> narvie[1]

Harry Sarson, Ryan Voo, Phillip Stanley-Marbell. **The University of Cambridge**.

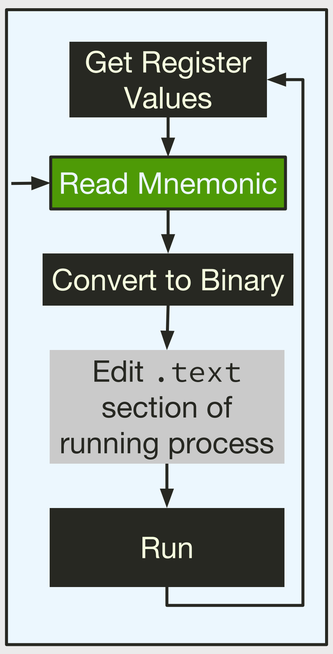
University of Cambridge
Department of Engineering

Physical Computation Laboratory

Abstract

Developing systems software for new hardware architectures is challenging. Developing low-level embedded system firmware or operating system software while evolving a processor architectures is harder still. `narvie` is a Read Evaluate Print Loop (REPL) that provides an interactive interface to an open-source RISC-V (RV32I) processor design[2]. `narvie` allows operating systems researchers and hardware architects to interact with a RISC-V processor implementation running in an iCE40 low-power low-pincount FPGA.

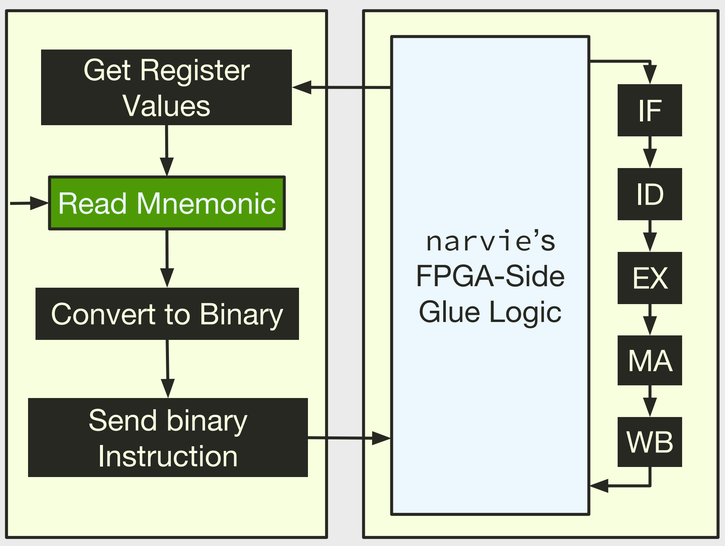
Method



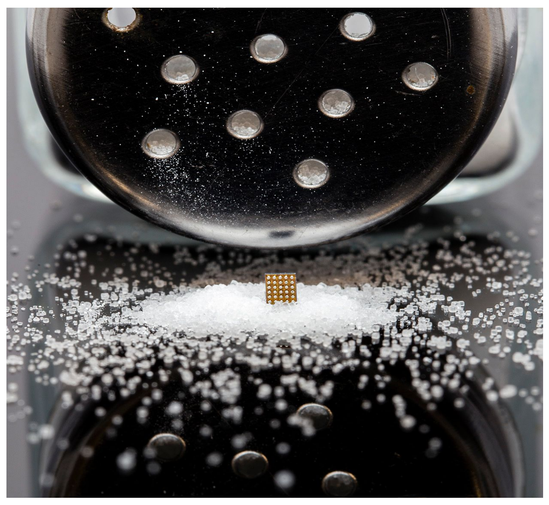
Existing instruction REPLs such as Rappel[3] (linux) and WinRepl[4] (Windows) execute individual x86 instructions by modifying the instructions of a child process in memory.

`narvie` REPL
Interface on host
Execution on FPGA

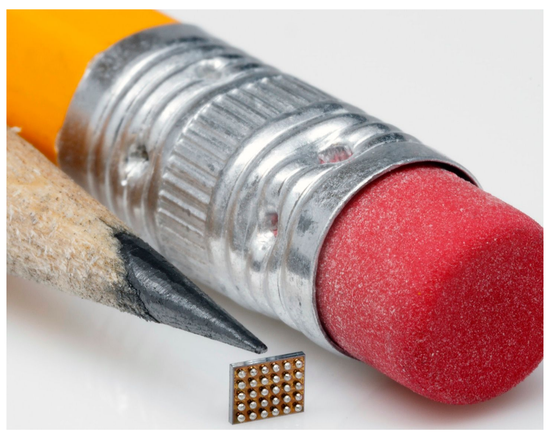
Pipelined RISC-V + `narvie`
On iCE40 FPGA



The processor clock does not run continuously. Only when an instruction is received does the clock start and it stops as soon as the instruction has been fully evaluated. `narive` uses a processor with a five stage pipeline and so five rising edges of the processor clock are required each time.



