

Adaptateur USB vers ESP-01



Généralités

Cet [adaptateur USB vers ESP-01 avec puce CH340](#) permet d'accéder facilement au circuit ESP-01 via le port USB d'un ordinateur. L'adaptateur dispose également d'un régulateur 3,3 V.

Cela permet de recevoir des informations de l'ESP-01 ou d'[envoyer des commandes AT](#).

Acheter un [adaptateur USB vers ESP-01 avec puce CH340](#)

Activer le mode PROGRAMMATION

Il faut basculer l'**ESP-01** du **mode UART** en **mode FLASH** ou mode PROGRAMMATION , ce qui permet de programmer l'ESP8266. Cette bascule n'est pas réalisée logiciellement. Il faut réaliser la bascule en **mode PROGRAMMATION** électriquement.

```

esptool.FatalError: Failed to connect to ESP8266: Timed out waiting for packet header

Executable segment sizes:
IROM   : 234612      - code in flash          (default or ICACHE_FLASH_ATTR)
IRAM   : 26888 / 32768 - code in IRAM          (ICACHE_RAM_ATTR, ISRs...)
DATA   : 1252 )      - initialized variables (global, static) in RAM/HEAP
RODATA : 1376 ) / 81920 - constants              (global, static) in RAM/HEAP
BSS    : 25112 )      - zeroed variables      (global, static) in RAM/HEAP

Le croquis utilise 264128 octets (25%) de l'espace de stockage de programmes. Le maximum est de 1044464 octets.
Les variables globales utilisent 27740 octets (33%) de mémoire dynamique, ce qui laisse 54180 octets pour les variables locales. Le
esptool.py v2.8
Serial port /dev/ttyUSB0
Connecting.....Traceback (most recent call last):
  File "/home/cedric24c/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/tools/upload.py", line 65, in <module>
    esptool.main(cmdline)
  File "/home/cedric24c/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/tools/esptool/esptool.py", line 2890, in main
    esp.connect(args.before)
  File "/home/cedric24c/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/tools/esptool/esptool.py", line 483, in connect
    raise FatalError('Failed to connect to %s: %s' % (self.CHIP_NAME, last_error))
esptool.FatalError: Failed to connect to ESP8266: Timed out waiting for packet header
esptool.FatalError: Failed to connect to ESP8266: Timed out waiting for packet header

```

Sans cette bascule, l'IDE Arduino affiche des erreurs graves.

