial port diagnostics.

The WEB CLI Page (4) allows you to enter any command, the same way as in ***CLI Mode***. The difference is that this page can be involved remotely via the Internet. (Future expansion can include NodeMCU I/O pin controls.)

In addition to the above pages, there are two more pages worth noting: the Help Page (1) and the local file & script Editing Page (2). The last one is available only in STA Mode, requiring full Internet connection – it allows the user to enhance and modify any of the web pages used either in ***AP Mode*** or ***STA mode***.

(1) Help Page

SimpleSRV Endpoints

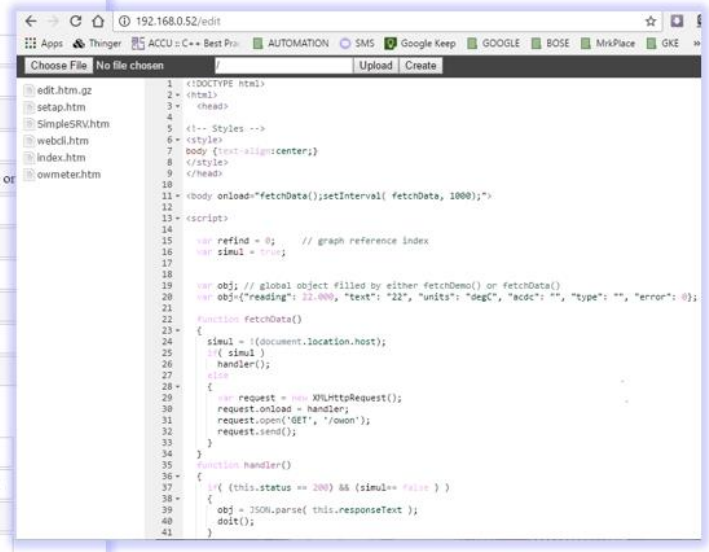
Inline endpoints

Endpoint	Function
/restart, /reset	Restarts the processor
/status	Displays in json format SSID, and PWD
/trace	Displays in json format the serial port trace
/set?trace={value}	Sets trace to specific value REQUEST, ARGS, FILEIO, JSON, or
/dir	Displays (in HTML) directory of all files
/eep	Displays HTML the WiFi EEPROM parameters
/ed?delete={fname}	Deletes specified file
/ed?new={fname}	Creates a new file
/upload	Uploads a file (overides previous one)
/favicon	Your favorite icon

Endpoints requiring .htm files

Endpoint	Function
/edit	Invokes an interactive editor of all files in the file system
/setAP.htm	Sets STA WiFi credentials
/webcli.htm	Invokes the Web CLI
/index.htm	Application specific screen with pointers to other pages

(2) File/Script Edit Page



Thinger.io Enabled!

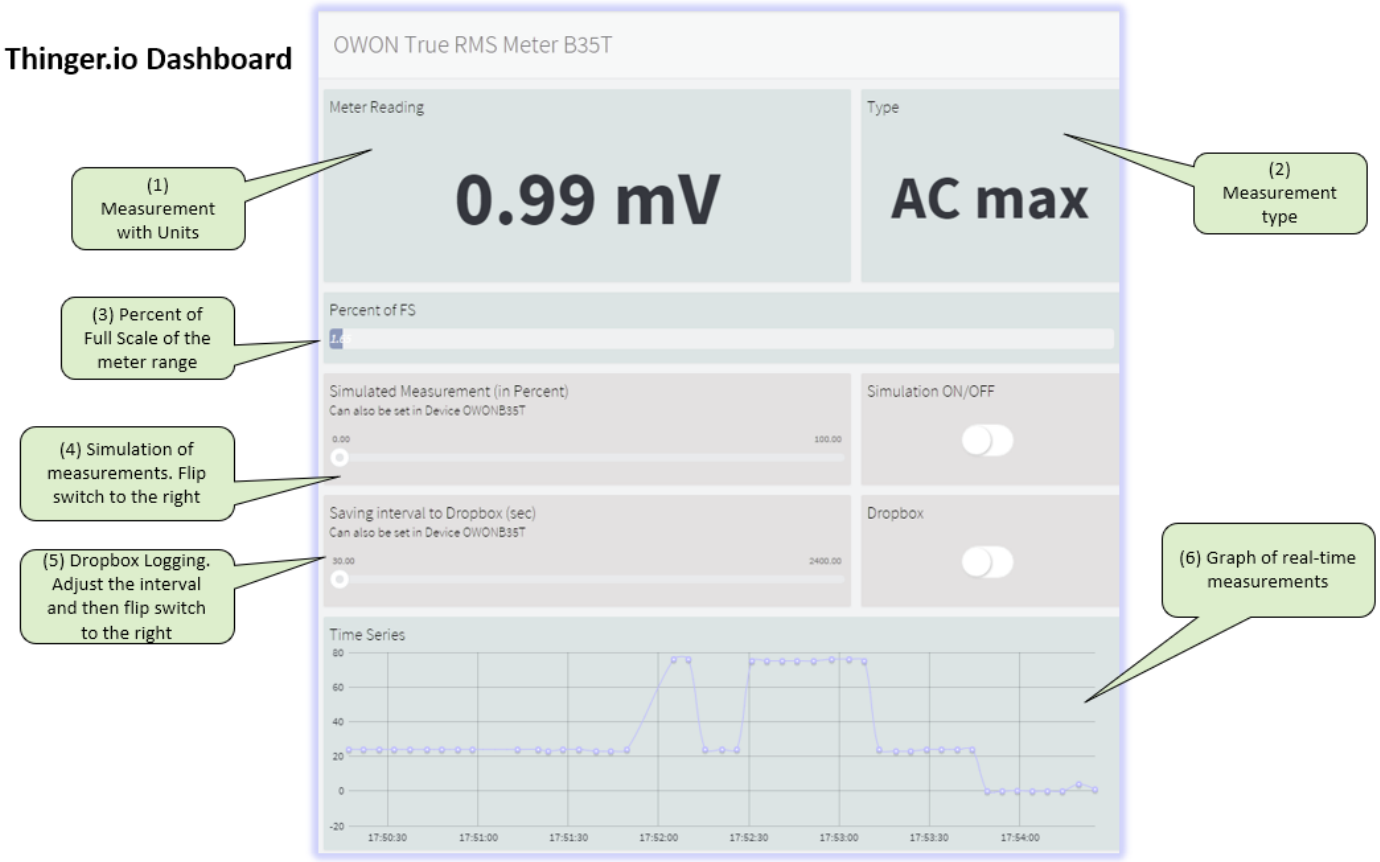
Thinker.io is an IoT platform which allows fast prototyping of internet connected “things”. The platform includes advanced embedded code to be included in “the things”, rich interfaces in form of dashboards and extensive connections to other web services.

