



Raspberry Pi Pico a MicroPython

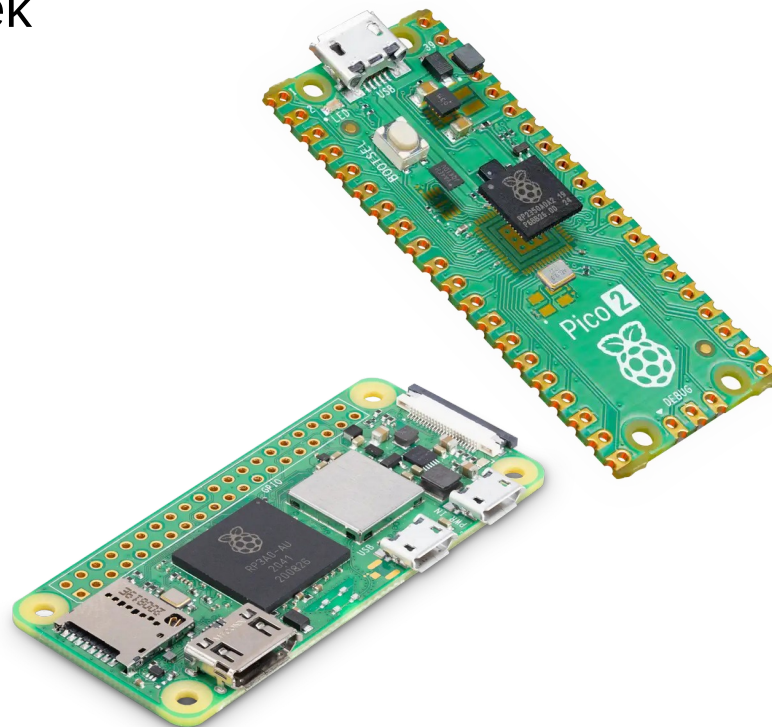
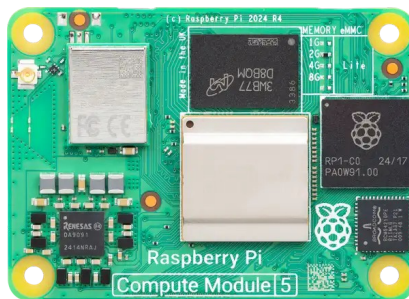
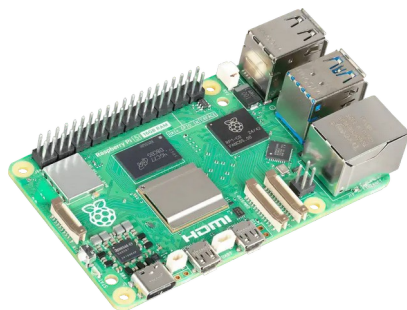
Martin Pěnička

Co se naučíte a budeme dělat ?

- Kurz pro úplné začátečníky – neočekávají se žádné znalosti a zkušenosti
- Mikrokontrolery a jednodeskové počítače
- Základy programování
- Vývojové prostředí, IDE
- Práce se vstupy, výstupy, senzory
- Domácí vybava
- A pak už jen hraní si

