Hardware Hacking Village

Intro to NodeMCU on the ESP8266

Morgan

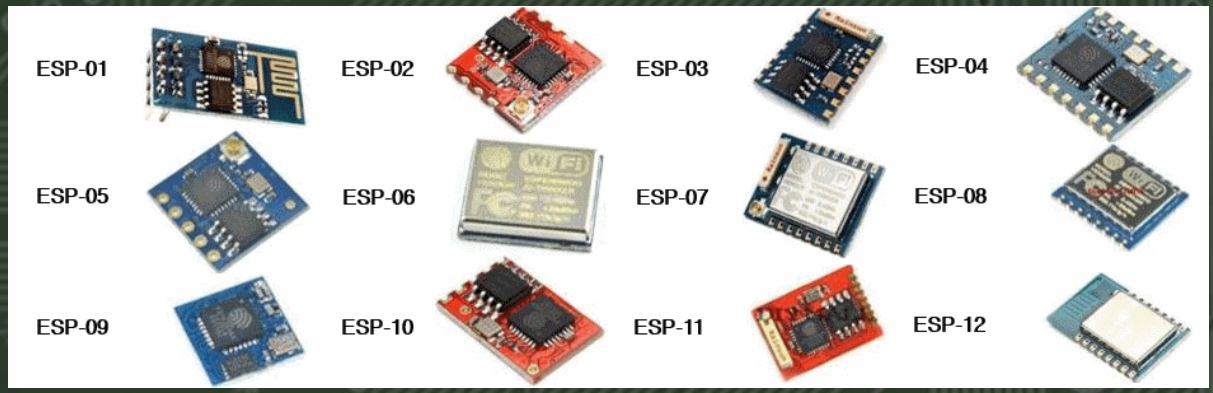
Ruxcon 2016

Overview

ESP8266 Overview

- ESP-01 was the first widely available module, sold as a low cost WiFi->UART adapter
- Actually it's a feature-rich 80MHz (or 160MHz) Xtensa MCU with;
  - I2S/SPI & I^2C
  - ADC
  - WiFi (Station/AP/Station+AP)
  - GPIO
  - UART
  - TCP/IP Stack
  - PWM

ESP8266 Modules



- ESP-01
  - First available model, probably still the most common, very limited IO, needs external wiring to enable deep sleep modes
- ESP-12/12E/12F
  - Castellated module, 2.0mm pitch (which makes it somewhat painful), exposes most IO
- These are just the Ai-Thinker modules, there are various other boards around, including a specific NodeMCU produced dev board

Why NodeMCU?

- Because I'm lazy
- Cross-compiling stuff is a pain in the arse
- It was quick
- Not a fan of "Arduino-ise everything"
- Worked for my applications
- Mostly because I'm lazy

Getting NodeMCU

- The easy way;

  https://nodemcu-build.com Most excellent "cloud-based" build service which allows you to easily build NodeMCU with arbitrary options

- The still relatively easy way;

  https://hub.docker.com/r/marcelstoer/nodemcu-build/ Docker image containing a pre-configured toolchain ready to go

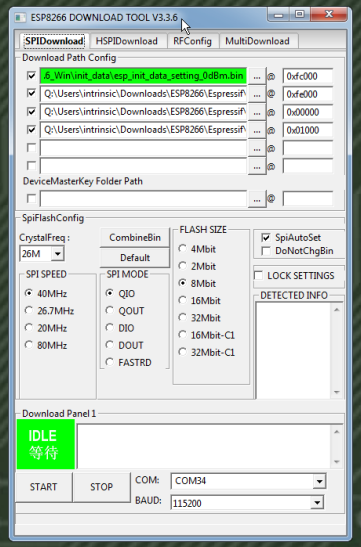
- The hard way;

  http://www.esp8266.com/wiki/doku.php?id=toolchain Manually setup your own NodeMCU build environment

NodeMCU ESP8266 Flasher

- ESP8266Flasher
  - Written by the NodeMCU people
  - Code on github
  - Written in Delphi
  - Next version will be cross-platform
  - Mostly “just works”

