Programování ESP8266 v Lue

Programování WiFi mikročipu ve skriptovacím jazyce?
Důvody pro a proti, a ukázky jak na to.

ESP8266

- WiFi SoC
- relativně výkonný 32bit CPU @ 80/160 MHz
- relativně dost paměti: 96 kB / 50 kB
- SDK je v C/C++
- výkon nazbyt svádí...

Jak programovat ESP8266

- (AT příkazy)
- oficiální SDK – verze Non-OS nebo FreeRTOS
- ESP8266 Arduino Core - „plugin“ do Arduino IDE
- skriptovací jazyky (díky výkonu CPU a velikosti RAM)

Důvody pro skriptování

- rychlost vývoje programu
- pohodlnost vývoje programu
- přenositelnost programu na jiné platformy
- snadnější ladění

Důvody proti skriptování

- větší vzdálenost od „železa“
- další vrstva s potenciálními vlastními chybami
- program bude zřejmě běžet pomaleji

Skriptovací jazyky a ESP8266

- Lua – www.nodemcu.com
- MicroPython – www.micropython.org
- Javascript – www.espruino.com
- Basic – www.esp8266basic.com

Lua

- Španělská vesnice nebo portugalsky měsíc?
- Lua je mocný, rychlý, lehký skriptovací jazyk
- kombinuje jednoduchou procedurální syntaxi s mocným popisem dat pomocí asociativních polí a rozšiřitelné syntaxe
- je dynamicky typovaný, za běhu interpretuje bajtkód ve virtuální mašině, má automatickou správu paměti
- ideální pro konfigurace, skriptování a rychlý vývoj
- http://www.luafaq.org/

Jak se naučit programovat v Lue?

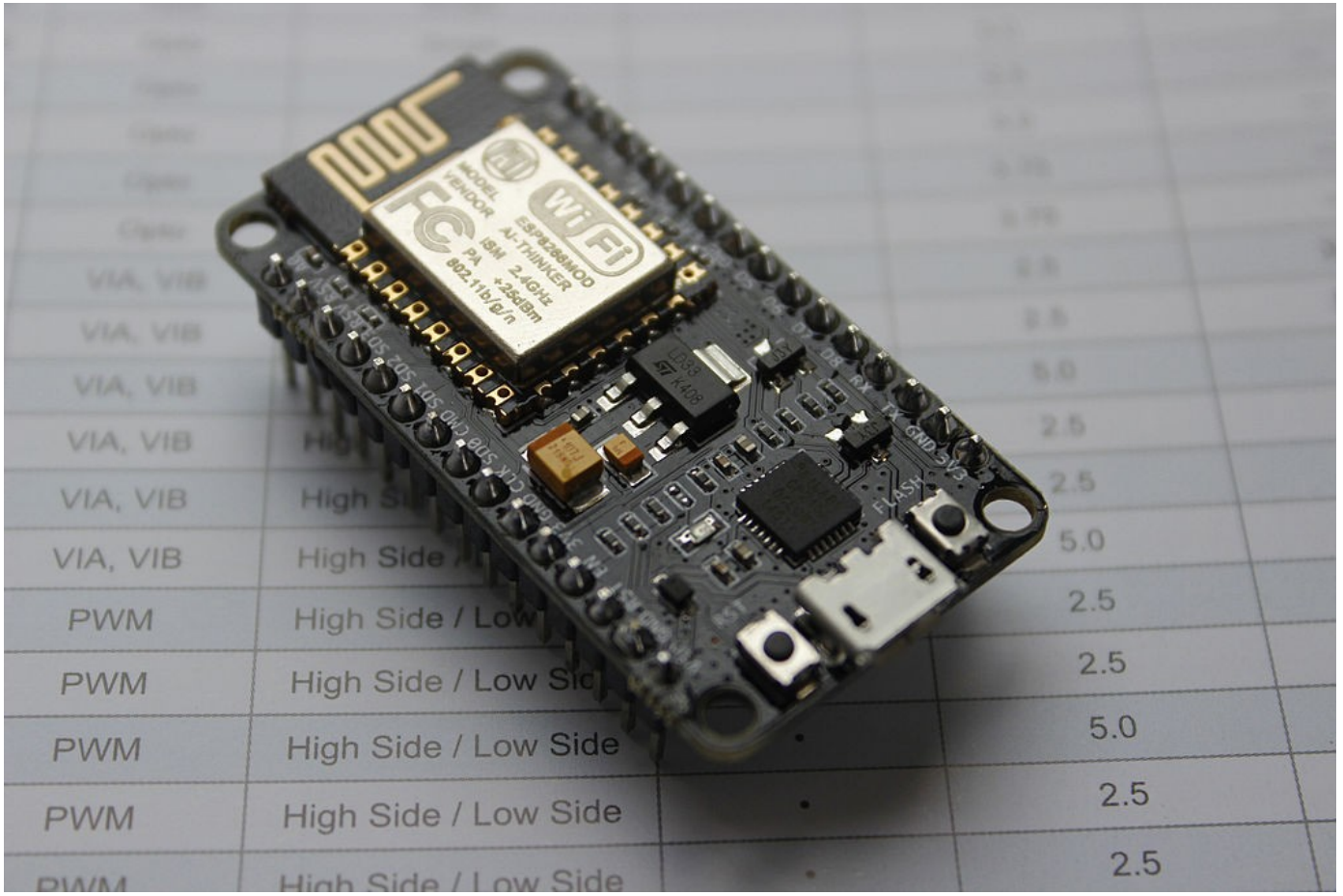
- František Fuka
  http://www.fffilm.name/2013/11/lua-krasa-v-jednoduchosti-video-kniha.html
- Pavel Tišnovský
  https://www.root.cz/serialy/programovaci-jazyk-lua/
- http://www.luafaq.org/gotchas.html
- http://lua-users.org/wiki/LearningLua
- https://learnxinyminutes.com/docs/lua/

