

Využití vývojové desky Raspberry Pi Pico pro výuku programování mikropočítačů

Dominik Strnad

Bakalářská práce
2025



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence
Akademický rok: 2024/2025

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Dominik Strnad
Osobní číslo: A22662
Studijní program: B0613A140020 Softwarové inženýrství
Forma studia: Prezenční
Téma práce: Využití vývojové desky Raspberry Pi Pico pro výuku programování mikropočítačů
Téma práce anglicky: Using the Raspberry Pi Pico Development Board for Teaching Microcontrollers Programming

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Navrhněte vhodnou hardwarovou platformu umožňující vytvoření ukázkových aplikací pro obsluhu běžných periférií mikropočítače.
3. Vytvořte sadu jednoduchých ukázkových aplikací s využitím navržené hardwarové platformy.
4. K vytvořeným aplikacím zpracujte dokumentaci popisující jejich implementaci.
5. Vytvořte komplexní ukázkový projekt demonstrující možnosti vývojové desky.

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. CATSOULIS, John. Designing embedded hardware. 2nd ed. Sebastopol, CA: O'Reilly, 2005. ISBN 0596007558.
2. DOGAN, Ibrahim. Raspberry Pi Pico Essentials. Elektor-Verlag, 2021. ISBN 3895764272.
3. PINKER, Jiří. Mikroprocesory a mikropočítače. 1. vyd. Praha: BEN – technická literatura, 2004. ISBN 80-7300-110-1.
4. RASPBERRY PI FOUNDATION. Getting started with Raspberry Pi Pico [online]. 2024 [cit. 2024-10-16]. Available online from: <https://projects.raspberrypi.org/en/projects/getting-started-with-the-pico>.
5. WHITE, Elecia. Making embedded systems. Sebastopol: O'Reilly, 2012. ISBN 9781449302146.

Vedoucí bakalářské práce: **Ing. Jan Dolinay, Ph.D.**
Ústav automatizace a řídicí techniky

Datum zadání bakalářské práce: **27. října 2024**
Termín odevzdání bakalářské práce: **2. června 2025**



doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

L.S.

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 17. listopadu 2024

PROHLÁŠENÍ AUTORA BAKALÁŘSKÉ PRÁCE

Beru na vědomí, že

- odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb., v platném znění bez ohledu na výsledek obhajoby;
- bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému a bude dostupná k nahlédnutí;
- jedno vyhotovení bakalářské práce v listinné podobě bude ponecháno Univerzitě Tomáše Bati ve Zlíně k uložení;
- na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- podle § 60 odst. 1 autorského zákona má Univerzita Tomáše Bati ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- podle § 60 odst. 2 a 3 mohu užít své dílo – bakalářskou práci – nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tj. k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá; neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji, že

- jsem na bakalářské práci pracoval(a) samostatně a použitou literaturu jsem řádně citoval(a); v případě publikace výsledků budu uveden(a) jako spoluautor;
- odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou obsahově totožné.

Ve Zlíně, dne

.....
podpis autora

ABSTRAKT

Tato práce se zaměřuje na mikrokontroler Raspberry Pi Pico a jeho praktické využití při výuce programování mikropočítačů. Cílem práce je představit možnosti tohoto mikrokontroleru a možnosti jeho programování. Součástí práce je i výběr hardwarové platformy, která umožní snadné připojování a ovládání externích periferních zařízení. Pro tuto zvolenou hardwarovou platformu byly napsány ukázkové programy v jazyce C.

Klíčová slova: Raspberry Pi Pico, mikrokontrolér, mikropočítač, embedded systémy

ABSTRACT

This work focuses on microcontroller Raspberry Pi Pico and its practical usage in teaching microcontroller programming. The aim of the work is to present the capabilities of this microcontroller and its programming options. The thesis also includes the selection of hardware platform that allows easy connection and control of external peripheral devices. Sample programs written in the C programming language were developed for the chosen platform.

Keywords: Raspberry Pi Pico, microcontroller, microcomputer, embedded system

Rád bych poděkoval panu Ing. Janu Dolinayovi, Ph.D. za odborné vedení, ochotu a cenné připomínky při psaní této práce.

OBSAH

ÚVOD	13
I TEORETICKÁ ČÁST	14
1 VÝVOJOVÁ DESKA RASPBERRY PI PICO.....	15
1.1 RASPBERRY PI.....	15
1.2 VÝVOJOVÁ DESKA RASPBERRY PI PICO.....	16
1.2.1 Pico 1	17
1.2.2 Pico 2	17
1.2.3 Alternativy Raspberry Pi Pico.....	17
1.3 ARCHITEKTURA PROCESORŮ NA RASPBERRY PI PICO.....	18
1.3.1 ARM	18
1.3.2 RISC-V	18
1.4 MOŽNOSTI PROGRAMOVÁNÍ.....	19
1.4.1 C/C++	19
1.4.2 MicroPython.....	19
1.4.3 PIO.....	20
1.5 PICOPAD	21
1.6 POROVNÁNÍ RASPBERRY PI PICO S ARDUINO UNO	22
2 POUŽITÍ RASPBERRY PI PICO VE VÝUCE.....	23
2.1 PROČ RASPBERRY PI PICO	23
2.2 RASPBERRY PI PICO SDK.....	23
2.3 DOSTUPNÉ MATERIÁLY PRO VÝUKU	24
2.3.1 Oficiální materiály.....	24
2.3.2 Výukové materiály v anglickém jazyce.....	25
2.3.3 Výukové materiály v českém jazyce	25
3 NÁVRH HARDWAROVÉ PLATFORMY PRO VÝUKU.....	26
3.1 DOSTUPNÁ KOMERČNÍ ŘEŠENÍ.....	27
3.1.1 Gravity Expansion Board.....	27
3.1.2 Grove Shield for Pi Pico v1.0	28
3.1.3 Maker Pi Pico Mini.....	28
3.1.4 Pico Breadboard KIT	29
3.2 ARDIPI.....	30
3.3 EXPLORER BOARD.....	30
3.4 KOMPONENTY ZVOLENÉ ROZŠÍŘUJÍCÍ DESKY EXPLORER BOARD	31
3.4.1 Tlačítka	31
3.4.2 Bzučák	32
3.4.3 RGB LED diody	32
3.4.4 Relé.....	32
3.4.5 TFT displej.....	32

3.4.6	DHT 11	33
3.4.7	LED diody	33
3.4.8	Analogové vstupy.....	33
II	PRAKTICKÁ ČÁST	34
4	PRÁCE S PROSTŘEDÍM VE VISUAL STUDIO CODE	35
4.1	NASTAVENÍ PROSTŘEDÍ.....	35
4.2	PROGRAMOVÁNÍ SAMOSTATNÉHO PICA	36
4.3	PROGRAMOVÁNÍ PŘES DEBUG PROBE	37
4.4	LADĚNÍ PROJEKTU.....	39
4.5	ZALOŽENÍ VLASTNÍHO PROJEKTU	39
5	ZAPOJENÍ.....	41
5.1	TLAČÍTKA.....	41
5.2	LED DIODY	41
5.3	POTENCIOMETR.....	42
5.4	DHT11.....	42
5.5	TFT DISPLEJ	42
5.6	NEOPIXEL LED	43
5.7	RELÉ	43
5.8	BZUČÁK	43
6	PROGRAMOVÁNÍ PERIFERIÍ	44
6.1	ZÁKLADNÍ STRUKTURA PROGRAMU PRO MIKROPOČÍTAČE.....	44
6.2	KONFIGURACE CMAKE.....	45
6.3	GPIO	47
6.4	AD PŘEVODNÍK.....	49
6.5	SOFTWAREVÉ ZPOŽDĚNÍ.....	49
6.6	ODEZVA PROGRAMU	50
6.7	ZÁKMIT TLAČÍTEK	51
6.8	STAVOVÝ AUTOMAT	52
6.9	RTOS.....	52
6.10	ČÍTAČ A ČASOVAČ.....	53
6.11	PŘERUŠENÍ	53
6.12	PWM	53
7	UKÁZKOVÉ PROJEKTY	55
7.1	IMPORT UKÁZEK Z GITHUBU DO VS CODE	55
7.2	POPIS UKÁZKOVÝCH PROJEKTŮ	55
7.2.1	Základní ukázky	56

7.2.2	Senzory	56
7.2.3	Výstupní zařízení.....	56
7.2.4	Řízení LED diod.....	56
7.2.5	Pokročilé ukázky	56
7.2.6	Komplexní ukázkový program.....	57
8	ZÁVĚR	58
	SEZNAM POUŽITÉ LITERATURY	61
	SEZNAM OBRÁZKŮ	64
	SEZNAM TABULEK	65
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	66
	SEZNAM PŘÍLOH	68

ÚVOD

Mikrokontroléry tvoří základní stavební kámen moderní elektroniky a vestavěných systémů. Nacházejí své uplatnění od domácích spotřebičů až po složité průmyslové řídicí systémy. S rostoucí oblíbeností chytrých a vzájemně propojených zařízení – tzv. Internetu věcí (IoT), se mikrokontroléry stávají stále důležitější i v oblasti vzdělávání, kde představují ideální prostředek pro pochopení základních principů elektroniky a programování.

Jednou z nejznámějších platforem, která si získala popularitu mezi studenty, pedagogy i hobby uživateli, je Arduino. Tato otevřená platforma si klade za cíl zpřístupnit programování mikrokontrolérů širší veřejnosti, zejména studentům a začínajícím vývojářům. Díky své jednoduchosti, přehlednému vývojovému prostředí a rozsáhlé komunitě je k dispozici velké množství návodů, knihoven a hotových projektů.

Na tento trend navazuje i nadace Raspberry Pi Foundation, známá především svými jednodeskovými počítači Raspberry Pi. V roce 2021 byla představena vývojová deska Raspberry Pi Pico, který reaguje na potřebu levné, otevřené a výkonné platformy pro výuku i praktické využití. Mezi podporované jazyky patří kromě C a C++ také jazyk MicroPython, který je vhodný i pro úplné nováčky v tomto odvětví. Pico se vyrábí i ve variantách s vestavěným modulem pro Wi-Fi a Bluetooth, což může zvyšovat zájem ze strany studentů.

Samotné Raspberry Pi Pico lze programovat i bez připojení dalších komponent, nicméně přidáním externích periferních zařízení, jako tlačítka, senzory nebo displeje, lze značně rozšířit možnosti programů. Připojení většího množství těchto zařízení pomocí vodičů může být poněkud nepřehledné, případně může vzniknout chyba v zapojení. Elegantním řešením může být hardwarová platforma s vestavěnými periferiemi. Díky tomu odpadne potřeba připojování přes vodiče, zvýší se pohodlnost ovládání komponent a bude zaručena funkčnost ukázkových příkladů při použití jednotné platformy.

Pro snadnější použití hardwarové platformy by bylo vhodné, aby pro ni byly k dispozici ukázkové programy v jazyce C, které by popisovaly základní práci s periferiemi této hardwarové platformy.

Tato práce si klade za cíl vybrat hardwarovou platformu pro výukové účely, vytvořit pro ni sadu snadno použitelných ukázkových programů a zpracovat dokumentaci popisující implementaci těchto ukázek.

I TEORETICKÁ ČÁST

1 VÝVOJOVÁ DESKA RASPBERRY PI PICO

V posledních letech došlo k výraznému rozvoji mikrokontrolerových platforem, které jsou cenově dostupné, snadno programovatelné a nachází široké využití – od výuky a domácích projektů až po prototypování a průmyslové nasazení. Mezi nejznámější platformy patří Arduino, která si díky jednoduchému prostředí a rozsáhlé komunitě získala oblibu především mezi studenty a hobby uživateli. Na Arduino se snaží navázat i firma Raspberry Pi se svým novým mikrokontrolérem.

1.1 Raspberry Pi

Raspberry Pi je série malých jednodeskových počítačů vyvinutých ve Spojeném království. Britský inženýr Eben Upton pracoval mezi lety 2006 a 2011 na projektu malého jednodeskového počítače. Vysoké ceny počítačů ve Spojeném království odrazovaly mladé studenty od studia informačních technologií. V roce 2009 založil Eben Upton nadaci Raspberry Pi Foundations, jež chce více zpřístupnit programování pro mladé studenty. Hlavním cílem bylo vytvořit počítač s cenou okolo 30 dolarů, což by školám usnadnilo výuku počítačů a studenti by se více zajímali o informační technologie. [14]

Raspberry Pi od roku 2012 navrhuje jednodeskové počítače přibližně o velikosti kreditní karty. Tyto počítače jsou založené na architektuře ARM, umožňují spuštění OS Linux a mohou být levnější alternativou běžných PC. [1]

Jméno Raspberry bylo zvoleno, protože názvy ovoce mají v počítačových firmách dlouhou tradici. „Pi“ je zkomolené slovo Python, jež odkazuje na použití Pythonu v prvních zařízeních. [14]

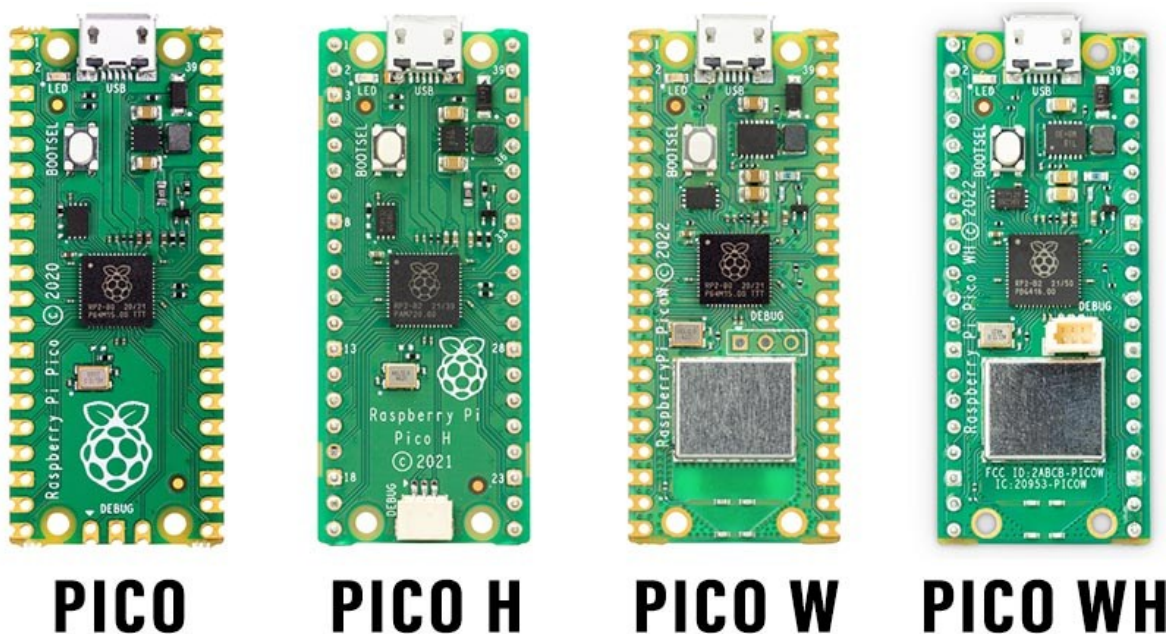


Obrázek 1 Raspberry Pi 5
(zdroj: [1])

1.2 Vývojová deska Raspberry Pi Pico

Přestože původní jednodeskové počítače Raspberry Pi umožňovaly ovládat i GPIO piny, Raspberry Pi přišlo s novou řadou velice levných vývojových desek Raspberry Pi Pico. Na rozdíl od jednodeskových počítačů tyto mikropočítače nevyžadují operační systém a jsou určeny pro embedded systémy.

