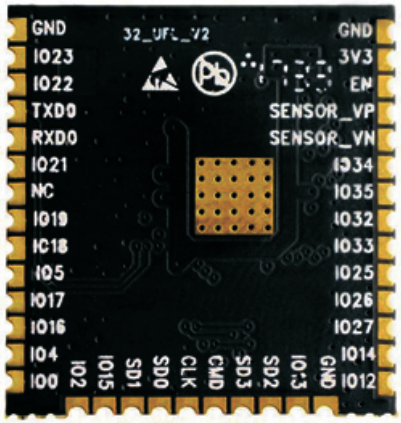
ESP 32 a Micropython
– pro roboty i Internet věcí

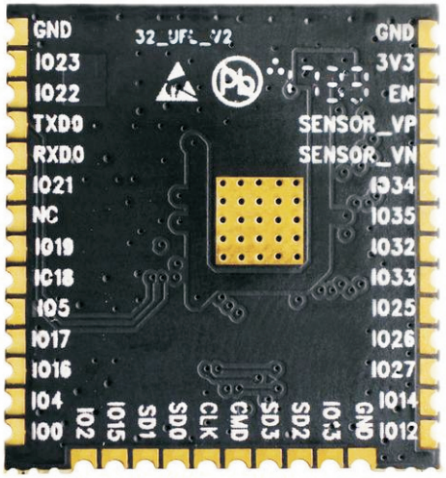


MicroPython

octopusengine

Co je to ESP32?

Úspěšným předchůdcem je **ESP8266 - čip s 32bitovým RISC**ovým jádrem "původně určený pro **doplnění Wi-Fi konektivity** k existujícím zařízením, komunikující přes sériovou linku pomocí AT příkazů."
(postupně různé verze: ESP-01 až ESP-12...)



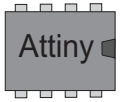
ESP nyní vyrábí **Espressif Systems**
- která se profiluje jako IoT leader

1.9.2016 představen > ESP32

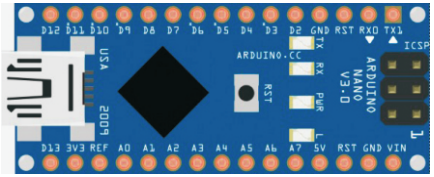
<u>Základní parametry:</u>
240 MHz dual core Tensilica LX6 microcontroller with 600 DMIPS
Integrated **520 KB SRAM**, 4 MB flash,
Transceiver; 802.11 a/b/g/n (**Wi-Fi**, WiFi, WLAN),
Bluetooth® Smart 4.x Low Energy (BLE)
2.3V to 3.6V operating voltage...

Hrubé HW porovnání

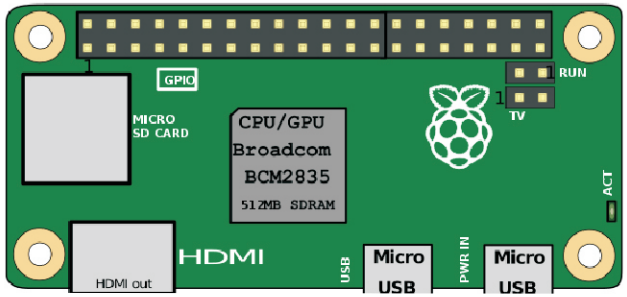
<u>Attiny</u>
1-8MHz
4-8kB Flash
256-512B RAM
I2C, ...

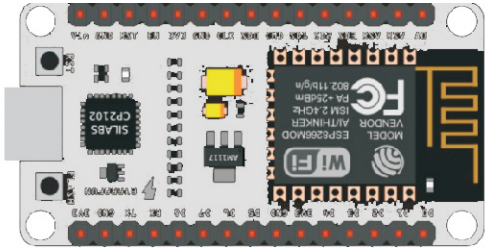


<u>Atmel</u>
Arduino Nano (Uno)
16MHz
32kB Flash
2kb RAM

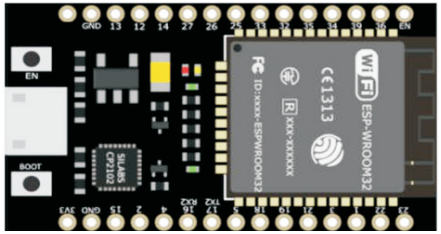


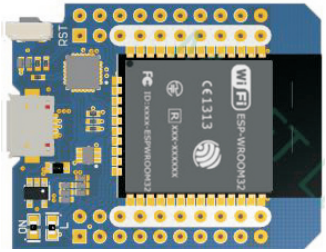
<u>Raspberry</u> Pi (Zero)
1GHz
512-1GHB RAM
SD - karta
40 PIN GPIO
HDMI 1080p
WiFi ...