Kde se vzala Lua na ESP8266

- NodeMCU = open source vývojová platforma pro IoT
- kombinuje hardware a Lua firmware (eLua port)
- v základu je spiffs, souborový filesystém v paměti
- od léta 2015 vývoj pokračuje dobrovolnicky
- za rok už NodeMCU obsahoval více než 40 modulů: http, mqtt, onewire, u8g, ...

NodeMCU Dev Kit V0.9

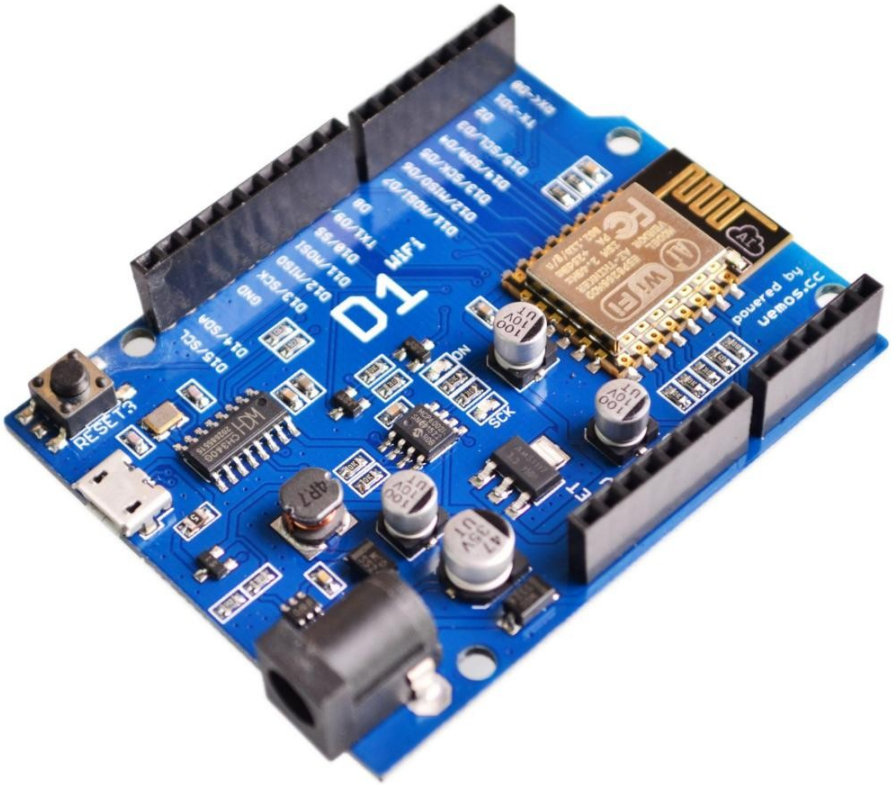
NodeMCU Dev Kit V1.0



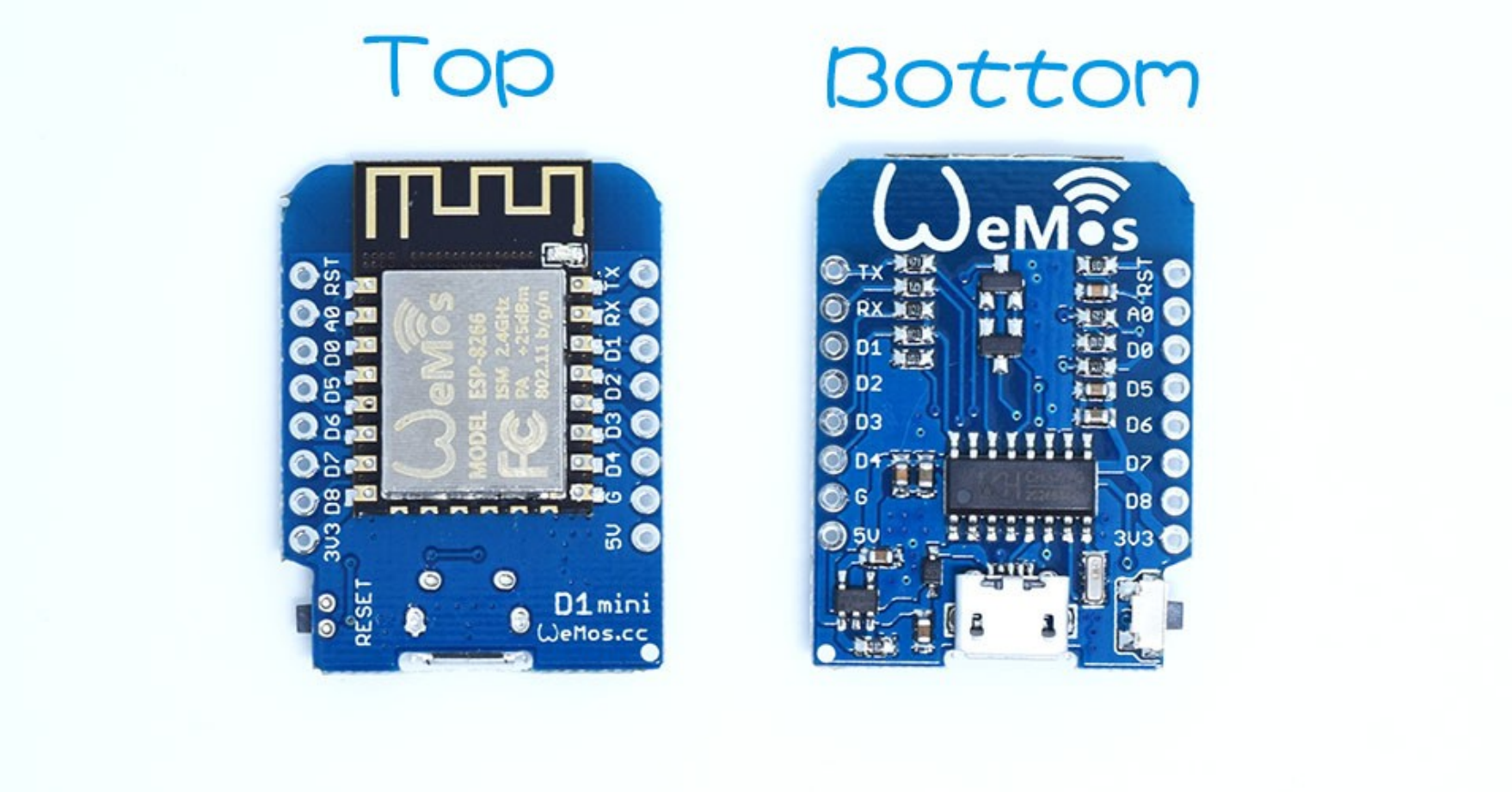
Na jakém HW programovat?

