

# Emulátor ARMv6 procesoru pro emulaci prostředí Raspberry Pi

Bc. Jakub Šilhavý

vedoucí práce: Ing. Martin Úbl

Katedra informatiky a výpočetní techniky  
Fakulta aplikovaných věd  
Západočeská univerzita v Plzni

17. 06. 2024

## Cíl práce

- návrh a implementace emulátoru platformy **Raspberry Pi Zero**
  - emulace instrukcí procesoru ARM1176JZF-S (ARMv6)
  - emulace základních periferií µC BCM2835 (GPIO, Mini\_UART, BSC, ...)

### ① vzdělávací účely

- vizualizace principů OS
- embedded vývoj

### ② testování a ladění SW

### ③ prototypování HW

- připojení externích periferií
- návrh vlastního systému



Figure 1: Raspberry Pi Zero

- použití **KIV-RTOS** pro ověření správnosti výsledného emulátoru

## ARM procesory a jejich aplikace



Figure 2: ARM aplikace

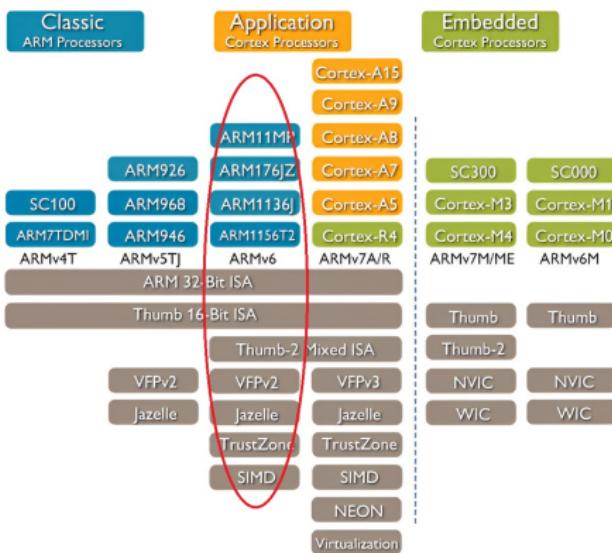


Figure 3: ARM CPU roadmap

- vhodné pro seznámení se s programováním v JSA
  - vizualizace, ladění programu
- floating-point instrukce
- platformová nezávislost

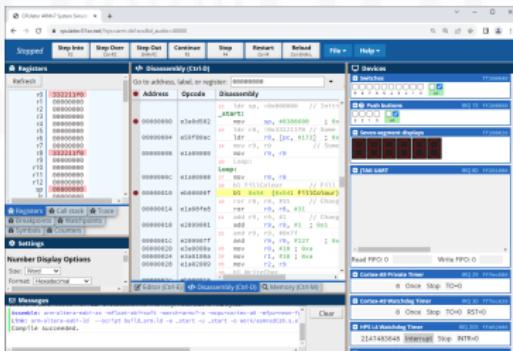


Figure 4: ▶ CPUlator

## Omezení

- emulace pouze CPU (ne celého SoC)
- minimální podpora pokročilých systémových operací
  - ⇒ nevhodné pro testování principů OS
- limitovaná podpora připojení externích periferií



Figure 5: ▶ QEMU

- podpora vícero architektur
  - x86, MIPS, ARM, ...
- připojení externího debuggeru (GNU)
- plná podpora systémových operací

## Omezení

- emulace pouze CPU (ne celého SoC)
- limitovaná podpora připojení externích periferií
- `_start` symbol očekáván na adrese 0x00010000 (ARM)
  - ⇒ nekompatibilita s *first-stage BL Rpi0* (0x00008000)
- problémy se `systimer`

## ARM1176JZF-S

- ARMv6 instrukce
- přepínání režimů CPU
- ALU, MAC a MMU (+ TLB)
- vyjímky a přerušení
- podpora ko-procesorů
  - CP15, CP10
- systémová sběrnice

## BCM2835

- RAM
- interrupt controller
- ARM timer
- TRNG
- GPIO
- BSC\_1 ( $I^2C$ )
- AUX (Mini\_UART)
- debug monitor\*

- 
- cílem bylo emulovat nejčastěji používané periferie
  - dekompoziční návrh architektury
    - ⇒ snadné rozšíření o další periferie

- jednotné rozhraní pro externí periferie
- načtené při inicializaci jako sdílené knihovny
  - možnost načtení vícero instancí (např. led.dll)
- nezávislé na toolchainu jádra emulátoru
  - `extern "C" __declspec(dllexport)`

