Implementation of intelligent modules with ESP8266

Semestral project

Tanasis Vlachopulos

May 12, 2017

Content

1. Project aims
2. Introduction of ESP8266
3. Development options
4. Introduction of EspHub
5. EspHub library
6. EspHub server
7. Future

Project aims

Project aims

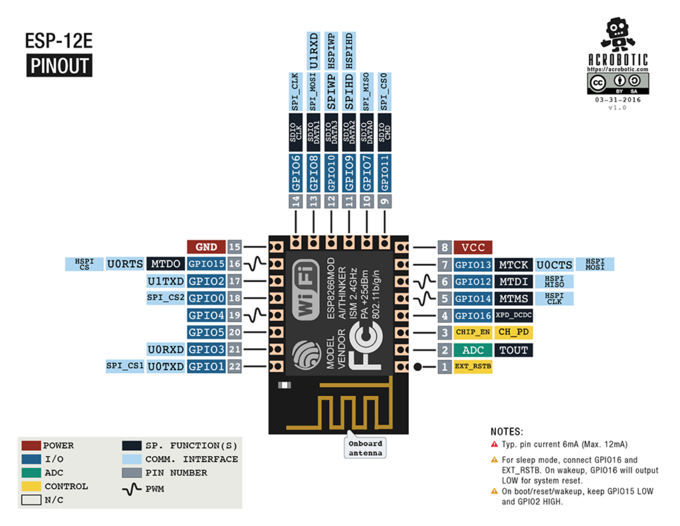
- Explore the development options for ESP8266 modules.
- Create a coherent system made up of ESP8266 modules connected with suitable sensors and display modules.
- Implement server that handles the communication between modules.
- Design a communication protocol that allows you to communicate with any number of modules, with PUSH notification and the ability to dynamically add modules.

Introduction of ESP8266

Introduction of ESP8266

- Originally a simple WiFi (802.11 b/g/n) to serial converter with full TCP/IP stack.
- Driven by 32-bit Tensilica Xtensa MCU, CPU frequency 80 or 160 MHz, with 96 KiB of RAM and up to 4 MiB external flash memory.
- Provide SPI, I2C, UART, 10-bit ADC and 16 multipurpose GPIO pins.

Introduction of ESP8266



NOTES:

- Typ. pin current 6mA (Max. 12mA)
- For sleep mode, connect GPIO16 and EXT_RSTB. On wakeup, GPIO16 will output LOW for system reset.
- On boot/reset/wakeup, keep GPIO15 LOW and GPIO2 HIGH.

Development options

Development options

- Official Non-OS SDK and RTOS SDK based on FreeRTOS real-time operation system.
- Community SDK based on Xtensa open-source GCC toolkit
- Arduino Core for ESP8266 - the most popular
- NodeMCU Lua
- MicroPython
- Espruino based on JavaScript

Arduino Core for ESP8266

Pros

+ Support wide spectrum of libraries.
+ Easy to use with Arduino IDE.
+ Great community support.

Cons

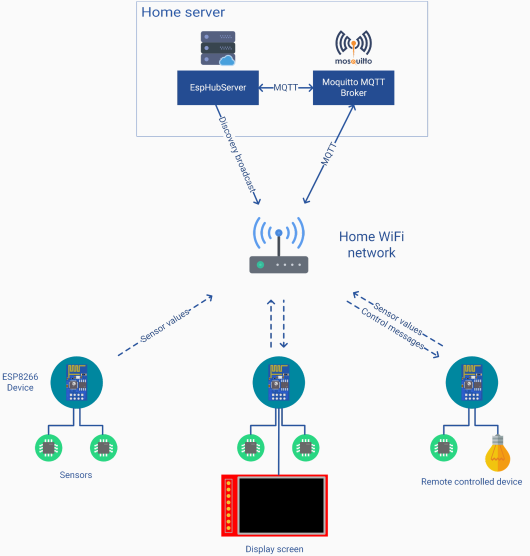
- Arduino vs ESP8266 hardware incompatibility.
- Forever loop.