- originál NodeMCU developer kity
- holé moduly ESP v01-12 (aspoň 1 MB flash)
- Wemos a Wemos D1 Mini
- Itead WiFi Sonoff relé či Smart žárovky
- a samozřejmě vlastní HW (smart watch, ...)

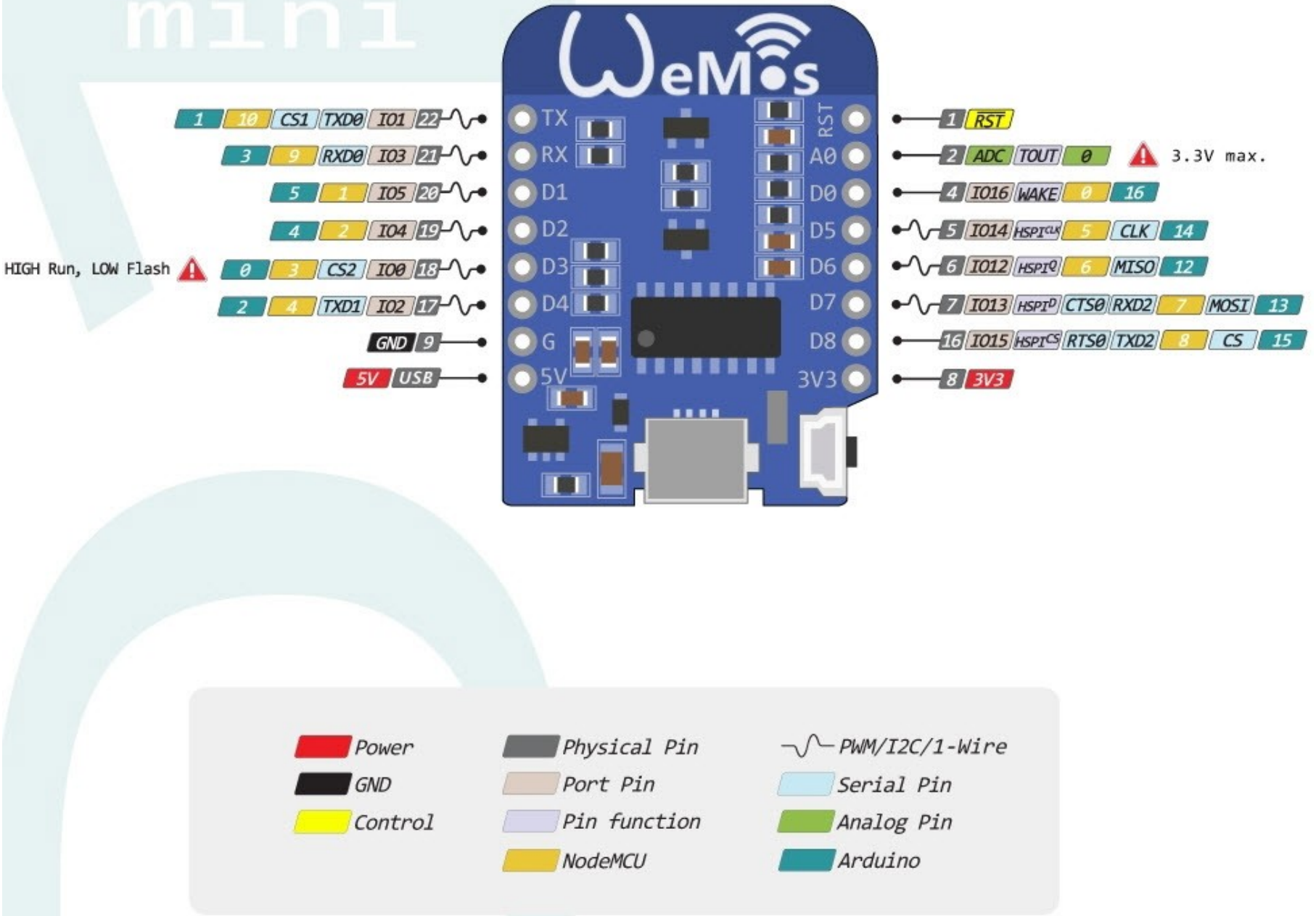
Wemos



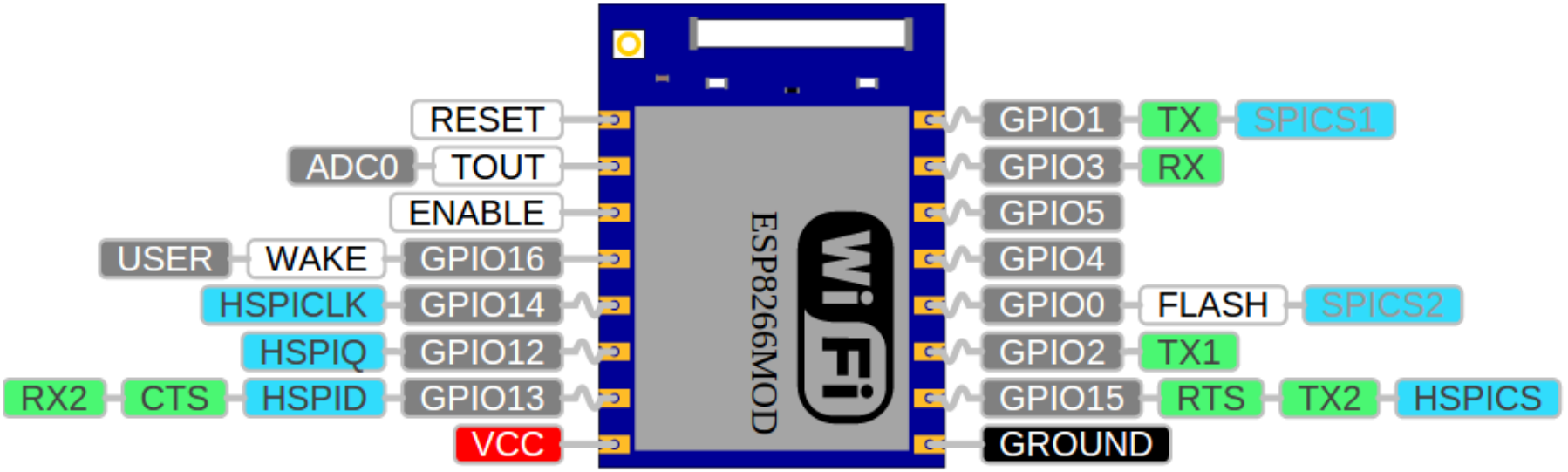
Wemos D1 mini



Moduly – shieldy pro Wemos Mini


WeMos
mini
TX
RX
D1
D2
D3
D4
G
5V
RST
A0
D0
D5
D6
D7
D8
3V3
1 10 CS1 TXD0 IO1 22
3 9 RXD0 IO3 21
5 1 IO5 20
4 2 IO4 19
HIGH Run, LOW Flash
0 3 CS2 IO0 18
2 4 TXD1 IO2 17
GND 9
5V USB
1 RST
2 ADC TOUT 0
3.3V max.
4 IO16 WAKE 0 16
5 IO14 HSPICLK 5 CLK 14
6 IO12 HSPIQ 6 MISO 12
7 IO13 HSPID CTS0 RXD2 7 MOSI 13
16 IO15 HSPICS RTS0 TXD2 8 CS 15
8 3V3
Power
GND
Control
Physical Pin
Port Pin
Pin function
NodeMCU
PWM/I2C/1-Wire
Serial Pin
Analog Pin
Arduino

ESP v07/12 – popis vývodů



Mapování NodeMCU ↔ ESP8266

IO index	ESP8266 pin	IO index	ESP8266 pin
0 [*]	GPIO16	7	GPIO13
1	GPIO5	8	GPIO15
2	GPIO4	9	GPIO3
3	GPIO0	10	GPIO1
4	GPIO2	11	GPIO9
5	GPIO14	12	GPIO10
6	GPIO12		

