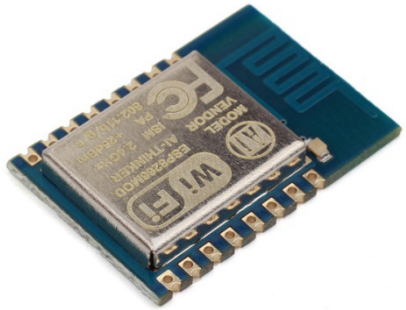
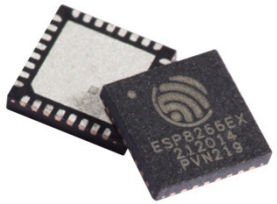
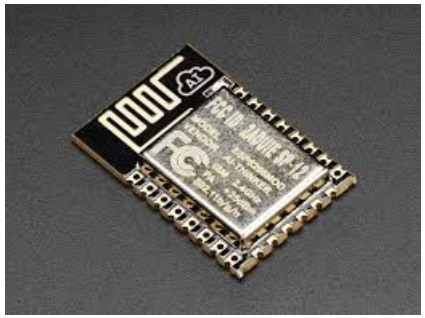






ESP8266

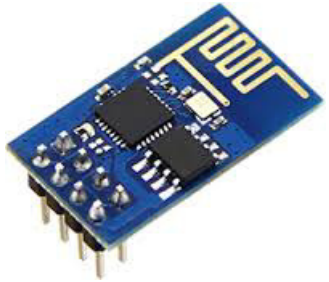
Une puce magique
pour le monde connecté ?

Rolf Ziegler, Oct. 2016

Agenda

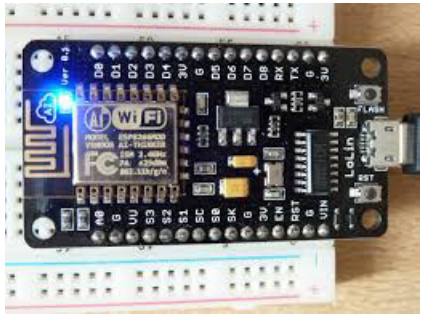
1. Circuit ESP8266, contenu, brochage, boot
2. Logiciels de programmation
3. Programmer ESP avec IDE Arduino
4. Modes sans fil (WIFI) ??
5. Fonctions de base WIFI (Arduino ESP)
6. Exemple pratique
7. Futur circuit ESP32
8. Q&R

1.1 Circuit ESP8266



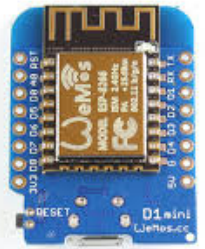
ESP-01

2x GPIO*
512kB Flash



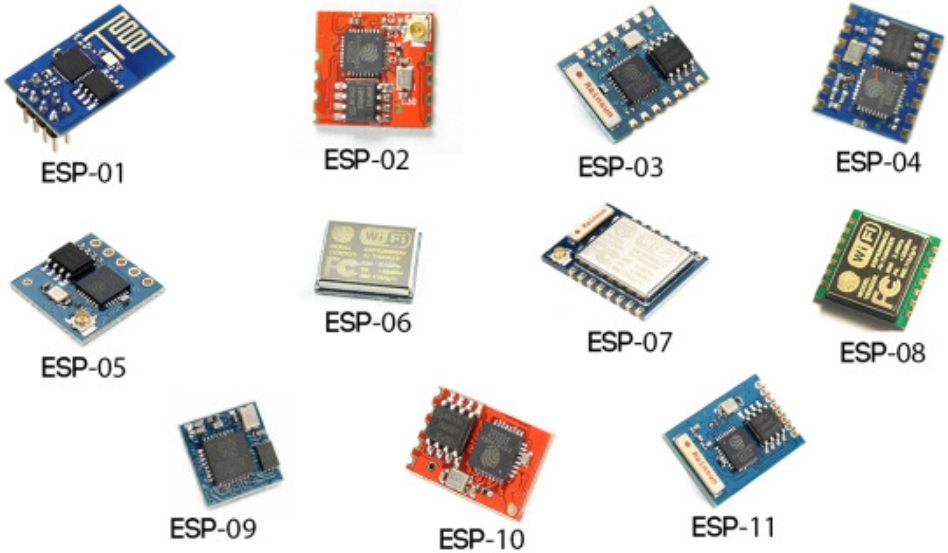
NodeMCU

11x GPIO
USB



WeMos D1

11x GPIO*
USB
Small Form Factor
4MB Flash **



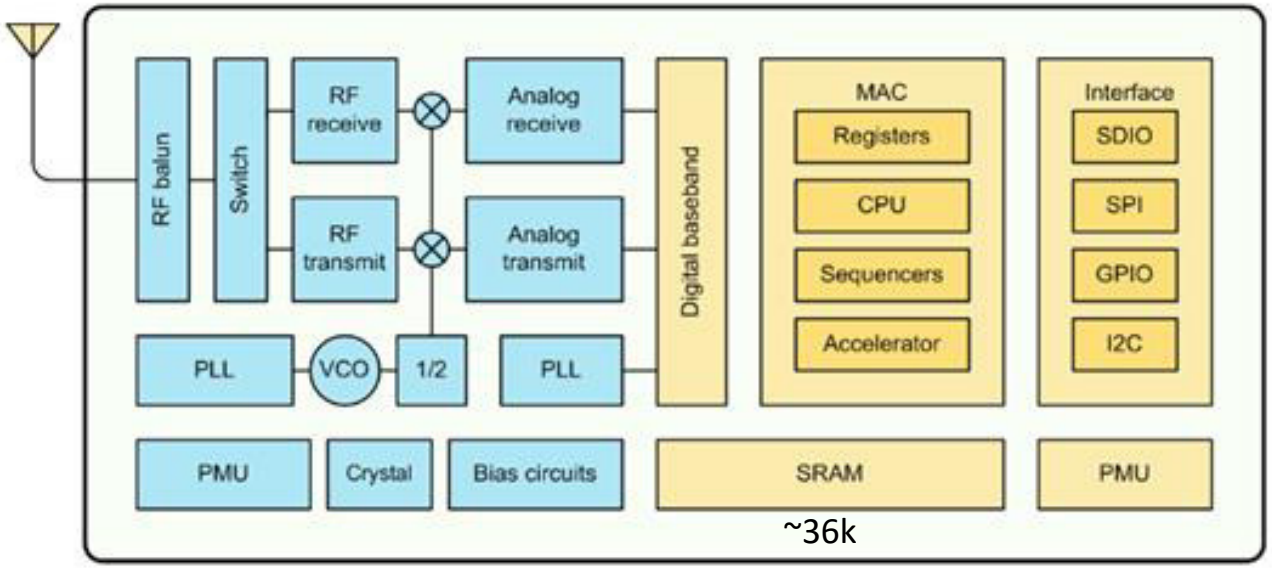
*Le chip a 12 GPIO (+
6 utilisés par la mémoire)

