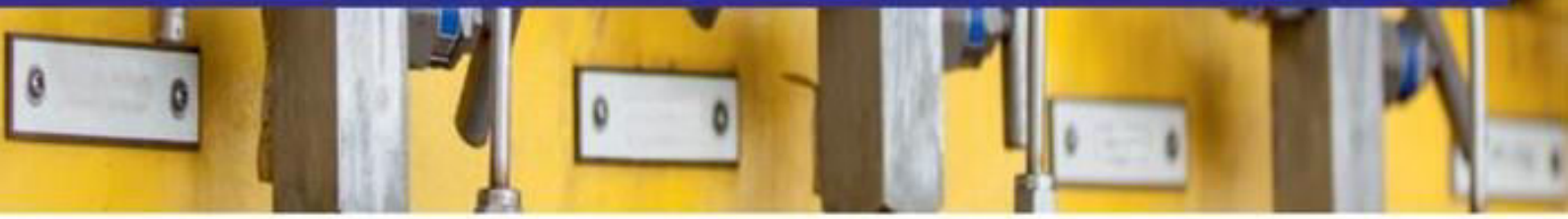
IJAREEIE

e-ISSN: 2278-8875
p-ISSN: 2320-3765

# International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

Volume 15, Issue 7, July 2026



ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 9.867

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (IJAREEIE)

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

# Design and RTL-to-GDSII Implementation of an 32-bit RV32I Single-Cycle RISC-V Processor With SRAM Blocks

**P. R. Sapkale**

Student, Dept. of EXTC, Sardar Patel Institute of Technology, Mumbai, Maharashtra, India

**ABSTRACT**: This study aims to design and implement a 32-bit RISC-V processor from scratch and to end with an entirely open-source flow of Sky130 PDK, to build a specialized architecture without a commercial EDA license. We also intend to make the use of SRAM macros in order to minimize the synthesis process as an alternative of individual registers. The study intends to ensure that the final layout passes all DRC and LVS checks. The processes of layout, antennas, floorplanning, CTS, macro placement, and power delivery are targeted to be manually tuned for the sake of optimization. The final system must also manage to pass the tests premade to test all the instructions within the full fledged RV32I base integer instruction set.

**KEYWORDS:** RISC-V, Processor, Sky130, LibreLane, SRAM, ASIC.

## I. INTRODUCTION

RISC-V has become the open standard that lets anyone build a processor, since being released in 2010, with their well documented and extensive instruction set. It is proven that it's modularity ranges from small ASICs to Industry grad Processors. It was chosen for the purposed of this study for those very reasons. What this study aims at is to design an 32-bit RV32I processor from the ground up with RTL written with Verilog HDL, and for the chipset to go through the entire Physical Design process with specialized parameters. The processor itself is a single-cycle implementation of the RV32I base integer instruction set, to ensure ground level simplicity. The tools we intend to use for the RTL to GDS-II process are Yosys for initial synthesis, OpenROAD for floorplanning to routing, KLayout for physical layout viewing and GDSII export, and the SkyWater Sky130 open PDK as the target process all orchestrated through LibreLane, which is an actively maintained continuation of the OpenLane RTL-to-GDSII flow by the Skywater Group.

We also replaced both memories with built 1-KB SRAM macros with a 1024×32-bit instruction memory and a matching data memory, synthesized with OpenRAM and dropped into the floorplan as hard macros rather than as synthesizeable registers. It is an minor change to describe and yet a significant implementation. The project aims at verification and physical design flow which are significant as a complete, reproducible open-source RTL-to-GDSII flow for a 32-bit RV32I single-cycle core that includes real hard-macro SRAM integration rather than inferred register based memory with a self-checking, instruction-level testbench that exercises all 40 RV32I base-integer instructions and gives real evidence before touching Physical Design.

## II. LITERATURE REVIEW

Building a single-cycle RV32I core from scratch is not exactly new ground, and it also helped to know what others had established in their research. One study designed and rigorously test-verified a single-cycle RISC-V microprocessor on FPGA with absolute implementation on system, while checking it against the official RISC-V compliance suite rather than a set ad hoc tests [1]. Another study worked through a single-cycle RISC-V implementation in Verilog with a close focus on datapath and control-unit design architecture. [2] The studies also took a similar single-cycle RV32I design further into instruction-decoding and ALU-control detail [3]. An implementation analyzed a RISC-V single-cycle core with an specialised work on monitoring and analyzing resource usage with an empasis on how and which parameters affect resource utilization in various ways. [4], The final stufy with single cycle architechture went one step further than most of these by carrying their single-cycle microarchitecture onto an actual FPGA prototype and reporting its operating frequency and power [5] all five collectively help us in confirming that the single-cycle datapath is a settled, explored in depth for the design target we have set rather than as an open research question. What is less settled, however, is what happens after the RTL is done, and an another thread of prior work speaks to that. A documented a comprehensive, fully open-source RTL-to-GDSII workflow for custom embedded FPGA architectures,

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

showing that synthesis through layout generation can be done end to end without a commercial license [6]; A study puts that claim to an direct test, running a synchronous FIFO through both the open-source Qflow and the commercial Cadence Encounter flow to see how the results actually compared on area, power, and turnaround time to find nearly zero difference or to even see that the open-source tools tend to provide a better result in some domains. [7] Jayasurya et al. pushes an open-source RTL-to-GDSII flow to further rigorous standards still while tuning it specifically for area and power on a telecommunications based receiver core[8]. On the other, a little closer to our project we find that Rao et al. and Kirali and Fidan each designed and implemented their own 32-bit RISC-V processors on FPGA and reported the resulting resource utilization [9], [10], giving us a rough utilization baseline to compare against. The RV32I single-cycle core itself is a mature, well-understood target, however research in the perspective of whether RISC-V architechture is mature enough to pass ASIC verification, is an aspect this study aims to find.