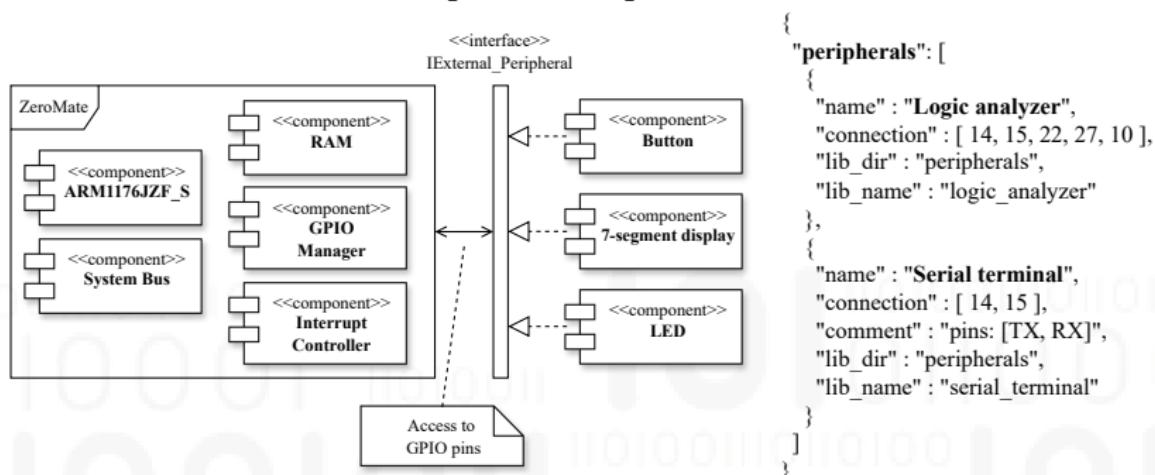
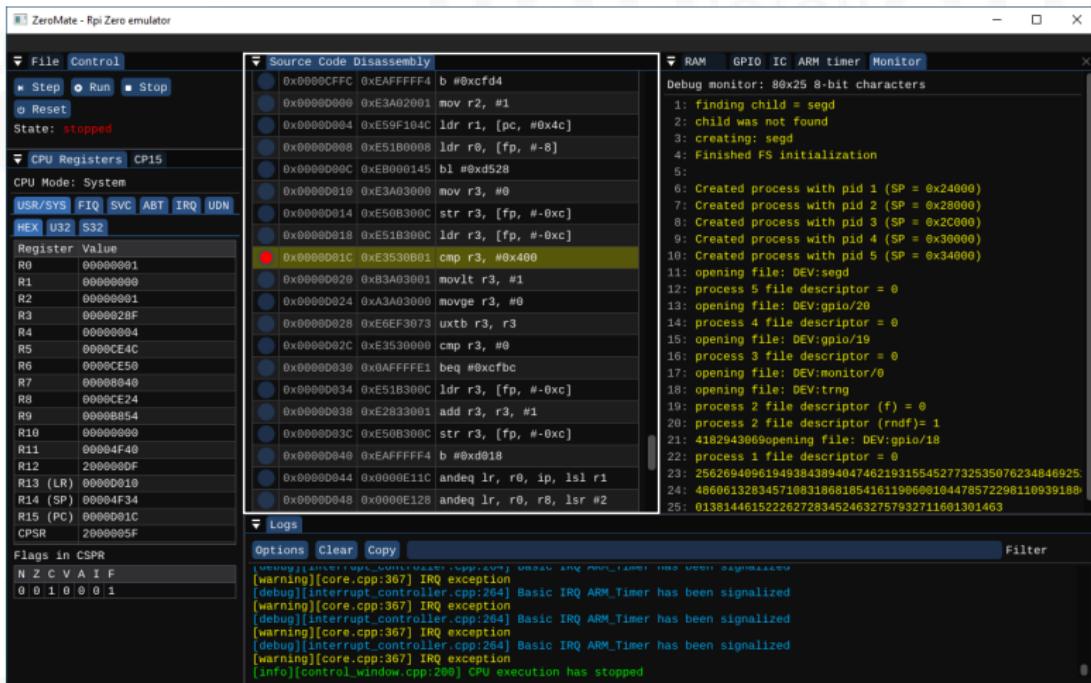
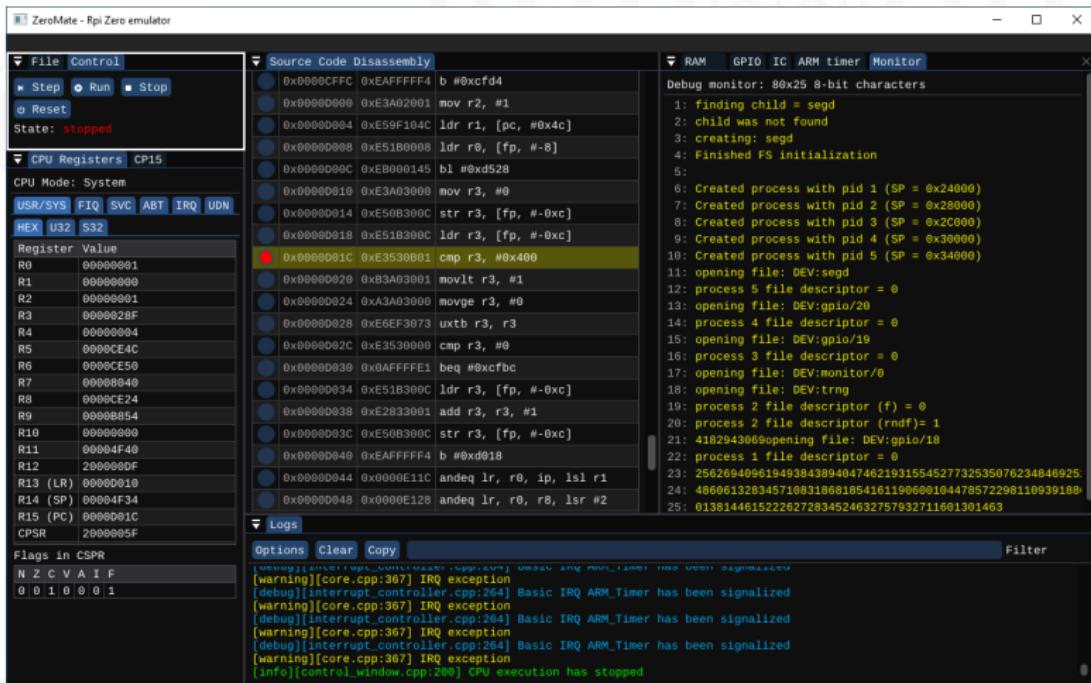