** Max 16MBits Flash

1.2 Circuit ESP8266

- Espressif Systems (Shanghai) Pte. Ltd.
  - 32 pin Ultra Low Power
  - WIFI 2.4GHz b,g,n WPA/WPA2
  - 80/160Mhz (pll)
  - 32 bits RTOS inclus (pas visible)
  - 80% de la puissance disponible (non wifi)

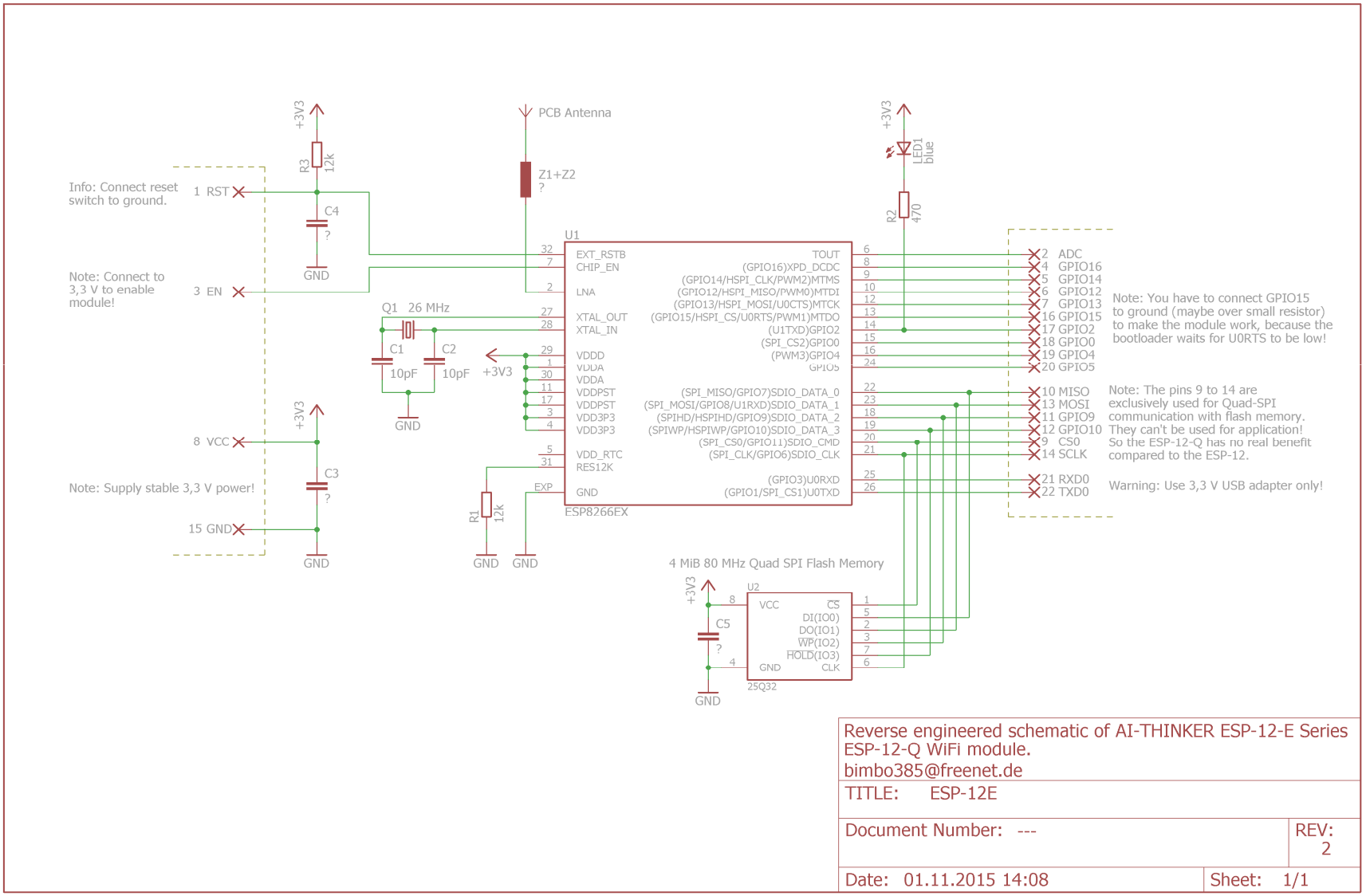
- Home Appliances
- Home Automation
- Smart Plug and lights
- Mesh Network
- Industrial Wireless Control
- Baby Monitors
- IP Cameras
- Sensor Networks
- Wearable Electronics