V základní variantě je možno Pico osadit hřebenou pinovou nebo dutinkovou lištou, ale také je možno Pico připájet přímo na povrch plošného spoje díky drobným výřezům. Varianta Pico H je již od výroby osazena pinovou lištou a konektorem pro ladění. Další variantou je Pico W, která obsahuje navíc Wi-Fi a Bluetooth modul. Pico WH je potom kombinací již zmíněných variant. Pro model Pico 2 existuje taktéž varianta Pico 2 W. [12]



Obrázek 2 Varianty Raspberry Pi Pico
(zdroj: [2])

Všechny varianty obou modelů Pico a Pico 2 mají shodné rozložení pinů, což umožňuje záměnu různých modelů v projektech. Verze s Wi-Fi neovládá zabudovanou LED diodu přímo přes GPIO, ale pomocí funkce Wi-Fi čipu (je potřeba jen drobná změna v kódu). Pico má vyvedeno 26 multifunkčních GPIO pinů, které nabízí:

- 3×12 bitový AD převodník
- $26 \times$ PWM kanálů
- $2 \times$ SPI
- $2 \times$ I²C
- $2 \times$ UART [16]

1.2.1 Pico 1

Raspberry Pi Pico je vývojová deska založená na mikropočítači Raspberry Pi RP2040 s Flash pamětí o velikosti 2 MB. RP2040 obsahuje 32bitový dvoujádrový procesor Arm Cortex M0+, který běží na frekvenci 133 MHz. [16]

1.2.2 Pico 2

Raspberry Pi Pico 2 je založeno na novějším mikropočítači RP2350 s duálním procesorem a Flash pamětí o velikosti 4 MB. Procesor běží na frekvenci 150 MHz a umožňuje volbu mezi dvěma jádry Arm Cortex-M33 nebo dvěma jádry procesoru Hazard3 s architekturou RISC-V. Jádra Arm a RISC-V nelze používat současně a je třeba si mezi nimi vybrat. Přestože je mikropočítač RP2350 výkonnější než předchůdce RP2040, má překvapivě nižší udávanou spotřebu. [16]

1.2.3 Alternativy Raspberry Pi Pico

Na trhu existuje mnoho alternativních řešení od dalších výrobců. Některé produkty se snaží Pico napodobovat svým vzhledem, ale přesto se mohou lišit rozložení pinů. Takovéto vývojové desky jsou většinou založeny na stejných mikropočítačích RP2040 a RP2350, jako má právě Pico. Také mohou obsahovat navíc vestavěnou RGB diodu nebo vestavěné tlačítko. Na trhu existují i desky s integrovaným kulatým displejem pro výrobu vlastních chytrých hodinek.

1.3 Architektura procesorů na Raspberry Pi Pico

V oblasti mikroprocesorů se lze setkat se dvěma hlavními typy procesorových architektur: CISC a RISC. Obě architektury mají své specifické vlastnosti, výhody a nevýhody, které mají značný vliv na výkon, spotřebu energie a velikost kódu.

Procesory s architekturou CISC (Complex Instruction Set Computing) mají mnoho instrukcí, které mohou vykonávat. Tyto instrukce mohou mít různou délku i dobu vykonání. Některé složitější instrukce sice umožňují napsat méně řádků kódu, ale celý procesor je velice složitý, může být pomalejší a mít větší spotřebu. [24]

V reakci na tyto nevýhody CISC procesorů byla vyvinuta nová architektura procesorů zvaná RISC (Reduced Instruction Set Computer). Tyto procesory mají zjednodušenou sadu instrukcí. Složitější instrukce je možno poskládat z několika jednodušších a rychlejších instrukcí. Většinu instrukcí lze provést během jednoho cyklu. Procesory RISC jsou díky tomu značně jednodušší a efektivnější. Nevýhodou je potřeba provádět více instrukcí, ale celkový výkon je vyšší díky zjednodušenému návrhu, rychlejšímu dekódování instrukcí a menší dobou vykonávání jednotlivých instrukcí. Procesory RISC používají architekturu typu load/store, což znamená, že přístup do paměti je umožněn pouze dvěma typům instrukcí (load a store). Všechny ostatní instrukce pracují výhradně s registry, a ne přímo s pamětí. [24]

1.3.1 ARM

Architektura ARM (Advanced RISC Machine) vychází z filozofie RISC. Procesory ARM jsou známé především díky své nízké spotřebě energie a efektivnímu výkonu. Díky tomu tyto procesory mají uplatnění pro embedded systémy nebo mobilní telefony. Firma ARM Holdings původně tyto procesory i vyráběla, ale v současné době procesory pouze navrhuje a následně licencuje výrobcům hardware. [26]

1.3.2 RISC-V

RISC-V je otevřená instrukční sada, vyvíjená na Kalifornské univerzitě v Berkley. Standard instrukční sady RISC-V nestanovuje žádnou konkrétní mikroarchitekturu, je také vysoce modulární a nezávislý na konkrétním výrobcí čipu. Základní instrukční sada je velmi minimalistická a obsahuje pouze 49 instrukcí pro 32bitový procesor. Vývojáři procesorů založených na architektuře RISC-V si mohou libovolně zvolit komerční nebo otevřenou licenci. [27]

Mikropočítače RP2040 a RP2350 jsou oba založeny na architektuře Arm. RP2350 obsahuje navíc architekturu RISC-V. Obě architektury nelze použít současně a je potřeba si mezi nimi vybrat při založení nového projektu v jazyce C pomocí oficiálního rozšíření pro Visual Studio Code.

1.4 Možnosti programování

Pico podporuje programování v jazycích C/C++ a MicroPython. Tato práce se bude zabývat především programování v jazyce C, protože je to mnohem používanější jazyk pro programování mikropočítačů.

Aby se dalo Pico naprogramovat, je potřeba přejít do módu **USB Mass Storage**. Toho se docílí podržením tlačítka **BOOTSEL** během připojování USB kabelu k počítači. Další možností je při stisknutém tlačítku **BOOTSEL** přivést GND na pin RUN. Tím se Pico zobrazí v Průzkumníkovi souborů, podobně jako třeba USB flash disk. Nejjednodušší možností programování je nahrání již hotového **UF2** souboru pouhým přetažením přímo do Pica podobně jako na flash disk. Po nahrání souboru se Pico samo zresetuje. Soubory UF2 nejsou kompatibilní mezi mikropočítači RP2040 a RP2350.

1.4.1 C/C++

Pro jazyky C/C++ existuje více možností programování. V této práci byla zvolena varianta programování ve **Visual Studio Code**, jak doporučuje oficiální dokumentace. Pro VS Code lze doinstalovat **oficiální rozšíření Raspberry Pi Pico** pro práci s SDK, vytváření projektů, usnadnění kompilace, nahrávání programů a možnost ladění. Výhodou VS Code je multiplatformnost, doplňování kódu IntelliSense, podpora rozšíření, vestavěný terminál nebo třeba integrovaná podpora pro Git. Oficiální dokumentace popisuje také nastavení prostředí pro další IDE jako Clion nebo Eclipse. [15]

1.4.2 MicroPython

MicroPython je plnohodnotná implementace jazyka **Python 3** pro mikropočítače jako je třeba Pico. Přímě na hardwaru mikropočítače běží interpret MicroPythonu. Výhodou je vestavěný souborový systém nebo např. interaktivní prompt **REPL** (Read-Evaluate-Print Loop). *Read* znamená, že MicroPython čeká, až uživatel zadá nějaký text a potvrdí klávesou Enter. *Evaluate* znamená, že cokoliv uživatel zadal, tak je interpretováno jako kód v Pythonu a okamžitě jej spustí. Tímto způsobem je možno ovládat třeba LED diodu. *Print* značí, že vypíše např. výsledek výpočtu do konzole. Poslední slovo *Loop* znamená, že se celý tento

proces opakuje. Komunikace s REPL může probíhat buď přes USB nebo přes sériové rozhraní UART na GPIO pinech Pica. [21]

Oproti jazyku C má MicroPython výrazně jednodušší syntaxi a umožňuje snadno ovládat periferie jako displeje nebo NeoPixel LED diody. Vzhledem k tomu, že interpret MicroPythonu běží přímo na mikropočítači, každý příkaz se zpracovává až při spuštění, zatímco v jazyce C je kód předem zkompilován do strojového kódu. Program v MicroPython proto může být pomalejší než program v C. U jednodušších úloh to nevadí, ale při složitějším programu se mohou objevit problémy s výkonem. MicroPython může spouštět skripty z .py souborů. Pokud je třeba, aby se skript spouštěl automaticky po resetu, je třeba jej pojmenovat *main.py*.

Pro instalaci MicroPythonu stačí z oficiální dokumentace stáhnout patřičný UF2 soubor podle verze Pica a nahrát jej pouhým přetažením do mikropočítače. Dle oficiální dokumentace lze programovat v MicroPythonu buď v **IDE Thonny** nebo ve **VS Code** pomocí **neoficiálního rozšíření MicroPico**, které umožňuje automatické doplňování kódu, integraci REPL a podporu pro spouštění a nahrávání souborů na Pico. [21]

1.4.3 PIO

PIO (Programmable I/O) jsou dva speciální identické bloky v mikropočítači RP2040, každý z nich obsahuje 4 programovatelné automaty (state machines), které mohou nezávisle vykonávat sekvenci programů pro práci s GPIO a pro přenos dat. PIO jsou vlastně takové velmi jednoduché samostatně programovatelné procesory s vlastní instrukční sadou a mohou běžet nezávisle na hlavním jádru. Pro programování se používá PIO assembler, který je součástí SDK. PIO assembler má celkem 9 instrukcí: JMP, WAIT, IN, OUT, PUSH, PULL, MOV, IRQ a SET. [22]

PIO se používá, když je potřeba přesné načasování např. pro ovládání NeoPixel LED diod nebo čtení dat z přesných senzorů. Využití naleznou i pro komunikaci s nestandardními protokoly nebo pokud je třeba vyhodnotit rychlou reakci na vstupní signály bez zatížení hlavního procesoru.

1.5 Picopad

Picopad je česká open source herní konzole. Picopad lze zakoupit jako stavebnici nebo již hotovou osazenou variantu. Picopad je založen na Raspberry Pi Pico. Konzole je vybavena IPS displejem s rozlišením 240×320 px a je napájena Li-ion baterií. Picopad je vybaven 8 tlačítky, vypínačem, resetovacím tlačítkem a tlačítkem BOOTSEL. Dále je přítomna uživatelsky ovládaná LED dioda a reproduktor. Nevyužité piny jsou vyvedeny bokem a lze k nim připojit externí modul či prototypovací desku pro rozšíření možností Picopadu. Přes MicroSD slot lze spouštět programy přímo z SD karty, tedy stačí uložit program na SD kartu ve standardním formátu UF2 a následně jej spustit na konzoli. Picopad podporuje programování v jazyce C pomocí vlastního C SDK s vlastním bootloaderem a dále podporuje CircuitPython a MicroPython. [13] [34]



Obrázek 3 Open source herní konzole Picopad
(zdroj: [3])

1.6 Porovnání Raspberry Pi Pico s Arduino UNO

Arduino UNO je velmi populární vývojová deska, která je oblíbená nejen pro výuku, ale i mezi hobby bastlíři. Arduino jako značka bylo taktéž navrženo za účelem podpory výuky podobně jako Raspberry Pi. Pico má v porovnání s UNO rychlejší procesor, větší paměť flash a datovou paměť. Také má o 6 více digitálních vstupů a výstupů. Výhoda UNO je vestavěná paměť EEPROM a šestikanálový AD převodník oproti tříkanálovému, který používá Pico. Další výhodou Arduina je jeho vlastní IDE. [23]

Tabulka 1 Porovnání Raspberry Pi Pico a Arduino UNO

Funkce	Raspberry Pi Pico	Arduino UNO
Mikrokontrolér	RP2040	Atmega328P
Jádro	Dual core, 32-bit, Cortex-M0+	Single-core 8-bit
Takt procesoru	48 MHz až 133 MHz	16 MHz
RAM	264 kB	2 kB
Paměť Flash	2 MB	32 kB
EEPROM	Žádná	1 kB
Napájení	5 V přes USB port	5 V přes USB port
Alternativní napájení	2-5 V přes pin VSYS	7-12 V
Provozní napětí	3,3 V	5 V
Počet GPIO	26	20
Počet AD kanálů	3	6
Počet UART rozhraní	2	1
Počet I2C rozhraní	2	1
Počet SPI rozhraní	2	1
Počet PWM kanálů	16	6
Programovací jazyky	MicroPython, C, C++	C, C++
Počet vestavěných LED	1	1
Cena (RPishop.cz, rok 2025)	109 Kč	709 Kč

2 POUŽITÍ RASPBERRY PI PICO VE VÝUCE

Při výběru vhodného mikrokontroleru pro výuku programování mikropočítačů je třeba zohlednit řadu faktorů, jako je cena zařízení, jednoduchost použití, dostupnost dokumentace a výukových materiálů nebo podpora pro práci s externími periferiemi.

2.1 Proč Raspberry Pi Pico

Raspberry Pi Pico je díky své velmi nízké ceně dostupné nejen pro školy, ale i pro studenty, kteří se mohou věnovat programování mikropočítačů i doma. Verze Pico W nabízí možnost Wi-Fi a Bluetooth, což přidává další možnosti využití třeba pro IoT (Internet of Things) a může také zvyšovat zájem studentů o programování mikropočítačů. Zejména jazyk MicroPython zpřístupňuje Pico všem začínajícím programátorům a hobby uživatelům. Podle informací výrobce by Pico mělo být vyráběno nejméně do ledna roku 2028 a varianta Pico 2 dokonce až do roku 2040, což zajistí jeho dostupnost v budoucnu a díky zpětné kompatibilitě lze bez problému přejít na novější Pico 2.

Velkou výhodou je, že Raspberry Pi Foundation poskytuje oficiální dokumentaci a knihy v elektronické i tištěné podobě. Raspberry Pi také poskytuje ukázkové příklady pro Pico v jazyce C ve svém repozitáři na Githubu. [32] Některé z těchto ukázek je možno jednoduše nainstalovat přímo přes oficiální rozšíření pro VS Code. Oficiální repozitář pro jazyk MicroPython od Raspberry Pi obsahuje poněkud méně ukázkových příkladů, ale to je vynahrazeno velkým množstvím dalších materiálů, které jsou většinou zaměřeny právě na programování v MicroPythonu.

2.2 Raspberry Pi Pico SDK

Raspberry Pi Pico SDK (Software Development Kit) je oficiální balíček, který poskytuje knihovny, hlavičkové soubory a nástroje pro sestavení nezbytné pro vývoj mikropočítačů řady Pico v jazycích C, C++ a Arm assembly language. Kromě standartních knihoven je poskytováno API (Application Programming Interface) pro přístup k hardwaru mikropočítače. SDK také poskytuje vysokoúrovňové knihovny pro práci s časovačem, Wi-Fi, Bluetooth, USB a knihovny pro vícejádrové programování. [17]

2.3 Dostupné materiály pro výuku

Pro použití vývojové desky Raspberry Pi Pico ve výuce je klíčová dostupnost kvalitní dokumentace a výukových materiálů. Tato kapitola se zaměřuje na hledání a výběr těchto materiálů. Pro efektivní výuku je nezbytné mít přístup k dobře strukturovaným návodům, příkladům kódu a odborným knihám, které pokrývají jak základní, tak i pokročilé funkce vývojové desky, včetně práce s periferiemi, senzory a dalšími technologiemi.

2.3.1 Oficiální materiály

Raspberry Pi Foundation poskytuje řadu oficiálních materiálů v anglickém jazyce. Tyto materiály jsou volně dostupné v elektronické podobě na oficiální stránce.

Příručka *Getting started with Raspberry Pi Pico-series* [15] vysvětluje nastavení vývojového prostředí VS Code včetně oficiálního rozšíření, instalaci SDK, možnosti ladění a sestavení projektu. Příručka popisuje i integraci do jiných IDE, jako CLion nebo Eclipse nebo i manuální vytvoření projektu.

Oficiální dokumentace *Raspberry Pi Pico-series C/C++ SDK* [17] podrobně popisuje kompletní SDK pro jazyky C, C++ a Arm assembly language. Obsahuje informace o knihovnách, vysokoúrovňových funkcích a příkladech jejich použití.

