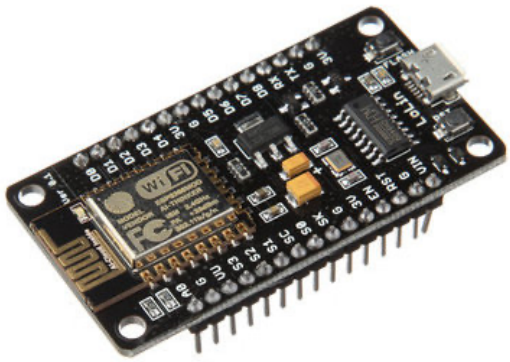
Le module NodeMCU Lolin ESP8266



NodeMCU a été créé peu de temps après l'apparition commerciale de l'ESP8266, lancé par Espressif Systems en décembre 2013. L'ESP8266 est un SoC (System on Chip) Wi-Fi intégrant un microprocesseur Tensilica Xtensa LX1062, souvent utilisé dans les applications IoT (Internet of Things). Le projet NodeMCU a démarré le 13 octobre 2014, lorsque Hong a publié le premier fichier de nodemcu-firmware sur GitHub. Deux mois plus tard, le projet a été étendu pour inclure une plate-forme matérielle ouverte (open-hardware) avec la publication du fichier à base du composant ESP8266 au format gerber, par le développeur Huang R. Le support du protocole de messagerie MQTT (Message Queuing Telemetry Transport) a ensuite été ajouté avec le port d'une bibliothèque du projet Contiki vers la plate-forme ESP8266. Dès lors NodeMCU a été en mesure de supporter le protocol MQTT IoT, à l'aide de Lua pour accéder au broker MQTT. Une autre mise à jour importante a été faite le 30 janvier 2015, avec le port de la bibliothèque d'affichage u8glib, permettant ainsi à une carte NodeMCU de gérer facilement des écrans LCD, OLED ou VGA.

Spécifications

Numéro de modèle	NodeMCU-LUA-V3 (CH340 ESP-12E)
Tension	3,3V
WiFi	Direct (P2P), soft-AP
Consommation courant	10uA~170mA
Mémoire flash	16MB max (512K normal)
Processeur	Tensilica L106 32-bit
Vitesse du processeur	80~160MHz
RAM	32K + 80K
GPIOs	17 (multiplexées avec d'autres fonctions)
Entrée analogique	1 avec résolution de 1024 niveau
Puissance de sortie RF	+19,5dBm (812.11b)
Protocol sans-fil supportés	802.11 b/g/n
Nombres de connexions simultanées TCP	5

Les broches de NodeMCU

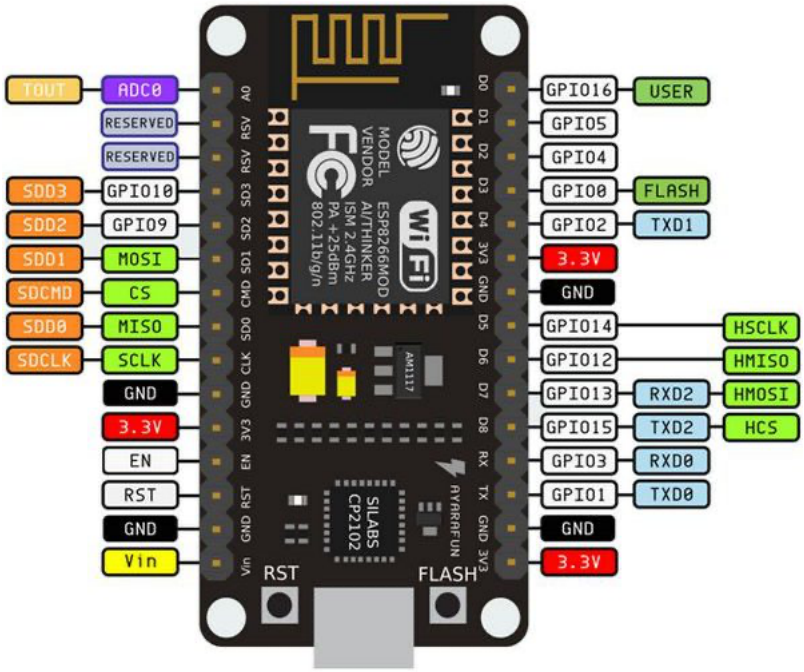
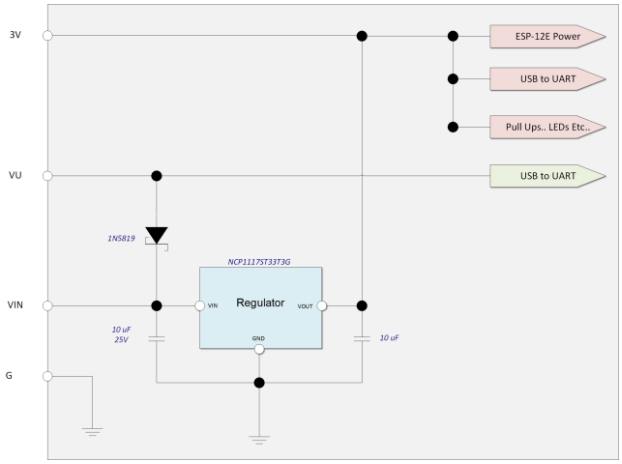


