

Lua auf dem ESP8266

Outline

ESP8266

Hardware

Software

Lua

Intro

Syntax

NodeMCU

Intro

Beispiele

Topic

ESP8266

Hardware

Software

Lua

Intro

Syntax

NodeMCU

Intro

Beispiele

Specs

- ▶ 32-bit, 80 MHz (160 MHz übertaktet) Tensilica Xtensa CPU
- ▶ 64 kB RAM für Programmcode, 96 kB RAM für Daten
- ▶ 4 MB Flash
- ▶ Wi-Fi (802.11 b/g/n)
- ▶ 16 GPIO (3,3 Volt)
- ▶ SPI
- ▶ I²C
- ▶ 10-bit ADC

Ausführungen



SDK / Firmware

- ▶ kurze Übersicht wie man den ESP8266 programmieren kann
- ▶ Durchgängiges Beispiel: Verbindung mit einem WiFi Access-Point herstellen

ESP8266 RTOS SDK ¹

- ▶ C auf dem ESP8266
- ▶ basiert auf FreeRTOS
- ▶ von Espressif gepflegt
- ▶ nach Änderungen muss die Firmware gebaut und geflasht werden

¹https://github.com/espressif/ESP8266_RTOS_SDK

Beispielcode I

```
1  #include <string.h>
2  #include "freertos/FreeRTOS.h"
3  #include "freertos/task.h"
4  #include "freertos/event_groups.h"
5  #include "rom/ets_sys.h"
6  #include "esp_wifi.h"
7  #include "esp_event_loop.h"
8  #include "esp_log.h"
9  #include "nvs_flash.h"
10
11 static EventGroupHandle_t wifi_event_group;
12 static const char *TAG = "simple wifi";
13
14 static esp_err_t event_handler(void *ctx, system_event_t *event) {
15     switch(event->event_id) {
16         case SYSTEM_EVENT_STA_START:
17             esp_wifi_connect();
18             break;
19         case SYSTEM_EVENT_STA_GOT_IP:
20             ESP_LOGI(TAG, "got ip:%s", ip4addr_ntoa(&event->event_info.got_ip.ip_in
```


Beispielcode II

```
21         xEventGroupSetBits(wifi_event_group, BIT0);
22         break;
23     default:
24         break;
25     }
26     return ESP_OK;
27 }
28
29 void app_main() {
30     esp_err_t ret = nvs_flash_init();
31     if (ret == ESP_ERR_NVS_NO_FREE_PAGES) {
32         ESP_ERROR_CHECK(nvs_flash_erase());
33         ret = nvs_flash_init();
34     }
35     ESP_ERROR_CHECK(ret);
36
37     wifi_event_group = xEventGroupCreate();
38
39     tcpip_adapter_init();
40     ESP_ERROR_CHECK(esp_event_loop_init(event_handler, NULL) );
41
```

Beispielcode III

```
42     wifi_init_config_t cfg = WIFI_INIT_CONFIG_DEFAULT();
43     ESP_ERROR_CHECK(esp_wifi_init(&cfg));
44     wifi_config_t wifi_config = { .sta = {
45         .ssid = EXAMPLE_ESP_WIFI_SSID,
46         .password = EXAMPLE_ESP_WIFI_PASS
47     }, };
48
49     ESP_ERROR_CHECK(esp_wifi_set_mode(WIFI_MODE_STA) );
50     ESP_ERROR_CHECK(esp_wifi_set_config(ESP_IF_WIFI_STA, &wifi_config) );
51     ESP_ERROR_CHECK(esp_wifi_start() );
52
53     ESP_LOGI(TAG, "wifi_init_sta finished.");
54     ESP_LOGI(TAG, "connect to ap SSID:%s password:%s",
55             EXAMPLE_ESP_WIFI_SSID,
56             EXAMPLE_ESP_WIFI_PASS);
57 }
```

https:

[//github.com/espressif/ESP8266_RTOS_SDK/blob/master/examples/wifi/simple_wifi/main/simple_wifi.c](https://github.com/espressif/ESP8266_RTOS_SDK/blob/master/examples/wifi/simple_wifi/main/simple_wifi.c)

Arduino ³

- ▶ C auf dem ESP8266
- ▶ basiert auf ESP8266 nonOS SDK von Espressif ²
- ▶ auch ein bauen / flashen Zyklus nach Quellcode Änderungen
- ▶ aber bedeutend weniger Code nötig

