

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

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

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

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

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

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

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



Figure 2: ARM aplikace

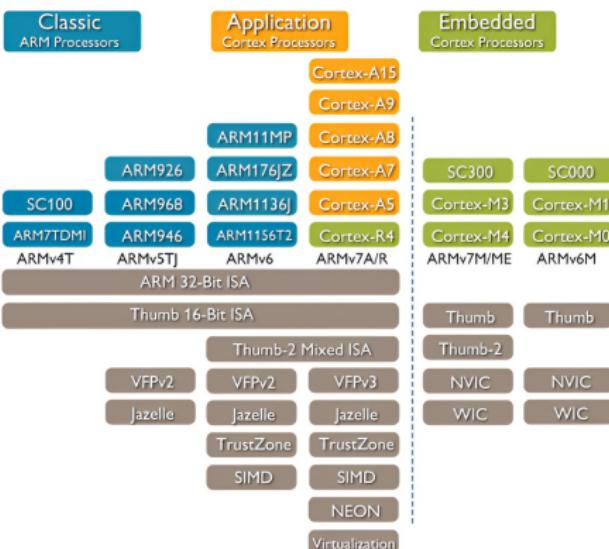


Figure 3: ARM CPU roadmap

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

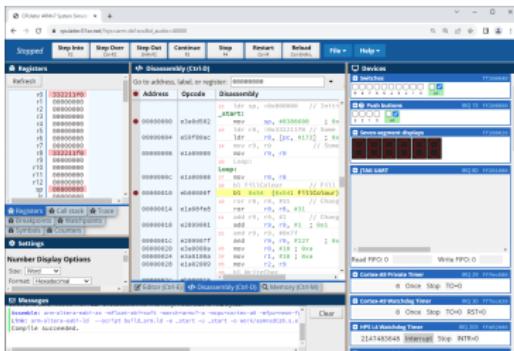


Figure 4: ▶ CPUlator

Omezení

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

- podpora různých architektur
 - x86, MIPS, ARM, ...
- připojení externího debuggeru
- plná podpora systémových oprací



Figure 5: ▶ QEMU

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
- 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 (připojeny přes GPIO)
- načtené při inicializaci jako sdílené knihovny (.dll, .so)
 - možnost načtení vícero instancí (např. led.dll)
- nezávislé na toolchainu jádra emulátoru

