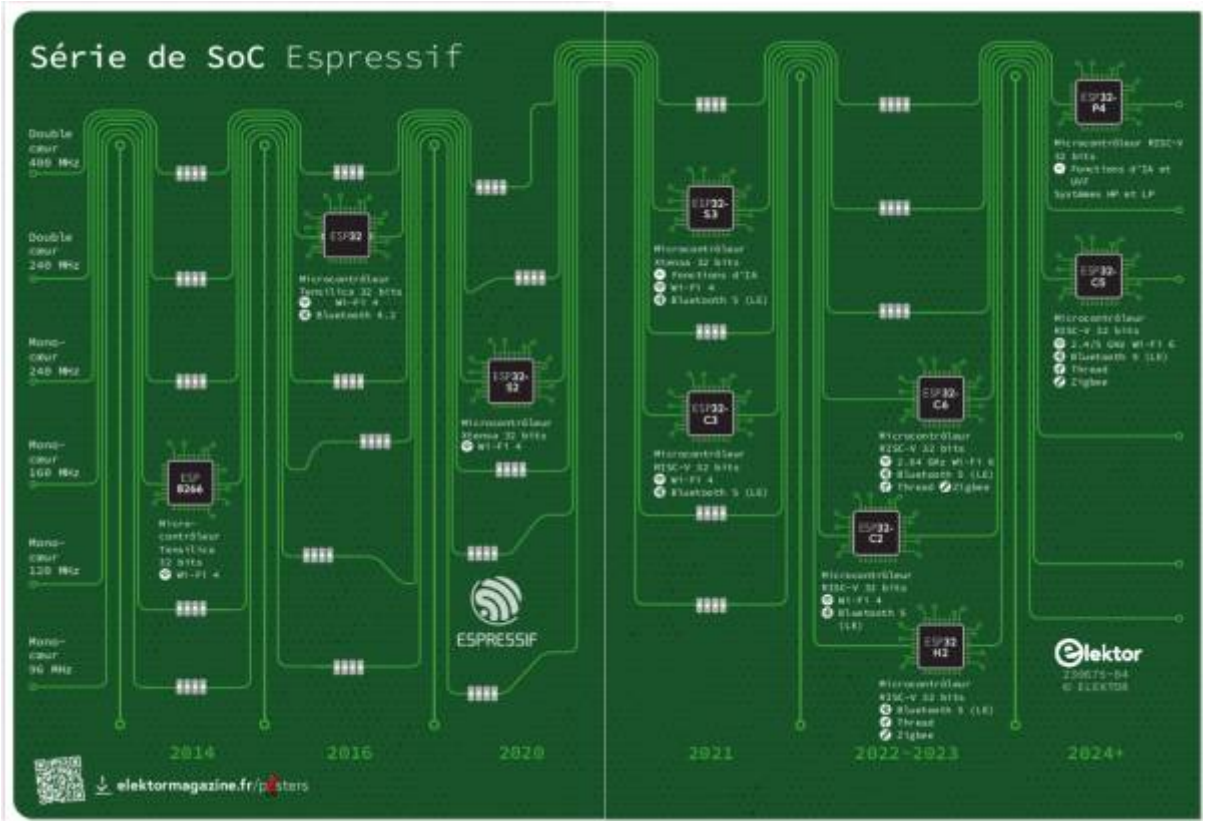
ESP32

Un site de simulation de l'ESP32 (entre autre...) https://wokwi.com/projects/new?lang=fr

ESP32 ---Arduino

Historique ESP32



Voici un tableau récapitulatif des principales différences techniques entre un ESP8266, un ESP32 , et un ESP32-C3 :

	ESP8266	**ESP32**	**ESP32-C3**
MCU	Xtensa Single-core 32-bit	Xtensa Dual-Core 32-bit	RISC-V 32-bits Single-Core
Wi-Fi	HT20 (802.11 b/g/n)	HT40 (802.11 b/g/n)	802.11.b b/g/n 2.4GHz
Bluetooth	Non	Oui (4.2 et BLE)	BLE 5.0
Frequence	80Mhz	160 a 240 Mhz	160 Mhz
SRAM	64kB	520kB + 10 pour RTC	400kB
DRAM/ROM	96kB -	328kB -	- 384kB
Flash	4mB	4mB – 16 mB	4mB
GPIO	17	36	22
Analog GPIO	1	18	2