Dokument *Connecting to the Internet with Raspberry Pi Pico W-series* [18] se zaměřuje na možnosti připojení Pico W k internetu prostřednictvím SDK pro jazyk C i pro MicroPython. Pro oba tyto jazyky také popisuje práci s technologií Bluetooth.

Tištěná kniha *Get Started With MicroPython on Raspberry Pi Pico* [20] je vhodná zejména pro začátečníky a díky mnoha ilustracím cílí i na mladší publikum. V knize je popsána práce s Picem od pájení hřebenové lišty, přes zapojení vybraných periférií až po následné psaní programů. Uživatelé se seznámí, jak pracovat s LED diodami, tlačítky, LCD displejem a několika senzory. Kniha také popisuje práci s AD převodníkem, práci s PWM modulací, ale i pokročilejší programování NeoPixel LED diod za pomoci PIO.

2.3.2 Výukové materiály v anglickém jazyce

Dostupné jsou také neoficiální knihy v elektronické i tištěné podobě v anglickém jazyce. Tyto knihy se zaměřují především na programování Pica v jazyce MicroPython, ale také i programování v jazyce C. Na platformě YouTube existuje také mnoho videí zaměřených na Pico. Tato videa se věnují jak teoretickému základu, tak i praktickým ukázkám, jako třeba programování krok za krokem. Kromě knih a videí lze nalézt spoustu cenných informací na různých diskusních fórech.

2.3.3 Výukové materiály v českém jazyce

Dle veřejně dohledatelných informací se Pico vyučuje i na českých školách a některé z těchto škol poskytují vlastní výukové materiály:

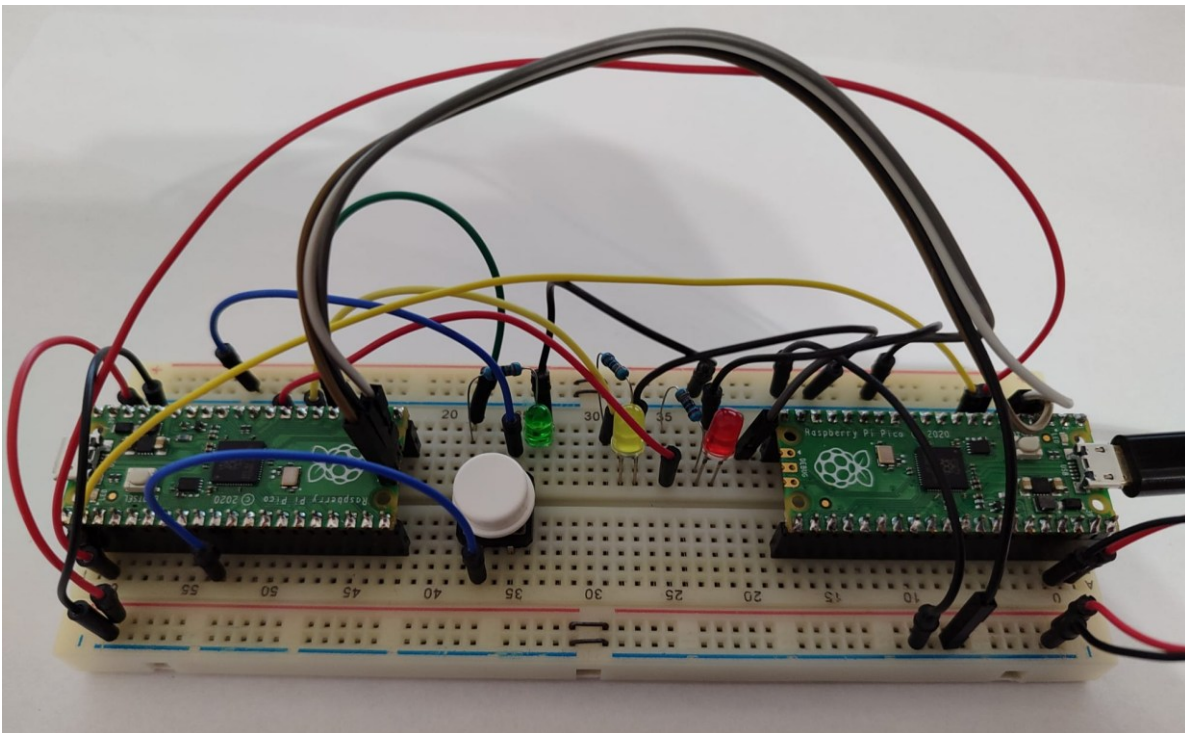
- Střední škola – Podorlické vzdělávací centrum, Dobruška [30]
- Gymnázium a Střední průmyslová škola elektrotechniky a informatiky, Frenštát pod Radhoštěm [31]

Kromě výše zmíněných škol se dle veřejných informací Pico používá při výuce i na Gymnáziu U Libeňského zámku v Praze nebo na Fakultě elektrotechnické ČVUT v Praze.

Několik článků o Raspberry Pi Pico bylo napsáno na webu *root.cz*. Za zmínku stojí seriál o PIO na mikropočítači RP2040. [33] Pico bylo i tématem na konferenci OpenAlt, kde je dostupný záznam přednášky s praktickou ukázkou MQTT v MicroPythonu a integrací Node-RED. [35]

3 NÁVRH HARDWAROVÉ PLATFORMY PRO VÝUKU

Pico lze programovat a používat i bez externích periférií. Dokáže komunikovat přes sériové rozhraní s počítačem, má zabudovanou LED diodu a vnitřní analogový teplotní senzor. Pro připojení externích periférií se nabízí možnost připájení dutinkové či pinové lišty přímo na Pico nebo použití nepájivého pole. Tyto varianty jsou vhodné na jednoduché a krátké prototypování, ovšem nejsou vhodné na častější přepojování obvodu při výuce. Na destičce Pica jsou názvy pinů natisknuty na spodní straně. Na vrchní straně jsou natisknuty pouze čísla pinů 1, 2 a 39, což může způsobovat chybné zapojení při špatném počítání, navíc když čísla pinů nesouhlasí s číslem GPIO.



Obrázek 4 Testování na nepájivém poli
(zdroj: vlastní)

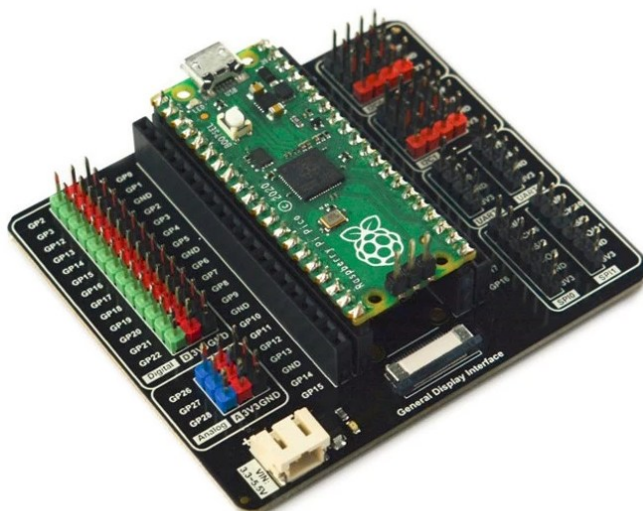
3.1 Dostupná komerční řešení

Tvorba vlastní hardwarové platformy by sice umožňovala přizpůsobení vybraný periferiím, nicméně by ve výsledku nemusela nabídnout více výhod, než mají některé již existující komerční řešení. V potaz byl brán i fakt, že by si uživatelé museli tuto vlastní platformu vyrobit, aby mohli jednoduše používat ukázkové programy. Z hlediska výroby by se toto řešení nemuselo vyplatit jak svou náročností, tak i finančně. Z tohoto důvodu bylo vhodnější vybrat některé existující komerční řešení. Bylo stanoveno, že komerční řešení musí obsahovat již osazené, byť základní periferie jako tlačítka nebo LED diody. Důležitým faktorem pro výběr byla i přítomnost resetovacího tlačítka.

Někteří výrobci nabízí rozšiřující desky pro připojení dalších periferií. Další výrobci zase nabízí modulární rozšiřující destičky tzv. shieldy, které se dají nasadit přímo na mikropočítač a mají nějakou konkrétní funkcionalitu. Shieldy pro Raspberry mohou být někdy označovány jako HAT. Aby bylo přístupné tlačítko BOOTSEL, tak je většina shieldů pro Pico navržena pro nasazení na spodní stranu.

3.1.1 Gravity Expansion Board

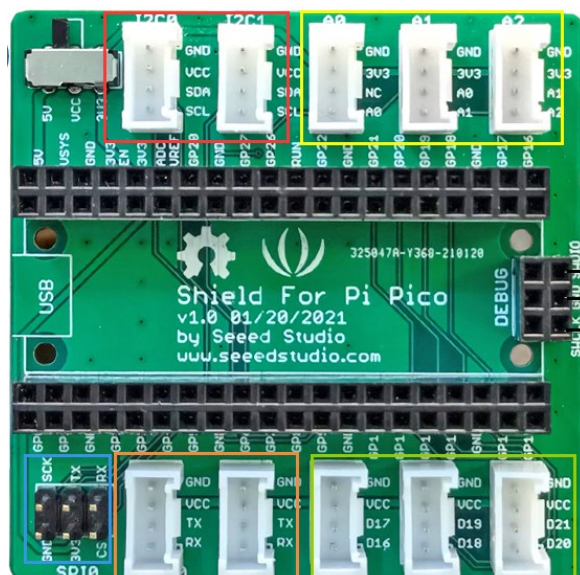
Tato rozšiřující deska má vyvedeny dutinkové lišty, kopírující piny Pica. Dále jsou vyvedeny pinové lišty pro připojení externích periferií včetně zdvojených rozhraní pro I²C, SPI a UART, 3 analogové piny a 13 digitálních pinů. Výrobce udává kompatibilitu s Grove senzory. Cena desky je okolo 180 Kč, avšak neobsahuje žádné základní periferie.



Obrázek 5 Gravity Expansion Board
(zdroj: [4])

3.1.2 Grove Shield for Pi Pico v1.0

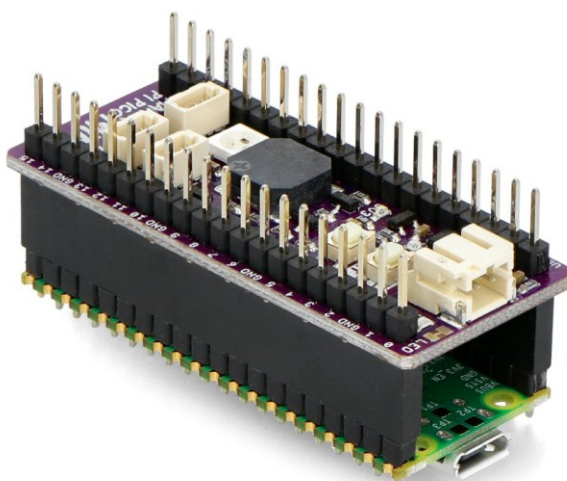
Tento shield má vyvedenou dutinkovou lištu, do které je možno zapojovat vodiče, ale také je zde možnost připojení periférií pomocí čtyřpinových konektorů pro jejich vlastní Grove senzory. Shield má vyvedeny konektory pro sběrnice SPI a oba kanály I²C. Dále jsou vyvedeny 3 analogové porty, 3 digitální porty a 2 UART porty. Pro připojení debug rozhraní musí mít Pico toto rozhraní vyvedeno ze spodní strany. Samotná deska se dá koupit za 120 Kč, ale je třeba ještě dokoupit Grove moduly. Nevýhodou může být, že tyto moduly jsou připojeny jen tak navolno.



Obrázek 6 Rozvržení Grove shieldu
(zdroj: [5])

3.1.3 Maker Pi Pico Mini

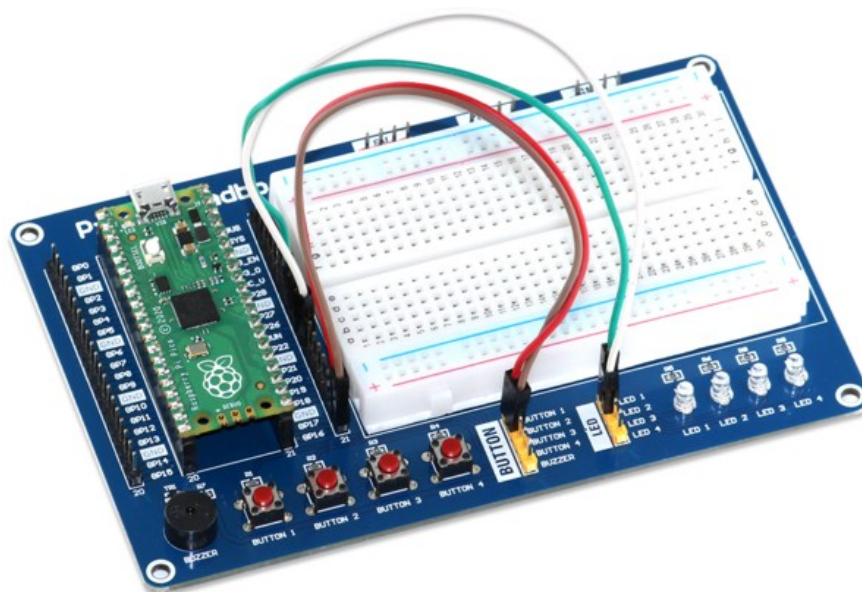
Tento shield od firmy Cytron se dá nasunout na Pico ze spodní strany, díky čemuž je zachován přístup k tlačítku BOOTSEL. Shield obsahuje programovatelné tlačítko, RGB LED diodu, bzučák. Také nabízí možnost připojení periférií pomocí konektorů nebo pinových lišt. Navíc obsahuje konektor pro připojení Li-pol nebo Li-Ion baterií. Tento shield se dá koupit ve variantě s již osazeným Pico W za 350 Kč. Tento shield nenabízí příliš komfortní řešení pro připojení periférií.



Obrázek 7 Maker Pi Pico Mini
(zdroj: [6])

3.1.4 Pico Breadboard KIT

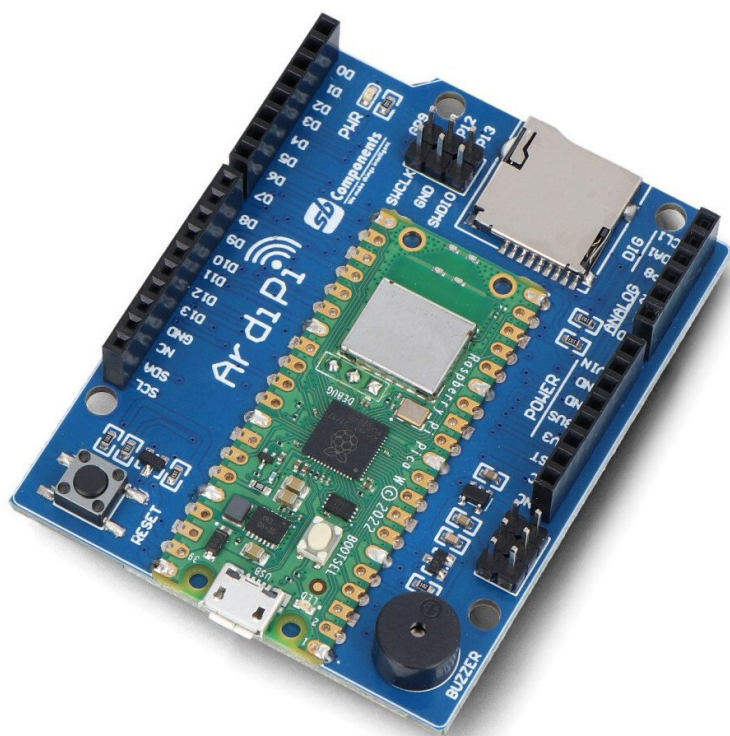
Tato open source rozšiřující deska od výrobce SB Components nabízí 4 tlačítka, 4 LED diody a bzučák. Pro připojení dalších periférií je možno použít nepájivé pole a pinové lišty rozšiřující vývody Pica. Deska nemá integrované resetovací tlačítko. Výrobce pro tuto desku poskytuje ukázkové příklady pro obsluhu LED diod a tlačítek v jazyce MicroPython. Cena této desky je necelých 500 Kč.



