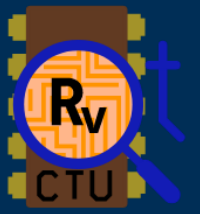
CTU
CZECH TECHNICAL UNIVERSITY IN PRAGUE

QtRvSim

RISC-V Simulator for Computer Architectures Classes

Czech Technical University in Prague

Jakub Dupák *dev@jakubdupak.com*
Pavel Píša *pisa@fel.cvut.cz*
Karel Kočí *cynerd@email.cz*
Michal Štepanovský *michal.stepanovsky@gmail.com*



comparch.edu.cvut.cz

RISC-V

RISC-V

Western Digital WD

Links to the relevant articles are available at the end on the presentation.

RISC-V

Western Digital WD

Alibaba Group 阿里巴巴集团

RISC-V®

Links to the relevant articles are available at the end on the presentation.

RISC-V

Links to the relevant articles are available at the end on the presentation.

RISC-V

Links to the relevant articles are available at the end on the presentation.

RISC-V

Links to the relevant articles are available at the end on the presentation.

RISC-V

Links to the relevant articles are available at the end on the presentation.

RISC-V

Links to the relevant articles are available at the end on the presentation.

What is RISC-V anyway?

- Instruction Set Architecture (ISA)
  - like ARM and x86
- Open standard managed by a non-profit org.
- Royalty-free & Open-Source
- Small and simple core
- Extensibility & Customization

RISC-V for education?

RISC-V for education?

comparch.edu.cvut.cz

WebAssembly Edition

comparch.edu.cvut.cz/qtrvsim/app

Presentation Outline

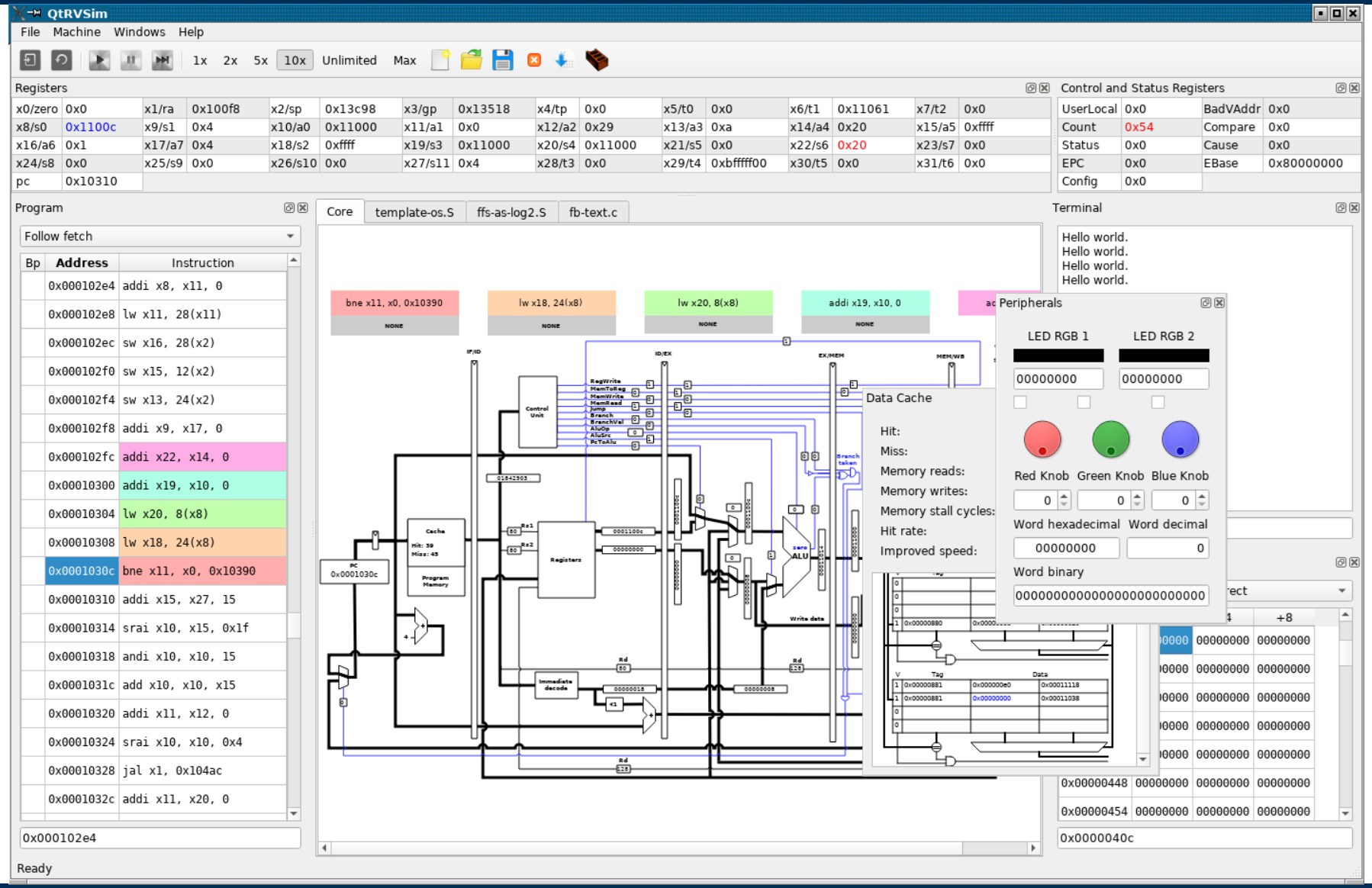
- Simulator Features
- Implementation Overview
- Available Materials and Releases

WebAssembly

Simulator Features

How does it look and what can it do…

Layout


QtRVSim
File Machine Windows Help
1x 2x 5x 10x Unlimited Max
Registers
x0/zero 0x0 x1/ra 0x100f8 x2/sp 0x13c98 x3/gp 0x13518 x4/tp 0x0 x5/t0 0x0 x6/t1 0x11061 x7/t2 0x0
x8/s0 0x1100c x9/s1 0x4 x10/a0 0x11000 x11/a1 0x0 x12/a2 0x29 x13/a3 0xa x14/a4 0x20 x15/a5 0xffff
x16/a6 0x1 x17/a7 0x4 x18/s2 0xffff x19/s3 0x11000 x20/s4 0x11000 x21/s5 0x0 x22/s6 0x20 x23/s7 0x0
x24/s8 0x0 x25/s9 0x0 x26/s10 0x0 x27/s11 0x4 x28/t3 0x0 x29/t4 0xbfffff00 x30/t5 0x0 x31/t6 0x0
pc 0x10310
Control and Status Registers
UserLocal 0x0 BadVAddr 0x0
Count 0x54 Compare 0x0
Status 0x0 Cause 0x0
EPC 0x0 EBase 0x80000000
Config 0x0
Program
Follow fetch
Bp Address Instruction
0x000102e4 addi x8, x11, 0
0x000102e8 lw x11, 28(x11)
0x000102ec sw x16, 28(x2)
0x000102f0 sw x15, 12(x2)
0x000102f4 sw x13, 24(x2)
0x000102f8 addi x9, x17, 0
0x000102fc addi x22, x14, 0
0x00010300 addi x19, x10, 0
0x00010304 lw x20, 8(x8)
0x00010308 lw x18, 24(x8)
0x0001030c bne x11, x0, 0x10390
0x00010310 addi x15, x27, 15
0x00010314 srai x10, x15, 0x1f
0x00010318 andi x10, x10, 15
0x0001031c add x10, x10, x15
0x00010320 addi x11, x12, 0
0x00010324 srai x10, x10, 0x4
0x00010328 jal x1, 0x104ac
0x0001032c addi x11, x20, 0
0x000102e4
Core
template-os.S
ffs-as-log2.S
fb-text.c
bne x11, x0, 0x10390
lw x18, 24(x8)
lw x20, 8(x8)
addi x19, x10, 0
Data Cache
Hit:
Miss:
Memory reads:
Memory writes:
Memory stall cycles:
Hit rate:
Improved speed:
Terminal
Hello world.
Hello world.
Hello world.
Hello world.
Peripherals
LED RGB 1
LED RGB 2
00000000
00000000
Red Knob
Green Knob
Blue Knob
0
0
0
Word hexadecimal
Word decimal
00000000
0
Word binary
00000000000000000000000000
+8
0x00000448 00000000 00000000 00000000
0x00000454 00000000 00000000 00000000
0x0000040c
Ready