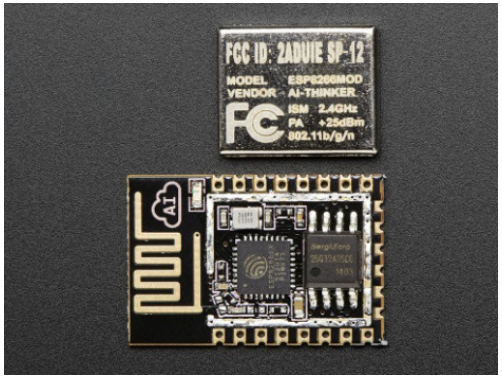
- Flash externe: 512b-16Mb

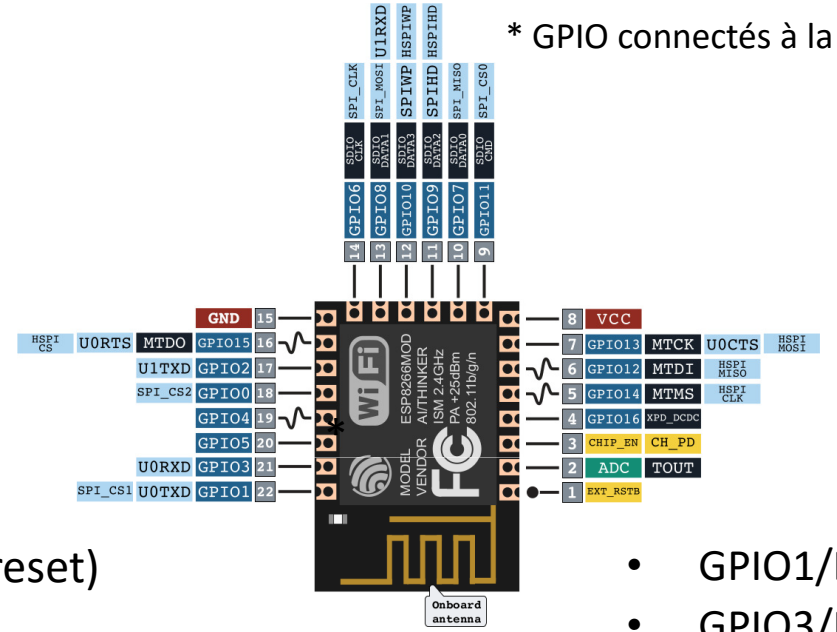
ARM Core: Tensilica LX106

ESP8266 Schéma



1.3 connexion de ESP8266





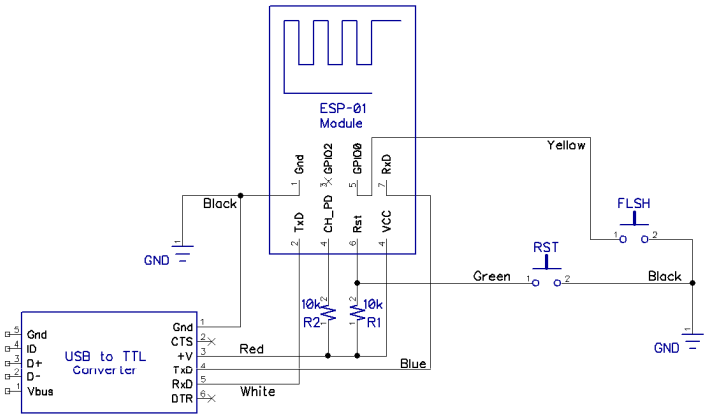
* GPIO connectés à la mémoire flash

- Reset (High=run, Low=reset)
- ADC
- CH_PD, Enable (High=on, Low=off)
- GPIO16/D0/USER/WEAK
- GPIO14/D5/SPI-SCK
- GPIO12/D6/MISO
- GPIO13/D7/MOSI

- GPIO1/D10 TX0
- GPIO3/D9 RX0
- GPIO5 /D1/SCL(I2C)
- GPIO4 /D2/SDA(I2C)
- GPIO0/D3/FLASH
- GPIO2/D4
- GPIO15/D8/TX2/SPI-CS

Si 6 pins en bas, elles sont connectées au SPI interne / en parallèle sur la mémoire

ESP8266 boot mode



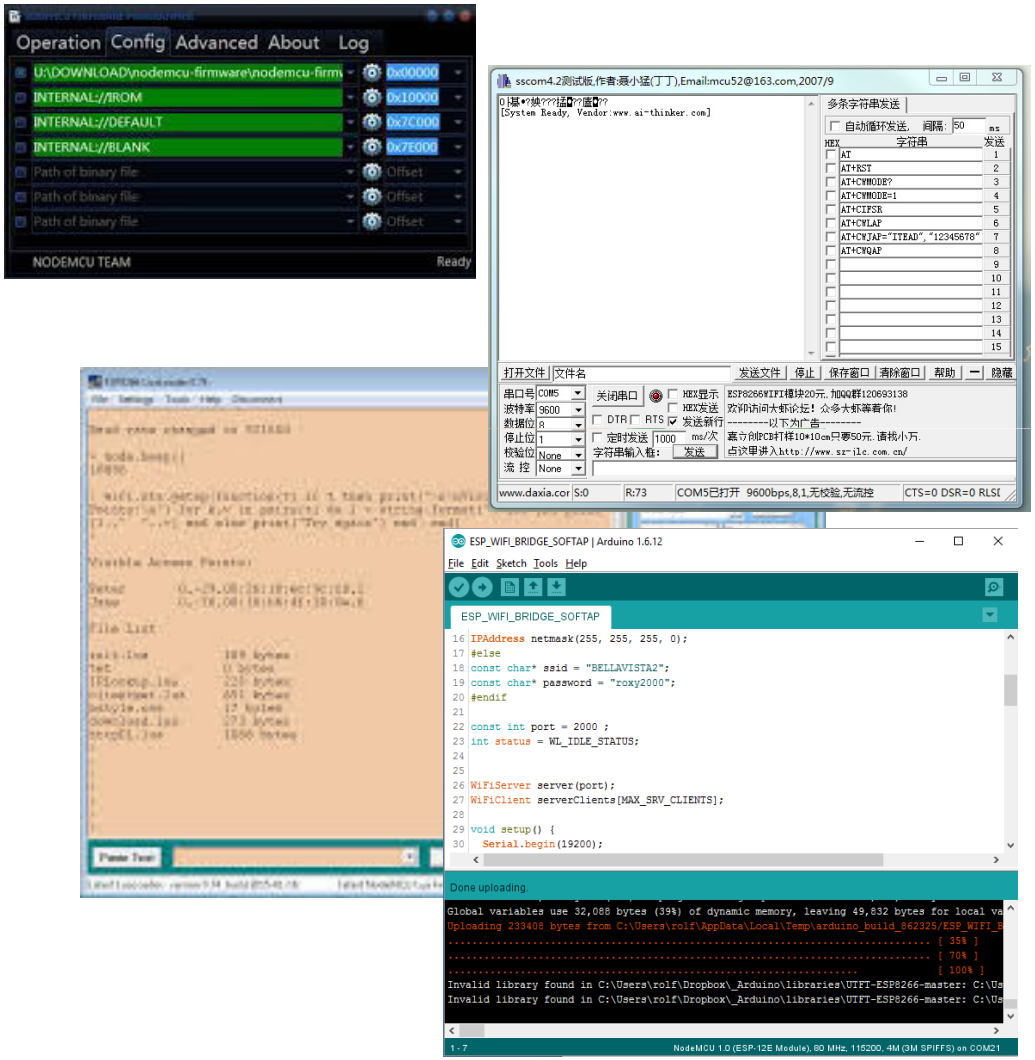
ESP-01 Connection Diagram

	GPIO 0	GPIO 2	GPIO 15
UART Download Mode (Programming)	0	1	0
Flash Startup (Normal)	1	1	0
SD-Card Boot	0	0	1