<u>ESP 8266</u>
80-190MHz
512-4M Flash
32-80kB RAM
Wifi / BT



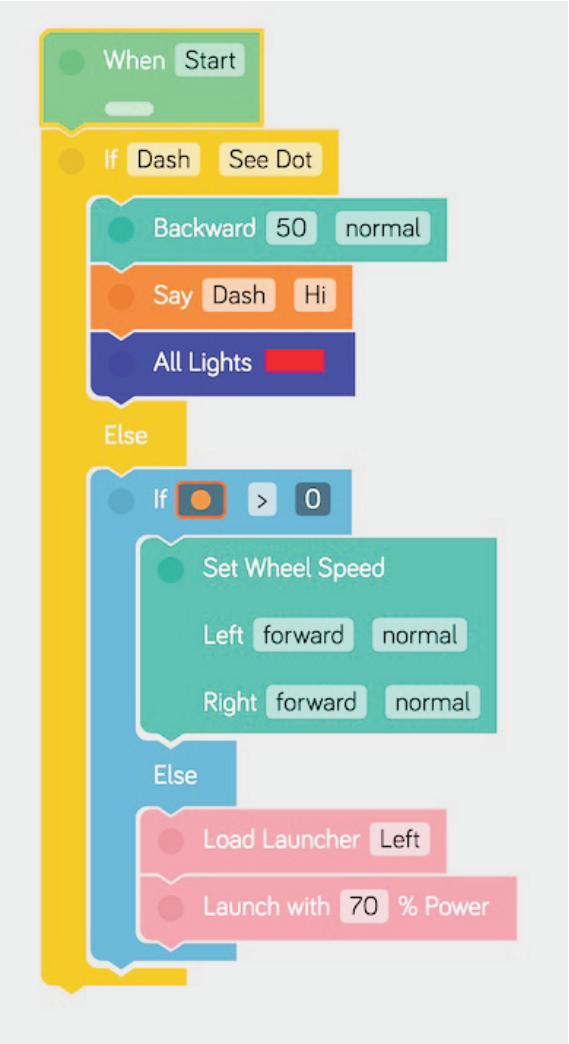


<u>ESP32</u>
160-240MHz
4-16MB Flash
520kB RAM
WiFi / BT
...
hall

octopusLAB.cz
facebook/octopuslabcz
github/octopusengine | instagram/octopusengine

SDK / FW / SW

Pro vytvoření vlastní aplikace je nutné kromě konkrétního programovacího jazyka "instalovat" také **SDK - Software development kit:**
"SDK je typická sada vývojových nástrojů (devkit) umožňující vytváření aplikací pro určité softwarové balíčky"



<u>Programování ESP32:</u>
Strojový kód / Assembler (nedostatečná dokumentace)
Arduino C (implementováno v Arduino studiu)
BASIC (některé modely mají vestavěný 68000 tinyB)
LUA - JS
Micropython >>
Blockly https://github.com/ItsMates/ardublockly-micropython

Nástroje: Esptool, apmy, ...

Python > Micropython

Python - programovací jazyk *1994 | 2.7 - 2010 | 3.6 - 2016 | 3.7 ...
Python komunita - pyladies, pyvo.cz...