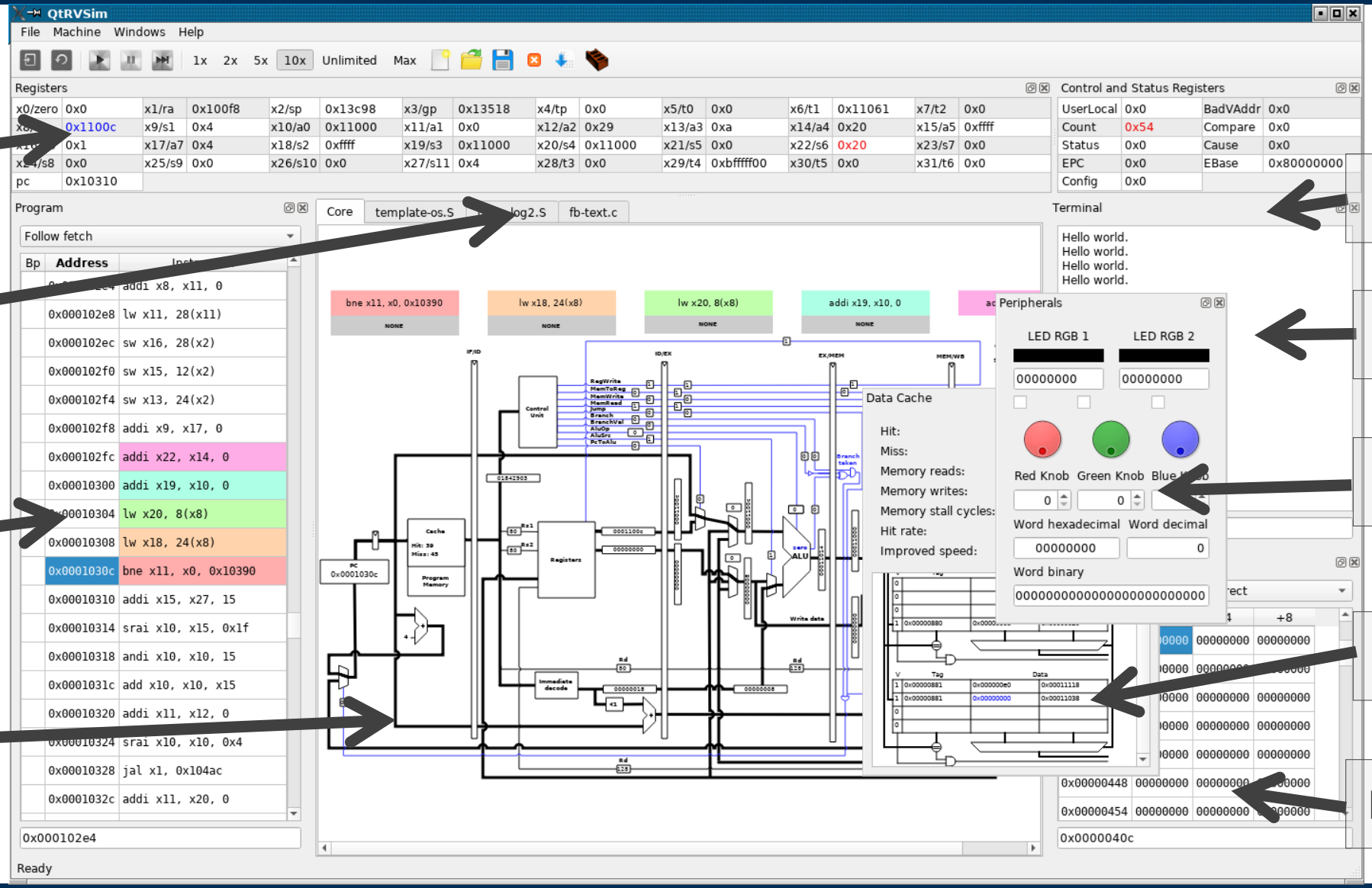
Layout

Registers

Editor With Assembler

Code

Core view single cycle pipelined



CSR

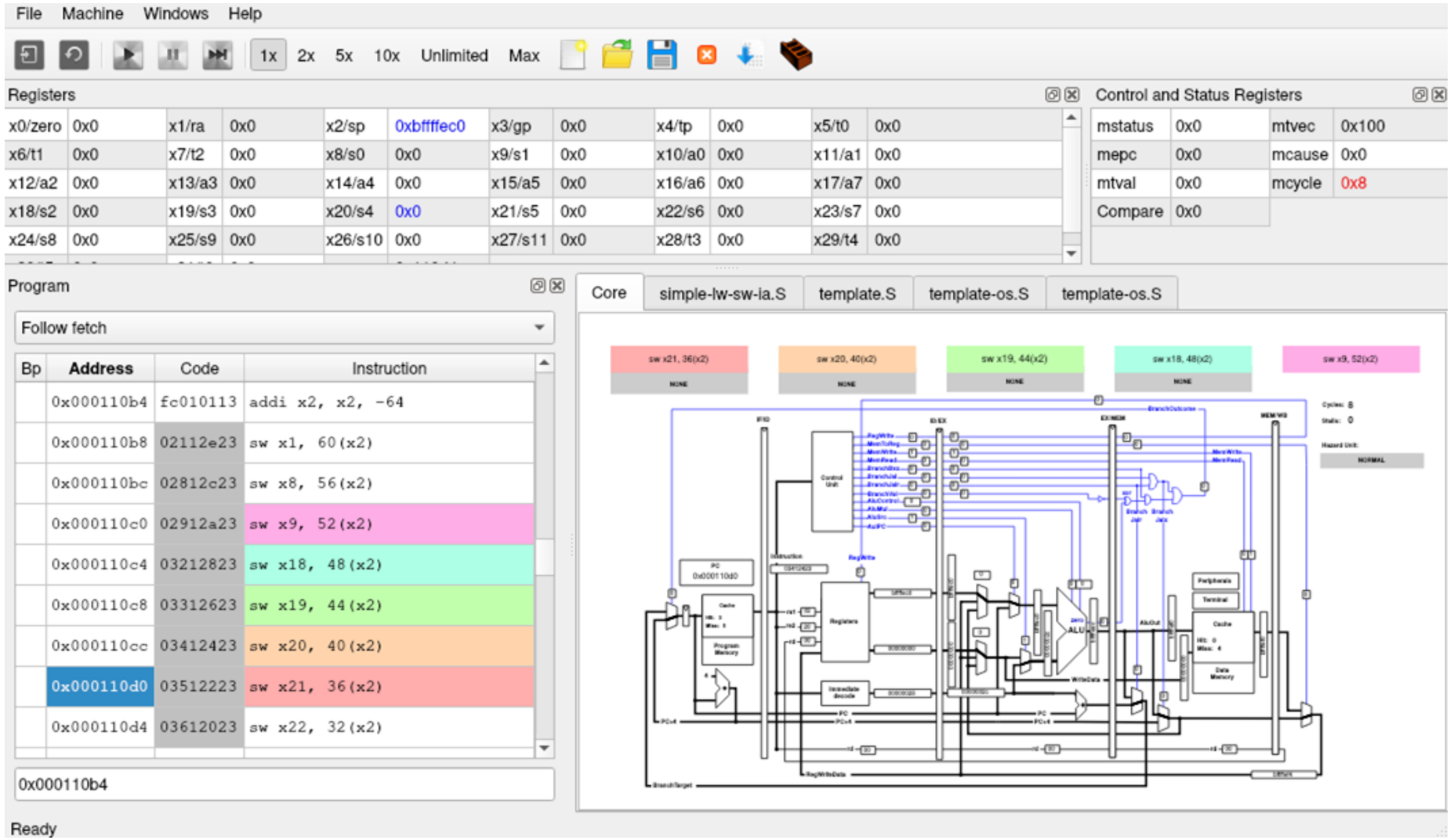
Terminal

Peripherals

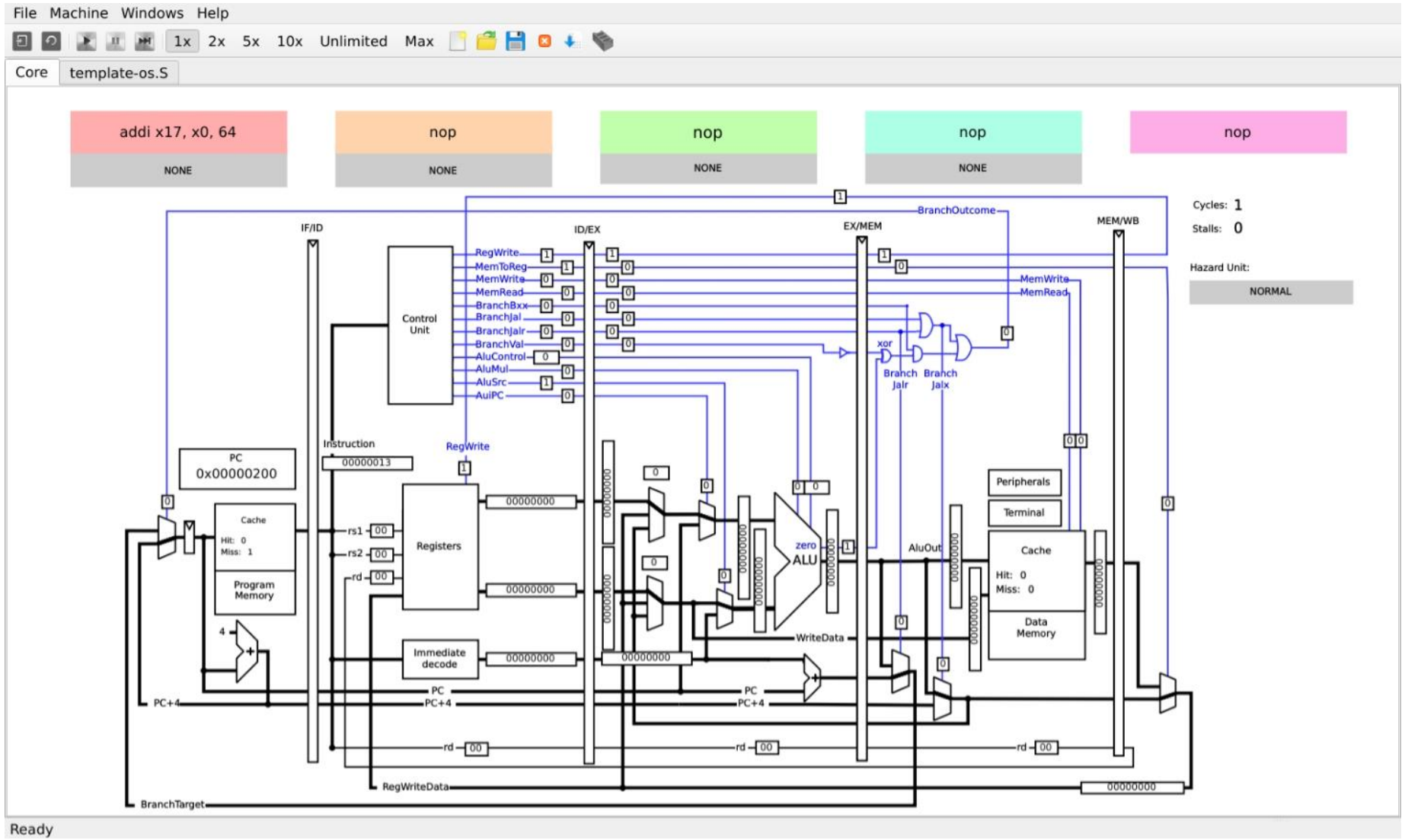
Cache

Data memory

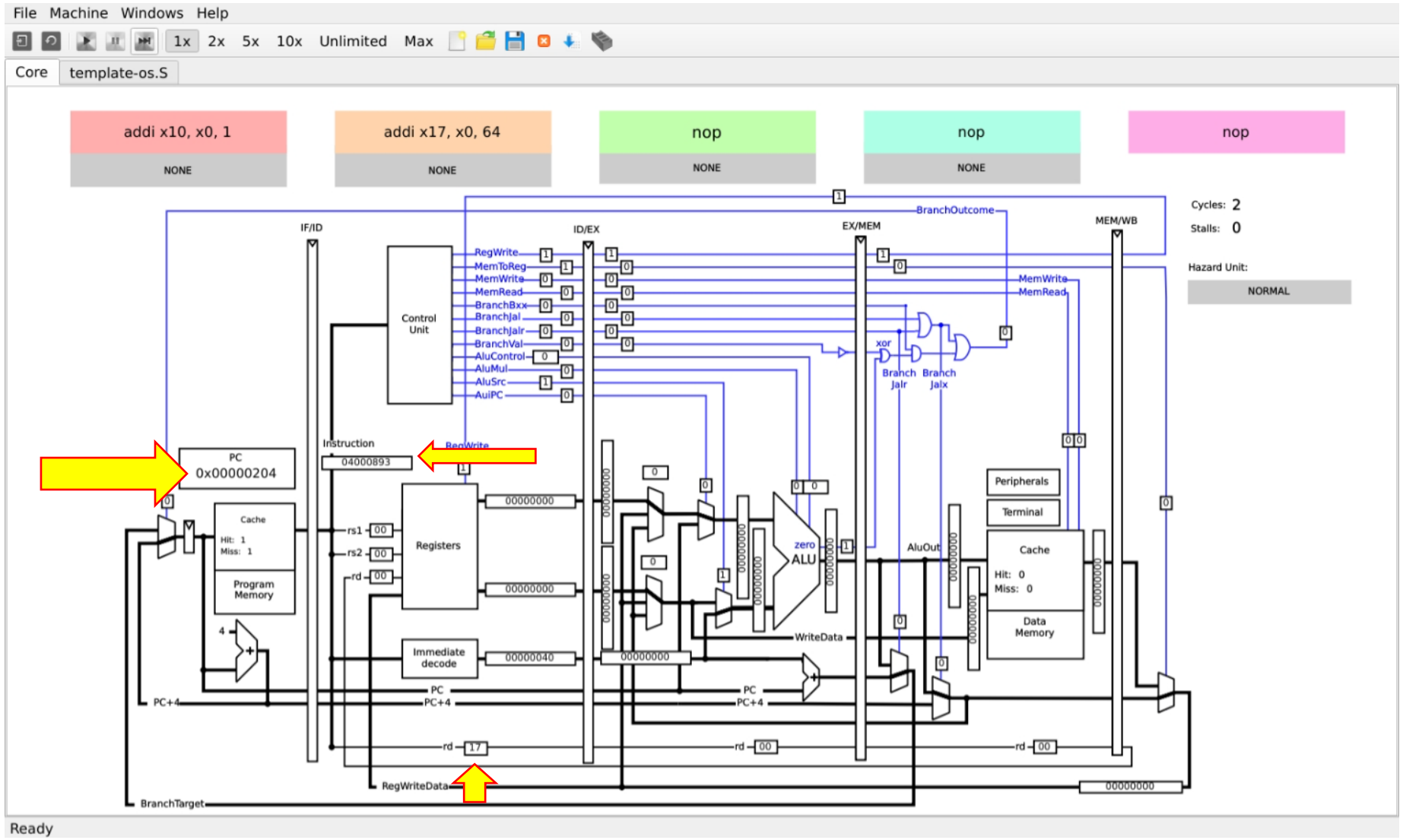
Basic view


File Machine Windows Help
1x 2x 5x 10x Unlimited Max
Registers
x0/zero 0x0 x1/ra 0x0 x2/sp 0xbffffec0 x3/gp 0x0 x4/tp 0x0 x5/t0 0x0
x6/t1 0x0 x7/t2 0x0 x8/s0 0x0 x9/s1 0x0 x10/a0 0x0 x11/a1 0x0
x12/a2 0x0 x13/a3 0x0 x14/a4 0x0 x15/a5 0x0 x16/a6 0x0 x17/a7 0x0
x18/s2 0x0 x19/s3 0x0 x20/s4 0x0 x21/s5 0x0 x22/s6 0x0 x23/s7 0x0
x24/s8 0x0 x25/s9 0x0 x26/s10 0x0 x27/s11 0x0 x28/t3 0x0 x29/t4 0x0
Control and Status Registers
mstatus 0x0 mtvec 0x100
mepc 0x0 mcause 0x0
mtval 0x0 mcycle 0x8
Compare 0x0
Program
Follow fetch
Bp Address Code Instruction
0x000110b4 fc010113 addi x2, x2, -64
0x000110b8 02112e23 sw x1, 60(x2)
0x000110bc 02812c23 sw x8, 56(x2)
0x000110c0 02912a23 sw x9, 52(x2)
0x000110c4 03212823 sw x18, 48(x2)
0x000110c8 03312623 sw x19, 44(x2)
0x000110cc 03412423 sw x20, 40(x2)
0x000110d0 03512223 sw x21, 36(x2)
0x000110d4 03612023 sw x22, 32(x2)
0x000110b4
Core
simple-lw-sw-ia.S
template.S
template-os.S
template-os.S
Ready