- Pour le mode normal,
  - GPIO15 est tenu à « 0 » par une résistance de 10k
  - GPIO 0 et GPIO 2 sont tenus à « 1 » par 10kOhm
- Mode Flash(par UART) GPIO0 est mis à 0 (Bouton,…)
- Seul avec une carte SD, GPIO2 doit être mise à 0 au démarrage (boot from SD card)

2. Logiciel de programmation

- Binary loader
- C code (linux)*
- Commandes AT
- Langage LUA**
- C code (Arduino)

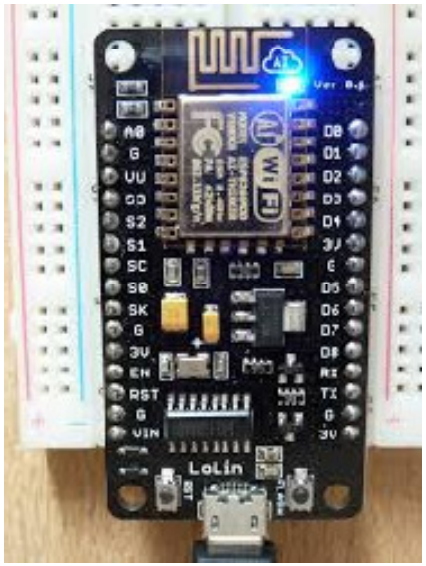


* a charger avec « binary loader »

**acronyme « love you always »

3. Programmer ESP avec IDE Arduino

- Un plug-in permet de faire l'extension Arduino-ESP8266, il suffit d'ajouter:
  http://arduino.esp8266.com/stable/package_esp8266com_index.json (40MB)

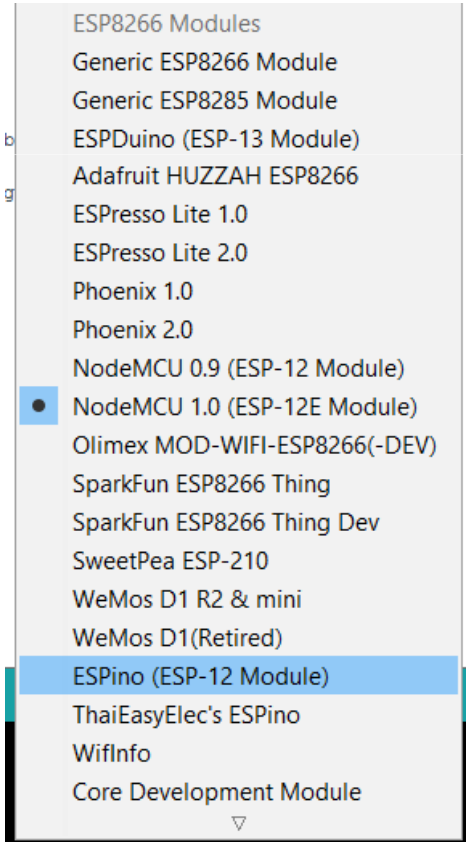


Débuter avec NodeMCU
-Grille 2.54
-Régulateur plus puissant
-Interface USB

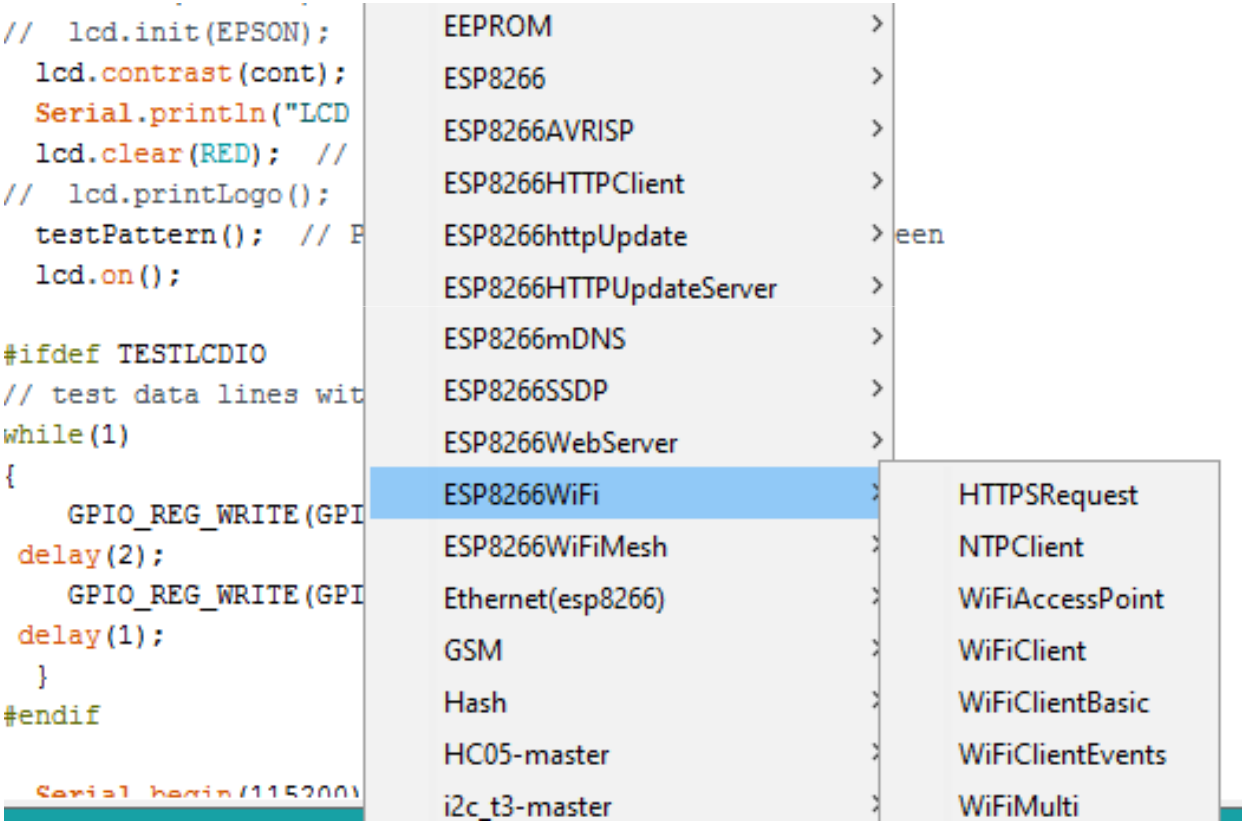
•De nombreuses **librairies** peuvent être ajoutée dans Arduino permettant le développement à partir d'exemples