+ jednoduchý
  (i složitější konstrukce)
+ přehledný
+ opensource
+ rozšiřitelné o "C moduly"
  (nebo assembbler přímo)

- někdy o dost pomalejší
(některý HW nepoužitelný)
- občas nestabilní
(timers, threads..)
- komunita teprve "začíná"
>

Petr Kracík - github - contributor > pull request >
(Ethernet time signal, bug při ext. odpojení wifi, síla wifi signálu...)
Petr Viktorín - python komunita (Robot board na dev conu)
https://github.com/encukou/hw

uPy: Blikání LEDkou | GPIO pinouts

```
from machine import Pin # GPIO <<<
from time import sleep

BUILT_IN_LED = 2

led = Pin(BUILT_IN_LED, Pin.OUT)
while True:
    led.value(0)
    sleep(1/2)
    led.value(1)
    sleep(1/2)
```

```
# device.json >>>>> pinouts

from micropython import const

# PIN as on octopusLAB Wemos ESP8266 IoTBoard1
BUILT_IN_LED = const(2)

BUTT1_PIN = 12 #d6 x gpio16=d0
PIEZZO_PIN = 14
WS_LED_PIN = 15 #wemos gpio14 = d5
ONE_WIRE_PIN = 13
```

```
octopuASCII = [
"        ,'''`.",
"       /     \",
"       |(@)(@)|",
"       )       (",
"      /,'))((`.\",
"     (( ((  )) ))",
"     )  \ `)(' / (",
]

def printOctopus():
    for ol in octopuASCII:
        print(str(ol))
    print()  # newline
```

```
# I2C:
I2C_SCL_PIN=5 #gpio5=d1
I2C_SDA_PIN=4 #gpio4=d2

# SPI:
SPI_CLK_PIN  = const(18)
SPI_MISO_PIN = const(19)
SPI_MOSI_PIN = const(23)
SPI_CS0_PIN  = const(5)
...
```

config | file | json | ...

pinout.py from device config json file

```
with open('config/device.json', 'r') as f:
            d = f.read()
            f.close()
            device_config = json.loads(d)

if device_config.get('soc_type') == "esp32":
    import pinouts.olab_esp32_default as pinout
```

```
>>> čtení teploměru:
from lib.temperature import TemperatureSensor
ts = TemperatureSensor(pinout.ONE_WIRE_PIN)
temp = ts.read_temp()

>>> 8x7 segment displej:
from lib.max7219_8digit import Display
#spi = SPI(-1, baudrate=100000)
#ss = Pin(15, Pin.OUT)
d7 = Display(spi, ss)
d7.write_to_buffer('12345678') # buffer <<<<<
d7.display()

>>> serial display:
from machine import UART
uart = UART(2, 9600) #UART2 sw <<<<<
uart.write('C')      # clear
uart.write('ROW2')   # row and color
uart.write('QoctopusLAB - UART2 test*')
```

uPy: pro robotická vozítka i IoT

```
>>> stepper
from lib.sm28byj48 import SM28BYJ48
ADDRESS = 0x23
MOTOR_ID1 = 1  # motor id 1 or 2
i2c_sda = Pin(pinout.I2C_SDA_PIN, Pin.IN,  Pin.PULL_UP)
i2c_scl = Pin(pinout.I2C_SCL_PIN, Pin.OUT, Pin.PULL_UP)

i2c = machine.I2C(scl=i2c_scl, sda=i2c_sda, freq=100000)
# 100kHz as Expander is slow :(
motor1 = SM28BYJ48(i2c, ADDRESS, MOTOR_ID1)
motor1.turn_degree(90) # turn right 90 deg
```

```
>>> wifi
from util.wifi_connect import read_wifi_config, WiFiConnect
wifi_config = read_wifi_config()
print("config for: " + wifi_config["wifi_ssid"])
w = WiFiConnect()
w.connect(wifi_config["wifi_ssid"], wifi_config["wifi_pass"])
# add timeout <<<<<
print("WiFi: OK")
```

octopus LAB - github

octopusengine / **octopuslab**

Unwatch 7

Code | Issues 3 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Branch: master | octopuslab / esp32-micropython / | Create new file

petrkr Micropython: wifi signal with oled example: wait for connect

..	
_examples	Micropython: wifi signal with oled example: wait for connect
_projects	Update README.md
assets	add 9x9 icons
config	d>D
lib	add ligit senzor
pinouts	add moisture pins
util	add IoT board test relay
wwwesp	add simple index.html example
README.CS_cz.md	Update README.CS_cz.md
README.md	Update README.md
boot.py	add octopus()