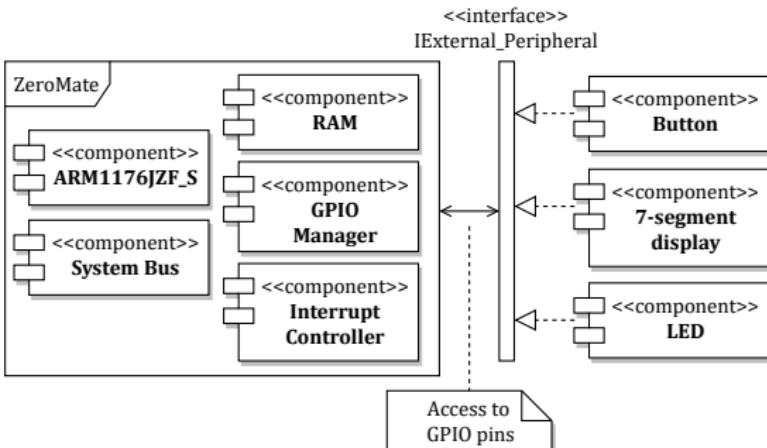


Figure 6: Rozhraní externích periferií

ZeroMate - uživatelské rozhraní

Source Code Disassembly

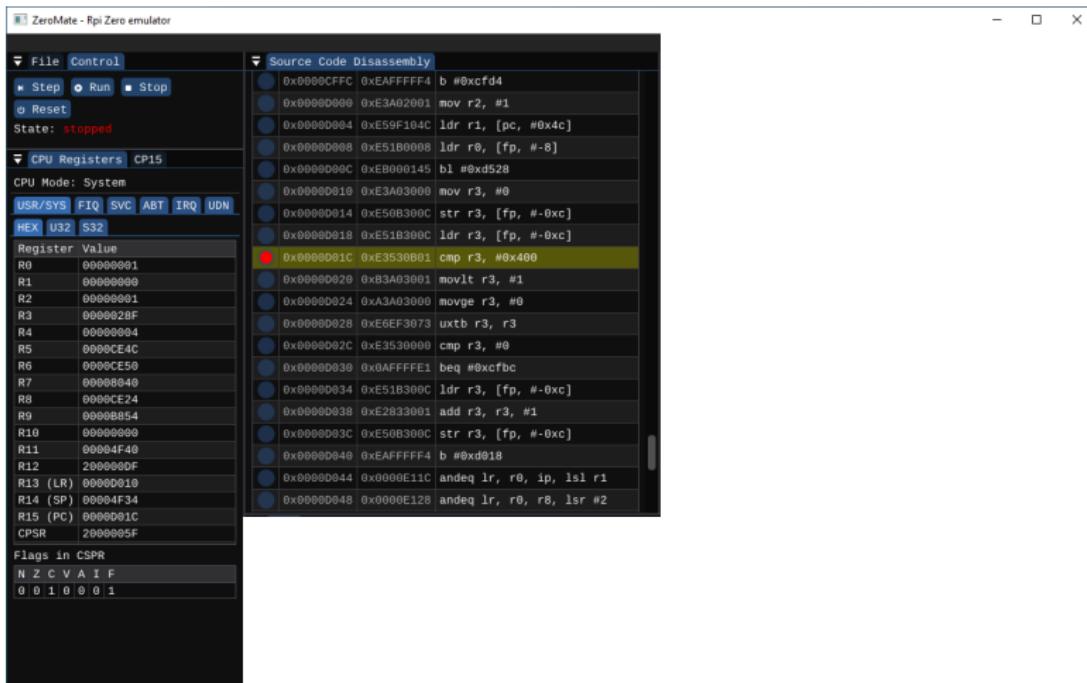
Address	OpCode	Assembly
0x0000000000000000	b	#0xfcfd4
0x0000000000000001	mov	r2, #1
0x0000000000000002	ldr	r1, [pc, #0x4c]
0x0000000000000003	ldr	r0, [fp, #-8]
0x0000000000000004	bl	#0xd528
0x0000000000000005	mov	r3, #0
0x0000000000000006	str	r3, [fp, #-0xc]
0x0000000000000007	ldr	r3, [fp, #-0xc]
0x0000000000000008	cmp	r3, #0x400
0x0000000000000009	movlt	r3, #1
0x000000000000000A	movge	r3, #0
0x000000000000000B	uxtb	r3, r3
0x000000000000000C	cmp	r3, #0
0x000000000000000D	beq	#0xfcfc
0x000000000000000E	ldr	r3, [fp, #-0xc]
0x000000000000000F	add	r3, r3, #1
0x0000000000000010	str	r3, [fp, #-0xc]
0x0000000000000011	b	#0xd018
0x0000000000000012	andeq	lr, r0, ip, lsl r1
0x0000000000000013	andeq	lr, r0, r0, lsr #2