Figure 6: Rozhraní externích periferií

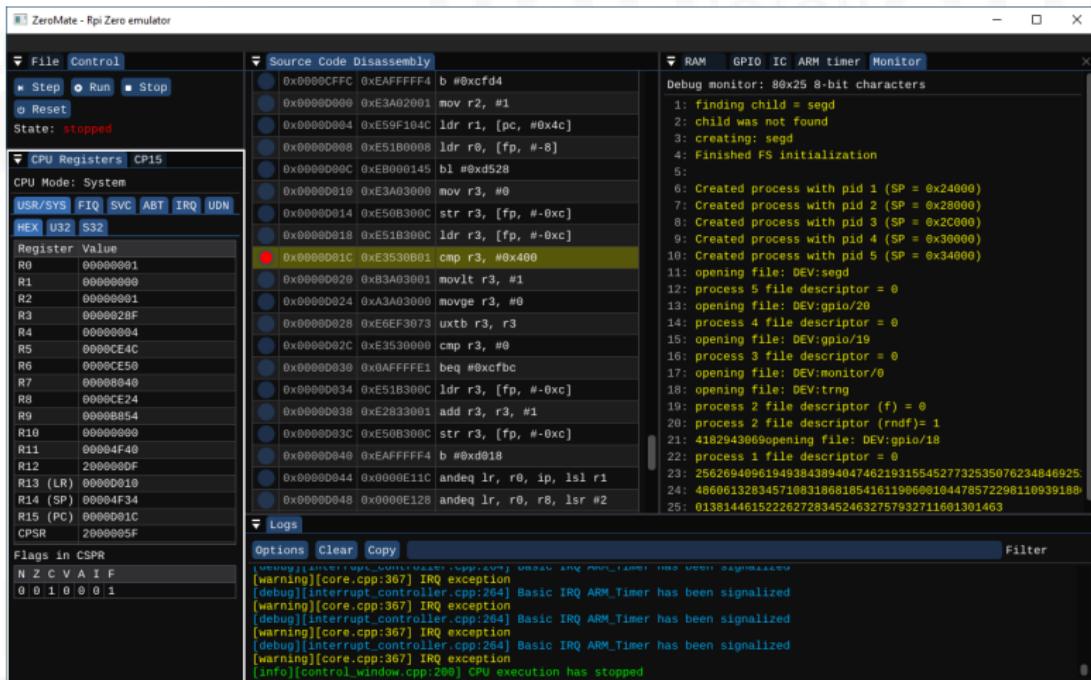
## ZeroMate - uživatelské rozhraní



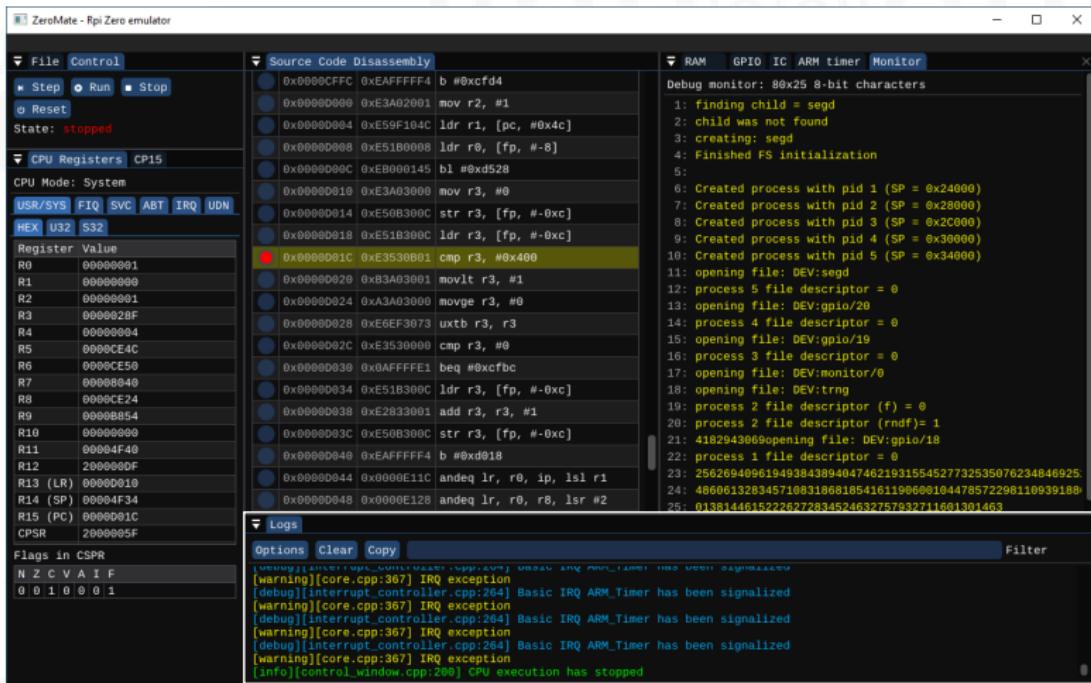
## ZeroMate - uživatelské rozhraní