Obrázek 8 Pico Breadboard KIT
(zdroj: [7])

3.2 ArdiPi

Další open source produkt firmy SB Components. Tato vývojová deska je alternativou k Arduino Uno. Obsahuje integrované Raspberry Pi Pico W. Pro tuto desku je možno použít Arduino shiely kompatibilní s napětím 3,3 V. Výrobce udává kompatibilitu se shiely pro relé, RFID čtečku, IPS displej a UHF shield. Tyto shiely jsou navrženy spíše pro specifické účely a neobsahují univerzálnější komponenty vhodné pro výuku. Samotná deska obsahuje bzučák, čtečku SD karet a resetovací tlačítko. Výrobce poskytuje ukázkové programy pro tuto desku v jazyce MicroPython. Cena této desky je kolem 540 Kč.



Obrázek 9 Vývojová deska ArdiPi
(zdroj: [6])

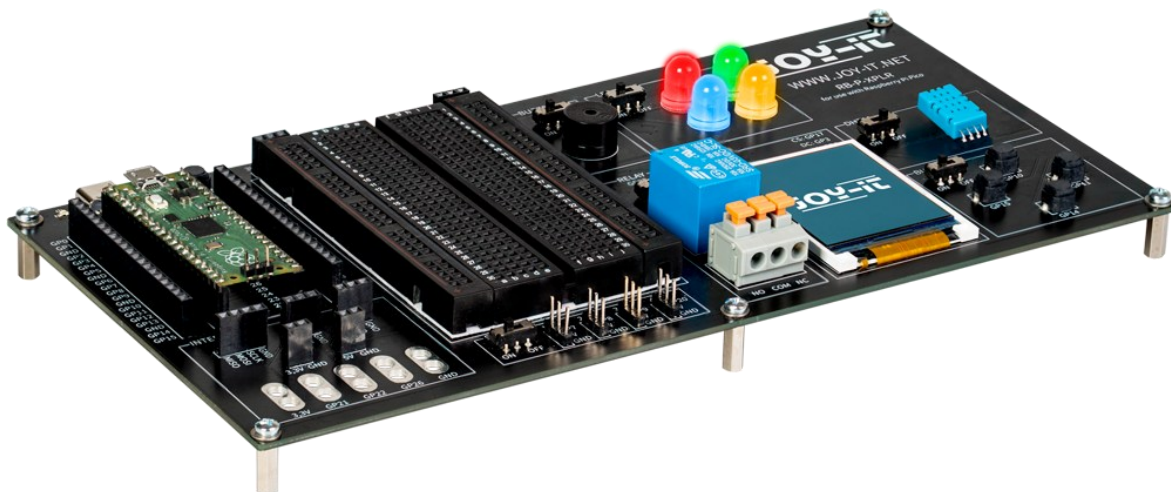
3.3 Explorer Board

Tato rozšiřující deska od výrobce Joy-IT již obsahuje od výrobce několik periférií. Tyto periférie je možno odpojit od obvodu pomocí přepínačů, takže je případně možno použít piny Pica pro jiné účely nebo jen odpojit nepoužívané periférie. Deska obsahuje 4 tlačítka, resetovací tlačítko, senzor teploty a vlhkosti DHT11, TFT displej, 4 NeoPixel LED diody, bzučák a relé. Pro připojení dalších periférií je možno využít rozšiřující dutinkové lišty, rozhraní pro servo motory, SPI, I²C, UART, dále rozhraní pro připojení pomocí krokosvorek

a také nepájivé pole. Výrobce poskytuje ukázkové programy pro všechny periferie na desce v jazyce MicroPython. Cena desky je 560 Kč.

Pico je standardně připojeno pomocí Micro-USB kabelu, kterým se programuje a dokáže napájet většinu periférií. Pro použití NeoPixel LED diod a relé je třeba napájet desku dodatečným kabelem USB-C.

Pro tuto práci byla vybrána právě deska Explorer Board vzhledem k již osazeným komponentám, jejich možnosti rozpojení od obvodu přepínačem, přítomnosti resetovacího tlačítka a možnosti nepájivého pole pro snadné připojení dalších komponent dle volby uživatele.



Obrázek 10 Vývojová deska Explorer Board
(zdroj: [8])

3.4 Komponenty zvolené rozšiřující desky Explorer Board

Pro ukázkové příklady budou použity všechny komponenty integrované na této rozšiřující desce. Pro jednodušší práci byly přidány ještě externí LED diody, protože integrované NeoPixel diody vyžadují použití speciální knihovny a pokročilejší konfiguraci. Pro jednoduché úlohy, kde je potřeba pouze základní ovládání LED diod, by takové řešení bylo zbytečně složité. Kromě již osazených komponent byly v ukázkách použity 3 klasické LED diody a potenciometr.

3.4.1 Tlačítka

Jako základní vstup od uživatele poslouží 4 mikropínače, jejichž rozmístění umožňuje např. detekci směru pohybu v menu programu nebo jako 2 páry tlačítek pro konkrétní úlohy jako např. zesílení/zeslabení.

K tlačítkům na desce jsou již připojeny externí pull up rezistory. Tyto rezistory zajistí, aby nebyl vstupní pin tzv. „ve vzduchu“ a mikropočítač by tím četl nedefinovanou hodnotu na vstupu.

Pro nahrání nového programu na Pico je třeba přejít do režimu USB Mass Storage. Je možno využít integrované resetovací tlačítko na desce, které přivede GND na pin RUN a Pico se zresetuje. Během držení tlačítka BOOTSEL se jen zmáčkne resetovací mikrospínač a Pico přejde do režimu USB Mass Storage. Díky tomu není třeba odpojovat Micro-USB kabel při každém nahrání nového programu.

3.4.2 Bzučák

Bzučák slouží k vytvoření zvukového signálu, ale na rozdíl od reproduktoru je limitován menším rozsahem frekvencí, takže neposkytuje kvalitní zvuk pro hudbu nebo řeč. Využití nalezne v projektech, kde bude třeba zvukově upozornit uživatele – třeba budík. Na desce Explorer Board je vestavěný pasivní bzučák.

3.4.3 RGB LED diody

Deska Explorer Board obsahuje 4 adresovatelné WS2812 (NeoPixel) RGB LED diody. NeoPixel je registrovaná značka firmy Adafruit. Tyto NeoPixel diody se používají pro různé světelné efekty či dekorativní osvětlení.

NeoPixel LED diody jsou připojeny pomocí jednoho datového vodiče. Každou z diod je možno ovládat samostatně. Barva se nastavuje pro každou z RGB hodnot v rozsahu 0 až 255, ovšem diody na vývojové desce jsou příliš silné a není doporučeno používat hodnotu větší než 10.

3.4.4 Relé

Relé je elektromechanická součástka, která slouží ke spínání zátěže. Relé na vývojové desce umožňuje spínat až 30 V stejnosměrné zátěže. Bezšroubové svorky umožňují zapojit zátěž v zapojení jak Normally Open (při sepnutém relé je zátěž sepnuta), tak i Normally Closed (při sepnutém relé je zátěž rozpojena).

3.4.5 TFT displej

TFT LCD (thin-film-transistor liquid-crystal display) je typ LCD displeje, který používá tenkovrstvé tranzistory pro zlepšení kvality a plynulosti obrazu. Pod každým pixelem je jeden tranzistor, ovládající jas tohoto pixelu.

Displej na desce Explorer Board nabízí úhlopříčku 1,8“ s rozlišením 128×160 px a možností až 65 000 barev. Komunikace s displejem probíhá přes SPI sběrnici.

3.4.6 DHT 11

Tento senzor slouží k měření teploty v rozsahu 0-50 °C s přesností ± 2 °C a k měření relativní vlhkosti v rozsahu 20-80 % s přesností ± 5 %. Přenos dat probíhá po jednom vodiči.

3.4.7 LED diody

Deska Explorer Board má integrované 4 adresovatelné RGB diody NeoPixel. Tyto diody se však neovládají jako běžné LED diody. Tyto diody vyžadují specifický řídicí signál pro nastavení barvy a jasu. Protože práce s NeoPixel diodami je složitější, byly na nepájivé pole přidány 3 LED diody – červená, žlutá, zelená, jejichž barvy umožní sestavit úlohu se semaforem. Tyto diody jsou připojeny přes rezistor 330 Ω .

Pico má vestavěnou LED diodu na pinu GP25. Dioda se rozsvítí přivedením logické jedničky na výstup tohoto pinu. Ve variantě Pico W se tato dioda neovládá nastavováním výstupu na pin GP25, ale musí se nastavit pomocí funkce přes Wi-Fi modul.

3.4.8 Analogové vstupy

Deska Explorer Board neobsahuje žádné komponenty pro práci analogovými hodnotami, a proto byl přidán potenciometr pro použití v ukázkových příkladech. Potenciometr se dá nahradit i fotorezistorem dle potřeby. Pico má také zabudovaný analogový senzor teploty.

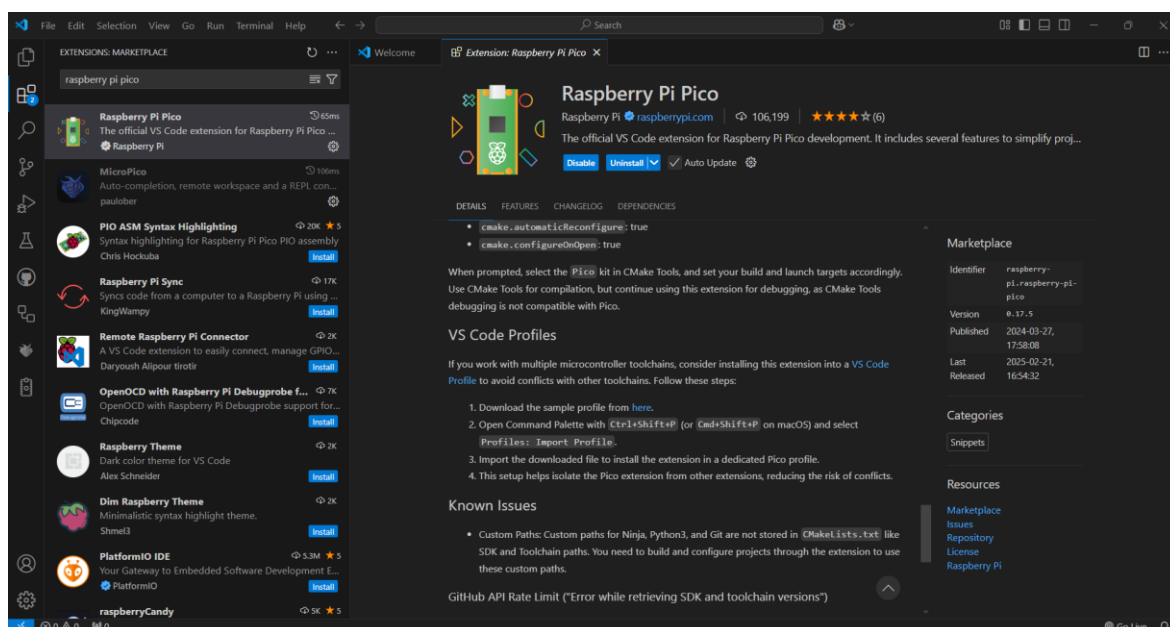
II PRAKTICKÁ ČÁST

4 PRÁCE S PROSTŘEDÍM VE VISUAL STUDIO CODE

Visual Studio Code je vývojové prostředí, které je doporučováno oficiální dokumentací pro programování v jazyce **C** a **C++**. Do tohoto prostředí je možno doinstalovat **oficiální rozšíření od Raspberry Pi** pro značené zjednodušení programování a nahrávání programu pro Pico.

4.1 Nastavení prostředí

Nejdříve je třeba v nabídce rozšíření (extensions) vyhledat **Raspberry Pi Pico**. Po instalaci se na levé straně objeví několik nových ikonek, z nichž nejdůležitější je ikonka reprezentující Pico s názvem **Raspberry Pi Pico Project**.



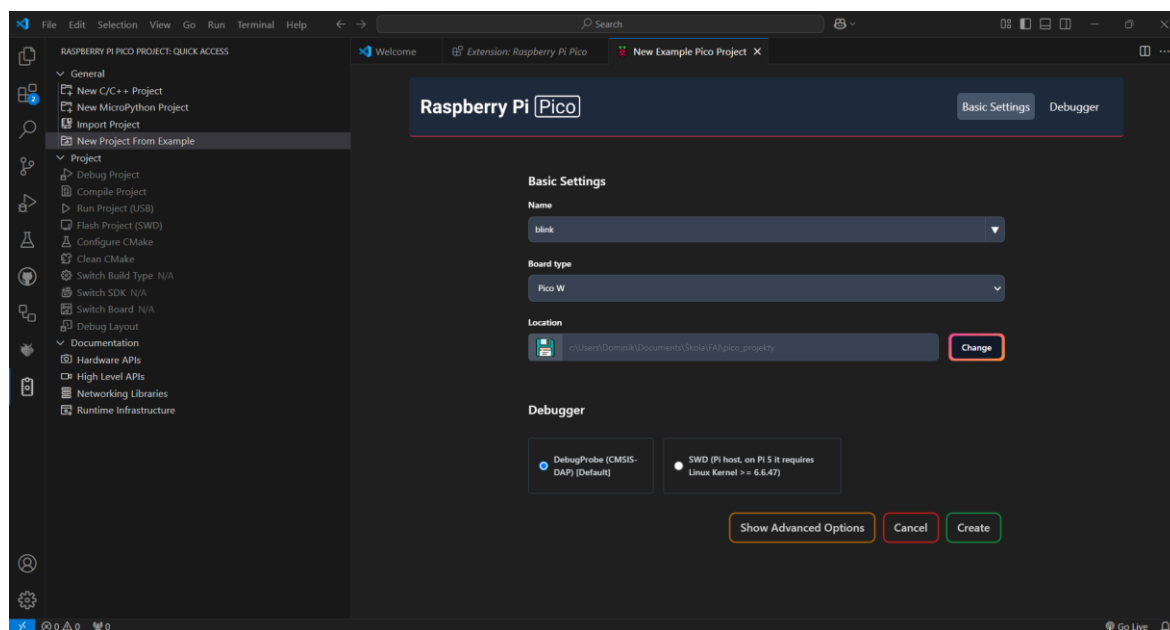
Obrázek 11 Rozšíření Raspberry Pi Pico pro VS Code
(zdroj: vlastní)

V dalším kroku je potřeba stáhnout Pico SDK pro C/C++. Toto SDK se stáhne automaticky po založení nového projektu. V rozšíření **Raspberry Pi Pico Project** se nový projekt založí výběrem možnosti **New Project from Example**.

Z rozbalovací nabídky **Board type** je třeba vybrat správnou **verzi Pica**. Tuto verzi desky je možno měnit i po založení projektu. Verze Pica s různými procesory nejsou vzájemně kompatibilní, Pico 2 má navíc možnost výběru procesorových jader mezi ARM a RISC-V. Z rozbalovací nabídky **Name** se vybere vzorový projekt **blink**. Dále je třeba zvolit **umístění projektu** tak, aby cesta **neobsahovala diakritiku ani speciální znaky** (podtržítka

může být), jinak nebude správně fungovat konfigurace CMake. V sekci **Debugger** ponechat výchozí nastavení **DebugProbe**.

Vytvoření projektu se dokončí kliknutím na tlačítko **Create**. Při tvorbě prvního projektu se začne stahovat SDK. Tato operace může trvat několik minut.



Obrázek 12 Založení nového C/C++ projektu
(zdroj: vlastní)

Vytvořil se nový projekt, který obsahuje soubor **blink.c**, jež obsahuje ukázkový zdrojový kód. Postup nahrání programu se liší podle toho, jestli máme připojeno Pico přímo nebo přes Debug Probe.

