Obsah obrázku symbol, logo, Písmo, Grafika

Obsah vygenerovaný umělou inteligencí může být nesprávný.

Kalkulačka v postfixové notaci na čipu RP2040

Křesťanské gymnázium, Kozinova 1000, 102 00 Praha 10 – Hostivař

Ondřej Běhal, 2025/26

|  |  |
| --- | --- |
| **Maturitní práce z informatiky**  ZADÁNÍ | |
| **Školní rok:** 2025/26 | **Škola:**  Křesťanské gymnázium  Kozinova 1000  Praha 10 - Hostivař  102 00 |
| **Jméno žáka:** | Ondřej Běhal |
| **Název maturitní práce:** | Kalkulačka v postfixové notaci na čipu RP2040 |
| **Termín odevzdání:** | 9. dubna 2026 |
| **Vedoucí maturitní práce:** | Mgr. Ivona Spurná |
| **Oponent maturitní práce:** |  |
| **Délka obhajoby:** | 15 min |
| **Způsob zpracování maturitní práce:** | Žák zpracuje **maturitní práci v elektronické podobě** v programu vhodném ke zpracování – dle charakteru práce.  Průběžné výstupy i finální výstup odevzdá elektronicky do vyhrazeného místa určeném vedoucím maturitní práce.  Dále žák připraví **prezentaci**, která bude dokumentovat danou práci, její průběh a závěrečný stav. Tato prezentace bude taktéž odevzdána jako soubor do vyhrazeného místa určeném vedoucím maturitní práce.  Dále žák připraví textovou zprávu – **dokumentaci** – v rozsahu a obsahu definovaném ve *Specifikaci textové části maturitní práce z informatiky*, která je na školních webových stránkách. Tuto dokumentaci odevzdá jak elektronicky v editovatelné podobě (MS Word) a verzi pro tisk (PDF) do vyhrazeného místa určeném vedoucím maturitní práce, tak v tištěné podobě v jednom výtisku.  Z hlediska autorství musí být práce provedena výše uvedeným žákem. Uvede použité zdroje v dokumentaci i prezentaci. |
| **Hodnocení:** | Z provedení maturitní práce (tj. vlastní praktická práce, soubory prezentace a dokumentace) získá žák na konci školního roku známku odpovídající celkovému bodovému zisku z výstupů.  100–90 bodů – výborný 89–75 bodů – chvalitebný 74–50 bodů – dobrý 49–25 bodů – dostatečný 24–0 bodů – nedostatečný  Celkové hodnocení maturitní zkoušky z informatiky: celková známka bude reflektovat z 50 % získané hodnocení z provedení maturitní práce (viz výše) a z 50 % ústní obhajobu této práce u maturitní zkoušky. Žádná z těchto známek nesmí být 5 (nedostatečná), jinak je celkové hodnocení 5 (nedostatečná). |
| **Výstupy:** | 1. výstup – 1. část maturitní práce – max 15 bodů 2. výstup – 2. část maturitní práce – max 15 bodů 3. výstup – 3. část maturitní práce – max 15 bodů 4. výstup – 4. část maturitní práce – max 15 bodů 5. výstup – soubory prezentace a dokumentace – max 40 bodů |
| **Termíny výstupů:** | 1. výstup – 13. října 2025 2. výstup – 1. prosince 2025 3. výstup – 12. ledna 2026 4. výstup – 23. února 2026 5. výstup – 9. dubna 2026 |
| **Zadání maturitní práce:** | Cíl práce:  Vytvořit jednoduchou kalkulačku využívající postfixovou notaci matematických výrazů (také známá jako Reverzní polská notace – RPN), jež je založena na zásobníku (stack) a nevyžaduje používání závorek ani vyhodnocování precedence operací, neboť pořadí operací je již v zápisu samotném. Kalkulačka není programem běžícím pod operačním systémem, ale je naprogramovaná na „holém železe“ (bare metal) přímo na mikrořadiči RP2040 od britské firmy Raspberry Pi Ltd., konkrétně na vývojové desce Raspberry Pico. Výstup ukazuje uživateli na 128x64bodovém OLED displeji založeném na čipu SSD1306. Vstup obdrží přes klávesnici přes sériové rozhraní UART nebo SWD.  Průběžné výstupy:   1. výstup: Žák dokáže využít již existující ekosystém knihoven a nástrojů, aby dokázal napsat kód pro mikrořadič, který posléze umí zkompilovat, nahrát na mikrořadič přes ladící sériové rozhraní SWD a spustit svůj program. 2. výstup: Žák dokáže pomocí vhodných knihoven komunikovat s OLED displejem přes sběrnici I2C, dokáže na displej vykreslovat jednoduché obrazce i text. 3. výstup: Žák dokáže vytvořit objekt reprezentující výše zmíněný zásobník. Dokáže ho inicializovat, přidat položku na jeho vrchol, odebrat položku z jeho vrcholu, dotázat se na vrchní položku bez jejího odstranění, a dotázat se na prázdnost zásobníku. 4. výstup: Žák dokáže vykreslovat zásobník na displej, i když se na displej nevejde celý; dokáže na něj přidávat nové číselné hodnoty a na nich provádět matematické operace jako sčítání, odčítání, násobení, dělení. |

# Prohlášení

Prohlašuji, že jsem maturitní práci zpracoval samostatně a že jsem uvedl všechny použité informační zdroje a literaturu v seznamu použitých zdrojů.