•Avec l'extension, les cartes sont alors visibles **dans tools** sous Arduino

•Moyennant un interface USB (Node MCU/D1 ou avec FTDI) on peut alors flasher le ESP directement depuis l'environnement Arduino



3.1 Exemples Arduino



4. Modes sans fil (WIFI) ??
AP, SoftAP, STA

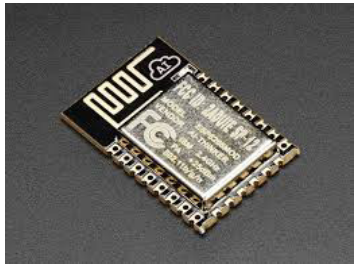
I2C/I2S
SPI
UART
GPIO

Niveau
Connexion (Wifi)

```
const char* ssid = « MICRONET";
const char* password = « 1234";
WiFi.begin(ssid, pass);
```

STA

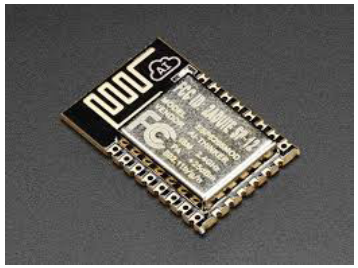
AP(server)



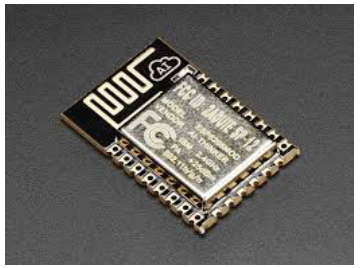
Niveau Protocol
(HTML, TCP, UDP)

```
Udp.begin(inPort)
Udp.read(buf,size);
….
```

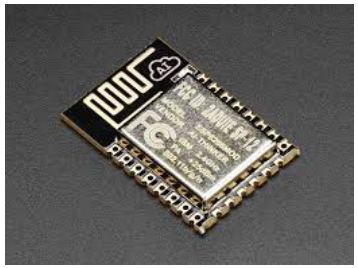
AP(server)



