Začínáme programovat Raspberry Pi Pico

Ing. Filip Šlapal, M.Sc. | Katedra teorie obvodů | slapafil@fel.cvut.cz

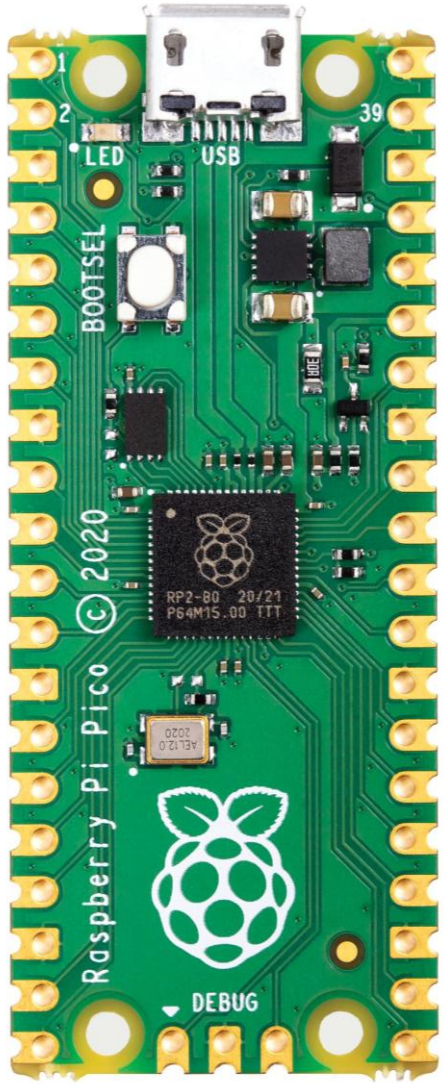


LÉKAŘSKÁ ELEKTRONIKA A BIOINFORMATIKA

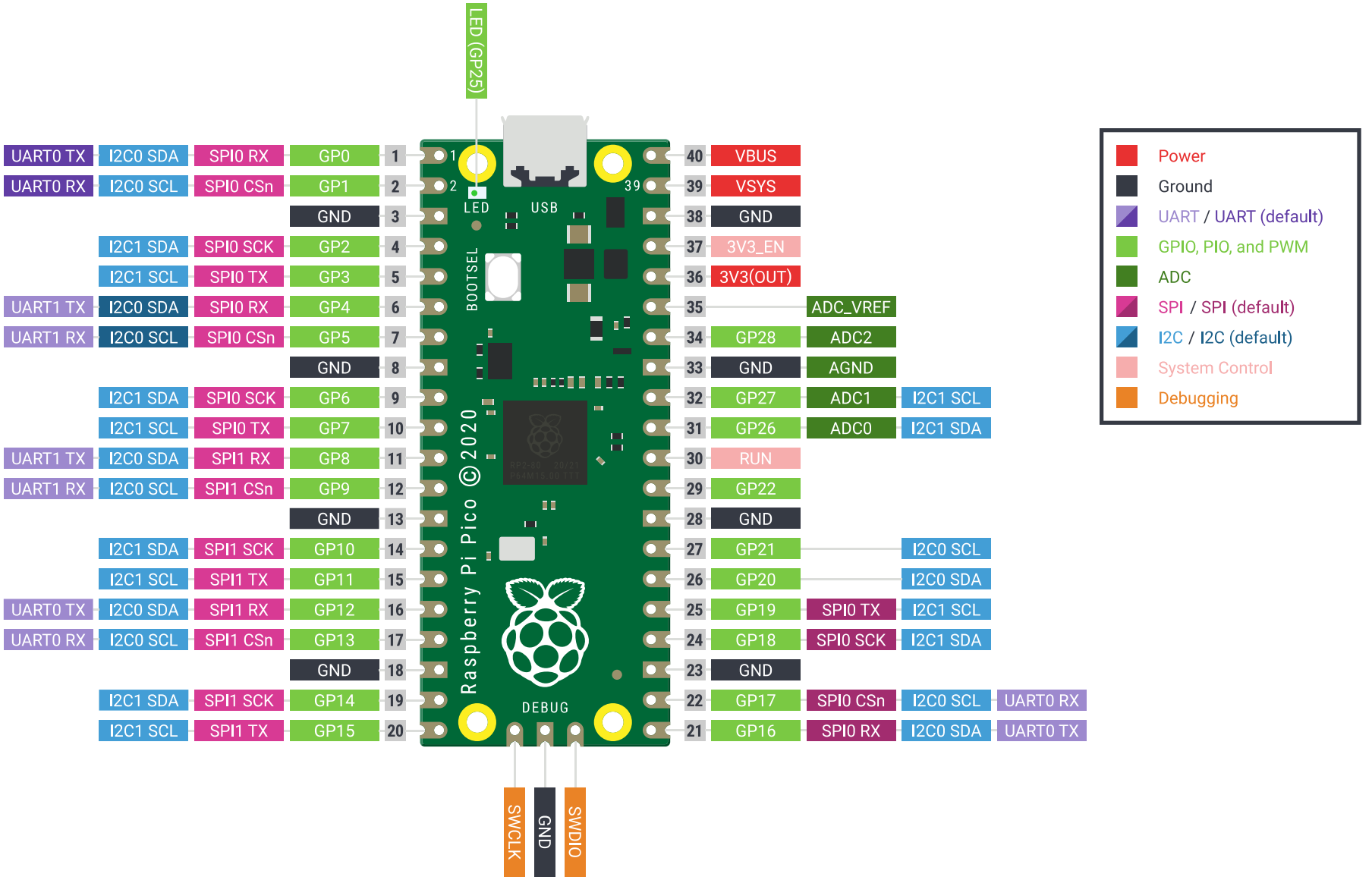
ČVUT ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Základní parametry

