Introduction (v0.15)

This document describes the connecting of an NMS1250 MSX modem cartridge with an ESP8266 / ESP-01 wifi adapter.

The ESP8266 can be used in several ways:

1. C++ using the Arduino IDE
2. Lua using NodeMCU formware
3. Hayes AT-commands, as was used in old modems, using a dedicated firmware (most ESP-01 are by default configured to use AT-commands)

For this project I will use the NodeMCU firmware. This means that the ESP01 has to be flashed first with the NodeMCU firmware.

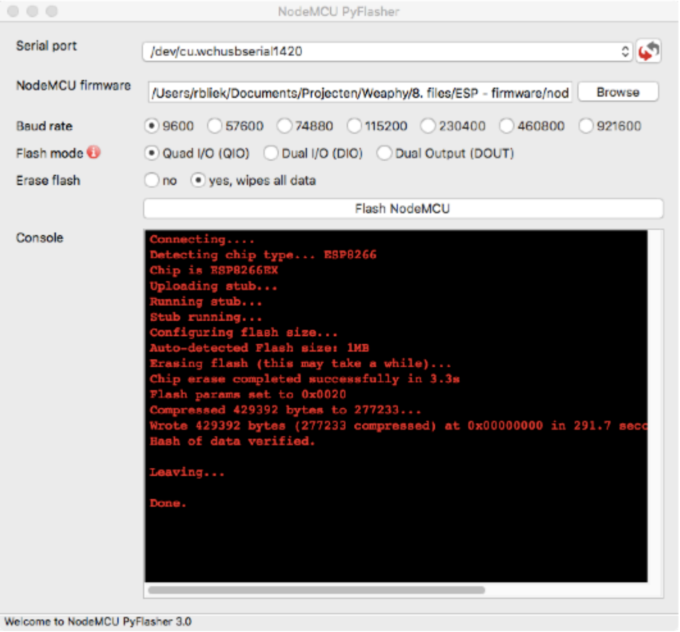
Flashing the ESP-01 wifi adapter

The ESP-01 wifi-adapter is by default configured to behave like an old modem, using the Hayes AT command set.

To flash the ESP-01 wifi-adapter, first a “NodeMCU-firmware” has to be built. This can be done using a free cloud service: <*https://nodemcu-build.com*>. You will be asked to fill in your email. This email address will be used to send the links for downloading your “firmware” after it has been built. You can select any module you would like to have included in the build. For this project the following modules are relevant: file, http, mdns, net, node, tmr, uart, wifi and tls (encryption).

You will get two links to the “NodeMCU firmware”: a ‘float' and an ‘integer’ version.

Flashing the firmware can be done via “command-line”, using a tool: esptool. And there is a more user friendly tool: NodeMCU PyFlasher:



Serial port: /dev/cu.wchusbserial1420

NodeMCU firmware: /Users/rbliek/Documents/Projecten/Weaphy/8. files/ESP - firmware/nod

Baud rate: ● 9600 ○ 57600 ○ 74880 ○ 115200 ○ 230400 ○ 460800 ○ 921600

Flash mode: ● Quad I/O (QIO) ○ Dual I/O (DIO) ○ Dual Output (DOUT)

Erase flash: ○ no ● yes, wipes all data

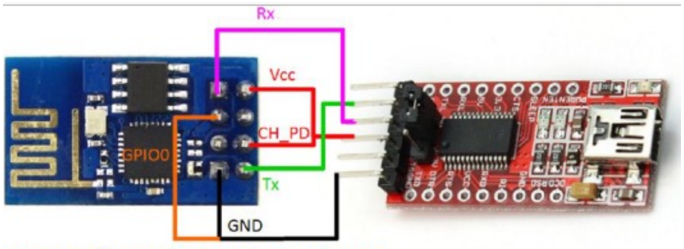
Console

```
Connecting....
Detecting chip type... ESP8266
Chip is ESP8266EX
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 1MB
Erasing flash (this may take a while)...
Chip erase completed successfully in 3.3s
Flash params set to 0x0020
Compressed 429392 bytes to 277233...
Wrote 429392 bytes (277233 compressed) at 0x00000000 in 291.7 seco
Hash of data verified.

Leaving...

Done.
```