ESP8266 Download Tool

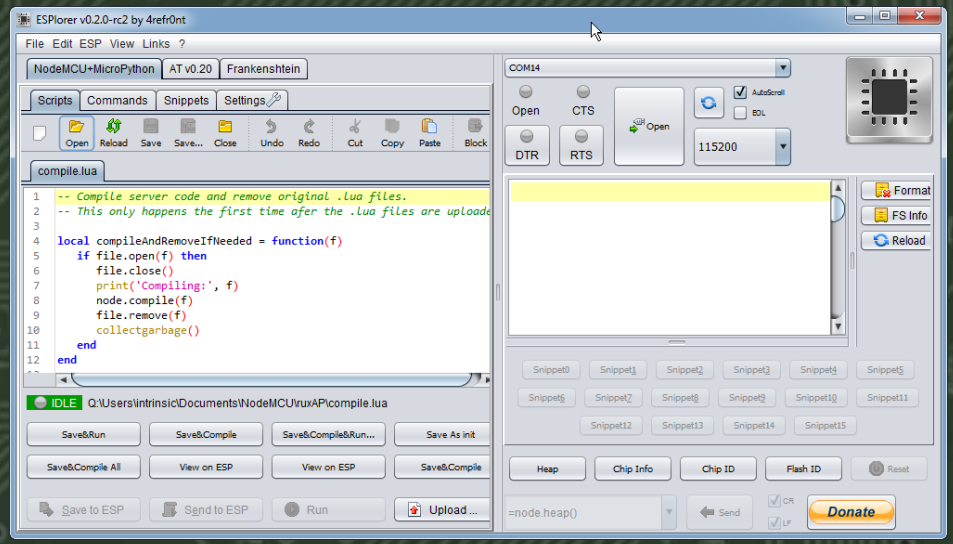


- ESP8266 Download Tool
  - Supplied by Espressif
  - Flexible configuration (perhaps too flexible)
  - Ugly implementation
  - Only reliable way to reprogram the init data block
  - Can also be used to generate new init data blocks

Linux

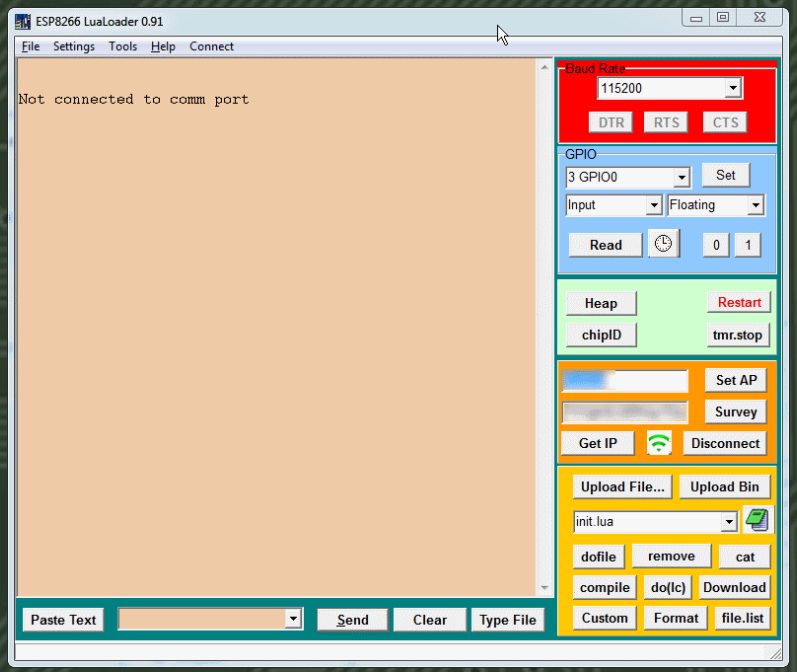
- esptool.py
  - Seems to be the only option for Linux presently
  - Works fine for NodeMCU firmware
  - Issues with flashing init data blocks

ESPlorer



- Pretty decent interface
- A little bit cluttered
- Works OK
- Supports firmwares other than ModeMCU
- Written in Java
- Windows and Linux

LuaLoader



- Basic interface
- Doesn't appear to be actively developed any more
- Seems to have issues with later versions of the NodeMCU firmware

nodemcu-uploader

- Commandline tool
- Written in Python
- Linux and OS X Support (Unspecified “significant issues” with Windows)
- Available in the Debian package repository (and presumably other distros too)

Software Limitations

- Limited SSL/TLS support (Client ONLY)
  - TLS 1.1 with 4 Cipher suites (best is TLS_RSA_WITH_AES256_CBC_SHA).
  - No cert verification
- Crypto module only supports AES-128 (CBC or ECB), MD5, SHA1, SHA256, SHA384, SHA512.
- Supports only one TCP and one UDP server
- No support for concurrent requests in HTTP module
- TX Power control API not exposed
- GPIO mapping is a bit weird (based on the NodeMCU dev board not the ESP pin numbers

Hardware Limitations

- Only 3.3V tolerant
- Excessively spikey power draw, which will kill small batteries in short order
- Some modules spam UART TX data out multiple GPIOs at boot
- Only one ADC channel, either external or battery voltage
- GPIO0 might as well be reserved for Flash mode selection
- ESP-01 has limited applicability outside of WiFi->UART applications due to limited IO

Links

- https://nodemcu.readthedocs.io - Official NodeMCU documentation repository
- http://nodemcu.com/index_en.html - Official NodeMCU website
- http://bbs.espressif.com - Official Espressif ESP8266 discussion forum, SDK releases and other downloads are published here
- https://github.com/marcoskirsch/nodemcu-httpserver - A really nice HTTP server for NodeMCU
- http://www.forward.com.au/pfod/secureChallengeResponse - Challenge-Response authentication protocol for simple devices