The main advantage in connecting “things” to Thinger.io is ***real-time connectivity behind the firewalls***. Client devices utilize Port 80 only; no need for DMZ enablement or port forwarding.

The WiFi Multimeter has a Thinger.io presence in form of:

- A Device
- A specific Dashboard
- A logging Bucket
- An IFTTT interface

The following screen illustrate the OWON B35T Dashboard



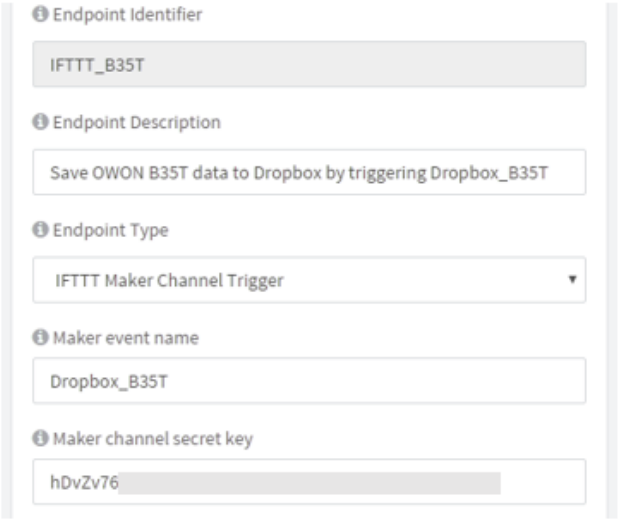
A Thinger.io Bucket is also used to log data to the Thinger.io cloud storage.