- Destička o rozměrech 21 x 51 mm
- Jednočipový počítač (mikrokontrolér) RP2040
- Dvoujádrový procesor Arm Cortex M0+
- 264 kB SRAM paměti
- 2MB QSPI flash paměti
- 26 multifunkčních GPIO pinů (3x AD převodník)
- Komunikační sběrnice
  - 2x UART, 2x SPI, 2x I2C, 16x PWM, 1x USB
- Vstupní napětí v rozsahu 1.8 - 5.5 V
- Vestavěný teploměr a LED dioda
- Možnost mass storage módu pro snadné programování
- Přesné hodiny, spánkové módy

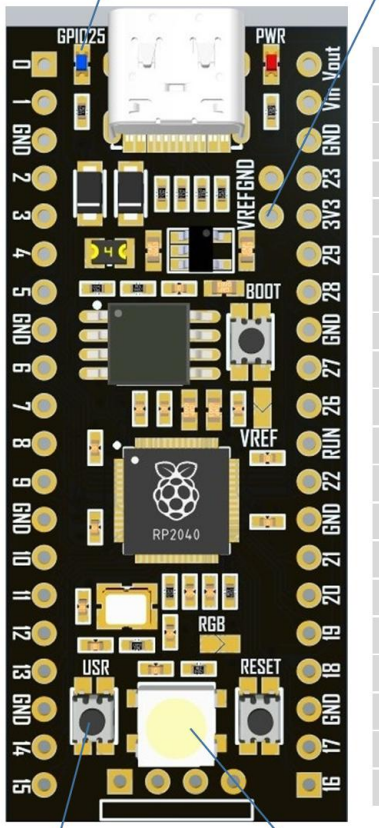


Pinout Rpi Pico



Pinout YD-RP2040

USART0 TX	I2C0 SDA	SPI0 RX	GP0	1
USART0 RX	I2C0 SCL	SPI0 CSn	GP1	2
			GND	3
	I2C1 SDA	SPI0 SCK	GP2	4
	I2C1 SCL	SPI0 TX	GP3	5
USART1 TX	I2C0 SDA	SPI0 RX	GP4	6
USART1 RX	I2C0 SCL	SPI0 SCn	GP5	7
			GND	8
	I2C1 SDA	SPI0 SCK	GP6	9
	I2C1 SCL	SPI0 TX	GP7	10
USART1 TX	I2C0 SDA	SPI1 RX	GP8	11
USART1 RX	I2C0 SCL	SPI1 CSn	GP9	12
			GND	13
	I2C1 SDA	SPI1 SCK	GP10	14
	I2C1 SCL	SPI1 TX	GP11	15
USART0 TX	I2C0 SDA	SPI1 RX	GP12	16
USART0 RX	I2C0 SCL	SPI1 CSn	GP13	17
			GND	18
	I2C1 SDA	SPI1 SCK	GP14	19
	I2C1 SCL	SPI1 TX	GP15	20