Comparaison ESP32 et STM32

Specs/Board	STM32F103C8T6	ESP32	ESP8266	Arduino Nano
CPU	Arm Cortex-M3	Xtensa LX6	Xtensa L106	ATmega328P
CPU Core	1	2*	1	1
Architecture	32 bits	32 bits	32 bits	8 bits
CPU Frequency	72 MHz	160 MHz	80 MHz	16 MHz
Wireless	No	WiFi, Bluetooth	WiFi	No
RAM	20 KB	512 KB	160 KB	2 KB
Flash	64 KB/128kB	4 MB	4 MB	32 KB
GPIO Pins	37	36	17	14
SPI/I2C/I2S/UART	2/2/0/3	4/2/2/3	2/1/2/2	1/1/0/1
ADC Res/Ch	10 * 12-bit	6 * 12-bit	1 * 10-bit	6 * 10-bit
DAC Pins	0	2	0	0

ESP 01

ESP01

ESP8266

ESP32-C3 RISC-V

le successeur de l'ESP8266

L'ESP32-C3 est un SoC à microcontrôleur Wi-Fi et Bluetooth 5 (LE) monocœur, basé sur l'architecture RISC-V open source. Il trouve le bon équilibre entre puissance, capacités d'E/S et sécurité, offrant ainsi la solution optimale et rentable pour les appareils connectés. La disponibilité de la connectivité Wi-Fi et Bluetooth 5 (LE) facilite non seulement la configuration de l'appareil, mais facilite également une variété de cas d'utilisation basés sur la double connectivité.

Les fonctionnalités incluent les suivantes :

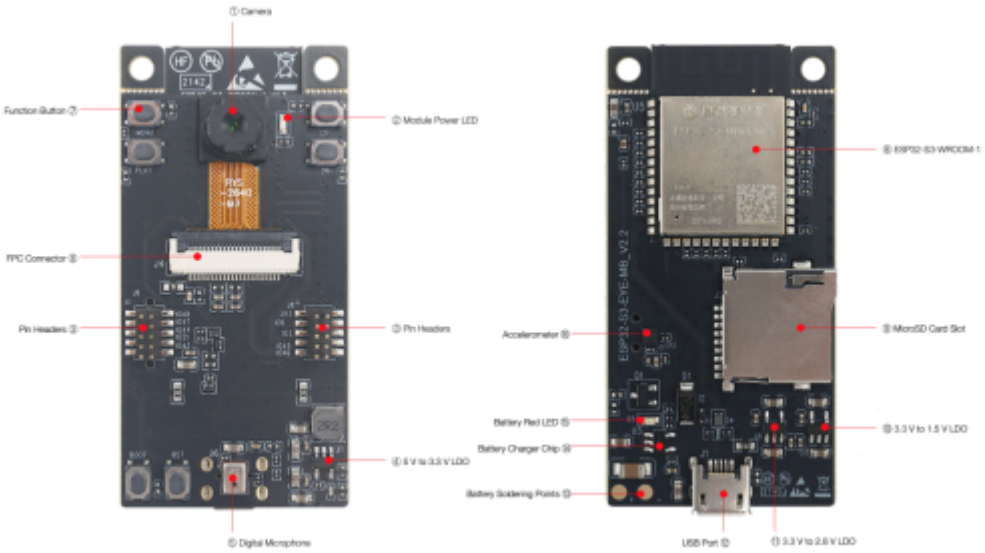
- Microcontrôleur RISC-V 32 bits avec une vitesse d'horloge maximale de 160 MHz
- 400 Ko de RAM interne
- 802.11b/g/n/e/i
- Un sous-système Bluetooth LE qui prend en charge les fonctionnalités de Bluetooth 5 et Bluetooth Mesh

Divers périphériques :