Omezení Luy na ESP8266

- http://www.eluaproject.net/
- Lua verze 5.1, chybí moduly debug, math a os
- http://nodemcu.readthedocs.io/en/dev/en/lua-developer-faq/
- integer a floating point verze firmware
- událostmi řízené programování
- kód by neměl běžet déle než 10 ms v kuse
- omezení daná velikostí volné RAM

Kde vzít Luu pro ESP8266

- vývoj pod OpenSource licencí na GitHubu
- https://github.com/nodemcu/nodemcu-firmware
- Docker image s build environmentem
- on-line buildovací služba www.nodemcu-build.com

Verze Luy pro ESP8266

- vývojové větve master a dev
- master = stabilní, dev = aktuální vývoj
- založeno na Espressif Non-OS SDK
- pozor na kompatibilitu při větším upgrade
  http://nodemcu.readthedocs.io/en/dev/en/flash/#upgrading-firmware

„Flashování“ firmware

- esptool.py – github.com/themadinventor/esptool
- https://nodemcu.readthedocs.io/en/master/en/flash/

NodeMCU příkazová řádka

- terminálový program – UART na 115200 bps 8N1
- DTR pulz resetuje ESP8266
- po startu se automaticky spouští init.lua
- programy nahráváme pomocí nodemcu-uploader.py
  http://nodemcu.readthedocs.io/en/dev/en/upload/

Esplorer

- ruský pokus o NodeMCU/Lua IDE v Javě
- žije na http://esp8266.ru/esplorer/
- obsahuje editor, uploader, terminál
- verze 0.2.0 odpovídá kvalitě programu
- první pokusy o podporu MicroPythonu

Příklad v Lue: rozsvítit LED

On-board LED je připojena mezi TX1 = GPIO2 a VCC

```
gpio.mode(4, gpio.OUTPUT)
gpio.write(4, gpio.LOW)
```

Příklad: blikání LED bez čekání

```
LED_PIN = 4
gpio.mode(LED_PIN, gpio.OUTPUT)
value = true

tmr.alarm(0, 500, tmr.ALARM_AUTO, function ()
   gpio.write(LED_PIN, value and gpio.HIGH or gpio.LOW)
   value = not value
end)
```

Příklad: připojení k AP

```
wifi.setmode(wifi.STATION)
wifi.sta.config("LinuxDays","linuxdays")
tmr.alarm(1, 1000, 1, function()
    if wifi.sta.getip() == nil then
        print("Připojuji...")
    else
        tmr.stop(1)
        print("Připojeno, IP je " .. wifi.sta.getip())
    end
end)
```

Příklad: HTTP GET

```
http.get("http://httpbin.org/ip", nil, function(code,
data)
    if (code < 0) then
        print("HTTP požadavek selhal")
    else
        print(code, data)
    end
end)
```

Příklad: web server

```
srv = net.createServer(net.TCP)
srv:listen(80, function(conn)
    conn:on("receive", function(sck, payload)
        print(payload)
        sck:send("HTTP/1.0 200 OK\r\nContent-Type:
text/html\r\n\r\n<h1>Ahoj z NodeMCU.</h1>")
    end)
    conn:on("sent", function(sck) sck:close() end)
end)
```

Příklad: WS2812 LED pásek

```
ws2812.init()
-- první dvě RGB LED budou zelené
ws2812.write(string.char(255, 0, 0, 255, 0, 0))
```

Příklad: WS2812 LED efekt

```
ws2812.init()
local i, buffer = 0, ws2812.newBuffer(10, 3)
buffer:fill(0, 0, 0)
tmr.alarm(0, 50, 1, function()
    i=i+1
    buffer:fade(2)
    buffer:set(i%buffer:size()+1, string.char(0, 0, 255))
    ws2812.write(buffer)
end)
```

Díky za pozornost

… a těším se na otázky

Petr Stehlík

www.pstehlik.cz

petr@pstehlik.cz