²https://github.com/espressif/ESP8266_NONOS_SDK

³<https://github.com/esp8266/Arduino>

Beispielcode

```
1  #include <ESP8266WiFi.h>
2
3  void setup() {
4      Serial.begin(115200);
5      Serial.println();
6
7      WiFi.begin("network-name", "pass-to-network");
8
9      while (WiFi.status() != WL_CONNECTED) {
10         delay(500);
11         Serial.print(".");
12     }
13     Serial.println();
14
15     Serial.print("Connected, IP address: ");
16     Serial.println(WiFi.localIP());
17 }
18
19 void loop() {}
```

MicroPython ⁶

- ▶ Python auf dem ESP8266
- ▶ unterstützt mehrere Boards
- ▶ basiert auf dem esp-open-sdk⁴
- ▶ falls man schon Python kann: Go for it!
- ▶ ist wohl noch neu auf dem ESP8266: ⁵

This is an experimental port of MicroPython for the WiFi modules based on Espressif ESP8266 chip.

WARNING: The port is experimental and many APIs are subject to change.

⁴<https://github.com/pfalcon/esp-open-sdk>

⁵<https://github.com/micropython/micropython/tree/master/ports/esp8266>

⁶<https://github.com/micropython/micropython>

Beispielcode

```
1 wlan = network.WLAN(network.STA_IF)
2 wlan.active(True)
3 if not wlan.isconnected():
4     print('connecting to network...')
5     wlan.connect('essid', 'password')
6     while not wlan.isconnected():
7         pass
8 print('network config:', wlan.ifconfig())
```

<http://docs.micropython.org/en/latest/esp8266/quickref.html>

uLisp ⁷

- ▶ Lisp auf dem ESP8266
- ▶ basiert auf Arduino (ESP8266 core for Arduino)
- ▶ unterstützt viele Boards
- ▶ leider noch sehr wenig eingebaute Module
 - ▶ kein One-Wire, kein MQTT, ...
 - ▶ sehr wenig Device driver

⁷<http://www.ulisp.com>

Beispielcode

```
1 (wifi-connect "<SSID>" "<PWD>")
```

- ▶ gibt die erhaltene IP Adresse als String zurück

NodeMCU ¹²

- ▶ Lua (5.1.4) auf dem ESP8266
- ▶ basiert auf ESP8266 nonOS SDK von Espressif ⁸ und eLua ⁹
- ▶ viele eingebaute Module (> 65)
- ▶ das NodeMCU Projekt begann 1 Jahr (am 13.10.2014) nachdem Espressif den ESP8266 produzierte ¹⁰

NodeMCU für ESP32

- ▶ basiert auf Espressif IoT Development Framework ¹¹

⁸https://github.com/espressif/ESP8266_NONOS_SDK

⁹<http://www.eluaproject.net/>

¹⁰<https://github.com/nodemcu/nodemcu-firmware/commit/9c98808289d0863a41c695e03d4067424fc1cdec>

¹¹<https://github.com/espressif/esp-idf>

¹²<https://nodemcu.readthedocs.io/en/master/>

Beispielcode

```
1  wifi.setmode(wifi.STATION)
2  wifi.sta.config{ ssid = "<SSID>", pwd = "<PWD>" }
3
4  tmr.alarm(0, 500, tmr.ALARM_AUTO, function()
5      if wifi.sta.getip() == nil then
6          print(".")
7      else
8          print("Connected, IP Address: " .. wifi.sta.getip())
9          tmr.stop(0)
10         end
11     end)
```

Topic

ESP8266

Hardware

Software

Lua

Intro

Syntax

NodeMCU

Intro

Beispiele

Was ist Lua?

- ▶ (embeddable) Script-Sprache
- ▶ interpretierte Sprache (es gibt aber auch einen jit)
- ▶ dynamisch getypt
- ▶ kleiner Sprachkern
- ▶ unterstützt in der stdlib nur was ANSI C unterstützt - es gibt aber Erweiterungen

Einsatz

- ▶ Scripten in Apps
 - ▶ Adobe Lightroom
 - ▶ Nginx
 - ▶ ...
- ▶ embedded devices
 - ▶ Autos (Volvo, ...?)
 - ▶ TV's (Samsung, ...?)
 - ▶ Routers (OpenWRT, ...?)
 - ▶ ...
- ▶ Spiele
 - ▶ Warcraft
 - ▶ Angry Birds
 - ▶ Löve 2D Engine¹³
 - ▶ ...