Thinger.io Databucket for the WiFi Multimeter

Bucket Explorer

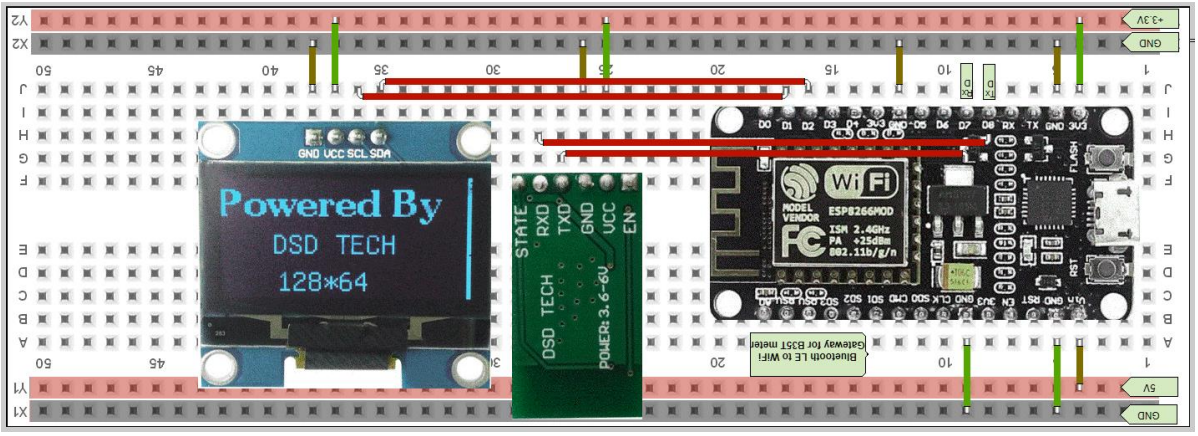
Date	acdcmax	percent	simValue	string	value
2019-01-27T17:55:01.037-0500	AC max	2.1	0	1.26 mV	1.26
2019-01-27T17:54:31.479-0500	AC max	1.7	0	1.02 mV	1.02
2019-01-27T17:54:01.473-0500	DC	-0.0666667	0	-0.04 mV	-0.04
2019-01-27T17:53:31.319-0500		23	0	23 °C	23
2019-01-27T17:53:01.459-0500		35.8491	0	76 °F	76
2019-01-27T17:52:30.941-0500	min	35.3774	0	75 °F	75
2019-01-27T17:52:01.391-0500		35.8491	0	76 °F	76
2019-01-27T17:51:41.421-0500	min	23	0	23 °C	23
2019-01-27T17:51:11.339-0500		24	0	24 °C	24

Logging of measurements to Dropbox is enabled using the IFTTT Webhook feature using the following End Point:

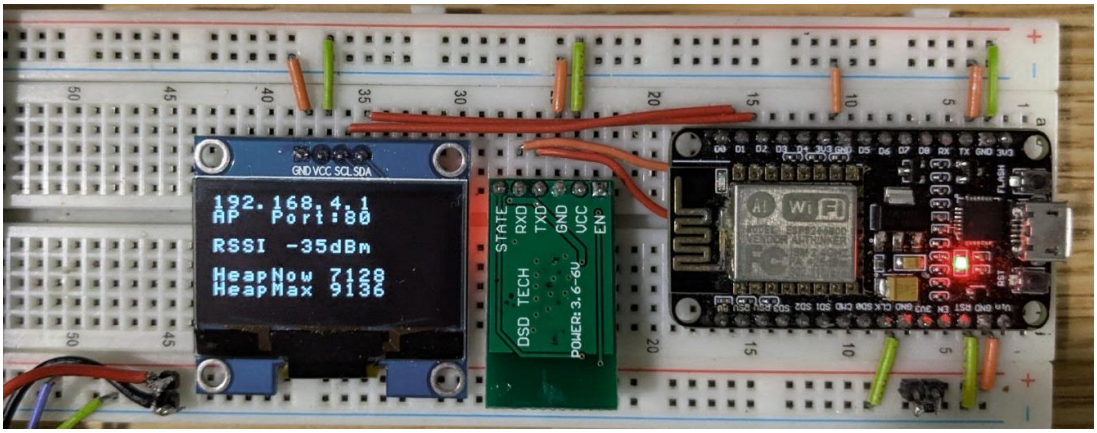


Prototyping

The PEBBLE layout package was used to construct the prototype. The NodeMCU, BTLE and OLED components were added. The finished design is shown below. (Labels were printed upside down!).



For comparison, the real/physical prototype is shown below.