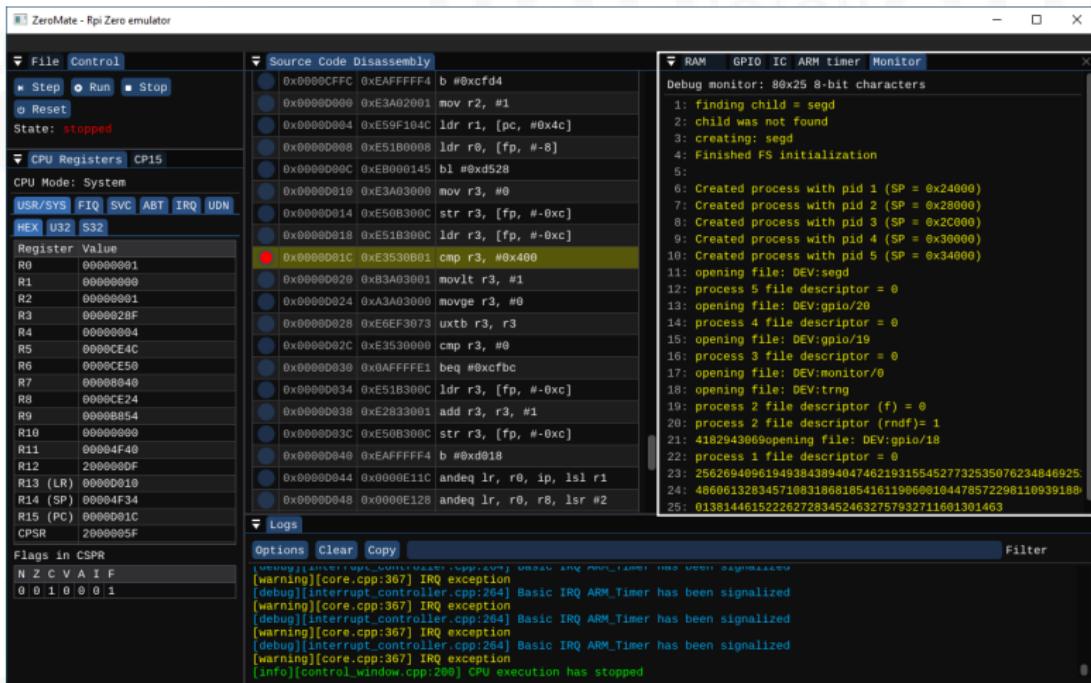
## ZeroMate - uživatelské rozhraní



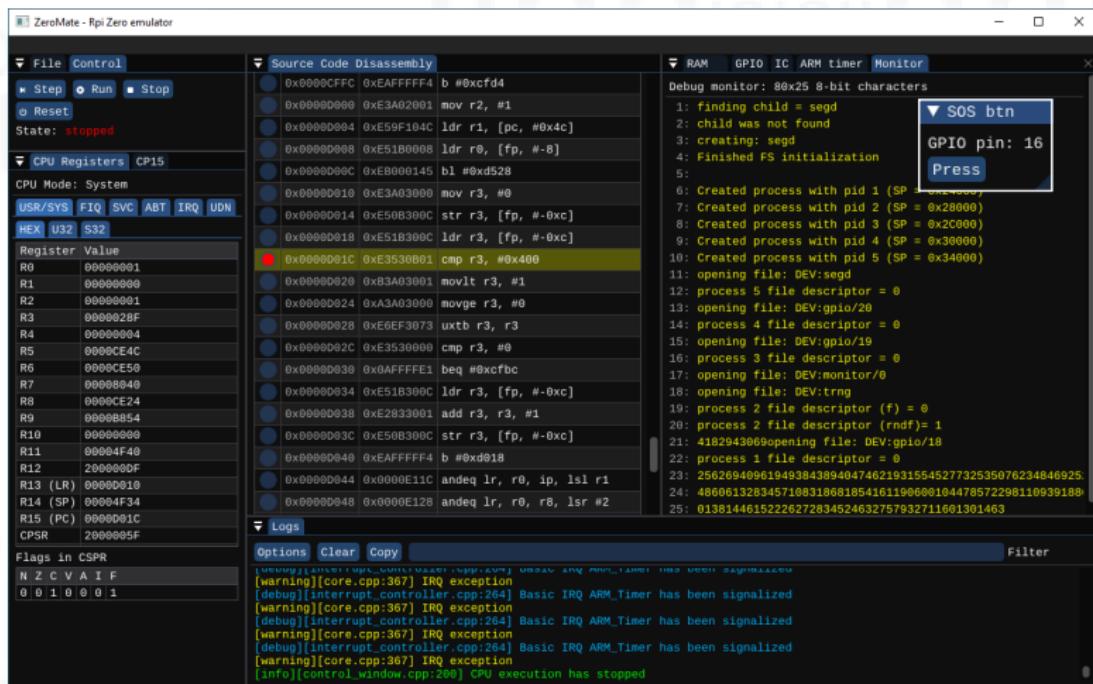
## ZeroMate - uživatelské rozhraní



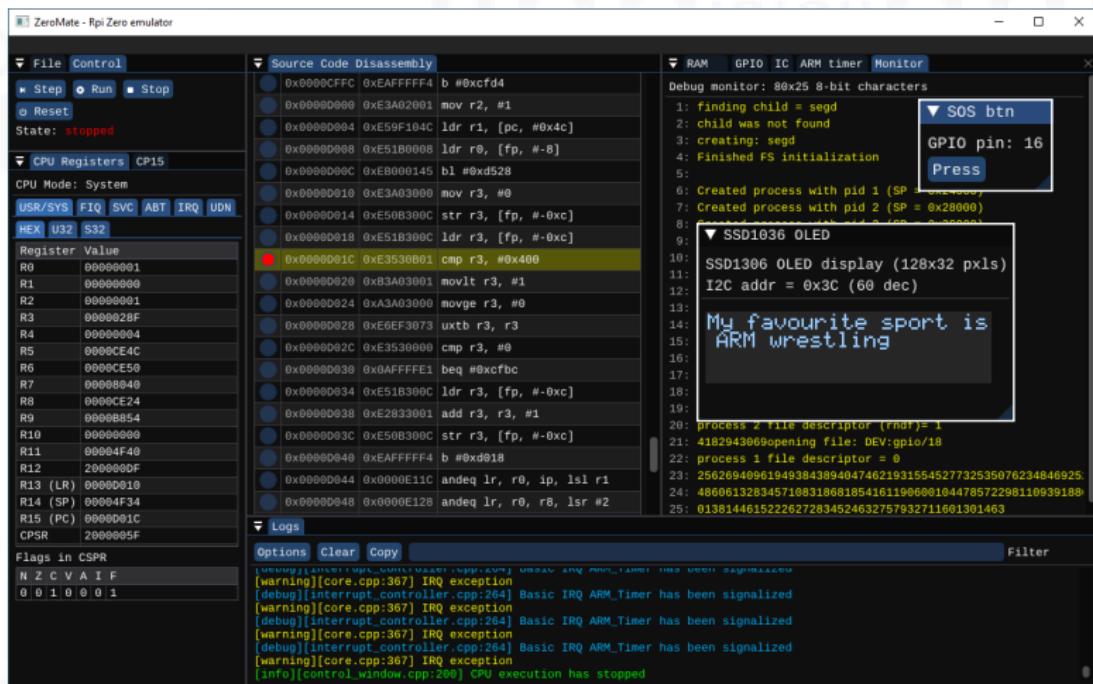