ZeroMate - uživatelské rozhraní

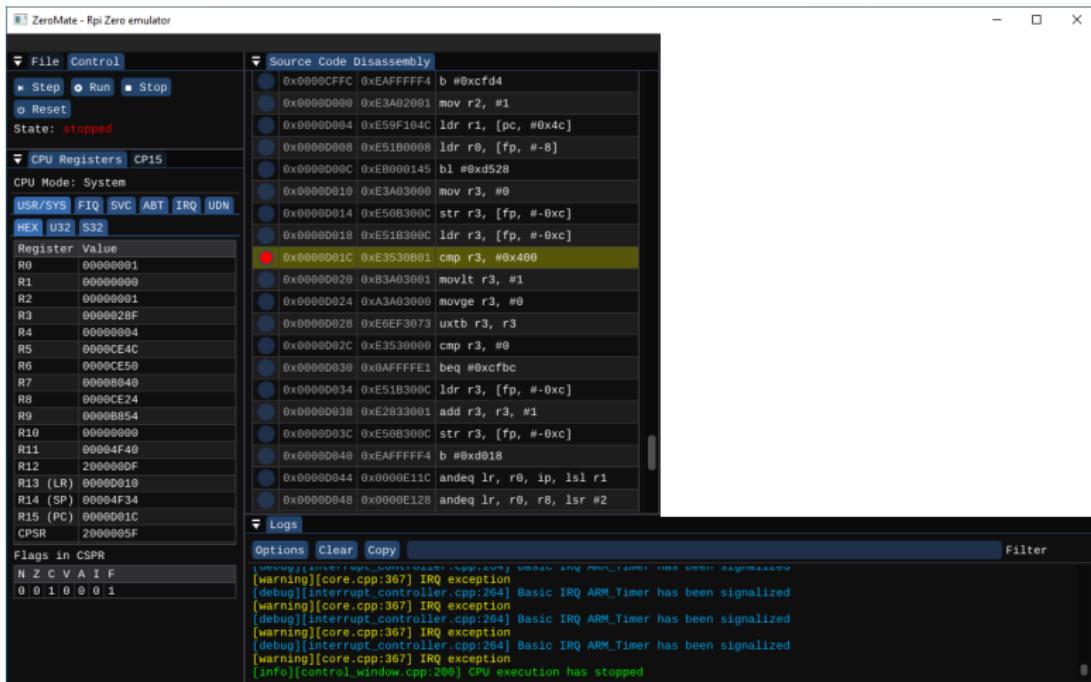
The screenshot shows the ZeroMate - Rpi Zero emulator interface. On the left, there is a control panel with buttons for Step, Run, Stop, and Reset, and a status indicator "State: stopped". On the right, the main window displays the "Source Code Disassembly" tab. The assembly code is listed in two columns: address and assembly instruction. One specific instruction at address 0x000000001C is highlighted with a red circle, indicating it is the current instruction being executed or selected.

Address	Assembly Instruction
0x00000CFFC	b #0xfcfd4
0x0000000001	0xE3A02001 mov r2, #1
0x0000000004	0xE59F104C ldr r1, [pc, #0x4c]
0x0000000008	0xE51B0008 ldr r0, [fp, #-8]
0x000000000C	0xEB000145 bl #0xd528
0x0000000010	0xE3A03000 mov r3, #0
0x0000000014	0xE50B300C str r3, [fp, #-0xc]
0x0000000018	0xE51B300C ldr r3, [fp, #-0xc]
0x000000001C	0xE3530B01 cmp r3, #0x400
0x0000000020	0xB3A03001 movlt r3, #1
0x0000000024	0xA3A03000 movge r3, #0
0x0000000028	0xE6EF3073 uxtb r3, r3
0x000000002C	0xE3530000 cmp r3, #0
0x0000000030	0x0AFFFE11 beq #0xfcfc
0x0000000034	0xE51B300C ldr r3, [fp, #-0xc]
0x0000000038	0xE2833001 add r3, r3, #1
0x000000003C	0xE50B300C str r3, [fp, #-0xc]
0x0000000040	0xEAFFFFF4 b #0xd018
0x0000000044	0x0000E11C andeq lr, r0, ip, lsl r1
0x0000000048	0x0000E128 andeq lr, r0, r0, lsr #2

ZeroMate - uživatelské rozhraní



ZeroMate - uživatelské rozhraní



ZeroMate - uživatelské rozhraní

ZeroMate - Rpi Zero emulator

File Control

- Step
- Run
- Stop
- Reset

State: stopped

CPU Registers CP15

CPU Mode: System

USR/SYS	FIQ	SVC	ABT	IRQ	UDN
HEX	U32	S32			
Register Value					
R0	00000001				
R1	00000000				
R2	00000001				
R3	000002BF				
R4	00000004				
R5	0000CE4C				
R6	0000CE50				
R7	00008B40				
R8	0000CE24				
R9	0000B854				
R10	00000000				
R11	00004F40				
R12	2000000F				
R13 (LR)	00000010				
R14 (SP)	00004F34				
R15 (PC)	0000001C				
CPSR	2000005F				
Flags in CPSR					
N	Z	C	V	A	I F
0	1	0	0	1	1