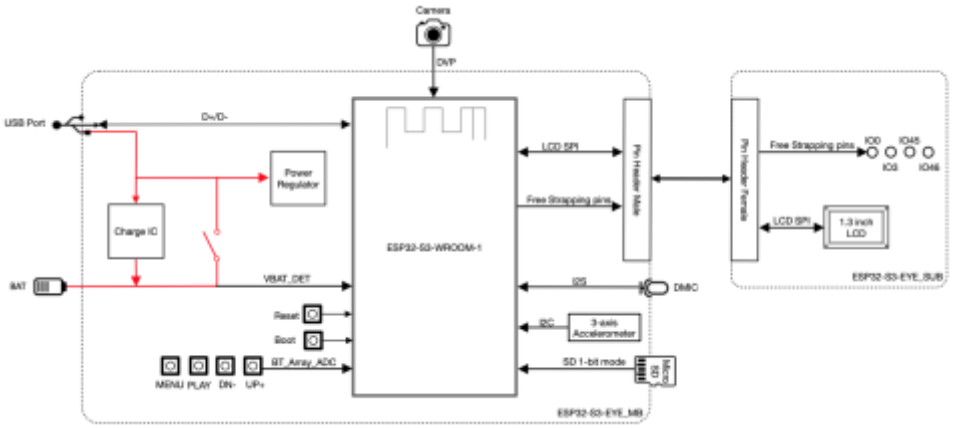
1. -CAN 12 bits avec jusqu'à 18 canaux
2. -TWAI compatible avec le bus CAN 2.0
3. -Capteur de température
4. -4x SPI
5. -2x I2S
6. -2x I2C
7. -3x UART
8. -LED PWM avec jusqu'à 16 canaux
9. -Accélération matérielle cryptographique (RNG, ECC, RSA, SHA-2, AES)