## ZeroMate - uživatelské rozhraní



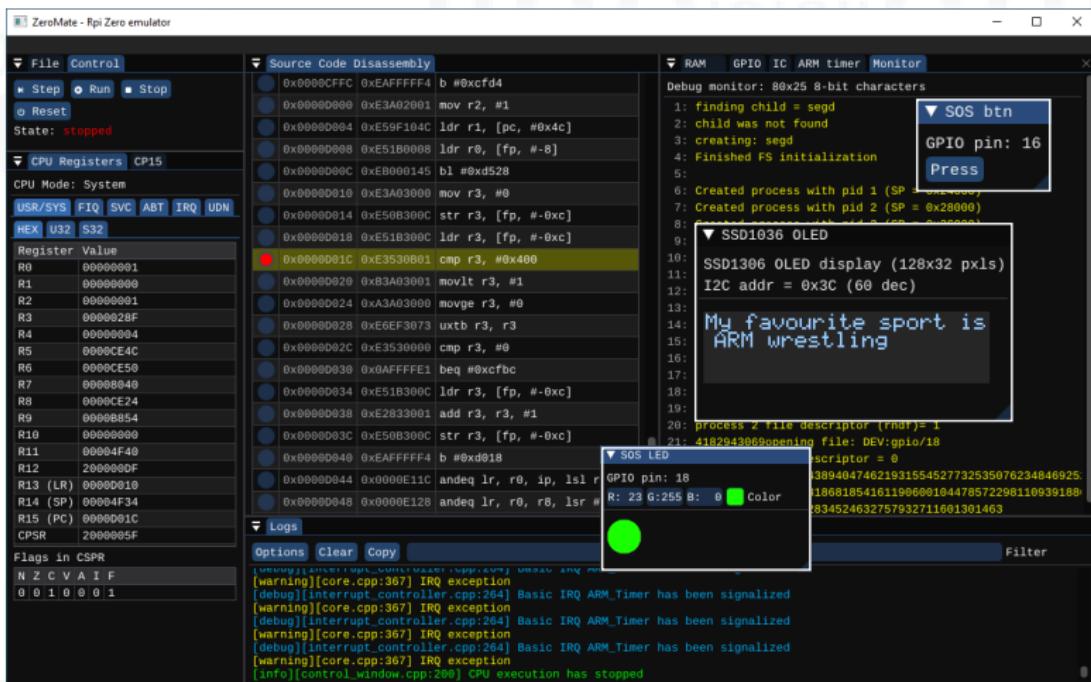
## ZeroMate - externí periferie



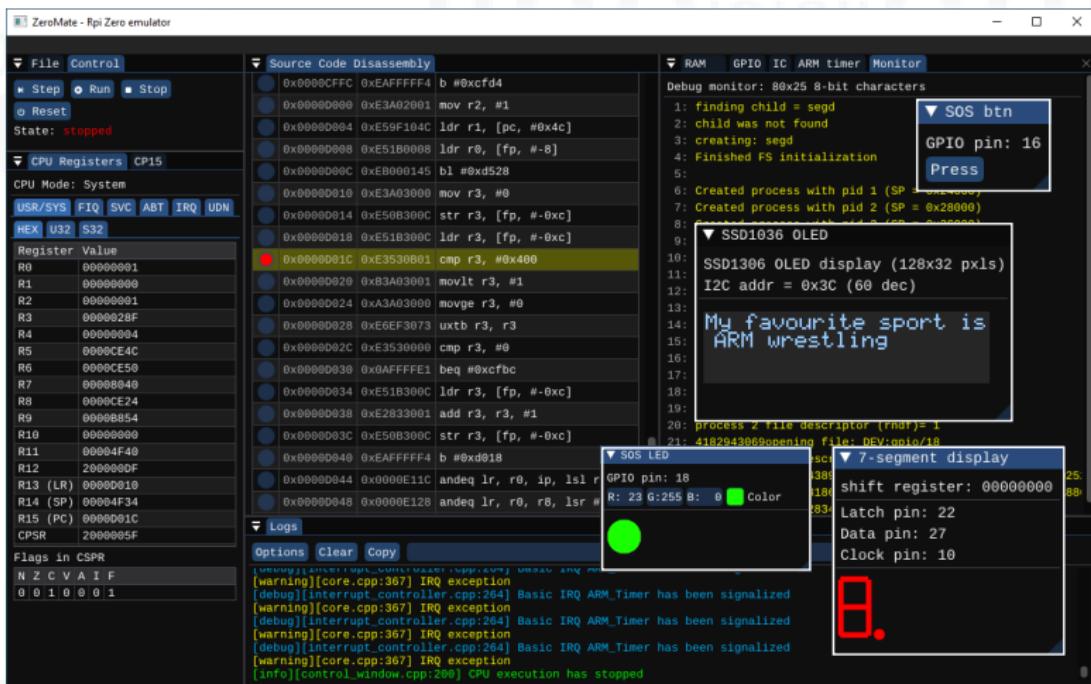