4.2 Programování samostatného Pica

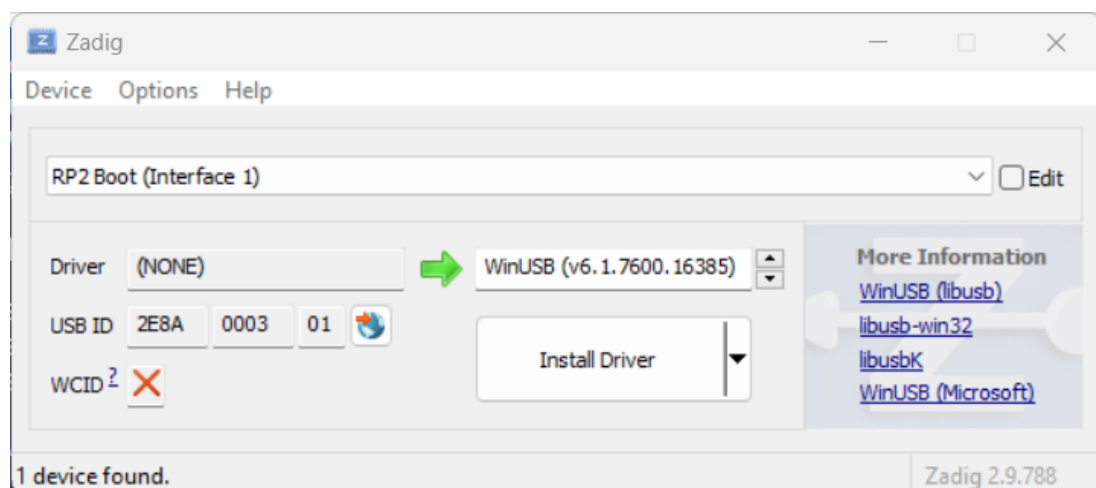
Aby se dal program nahrát, musí být Pico v módu **USB Mass Storage**, toho lze dosáhnout držením tlačítka **BOOTSEL** během připojování k počítači Micro-USB kabelem. Pokud již bylo Pico připojeno, je možno jej resetovat přivedením GND na pin RUN. Vývojová deska Explorer Board k tomuto účelu již obsahuje resetovací tlačítko.

Pro reset je třeba držet stisknuté tlačítko **BOOTSEL** a během toho kliknout na resetovací tlačítko. Jakmile počítač detekuje Pico, je možno nahrát program stisknutím tlačítka **Run** na dolní liště. Před každým nahráním nového programu je třeba znovu přejít do režimu **USB Mass Storage**.

Pokud je Pico v režimu USB Mass Storage a přesto se při nahrání programu zobrazí hláška, jako je na obrázku, je třeba nainstalovat ovladač přes program **Zadig**, jak radí oficiální dokumentace. [15]

```
No accessible RP-series devices in BOOTSEL mode were found.  
  
but:  
  
RP2040 device at bus 1, address 9 appears to be in BOOTSEL mode, but picotool was unable to connect. You may  
need to install a driver via Zadig. See  
"Getting started with Raspberry Pi Pico" for more information
```

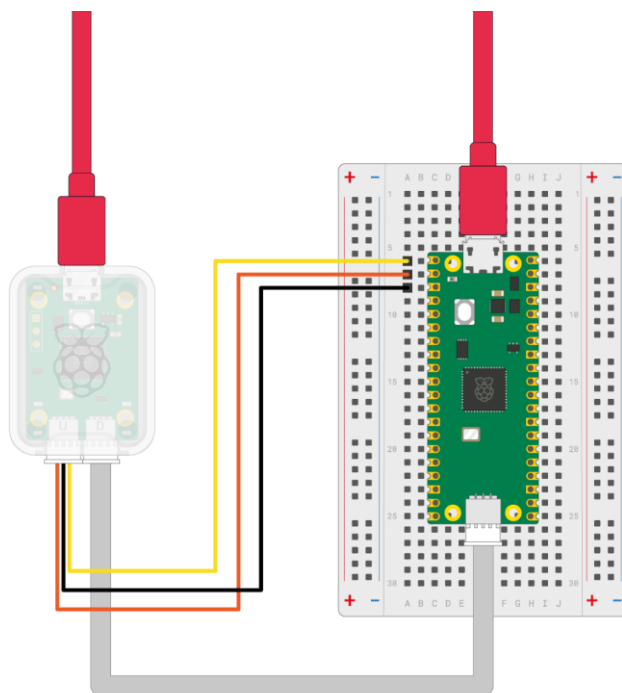
Obrázek 13 Hláška vyžadující instalaci driveru přes program Zadig
(zdroj: vlastní)



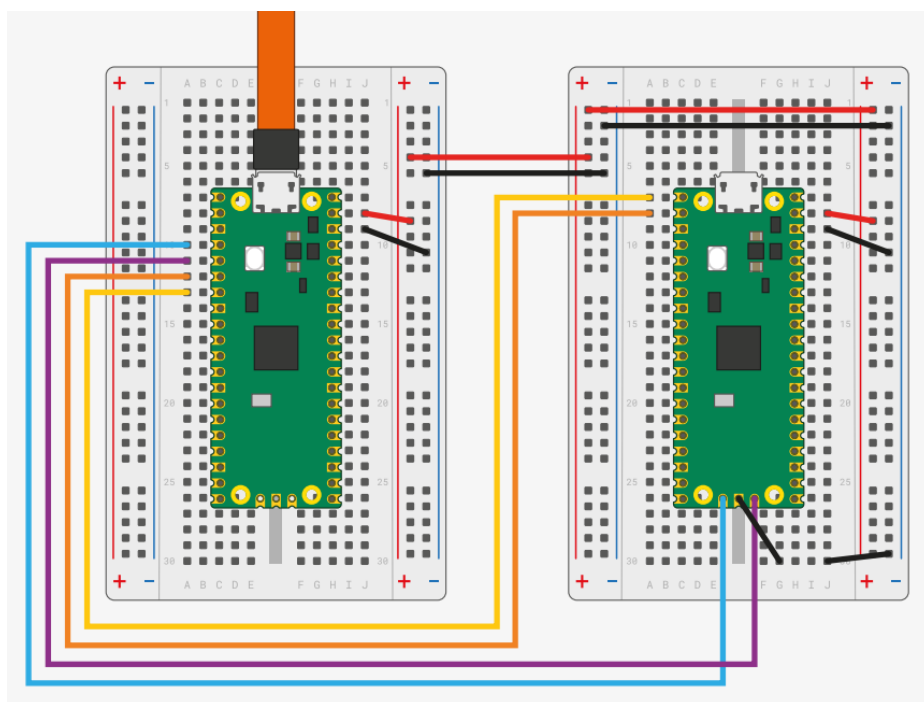
Obrázek 14 Instalace driveru přes program Zadig
(zdroj: vlastní)

4.3 Programování přes Debug Probe

Pokud je potřeba program ladit detailněji, krokovat nebo zobrazovat aktuální stav proměnných, je potřeba mít Pico připojeno přes Debug Probe, a to buď přes oficiální **Raspberry Pi Debug Probe** nebo za pomoci dalšího Pica s nainstalovaným firmwarem **debugprobe**.



Obrázek 15 Zapojení Raspberry Pi Debug Probe
(zdroj: [12])



Obrázek 16 Zapojení druhého Pica ve funkci Debug Probe
(zdroj: [12])

V tomto režimu nelze nahrát program tlačítkem Run v dolní liště, ale je třeba jít do nabídky **Raspberry Pi Pico Project**, kde je možno nejdříve zkompileovat projekt tlačítkem **Compile Project** nebo nahrát program přímo přes tlačítko **Flash Project (SWD)**.

4.4 Ladění projektu

Pokud je třeba program krokovat, je možno nastavit breakpoint kliknutím vlevo od čísel řádků v místě, kde je třeba program pozastavit. V tomto místě se objeví červená tečka. V nabídce **Raspberry Pi Pico Project** se vybere možnost **Debug Project** a zvolí se možnost **Pico Debug (Cortex-Debug)**. Po několika sekundách se zobrazí debug layout a program vyčkává na spuštění. V horní liště se objeví panel s několika tlačítky pro krokování, spouštění/pozastavení programu nebo reset Pica. Debugování je třeba ukončit tlačítkem **Stop**, aby bylo možno nahrát další program.

4.5 Založení vlastního projektu

Pro založení nového projektu se vybere v nabídce rozšíření **Raspberry Pi Pico Project** a následná volba **New C/C++ Project**. Je třeba zvolit správnou variantu Pica, vyplnit název a vybrat umístění projektu, které nesmí obsahovat žádné speciální znaky kromě podtržítka. Pro výpis textu na sériovou linku, je možno zaškrtnout možnost **Console over USB**. Pokud je Pico připojeno přes Debug Probe, vybere se možnost **Console over UART**. Ostatní možnosti stačí nechat ve výchozím stavu a projekt se vytvoří tlačítkem **Create**.

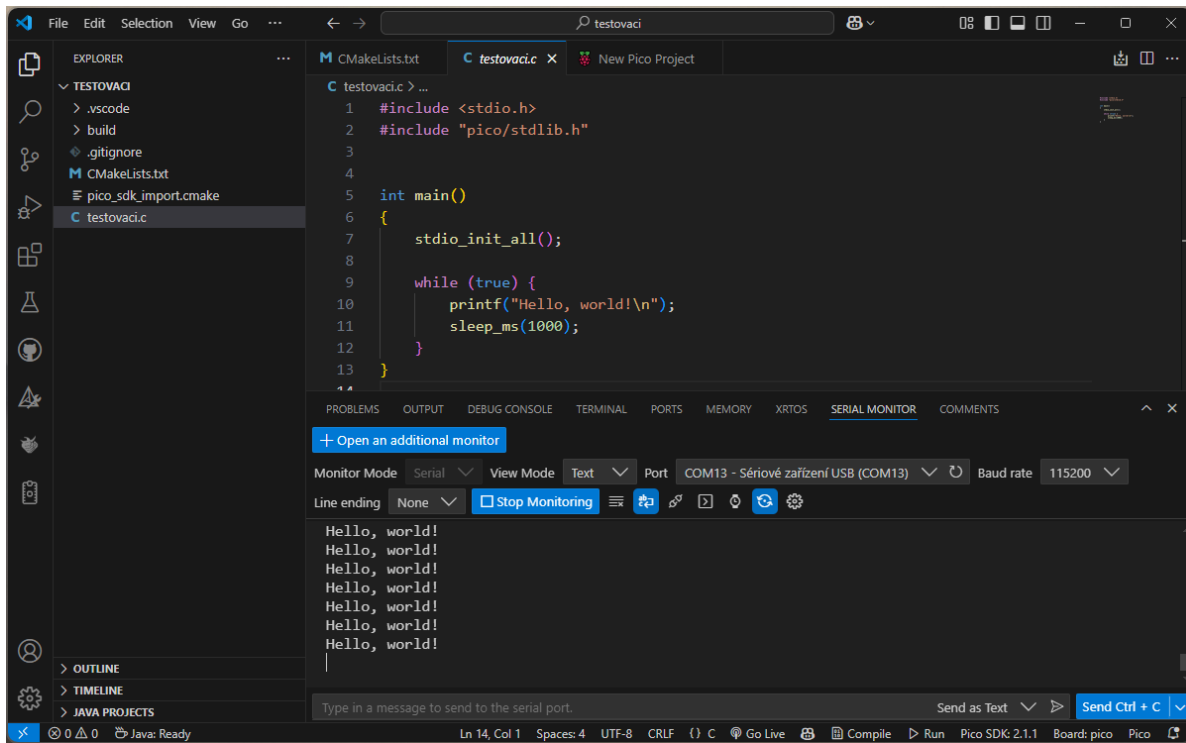
Nový projekt se otevře jako nové okno VS Code, starý projekt se může zavřít, pokud není potřeba. V souboru s příponou .c, který má shodný název jako název projektu, se již nachází ukázkový program **Hello World**.

Program je možno nahrát tlačítkem **Run** ve spodní liště. V této liště jsou také tlačítka pro kompilaci programu nebo pro změnu verze Pica. Tyto funkce jsou také dostupné z nabídky rozšíření.

Pro otestování, zda vše funguje se vybere záložka **SERIAL MONITOR** (pokud je panel skrytý, je možno jej zobrazit stisknutím **Ctrl+J**). Zde se musí vybrat správný port (na obrázku COM13) a kliknutím na **Start Monitoring** se začne vypisovat text do konzole. Pokud se zobrazuje pouze port COM1, je možné, že při založení souboru nebyla zaškrtnuta volba pro nastavení sériové linky. V tomto případě se musí překontrolovat soubor **CMakeLists.txt**, kde se upraví následující řádky podle toho, jestli se používá komunikace přes UART nebo přes USB:

```
# Modify the below lines to enable/disable output over UART/USB  
pico_enable_stdio_uart(testovaci 0)  
pico_enable_stdio_usb(testovaci 1)
```

Pokud je vše v pořádku, v konzoli se zobrazí výpis „Hello, world!“



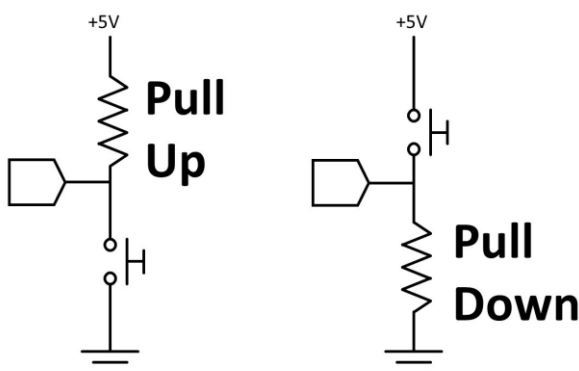
Obrázek 17 Výpis textu na sériové lince
(zdroj: vlastní)

5 ZAPOJENÍ

Tato kapitola popisuje zapojení jednotlivých komponent, které byly použity pro ukázkové příklady. Pro komponenty, které nejsou součástí desky Explorer Board je možno využít vestavěné nepájivé pole a vyvedené pinové lišty. Pico používá napětí 3,3 V, proto je třeba dávat pozor při připojení komponent, které pracují s 5 V, aby nedošlo k poškození Pica.

5.1 Tlačítka

K tlačítkům na desce Explorer Board jsou již připojeny externí pull up rezistory. Tyto pull up rezistory zajistí, aby nebyl vstupní pin „ve vzduchu“ a tím by mikropočítač četl nedefinovanou hodnotu na vstupu. Při použití pull up rezistoru je na vstupu pinu logická jednička, když je tlačítko rozepnuto. Když je tlačítko sepnuto, na vstupu pinu je logická nula. U pull down rezistoru je na vstupu při rozepnutém tlačítku logická nula a při sepnutém tlačítku je na vstupu logická jednička. [19]



Obrázek 18 Zapojení pull up a pull down rezistorů
(zdroj: [9])

5.2 LED diody

V některých ukázkových příkladech byly použity 3 externí LED diody na nepájivém poli. Diody se rozsvítí přivedením kladného napětí. Pro omezení proudu je zapojen do série rezistor o hodnotě 330 Ω . Dioda musí být zapojena v propustném směru. U nové nezkrácené diody je delší vývod anoda (+), kratší vývod je katoda (-). Polaritu lze zjistit také podle velikosti elektrod (plošek uvnitř diody), kdy anoda má menší plošku než katoda. Orientace diody se dá ověřit i pomocí multimetru.