Introduction of EspHub

Motivation

- Create system that allows easily connect new device to existing infrastructure.
- Notify user when a new device pops up.
- Manage devices and collect data from sensors.
- Visualize data.
- Visualize data on remote display.

EspHub overview



EspHub library

EspHub library

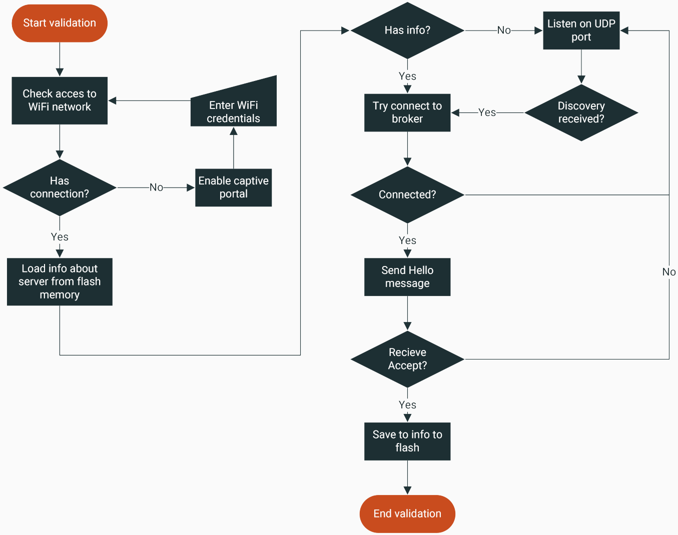
- Library for Arduino platform which handles connection between ESP8266 and EspHub server.
- Searches for active servers on the network and tries connect to them.
- Save info about server into internal flash memory if server is authorized.
- Provide simple user API for communication with server.
- Send device telemetry data, such as SSID, RSSI, IP address, etc..
- Allows the user to set a callback on specific messages from the server, such as `turn_on_led`, `push_button`, etc..

User API

Library provide some basic methods for communication with server:

- `setAbilities(abilities)`
- `sendData(type, value)`
- `sendJson(topic, JSON)`
- `setCallback()`

Communication protocol



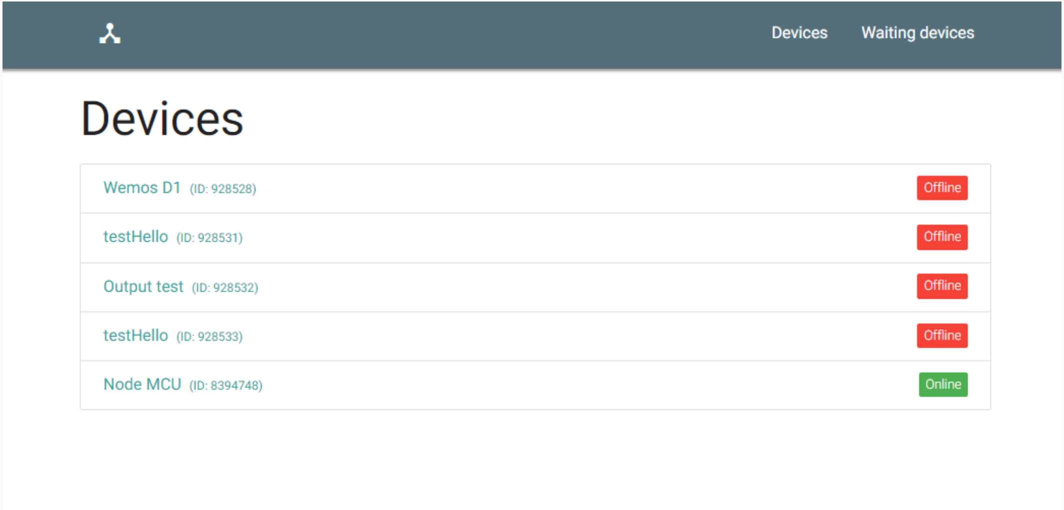
EspHub server

EspHub server

- Collecting data and telemetry from devices.
- Visualize data.
- Periodically send data to remote display.
- Broadcasts discovery messages.
- Notification when new device appear on network.