Welcome to NodeMCU PyFlasher 3.0

```
esptool.py --port /dev/cu.wchusbserial1420 write_flash -fm qio 0x00000 nodemcu-master-10-modules-2017-12-12-16-22-29-float.bin
```

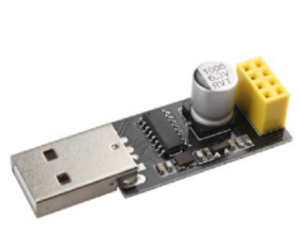
Please note: to“flash" the ESP-01 you have to do some soldering. Besides Vcc, GND, RX, TX and CH_PD you also need to connext GPIO0 to GND (GPIO2 can be left “not-connected”). The following picture shows the wiring:

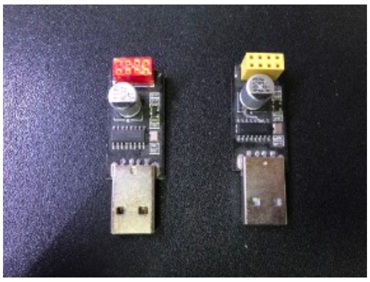


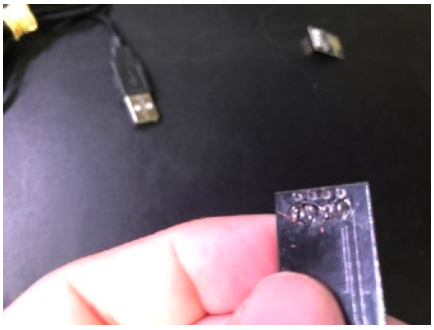
Connect GPIO pin to GND only to refash firmware

I used several solutions, but the easiest one is to buy a USB-serieel adapter and prepare it for flashing (A tip: buy two of those adapters, one prepared for flashing and another, left untouched, to upload Lua files… and paint the one used for flashing to avoid confusion)