ESP32 S3 EYES

ESP32 EYES Guide github EN



- Diagramme interne de l'ESP32 EYES

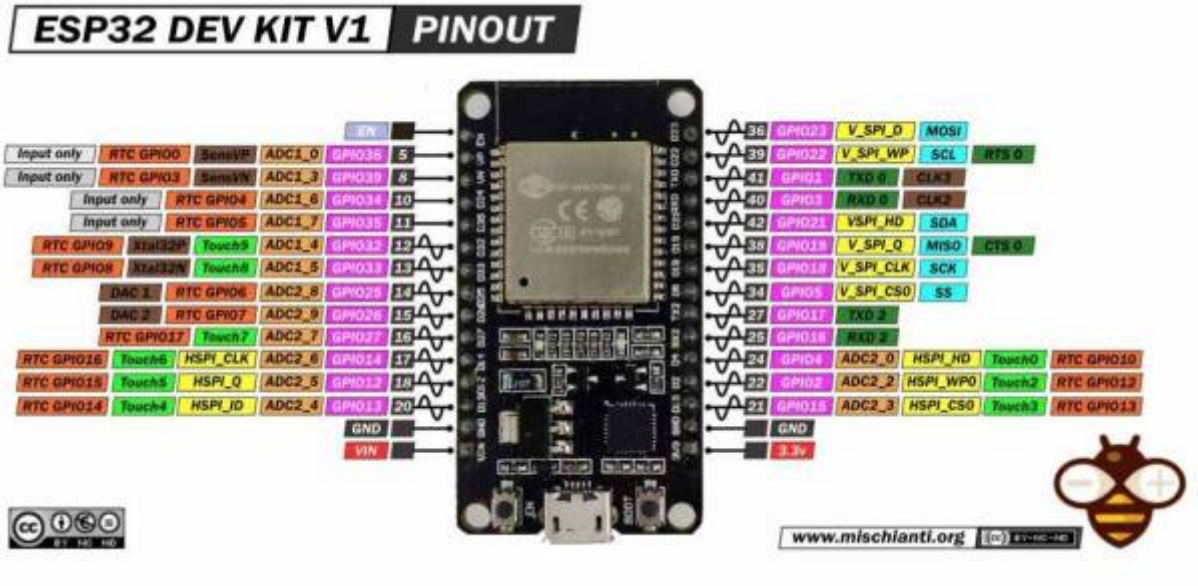


ESP32

- Créé par Espressif Systems, ESP32 est un système à faible coût et faible consommation d'énergie sur une série de puces (SoC) avec des fonctionnalités Bluetooth Wi-Fi et bi-mode!
- La famille ESP32 comprend les puces ESP32- D0WDQ6 (et ESP32- D0WD), ESP32- D2WD , ESP32- S0WD et le système en package (SiP) ESP32-PICO-D4.
- En son cœur, il y a un microprocesseur Tensilica Xtensa LX6 à un ou deux coeurs avec une fréquence d'horloge allant jusqu'à 240 MHz.
- L'ESP32 est hautement intégré avec des commutateurs d'antenne intégrés,
    1. - un balun RF,
    2. - un amplificateur de puissance,
    3. - un amplificateur de réception à faible bruit,
    4. - des filtres et des modules de gestion de l'alimentation.

- Conçu pour les appareils mobiles, les appareils électroniques portables et les applications IoT, ESP32 consomme très peu d'énergie grâce à des fonctions d'économie d'énergie telles que la synchronisation d'horloge à résolution fine, les modes d'alimentation multiples et la mise à l'échelle dynamique.

GPIO



Interruptions sur ESP32

Guide GPIO ESP32

L'I2C sur l'ESP32

L'ESP32 possède 2 bus I2C :

- Le bus I2C0 est celui qui est utilisé par défaut par les librairies Arduino. Il est relié aux pins GPIO22(SCL) et GPIO21(SDA) de l'ESP32. Il peut être utilisé sur n'importe quel pin de l'ESP32 quand vous utilisez la librairie Wire.h en précisant les pins avec la fonction Wire.begin(SDA_PIN, SCL_PIN)
- Le bus I2C1 peut aussi être utilisé sur n'importe quel pin (faire attention aux pins limités). Voici un exemple qui utilise le 2ème bus I2C:

I2C.ino

```
    #include <Wire.h>

    TwoWire I2C1 = TwoWire(1);

    void setup() {
      I2C1.begin(14,12,400000); // SDA pin 14, SCL pin 12, 400kHz
frequency
    }

    void loop() {
      I2C1.beginTransmission(0x42);
      I2C1.write(140);
      I2C1.endTransmission();
      delay(100);
    }
```

Esp32 LoraWan

Esp32 LoraWan

Liens

- ESP 32 EN
- Tuto ESP32 FR
- Kit Esp32 Pico-Board Elektor FR
- Micro python pour ESP32 EN
- ESP 32 - Arduino EN
- Programmer un microcontrôleur 32 bits en langage Python dans l'EDI Zerynth Studio
- Librairies arduino pour ESP32
- ESP32 - Arduino
- Un tutoriel pour débutant pour l'ESP32
- envoi-de-donnees-de-lesp32-esp8266-vers Google Sheets
- Tuto ESP32 MicroPython FR
- MOOC : fabriquer un objet connecté
- Tutos sur l 'ESP32 EN
- Esp32 Wiki

Videos

- ESP32 arduino and esp-idf installation
- Presentation ESP 32 : MicroPython, At-Commandes, IDE Arduino, ESP-IDF

Installation sur IDE Arduino

ESP8266

Une fois l'IDE installé, ouvrez le et rendez-vous dans les préférences. Dans la case “Additional Boards Manager URLs”, entrez l'adresse suivante :

http://arduino.esp8266.com/staging/package_esp8266com_index.json

ESP32

copier ce lien dans les préférences de l'IDE Arduino :

https://dl.espressif.com/dl/package_esp32_index.json

ESP32 et IDE Arduino

ESP32 ARDUINO

Resolution probleme sur Linux "import serial"

Processeurs:

```
  Processeur principal: microprocesseur LX6 32 bits Tensilica Xtensa
      Noyaux: 2 ou 1 (selon la variation)
      Toutes les puces de la série ESP32 sont à double cœur, à l'exception
de ESP32-S0WD, à cœur unique.
      Fréquence d'horloge: jusqu'à 240 MHz
      Performance: jusqu'à 600 DMIPS
  Co-processeur à très faible consommation: permet d'effectuer des
conversions, des calculs et des seuils de niveau ADC en sommeil profond.
```