## ZeroMate - externí periferie



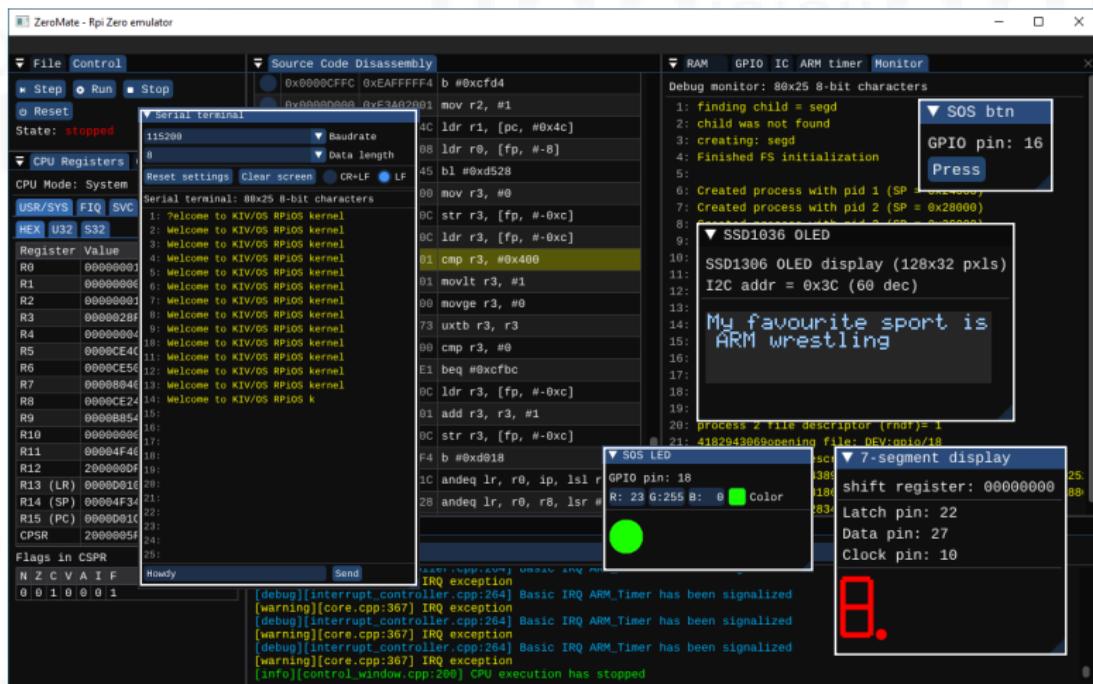
## ZeroMate - externí periferie



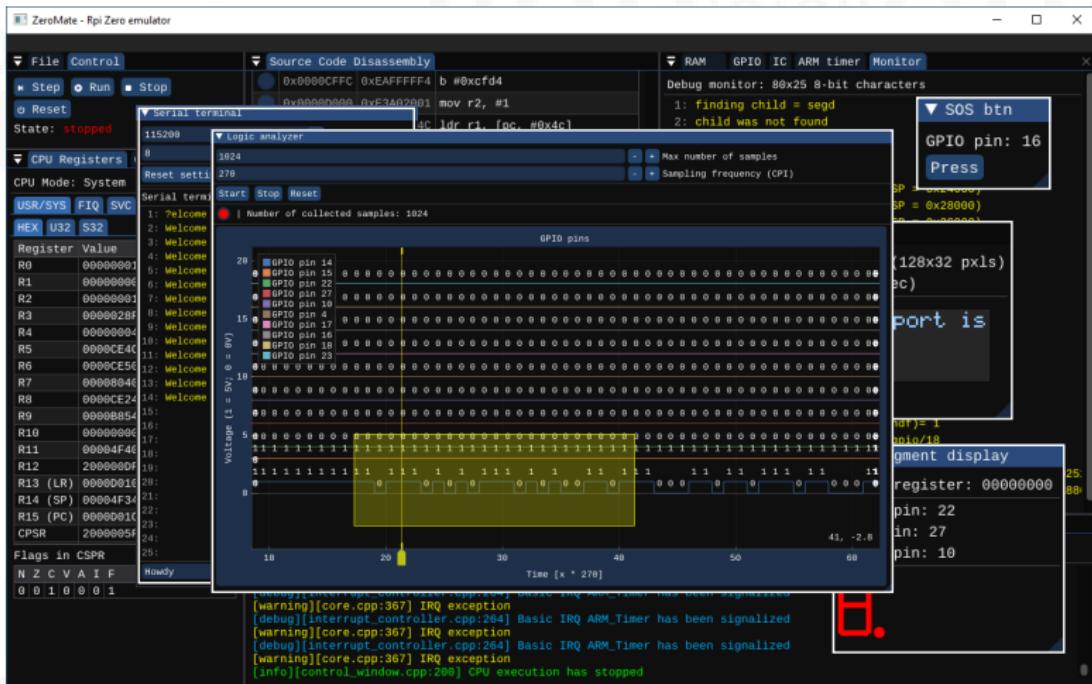
## ZeroMate - externí periferie



## ZeroMate - externí periferie



ZeroMate - externí periferie



## Vývoj a použité technologie



► <https://github.com/silhavyj/ZeroMate>



- knihovny třetích stran automaticky staženy a sestaveny (.gitmodules + CMake)