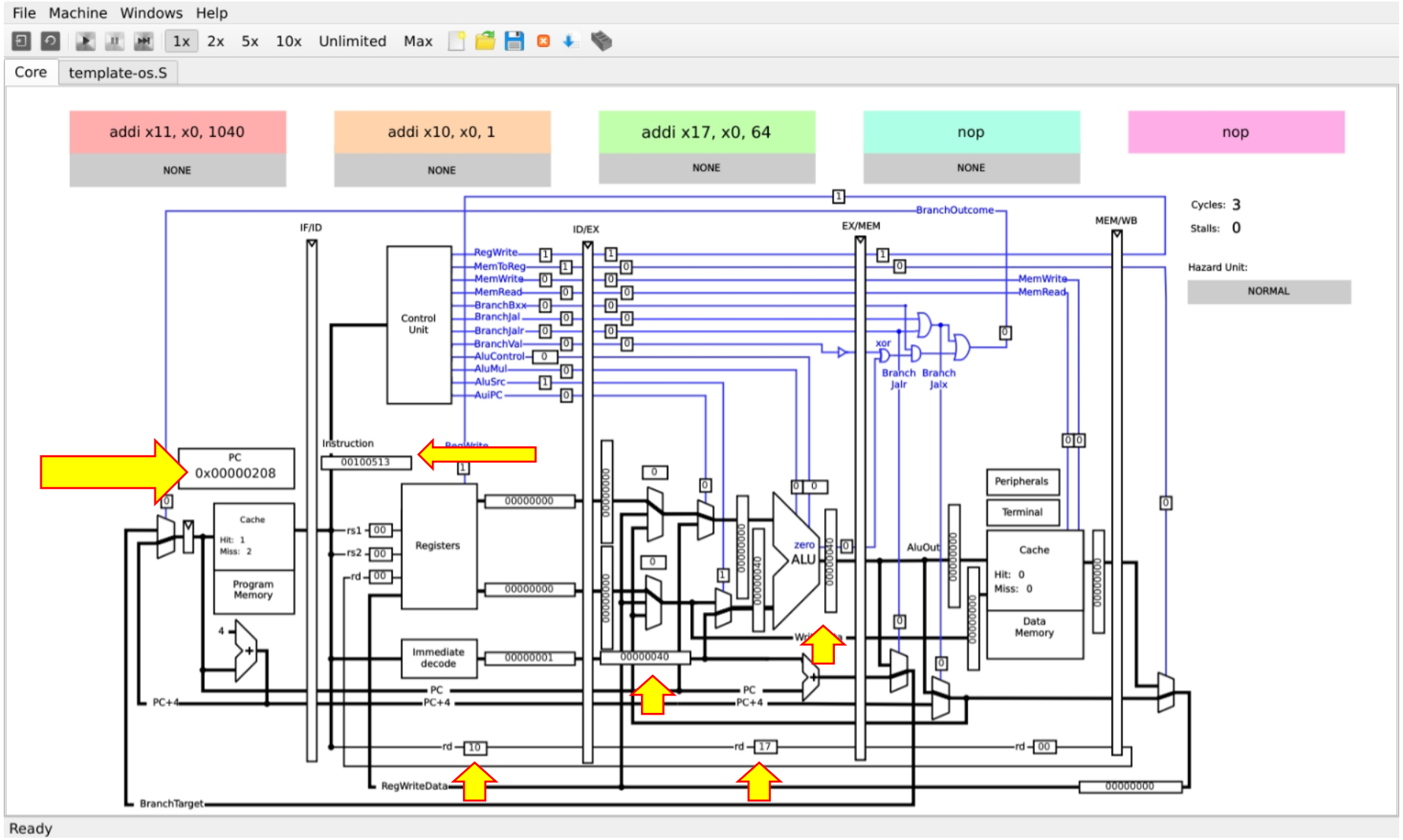
Pipeline Visualization 1



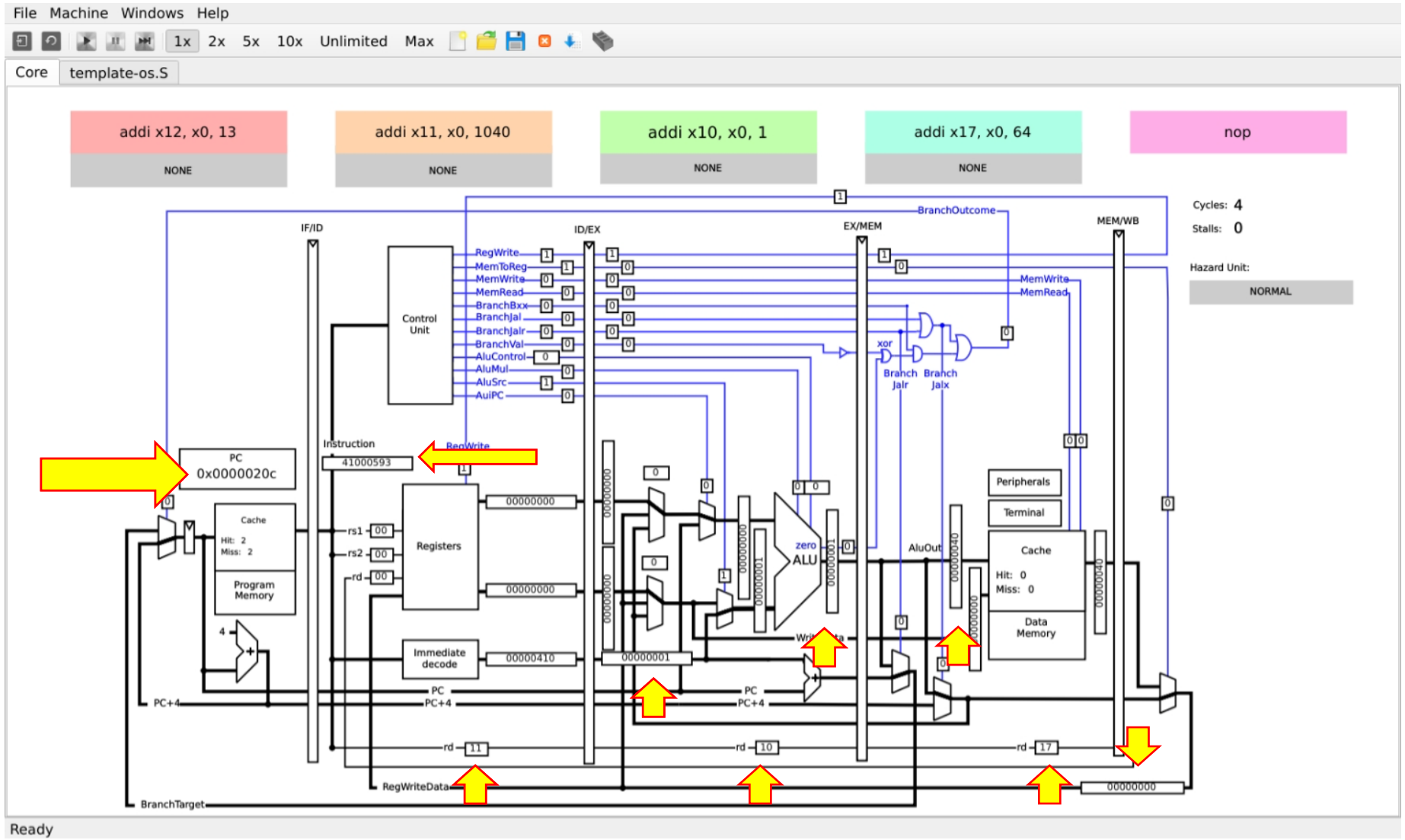
Pipeline Visualization 2



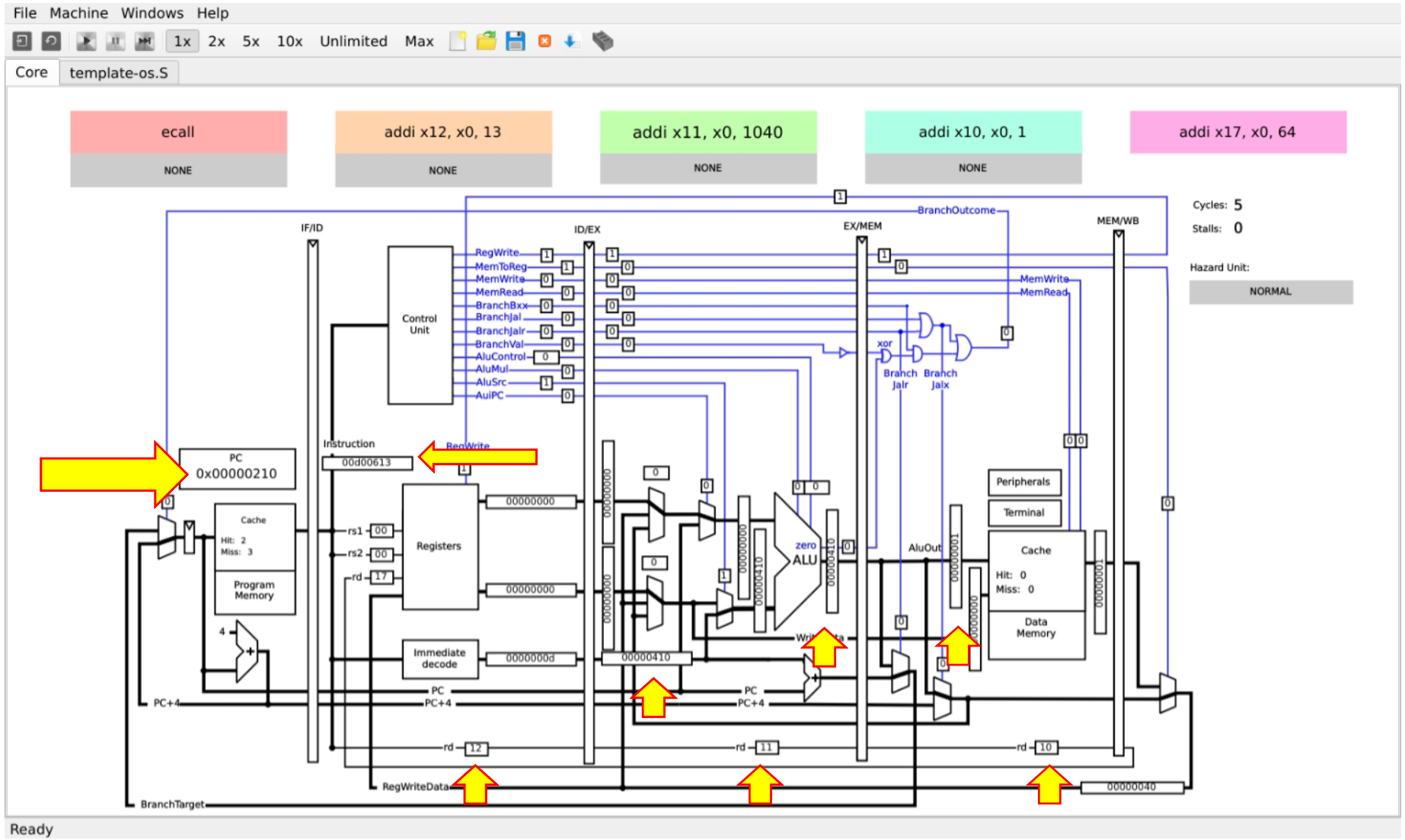
Pipeline Visualization 3



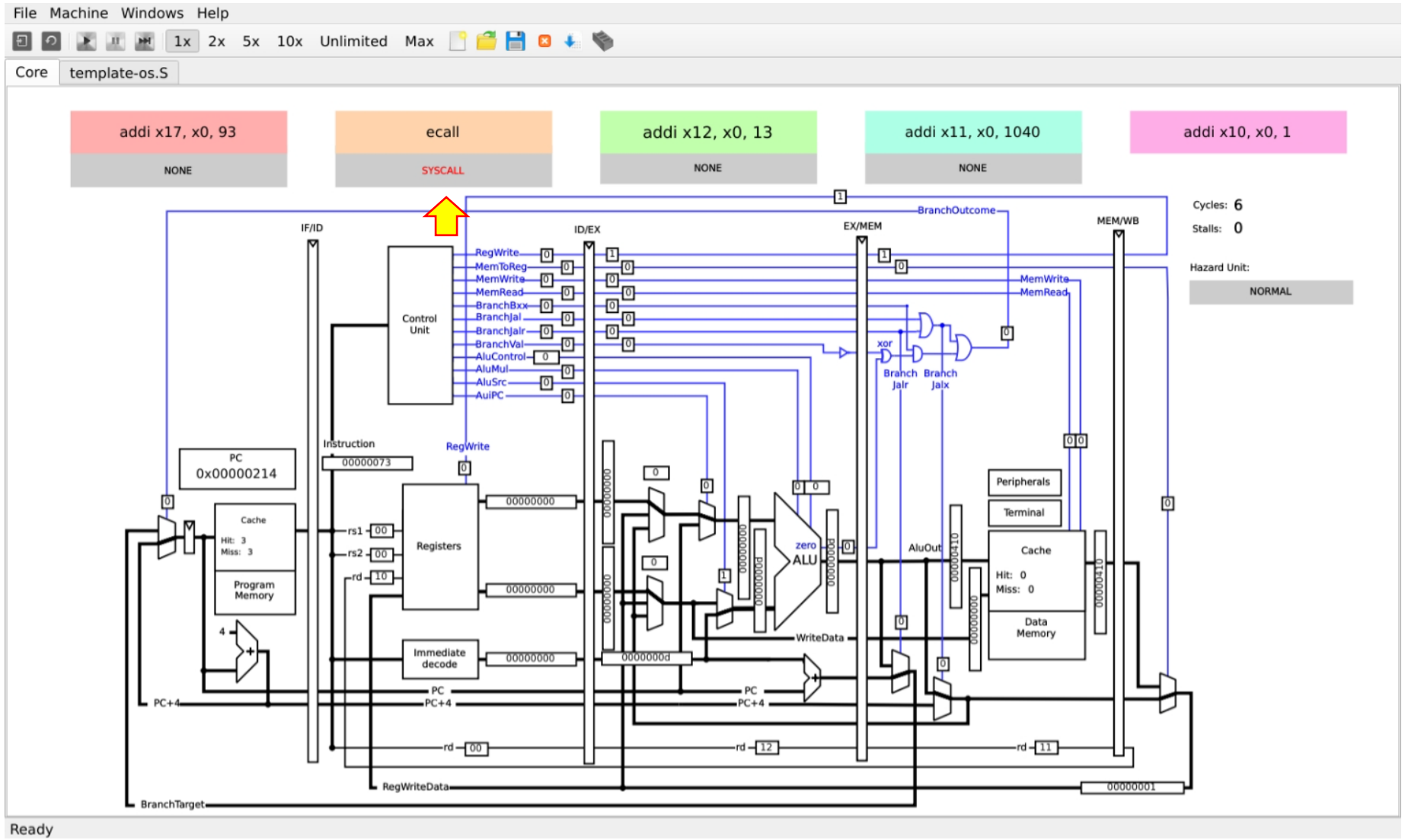
Pipeline Visualization 4



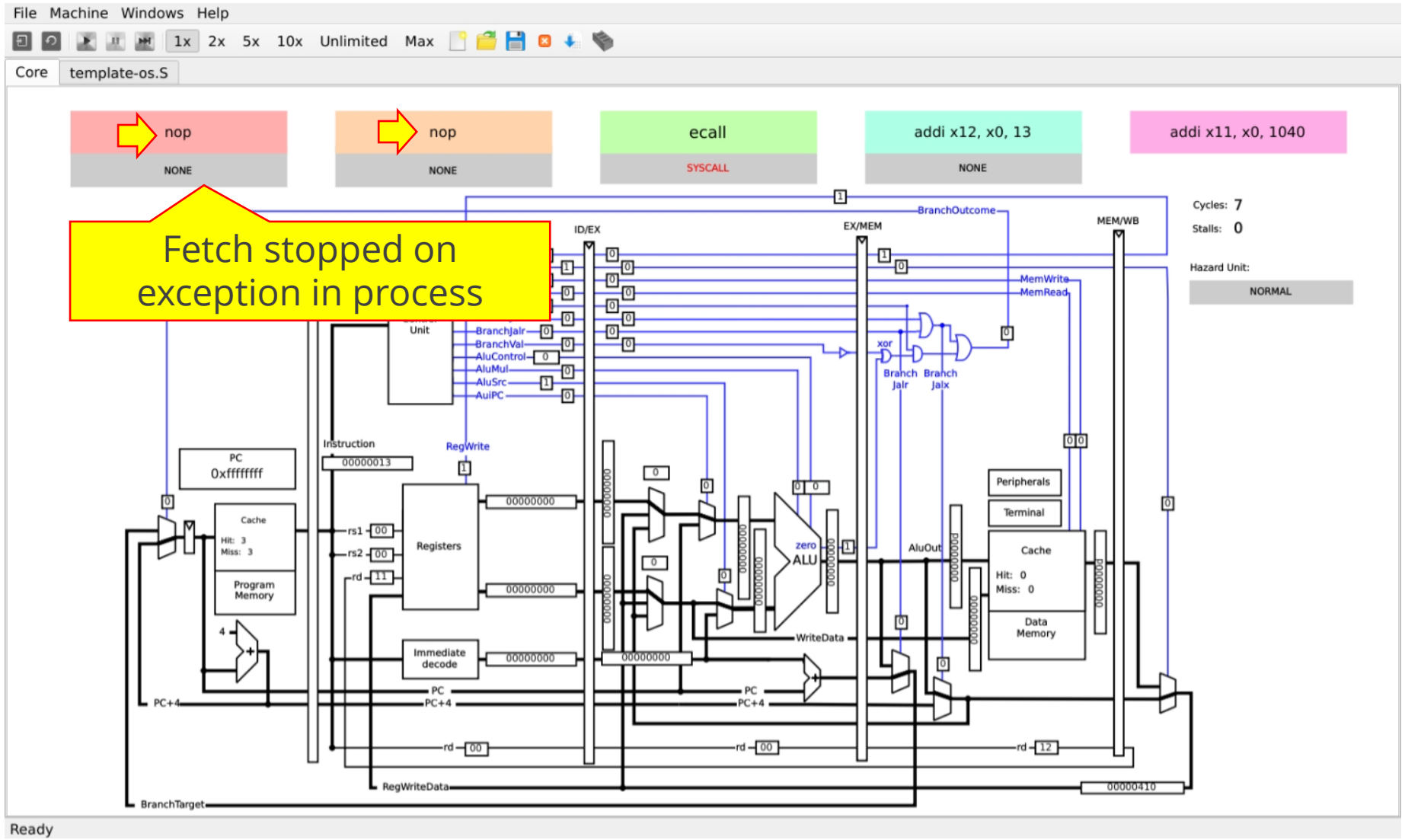
Pipeline Visualization 5



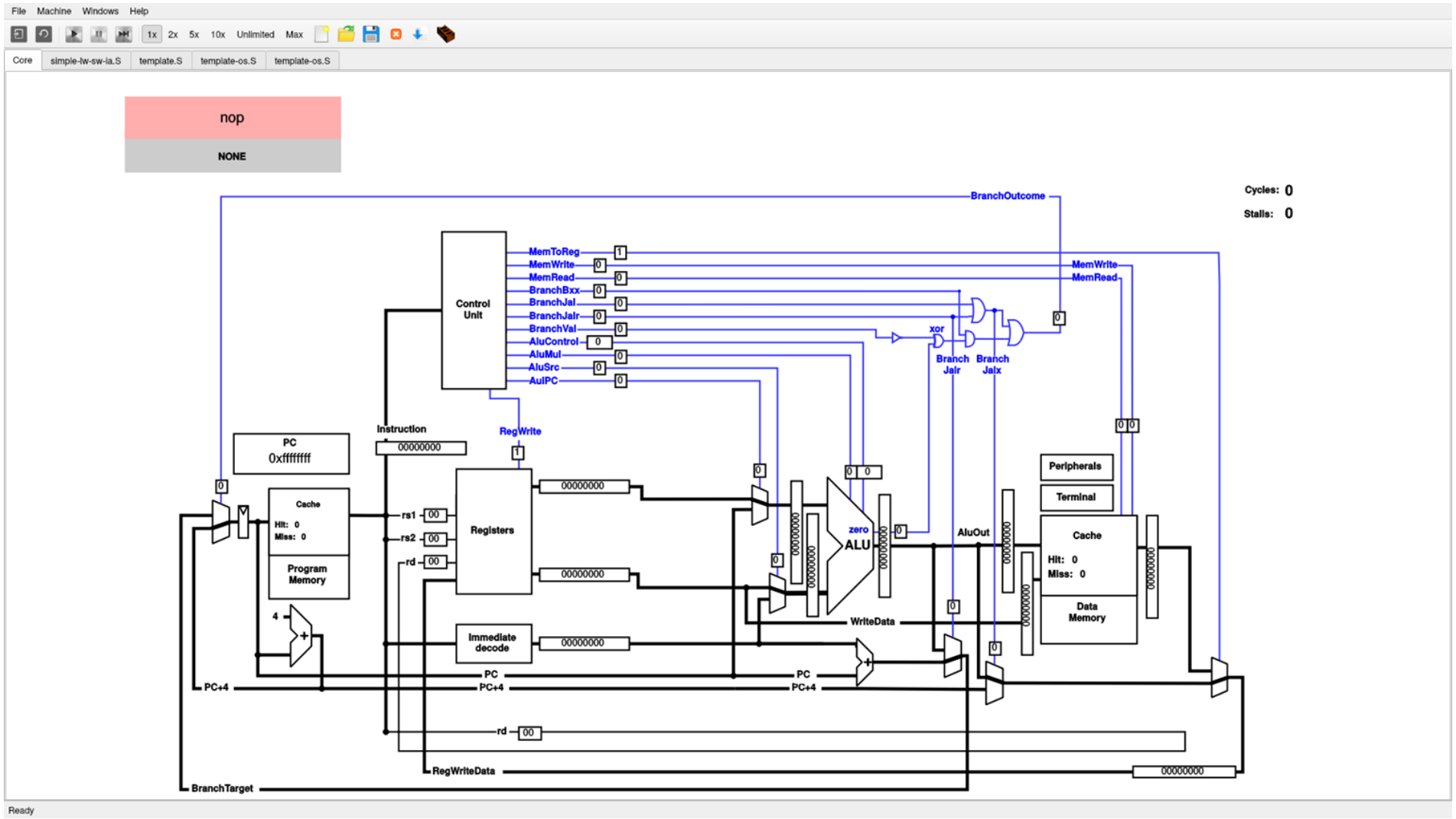
Pipeline Visualization 6



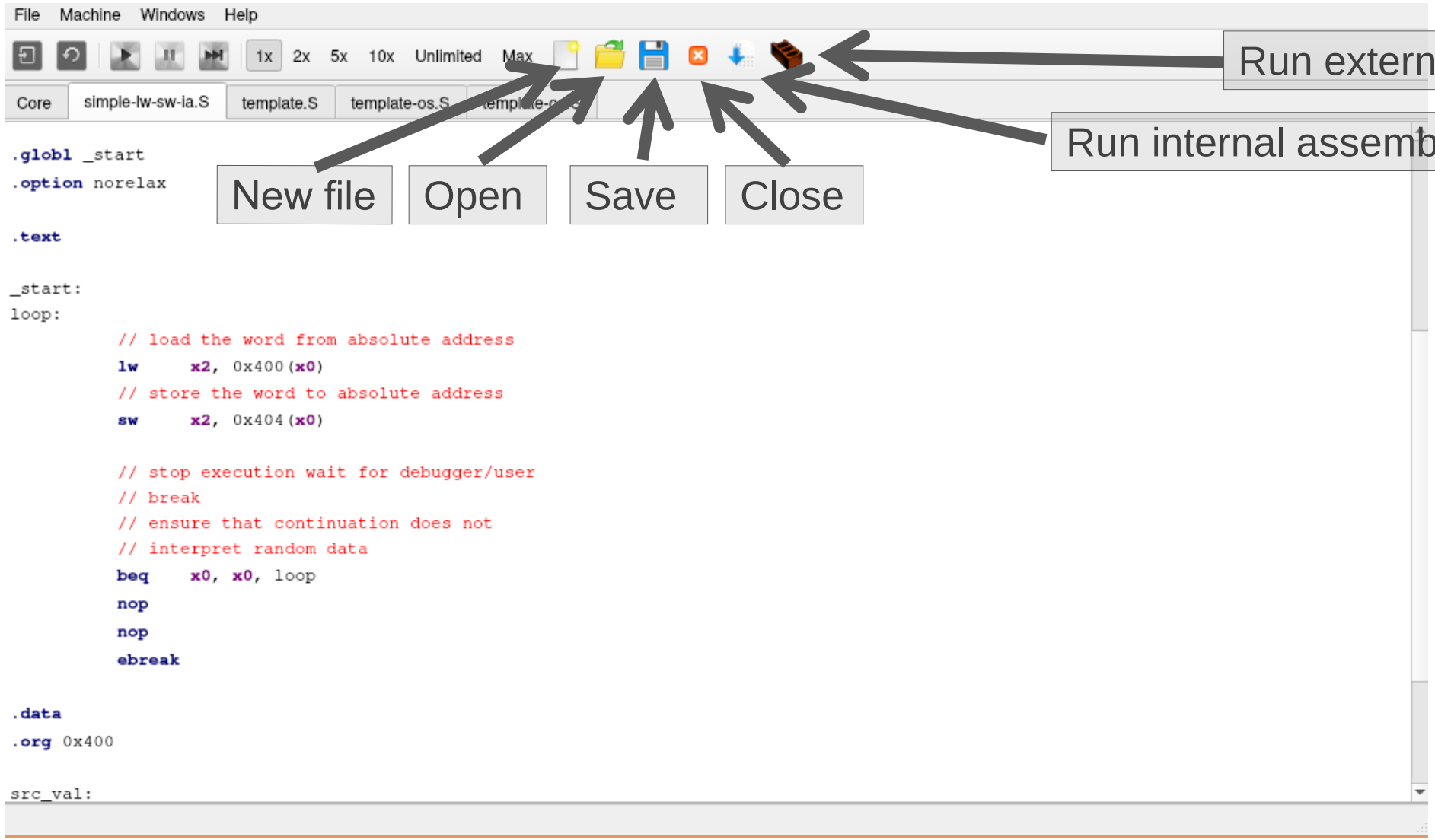
Pipeline Visualization 7



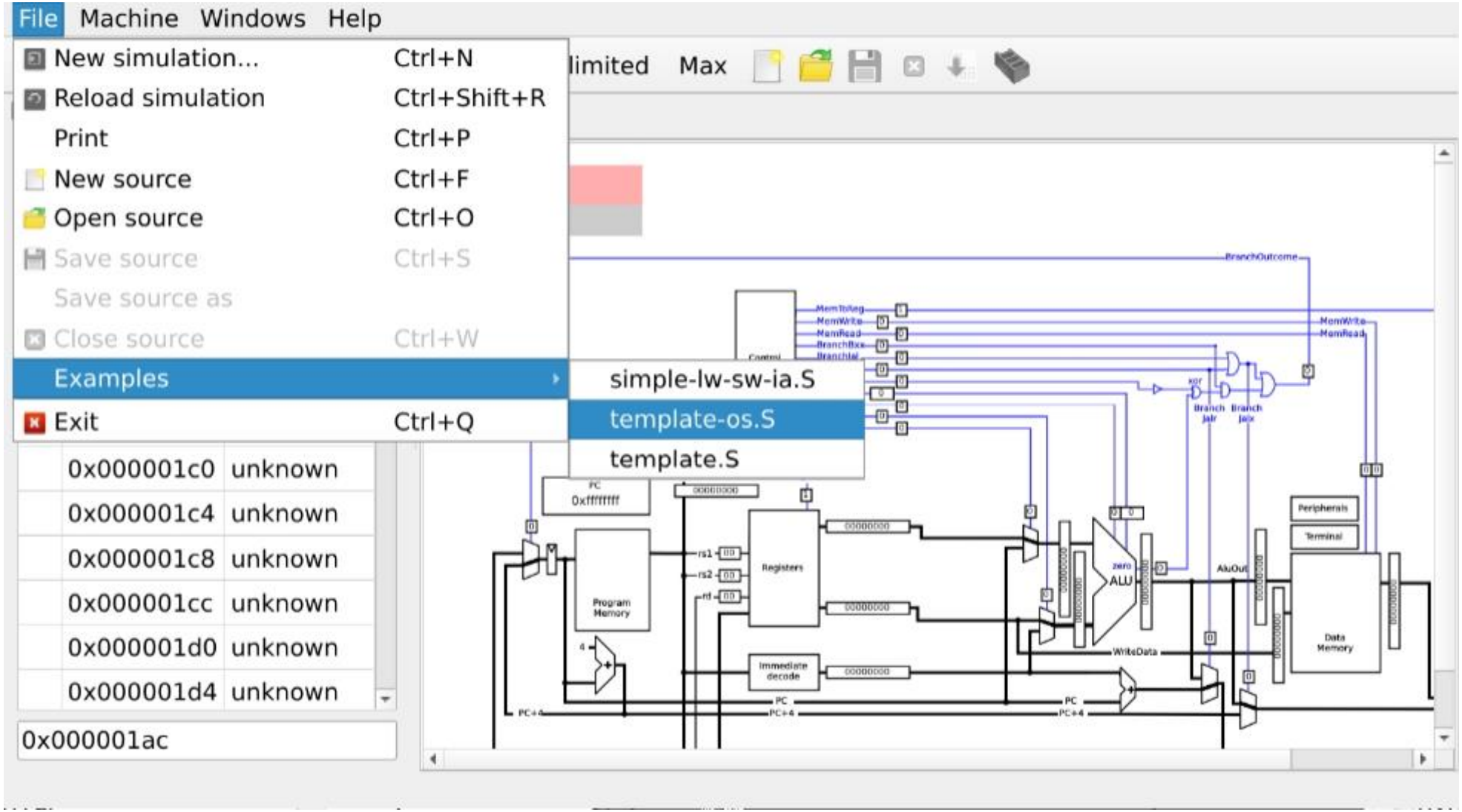
Single Cycle Core



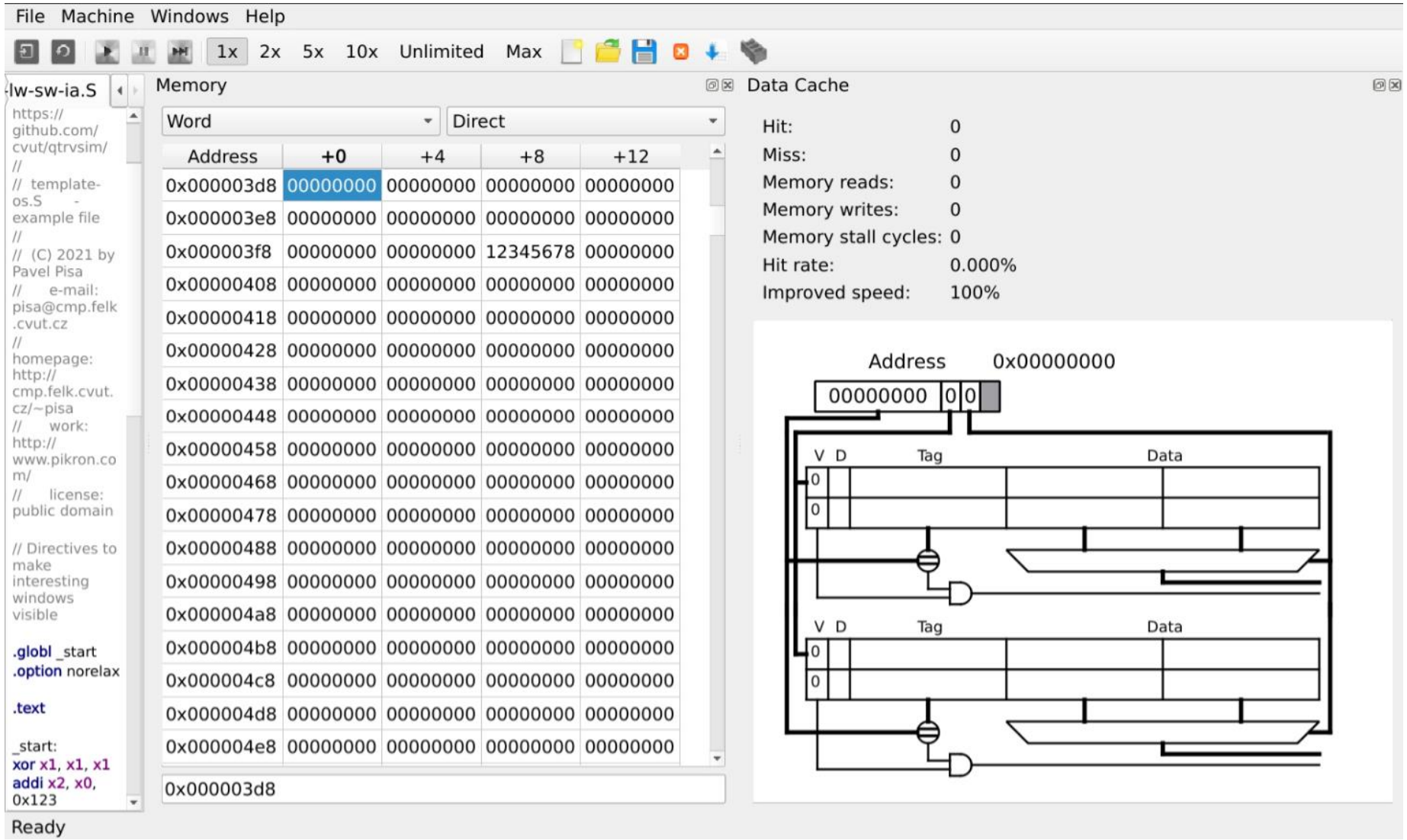
Integrated editor



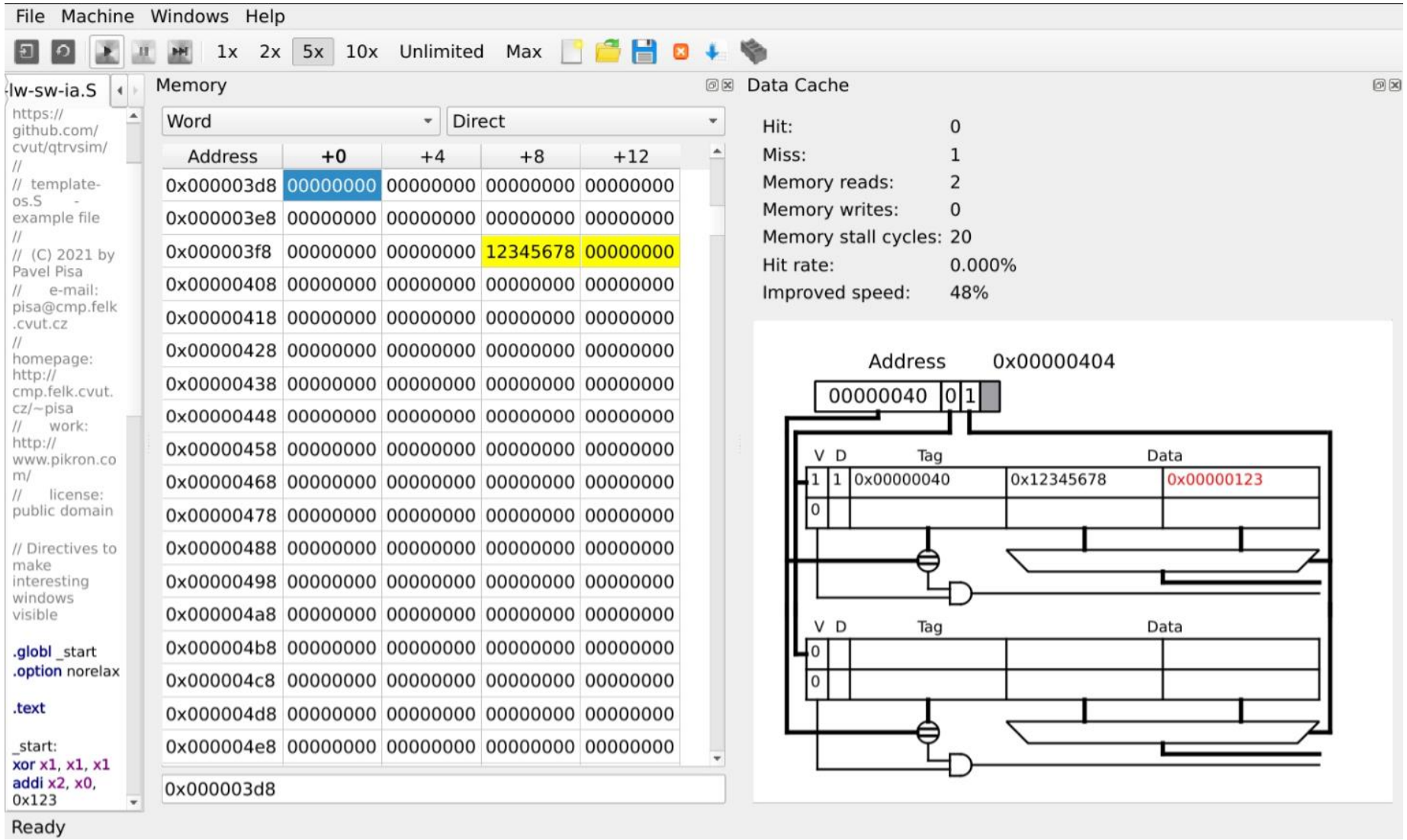
Built-in Examples



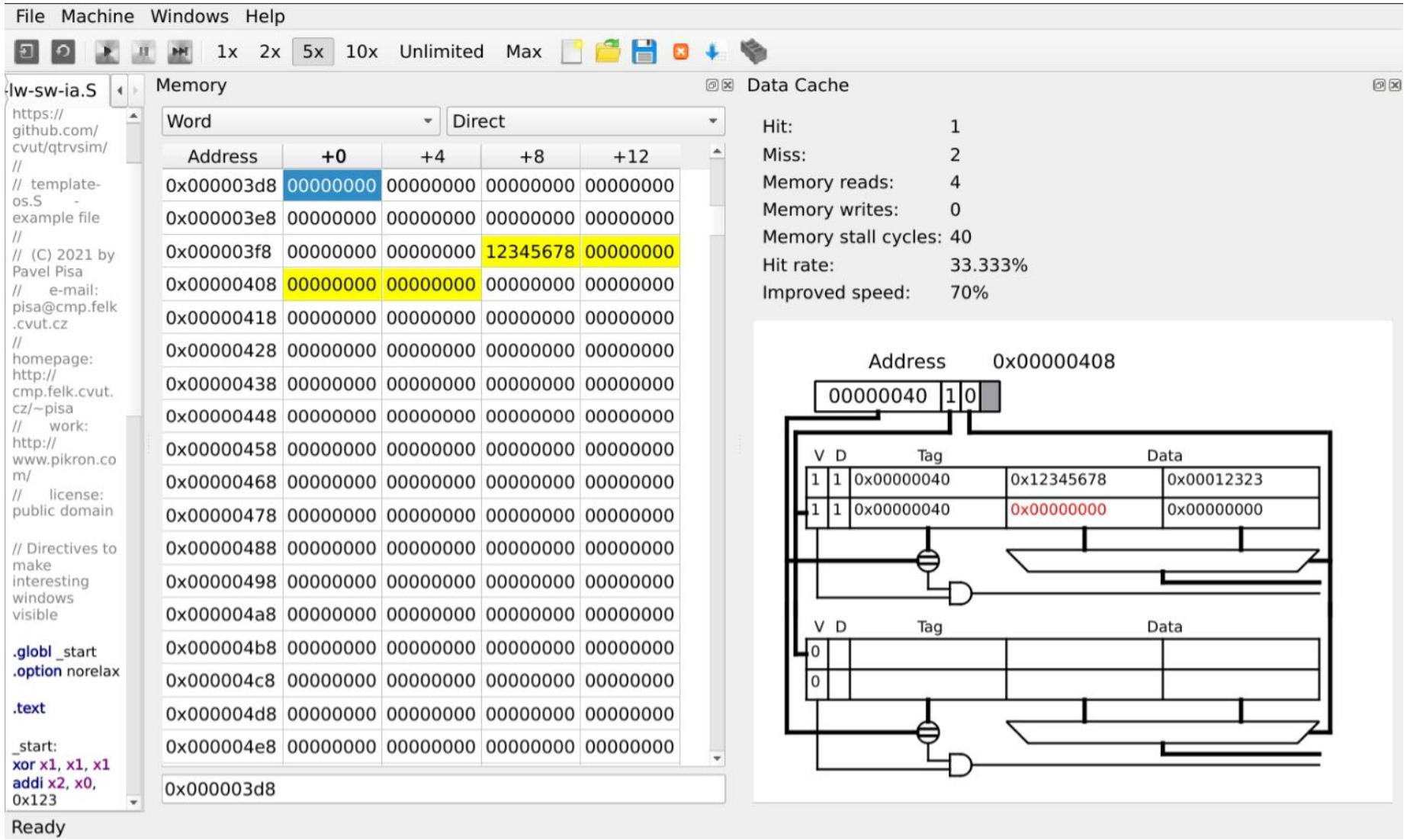
Memory and Cache



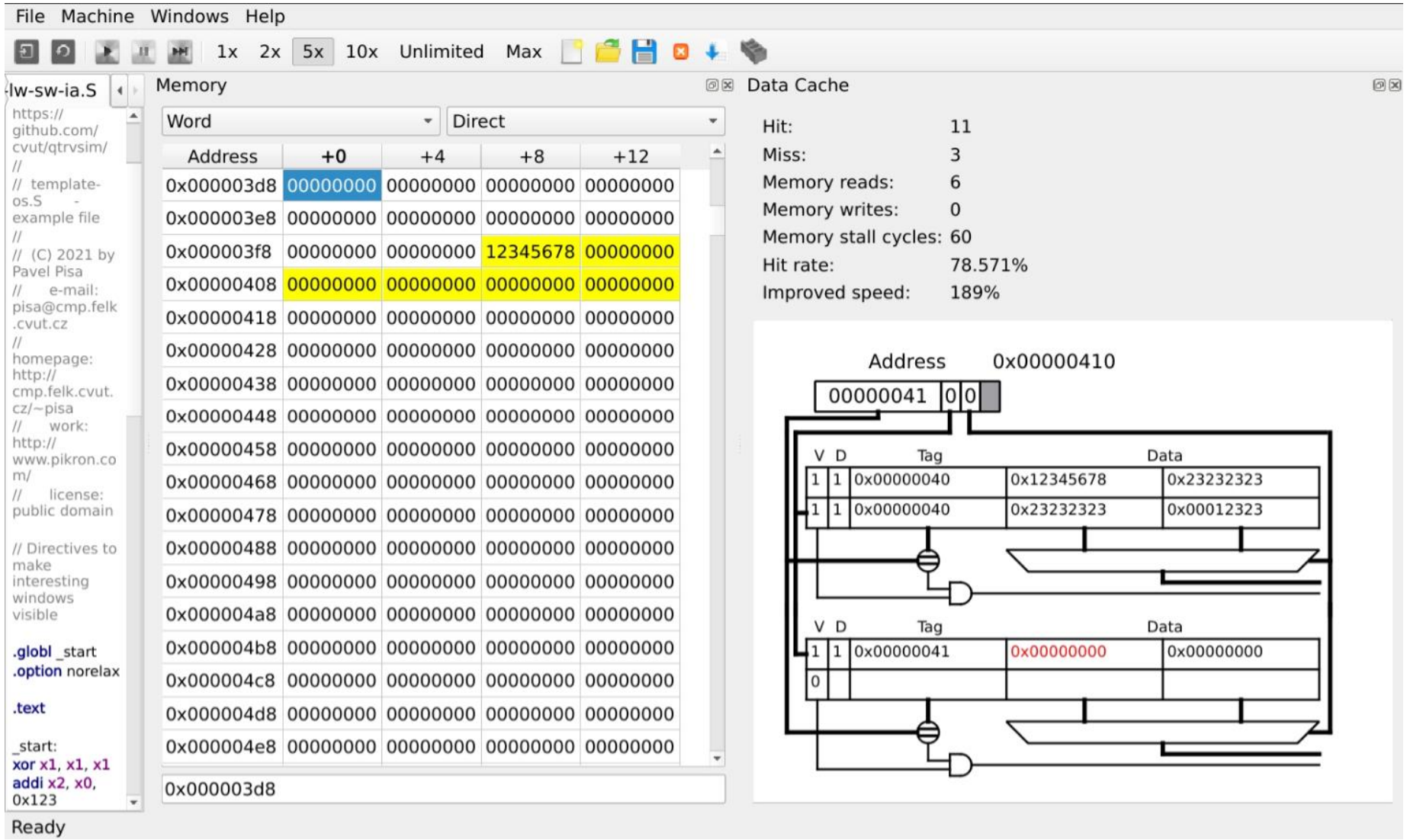
Memory and Cache 1



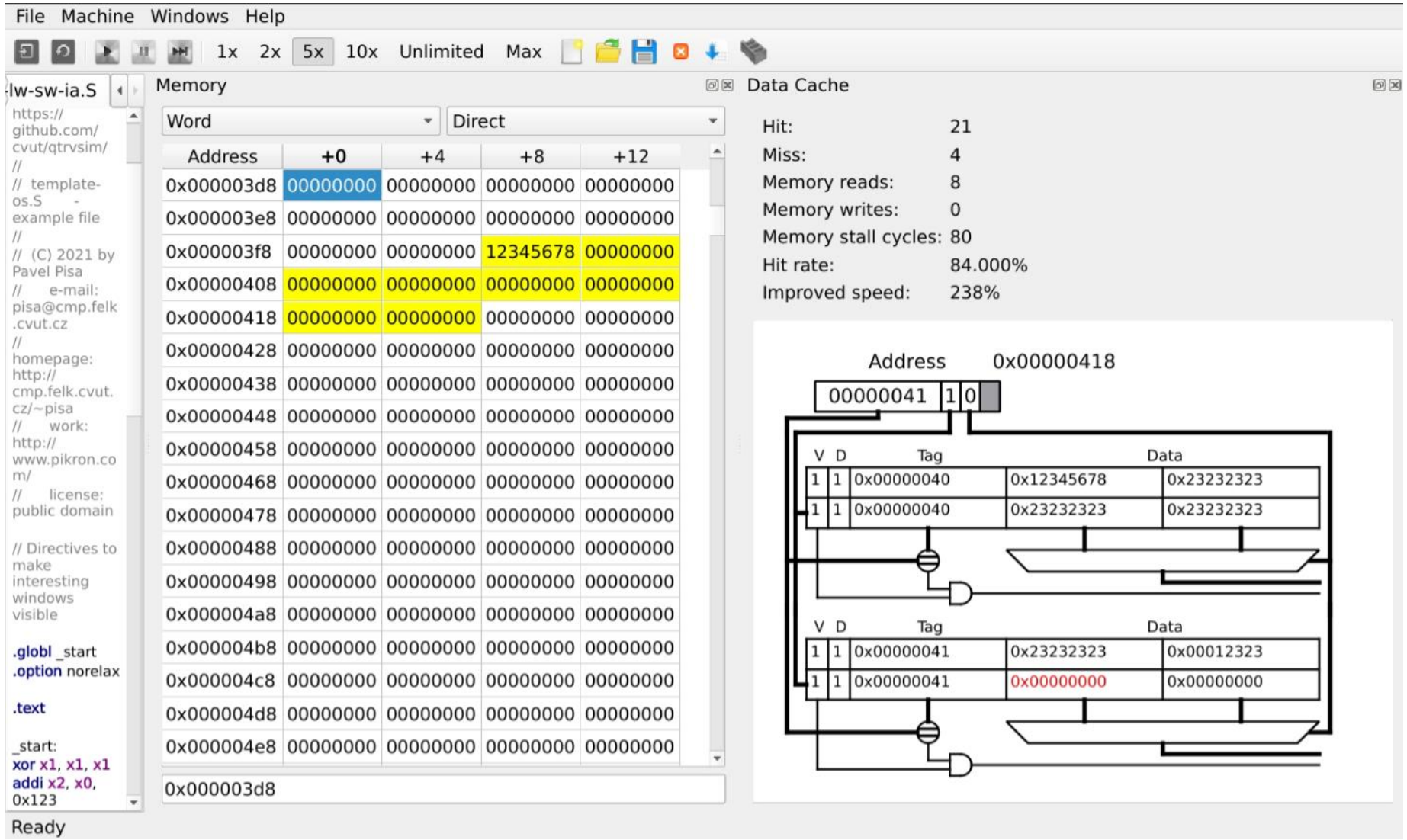
Memory and Cache 2



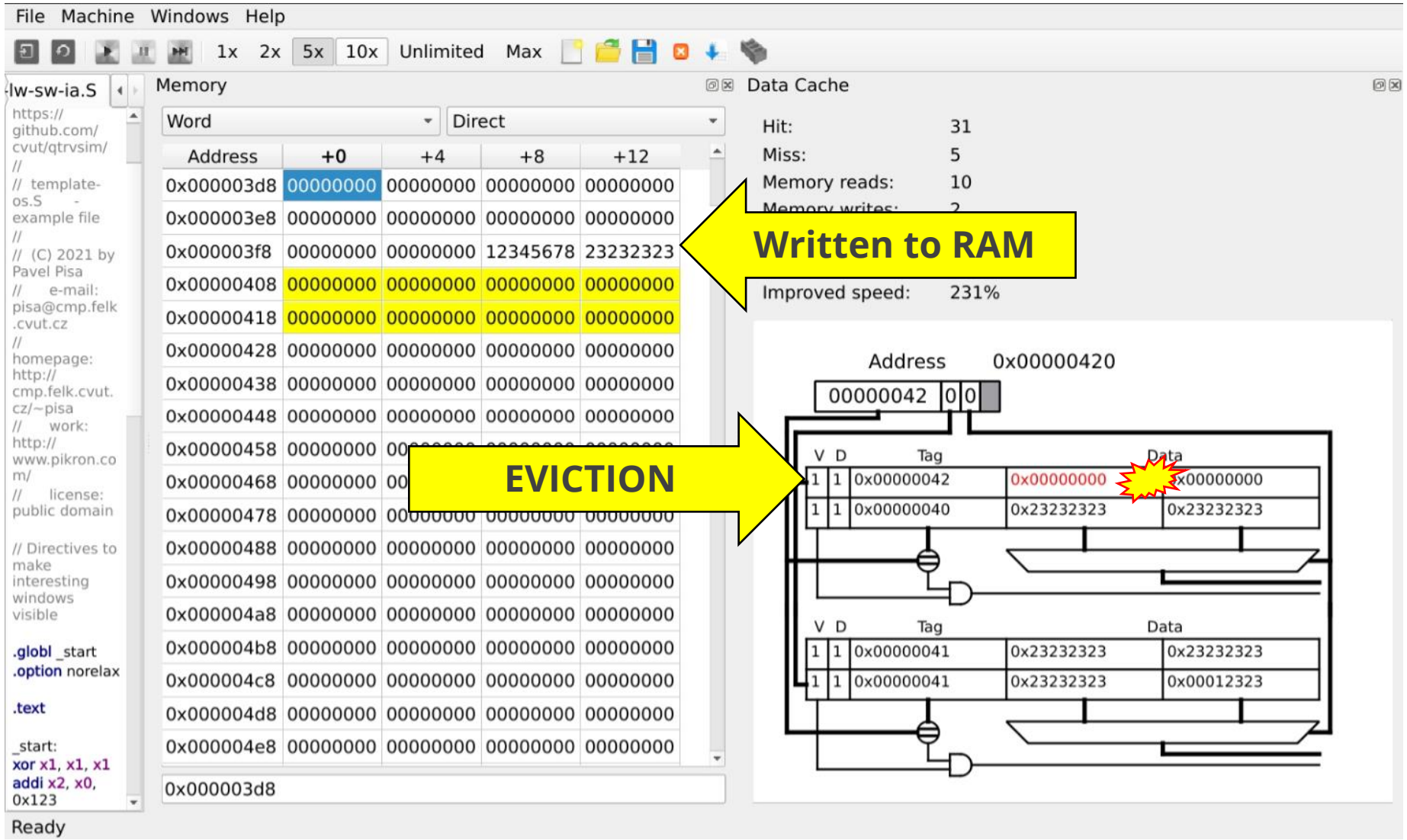