40	Vout			
39	Vin			
38	GND			
37	GP23			
36	3V3			
35	GP29	ADC3		
34	GP28	ADC2		
33	GND	AGND		
32	GP27	ADC1	I2C1 SCL	
31	GP26	ADC0	I2C1 SDA	
30	RUN			
29	GP22			
28	GND			
27	GP21		I2C0 SCL	
26	GP20		I2C0 SDA	
25	GP19	SPI0 TX	I2C1 SCL	
24	GP18	SPI0 SCK	I2C1 SDA	
23	GND			
22	GP17	SPI0 CSn	I2C0 SCL	USART0 RX
21	GP16	SPI0 RX	I2C0 SDA	USART0 TX

VCC-GND® Studio
源地®工作室

	Power
	Ground
	USART
	USART (default)
	GPIO and PIO
	ADC
	SPI
	I2C
	SYS Control
	Debugging

Programování

- Kódy můžeme psát ve dvou jazycích
  - C/C++
    - Potřeba kompilace kódu před nahráním do mikrokontroléru
    - Možnost programovat v Arduino IDE obdobně jako Arduino
  - MicroPython
    - Potřeba stažení MicroPythonu před nahráním .py souboru do mikrokontroléru
    - https://www.raspberrypi.com/documentation/microcontrollers/micropython.html
    - Prostředí Thonny umožňuje také nahrání Micropythonu do mikrokontroléru
- Raspberry poskytuje SDK pro oba jazyky
- Oficiální rozšíření do VS Code
  - umožňuje programovat v C/C++ i MicroPythonu

MicroPython

- Zmenšená a efektivní implementace Python 3
- Obsahuje malou podmnožinu standardních knihoven Pythonu
- Optimalizováno pro běh na mikrokontrolérech a v omezených prostře
- Jednoduchá instalace na Raspberry Pi Pico (Rpi Pico)
  - Stažení aktuální verze MicroPythonu pro RPi Pico (.uf2)
    - https://micropython.org/download/RPI_PICO/
  - Připojení Raspberry Pi Pico k PC v mass storage módu
    - Zmáčknutí tlačítka BOOTSEL
    - Zapojení kabelu s RPi Pico do PC
  - Připojení k YD-RP2040 k PC v mass storage módu
    - Zapojení kabelu s RPi Pico do PC
    - Zmáčknutí a držení tlačítka BOOTSEL
    - Zmáčnkutí tlačítka RST na 1-2 sekundy
    - Puštění tlačíka BOOTSEL
  - Přesunutí staženého MicroPythonu do RPi Pico
    - Stejně jako by se jednalo například o flash disk
  - Po úspěšném stažení se RPi Pico jeví v PC jako další COM port