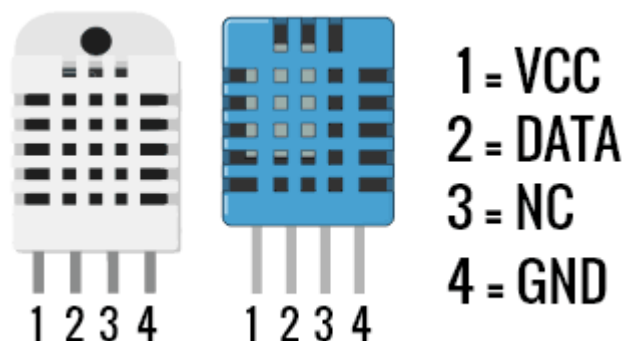
5.3 Potenciometr

Jako zástupce komponent pro čtení analogových hodnot byl zvolen potenciometr o hodnotě 10 k Ω . Deska Explorer Board neobsahuje potenciometr, a proto je třeba jej připojit pomocí nepájivého pole. Prostřední vývod potenciometru (jezdec) je připojen na pin GP26, ze zbylých vývodů se jeden připojí na napětí 3,3 V a druhý na GND. Pro změnu směru otáčení potenciometru lze tyto dva vývody prohodit.

Potenciometr funguje jako napěťový dělič. Otáčením potenciometru se mění poměr odporů děliče a na jezdcí se mění napětí. Místo potenciometru lze použít fotorezistor a pomocí rezistoru 10 k Ω vytvořit dělič napětí obdobným způsobem, jako funguje potenciometr.

5.4 DHT11

Tento senzor teploty a vlhkosti komunikuje pomocí svého vlastního protokolu, který používá sběrnici s jedním vodičem. Vývod DATA se připojí na pin Pica (v ukázkách GP0). Zapojení DHT11 je stejné jako zapojení DHT22, který má větší přesnost.



Obrázek 19 Rozložení pinů pro DHT22 a DHT11
(zdroj: [10])

5.5 TFT displej

Tento displej komunikuje pomocí SPI sběrnice. Kromě napájení je třeba připojit větší počet signálních vodičů. Pin SCK (Serial Clock), někdy označován jako SCL, slouží k přenosu hodinového signálu z mikrokontroléru. Pomocí datového vodiče MOSI (Master Out Slave In) se posílají data do displeje. Oba signály SCK a MOSI jsou sdílené pro všechna zařízení na stejné SPI sběrnici. Pro komunikaci s displejem, je třeba vybrat konkrétní zařízení přes pin CS (Chip Select), který se nastaví do log. 0, když je třeba s displejem aktivně

komunikovat. Pin DC (Data/Command) určuje, zda se posílá příkaz nebo data pro zobrazení. K resetu displeje slouží pin RST. Poslední pin BL (Backlight) slouží k napájení podsvícení displeje.

Tabulka 2 Zapojení TFT displeje na desce Explorer Board

Pin displeje	SCK	MISO	MOSI	CS	DC	RES	BL
Pin na Pico	GP18	GP16	GP19	GP17	GP3	GP6	GP2

5.6 NeoPixel LED

Deska Explorer Board má již osazené 4 RGB NeoPixel LED diody. Tyto diody je třeba napájet 5 V a Desku Explorer Board je třeba napájet dodatečným USB-C kabelem. NeoPixel diody obsahují 3 LED diody (červenou, zelenou a modrou) a malý mikrokontrolér. Diody NeoPixel je možno zapojovat do série a řízení probíhá přes jeden datový vodič (na Explorer Board GP1). Pro řízení diod je třeba velmi přesné časování signálu. Tento signál se potom šíří z první diody do dalších diod v řetězci.

5.7 Relé

Napájení relé vyžaduje 5 V přes dodatečný USB-C kabel. Relé se sepne přivedením kladného napětí na řídicí pin IN (GP27 na Explorer Board). Na výstupní bezšroubové svorky je možno připojit spínanou zátěž. Přívodní vodič se zapojí na společnou svorku COM. Pokud má být zátěž sepnutá, když je relé sepnuto, zátěž se zapojí na svorku NO (Normally Open). Pokud se má zátěž rozpojit, když je relé sepnuto, připojí se na svorku NC (Normally Closed).

5.8 Bzučák

Deska Explorer Board má vestavěný pasivní bzučák. Záporný pól bzučáku se jednoduše připojí na GND a na výstup z mikrokontroleru se připojí kladný pól (na Explorer Board GP27).

6 PROGRAMOVÁNÍ PERIFERIÍ

Tato kapitola popisuje princip jednotlivých periférií vývojové desky Raspberry Pi Pico, které byly použity pro ukázkové programy. Samotná implementace v kódu je součástí ukázkových příkladů.

6.1 Základní struktura programu pro mikropočítače

Na začátek kódu se pomocí **#include** připojují knihovny pro použití funkcí. Pomocí direktivy **#define** je možno vytvořit **makro**, které definuje **symbolickou konstantu**, jako třeba číslo pinu. Tato konstanta následně nahrazuje všechny její výskyty za konkrétní hodnotu, ještě před samotným překladem programu a díky tomu makro nezabírá žádnou paměť během běhu programu. Výhodou je, že v případě změny pinu potom stačí změnit jen hodnotu makra na jednom místě.

Následuje funkce **main**, která musí být v programu vždy přítomna, protože je to první funkce, která se zavolá po spuštění programu. Obvykle vrací celé číslo (**integer**). Při úspěšném dokončení vrací hodnotu **0**, jiné hodnoty mohou sloužit k indikaci **chyby**. Parametr **void** uvnitř závorek znamená, že funkce **main** nepřijímá žádné vstupní parametry.

Uvnitř funkce **main** je obvykle potřeba provést **inicializace** pro **GPIO piny**, **komunikační rozhraní**, **senzory** nebo **další periferní zařízení**, aby byly připraveny pro použití v programu. Následuje **nekonečná smyčka** programu, kterou obsahuje téměř každý program pro mikropočítače. Tato smyčka zajišťuje trvalý běh programu, dokud je zařízení zapnuto. Obsahuje kód, který se má provádět opakovaně.

Za nekonečnou smyčkou vrací funkce **main** hodnotu **0**, což je zde jen pro úplnost kódu a ve skutečnosti se tento kód nikdy neprovede, protože program bude stále běžet v nekonečné smyčce.

Pokud je potřeba vytvořit si **uživatelské funkce**, je potřeba je definovat **nad** funkcí **main**, protože překladač jazyka C „čte“ kód shora dolů, proto tyto funkce „nevidí“ a může zahlásit chybu. Pro větší přehlednost kódu je možno mít tyto funkce definované i **pod** funkcí **main**, ale tyto funkce musí mít deklarován svůj **prototyp funkce** ještě **před** funkcí **main**.

```
// Knihovny pro další funkce
#include <stdio.h> // Pro základní funkce Pico SDK
#include "pico/stdlib.h" // Pro funkce jako třeba printf()

#define LED_PIN 20 // Definice makra LED_PIN

void init_led(void); // Prototyp uživatelské funkce (deklarace)

int main(void) {
    stdio_init_all(); // Inicializace USB/UART pro výpis do konzole
    init_led(); // Inicializace LED pomocí uživatelské funkce

    while(1){
        // Hlavní smyčka programu
    }
    return 0; // Tento kód se nikdy neprovede
}

// Definice uživatelských funkcí
void init_led(void) {
    // Tělo funkce init_led()
}
```

6.2 Konfigurace CMake

Kromě hlavního C souboru obsahuje každý projekt ještě soubor *CMakeLists.txt*, který říká SDK, jak převést C soubor na binární aplikaci pro mikropočítač. Po úpravě tohoto souboru je potřeba jej nakonfigurovat. V rozšíření pro Pico lze toto udělat jednoduše kliknutím na tlačítko **Configure CMake**. Pokud proběhla konfigurace v pořádku, zobrazí se hláška „*CMake has configured your build.*“, v opačném případě je potřeba překontrolovat soubor *CMakeLists.txt* soubor a znovu nakonfigurovat. Minimální podoba souboru *CMakeLists.txt* pro příklad *blink* by vypadala následovně:

```
cmake_minimum_required(VERSION 3.13)

set(CMAKE_C_STANDARD 11)
set(CMAKE_CXX_STANDARD 17)
set(CMAKE_EXPORT_COMPILE_COMMANDS ON)

set(PICO_BOARD pico_w CACHE STRING "Board type")
```

```
include(pico_sdk_import.cmake)

project(blink C CXX ASM)

pico_sdk_init()

add_executable(blink blink.c)

target_link_libraries(blink pico_stdlib)

if (PICO_CYW43_SUPPORTED)
    target_link_libraries(blink pico_cyw43_arch_none)
endif()

pico_add_extra_outputs(blink)
```

Na začátku kódu je uvedena **minimální verze CMake**, která zajišťuje kompatibilitu s verzí **3.13 a vyšší**. Dále jsou nastaveny **standarty jazyka C a C++** a je vygenerován soubor *compile_commands.json*, který je užitečný pro nástroje jako třeba IntelliSense v editoru, což značně usnadňuje programování.

Dále je nastavena **cílová deska**, kterou je možno snadno přepínat pomocí oficiálního rozšíření pro Pico ve VS Code. Následně se načte **Pico SDK**, nadefinuje se nový **CMake projekt** s názvem *blink* a podporou **C, C++ a assembleru**. Po následné inicializaci **Pico SDK** se připraví všechny potřebné **knihovny**.

Funkce *add_executable* vytvoří spustitelný program s názvem *blink* ze zdrojového souboru *blink.c*. Jde o tzv. **cílový název (target)**, který se dále používá při překladu programu. Pomocí funkce *target_link_libraries* se připojí standardní knihovna *pico_stdlib*, která poskytuje základní funkce pro práci s **hardwarem Pica**. Také je možno podmíněně připojit podporu pro komunikaci s **Wi-Fi čipem**, pokud byla zvolena deska Pico W.

Dále se pomocí funkce *pico_add_extra_outputs* vytvoří kromě základního **ELF** souboru, také další **výstupní formáty** jako třeba soubor **UF2**, který je potom možno na Pico nahrát přetažením tohoto souboru v režimu **USB Mass Storage** podobně jako při práci s běžným flash diskem.

Po importu projektu přes VS Code se do souboru *CMakeLists.txt* automaticky vygenerují další řádky, které využívá rozšíření pro Pico.

6.3 GPIO

Mikrokontrolery obvykle obsahují vstupně-výstupní piny, označované jako GPIO (General-purpose input/output). Tyto piny často nemají pouze jednu pevně danou funkci, ale jejich aktuálně potřebnou funkci je možno nastavit softwarově jako vstupy (např. pro čtení tlačítek nebo senzorů), nebo jako výstupy (např. ovládání LED diod). Kromě digitálních vstupů a výstupů mohou být vybaveny i dalšími periferiemi, jako jsou třeba analogové vstupy, rozhraní UART, SPI nebo I²C pro komunikaci s dalšími zařízeními. Dále mohou být vybaveny časovači a čítači pro generování signálů v určitých intervalech. [24]

Mikrokontrolery Pico a Pico 2 včetně jejich variant s Wi-Fi čipem mají shodné rozložení vyvedených pinů na desce. Na Pico je vyvedeno 26 GPIO pinů z původních 30 na čipu RP2040 nebo RP2350. Piny GPIO0 až GPIO22 jsou pouze digitální a piny GPIO26 až GPIO28 mohou navíc sloužit jako analogové vstupy. Pico používá pro označení GPIO pinů zkrácené názvy jako třeba GP22 místo GPIO22.


```
#include "pico/stdlib.h"

#define LED_PIN 20
#define BUTTON_PIN 14

int main() {
    gpio_init(LED_PIN); // inicializace GPIO pinu LED_PIN
    gpio_set_dir(LED_PIN, GPIO_OUT); // nastavení jako výstup

    gpio_init(BUTTON_PIN); // inicializace GPIO pinu BUTTON_PIN
    gpio_set_dir(BUTTON_PIN, GPIO_IN); // nastavení jako vstup
    gpio_pull_up(BUTTON_PIN); // Aktivace vnitřního pull up rezistoru

    while (true) {
        bool button = gpio_get(BUTTON_PIN); // čtení stavu pinu
        gpio_put(LED_PIN, !button); // nastavení hodnoty pinu
        sleep_ms(10); // čekej 10 ms
    }
}
```

6.4 AD převodník

AD převodník převádí analogový signál na digitální hodnotu. Mikropočítač RP2040 má tento převodník zabudovaný v sobě s rozlišením 12 bitů. To znamená, že může převést analogový signál na číselné hodnoty 0-4095 (2^{12}). V implementaci MicroPythonu se z důvodu kompatibility s dalšími mikropočítači převádí rozlišení z 12 bitů na 16, tedy hodnoty 0-65535 (2^{16}). [20]

Pico má celkem 5 analogových kanálů. Kanály 0, 1 a 2 jsou vyvedeny na piny GP26 až GP28. Kanál 3 slouží k měření systémového napětí VSYS a kanál 4 je napojen na senzor teploty uvnitř čipu RP2040. [23]

6.5 Softwarové zpoždění

Často je potřeba vytvořit určitý časový interval např. když se má po určité době změnit hodnota na výstupu. Toho lze dosáhnout několika způsoby s různou časovou přesností a náročností na procesor.

Jednou z možností, jak vytvořit softwarové zpoždění, je použití funkce **sleep**, která pozastaví běh programu na určitý čas. Pico SDK poskytuje funkce **sleep_ms()** a **sleep_us()**,

kteřé pozdrží program na požadovaný počet milisekund, respektive mikrosekund. Výhodou je snadné použití v programu a také šetří energii tím, že procesor přejde do režimu s nízkou spotřebou. Tento způsob je vhodný pro jednoduché časování jako třeba pro blikání LED diody. Nevýhodou může být nižší přesnost pro velmi krátké intervaly a hlavním problémem je, že během čekání se nemohou provádět jiné úlohy, protože je po celou dobu blokován procesor.

Další možností, kterou také poskytuje Pico SDK, jsou funkce **timer_busy_wait_ms()** a **timer_busy_wait_us()**. Tyto funkce opět pozdrží program na požadovaný počet milisekund, respektive mikrosekund. Na rozdíl od funkce `sleep`, se zde používá **aktivní čekání** (busy-wait), což znamená, že program během tohoto zpoždění neustále kontroluje, jestli už uplynul požadovaný čas. Výhodou je větší přesnost v porovnání s funkcí `sleep`. Nevýhodou je ale plné zatížení procesoru.

Také je možno vytvořit si vlastní zpoždění bez použití časovače. Toho se dosáhne pomocí prázdné smyčky, která opakuje jednoduchou operaci, jako je pouhá inkrementace počítadla, jen aby se zabíral čas. Takovéto zpoždění je velmi nepřesné a je velice závislé na rychlosti procesoru, nicméně pro jednoduché účely může stačit. Výhodou je také nezávislost na konkrétních knihovnách, a proto jej lze snadno použít v jiných projektech. Stejně jako předchozí možnosti je procesor plně vytižen a není možno provádět další úlohy během čekání.

6.6 Odezva programu

Se zpožděním programu souvisí tzv. **odezva programu** (program response). Tato odezva označuje, jak rychle dokáže mikrokontroler zareagovat na nějakou událost, jako třeba stisk tlačítka. Funkce jako `sleep_ms` využívá blokující čekání, tedy procesor v tuto dobu nic jiného nedělá. Během tohoto čekání není možno reagovat na události. Odezva je velmi špatná, protože nejdříve musí skončit blokující čekání a až poté je možno reagovat na další události.