Source Code Disassembly

```

0x00000000 0xEFFFFF4 b #0xfcfd4
0x00000001 0xE3A02001 mov r2, #1
0x00000002 0xE59F104C ldr r1, [pc, #0x4c]
0x00000003 0xE51B0008 ldr r0, [fp, #-8]
0x00000004 0xEB000145 bl #0xd528
0x00000005 0xE3A03000 mov r3, #0
0x00000006 0xE50B300C str r3, [fp, #-0xc]
0x00000007 0xE51B300C ldr r3, [fp, #-0xc]
0x00000008 0xE3530001 cmp r3, #0x400
0x00000009 0xB3A03001 movlt r3, #1
0x0000000A 0xA3A03000 movge r3, #0
0x0000000B 0xE6EF3073 uxtb r3, r3
0x0000000C 0xE3530000 cmp r3, #0
0x0000000D 0x0AFFFE11 beq #0xcfcfc
0x0000000E 0xE51B300C ldr r3, [fp, #-0xc]
0x0000000F 0xE2833001 add r3, r3, #1
0x00000010 0xE50B300C str r3, [fp, #-0xc]
0x00000011 0xE5000040 0xEAFFFFF4 b #0xd018
0x00000012 0x0000E11C andeq lr, r0, ip, lsl r1
0x00000013 0xE5000048 0x0000E128 andeq lr, r0, r8, lsr #2

```

RAM GPIO IC ARM timer Monitor

Debug monitor: 0x25 8-bit characters

```

1: finding child = segd
2: child was not found
3: creating: segd
4: Finished FS initialization
5:
6: Created process with pid 1 (SP = 0x24000)
7: Created process with pid 2 (SP = 0x28000)
8: Created process with pid 3 (SP = 0x2C000)
9: Created process with pid 4 (SP = 0x30000)
10: Created process with pid 5 (SP = 0x34000)
11: opening file: DEV:segd
12: process 5 file descriptor = 0
13: opening file: DEV:gpio/20
14: process 4 file descriptor = 0
15: opening file: DEV:gpio/19
16: process 3 file descriptor = 0
17: opening file: DEV:monitor/0
18: opening file: DEV:trng
19: process 2 file descriptor (f) = 0
20: process 2 file descriptor (rndf)= 1
21: 4182943060Opening file: DEV:gpio/18
22: process 1 file descriptor = 0
23: 2562694096194938438940474621931554527732535076234846925
24: 466061328345710831868185416119060104478572298110939188
25: 0136144615222672834524632757932711601301463

```

Logs

Options Clear Copy Filter

```

[warning][core.cpp:367] IRQ exception
[debug][interrupt_controller.cpp:264] Basic IRQ ARM_Timer has been signaled
[warning][core.cpp:367] IRQ exception
[debug][interrupt_controller.cpp:264] Basic IRQ ARM_Timer has been signaled
[warning][core.cpp:367] IRQ exception
[debug][interrupt_controller.cpp:264] Basic IRQ ARM_Timer has been signaled
[warning][core.cpp:367] IRQ exception
[info][control_window.cpp:200] CPU execution has stopped

```

ZeroMate - externí periferie

ZeroMate - Rpi Zero emulator

File Control

- Step
- Run
- Stop

Reset

CPU Registers CP15

CPU Mode: System

USR/SYS	FIQ	SVC	ABT	IRQ	UDN
HEX	V32	S32			
Register Value					
R0	00000001				
R1	00000000				
R2	00000001				
R3	000002BF				
R4	00000004				
R5	0000CE4C				
R6	0000CE50				
R7	00008B40				
R8	0000CE24				
R9	0000B854				
R10	00000000				
R11	00004F40				
R12	2000000F				
R13 (LR)	00000010				
R14 (SP)	00004F34				
R15 (PC)	0000001C				
CPSR	2000005F				
Flags in CPSR					
N	Z	C	V	A	I F
0	1	0	0	1	1