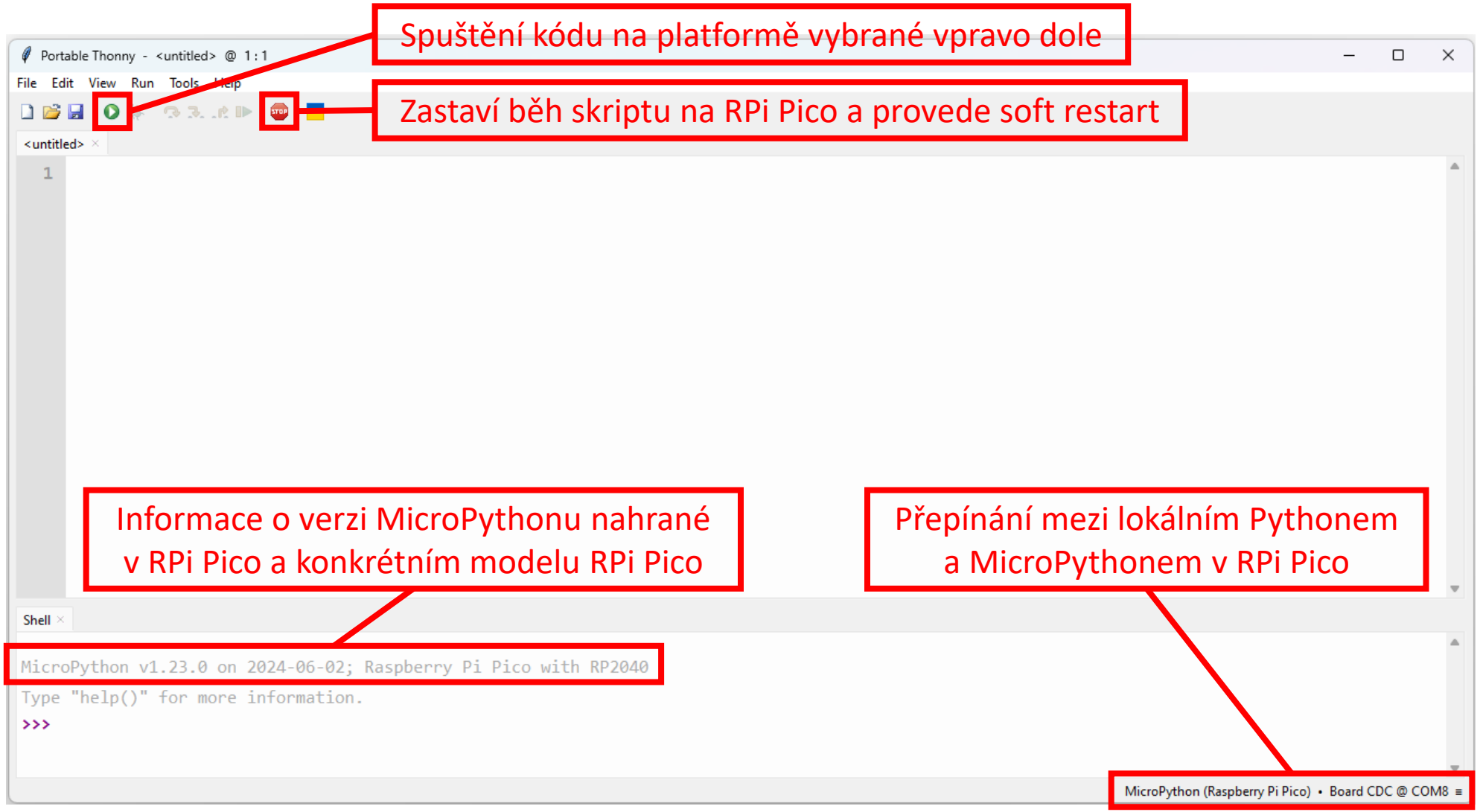


Thonny

- Jednoduché prostředí pro programování v Pythonu
- Umožňuje přímé nahrávání a ukládání kódů na RPi Pico



Thonny

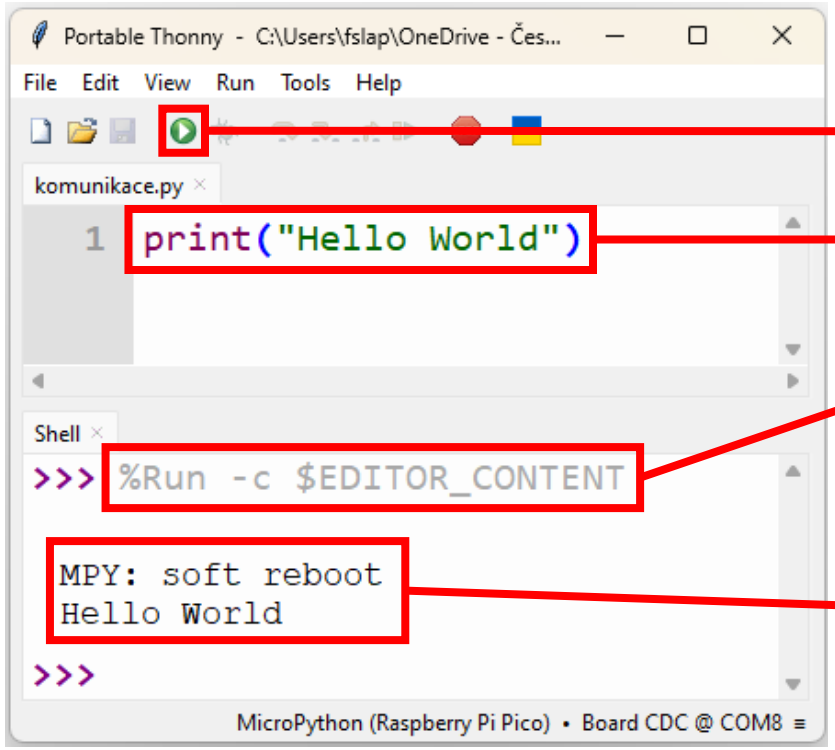

Spuštění kódu na platformě vybrané vpravo dole
Zastaví běh skriptu na RPi Pico a provede soft restart
Portable Thonny - <untitled> @ 1 : 1
File Edit View Run Tools Help
<untitled>
1
Informace o verzi MicroPythonu nahrané v RPi Pico a konkrétním modelu RPi Pico
Přepínání mezi lokálním Pythonem a MicroPythonem v RPi Pico
Shell
MicroPython v1.23.0 on 2024-06-02; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
MicroPython (Raspberry Pi Pico) • Board CDC @ COM8