Memory and Cache 3



Memory and Cache 4



Memory and Cache 5



Different Cache Configuration

File Machine Windows Help

1x 2x 5x 10x Unlimited Max

lw-sw-ia.S

```
https://
github.com/
cvut/qtrvsim/
//
// template-
os.S      -
example file
//
// (C) 2021 by
Pavel Pisa
//     e-mail:
pisa@cmp.felk
.cvut.cz
//
homepage:
http://
cmp.felk.cvut.
cz/~pisa
//     work:
http://
www.pikron.co
m/
//     license:
public domain

// Directives to
make
interesting
windows
visible

.globl _start
.option norelax

.text

_start:
xor x1, x1, x1
addi x2, x0,
0x123
```

Memory

Word | Direct

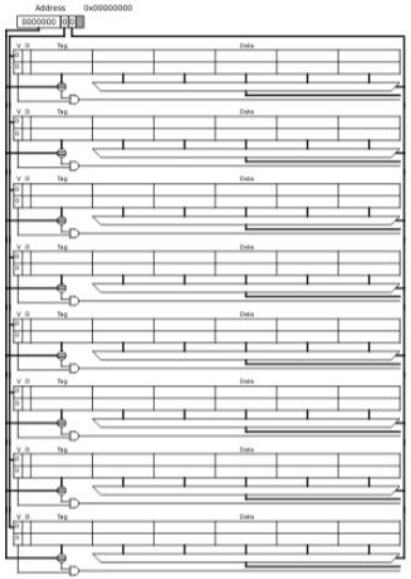
Address	+0	+4	+8	+12
0x000003d8	00000000	00000000	00000000	00000000
0x000003e8	00000000	00000000	00000000	00000000
0x000003f8	00000000	00000000	12345678	23232323
0x00000408	00000000	00000000	00000000	00000000
0x00000418	00000000	00000000	00000000	00000000