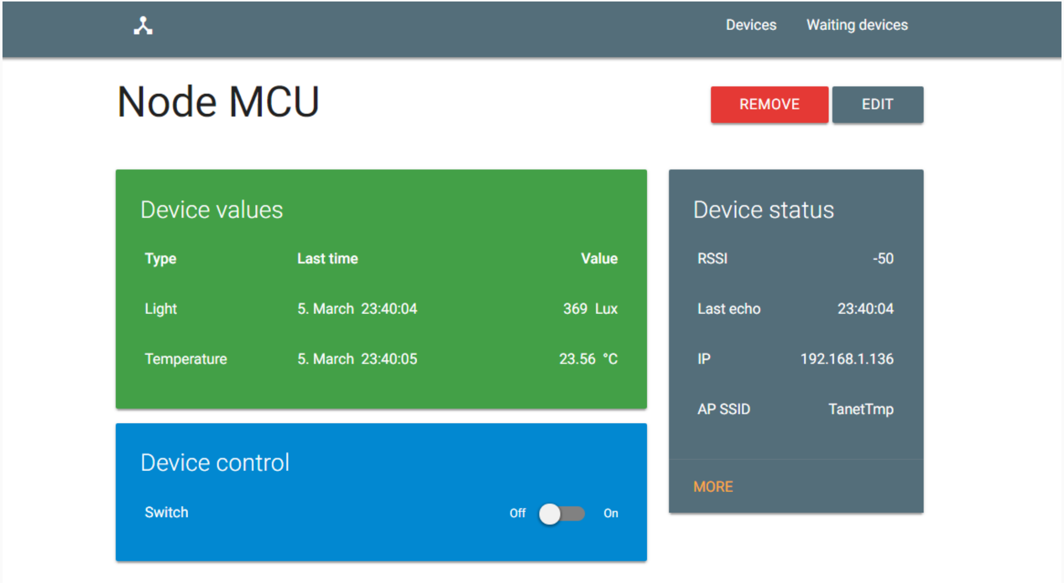
In background

- Python + Django framework + Click library.
- Installation with PIP.
- Simple CLI for testing and configuration.
- Internal scheduled tasks.



Devices

Wemos D1 (ID: 928528)	Offline
testHello (ID: 928531)	Offline
Output test (ID: 928532)	Offline
testHello (ID: 928533)	Offline
Node MCU (ID: 8394748)	Online



Node MCU

Device values

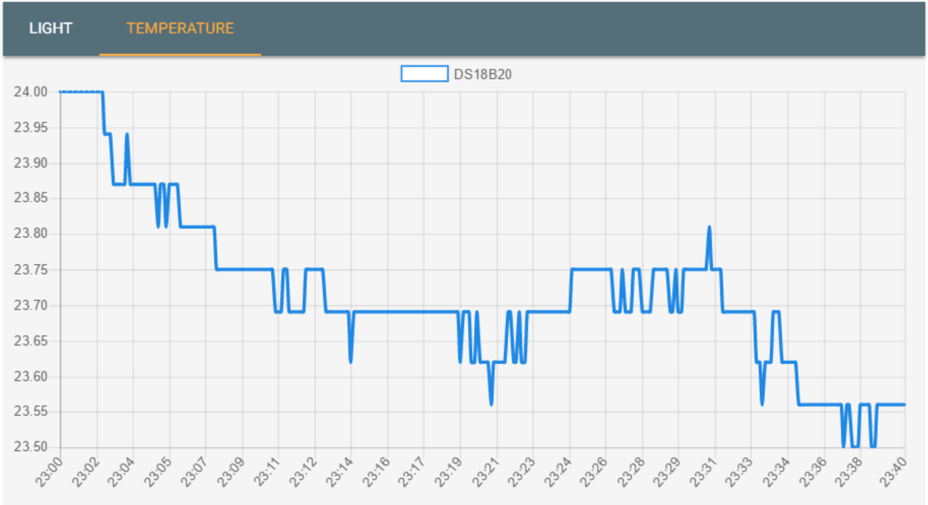
Type	Last time	Value
Light	5. March 23:40:04	369 Lux
Temperature	5. March 23:40:05	23.56 °C