Thonny

- Možnost uložení kódu na PC nebo RPi Pico
  - Pro uchování aktuálního skriptu v paměti RPi Pico a automatické spuštění po připojení k napájení je nutné skript uložit pod jménem main.py
- Možnost zobrazení grafu s hodnotami, které jsou posílány z RPi Pico
  - Menu View -> Plotter
  - Vpravo dole vedle konzole se zobrazí graf s aktuálními hodnotami
- Možnost spravování MicroPython knihoven v RPi Pico
  - Menu Tools -> Manage packages
  - Například pro zjednodušení připojení senzoru komunikujícího přes I2C

První program – Hello World

- MicroPython nahraný na RPi Pico
- Vyzkoušíme komunikaci přes USB mezi RPi Pico a PC
- Pošleme textový řetězec z RPi Pico do PC



Po uložení souboru necháme kód proběhnout na RPi Pico

Poslání textového řetězce „Hello World“ přes USB do PC

Po zmáčknutí tlačítka se provede tento kód, který nechá na RPi Pico proběhnout kód napsaný nahoře v okně editoru

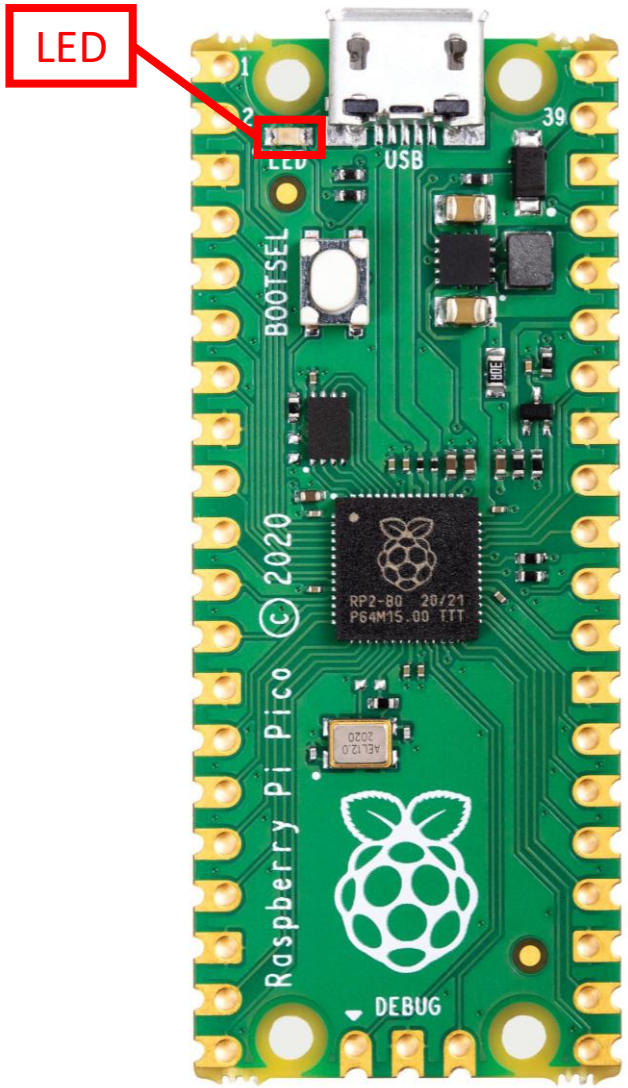
Zprávy z Rpi Pico, které dorazily po sériové lince na COM8
1. Proběhl soft reset
2. Přišel textový řetězec „Hello World“

Blikání LEDkou

- Využijeme vestavěnou LEDku
  - Připojena na pinu GP25
- Nadefinujeme pin GP25 jako výstupní (output)
- K ovládání LEDky využijeme knihovnu machine
  - Obsahuje funkce pro práci s GP piny