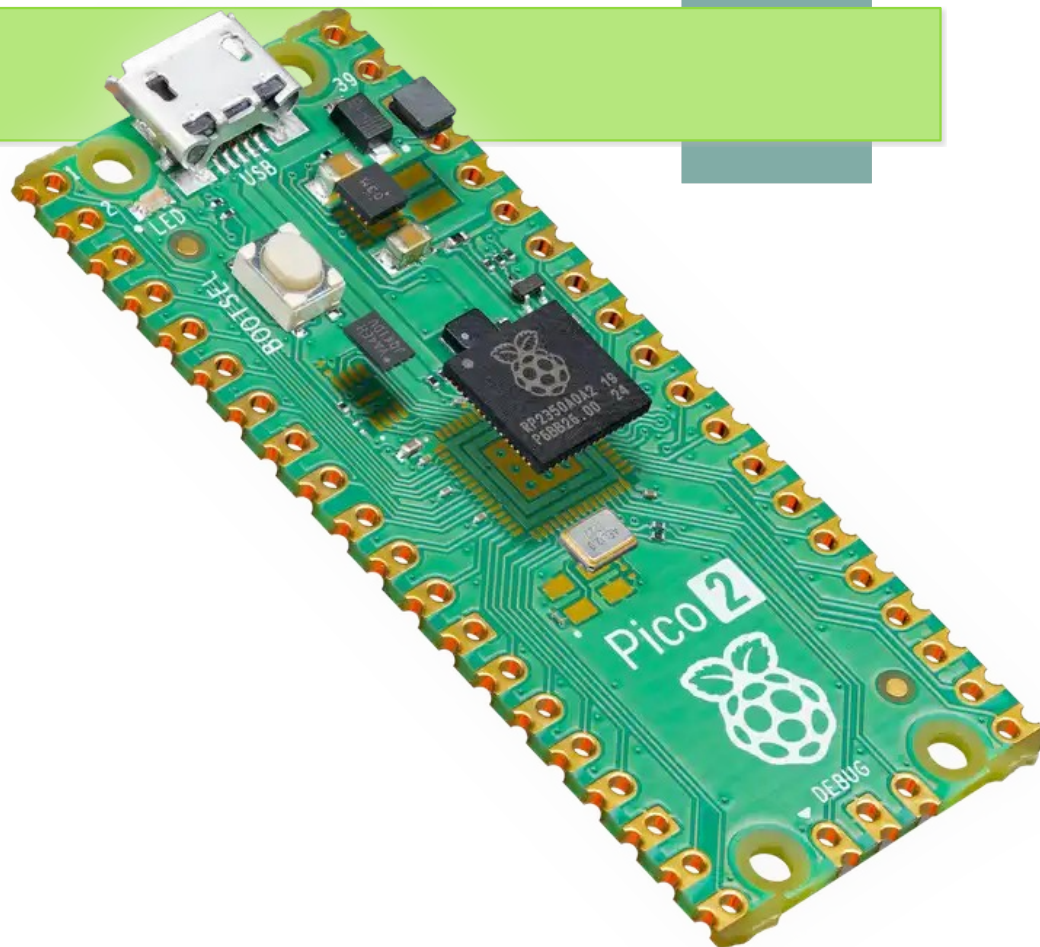
Platforma Raspberry Pi

- Vznik v roce 2012 jako výukový prostředek
- Jednodeskový počítač vs mikrontroler
- HATy