Source Code Disassembly

```

0x000000FC 0xEFFFFF4 b #0xfcfd4
0x00000009 0xE3A02001 mov r2, #1
0x00000004 0xE59F104C ldr r1, [pc, #0x4c]
0x00000008 0xE51B0008 ldr r0, [fp, #-8]
0x00000014 0xEB000145 bl #0xd528
0x00000010 0xE3A03000 mov r3, #0
0x00000014 0xE50B300C str r3, [fp, #-0xc]
0x00000018 0xE51B300C ldr r3, [fp, #-0xc]
0x0000000C 0xE3530001 cmp r3, #0x400
0x00000020 0xB3A03001 movlt r3, #1
0x00000024 0xA3A03000 movge r3, #0
0x00000028 0xE6EF3073 uxtb r3, r3
0x0000002C 0xE3530000 cmp r3, #0
0x00000030 0x0AFFFE11 beq #0xcfcfc
0x00000034 0xE51B3001 ldr r3, [fp, #-0xc]
0x00000038 0xE2833001 add r3, r3, #1
0x0000003C 0xE50B300C str r3, [fp, #-0xc]
0x00000040 0xEAFFFFF4 b #0xd018
0x00000044 0x0000E11C andeq lr, r0, ip, lsl r1
0x00000048 0x0000E128 andeq lr, r0, r0, lsr #2

```

RAM **GPIO** **IC** **ARM timer** **Monitor**

Debug monitor: 0x25 8-bit characters

- 1: finding child = segd
- 2: child was not found
- 3: creating: segd
- 4: Finished FS initialization
- 5:
- 6: Created process with pid 1 (SP = 0x20000)
- 7: Created process with pid 2 (SP = 0x28000)
- 8: Created process with pid 3 (SP = 0x2C000)
- 9: Created process with pid 4 (SP = 0x30000)
- 10: Created process with pid 5 (SP = 0x34000)
- 11: opening file: DEV:segd
- 12: process 5 file descriptor = 0
- 13: opening file: DEV:gpio/20
- 14: process 4 file descriptor = 0
- 15: opening file: DEV:gpio/19
- 16: process 3 file descriptor = 0
- 17: opening file: DEV:monitor/0
- 18: opening file: DEV:ttrng
- 19: process 2 file descriptor (f) = 0
- 20: process 2 file descriptor (rndf)= 1
- 21: 4182943060Opening file: DEV:gpio/18
- 22: process 1 file descriptor = 0
- 23: 2562694096194938438940474621931554527732535076234846925
- 24: 466061328345710831868185416119060104478572298110939188
- 25: 01361446152226272834524632757932711601301463