esp8266-01 Programmer CH340

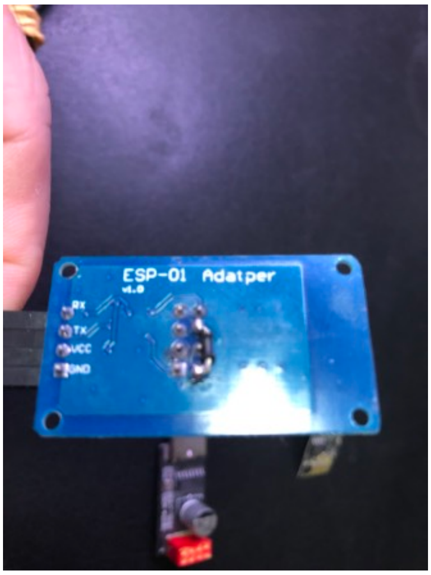




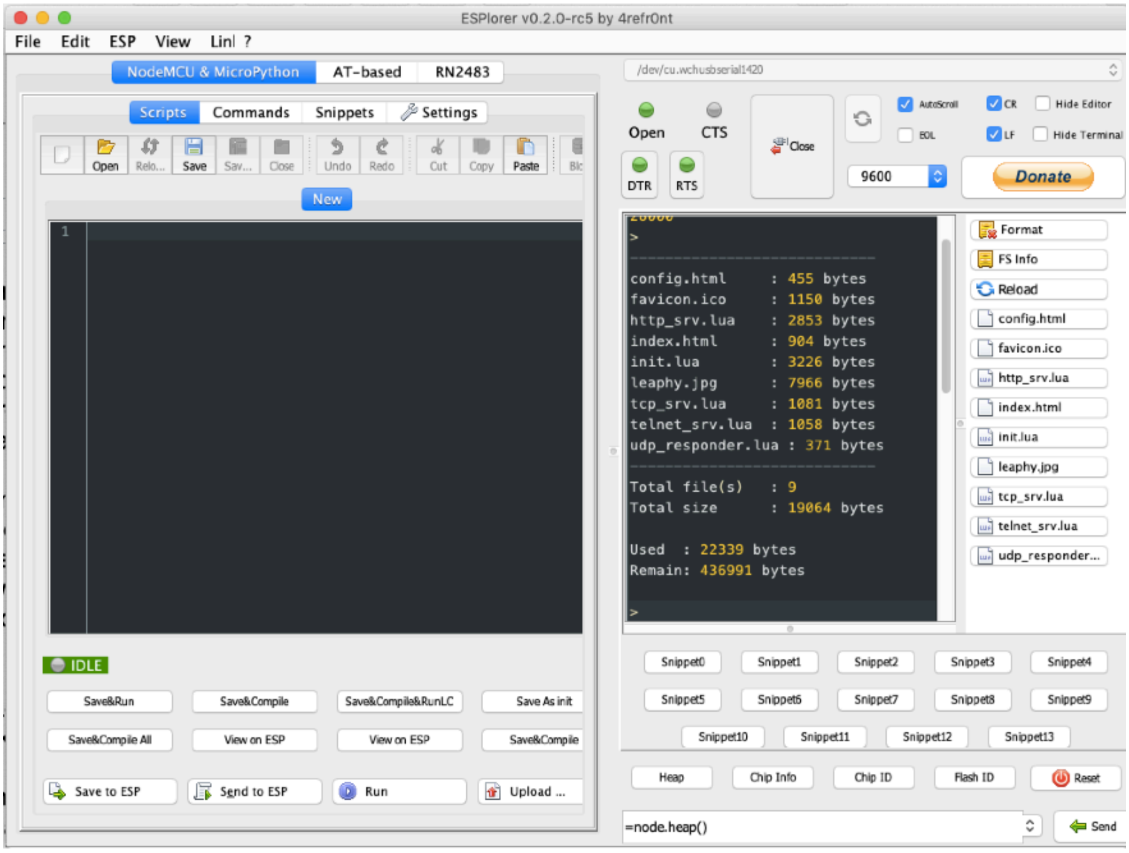


You need to solder a “jumper" between the first and third pin, bottom row. I painted this one red (see picture to the left).

If you use a 5V serial adapter, instead of the programmer, you need to solder a jumper as shown in the picture below (right row, between 2nd and 4th pin):



Sending Lua files to the ESP-01 wifi adapter
To upload Lua files to the ESP-01, you can use a tool called ESPlorer:



```
>
-----------------------------
config.html        : 455 bytes
favicon.ico        : 1150 bytes
http_srv.lua       : 2853 bytes
index.html         : 904 bytes
init.lua           : 3226 bytes
leaphy.jpg         : 7966 bytes
tcp_srv.lua        : 1081 bytes
telnet_srv.lua     : 1058 bytes
udp_responder.lua : 371 bytes
-----------------------------
Total file(s)      : 9
Total size         : 19064 bytes

Used   : 22339 bytes
Remain: 436991 bytes

>
```

```
=node.heap()
```

The ESP-01 must be connected to USB, in “normal mode” (use the "untouched" esp-usb adapter). Be sure to set the right bps-rate. For most ESP adapters this is 115200 bps out-of-the-box. For this project, after the Lua files have been uploaded and activated the rate will be changed to 9600 bps to ensure that the NMS1250 can keep up. So, if you see garbage in ESPlorer, change the bps rate between 9600 and 115200.

To upload the files to the ESP use the “upload-button“, select one or more files and confirm. At the right of the screen in ESPlorer there is a button “reload”. Press this button to see the actual status of the files on the ESP.

Appendix A - Software

- **NodeMCU-pyflasher**
  https://github.com/marcelstoer/nodemcu-pyflasher/releases.

- **ESPlorer**
  http://esp8266.ru/esplorer/.

- **esptool.py** (optional, as an alternative to NodeMCU-pyflasher)
  https://github.com/espressif/esptool