Neosyn · IP Core

DATASHEET · REV A (PRELIMINARY)

RISC-V (RV32IM)

RV32IM single-cycle CPU with a loadable program

• PRODUCTION READY

Product code: NSN-CPU-RV32IM · Implemented in C⏚ · Source-included · Verilog / VHDL output

The Neosyn RISC-V core is a 32-bit processor implementing the RV32I base integer ISA plus the standard M extension (multiply, divide, remainder), with the Zicsr and Zifencei extensions. The datapath is single-cycle: each instruction completes in one clock.

The program is loaded at run time over a count-prefixed boot stream rather than baked into a ROM, so a single datapath runs any program. The core has been exercised against a library of real programs — a self-test, bubble sort, recursive Fibonacci on a stack, memcpy, strlen, Euclid GCD, bitwise CRC-8, multiply/divide and a 3×3 matrix multiply.

Written in readable, extensible C⏚ and shipped with source, it is a clean base for adding custom instructions — and it is the reference CPU that gates every Neosyn compiler release, so it is continuously re-verified.

KEY FEATURES

- RV32I + M extension
- Zicsr & Zifencei
- Single-cycle datapath
- Loadable program
- Sub-word load/store
- Harvard memory
- Readable, extensible C⏚
- Source-included

— Architecture

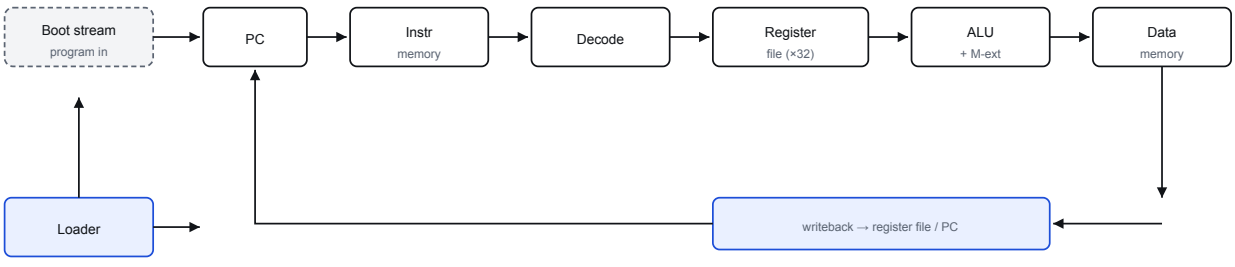

Figure 1. RV32IM single-cycle datapath. The loader fills instruction memory from the boot stream; each cycle the PC fetches an instruction, which is decoded, reads the register file, executes in the ALU (with the M-extension multiplier/divider), optionally accesses data memory, and writes back.

1 Overview

The core implements RV32I plus the M extension and the Zicsr/Zifencei extensions. The single-cycle datapath fetches an instruction from instruction memory, decodes it, reads operands from the 32-entry register file, executes in the ALU (including the M-extension multiply/divide and a barrel shifter), optionally accesses data memory for loads and stores, and writes the result back — all in one clock.

A boot loader fills instruction memory from a count-prefixed stream, so one datapath runs any program. CSR accesses and FENCE.I are handled non-trappingly; ECALL/EBREAK halt the core.

2 Features

- **ISA.** RV32I base integer ISA + M extension (MUL, MULH/MULHSU/MULHU, DIV/DIVU, REM/REMU).
- **Privileged glue.** Zicsr (CSR access) and Zifencei (FENCE.I) — non-trapping.
- **Datapath.** Single-cycle; one instruction per clock. Harvard instruction + data memory.
- **Program loading.** Count-prefixed boot stream — one core runs any program, no ROM rebuild.
- **Memory access.** Sub-word loads and stores (LB/LH/LW, SB/SH/SW).
- **Extensible.** Readable C⏚ source — a clean base to add custom instructions.

3 Functional description

Refer to Figure 1.

3.1 Program load

On reset the core consumes a boot stream: the first word is the program length, followed by that many instruction words, which fill instruction memory. The core then begins fetching from address 0.

3.2 Execution

Each cycle: fetch (PC → instruction memory), decode, register read, execute (ALU, including the M-extension multiplier/divider and barrel shifter), optional data-memory access, and writeback. Branches and jumps redirect the PC within the same cycle.

3.3 System instructions

CSR accesses (Zicsr) and FENCE/FENCE.I (Zifencei) are non-trapping no-ops on this in-order single-cycle core; ECALL/EBREAK halt execution.

> **Note.** A single-cycle core trades maximum clock frequency for simplicity and determinism; it is an ideal control processor and a clean base to extend. It is the reference CPU that gates every Neosyn compiler release.

4 Interfaces & signals

The core loads its program over a stream and exposes a memory-write observation interface.

GROUP	DIRECTION	DESCRIPTION
Program load	in	Count-prefixed instruction boot stream
Store observation	out	Address / data pulse on each store (for test/observation)
Memory	internal	Word-addressed instruction + data RAM
Halt	internal	ECALL / EBREAK halts the core
Clock / reset	in	Core clock domain and synchronous reset

Detailed signal-level pinout (per-bit widths, polarities, timing) is available on request and ships with the core.

5 ISA & standards compliance

ITEM	DETAIL
Base ISA	RISC-V RV32I (Unprivileged ISA)
Extension	"M" — RV32IM (mul/div/rem)
System	Zicsr, Zifencei (non-trapping)
Pipeline	Single-cycle (1 instr/clock)
Memory	Harvard; word-addressed; sub-word LD/ST

6 Performance

PARAMETER	VALUE	NOTES
ISA	RV32IM + Zicsr/Zifencei	
Pipeline	Single-cycle	1 instruction / clock
CPI	1	single-cycle
Max clock (f_{MAX})	Available on request	per target device

7 Resource utilization

Representative utilization per target FPGA family is provided on request from current synthesis reports.

TARGET FAMILY	LUTS / CELLS	REGISTERS	BLOCK RAM	F_{MAX}
Lattice ECP3	on request	on request	on request	on request
AMD/Xilinx 7-series	on request	on request	on request	on request
Intel Cyclone	on request	on request	on request	on request

Figures are supplied per project from characterised synthesis runs to avoid quoting unverified numbers.

8 Verification & validation

- Bottom-up tier gate — 13/13 atoms green on the bytecode simulator and on Verilog (Icarus) elaboration.
- Runs a 10-program library end to end (self-test, sort, recursive Fibonacci, memcpy, strlen, GCD, CRC-8, mul/div, 3×3 matmul, FENCE/CSR).
- Gates every Neosyn compiler release — continuously re-verified.

9 Deliverables

- C⏚ source for the core (readable, modifiable)
- Generated synthesizable Verilog (VHDL on request)
- Self-checking testbench
- Integration guide and this datasheet
- Email integration support per the licensed tier

10 Ordering & licensing

ITEM	DETAIL
Product code	NSN-CPU-RV32IM
License	Single-project or perpetual; full C⏚ source included
Pricing	Quoted per use (project, volume, support tier) — contact Neosyn
Support	Email integration support; custom development available
Contact	neosyn.io/contact · info@neosyn.io

11 Revision history

REV	DATE	CHANGE
A	2026	Preliminary datasheet (Neosyn / C⏚ release).