SOFTAP



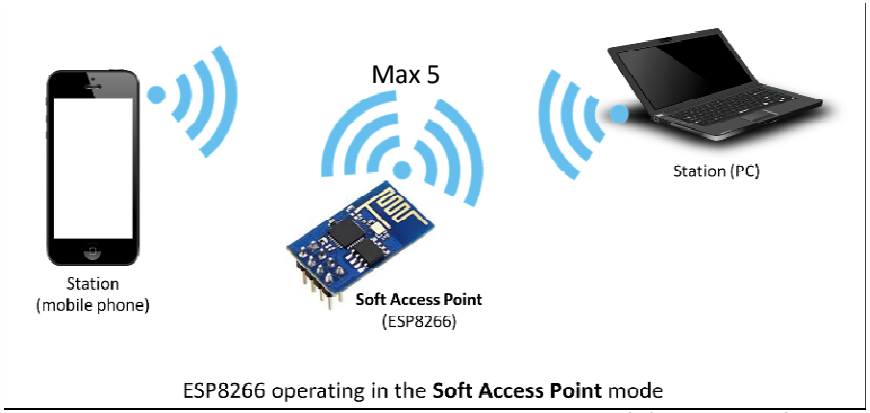
Client



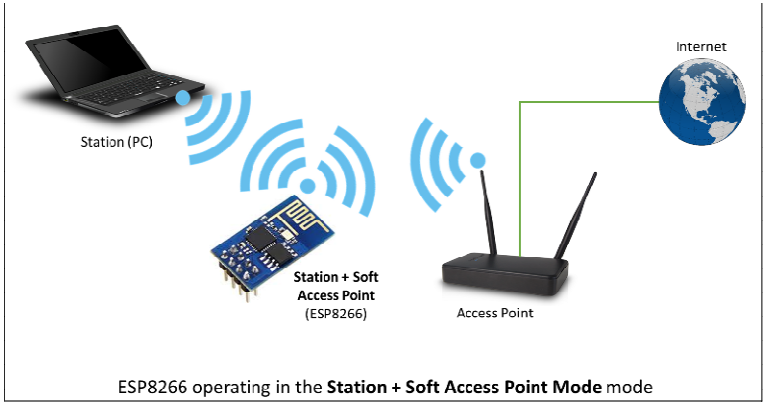
STA

Niveau Périphérie
I2C/I2S
SPI
UART
GPIO

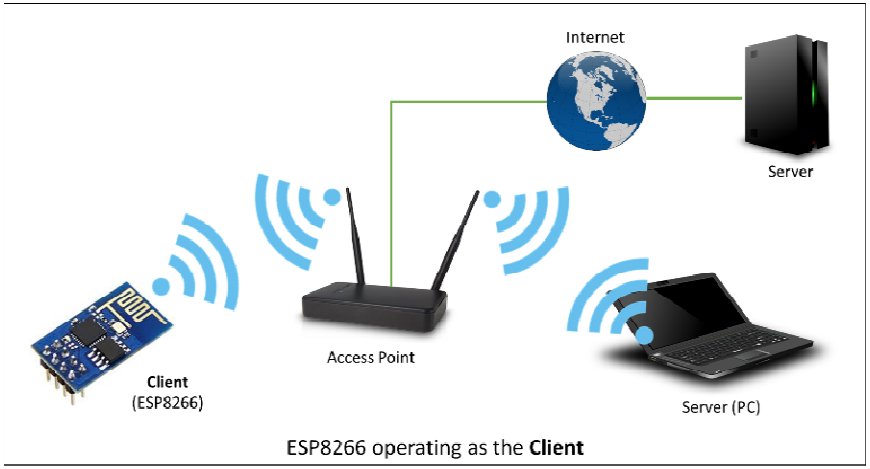
4.1Modes WIFI



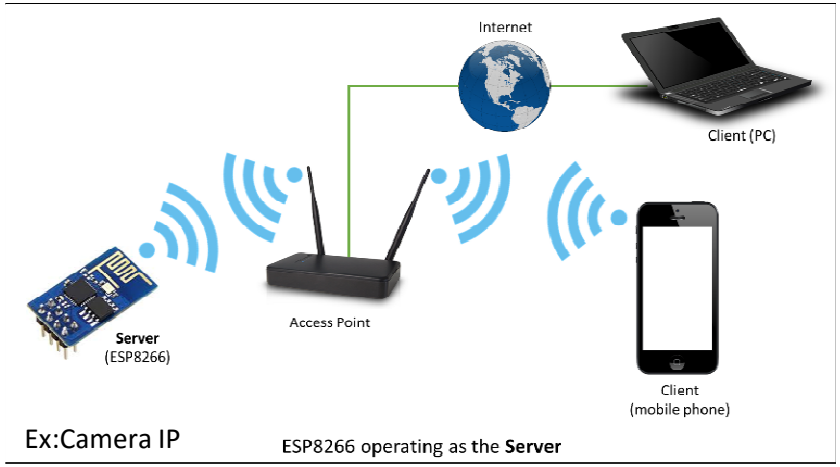
Connexion point-point sans PWD Ex: robot télécommandé



Connexion point-point sans PWD puis Station sur AP



Connexion sur AP, puis requête du réseau.
Ex. Horloge/Display Météo



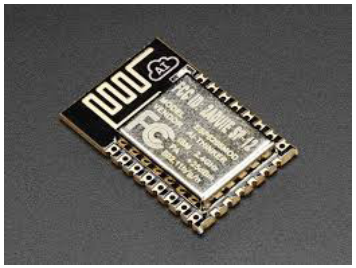
Connexion sur AP et répond aux requêtes du réseau
Les clients de l'AP peuvent se connecter sur le server

https://www.gitbook.com/book/krzychb/esp8266wifi-library/details

5. Niveau Application

Browser
WEB

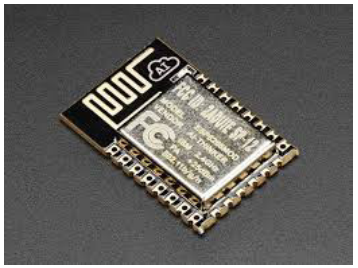
HTML



-On met le module ESP en mode « SoftAP » ou « Serveur * »
-On prépare des fonctions qui seront envoyées au browser appelant
-On attend un appel
-Dès l'appel entrant on envoie la page
-En boucle, on interagit avec l'appelant
-Les périphériques sont activés selon besoins

Applic UDP
Ex. OSC

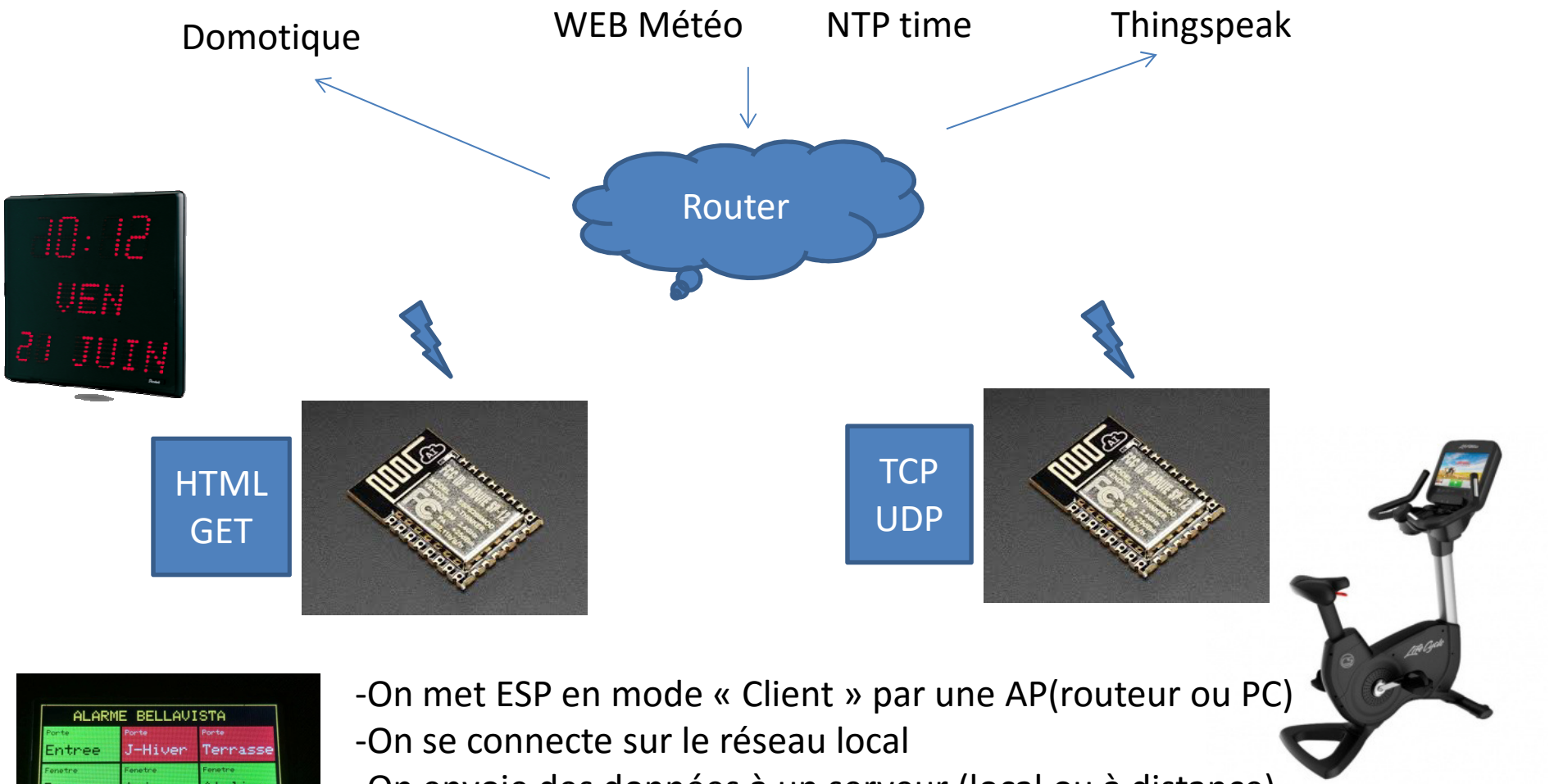
TCP
UDP



On ouvre une instance UDP
On se met en mode «SoftAP » ou « Server *»
On attend une connexion
-On lit instructions ou données entrantes
-On réagit sur SPI/I2C/UART etc
-On envoie les données vers l'application