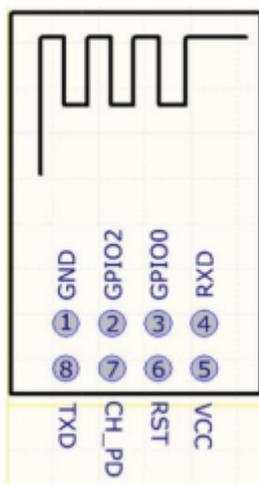
```

Executable segment sizes:
IROM   : 234612      - code in flash          (default or
ICACHE_FLASH_ATTR)
IRAM   : 26888 / 32768 - code in IRAM          (ICACHE_RAM_ATTR, ISRs...)
DATA   : 1252 )      - initialized variables (global, static) in
RAM/HEAP
RODATA : 1376 ) / 81920 - constants              (global, static) in
RAM/HEAP

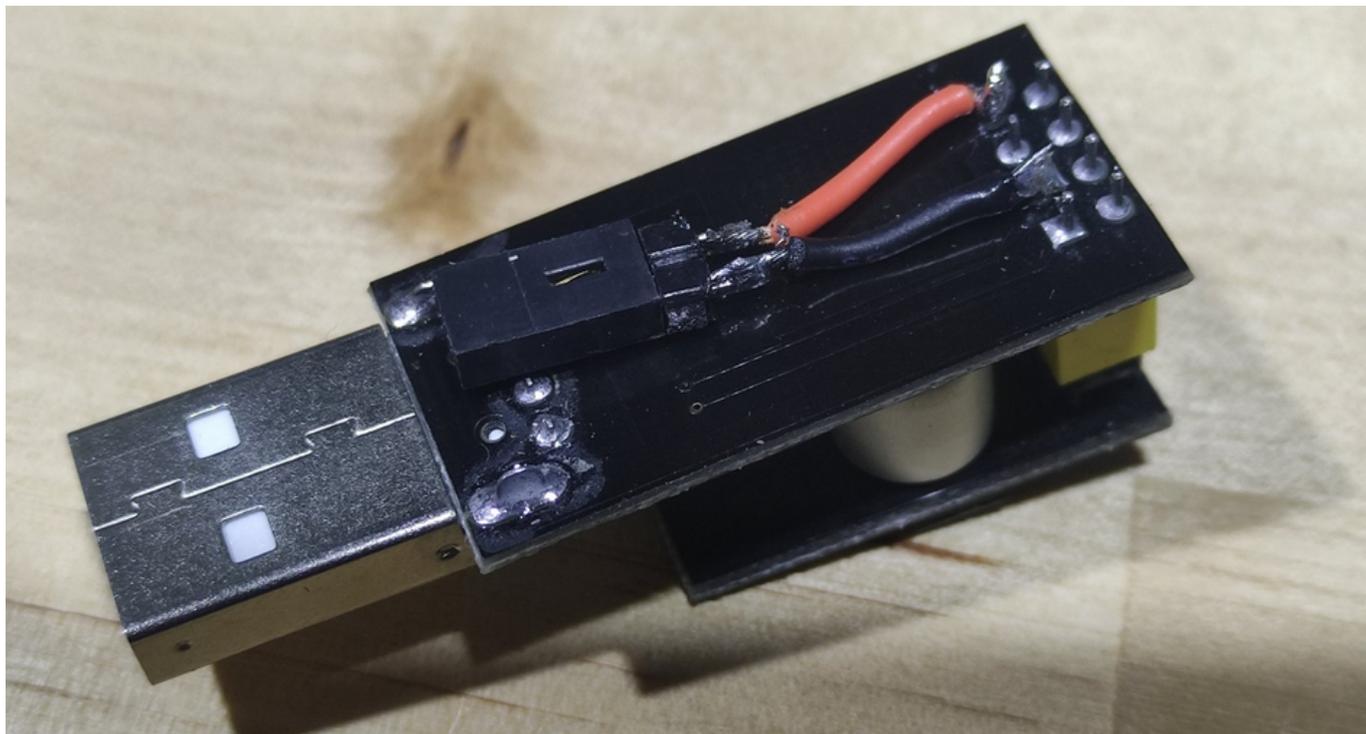
```

```
BSS      : 25112 )          - zeroed variables      (global, static) in
RAM/HEAP
Le croquis utilise 264128 octets (25%) de l'espace de stockage de
programmes. Le maximum est de 1044464 octets.
Les variables globales utilisent 27740 octets (33%) de mémoire dynamique, ce
qui laisse 54180 octets pour les variables locales. Le maximum est de 81920
octets.
esptool.py v2.8
Serial port /dev/ttyUSB0
Connecting.....
..__Traceback (most recent call last):
  File
"/home/cedric24c/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/tools/up
load.py", line 65, in <module>
  esptool.main(cmdline)
  File
"/home/cedric24c/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/tools/es
ptool/esptool.py", line 2890, in main
  esp.connect(args.before)
  File
"/home/cedric24c/.arduino15/packages/esp8266/hardware/esp8266/2.7.4/tools/es
ptool/esptool.py", line 483, in connect
    raise FatalError('Failed to connect to %s: %s' % (self.CHIP_NAME,
last_error))
esptool.FatalError: Failed to connect to ESP8266: Timed out waiting for
packet header
esptool.FatalError: Failed to connect to ESP8266: Timed out waiting for
packet header
```