Často je potřeba provádět současně více úloh, jako třeba když se má stav LED diody změnit každých 10 sekund a zároveň je třeba číst stav tlačítka. Pro takovýto případ není vhodné blokující čekání (např. pomocí `sleep`), protože znemožňuje okamžité zpracování dalších událostí během tohoto čekání. Efektivním řešením pro snížení odezvy programu může být použití tzv. dotazování (polling), kdy program místo skutečného čekání pouze pravidelně kontroluje, zda již uběhl požadovaný čas. Díky tomu může program v nekonečné smyčce střídavě provádět kontrolu tlačítka stavu tlačítka a nastavovat stav LED, pokud

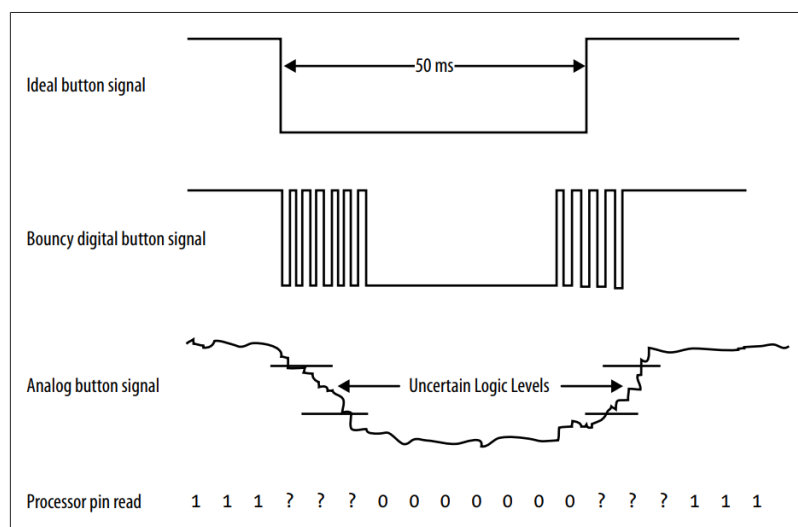
uplynul stanovený čas. Výhodou tohoto přístupu je, že procesor není blokován, odezva se zvýší a implementace je stále velmi jednoduchá. [25]

6.7 Základní tlačítek

Při stisku tlačítka může dojít k tzv. **zakmitávání tlačítka** (switch bouncing). Jedná se o jev, kdy se logická hodnota tlačítka nezmění skokově, ale po dobu několika milisekund nepravidelně přeskakuje mezi logickou nulou a logickou jedničkou. Nejedná se pouze o mechanické odskakování kontaktů při stisku tlačítka, ale také o elektrické jevy, jako je induktivní zvonění (ringing) nebo přenosové efekty na signálových vodičích. Tento jev může způsobit vícenásobné falešné detekce jednoho stisku, případně detekuje nesprávnou hodnotu. [25]

Softwarové řešení tohoto problému spočívá v tom, že se přečte stav tlačítka a po určitém čase (třeba 50 ms) se stav překontroluje. Pokud oba stavy souhlasí, lze předpokládat, že se nejedná o závit a tlačítko bylo stisknuto úmyslně. Výhoda tohoto řešení je, že nejsou potřeba žádné hardwarové součástky. Nevýhodou je zátěž procesoru a časová náročnost. Při stisku tlačítka, který je delší než doba potřebná pro odstranění závitů tlačítka, ale stále bude docházet k detekci dalších stisků. Řešení spočívá v detekci náběžné hrany. Toho lze docílit např. porovnáním aktuálního stavu tlačítka s jeho předchozím stavem.

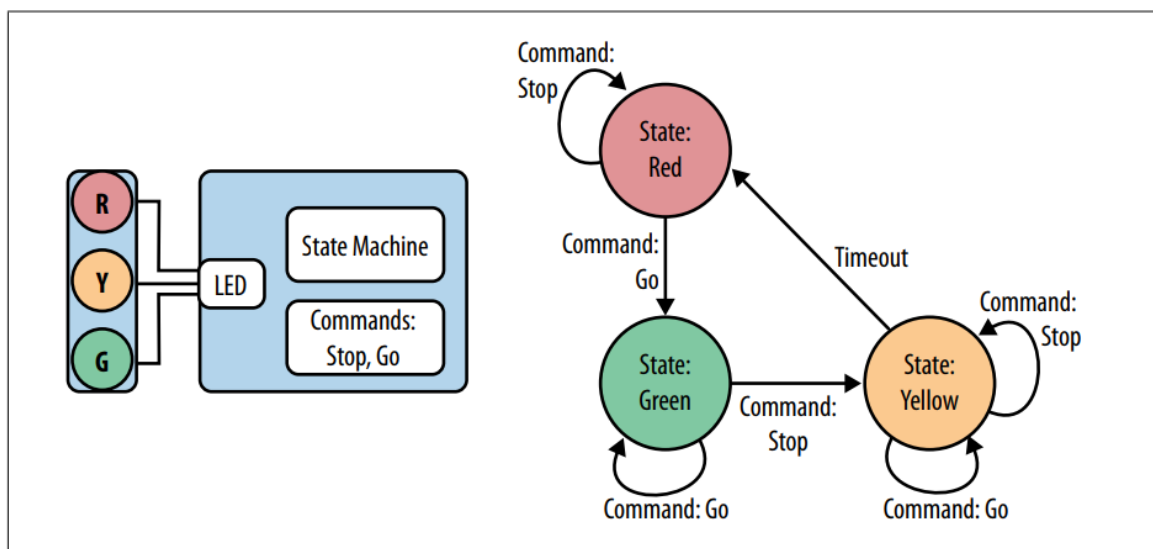
Hardwarové řešení spočívá v paralelním připojení kondenzátoru k tlačítku, tím se napětí na tlačítku bude měnit pomaleji. Výhoda je rychlejší odezva, ale v obvodu je potřeba použít kondenzátor. [31]



Obrázek 21 Základní tlačítek
(zdroj: [25])

6.8 Stavový automat

Stavový automat (state machine) patří mezi běžně používané návrhové vzory v embeddech systémech. Umožňuje strukturovaně řídit chování systému. Místo provádění stejné činnosti se rozhoduje na základě aktuálního stavu a příchozích událostí. Díky tomu je možné řídit složitější logiku. Stavový automat lze znázornit stavovým diagramem (flow chart). Stavové automaty lze také reprezentovat jako tzv. **konečné automaty** (finite state automata), kde je každý stav znázorněn kruhem a přechod mezi stavy je znázorněn šipkou. Obrázek znázorňuje stavový automat semaforu se třemi stavy (není uvažován stav, kdy svítí červená zároveň se žlutou). Mezi stavy na obrázku dochází k přechodům na základě příkazů (Go, Stop) a časovače (Timeout). V jazyce C může být stavový automat implementován použitím konstrukce switch-case. Každá větev (case) představuje jeden stav, ve kterém program zpracovává události, provádí potřebné akce a určuje přechod do dalšího stavu. [25]



Obrázek 22 Stavový diagram konečného automatu
(zdroj: [25])

6.9 RTOS

Operační systém reálného času – RTOS (Real-Time Operating System) je určený pro systémy, kde je nutná rychlá reakce na události, zejména v embeddech systémech. Zaručuje dokončení úkolu v daném časovém limitu. Charakteristikou RTOS systému je deterministické chování, nízká odezva a používání preemptivního plánování, které umožňuje okamžitě přerušit méně důležitý proces a uvolnění prostředků pro proces s vyšší prioritou. RTOS používá plánovač procesů (scheduler), který přiděluje běžícím procesům systémové prostředky procesoru na základě priorit a časových požadavků. [28]

Pro Pico je možno použít různé systémy RTOS, v této práci byl použit **FreeRTOS**, který je i oficiálně podporovaný v dokumentaci a je dostupný pro mikropočítače RP2040 a RP2350, a to jak pro architekturu Arm, tak i RISC-V. FreeRTOS je možno použít v režimu jednoho i dvou jader. Pico SDK poskytuje knihovny, které jsou kompatibilní s FreeRTOS. [17]

6.10 Čítač a časovač

Součástí mikropočítače RP2040 je i univerzální modul čítače a časovače. Tento modul umožňuje čítání impulsů, časování a generaci periodických pulzů. Mikropočítač RP2040 obsahuje 64bitový čítač, u kterého dochází k inkrementaci každou mikrosekundu díky přivedenému referenčnímu kmitočtu z krystalového oscilátoru. Hodnotu čítače lze vyčíst pomocí dvojice 32bitových registrů bez rizika race condition. Pomocí 4 alarmů je možno porovnávat spodních 32 bitů čítače s nastavenou hodnotou a při shodě se vygeneruje přerušení. Přetečení 64bitového čítače nehrozí, protože i při kmitočtu 1 MHz by trvalo tisíce let, než by čítač přetekl. [22]

6.11 Přerušení

Jedna z možností, jak okamžitě reagovat na událost je tzv. přerušení, to je metoda, kdy mikropočítač zareaguje na vzniklou událost. Tato událost může být stisk tlačítka, doběhnutí časovače, přetečení čítače, dokončení převodu v AD převodníku nebo třeba chyba při komunikaci s periferií. Když dojde k přerušení, procesor uloží svůj aktuální stav (registry a programový čítač) na zásobník. Následně se načte vektor přerušení do programového čítače. Tento vektor je adresa, na které se nachází obsluha přerušení, která provede požadovanou akci. Po provedené obsluze dojde k návratu z přerušení a obnoví se uložený stav procesoru ze zásobníku. Díky přerušení odpadá neustálá potřeba kontroly stavu zařízení a procesor může mezitím vykonávat jiné úlohy, dokud skutečně nedojde k události. Přerušení mohou mít různé priority. Pokud procesor obsluhuje přerušení s nižší prioritou a bylo vyvoláno přerušení s vyšší prioritou, dočasně přeruší zpracování a obslouží naléhavější událost. [24]

6.12 PWM

Digitální výstupy mikrokontroléru mohou mít pouze stavy vypnuto a zapnuto. Z tohoto důvodu není možno nastavit konkrétní analogovou hodnotu napětí např. pro řízení jasu diody. Takovouto analogovou hodnotu je možno nahradit pomocí tzv. **pulzně šířkové modulace** (Pulse Width Modulation – PWM). Toho je dosaženo pomocí opakujícího se

digitálního signálu, jehož střední hodnota napětí se mění podle délky trvání pulzů. Tzv. **střída** (duty cycle) udává, jak velká část jednoho cyklu je signál v aktivním (vysokém stavu). Například střída 25 % znamená, že signál je aktivní v 25 % jedné periody a 75 % periody je signál neaktivní. Pokud je amplituda výstupního signálu 3,3 V, potom střední hodnota napětí odpovídá čtvrtině této amplitudy, tedy přibližně 0,83 V. [29]

Každý GPIO pin na Picu umožňuje použití pulzně šířkové modulace. Modul PWM, který se nachází uvnitř mikrokontroleru, má 16 kanálů a je tvořen 8 částmi (**slices**), z nichž každá má 2 kanály. Kanály PWM jsou značeny písmenem a číslem. Číslo udává číslo slice a písmeno udává kanál na daném slice. Vždy je třeba se ujistit, že se nepoužijí vícekrát shodně označené PWM kanály na různých pinech. Jak je patrné z tabulky, některé GPIO piny, jako třeba GP0 a GP16 používají shodný kanál PWM 0A. Při nastavení tohoto kanálu jako výstupu se stejný PWM signál objeví na obou GPIO pinech. Při použití tohoto kanálu jako vstupu, potom daný slice PWM modulu čte logický OR těchto dvou GPIO.

Tabulka 3 Mapování PWM kanálů na GPIO mikropočítače RP2040

GPIO	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Kanál PWM	0A	0B	1A	1B	2A	2B	3A	3B	4A	4B	5A	5B	6A	6B	7A	7B
GPIO	16	17	18	19	20	21	22	23	24	25	26	27	28	29		
Kanál PWM	0A	0B	1A	1B	2A	2B	3A	3B	4A	4B	5A	5B	6A	6B		

7 UKÁZKOVÉ PROJEKTY

Ukázkové příklady vytvořené v rámci této práce jsou volně dostupné v git repozitáři na Githubu na adrese: https://github.com/dominstr/Pico_vyukove_prikklady.

V repozitáři s ukázkami se nachází složka *shared*. Tato složka obsahuje soubory sdílené mezi více projekty. Nachází se zde knihovny pro displej, FreeRTOS, parser pro JSON, konfigurační soubory pro Wi-Fi a také hlavičkový soubor **xplr.h**, který obsahuje definice čísel pinů pro snadnější použití v kódu. Aby bylo možné tento soubor naimportovat do vlastního projektu, je třeba přidat složku *shared* obsahující **xplr.h** do *CMakeLists.txt* následovně:

```
# Add the standard include files to the build
target_include_directories(blink PRIVATE
    ${CMAKE_CURRENT_LIST_DIR}
    ${CMAKE_SOURCE_DIR}/../shared
)
```

V nabídce rozšíření Raspberry Pi Pico Project se zvolí možnost **Configure CMake** a poté je možno tento soubor naimportovat pomocí **include** do hlavního .c souboru.

7.1 Import ukázek z Githubu do VS Code

Na úvodní stránce ve Visual Studio Code se zvolí možnost **Clone Git Repository**, kde se zadá adresa repozitáře: https://github.com/dominstr/Pico_vyukove_prikлады. Dále je třeba vybrat umístění do své uživatelské složky. Celá tato cesta **nesmí obsahovat diakritiku, ani speciální znaky** (podtržítka může být), jinak by nefungovala správně konfigurace CMake. Alternativně je možno stáhnout celý repozitář jako ZIP soubor a rozbalit jej ručně do své uživatelské složky.

7.2 Popis ukázkových projektů

Jednotlivé verze Pica mají shodné rozložení pinů, avšak varianty W neovládají vestavěnou LED diodu přímo, ale prostřednictvím Wi-Fi modulu. Ukázkové projekty byly stejně jako ukázky od Raspberry Pi psány univerzálně pro Pico i Pico W. Díky zpětné kompatibilitě tyto příklady fungují i pro Pico 2 a Pico 2 W. Každá z ukázek má ve své složce **Readme** soubor, který slouží jako dokumentace k ukázkovému projektu a blíže popisuje kód programu.

Část ukázek vychází z oficiálních příkladů, které poskytuje Raspberry Pi. Tyto ukázky jsou zamýšleny také jako seznámení uživatele s importem již existujících oficiálních ukázek skrz

rozšíření pro VS Code. Aby tyto oficiální ukázky bylo možno použít na vývojové desce Explorer Board, je třeba změnit použité GPIO piny nebo nastavit sériovou komunikaci. Pro odlišení jsou takovéto ukázky označeny koncovkou „_from_example“. V případě potřeby je možno i tyto ukázky přímo spustit bez nutnosti importovat jejich oficiální předlohy.

Ukázkové programy nejsou členěny do dalších složek, ale v rámci Readme souboru jsou rozděleny do několika tematických bloků podle zaměření nebo složitosti.

7.2.1 Základní ukázky

Tyto ukázky se zaměřují na úplné základy programování Pica. Obsahují jednoduché příklady jako je práce s vestavěnou diodou, čtení stavu tlačítek a nastavení sériové komunikace. Těmito ukázkami je vhodné začít, protože poskytují základy, ze kterých vycházejí složitější ukázkové příklady.

7.2.2 Senzory

V této kategorii se nachází ukázka pro čtení dat ze senzoru DHT11, který ke komunikaci používá svůj vlastní protokol a dále je zde ukázka pro čtení analogových hodnot z potenciometru. Místo potenciometru lze v případě potřeby použít i fotorezistor.

7.2.3 Výstupní zařízení

V této části se nachází ukázky zaměřené na ovládání základních výstupních zařízení. Konkrétně se jedná o příklad pro ovládání relé, které může sloužit ke spínání zátěže a dále je zde příklad pro pasivní bzučák, který je řízen pomocí pulzně-šířkové modulace.

7.2.4 Řízení LED diod

Tato skupina ukázek se zaměřuje na různé způsoby ovládání LED diod, včetně řízení jasu pomocí PWM, a to i v kombinaci s potenciometrem. Také je zde řízení světelného semaforu několika způsoby včetně konečného automatu. Dále je zde popsána práce s adresovatelnými NeoPixel diodami, jejichž programování vyžaduje použití PIO.

7.2.5 Pokročilé ukázky

V poslední kategorii ukázek jsou obsaženy složitější příklady, jako je ovládání TFT displeje pro grafické zobrazování dat, práce s časovačem pro přesnější odměřování času, komunikace přes Wi-Fi včetně získání dat pomocí HTTP požadavku. Dále ukázky obsahují příklad pro odezvu programu a implementaci systému FreeRTOS.