0x00000428	00000000	00000000	00000000	00000000
0x00000438	00000000	00000000	00000000	00000000
0x00000448	00000000	00000000	00000000	00000000
0x00000458	00000000	00000000	00000000	00000000
0x00000468	00000000	00000000	00000000	00000000
0x00000478	00000000	00000000	00000000	00000000
0x00000488	00000000	00000000	00000000	00000000
0x00000498	00000000	00000000	00000000	00000000
0x000004a8	00000000	00000000	00000000	00000000
0x000004b8	00000000	00000000	00000000	00000000
0x000004c8	00000000	00000000	00000000	00000000
0x000004d8	00000000	00000000	00000000	00000000
0x000004e8	00000000	00000000	00000000	00000000

0x000003d8

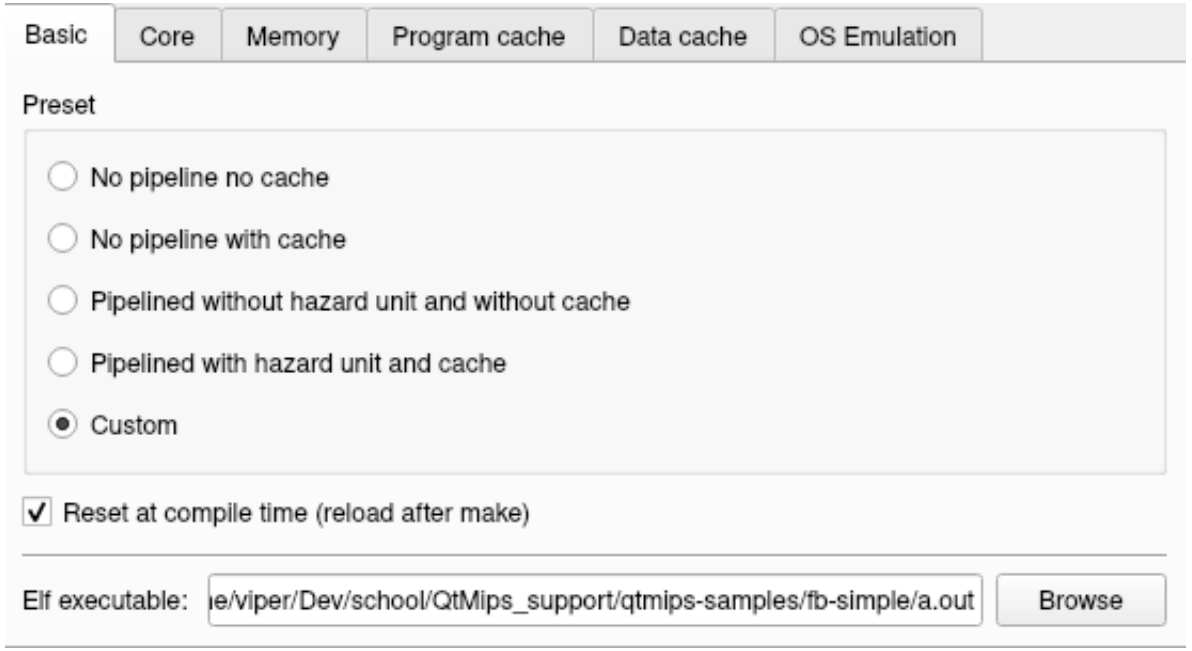
Ready

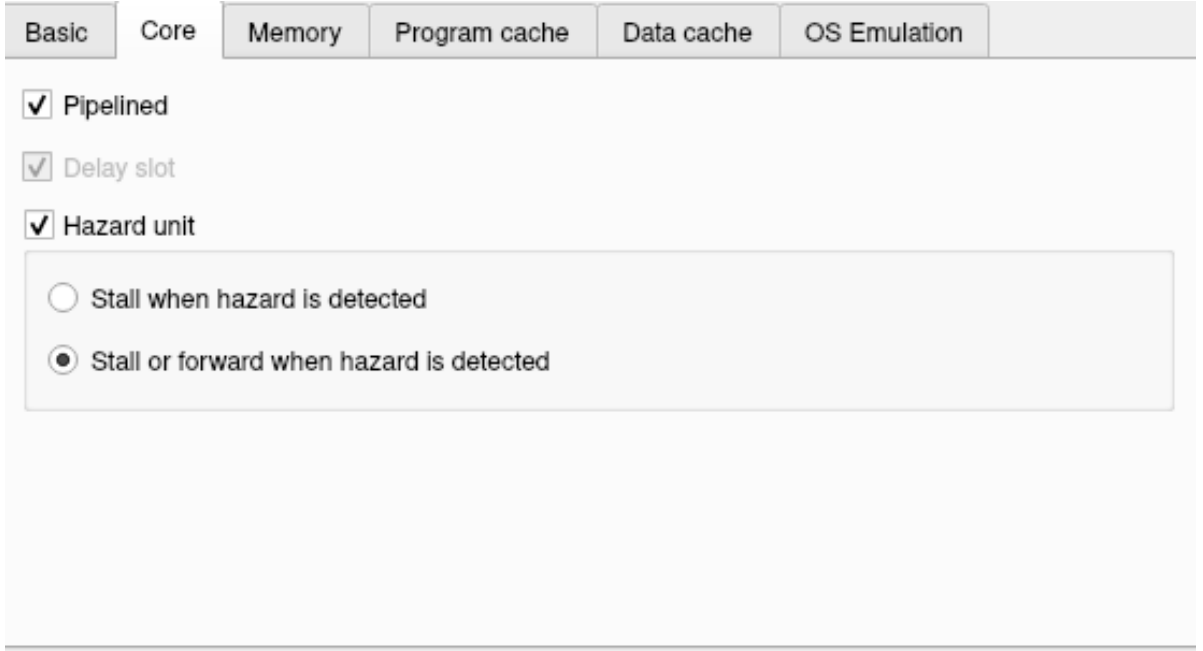
Data Cache

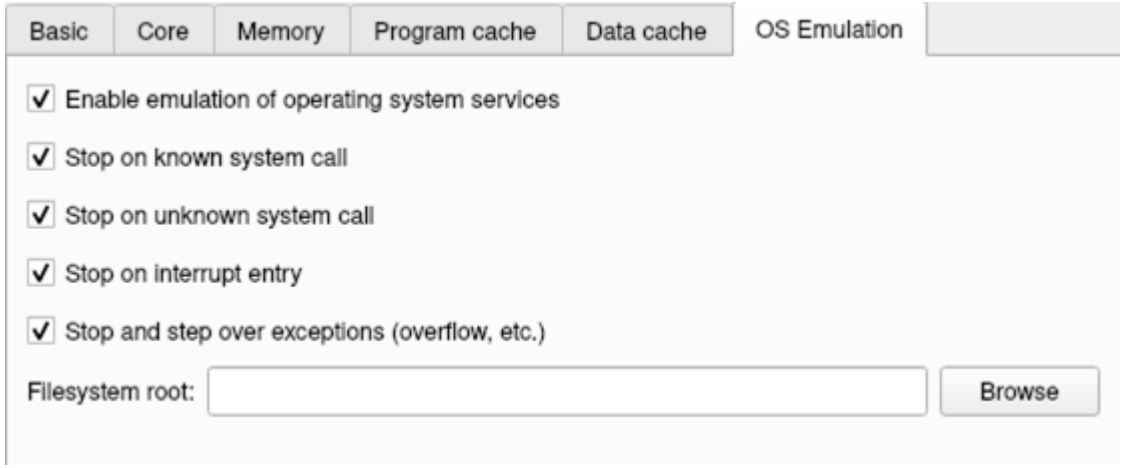
Hit:	0
Miss:	0
Memory reads:	0
Memory writes:	0
Memory stall cycles:	0
Hit rate:	0.000%
Improved speed:	100%