* Mode server par un AP(Routeur)

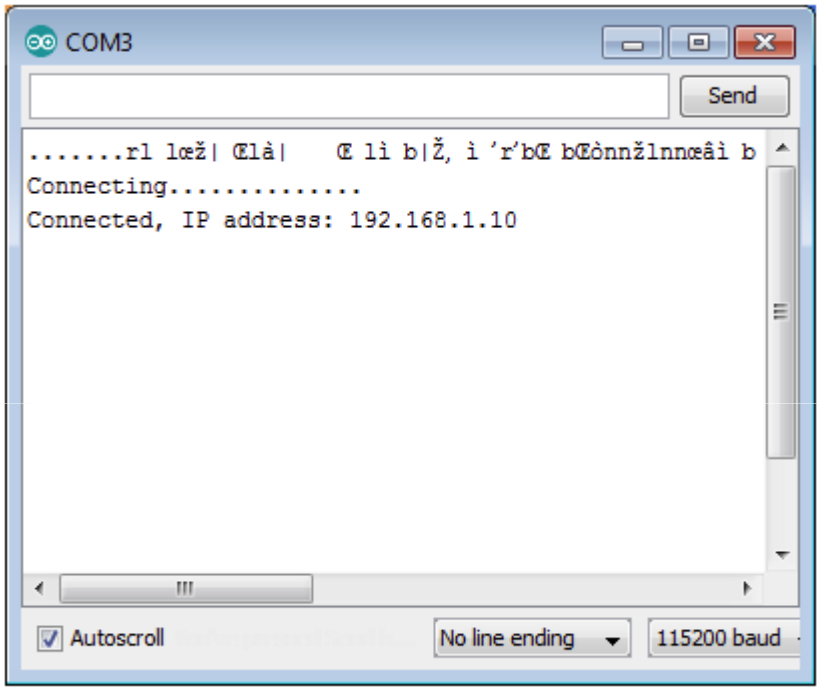
5.1 Niveau Application



-On met ESP en mode « Client » par une AP(routeur ou PC)
-On se connecte sur le réseau local
-On envoie des données à un serveur (local ou à distance)
-On envoie une demande à un serveur (avec ou sans identification)
-On reçoit une réponse (XML,…)

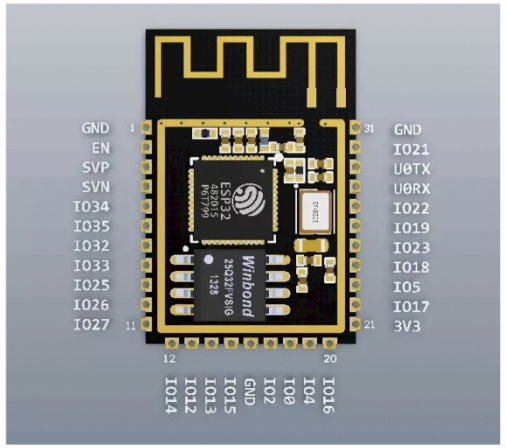
6. Librairie exemple de code

```
#include <ESP8266WiFi.h>

void setup()
{
  Serial.begin(115200);
  Serial.println();

  WiFi.begin("network-name", "pass-to-network");

  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println();

  Serial.print("Connected, IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {}
```



On voit ici la connexion sur un AP
-wifi.begin(….) se connecte sur un AP connu (SSID et PWD)
-puis attend que la connexion soit établie avant de continuer
! il manque le code à exécuter après.

7. Future produits ESP3212



- 12-bit SAR ADC 18 canaux
- 2 × convertisseur 8-bit D/A
- 10 × capteurs « touch »
- Capteur de température
- 4 × SPI, 2 × I2S, 2 × I2C, 3 × UART
- 1 host (SD/eMMC/SDIO), 1 slave (SDIO/SPI)
- Interface Ethernet MAC avec DMA dédié et support IEEE 1588
- CAN 2.0
- Interface InfraRouge (TX/RX)
- PWM Moteur, LED PWM up to 16 channels
- Capteur de Hall
- Ultra low power analog pre-amplifier