¹³<https://love2d.org/>

Variablen

```
1 number = 4
2 local string = "hallo"
```

1. "globale" Variable
2. lokale Variable - nur innerhalb des aktuellen Blocks gültig

Funktionen

Function definition

```
1 function greet(name)
2     print("Hello, " .. name)
3 end
```

Function call

```
> greet("<NAME>")
Hello, <NAME>
```

Tabellen

Definition

```
1 local devices = {}  
2 devices.type = "esp"  
3  
4 local key = "location"  
5 devices[key] = "oben"  
6  
7 local devices = {  
8     type = "esp",  
9     [key] = "oben"  
10 }
```

Access

```
1 devices.type      -> "esp"  
2 devices.location  -> "oben"  
3 devices["location"] -> "oben"
```

- ▶ so was wie Associative Array, Hash, Map oder Dictionary
- ▶ Die Datenstruktur in Lua
- ▶ Tabellen werden in Lua benutzt für: Packages, OOP features, Environments, Tracing, ...

Topic

ESP8266

Hardware

Software

Lua

Intro

Syntax

NodeMCU

Intro

Beispiele

Ablauf

1. Firmware bauen / besorgen
2. NodeMCU Firmware auf ESP flashen (`esptool.py`)
3. Lua Programm-Code auf ESP kopieren (`nodemcu-uploader`)
4. NodeMCU neu starten
5. beim booten evaluiert das Board die `init.lua` Datei

1. Firmware bauen / besorgen

- ▶ Da die NodeMCU Firmware so viele Module enthält, und der ESP8266 wenig RAM hat, muss beim Firmware bauen / besorgen die gewünschten Module gewählt werden
- ▶ Es gibt drei verschiedene Möglichkeiten sich die Firmware zu besorgen
 - ▶ Cloud Build Service ¹⁴
 - ▶ Docker Image ¹⁵
 - ▶ Auf dem eigenen Rechner bauen ¹⁶

¹⁴<https://nodemcu-build.com/>

¹⁵<https://hub.docker.com/r/marcelstoer/nodemcu-build/>

¹⁶<https://nodemcu.readthedocs.io/en/master/en/build/#linux-build-environment>

2a. NodeMCU Firmware auf ESP flashen

- ▶ hier gibt es verschiedene Tools
- ▶ ich benutzte `esptool` ¹⁷ von Espressif

```
1 esptool.py write_flash 0x0 nodemcu-dev-16-modules-<DATE>-integer.bin
```

```
esptool.py v2.5.0
Found 1 serial ports
Serial port /dev/ttyUSB0
Connecting...
Detecting chip type... ESP8266
Chip is ESP8266EX
Features: WiFi
MAC: 5c:cf:7f:24:88:5f
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Auto-detected Flash size: 4MB
Flash params set to 0x0240
Compressed 524288 bytes to 298520...
Wrote 524288 bytes (298520 compressed) at 0x00000000 in 26.4 seconds (effective 158.8 kbit/s)...
Hash of data verified.

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

¹⁷<https://github.com/espressif/esptool>

2b. Wenn der Firmware Upload nicht funktioniert

1 `esptool.py erase_flash`

```
esptool.py v2.5.0
Found 1 serial ports
Serial port /dev/ttyUSB0
Connecting...
Detecting chip type... ESP8266
Chip is ESP8266EX
Features: WiFi
MAC: 5c:cf:7f:24:88:5f
Uploading stub...
Running stub...
Stub running...
Erasing flash (this may take a while)...
Chip erase completed successfully in 9.9s
Hard resetting via RTS pin...
```

3. Lua Programm-Code auf ESP kopieren

- ▶ auch hier gibt es wieder verschiedene Tools
- ▶ ich benutzte dafür `nodemcu-uploader`¹⁸

```
1 nodemcu-uploader upload led-stripe.lua
```

```
opening port /dev/ttyUSB0 with 115200 baud
Preparing esp for transfer.
Transferring led-stripe.lua as led-stripe.lua
All done!
```

¹⁸<https://github.com/kmpm/nodemcu-uploader>

4. NodeMCU neu starten

1 nodemcu-uploader node restart

```
opening port /dev/ttyUSB0 with 115200 baud
Restart
node.restart()
```

1 nodemcu-uploader terminal

```
ets Jan  8 2013,rst cause:2, boot mode:(3,7)
load 0x40100000, len 26424, room 16
tail 8
.... <GARBAGE OMITTED>
No LFS image loaded