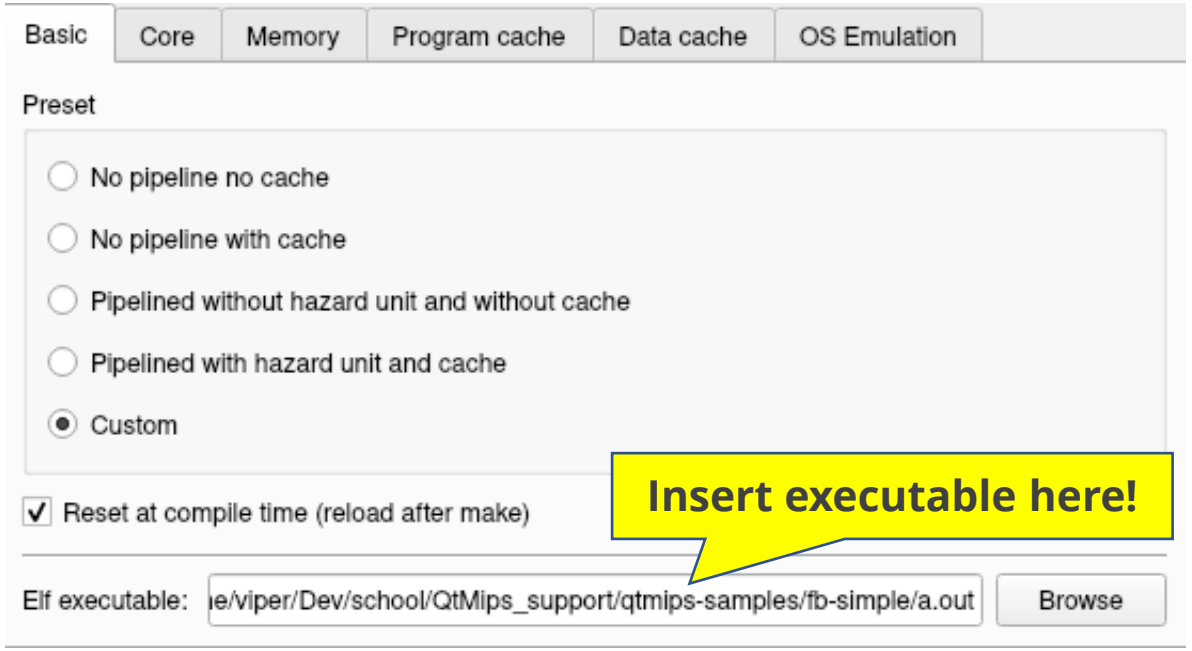
Configuration

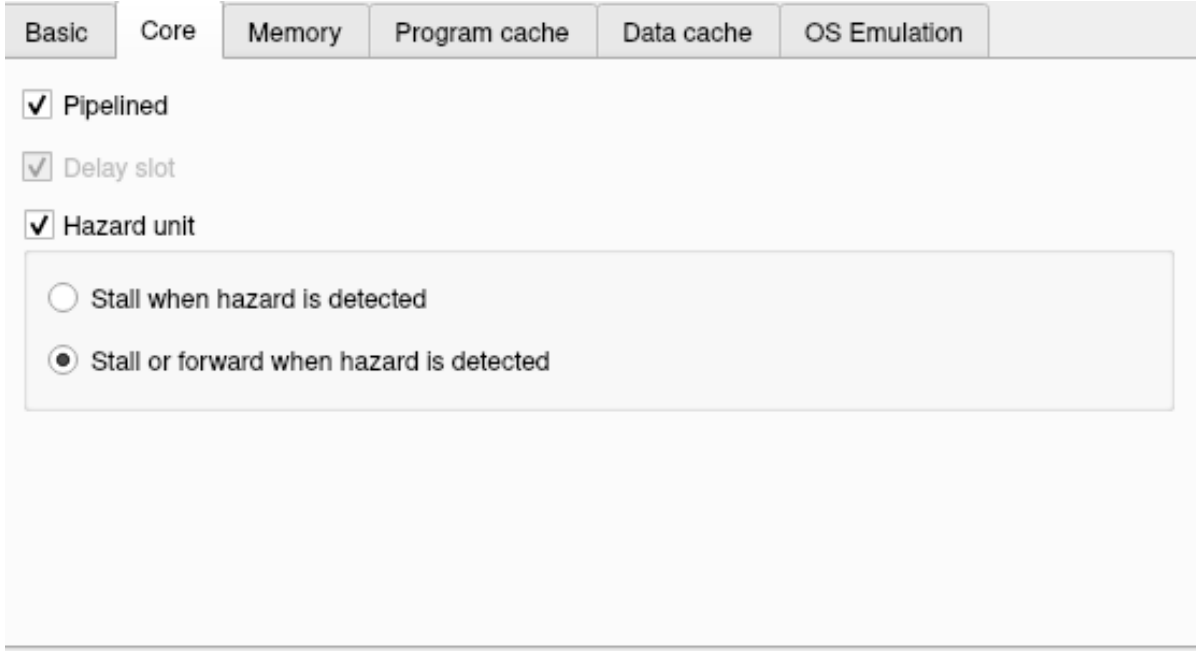


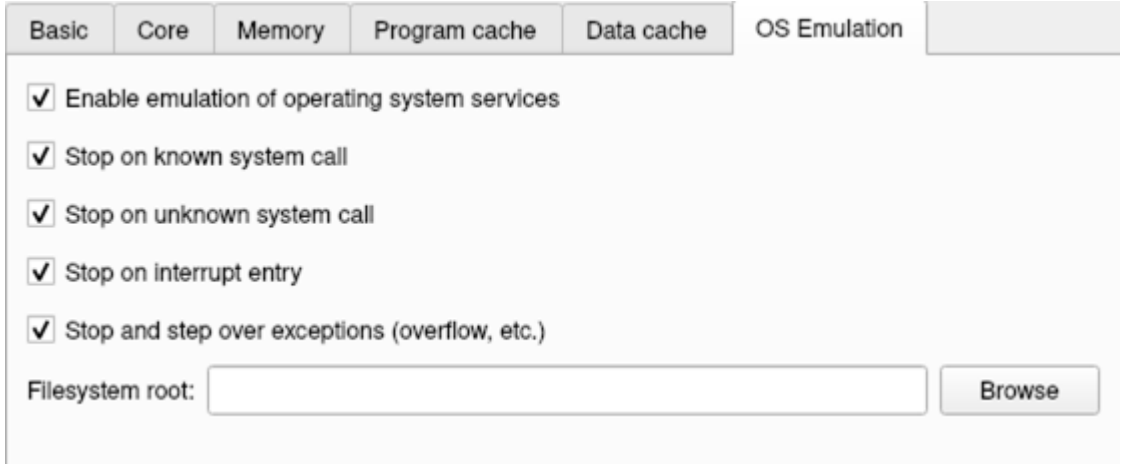




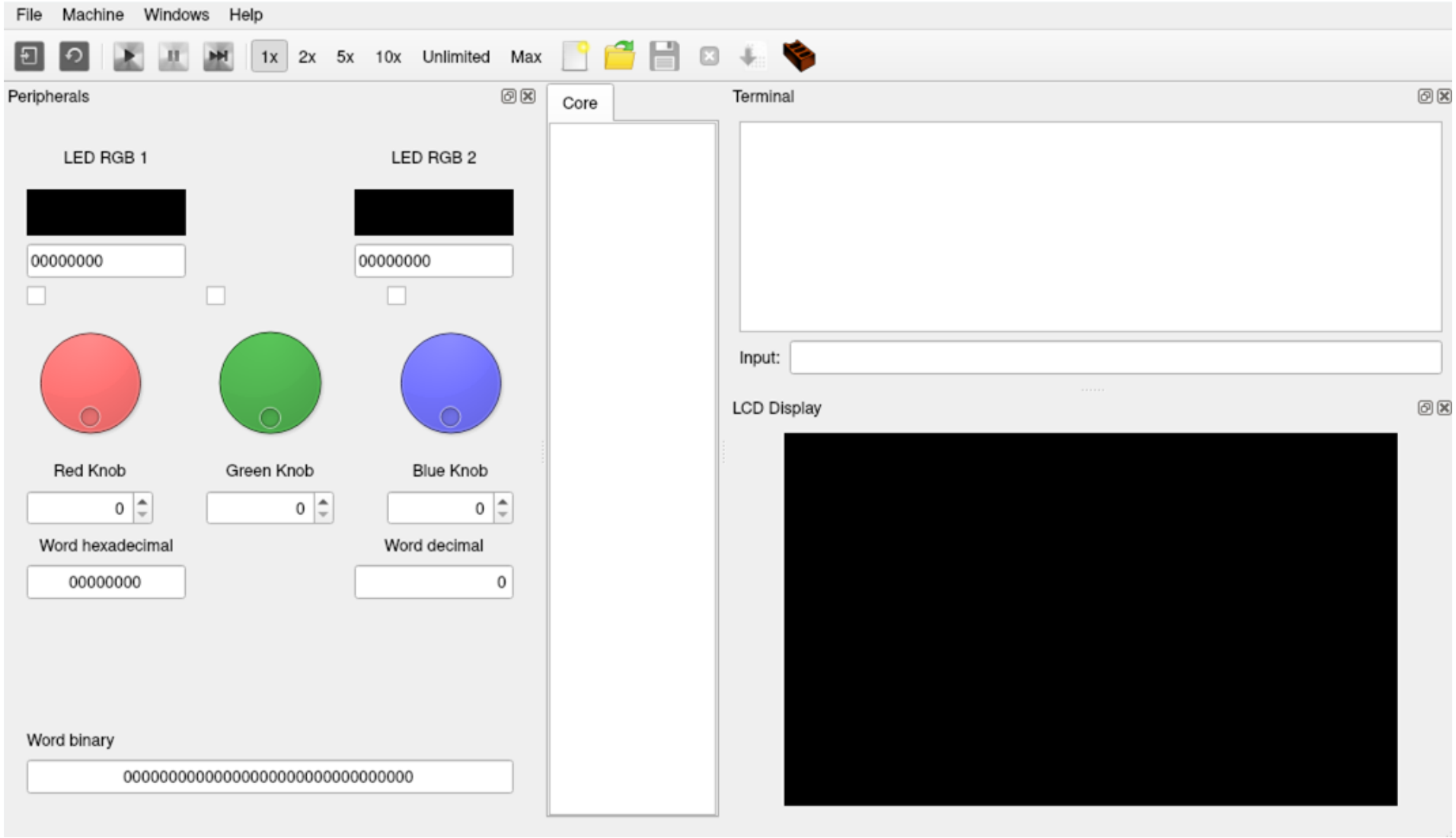
Configuration



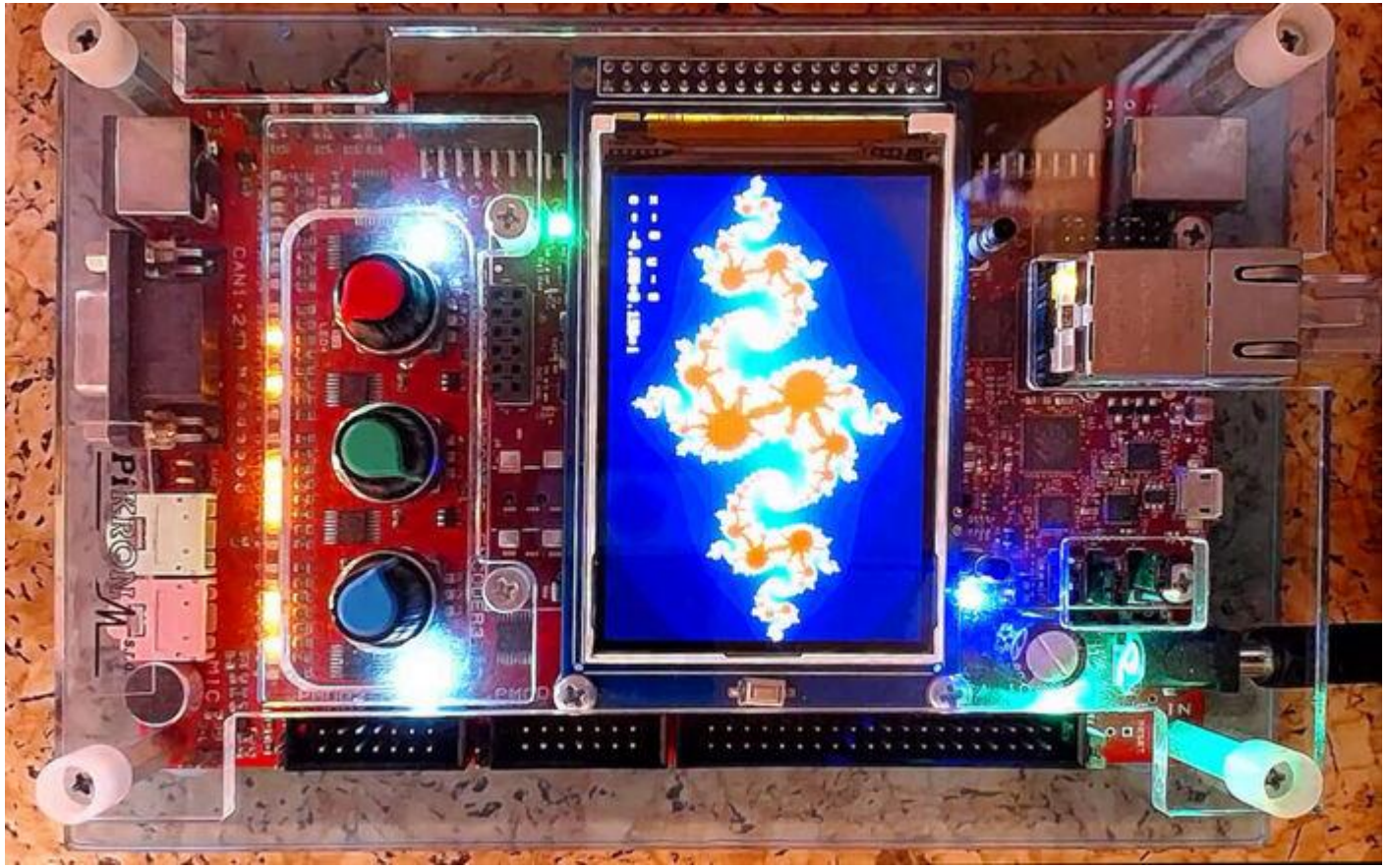




Memory Mappped Peripherals Emulation



Motivation for Peripherals Emulation



https://cw.fel.cvut.cz/b212/courses/b35apo/en/documentation/mz_apo/startv

Command Line Interface

```
>>> qtrvsim_cli --asm program.S --trace-fetch
Fetch: addi x1, x0, 17
Fetch: addi x2, x0, 34
Fetch: addi x3, x0, 51
Fetch: lw x4, 68(x0)
Fetch: addi x5, x4, 85
Fetch: beq x0, x0, 0x22c
Fetch: addi x21, x0, 17
Fetch: addi x22, x0, 34
Fetch: addi x23, x0, 51
Fetch: addi x24, x0, 68
Fetch: addi x25, x0, 85
Machine stopped on BREAK exception.
Fetch: ebreak
```