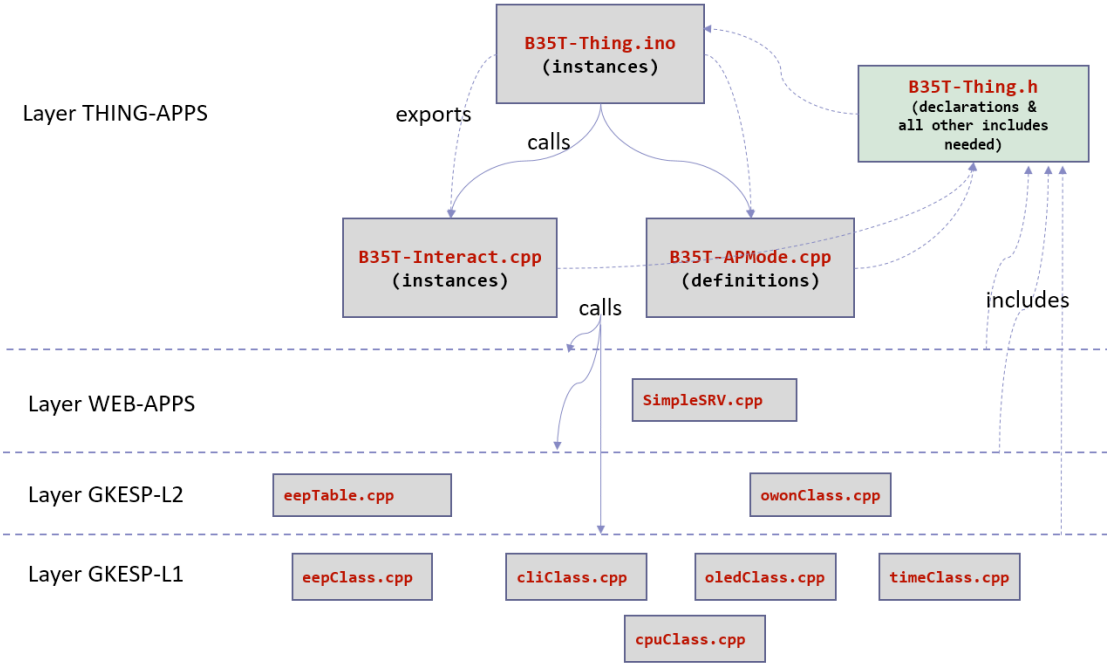


The PEBLE Source code is listed below:

```
IC||60|128|1||U?|||NodeMCU|IC||NodeMCU_1
Wire||155|402|21||11|#FF9900|2|11||10|
Note||13|448|2|||1||GND||NOTEPAD_12
Note||13|476|2|||1||+3.3v||NOTEPAD_12
Wire||128|402|21||11|#FFFF00|3|11||10|
Wire||128|44|21||11|#FF9900|2|11||10|
Note||15|3|2|||1||GND||NOTEPAD_12
Note||14|36|2|||1||5V||NOTEPAD_12
Wire||155|17|21||11|#FFFF00|3|11||10|
Wire||263|17|21||11|#FFFF00|3|11||10|
Wire||344|402|21||11|#FF9900|2|11||10|
IC||656|9|1||U?|||HM10|IC||HM10_1
Wire||723|402|21||11|#FF9900|2|11||10|
Wire||695|402|21||11|#FFFF00|3|11||10|
IC||837|46|2||U?|||OLED128x64|IC||OLED128x64_2
Wire||1047|402|21||11|#FF9900|2|11||10|
Wire||1020|402|21||11|#FFFF00|3|11||10|
Wire||481|391|12||11|#FF0000|19|11||10|
Wire||454|403|13||22|#FF0000|19|11||12|
Wire||238|336|12||11|#FF0000|20|11||10|
Wire||265|317|13||11|#FF0000|18|11||10|
Note||233|391|2|||3||Tx<br/>D||NOTEPAD_32
Note||260|392|2|||3||Rx<br/>D||NOTEPAD_32
Note||385|60|1|||2||<h4>Bluetooth LE to WiFi Gateway for B35T meter</h4>||NOTEPAD_21
BREADBOARDSTYLE=BB36
SHOWTHETOPAREA=false
```

Source Code Organization

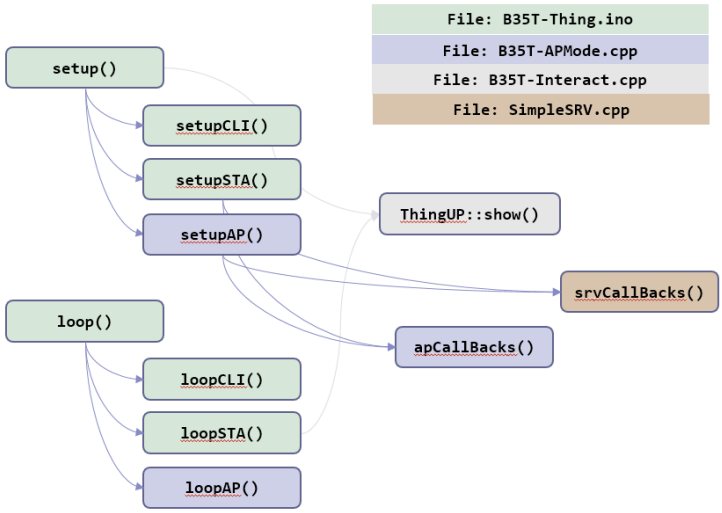
The code is organized in three main files, B35T-Thing.ino, B35T-Interact.cpp and B35T-APMode.cpp. These files use the lower layer libraries extensively. The following hierarchical scheme is used.



All conventions listed in GKESP-L1 repo are followed, i.e. each layer can only call functions of the lower levels and each module can be combined independently of other modules.

The main code, at a functional level, is organized also hierarchically. The AP Mode functions have been separated to allow clean allocations if AP Mode is not needed. Also notice that all OLED interfaces are handled by a single class, called ThingUO.

Main Functions



Finally, all instantiation of classes is done by the main code; this way heap is used very carefully and the limited stack does not overflow. Because important classes are allocated by the main code, it is unavoidable to use a couple global variables/classes (as few as absolutely necessary!). These are shown in the diagram below.

Class Allocation and Instance Access