Source: www.semiconductorstore.com



Source: www.electronicsweekly.com

① `> addi x1, x1, 0x567`

② Instruction: `addi x1,x1,1383` | imm[11:0]: `010101100111` | rs1: `00001` | funct3: `000` | rd: `00001` | opcode: `0010011`

③ `Writing addi x1,x1,1383 to to riscv processor: Success`

④ `Reading updated registers from riscv processor: Success`

Name	ABI	Value
x0	zero	0x00000000
x1	ra	0x01234567
x2	sp	0xFFFFE020
x3	gp	0x00000000
x4	tp	0x00000000
x5	t0	0x00000000
x6	t1	0x00000000
x7	t2	0x00000000

Name	ABI	Value
x16	a6	0x00000000
x17	a7	0x00000000
x18	s2	0x00000000
x19	s3	0x00000000
x20	s4	0x00000000
x21	s5	0x00000000
x22	s6	0x00000000
x23	s7	0x00000000

`narvie` evaluating the right shift operation. The bits in `x1` are shift left by 4 and stored in `x2`.

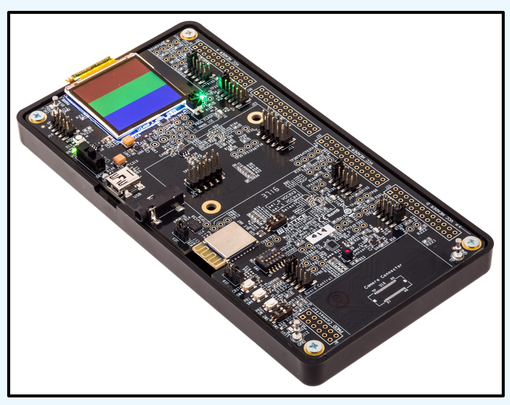
Idea

`narvie` provides a terminal user interface. Once the user enters an RISC-V instruction mnemonic ①, `narvie` converts the input into a binary instruction ② and sends it to the FPGA via a serial port ③. Finally, the registers are read back and displayed to the user ④.

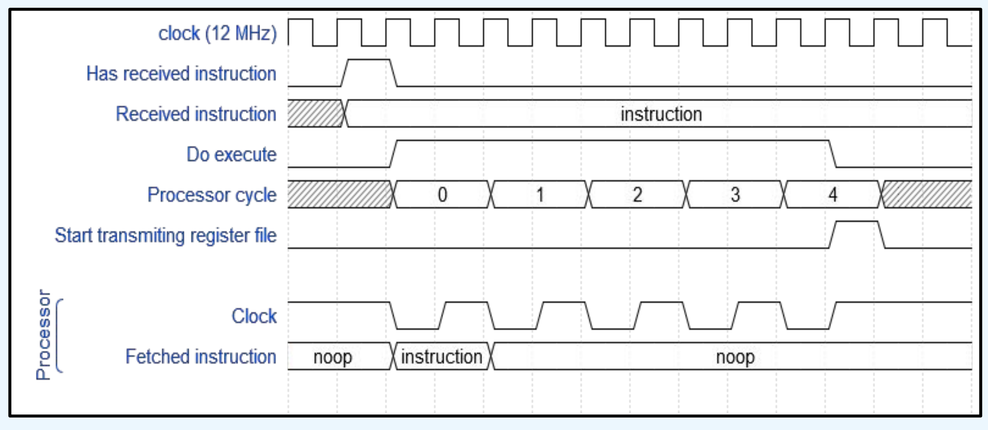
`narvie` exposes the RISC-V instruction set architecture in a very basic form and demonstrate in a concrete way what individual instructions do. By allowing evaluation of individual instructions, narvie will help programmers to gain understanding of RISC-V.

Hardware Implementation

1. Load the instruction from UART.
2. Clock the processor once.
3. Replace the instruction by a no-op.
4. Clock the processor four more times.
5. Send the registers over UART.



Lattice mobile development board containing iCE40 FPGAs that can be programmed and communicated with via USB. Source: www.latticesemi.com



Timing diagram showing the five processor clocks that run to evaluate an instruction. Has received instruction and Received instruction are set by hardware listening to UART.
Source: wavedrom.com

Contributions: `narvie` is a new tool enabling evaluation of individual RISC-V instructions, enable experimentation and exploration.

References:

1. `narvie` is an acronym of native RISC-V instruction evaluator.
2. Andrew Waterman and Yunsup Lee and David Patterson and Krste Asanovic. The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.0, 2014.
3. yrp60. rappel, 2018. URL: https://github.com/yrp604/rappel.
4. zerosum0x0. WinRepl, 2017. URL: https://github.com/zerosum0x0/WinREPL

UNIVERSITY OF CAMBRIDGE