# Poděkování

# Abstrakt

Tato maturitní práce se zabývá vytvořením jednoduché kalkulačky vytvořené v programovacím jazyce Rust. Program kalkulačky je spouštěn na mikrořadiči RP2040 od společnosti Raspberry Pi Ltd., resp. na jeho vývojové desce Raspberry Pi Pico od stejného výrobce, a svůj výstup vykresluje na OLED displeji. Kalkulačka používá tzv. postfixovou notaci matematických výrazů (též známá jako reverzní polská notace – RPN), kdy operátor (například +) následuje *po* obou hodnotách, na kterých má být operace vykonána. Cílem práce bylo se naučit pracovat s čipem RP2040, obzvláště při jeho programování v jazyce Rust, a ne v jazycích C a C++, které podporuje oficiální balíček pro vývoj softwaru od výrobce čipu.

## Klíčová slova

Kalkulačka, Rust, RP2040, Raspberry Pi Pico, nízkoúrovňové programování, postfixová notace

# Abstract (in English)

This graduation thesis focuses on creating a simple calculator created in the programming language Rust. The programme is run on the RP2040 microcontroller made by Raspberry Pi Ltd., respectively its development board Raspberry Pi Pico by the same maker, and it renders its output on an OLED display. The calculator uses the so-called postfix notation of mathematical expressions (also known as reverse Polish notation – RPN), when the operator comes *after* both operands, on which the operation shall be carried out. The goal of the project was to learn to work with the RP2040 chip, especially when programming it in the Rust language, and not the C or C++ languages, which are supported by the official software development kit made by the chip maker.

## Keywords

Calculator, Rust, RP2040, Raspberry Pi Pico, low-level programming, postfix notation

Obsah

[Prohlášení 4](#_Toc209783467)

[Poděkování 5](#_Toc209783468)

[Abstrakt 6](#_Toc209783469)

[Klíčová slova 6](#_Toc209783470)

[Abstract (in English) 6](#_Toc209783471)

[Keywords 6](#_Toc209783472)

[1. ÚVOD 9](#_Toc209783473)

[2. TEORETICKÁ ČÁST 10](#_Toc209783474)

[2.1 Mikrořadiče 10](#_Toc209783475)

[3. PRAKTICKÁ ČÁST 11](#_Toc209783476)

[4. ZÁVĚR 12](#_Toc209783477)

[Zdroje 13](#_Toc209783478)

# 1. ÚVOD

Tato maturitní práce se zaměřuje na vytvoření funkční kalkulačky. Hlavním cílem je vytvořit program, který dokáže přijímat vstup od uživatele, provádět matematické operace a vykreslovat výstup zpět na displej.

Součástí práce je také použití již existujícího *toolchainu* jazyka Rust, jako i mnoha knihoven a projektů, pro jednodušší tvorbu samotného programu a jeho následného nahrání na paměť flash, ze které čip čte své instrukce. Projekt slouží jako praktická ukázka nízkoúrovňového programování v jazyce Rust namísto obvyklého C nebo C++.

Výsledkem je jednoduchá kalkulačka v postfixové notaci, která lze pak dále rozšířit či upravit. Zároveň představuje zkušenost v programování mikrořadičů, což je mnohdy složitější pro začátečníky a pro což není online zdaleka tolik tutoriálů.

# 2. TEORETICKÁ ČÁST

Není žádným tajemstvím, že celý náš současný svět je postaven na počítačích. Ale ne každý počítač musí být krabice ležící na stole. Existují i drobné počítače, které zabírají prostor jednotek centimetrů krychlových, které mají omezený výkon a specifický účel. Těmto typům počítačů říkáme mikropočítače. Najdeme je prakticky všude kolem nás: od domácích spotřebičů jako pračka nebo mikrovlnka, přes robotické vysavače a chytré žárovky, až po industriální procesy, kde se čas měří na mikrosekundy a kde to nejdrobnější opoždění může mít vážné následky.

S těmito mikropočítači se zachází zcela jinak než s běžnými počítači, jinak se napájí, jinak s nimi komunikujeme, jinak je programujeme. A i to, jakým způsobem jsou programovány, se může mnohdy lišit od klasického způsobu programování.