Connectivité sans fil:

```
  Wi-Fi: 802.11 b / g / n / e / i (802.11n @ 2,4 GHz jusqu'à 150 Mbit / s)
  Bluetooth: v4.2 BR / EDR et Bluetooth Low Energy (BLE)
```

Reconnecter ESP32 au WIFI après une connexion perdue

Reconnexion WIFI sur ESP32

Mémoire:

```
  Mémoire interne:
      ROM: 448 KiB
      Pour le démarrage et les fonctions de base.
      SRAM: 520 KiB
      Pour les données et les instructions.
      RTC rapide SRAM: 8 KiB
      Pour le stockage de données et le processeur principal lors du
démarrage RTC à partir du mode veille profonde.
      RTC lente SRAM: 8 Ko
      Pour co-processeur accédant en mode veille profonde.
      eFuse: 1 Kibit
      Dont 256 bits sont utilisés pour le système (adresse MAC et
configuration de la puce) et les 768 bits restants sont réservés aux
applications client, y compris Flash-Encryption et Chip-ID.
      Flash intégré:
      Flash connecté en interne via IO16, IO17, SD_CMD, SD_CLK, SD_DATA_0 et
SD_DATA_1 sur ESP32-D2WD et ESP32-PICO-D4.
          0 Mio (puces ESP32-D0WDQ6, ESP32-D0WD et ESP32-S0WD)
          2 Mio (puce ESP32-D2WD)
          4 Mio (module ESP32-PICO-D4 SiP)
```

```
  Flash externe et SRAM: ESP32 prend en charge jusqu'à quatre flashs QSPI
externes de 16 Mio et des SRAM avec chiffrement matériel basé sur AES pour
protéger les programmes et les données des développeurs. L'ESP32 peut
accéder à la mémoire flash QSPI externe et à la mémoire SRAM via des caches
à grande vitesse.
      Jusqu'à 16 Mo de mémoire flash externe sont mappés en mémoire sur
l'espace de code du processeur, prenant en charge l'accès 8 bits, 16 bits et
32 □□bits. L'exécution du code est prise en charge.
      Jusqu'à 8 Mo de mémoire flash / SRAM externe sont mappés sur l'espace
de données du processeur, prenant en charge l'accès 8 bits, 16 bits et 32 □
bits. La lecture de données est prise en charge sur le flash et la SRAM.
L'écriture de données est prise en charge sur la mémoire SRAM.
  Les puces ESP32 avec flash intégré ne prennent pas en charge le mappage
d'adresses entre les flashs externes et les périphériques.
```

EEPROM et ESP32

C'est un peu différent de la classe EEPROM standard. Vous devez appeler **EEPROM.begin(size)**avant de commencer à lire ou à écrire, la taille étant le nombre d'octets que vous souhaitez utiliser. La taille peut être comprise entre 4 et 4 096 octets.

EEPROM.writen'écrit pas immédiatement dans la mémoire flash, mais vous devez appeler **EEPROM.commit()**chaque fois que vous souhaitez enregistrer les modifications dans la mémoire flash. **EEPROM.end()**s'engagera également et libérera la copie RAM du contenu de l'EEPROM.

La bibliothèque EEPROM utilise un secteur de flash situé juste après le système de fichiers intégré.

Trois exemples inclus.

Notez que le secteur doit être re-flashé chaque fois que les données EEPROM modifiées doivent être sauvegardées, donc épuisera la mémoire flash très rapidement même si de petites quantités de données sont écrites.

https://www.aranacorp.com/fr/utilisation-de-leeprom-avec-lesp32/

https://projetsdiy.fr/esp8266-comment-lire-ecrire-effacer-eeprom/

Enregistrer les données et les paramètres ESP32 en permanence

Entrée / sortie périphérique:

1. Interface périphérique riche avec DMA qui inclut la technologie capacitive tactile,
2. CAN (convertisseur analogique-numérique),
3. DAC (convertisseur numérique-analogique),
4. I²C (circuit inter-intégré),
5. UART (récepteur / émetteur asynchrone universel),
6. CAN 2.0 (Controller Area Network),
7. SPI (interface périphérique série),
8. I²S (son inter-CI intégré),