Implementation Overview

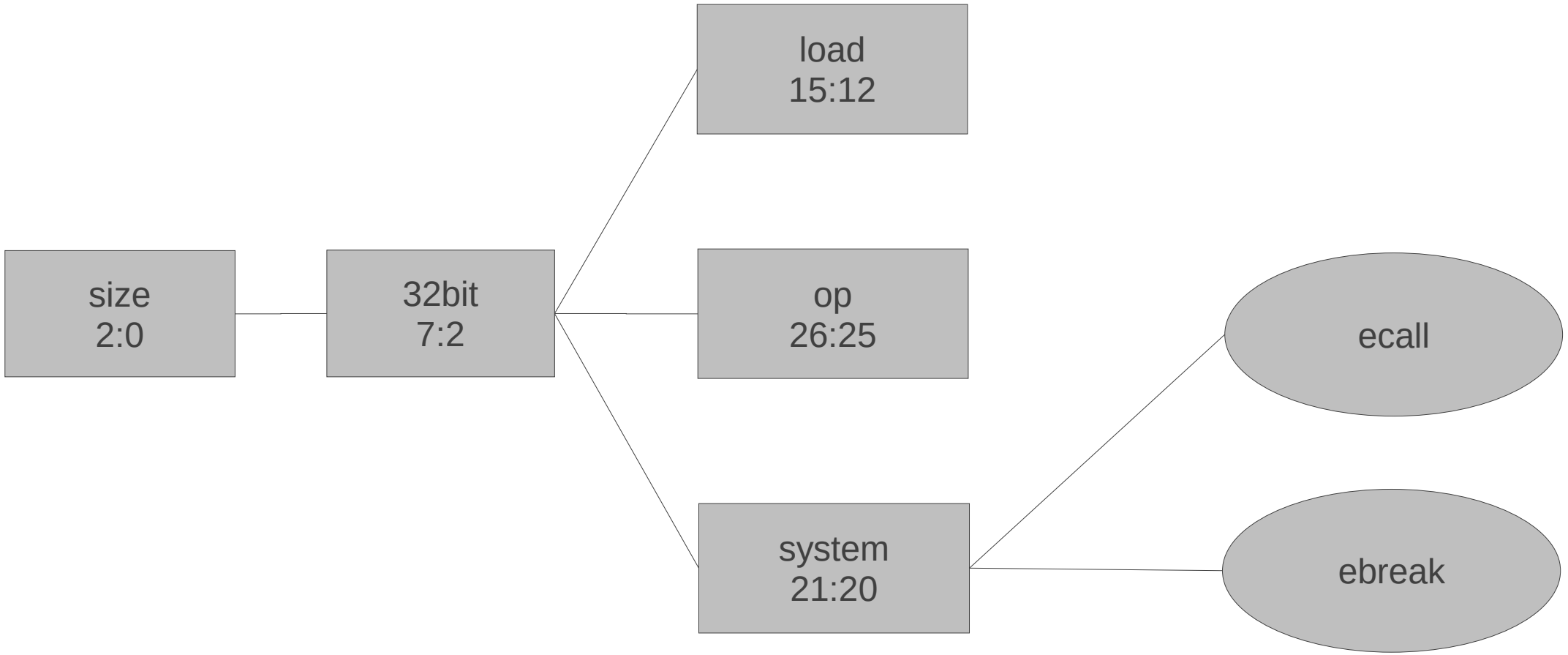
Instruction Decoding

Execution

Memory Model

Core Visualization

Instruction Decoding



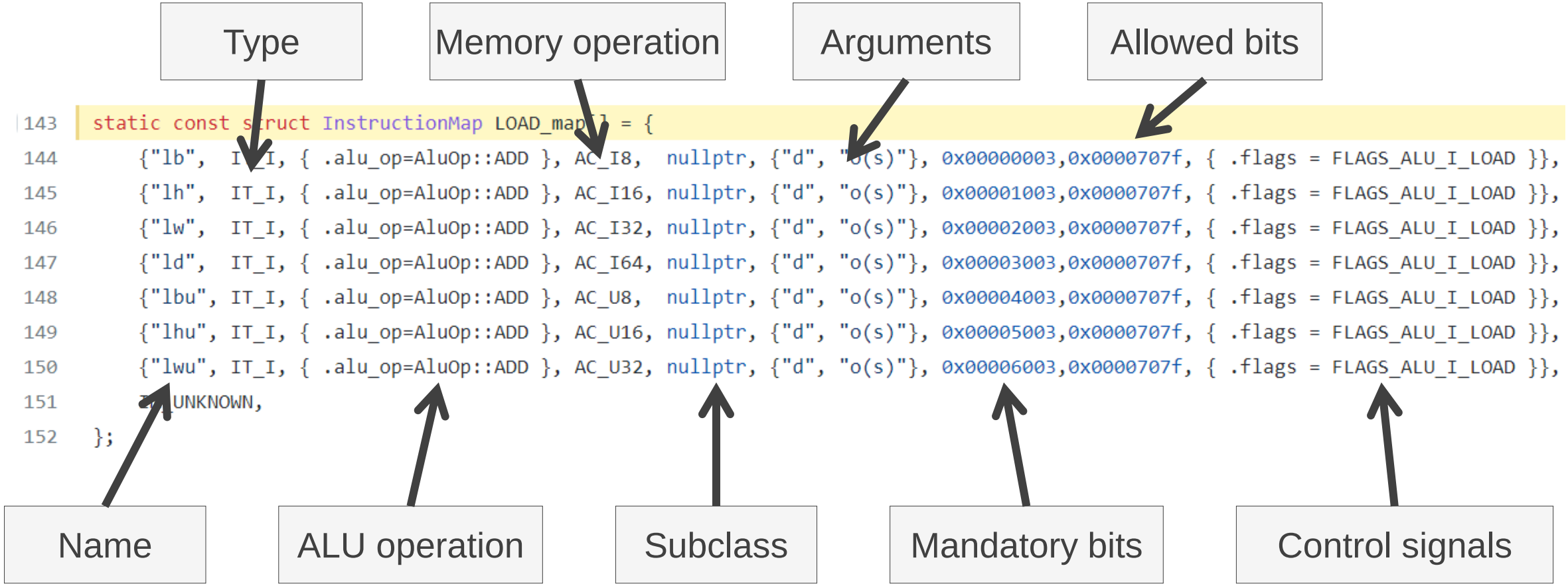
Instruction Decoding - Example

```
327 static const struct InstructionMap I_inst_map[] = {
328     {"load", IT_I, NOALU, NOMEM, LOAD_map, {}, 0x03, 0x7f, { .subfield = {3, 12} }}, // LOAD
329     IM_UNKNOWN, // LOAD-FP
330     IM_UNKNOWN, // custom-0
331     {"misc-mem", IT_I, NOALU, NOMEM, MISC_MEM_map, {}, 0x0f, 0x7f, { .subfield = {3, 12} }}, // MISC-MEM
332     {"op-imm", IT_I, NOALU, NOMEM, OP_IMM_map, {}, 0x13, 0x7f, { .subfield = {3, 12} }}, // OP-IMM
333     {"auipc", IT_U, { .alu_op=AluOp::ADD }, NOMEM, nullptr, {"d", "u"}, 0x17, 0x7f, { .flags = IMF_SUPPORTED | IMF_ALUSRC | IMF
334     {"op-imm-32", IT_I, NOALU, NOMEM, OP_IMM_32_map, {}, 0x1b, 0x7f, { .subfield = {3, 12} }}, // OP-IMM-32    IM_UNKNOWN, // O
335     IM_UNKNOWN, // 48b
336     {"store", IT_I, NOALU, NOMEM, STORE_map, {}, 0x23, 0x7f, { .subfield = {3, 12} }}, // STORE

143 static const struct InstructionMap LOAD_map[] = {
144     {"lb",  IT_I, { .alu_op=AluOp::ADD }, AC_I8,  nullptr, {"d", "o(s)"}, 0x00000003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
145     {"lh",  IT_I, { .alu_op=AluOp::ADD }, AC_I16, nullptr, {"d", "o(s)"}, 0x00001003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
146     {"lw",  IT_I, { .alu_op=AluOp::ADD }, AC_I32, nullptr, {"d", "o(s)"}, 0x00002003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
147     {"ld",  IT_I, { .alu_op=AluOp::ADD }, AC_I64, nullptr, {"d", "o(s)"}, 0x00003003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
148     {"lbu", IT_I, { .alu_op=AluOp::ADD }, AC_U8,  nullptr, {"d", "o(s)"}, 0x00004003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
149     {"lhu", IT_I, { .alu_op=AluOp::ADD }, AC_U16, nullptr, {"d", "o(s)"}, 0x00005003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
150     {"lwu", IT_I, { .alu_op=AluOp::ADD }, AC_U32, nullptr, {"d", "o(s)"}, 0x00006003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
151     IM_UNKNOWN,
152 };
```

Instruction Decoding - Example

```
327 static const struct InstructionMap I_inst_map[] = {
328     {"load", IT_I, NOALU, NOMEM, LOAD_map, {}, 0x03, 0x7f, { .subfield = {3, 12} }}, // LOAD
329     IM_UNKNOWN, // LOAD-FP
330     IM_UNKNOWN, // custom-0
331     {"misc-mem", IT_I, NOALU, NOMEM, MISC_MEM_map, {}, 0x0f, 0x7f, { .subfield = {3, 12} }}, // MISC-MEM
332     {"op-imm", IT_I, NOALU, NOMEM, OP_IMM_map, {}, 0x13, 0x7f, { .subfield = {3, 12} }}, // OP-IMM
333     {"auipc", IT_U, { .alu_op=AluOp::ADD }, NOMEM, nullptr, {"d", "u"}, 0x17, 0x7f, { .flags = IMF_SUPPORTED | IMF_ALUSRC | IMF
334     {"op-imm-32", IT_I, NOALU, NOMEM, OP_IMM_32_map, {}, 0x1b, 0x7f, { .subfield = {3, 12} }}, // OP-IMM-32    IM_UNKNOWN, // OI
335     IM_UNKNOWN, // 48b
336     {"store", IT_I, NOALU, NOMEM, STORE_map, {}, 0x23, 0x7f, { .subfield = {3, 12} }}, // STORE

143 static const struct InstructionMap LOAD_map[] = {
144     {"lb",  IT_I, { .alu_op=AluOp::ADD }, AC_I8,  nullptr, {"d", "o(s)"}, 0x00000003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
145     {"lh",  IT_I, { .alu_op=AluOp::ADD }, AC_I16, nullptr, {"d", "o(s)"}, 0x00001003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
146     {"lw",  IT_I, { .alu_op=AluOp::ADD }, AC_I32, nullptr, {"d", "o(s)"}, 0x00002003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
147     {"ld",  IT_I, { .alu_op=AluOp::ADD }, AC_I64, nullptr, {"d", "o(s)"}, 0x00003003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
148     {"lbu", IT_I, { .alu_op=AluOp::ADD }, AC_U8,  nullptr, {"d", "o(s)"}, 0x00004003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
149     {"lhu", IT_I, { .alu_op=AluOp::ADD }, AC_U16, nullptr, {"d", "o(s)"}, 0x00005003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
150     {"lwu", IT_I, { .alu_op=AluOp::ADD }, AC_U32, nullptr, {"d", "o(s)"}, 0x00006003,0x0000707f, { .flags = FLAGS_ALU_I_LOAD }},
151     IM_UNKNOWN,
152 };
```

Instruction Decoding - Example



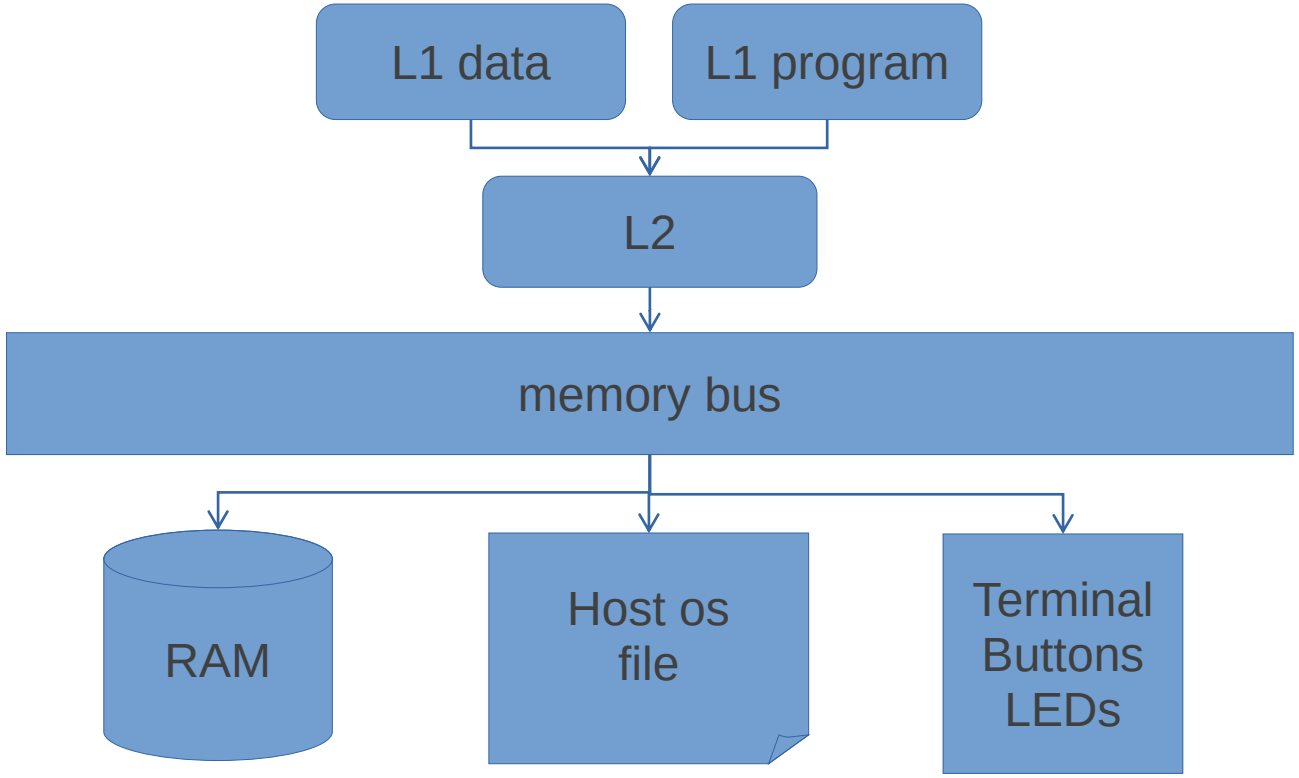
Execution – Pipeline

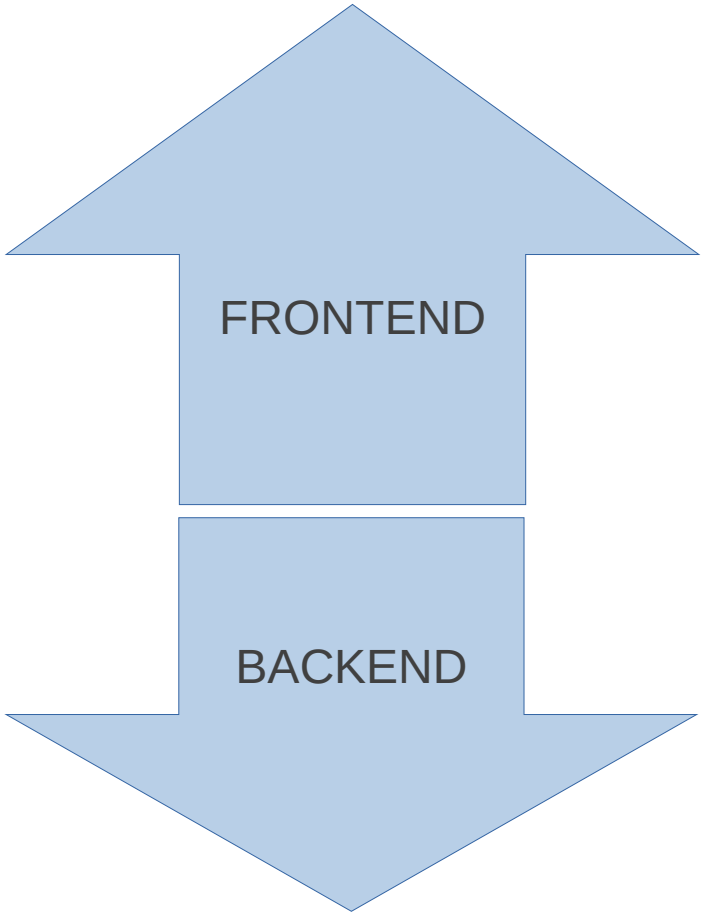
```
484 void CoreSingle::do_step(bool skip_break) {
485     Pipeline &p = state.pipeline;
486
487     p.fetch = fetch(pc_if, skip_break);
488     p.decode = decode(p.fetch.final);
489     p.execute = execute(p.decode.final);
490     p.memory = memory(p.execute.final);
491     p.writeback = writeback(p.memory.final);
492
493     regs->write_pc(mem_wb.computed_next_inst_addr);
494
495     if (mem_wb.excause != EXCAUSE_NONE) {
496         handle_exception(
497             mem_wb.excause, mem_wb.inst, mem_wb.inst_addr, regs->read_pc(), prev_inst_addr,
498             mem_wb.mem_addr);
499         return;
500     }
501     prev_inst_addr = mem_wb.inst_addr;
502 }
```