## 2.1 Mikrořadiče

Konkrétní čipy, které jsou pomyslnými srdci těchto mikropočítačů, nazýváme mikrořadiče (nebo mikrokontroléry, z anglického „microcontroller“). Tyto čipy do sebe integrují procesor, paměť, a obvykle i další periferie usnadňující komunikaci s jinými zařízeními.

Procesor („CPU = central processing unit“) je mozkem samotného mikrořadiče, vykonává strojový kód – instrukce v jazyku symbolických adres („assembly“) odpovídající instrukční sadě daného CPU. Ačkoli současné počítače používají až na pár výjimek (jako např. M-čipy v nových Apple počítačích) instrukční sadu x86-64, v kontextu mikrořadičů se používají častěji jednodušší a energeticky efektivnější sady jako ARM, Atmel AVR nebo RISC-V. Procesor pracuje s registry, což je několik míst přímo uvnitř procesoru, které uchovávají data. Obvykle je jich sice velmi málo (několik desítek) a každý z nich dokáže uchovat jen jedno „slovo“ („word – jednotka informace závislá na typu procesoru, u 32bitových procesorů je rovna 32 bitům, u 64bitových je to 64 bitů atd.), ale jsou velmi těsně přimknuty k procesoru, a tudíž velmi rychlé, a jen s daty v nich dokáží instrukce pracovat. Ostatní data musí být do nich zkopírována z adresního prostoru (viz níže).

Paměť dělíme na paměť flash, RAM, ROM a OTP. Paměť flash je přepisovatelná, ale poněkud pomalá, takže ji CPU používá ke čtení dat, které se nemění (jako třeba instrukce, které má vykonávat). Její výhodou je schopnost uchovat si data i po odpojení napájení. Paměť RAM (=random access memory – náhodně přistupovaná paměť) si sice data neuchovává, ale je možné do ní poměrně rychle přistupovat a přepisovat ji např. pro mezivýsledky operací a jiných dat, se kterými procesor pracuje. Procesor s ní pracuje tak, že ji namapuje do svého adresního prostoru (viz níže). Paměť ROM (=read only memory – paměť pouze pro čtení) je nepřepsatelná a její obsah je určen již během výroby čipu, proto do ní ukládá data výrobce čipu, ať už jsou to informace o čipu samém, nebo kód pro inicializaci čipu a spuštění uživatelského programu. Do OTP (=one-time programmable – jednorázově programovatelná) paměti se sice zapisovat dá, ale pouze jednou: po prvním přepsání již nelze změnit. Uchovávají se v ní např. kryptografické klíče nebo jiná data, které si uživatel čipu nastaví obvykle již v továrně při vložení čipu do koncového produktu, a které uživatel koncového produktu nesmí změnit.

Pro přístup k čemukoli jinému než vlastním registrům potřebuje procesor adresy – číselné hodnoty jednoznačně označující místo v paměti nebo registr periferie. Rozsah všech těchto adres se nazývá adresní prostor a

Periferie jsou další jednotky na čipu, které slouží k jiné funkci než vykonávání instrukcí a uchovávání dat a které lze ovládat z programu právě tak, že jsou namapovány do adresního prostoru CPU. Některé z nich jsou zodpovědné za základní funkce nutné ke správnému fungování mikrořadiče, jiné za komunikaci s okolním světem skrze elektrické signály, další za různé interní funkce. Periferie komunikující s okolním světem jsou například univerzální vstup-výstup („GPIO = general purpose input/output“), periferie implementující určitý komunikační protokol, nebo i generátory signálu PWM (=pulse width modulation – signál modulovaný šířkou pulzu) a analogově-digitální převodníky („ADC = analog-digital converter“). GPIO je asi nejjednodušší z nich, pouze přepínající elektrický signál mezi vypnuto a zapnuto.

– hodinový signál taktující celý systém, aby se všechno dělo ve správný moment, komunikace s pamětí flash, nebo tzv. hlídací pes („watchdog“), což je speciální časovač, který, když vyprší, vynutí restart CPU a který je potřeba pravidelně resetovat na nenulovou hodnotu v místech, které stanoví programátor. Watchdog slouží k automatickému zotavení se v případě, kdy systém zamrzne. Další periferie slouží ke komunikaci mikrořadiče s okolním světem – univerzální vstup-výstup („GPIO = general purpose input/output“), který umožňuje programu vytvořit jednoduchý elektrický signál zapnuto-vypnuto na elektrickém vývodu („pinu“) mikrořadiče; periferie implementující obvyklé komunikační protokoly jako UART, I2C nebo SPI, které si rezervují jeden nebo více pinů, po kterých vysílají či přijímají elektrické signály;

# 3. PRAKTICKÁ ČÁST

# 4. ZÁVĚR

# Zdroje