!!!!! Pas encore supporté dans l'environnement Arduino !!!!
Prix ~ $7.0

8. Questions/Réponses

Annexe A: Liens

https://nurdspace.nl/ESP8266#Technical_Overview

http://benlo.com/esp8266/esp8266QuickStart.html

https://github.com/esp8266

https://github.com/esp8266/Arduino

français

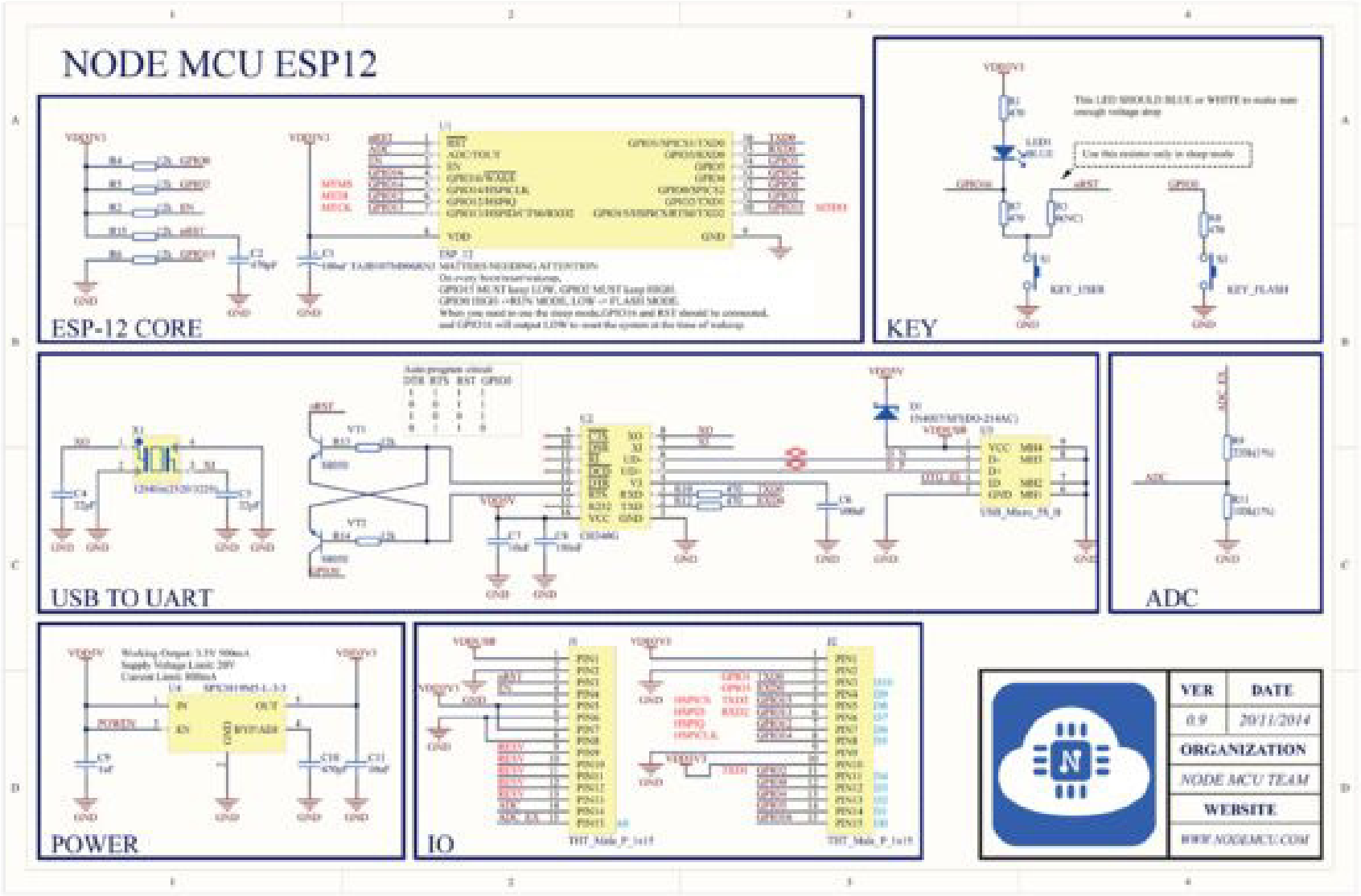
http://easydomoticz.com/forum/viewtopic.php?t=1840

https://www.gitbook.com/book/krzychb/esp8266wifi-library/details

Annexe B: Consommation

Mode	Min	Typical	Max	Units
802.11b, CCK 1Mbps, POUT=+19.5dBm		215		mA
802.11b, CCK 11Mbps, POUT=+18.5dBm		197		mA
802.11g, OFDM 54Mbps, POUT=+16dBm		145		mA
802.11n, MCS7, POUT =+14dBm		135		mA
802.11b, packet size of 1024 bytes, -80dBm		60		mA
802.11b, packet size of 1024 bytes, -70dBm		60		mA
802.11b, packet size of 1024 bytes, -65dBm		62		mA
Standby		0.9		uA
Deep sleep		10		mA
Saving mode DTIM 1		1.2		mA
Saving mode DTIM 3		0.86		mA
Shutdown		0.5		uA

Annexe C: Node MCU Schematic



Annexe D: AT commands

Commands	Description	Type	Set/Execute	Inquiry	test	Parameters	Examples
AT+RST	restart the module	basic	-	-	-	-	
AT+CWMODE	wifi mode	wifi	AT+CWMODE=<mode>	AT+CWMODE?	AT+CWMODE=?	1= Sta, 2= AP, 3=both	
AT+CWJAP	join the AP	wifi	AT+ CWJAP =<ssid>,<pwd >	AT+ CWJAP?	-	ssid = ssid, pwd = wifi password	
AT+CWLAP	list the AP	wifi	AT+CWLAP				
AT+CWQAP	quit the AP	wifi	AT+CWQAP	-	AT+CWQAP=?		
AT+ CWSAP	set the parameters of AP	wifi	AT+ CWSAP= <ssid>, <pwd>,<chl>, <ecn>	AT+ CWSAP?		ssid, pwd, chl = channel, ecn = encryption	Connect to your router: ; AT+CWJAP="YOURSSID","helloworld"; and check if connected: AT+CWJAP?
AT+ CIPSTATUS	get the connection status	TCP/IP	AT+ CIPSTATUS				
AT+CIPSTART	set up TCP or UDP connection	TCP/IP	1)single connection (+CIPMUX=0) AT+CIPSTART= <type>, <addr>,<port>; 2) multiple connection (+CIPMUX=1) AT+CIPSTART= <id> <type>,<addr>, <port>	-	AT+CIPSTART=?	id = 0-4, type = TCP/UDP, addr = IP address, port= port	Connect to another TCP server, set multiple connection first: AT+CIPMUX=1; connect: AT+CIPSTART=4,"TCP","X1.X2.X3.X4",9999
AT+CIPSEND	send data	TCP/IP	1)single connection(+CIPMUX=0) AT+CIPSEND= <length>; 2) multiple connection (+CIPMUX=1) AT+CIPSEND= <id>, <length>		AT+CIPSEND=?		send data: AT+CIPSEND=4,15 and then enter the data
AT+CIPCLOSE	close TCP or UDP connection	TCP/IP	AT+CIPCLOSE=<id> or AT+CIPCLOSE		AT+CIPCLOSE=?		
AT+CIFSR	Get IP address	TCP/IP	AT+CIFSR		AT+ CIFSR=?		
AT+ CIPMUX	set mutiple connection	TCP/IP	AT+ CIPMUX=<mode>	AT+ CIPMUX?		0 for single connection 1 for mutiple connection	
AT+ CIPSERVER	set as server	TCP/IP	AT+ CIPSERVER= <mode>[,<port>]			mode 0 to close server mode, mode 1 to open; port = port	turn on as a TCP server: AT+CIPSERVER=1,8888, check the self server IP address: AT+CIFSR=?

Annexe E: ESP3212 block diagram