SOS btn

GPIO pin: 16

Press

Logs

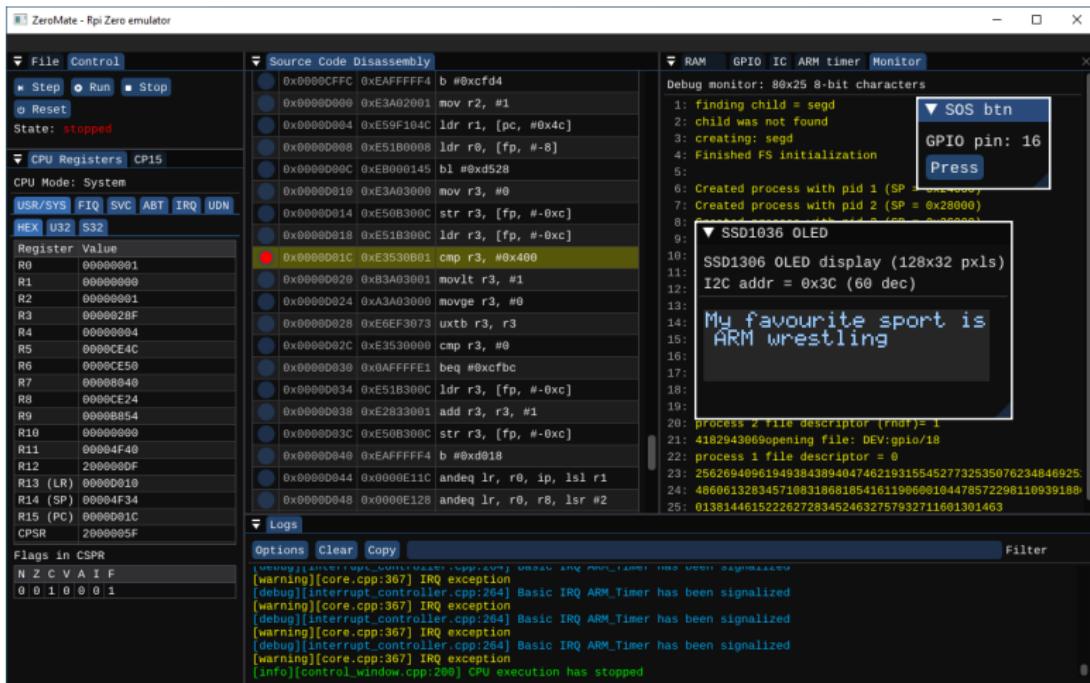
Options Clear Copy Filter

```

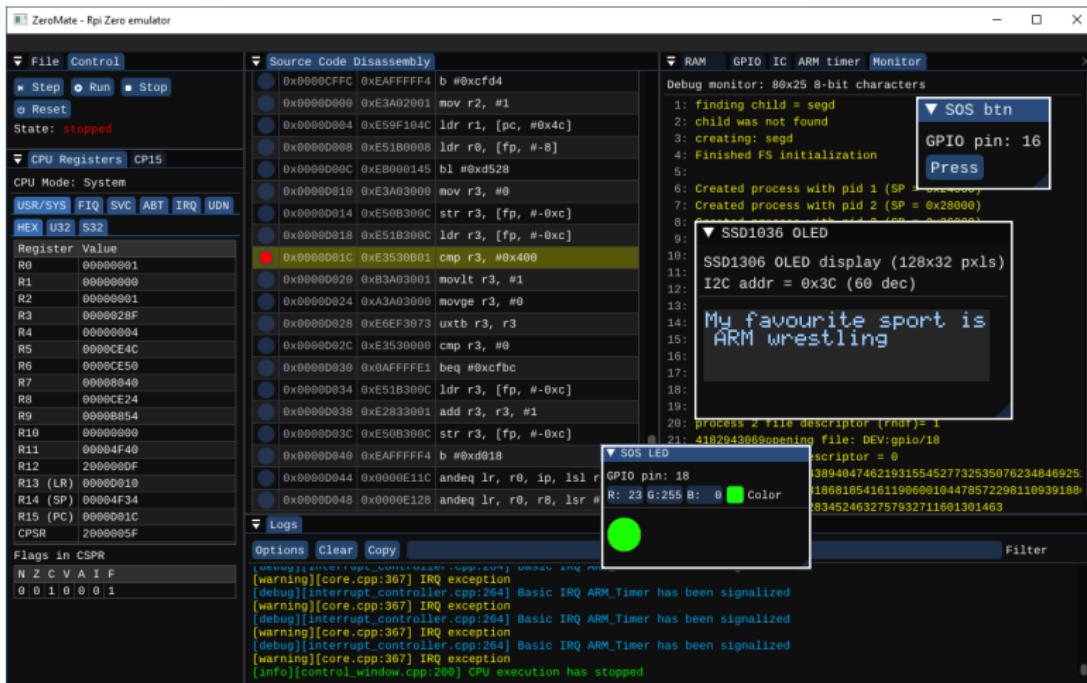
[warning][core.cpp:367] IRQ exception
[debug][interrupt_controller.cpp:264] Basic IRQ ARM_Timer has been signaled
[warning][core.cpp:367] IRQ exception
[debug][interrupt_controller.cpp:264] Basic IRQ ARM_Timer has been signaled
[warning][core.cpp:367] IRQ exception
[debug][interrupt_controller.cpp:264] Basic IRQ ARM_Timer has been signaled
[warning][core.cpp:367] IRQ exception
[info][control_window.cpp:208] CPU execution has stopped