NodeMCU custom build by frightanic.com
  branch: dev
  commit: 87b3ffa6bd909412bb6de6b9a056cbcf64d1bed1
  SSL: false
  modules: adc,am2320,ds18b20,file,gpio,i2c,mqtt,net,node,ow,pwm,rfswitch,
           sjson,tmr,uart,wifi
  build created on 2019-01-02 12:26
  powered by Lua 5.1.4 on SDK 2.2.1(6ab97e9)
  lua: cannot open init.lua
```

5. beim booten evaluiert das Board die `init.lua` Datei

Typische Inhalt meiner `init.lua`

```
1 dofile("wifi.lua")
2 wifi_ap_connect("<SSID>", "<PWD>", function()
3     dofile("<moduleA>.lua")
4     dofile("project.lua")
5 end)
```

1. lade mein WiFi helper script
2. verbinde dich mit dem AP, und nachdem der ESP eine IP hat, führe die Befehle aus dem übergebenen Callback aus

Temperatur

- ▶ unterstützt werden LM92, SI7021, DHT11/22, BMP085, HDC1000, BME280, BME680, HDC1080, AM2320 und DS18B20
- ▶ Beispiel für AM2320

```
1  function setup()
2      local sda, scl = 1, 2
3      i2c.setup(0, sda, scl, i2c.SLOW)
4      am2320.setup()
5  end
6
7  function current_temperature()
8      relative_humidity, temperature = am2320.read()
9  end
```

Funk-Schalter (433MHz Steckdose)

```
1  local function send(value)
2      rfswitch.send(
3          1,      -- protocol id from the switch
4          317,   -- pulse length in microseconds
5          4,      -- number of transmit repeats
6          3,      -- GPIO pin
7          value,
8          24      -- bit length of value
9      )
10 end
11
12 function lightOn()
13     send(4433)
14 end
15
16 function lightOff()
17     send(4436)
18 end
```

meine wifi.lua

```
1  --
2  -- connect to wifi ap
3  --
4  function wifi_connect_ap(ssid, pwd, cb)
5
6      wifi.setmode(wifi.STATION)
7      wifi.sta.config{ ssid = ssid, pwd = pwd }
8
9      tmr.alarm(0, 1000, tmr.ALARM_AUTO, function()
10         if wifi.sta.getip() == nil then
11             print("connecting ...")
12         else
13             local ip, nm, gw = wifi.sta.getip()
14             print("my ip: " .. ip)
15             print("netmask: " .. nm)
16             print("gw: " .. gw)
17             tmr.stop(0)
18
19             cb()
20         end
21     end)
22 end
```

led-stripe

Zutaten

- ▶ ESP8266
- ▶ LED Stripe
- ▶ 3x IRLZ34 N-Channel MOSFET

```
1  --
2  -- setup gpio pins
3  --
4  leds = { red = 7, green = 1, blue = 4 }
5  for color, pin in pairs(leds) do
6      print(string.format("setup: %s on pin: %d", color, pin))
7      pwm.setup(leds[color], 500, 0)
8      pwm.start(leds[color])
9  end
10
11 --
12 -- initialize the actor
13 --
14 Actor = require("actor")
15 actor = Actor:new{name = "led-stripe"}
16
17 actor:connect("192.168.1.20", function()
18     actor:subscribe("actors/led-stripe/color")
19 end)
20
21
22
```

```
23  --
24  -- helper
25  --
26
27  -- collect the given iterator in a list
28  local function collect(...)
29      local acc = {}
30      for x in ... do
31          acc[#acc + 1] = x
32      end
33      return acc
34  end
35
36  -- convert the given r/g/b value to a pwm value
37  -- and set it on the given pin
38  local function set(pin, value)
39      local pwmValue = 102300 / 255 * value / 100
40      print("set pwm pin: " .. pin .. " to: " .. pwmValue)
41      pwm.setduty(pin, pwmValue)
42  end
43
44
45
```

```
46  --
47  -- register the message handler
48  --
49  actor:onMessage(function(topic, data)
50
51     -- data are 'r,g,b' values - somethink like '120,0,30'
52     -- red: 120, green: 0, blue: 30
53     local values = collect(string.gmatch(data, "%d+"))
54     if #values == 3 then
55         set(leds.red, values[1])
56         set(leds.green, values[2])
57         set(leds.blue, values[3])
58     end
59 end)
```

▶ nonOS vs RTOS:

<https://bbs.espressif.com/viewtopic.php?t=838>