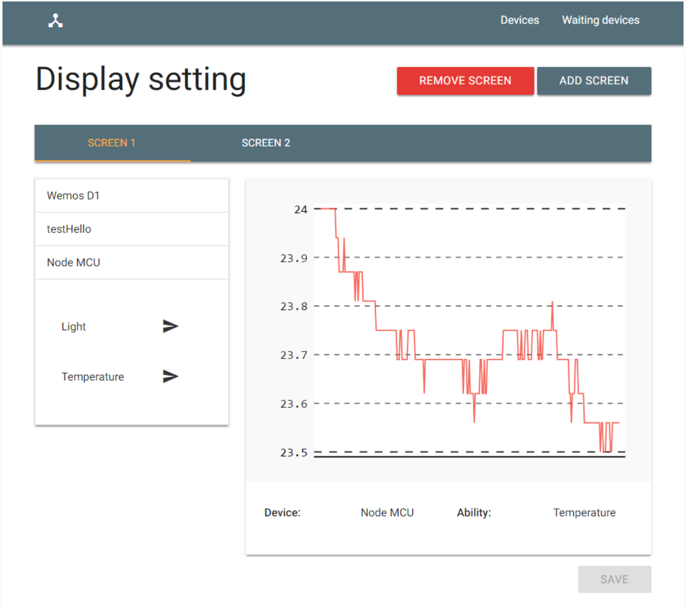
Device control

Switch Off ○ On

Device status

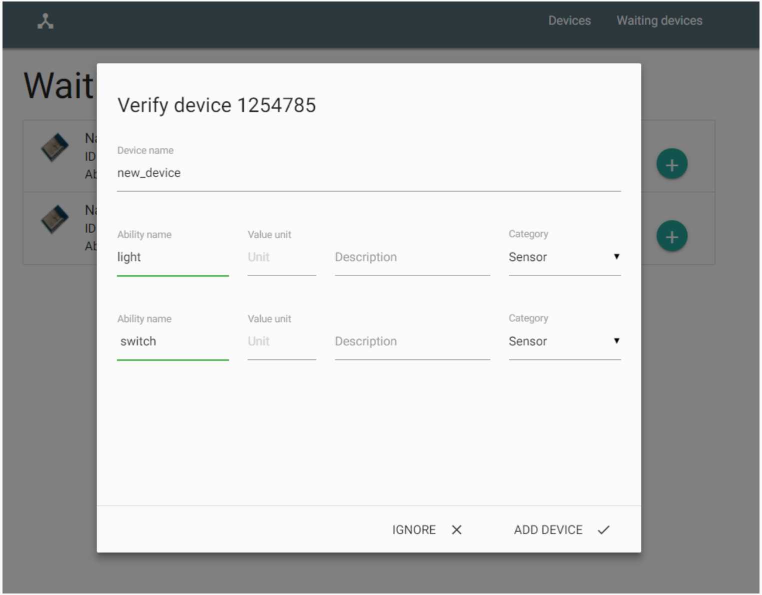
RSSI	-50
Last echo	23:40:04
IP	192.168.1.136
AP SSID	TanetTmp


LIGHT
TEMPERATURE
□ DS18B20
24.00
23.95
23.90
23.85
23.80
23.75
23.70
23.65
23.60
23.55
23.50
23:00
23:02
23:04
23:05
23:07
23:09
23:11
23:12
23:14
23:16
23:17
23:19
23:21
23:23
23:24
23:26
23:28
23:29
23:31
23:33
23:34
23:36
23:38
23:40



Device: Node MCU

Ability: Temperature



Verify device 1254785

Device name: new_device

Ability name: light
Value unit: Unit
Description
Category: Sensor ▾

Ability name: switch
Value unit: Unit
Description
Category: Sensor ▾

Future

Future

- Better visualization.
- Support custom display layout.
- Periodic tasks.
- Support ESP32.

Thanks for attention

github.com/TanasVlachopulos/EspHub