7.2.6 Komplexní ukázkový program

Součástí pokročilých ukázek je i větší ukázka *weather_station*, která představuje jednoduchou meteostanici s více zdroji dat. Pomocí senzoru DHT11 se měří aktuální teplota a vlhkost v místnosti. Zároveň je zařízení připojeno k síti Wi-Fi, přes kterou získává data o aktuálním počasí z OpenWeather API pro tři města přednastavená v kódu. Data jsou získána ve formátu JSON a jsou dále zpracována. Přepínání mezi jednotlivými městy se provádí dvěma tlačítky, která umožňují cyklování mezi těmito městy. Reakce na tlačítka je řešeno prostřednictvím přerušením.

Naměřené údaje a také údaje získané z API jsou následně zobrazeny na TFT displeji. Font displeje neumí zobrazit českou diakritiku, proto je grafický výstup na displeji v anglickém jazyce. V horní části displeje je zobrazeno vybrané město zelenou barvou. Údaje z OpenWeather API, jako teplota, vlhkost vzduchu, atmosférický tlak, rychlost a směr větru jsou psány bílým písmem. Data ze senzoru DHT11, který měří vnitřní teplotu a vlhkost vzduchu, jsou zobrazeny modrým písmem. Data se aktualizují po startu programu a poté každou další minutu.

Tato ukázka používá knihovny třetích stran pro TFT displej, parsování JSON a oficiální ukázky od Raspberry Pi pro komunikaci s Wi-Fi. Ukázka byla napsána v jazyce C++, pro potřeby komunikace s TFT displejem.



Obrázek 23 Výstup z programu *weather_station*
(zdroj: vlastní)

8 ZÁVĚR

Cílem této práce bylo seznámení se s vývojovou deskou Raspberry Pi Pico a prozkoumání možností jejího využití pro výuku programování mikropočítačů. Pro usnadnění výuky bylo třeba vybrat vhodnou hardwarovou platformu a na základě této volby vytvořit sadu ukázkových programů, které by pokrývaly základní obsluhu periférií obsažených na této platformě.

Nejdříve byl proveden teoretický rozbor pro seznámení s vývojovou deskou Raspberry Pi Pico, možnostech jejího programování a dostupné dokumentace. Z různých komerčních hardwarových platforem byla vybrána rozšiřující deska Explorer Board od firmy Joy-IT. Pro tuto desku byly následně napsány ukázkové programy ve Visual Studio Code za pomoci oficiálního rozšíření.

Vývojová deska Raspberry Pi Pico nabízí velmi dobrý poměr cena/výkon a práce s ní je relativně jednoduchá. Z hlediska programování je velice snadné použití jazyka MicroPython, který má výrazně jednodušší syntaxi než jazyk C a také umožňuje snadnější ovládání NeoPixel LED diod a TFT displeje pomocí externích knihoven. Pro programování v MicroPythonu lze navíc použít Thonny IDE. Programování v jazyce C je přeci jen trochu složitější z hlediska kompilace a nahrávání kódu na mikrokontrolér. Tuto práci značně usnadňuje oficiální rozšíření pro Visual Studio Code, kde lze tyto věci provést pohodlně tlačítkem namísto psaní příkazů do konzole.

Pro Pico je dostupné velké množství výukových materiálů, podrobná oficiální dokumentace a také řada knih v tištěné i elektronické podobě. Tyto materiály jsou většinou dostupné v anglickém jazyce, ale několik materiálů existuje i v českém jazyce. Raspberry Pi také poskytuje ukázkové příklady pro jazyky Micropython a C.

Zvolená rozšiřující deska Explorer Board od firmy Joy-IT se ukázala jako dobrá volba. Obsahuje většinu potřebných periférií, a to i s možností jejich odpojení přepínačem, kdyby bylo potřeba použít piny pro jiné účely. Výhodou je i přítomné resetovací tlačítko pro usnadnění nahrávání programu. I když umístění tohoto tlačítka není ideální, lze Pico relativně bez obtíží resetovat jednou rukou současným držením tlačítka BOOTSEL na Picu a resetovacího tlačítka na rozšiřující desce. Desce lze vytknout, že místo klasických LED diod obsahuje NeoPixel LED diody, jejichž programování je v jazyce C výrazně složitější než v jazyce MicroPython. Deska také nenabízí žádnou komponentu pro analogového

veličiny. Z tohoto důvodu byly dodatečně přidány LED diody a potenciometr na nepájivé pole pro využití v ukázkových příkladech.

Pro zvolenou desku byly napsány ukázkové příklady v jazyce C, které popisují základní obsluhu všech periférií na vývojové desce. Součástí je i ukázka projektu meteostanice, který kombinuje práci s displejem, senzorem DHT11, Wi-Fi a získání aktuálního počasí z OpenWeather API. Všechny tyto ukázky jsou volně dostupné ve veřejném repozitáři na Githubu. Tyto ukázky mohou být dále rozšířeny o další příklady různých senzorů, použití I²C sběrnice nebo například Bluetooth pro další možnost bezdrátové komunikace.

SEZNAM POUŽITÉ LITERATURY

- [1] *Raspberry Pi* 5. Online. 2025. Dostupné z: <https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf>. [cit. 2025-03-02].
- [2] *DeepBlueMbedded*. Online. 2025. Dostupné z: <https://deepbluembedded.com> [cit. 2025-03-09].
- [3] *Picopad*. Online. 2025. Dostupné z: <https://github.com/Pajenicko/Picopad> [cit. 2025-04-21].
- [4] *DFRobot*. Online. 2025. Dostupné z: <https://www.dfrobot.com> [cit. 2025-03-25].
- [5] *Seed Studio Wiki Platform*. Online. 2025. Dostupné z: <https://wiki.seeedstudio.com> [cit. 2025-03-11].
- [6] *Botland*. Online. 2025. Dostupné z: <https://botland.cz> [cit. 2025-03-09].
- [7] *RPishop.cz*. Online. 2025. Dostupné z: <https://rpishop.cz> [cit. 2025-05-02].
- [8] *Joy-IT*. Online. 2025. Dostupné z: <https://joy-it.net/en/> [cit. 2025-03-09].
- [9] Finio Ben. *Tinkercad + Arduino Lesson 6: Pull-up and Pull-down Resistors*. Online. 2023. Dostupné z: <https://www.youtube.com/watch?v=aPVMKyZpaPA>. [cit. 2025-04-13].
- [10] *OCFreaks!*. Online. 2025. Dostupné z: <https://www.ocfreaks.com> [cit. 2025-03-09].
- [11] *Joy-IT*. Online. 2025. Dostupné z: <https://joy-it.net/en/> [cit. 2025-03-09].
- [12] *Oficiální stránka Raspberry Pi*. Online. 2025. Dostupné z: www.raspberrypi.com. [cit. 2025-03-02].
- [13] *Oficiální stránka Picopad*. Online. 2025. Dostupné z: <https://picopad.eu/cs/>. [cit. 2025-03-02].
- [14] FROMAGET, Patrick. The Epic Story of the Raspberry Pi. Online. In: . Dostupné z: <https://raspberrytips.com/raspberry-pi-history/>. [cit. 2025-03-02].

- [15] *Getting started with Raspberry Pi Pico-series*. Online. 2024. Dostupné z: <https://datasheets.raspberrypi.com/pico/getting-started-with-pico.pdf>. [cit. 2025-03-02].
- [16] RASPBERRY PI. *Pico-series Microcontrollers*. Online. Dostupné z: <https://www.raspberrypi.com/documentation/microcontrollers/pico-series.html>. [cit. 2025-03-02].
- [17] *Raspberry Pi Pico-series C/C++ SDK*. Online. 2025. Dostupné z: <https://datasheets.raspberrypi.com/pico/raspberry-pi-pico-c-sdk.pdf>. [cit. 2025-04-26].
- [18] *Connecting to the Internet with Raspberry Pi Pico W-series*. Online. 2024. Dostupné z: <https://datasheets.raspberrypi.com/picow/connecting-to-the-internet-with-pico-w.pdf>. [cit. 2025-04-26].
- [19] *Pull up rezistor*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-2025, 2024-06-04. Dostupné z: https://cs.wikipedia.org/wiki/Pull_up_rezistor. [cit. 2025-04-27].
- [20] *Getting Started with MicroPython*. Online. Dostupné z: https://docs.sunfounder.com/projects/umsk/en/latest/04_pi_pico/pico_start/00_pico_start.html. [cit. 2025-03-02].
- [21] *Raspberry Pi Pico-series Python SDK*. Online. 2024. Dostupné z: <https://datasheets.raspberrypi.com/pico/raspberry-pi-pico-python-sdk.pdf>. [cit. 2025-04-26].
- [22] *RP2040 Datasheet*. Online. 2025. Dostupné z: <https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf>. [cit. 2025-04-26].
- [23] IBRAHIM, Dogan. *Raspberry Pi Pico Essentials*. Elektor International Media, 2021. ISBN 978-3-89576-427-1.
- [24] CATSOULIS, John. *Designing Embedded Hardware*. 2nd Edition. O'Reilly Media, 2005. ISBN 978-0596007553.
- [25] WHITE, Elecia. *Making embedded systems*. Sebastopol: O'Reilly, [2012]. ISBN 978-1-4493-0214-6.

- [26] *ARM*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-2025. Dostupné z: <https://cs.wikipedia.org/wiki/ARM>. [cit. 2025-05-10].
- [27] *RISC-V*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-2025. Dostupné z: <https://cs.wikipedia.org/wiki/RISC-V>. [cit. 2025-05-10].
- [28] *Operační systém reálného času*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-2025. Dostupné z: https://cs.wikipedia.org/wiki/Operační_systém_reálného_času. [cit. 2025-05-10].
- [29] PINKER, Jiří. *Mikroprocesory a mikropočítače*. Praha: BEN - technická literatura, 2004. ISBN 80-7300-110-1.
- [30] CHRÁSKA, Jiří. *Materiály pro výuku na střední škole SSPVC*. Online. Střední škola – Podorlické vzdělávací centrum, Dobruška. Dostupné z: <https://sspvc.lixis.cz>. [cit. 2025-04-27].
- [31] CHOVANEC, Tomáš. *Cvičení z Mikroprocesorové techniky*. Online. Dostupné z: <https://tomaschovanec.github.io/MIT/>. [cit. 2025-04-27].
- [32] *Raspberry Pi Pico SDK Examples*. Online. Dostupné z: <https://github.com/raspberrypi/pico-examples>. [cit. 2025-04-27].
- [33] TIŠNOVSKÝ, Pavel. *Programovatelné IO na mikrořadičích RP2040 v Raspberry Pi Pico*. Online. In: Root. 2025. Dostupné z: <https://www.root.cz/clanky/programovatelne-io-na-mikroradicich-rp2040-v-raspberry-pi-pico/>. [cit. 2025-03-02].
- [34] LINUXDAYS. *Picopad - open source herní konzole (Igor Šverma)*. Online. 2023. Dostupné z: <https://www.youtube.com/watch?v=ODtLUG2z0T0>. [cit. 2025-04-27].
- [35] OPENALT. *Štěpán Bechynský: Raspberry Pi Pico a Python pro začátečníky*. Online. 2023. Dostupné z: <https://www.youtube.com/watch?v=yMk6SSBKK3o>. [cit. 2025-04-27].

SEZNAM OBRÁZKŮ

Obrázek 1 Raspberry Pi 5 (<i>zdroj: [1]</i>)	15
Obrázek 2 Varianty Raspberry Pi Pico (<i>zdroj: [2]</i>)	16
Obrázek 3 Open source herní konzole Picopad (<i>zdroj: [3]</i>)	21
Obrázek 4 Testování na nepájivém poli (<i>zdroj: vlastní</i>)	26
Obrázek 5 Gravity Expansion Board (<i>zdroj: [4]</i>)	27
Obrázek 6 Rozvržení Grove shieldu (<i>zdroj: [5]</i>)	28
Obrázek 7 Maker Pi Pico Mini (<i>zdroj: [6]</i>)	29
Obrázek 8 Pico Breadboard KIT (<i>zdroj: [7]</i>)	29
Obrázek 9 Vývojová deska ArdiPi (<i>zdroj: [6]</i>)	30
Obrázek 10 Vývojová deska Explorer Board (<i>zdroj: [8]</i>)	31
Obrázek 11 Rozšíření Raspberry Pi Pico pro VS Code (<i>zdroj: vlastní</i>)	35
Obrázek 12 Založení nového C/C++ projektu (<i>zdroj: vlastní</i>)	36
Obrázek 13 Hláška vyžadující instalaci driveru přes program Zadig (<i>zdroj: vlastní</i>)	37
Obrázek 14 Instalace driveru přes program Zadig (<i>zdroj: vlastní</i>)	37
Obrázek 15 Zapojení Raspberry Pi Debug Probe (<i>zdroj: [12]</i>)	38
Obrázek 16 Zapojení druhého Pica ve funkci Debug Probe (<i>zdroj: [12]</i>)	38
Obrázek 17 Výpis textu na sériové lince (<i>zdroj: vlastní</i>)	40
Obrázek 18 Zapojení pull up a pull down rezistorů (<i>zdroj: [9]</i>)	41
Obrázek 19 Rozložení pinů pro DHT22 a DHT11 (<i>zdroj: [10]</i>)	42
Obrázek 20 Rozložení pinů pro variantu Pico (<i>zdroj: [12]</i>)	48
Obrázek 21 Zákmit tlačítka (<i>zdroj: [25]</i>)	51
Obrázek 22 Stavový diagram konečného automatu (<i>zdroj: [25]</i>)	52
Obrázek 23 Výstup z programu weather_station (<i>zdroj: vlastní</i>)	57

SEZNAM TABULEK

Tabulka 1 Porovnání Raspberry Pi Pico a Arduino UNO	22
Tabulka 2 Zapojení TFT displeje na desce Explorer Board	43
Tabulka 3 Mapování PWM kanálů na GPIO mikropočítače RP2040	54

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

AD	Analogově-digitální
API	Application programming interface
ARM	Advanced RISC Machine
BL	Backlight
CISC	Complex instruction set computer
CS	Chip select
COM	Common
DC	Data/Command
GND	Ground
GPIO	General-Purpose Input/Output
HW	Hardware
IDE	Integrated development environment
I ² C	Inter-Integrated Circuit
IoT	Internet of Things
LED	Light-emitting diode
MISO	Master In Slave Out
MOSI	Master Out Slave In
MQTT	MQ Telemetry Transport
NC	Normally closed
NO	Normally open
PIO	Programmable input-output
PWM	Pulse-width modulation
REPL	Read-eval-print loop
RFID	Radio-frequency identification
RGB	Red-green-blue

RISC	Reduced instruction set computer
RST	Reset
RTOS	Real-time operating system
SCK	Serial clock
SCL	Serial clock
SDK	Software development kit
SPI	Serial Peripheral Interface
TFT	Thin-film transistor
UART	Universal asynchronous receiver-transmitter
UHF	Ultra high frequency
USB	Universal Serial Bus
VS	Visual Studio

SEZNAM PŘÍLOH

Příloha P I: Ukázkové příklady

PŘÍLOHA P I: UKÁZKOVÉ PŘÍKLADY

Příloha ve formátu ZIP obsahuje soubory s ukázkovými příklady pro vývojovou desku Raspberry Pi Pico a její použití na rozšiřující desce Explorer Board. Každá ukázka má svoji složku včetně dokumentace ve formátu Readme souboru. Seznam ukázkových příkladů:

- `blink_from_example`
- `blink_traffic_lights`
- `button_debounce`
- `buzzer`
- `dht_from_example`
- `freertos_setup`
- `hello_adc_from_example`
- `hello_st7735`
- `http_request`
- `neopixel_leds`
- `potentiometer_brightness`
- `pwm_setup`
- `read_button`
- `relay`
- `response`
- `serial_setup_from_example`
- `time_measure`
- `timer_blink`
- `traffic_lights_state_machine`
- `weather_station`

V ukázce se dále nachází složka *shared*, která obsahuje potřebné knihovny třetích stran a složka *readme_images*, ve které jsou obrázky použité v dokumentaci jednotlivých ukázek.