Raspberry Pi Pico

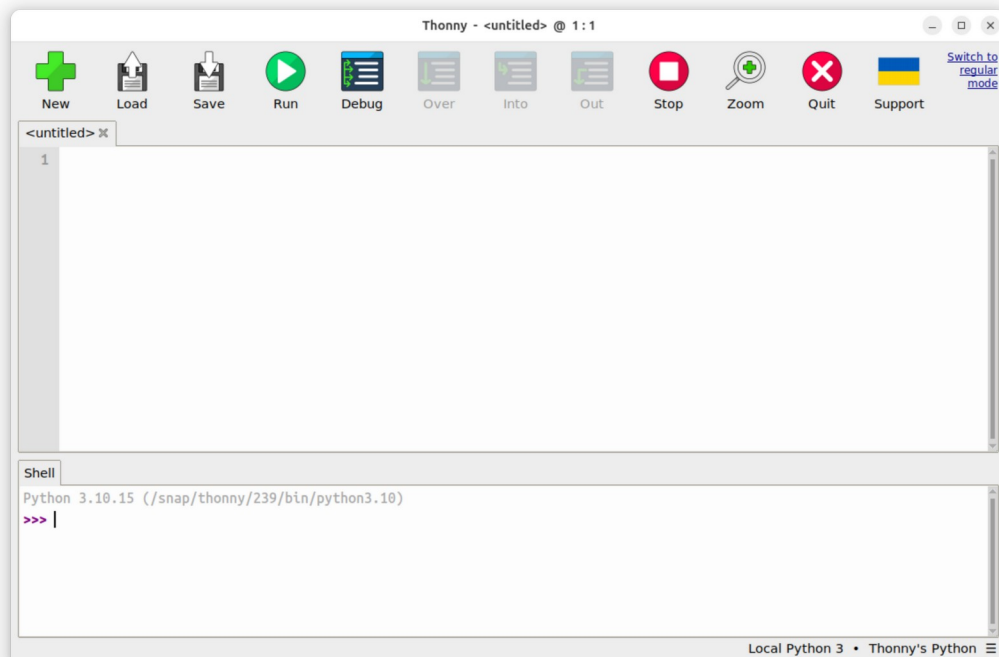
- Základní varianta, W a H
- Cena 100 – 200 Kč
- RP 2040 2 core ARM procesor
- 264 KB RAM, 2MB flash
- Logika na 3.3V (!)
- WiFi a Bluetooth pro W variantu
- Multifunkční piny
 - Digitální
 - Analogové
 - Sběrnice I2C, SPI, UART
 - PWM



Co si pořídit domů ?

- Mikrokontroler – Raspberry Pi Pico (Arduino, micro:bit, ...)
- Nepájivé pole
- Sadu propojovacích drátů
- Základní sadu součástek – LED, tlačítka, odpory, senzory
- Nářadí
- Multimetr, později zdroj
- Vše potřebné poskládáte do 1000Kč

Vývojové prostředí - Thonny IDE



MicroPython / Python a programování

- Python je univerzální programovací jazyk vhodný pro začátečníky
- Jednoduchá syntaxe
- Jednoduchý začátek s programováním – žádná kompilace a příprava prostředí
- MicroPython je odlehčená varianta pro mikrokontrolery
- Pozor – knihovny které možná znáte z Python spíše v microPythonu nečekejte

Základy programování - blikání ledkou

```
from machine import Pin  
import time
```

```
# import knihoven
```

```
print("ahoj")  
led = Pin(25, Pin.OUT)
```

```
# výpis do konzole  
# definice proměnné
```

```
while True:  
    led.toggle()  
    time.sleep(1)
```

```
# nekonečná smyčka  
# přepnutí stavu ledky  
# uspání programu
```


Blikání více ledkami, posuny

```
from machine import Pin
import time
```

```
led1 = Pin(4, Pin.OUT)
led2 = Pin(5, Pin.OUT)
led3 = Pin(6, Pin.OUT)
duration = 0.5
```

```
while True:
    led1.value(1)
    time.sleep(duration)
    led1.value(0)
    time.sleep(duration)
    led2.value(1)
    time.sleep(duration)
    led2.value(0)
    time.sleep(duration)
```

Pole a cykly - úvod

- Více ledek = delší téměř totožný kód, není udržitelné
- Řešení jsou pole a cykly
- Pole umožňuje pracovat s více objekty dohromady
- Cyklus umožňuje volat stejný kód na více objektech
- Typy
 - While
 - for

Pole a cykly - ukázka

```
pins = [4, 5, 6]
```

```
leds = []
```

```
for pin in pins:
```

```
    leds.append(Pin(pin, Pin.OUT))
```

Pole a cykly – blikání ledkami

```
from machine import Pin
import time
```

```
pins = [4, 5, 6]
leds = []
```

```
for pin in pins:
    leds.append(Pin(pin, Pin.OUT))
```

```
duration = 0.5
while True:
    for led in leds:
        led.value(1)
        time.sleep(duration)
        led.value(0)
        time.sleep(duration)
```