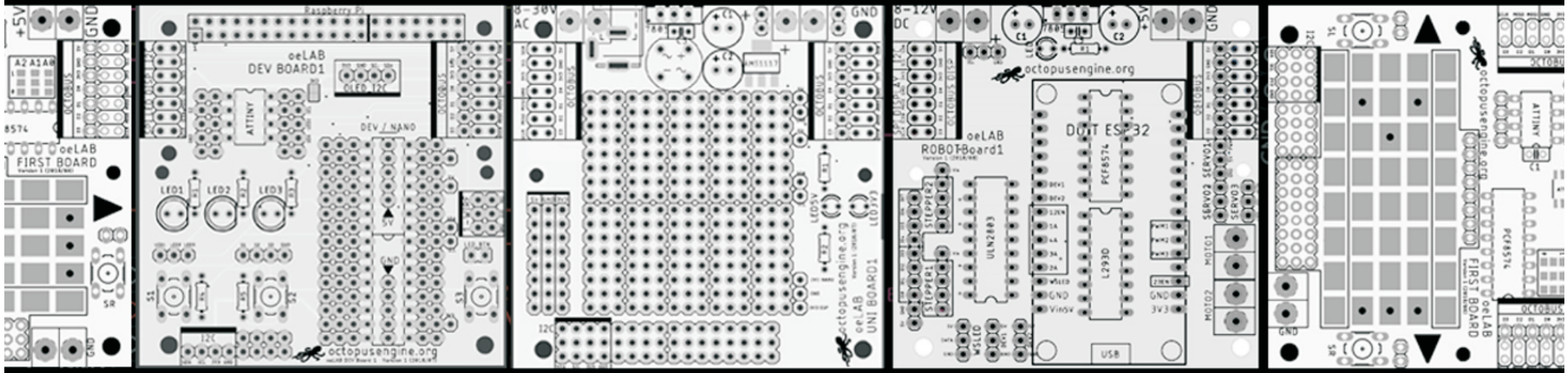
octopusLAB.cz

facebook/octopuslabcz

github/octopusengine | instagram/octopusengine

Octopus LAB

experimental, development & learning modules



octopusengine

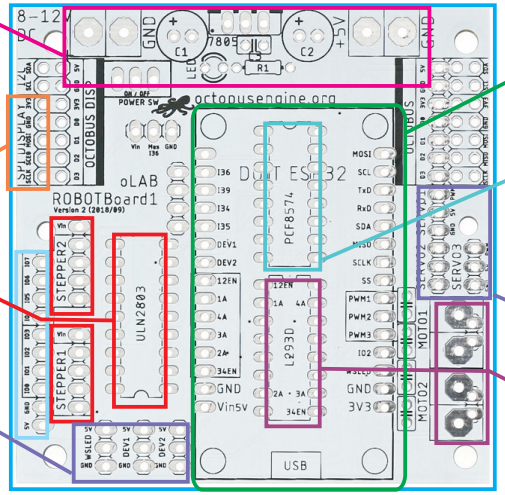
octopus LAB - ROBOTboard | IOTboard

ROBOT board

Napájecí část
konektor pro
výkonových 8-12V
(pak se stabilizuje
i napájecích 5V /7805)
Variantně vstup jen 5V
(externí napájení
je pro motory nutné!)
Signalizační LED
SPI DISPLAY
(max7219)

ULN2803
výkonový budič
sběrnice pro přímé
připojení dvou krokových
motorů (nebo 8 x relé)

WS/IO
další vstupy/výstupy
(např RGB LED WS)
senzory / signalizace



ESP32
jako hlavní řídící obvod
(může být i neosazen,
propojení externě
nebo I2C)
Variantně I2C master
s nadřazeným řídícím RPi

PCF8574
8-bit I2C expandér
pro pomalejší řízení
krokových motorů
nebo vstupně/výstupní
sběrnice