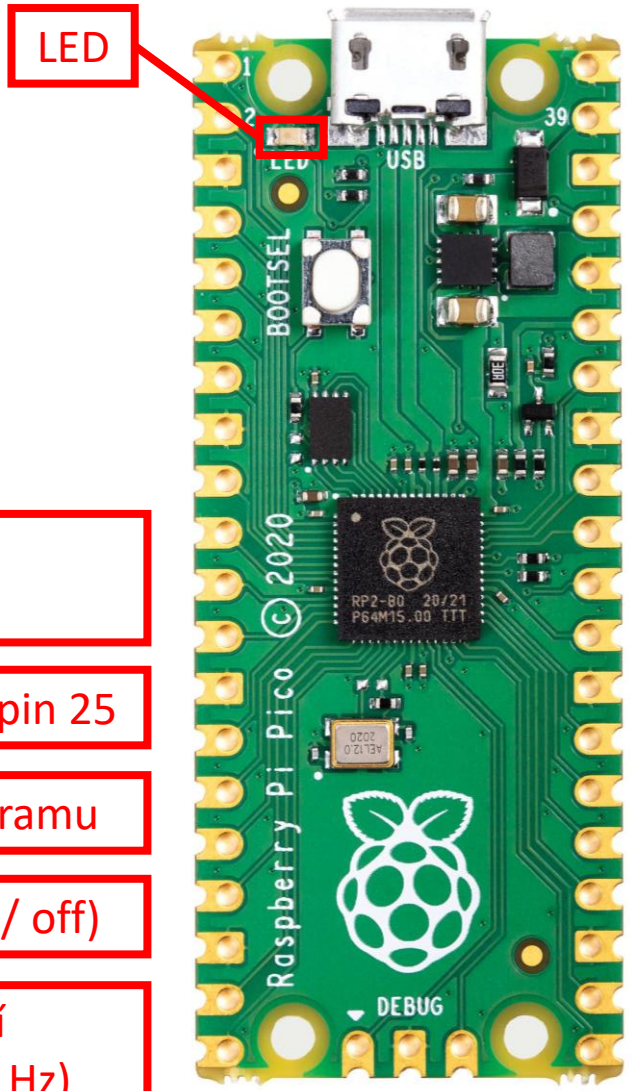
```
from machine import Pin
from time import sleep

led = Pin(25, Pin.OUT)
while True:
    led.toggle()
    sleep(0.5)
```



Blikání LEDkou

- Využijeme vestavěnou LEDku
  - Připojena na pinu GP25
- Nadefinujeme pin GP25 jako výstupní (output)
- K ovládání LEDky využijeme knihovnu machine
  - Obsahuje funkce pro práci s GP piny




```
from machine import Pin
from time import sleep

led = Pin(25, Pin.OUT)
while True:
    led.toggle()
    sleep(0.5)
```

Import knihoven pro ovládání LEDky a časového zpožedění

Definujeme objekt led jako výstupní GP pin 25

Nekonečná smyčka pro opakování programu

Převrácení napěťové úrovně LEDky (on / off)

Program čeká 0.5 s a nic neprovádí
= nastavení periody blikání 0.5 s (f = 2 Hz)

Regulování intenzity LEDky

- Využijeme vestavěnou LEDku
- Lidské oko má vzorkovací frekvenci kolem 30 Hz
  - Blikáním LEDkou pomocí pulzně šířkové modulace (PWM) o dostatečně vysoké frekvenci (desítky Hz) oko přesvědčíme, že LEDka svítí různou intenzitou

```
import time
from machine import Pin, PWM

pwm = PWM(Pin(25))
pwm.freq(50)
pwm20 = round(0.2*2**16)
pwm80 = round(0.8*2**16)
while True:
    pwm.duty_u16(pwm20)
    time.sleep(0.5)
    pwm.duty_u16(pwm80)
    time.sleep(0.5)
```