Execution – Pipeline Step

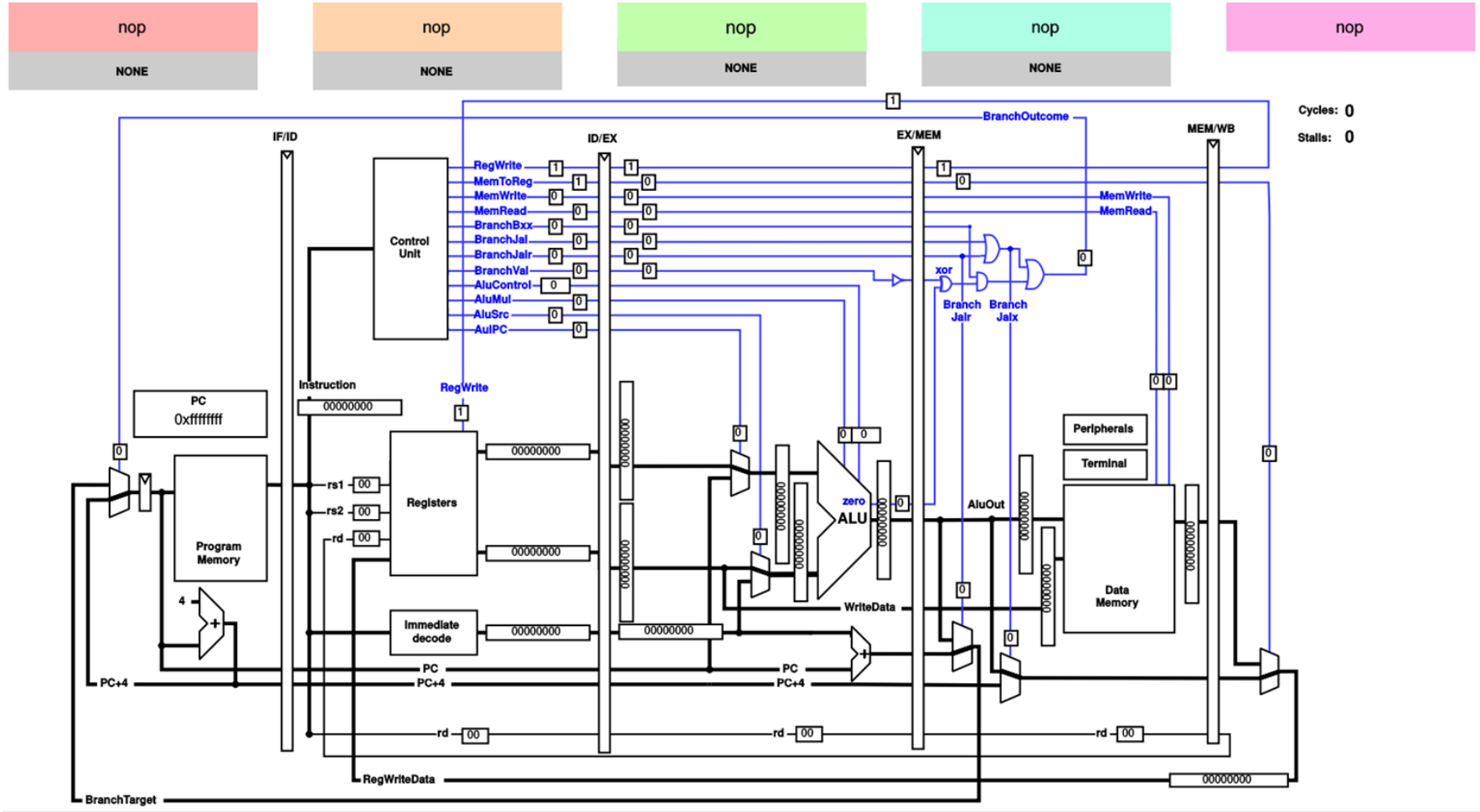
```
192 FetchState Core::fetch(PCInterstage pc, bool skip_break) {
193     if (pc.stop_if) { return {}; }
194
195     const Address inst_addr = Address(regs->read_pc());
196     const Instruction inst(mem_program->read_u32(inst_addr));
197     ExceptionCause excause = EXCAUSE_NONE;
198
199     if (!skip_break && hw_breaks.contains(inst_addr)) { excause = EXCAUSE_HWBREAK; }
200
201     if (control_state != nullptr) { control_state->increment_internal(CSR::Id::MCYCLE, 1); }
202
203     if (control_state != nullptr && excause == EXCAUSE_NONE) {
204         if (control_state->core_interrupt_request()) { excause = EXCAUSE_INT; }
205     }
206
207     return { FetchInternalState { .fetched_value = inst.data() },
208              FetchInterstage {
209                  .inst = inst,
210                  .inst_addr = inst_addr,
211                  .next_inst_addr = inst_addr + inst.size(),
212                  .predicted_next_inst_addr = predictor->predict(inst, inst_addr),
213                  .excause = excause,
214                  .is_valid = true,
215              } };
216 }
```

Memory Model

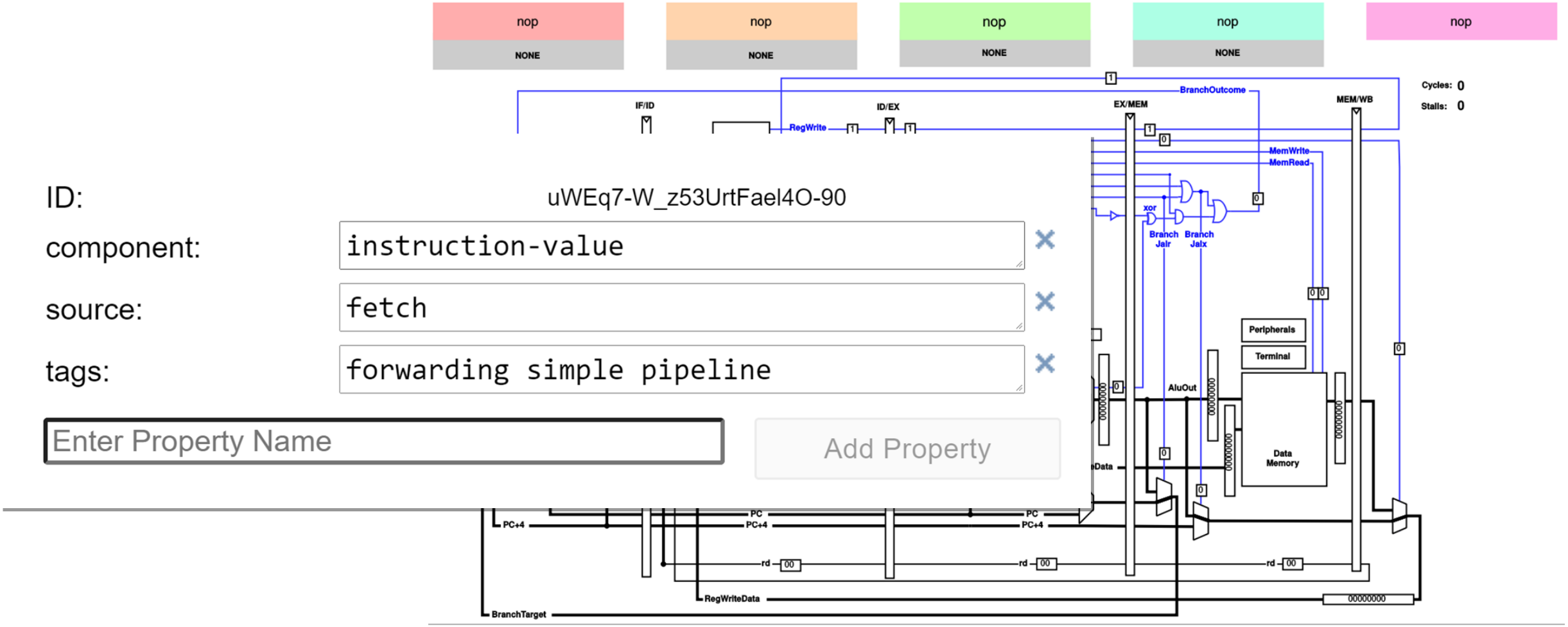




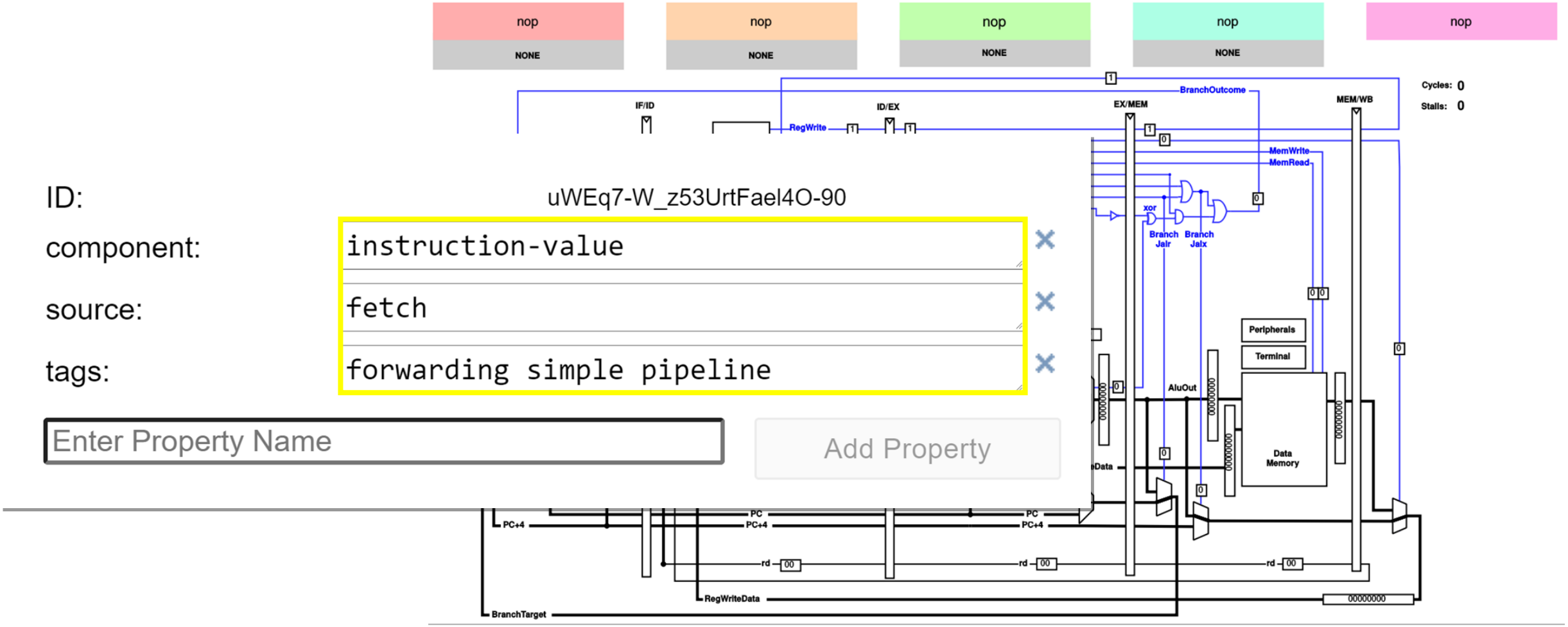
Core View Implementation - TAGS



Core View Implementation - TAGS



Core View Implementation - TAGS



FAQ

- Cycle accuracy

FAQ

- Cycle accuracy
  - YES*

FAQ

- Cycle accuracy
  - YES*
- RISC-V official tests compliant

FAQ

- Cycle accuracy
  - YES*
- RISC-V official tests compliant
  - YES (NEW in 0.9.4), part of the CI

FAQ

- Cycle accuracy
  - YES*
- RISC-V official tests compliant
  - YES (NEW in 0.9.4), part of the CI
- ISA extensions support

FAQ

- **Cycle accuracy**
  - **YES***
- **RISC-V official tests compliant**
  - **YES** (NEW in 0.9.4), part of the CI
- **ISA extensions support**
  - **RV32IM**, **RV64IM** (CLI only), **Zicsr** (NEW in 0.9.4!)

FAQ

- **Cycle accuracy**
  - **YES***
- **RISC-V official tests compliant**
  - **YES** (NEW in 0.9.4), part of the CI
- **ISA extensions support**
  - **RV32IM**, **RV64IM** (CLI only), **Zicsr** (NEW in 0.9.4!)
- Virtual memory / MMU

FAQ

- **Cycle accuracy**
  - **YES***
- **RISC-V official tests compliant**
  - **YES** (NEW in 0.9.4), part of the CI
- **ISA extensions support**
  - **RV32IM**, **RV64IM** (CLI only), **Zicsr** (NEW in 0.9.4!)
- Virtual memory / MMU
  - Not yet

Future Plans & Wishes

- **Compressed ISA** support
  - Visualization
- **Instruction encoding detailed view**
- Run minimal **riscv64-linux-elf**
- **64bit visualization**
- **MMU – virtual memory**
- **Pipeline utilization graph**
- (Limited) **time-travel debugging** (step back)

Calls for Cooperation

- **Teachers, Educational Institutions**
  - Use the simulator
  - Cooperate on open materials

Calls for Cooperation

- **Teachers, Educational Institutions**
  - Use the simulator
  - Cooperate on open materials
- **Students, Developers**
  - Help develop the simulator
  - Great for final thesis

Calls for Cooperation

- **Teachers, Educational Institutions**
  - Use the simulator
  - Cooperate on open materials
- **Students, Developers**
  - Help develop the simulator
  - Great for final thesis
- **Distribution maintainers**
  - Help us with packaging to official repositories

How to Get It?

Project Sources
https://github.com/cvut/qtrvsim

Windows, Linux, Mac
https://github.com/cvut/qtrvsim/releases

Ubuntu
https://launchpad.net/~qtrvsimteam/+archive/ubuntu/ppa

Suse, Fedora and Debian
https://software.opensuse.org/download.html?project=home%3Ajdupak&package=qtrvsim

Online version
https://comparch.edu.cvut.cz/qtrvsim/app

GitHub

Publications

Graphical CPU Simulator with Cache Visualization
Karel Kočí; Diploma Thesis
https://dspace.cvut.cz/bitstream/handle/10467/76764/F3-DP-2018-Koci-Karel-diploma.pdf

Graphical RISC-V Architecture Simulator - Memory Model and Project Management
Jakub Dupák; 2021; Bachelor Thesis
https://dspace.cvut.cz/bitstream/handle/10467/94446/F3-BP-2021-Dupak-Jakub-thesis.pdf

Graphical RISC-V Architecture Simulator - Instructions Decode and Execution and OS Emulation
Max Hollmann; 2021; Bachelor Thesis
https://dspace.cvut.cz/bitstream/handle/10467/96707/F3-BP-2021-Hollmann-Max-thesis.pdf

Dupák, J.; Píša, P.; Štepanovský, M.; Kočí, K.
QtRVSim – RISC-V Simulator for Computer Architectures Classes
In: embedded world Conference 2022. Haar: WEKA FACHMEDIEN GmbH, 2022. p. 775-778. ISBN 978-3-645-50194-1.
https://comparch.edu.cvut.cz/publications/ewC2022-Dupak-Pisa-Stepanovsky-QtRvSim.pdf

Teaching Materials

Faculty of Electrical Engineering

- B35APO - **Computer Architectures**
  - CZ + EN materials and videos
- BE4M35PAP - **Advanced Computer Architectures**
  - CZ materials and videos

Faculty of Information Technology

- BI-APS - **Architectures of Computer Systems**
  - EN materials

https://comparch.edu.cvut.cz/

Thank you

comparch.edu.cvut.cz

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE

FACULTY OF INFORMATION TECHNOLOGY CTU IN PRAGUE

References: RISC-V

RISC-V open source CPU | Deloitte Insights

https://www2.deloitte.com/xe/en/insights/industry/technology/technology-media-and-telecom-predictions/2022/risc-v-open-source-cpu.html

RISC-V | Western Digital

https://www.westerndigital.com/company/innovation/open-source/risc-v

Alibaba Cloud launches RISC-V chip-development platform | TechNode - RISC-V International (riscv.org)

https://riscv.org/news/2022/08/alibaba-cloud-launches-risc-v-chip-development-platform-technode/

NASA Makes RISC-V the Go-to Ecosystem for Future Space Missions – SiFive

https://www.sifive.com/press/nasa-selects-sifive-and-makes-risc-v-the-go-to-ecosystem

MIPS says first RISC-V chips coming in Q4 2022 • The Register

https://www.theregister.com/2022/05/11/mips_riscv_chips/

Google Announces New Open-source OS for RISC-V Chips | Darshil Patel, All About Circuits - RISC-V International (riscv.org)

https://riscv.org/news/2022/10/google-announces-new-open-source-os-for-risc-v-chips-darshil-patel-all-about-circuits/

Intel invests in open-source RISC-V processors, creates billion-dollar fund | ZDNET

https://www.zdnet.com/article/intel-invests-in-open-source-risc-v-processors-with-a-billion-dollars-in-new-chip-foundries/

SiFive Tapes Out First 5nm TSMC RISC-V Chip With 7.2 Gbps HBM3 | Tom's Hardware (tomshardware.com)

https://www.tomshardware.com/news/openfive-tapes-out-5nm-risc-v-soc