```

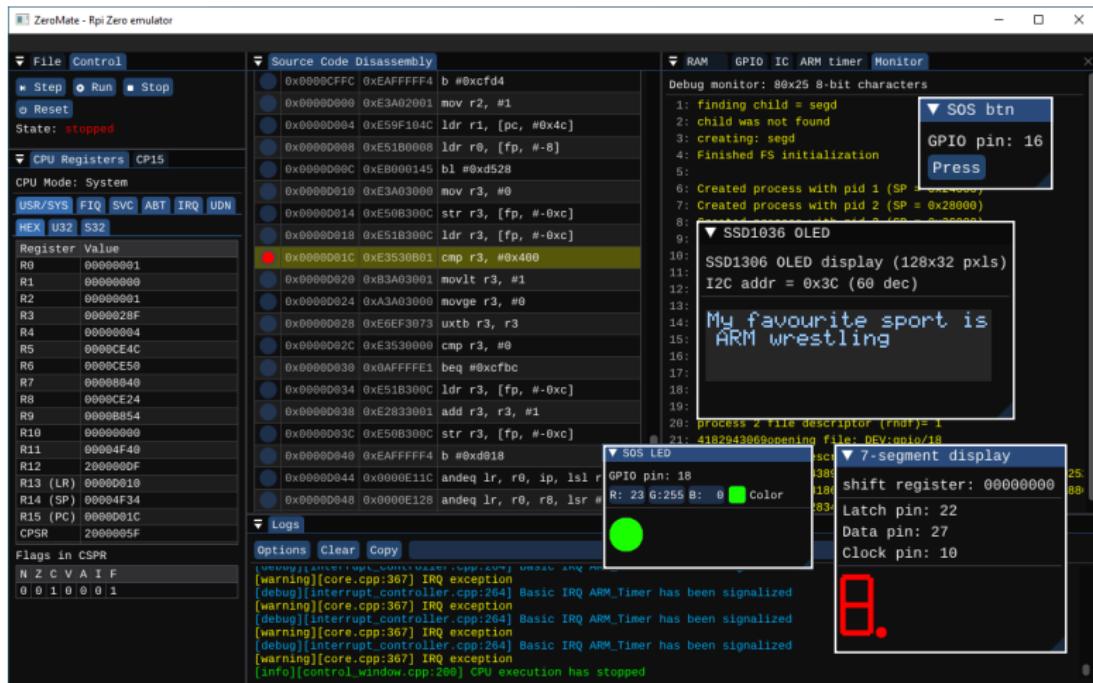
ZeroMate - externí periferie



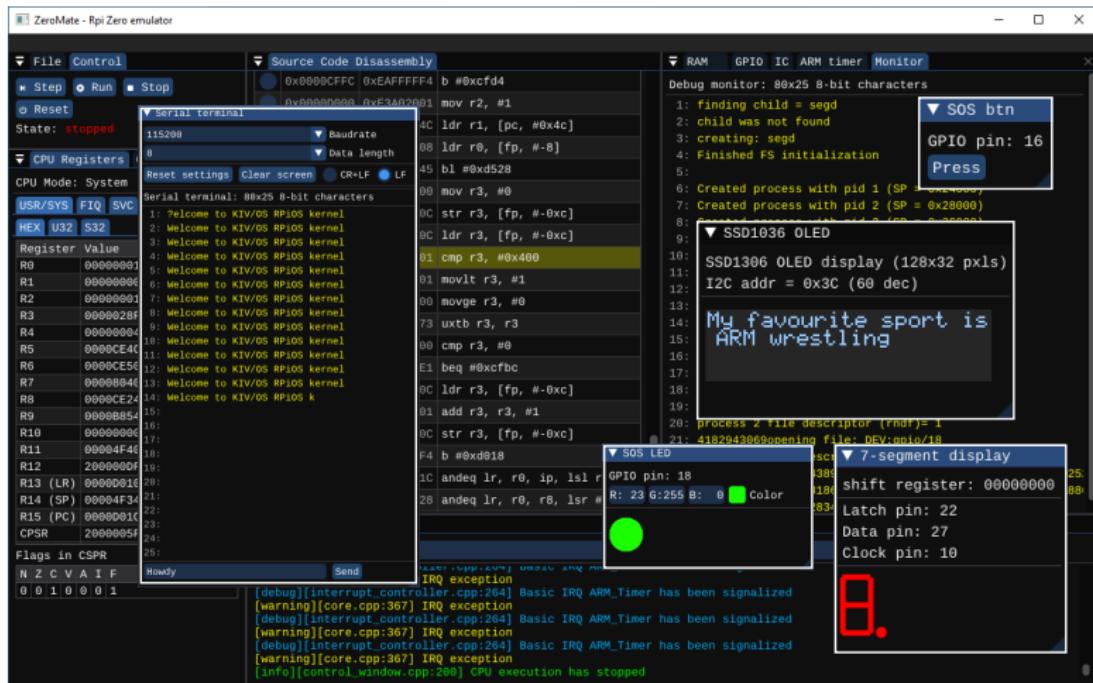
ZeroMate - externí periferie



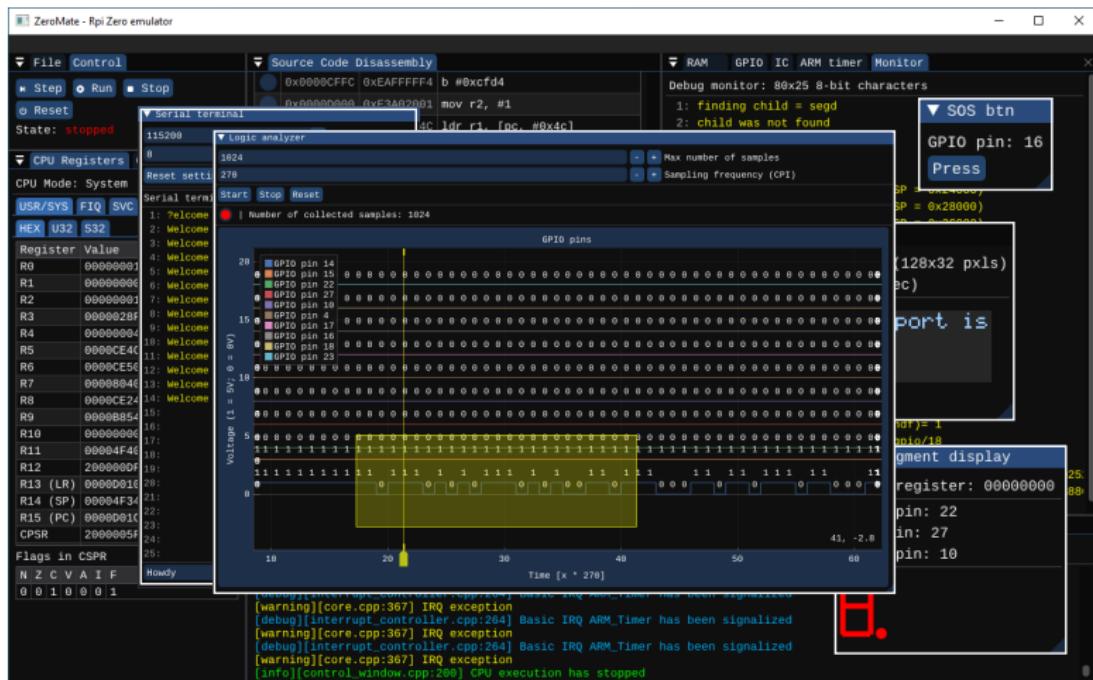
ZeroMate - externí periferie



ZeroMate - externí periferie



ZeroMate - externí periferie





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



- OpenGL jako jediná závislost
 - zbytek automaticky stažen a sestaven (.gitmodules + CMake)

Děkuji Vám za pozornost

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