Import knihoven pro ovládání LEDky, PWM a časového zpožedění

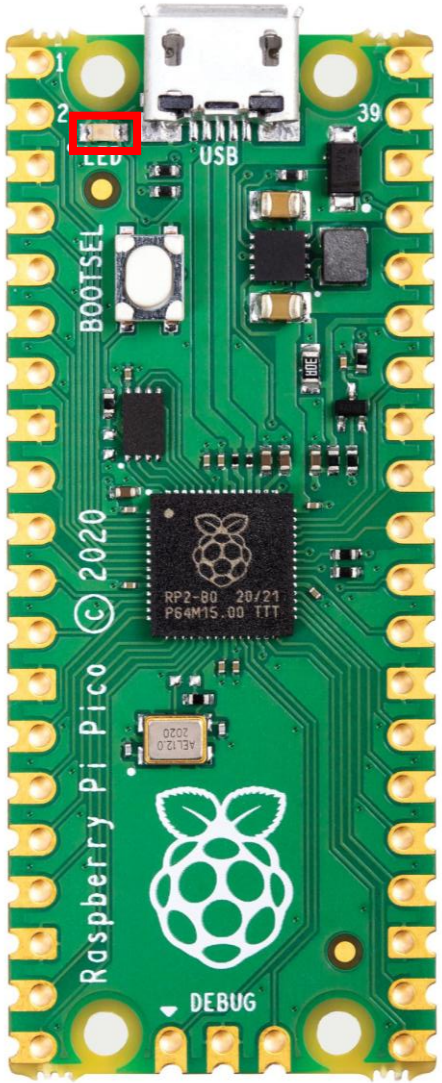
Definujeme objekt pwm jako PWM na GP pinu 25

Základní frekvence PWM 50 Hz (T = 20 ms)

Výpočet 0.2 a 0.8 z 16b maximální hodnoty

Nastavení duty cycle (střídy) na 20 % (0.2) = 20 % periody je signál v logické 1 ~ 3.3 V

Nastavení duty cycle (střídy) na 80 % (0.8) = 80 % periody je signál v logické 1 ~ 3.3 V