9. RMII (interface indépendante du support réduite),
10. PWM (modulation de largeur d'impulsion),
11. etc.

Sécurité:

- Fonctions de sécurité standard IEEE 802.11 toutes prises en charge, notamment WFA, WPA / WPA2 et WAPI
- Démarrage sécurisé
- Cryptage Flash
- OTP 1024 bits, jusqu'à 768 bits pour les clients
- Accélération matérielle cryptographique: AES, SHA-2, RSA, cryptographie à courbe elliptique (ECC),
- générateur de nombres aléatoires (RNG)

Bibliotheques pour ESP32

Librairies pour ESP 32

Sur Github : les librairies ESP32

Une librairie ESP32 à étudier

Enregistrer les données et les paramètres ESP32 en permanence

Alimentation ESP32

Pour alimenter votre kit de développement ESP32, vous avez trois options:

1. -Via le port USB.
2. -Utilisation d'une tension non régulée entre 5V et 12V, connectée aux broches 5V et GND. Cette tension est régulée à bord.
3. -Utilisation d'une tension régulée de 3,3 V, connectée aux broches 3,3 V et GND. Soyez très prudent avec cela: ne dépassez pas la limite de 3,3V, ou votre module ESP32 sera endommagé.

Attention : soyez très, très prudent de n'utiliser qu'une seule de ces options à la fois.

Par exemple, n'alimentez pas votre kit de développement ESP32 via la broche 5V en utilisant une entrée 10V alors qu'en même temps vous avez le module connecté à votre ordinateur via USB. Cela endommagera sûrement votre module, et peut-être même votre ordinateur.

Alimentation autonome pour ESP32

ESP32 ou esp8266 en mode autonome

esp32-alimente-par-batterie

Meilleur batterie pour ESP32

ESP 32 Alimentation autonome

Mode Veille profond sur ESP32

Guide pour réduire la consommation d'énergie de l'ESP32 de 95%

Les sleep modes de l’ESP32 et Programmation d’un ESP32 Basic Over The Air (OTA) avec l’IDE arduino

Mode deep sleep (Sommeil profond) sur ESP32

Produits à tester

Regulateur 3.3v à partir de batteries 1.2V

Réaliser un chargeur solaire

alimentation-autonome-de-petits-projets-electroniques

Chargeur batterie Lithium

Tutoriel ESP32 Deep Sleep

PCF8574 Multiplexeur IO

PCF8574

ESP NOW

ESP NOW

RIOT : OS pour l'internet des objets

RIOT

MQTT Protocole Iot

MQTT

MQTT et ESP32

FREERTOS : OS pour ESP32

FREERTOS

Objets connectes

Debuter projet Objets connecté

Interface Web pour ESP32

Les_reseaux_informatiques_par_la_pratique_ed1_v1.pdf Demande au Castellab pour l'emprunter

Station Météo ESP32

Station météo bricolage avec ESP32

Construire-un-pluviometre-usb

ESP32 arrosage automatique

arrossage_automatique

Tasmota

Tasmota

Projets en cours sur ESP32/ESP8266

ESP32/ESP8266 projets en cours

Esp32 cam

Camera de surveillance ESP32

Esp32 cam avec envoi SMS

esp32-cam-avec-arduino

5 astuces pour ESP32-CAM

ESP32 cam

ESP32-CAM: gestion à distance de la carte SD Programme .ino : esp32cam SD

Caméra de surveillance avec MAIL avec l'Esp32cam

Esp32 cam Github

ESP32 et telegram

Esp32 et Telegram

Propositon formation ESP32

Cours ESP 32

M5stack

M5stack

Iot-Cricket

Iot-Cricket

Protocole Iot : MQTT

MQTT

Micropython pour ESP32

micropython pour ESP32 FR

Doc FR en PDF MicroPython et ESP32

ESP32 SmartHome

Esp32 SmartHome

ESP32 Travaux pratiques

* TP ESP32 Tasmota MQTT Nodered