Definice funkcí

- Funkce umožňuje vyčlenit část kódu a opakovaně ho volat z jiných částí
- Lze také umístit do jiných souborů

```
def blink(dur):
```

```
    for led in leds:
```

```
        led.value(1)
```

```
        time.sleep(dur)
```

```
        led.value(0)
```

```
        time.sleep(dur)
```

Cvičení

- Vytvořte si pomocí funkcí několik různých blikacích metod
- Všechny najednou, postupné naplnění, posun jedné/více ledek
- Ledky inicializujte pomocí pole a cyklu, čas čekání je předán do funkce zvenku
- Spustěte postupně blikací programy, měňte čas čekání

Čtení digitálních vstupů a tlačítko

```
from machine import Pin
import time
```

```
led = Pin(4, Pin.OUT)
button = machine.Pin(3, machine.Pin.IN, machine.Pin.PULL_UP)
```

```
while True:
    if button.value() == 0:
        led.value(1)
    else:
        led.value(0)
    time.sleep(0.2)
```

Konstrukce if, else, elif

- Podmíněné vykonání podprogramu
- Umožňuje větvit program na základě podmínek

Cvičení - Přepínání blikacího programu pomocí tlačítka

- Vytvořte několik blikacích programů
- Po stisku tlačítka se zapne následující
- Složitější – každý program má svoje číslo, opakovaným stiskem tlačítka se zvolí následující blikací program
- Zkuste se stávajícími znalostmi, pro bezproblémový chod budeme potřebovat novou znalost - přerušení

Nestíhá to číst tlačítko ? Přerušení !

- Přerušení umožňuje definovat funkci které je zavolaná okamžitě po definované události – třeba stisku tlačítka

```
mode = 0
```

```
def button_click(pin):
```

```
    global mode
```

```
    if mode == 0:
```

```
        mode = 1
```

```
        return
```

```
    Mode = 0
```

```
button.irq(trigger=machine.Pin.IRQ_FALLING, handler=button_click)
```

Přerušení - debouncing

- Lze řešit hardwarově – přidáním rezistoru a kondenzátoru
- Softwarově – měření času mezi stisky pomocí `utime.ticks_ms()` a zapamatování si času posledního stisku

Cvičení - Ovládání ledek tlačítkem - posun

- Napište program který na stisk tlačítka posune svítící ledky o jedno místo
- V případě že dojde na konec řady, objeví se na druhé straně

Velké cvičení – blikací program

- Zadání:
 - Mikrokontroler obsahuje několik blikacích režimů
 - Prvním tlačítkem zvolím program, ledky jsou použity jako ukazatel který program je vybrán
 - Stiskem druhého tlačítka ho zapnu, druhým stiskem vypnu
 - Budete muset použít všechny dosavadní konstrukce, přerušení a debouncing jinak to nebude fungovat hezky

Analogové vstupy a teploměr

- Digitální vstupy pracují s hodnotou 0/1
- Analogové vrací hodnotu
 - 0 – 255 pro 8bit
 - 0 – 65535 pro 16bit
- Lze měřit spojitě veličiny – hodnotu napětí, teplotu, ...
- Lze vyslat specifickou hodnotu – intenzita světla ledky

Teploměr - příklad

```
from machine import Pin
import utime
```

```
sensor_temp = machine.ADC(4)
conversion_factor = 3.3 / 65535
```

```
while True:
```

```
    reading = sensor_temp.read_u16()
    voltage = reading * conversion_factor
    temperature = 27 - (voltage - 0.706) / 0.001721
    print("Teplota:", temperature, "°C")
    utime.sleep(1)
```

Závěr

- Naučili jsme se základní práci s Raspberry Pi Pico
- Základy programování v Pythonu a pokročilejší konstrukce
- Práci s digitálními vstupy a výstupy
- Problémy při použití vstupů – přerušení, debouncing
- Analogové vstupy