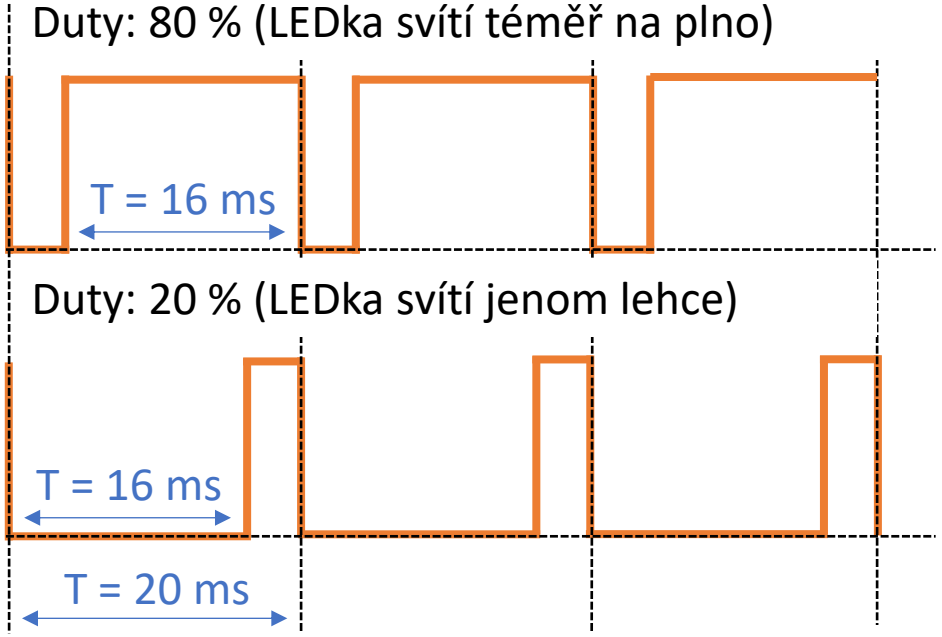


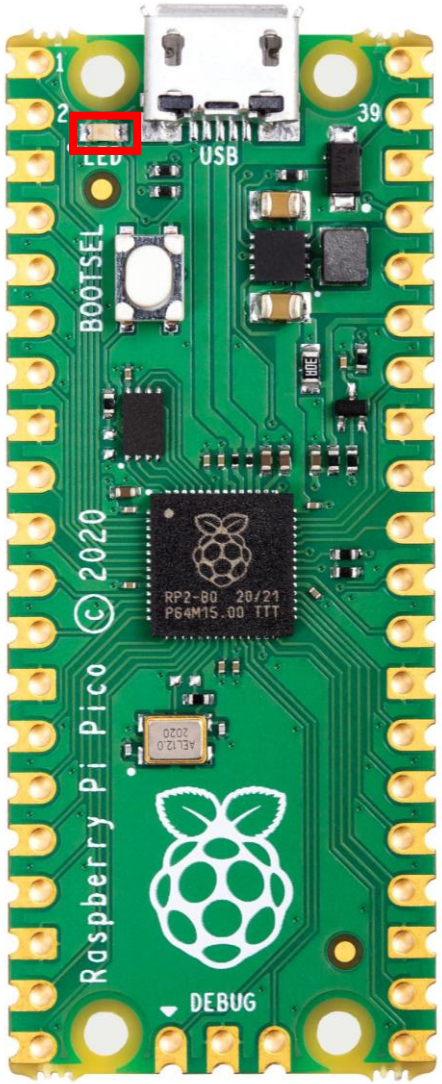
Rozsvěcení a zhasínání LEDky

- Využijeme vestavěnou LEDku
- Lidské oko má vzorkovací frekvenci kolem 30 Hz
  - Blikáním LEDkou pomocí pulzně šířkové modulace (PWM) o dostatečně vysoké frekvenci (desítky Hz) oko přesvědčíme, že LEDka svítí různou intenzitou

```
import time
from machine import Pin, PWM

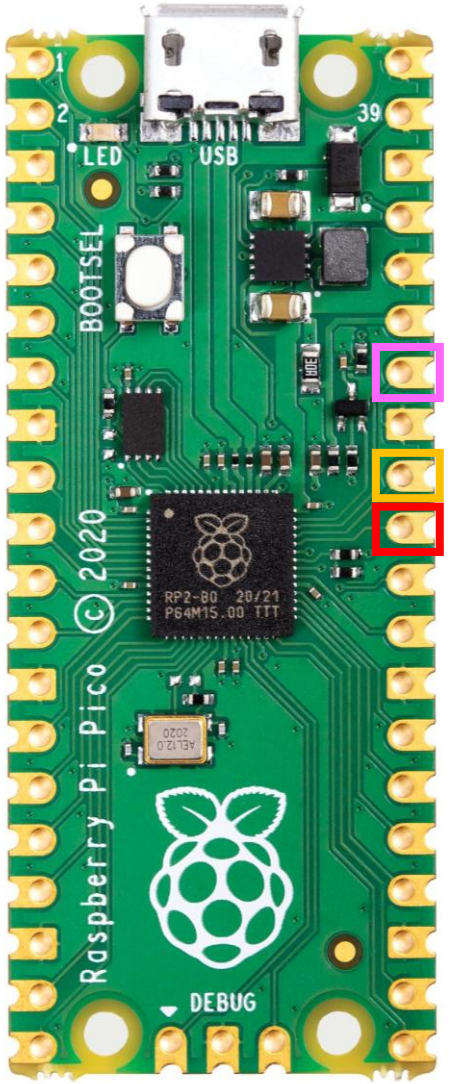
pwm = PWM(Pin(25))
pwm.freq(50)
pwm20 = round(0.2*2**16)
pwm80 = round(0.8*2**16)
while True:
    pwm.duty_u16(pwm20)
    time.sleep(0.5)
    pwm.duty_u16(pwm80)
    time.sleep(0.5)
```





AD převodníky

- K dispozici jsou čtyři AD převodníky
  - Pro převod analogového signálu na digitální
  - Využívají systému postupné aproximace (SAR)
  - Jeden kanál připojen na vnitřní teploměr
  - Tři kanály volně k dispozici pro měření napětí
    - GP26 (ADC0), GP27 (ADC1), GP28 (ADC2)



AD převodníky

- K dispozici jsou čtyři AD převodníky
  - Pro převod analogového signálu na digitální
  - Využívají systému postupné aproximace (SAR)
  - Jeden kanál připojen na vnitřní teploměr
  - Tři kanály volně k dispozici pro změření napětí
    - GP26 (ADC0), GP27 (ADC1), GP28 (ADC2)

```
from machine import ADC, Pin
import time

adc = ADC(Pin(26))
while True:
    print(adc.read_u16())
    time.sleep(0.02)
```

Import knihoven pro ovládání ADC a časového zpožedění

Definujeme objekt adc jako AD převodník na GP pinu 26

Změřené napětí jako 16b číslo (0 ~ 0 V, 65535 ~ 3.3 V) → odeslání do PC

Nastavení vzorkovací frekvence na 50 Hz (T = 20 ms)