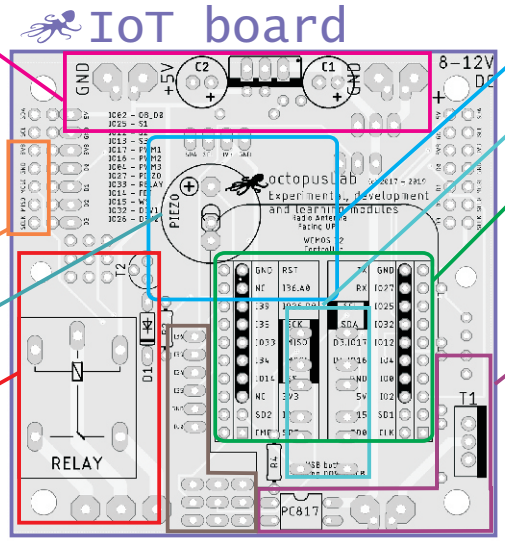
3x PWM výstupy
(např. pro servo motory)

Konektory dvou DC motorů
řízených h-můstekem L293D
(s ochrannými diodami)

IoT board

Napájecí část
konektor pro
výkonových 8-12V
(pak se stabilizuje
i napájecích 5V /7805)
Variantně vstup jen 5V
(externí napájení je pro
motor větráku vhodné)
Signalizační LED
SPI DISPLAY
(max7219)
se sběrnicíOCTOBUS
Piezzo pro akustickou
signalizaci

Relé (na 5V)
umožní spínat
výkonnější "zátěž"
(po propojení na stykač
může například ovládat
i přímotop)

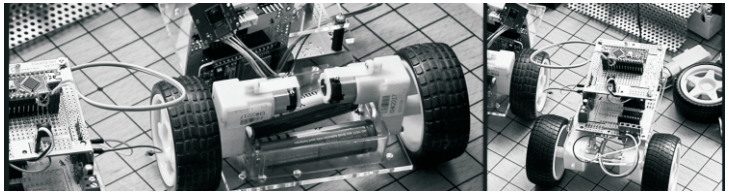


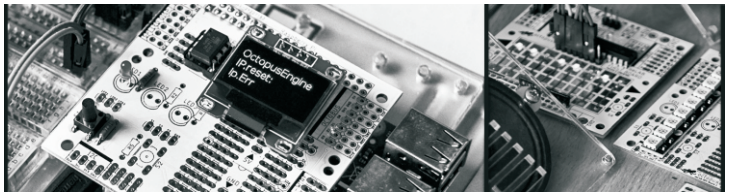
OLED I2C display
je na čelní straně
(pak už jenom 1-3
ovládací tlačítka)

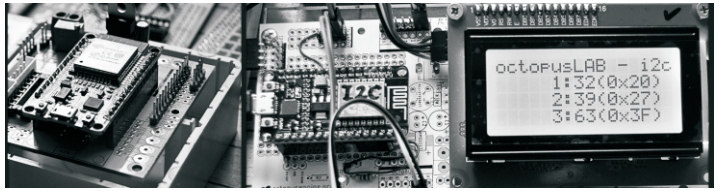
Modul Wemos ESP32
jako hlavní řídící obvod
je tentokrát z druhé
strany desky
(variantně lze
osadit i modul
Wemos 8266)

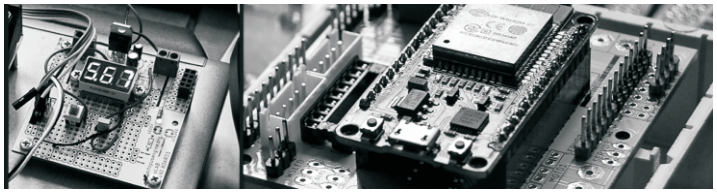
MosFet přes optočlen spínaný
PWM výkonný výstup.
Hlavní záměr pro LED pásek
nebo motor 12/24V větráku
nebo čerpadla

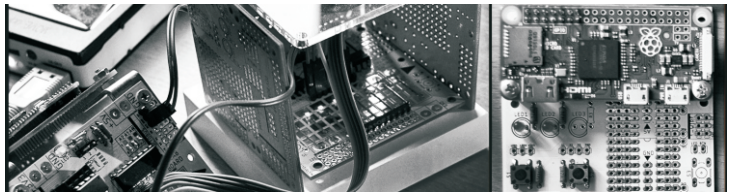
Další I/O piny
(DA...)











octopusLAB.cz
facebook/octopuslabcz
github/octopusengine | instagram/octopusengine

Děkuji za pozornost

Honza S. Čopák
honza.copak@gmail.com

octopusengine