- [GoogleTest](#) [ImGui](#) [dylib](#) [ELFIO](#) [GLFW](#) [capstone](#)

## 1) Unit testy

- jádro emulátoru (exekuce ARMv6 instrukcí)
- GoogleTest, CI (regresní testy), **pokrytí  $\approx 78\%$**

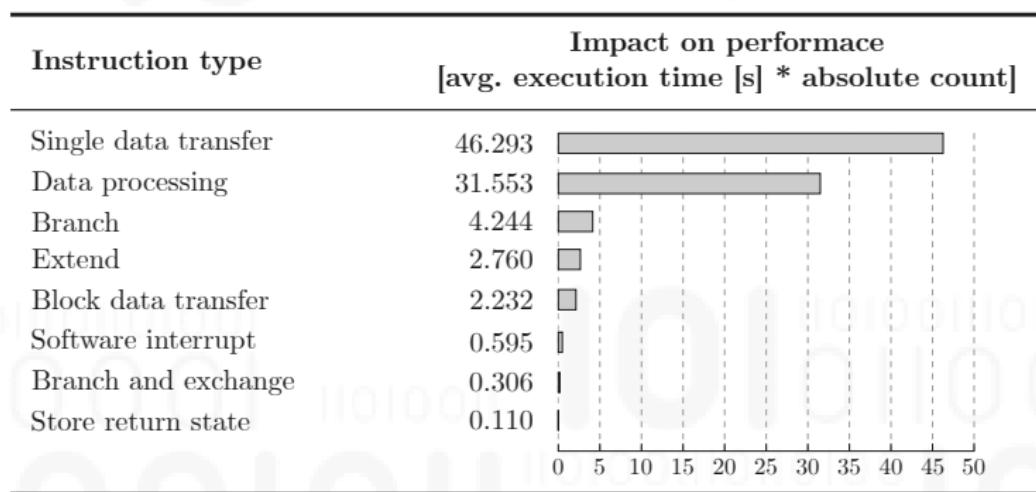
## 2) Funkční testy

- BCM2835 periferie
- připravená sada příkladů (.ELF)
  - UART, FPU, GPIO, I<sup>2</sup>C, plánování procesů (EDF, RTL), ARM timer, ...
  - součástí repozitáře jako examples

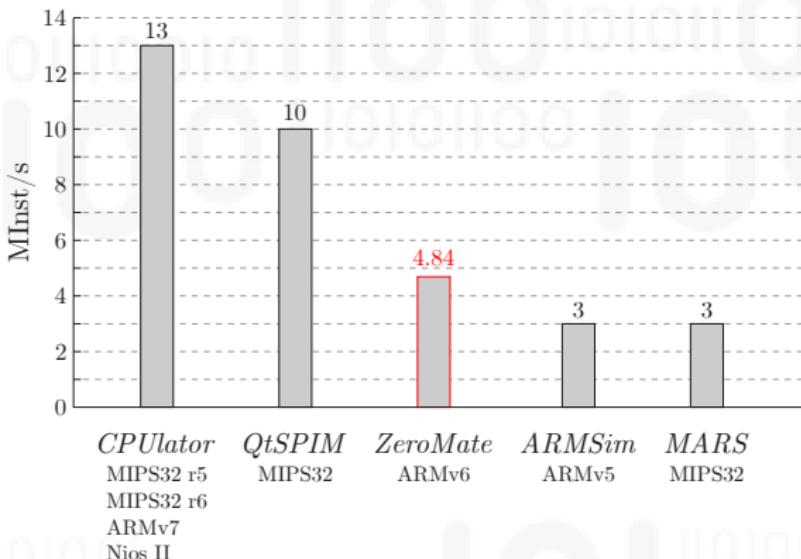
## 3) Aplikační testy, GUI

- KIV/OS cvičení
- dotazník hodnocení celkové použitelnosti

- **výkon emulátoru  $\approx 4\text{-}5 \text{ Minst/s}$**  ([► KIV-RTOS](#))
- analýza rychlosti emulace jednotlivých instrukcí
  - $\Rightarrow$  potenciální optimalizace



## Porovnání výkonnosti s ostatními emulátory



- pouze orientační (<https://cpulator.01xz.net/>, 2023)
- ZeroMate jako jediný podporuje MMU
  - využití s každým přístupem do paměti ( $\approx 42\%$  instrukcí)

## IPC

- např. pomocí socketů
- spuštění vícero instancí ZeroMate emulátoru
  - ⇒ emulace distribuovaných systémů & algoritmů

## CLI mód

- integrace do CI/CD pipeline
- validace programů v rámci výuky
  - např. definováním očekáváné UART komunikace (I/O)

---

Další viz text diplomové práce...

Návrhy na budoucí vylepšení

Děkuji Vám za pozornost

▶ <https://github.com/silhavyj/ZeroMate>



Otázka

Odpověď

Otázka

Odpověď