Schéma simplifié de l'alimentation NodeMCU ESP-12E

La diode Schottky signifie que vous pouvez fournir une alimentation VIN et USB simultanée en toute sécurité. Quelque chose qui est nécessaire pour ne pas détruire le port USB.

Gardez à l'esprit qu'il y a cinq broches de terre et trois broches de 3,3 V. Ils sont le même point électrique et ne sont donc pas inclus dans le schéma ci-dessous pour des raisons de simplicité.



Aperçu de vos options d'alimentation

En examinant le schéma, les options se résument à trois approches de base.

1. Utilisez l'alimentation USB
   Idéal pour charger des programmes, mais pas si bon si vous voulez réellement déconnecter votre projet de l'ordinateur.

2. Fournir 3,3 V directement
   C'est une option solide. Avec votre propre régulateur hors carte, vous pouvez fournir une source d'alimentation robuste pour votre appareil.

3. Fournir l'alimentation au VIN (3,3V~9V)
   Le régulateur est évalué jusqu'à 800mA. Dans certains cas, c'est plus que suffisant. Des précautions doivent être prises pour garder une trace de votre charge si votre intention est d'alimenter d'autres appareils à partir de la broche 3,3V.

Programmation

L'ESP8266 peut se programmer de plusieurs façons:

- Avec des scripts Lua, interprétés ou compilés, avec le firmware NodeMCU
- En C++, avec l'IDE Arduino
- En JavaScript, avec le firmware Espruino
- En MicroPython, avec le firmware MicroPython
- En C, avec le SDK d'Espressif ou avec le SDK esp-open-sdk3

Pour que cela soit possible, il faut bien entendu avoir installé l'IDE, installé le pilote Windows pour que l'USB arrive à communiquer avec la carte (CH340), installé dans l'IDE les modules et librairies qui vont permettre de compiler pour l'ESP8266. Tout cela est magnifiquement décrit sur le site https://hass.sebastienjean.me dans la section tutoriels.

ESP8266 and ESP32 serial bootloader utility

https://github.com/espressif/esptool

La mémoire

Les ESP8266 embarquent une plus ou moins grosse mémoire flash accessible en SPI. Cette mémoire peut être intégrée au processeur ou alors associée sur la carte NodeMCU comme mémoire flash externe.

Ce qui est intéressant à connaitre, c'est que la mémoire flash est structurée de la manière suivante :

- Un espace de stockage pour le firmware
- Un espace de stockage temporaire pour les mises à jour OTA (Over The Air) du firmware
- Un système de fichier SPIFFS
- Un emplacement EEPROM pour la sauvegarde de données par les programmes
- Un emplacement pour stocker la configuration du WIFI dans le cas de l'utilisation du SDK natif

Dans la carte NodeMCU v3 il y a 4M de mémoire, dont 3 peuvent être dédiés au système de fichier. Ce système de fichier peut être utilisé pour y stocker des données et des fichiers, pour un serveur web par exemple. Cependant n'y voyez pas l'équivalent d'un système de fichier moderne. Il n'y a pas de correction d'erreur et il n'y a pas d'arborescence de fichier (répertoires et sous répertoires), tout est au même niveau. Mais comme le caractère « / » est accepté dans un nom de fichier, vous pouvez stocker un fichier du nom de « /web/index.htm » si vous voulez avoir quelque chose de structuré. Attention cependant les noms de fichiers sont limités à 32 caractères, y compris le « '\0' » de fin de chaine (donc 31 caractères utiles).

La mémoire EEPROM est particulièrement intéressante car c'est dans cette dernière que pourront être sauvegardées de données persistantes pour nos programmes. Par exemple, si une variable de notre programme sert à mémoriser un mot de passe et que ce mot de passe peut être changer, s'il est stocké dans la mémoire EEPROM nous pourrons retrouver ce changement en cas de reboot ou de coupure électrique.