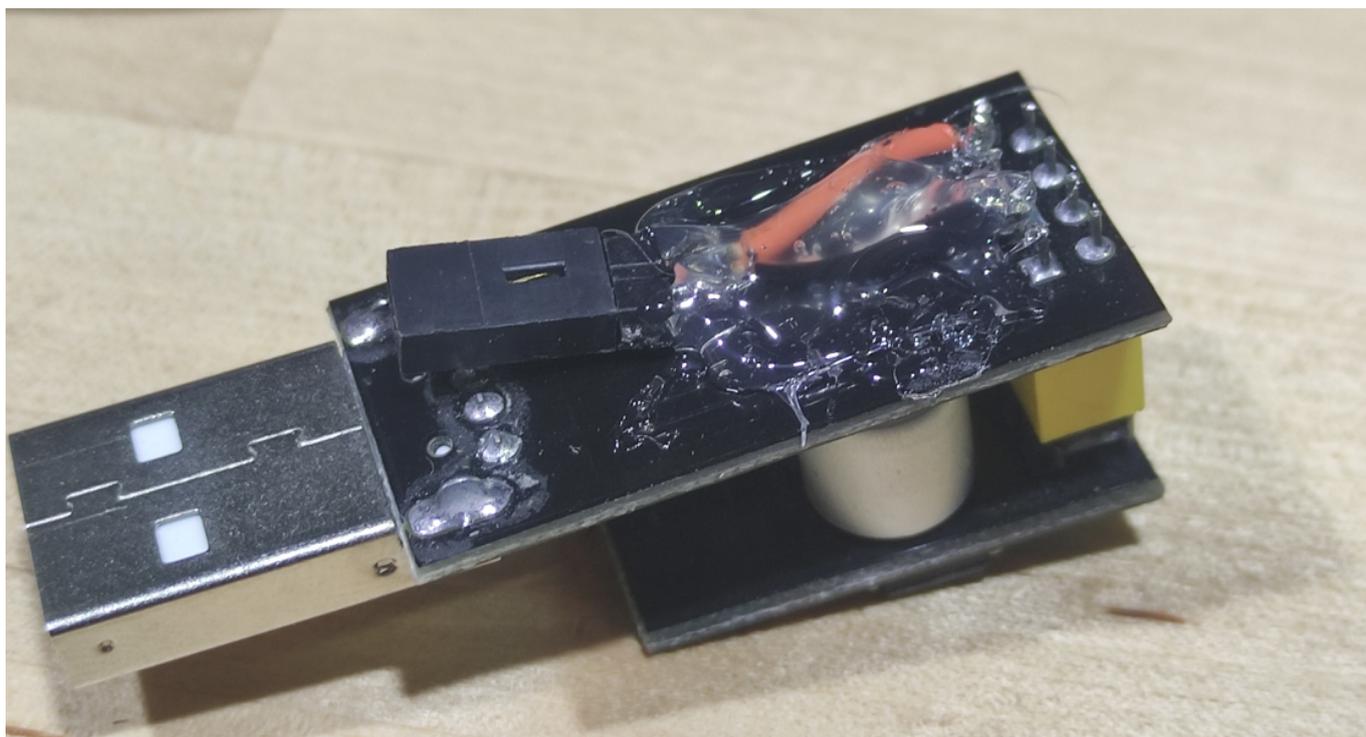
Certains [adapteurs USB vers ESP-01](#) ne sont pas équipés de cette bascule. Voici quelques modifications matérielles à effectuer pour rendre programmable l'ESP8266. Cette modification est à effectuer sur l'adaptateur. Il est nécessaire d'avoir de l'**étain** et un **fer à souder**. J'ai utilisé **deux fils** et **une broche de connexion** avec un **jumper**.



Il faut relier les broches **GPIO0** et **GND**.



J'ajoute un peu de colle blanche, avec un pistolet à colle afin de rigidifier l'ensemble et de ne pas tirer sur les soudures.



A partir de maintenant, vous pouvez téléverser votre programme.

```
Téléversement terminé
Executable segment sizes:
IRAM : 234612      - code in flash      (default or ICACHE_FLASH_ATTR)
IRAM : 26888 / 32768 - code in IRAM      (ICACHE_RAM_ATTR, ISRs...)
DATA : 1252       - initialized variables (global, static) in RAM/HEAP
RODATA : 1376    - constants          (global, static) in RAM/HEAP
BSS : 25112      - zeroed variables   (global, static) in RAM/HEAP
Le croquis utilise 264128 octets (25%) de l'espace de stockage de programmes. Le maximum est de 1044464 octets.
Les variables globales utilisent 27740 octets (33%) de mémoire dynamique, ce qui laisse 54180 octets pour les variables locales. Le maximum est de 81920 octets.
esptool.py v2.8
Serial port /dev/ttyUSB0
Connecting...
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: 18:fe:34:9a:36:06
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 1MB
Flash params set to 0x0020
Compressed 268288 bytes to 197383...
Wrote 268288 bytes (197383 compressed) at 0x00000000 in 17.5 seconds (effective 122.7 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

Le programme téléversé écrasera le code déjà chargé et notamment celui avec les commandes AT ([commandes Hayes](#)).

From: <https://www.abonnel.fr/> - **notes informatique & technologie**

Permanent link: <https://www.abonnel.fr/electronique/esp/adapteur-usb-esp01>

Last update: **2020/12/13 08:44**