## III. RESEARCH METHODOLOGY

We start this study with the datapath, building it block by block rather than writing the core blocks as one monolithic module. Each piece is built one by one based on their functionality such as program counter, instruction memory, register file, immediate generator, ALU, control unit, branch logic and finally the data memory. Once all the separate blocks are built, we move to building the edge based modules such as the adders on the side which handle the incrementation of the program counter and other applications. These are all then connected together finally in the top level module. Finally when all the pieces are wired together, the processor covers the full RV32I base integer instruction set which includes types of instructions ranging from R-type which include basic arithmetic and logical instructions like add, sub, sll, slt, sltu, xor, srl, sra, or, and. I-type or immediate instructions which are specifically handled by the immediate generation block such as addi, slli, slti, sltiu, xori, srli, srai, ori, andi instructions. Load/Store memory instructions which are essential in navigation and data handling in the system of the core include lw, lh, lhu, lb, lbu, sw, sh, sb, and down to branch instructions such as beq, bne, blt, bge, bltu, bgeu with jump instructions like jal, jalr and finally upper-immediate instructions which are lui, auipc, used in special cases and are an essential part of any single core system. The register file contains 32 register of 32-bit width, with specifically the x0 register being hardwired to zero, the ALU functions with a secondary ALU-control decoder to interpret funct3/funct7 as the control unit would need a two way bridge instead if these were controlled through the control unit. The reusable adders handle both PC+4 and branch-target arithmetic, with multiplexers routing the writeback and operand paths. Because of the single-cycle design, every block finishes it's work within one cycle and only the architectural state inclsive of the PC, registers, memory updates on the clock edge which is exactly what makes the block diagram in Fig. 1 as a single pass through the hardware rather than a multi-stage pipeline.

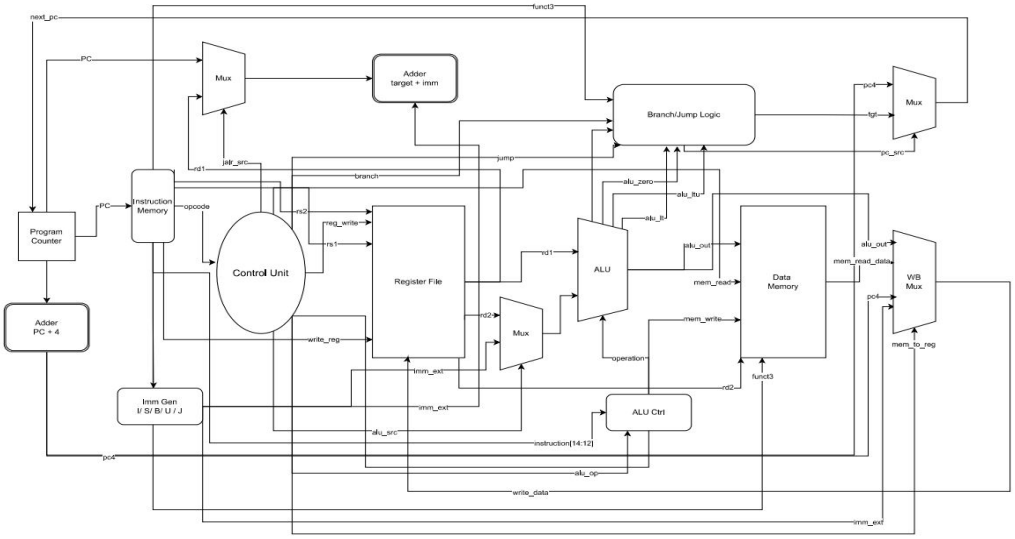



Fig. 1 Block diagram of the RV32I single-cycle datapath and control path, showing the program counter, instruction and data memory interfaces, register file, ALU, immediate generation, and branch/jump logic.

Once the datapath was verified, we swapped the behaviorally inferred instruction and data memories for two dedicated SRAM macros a 1-KB instruction memory and a 1-KB data memory, each organized as a 1024×32-bit words with one

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

read/write and one read port generated with the open-source tool of OpenRAM and dropped into the design in the form of hard macros instead of synthesized flip-flop arrays. This achieves two things, firstly it bounds the instructions and data address space to a clean 1024-word limit, and it gives us the downstream physical design flow with something to place and route that can function as a memory unit in a final fabricated chip, rather than a few thousand flip-flops standing in for one eating away at the verification times. With the RTL ready and the memories added in, the next step becomes to test the functionality of the current design, we self built a testbench to test all 40 instructions of the RV32I base integer ISA, organized into twelve test groups: R-type ALU operations (ADD, SUB, AND, OR, XOR, SLL, SRL, SRA, SLT, SLTU), I-type ALU operations (ADDI, SLTI, SLTIU, XORI, ORI, ANDI, SLLI, SRLI, SRAI), loads (LW, LH, LB, LHU, LBU), stores (SW, SH, SB), taken and not-taken conditional branches (BEQ, BNE, BLT, BGE, BLTU, BGEU), upper-immediate instructions (LUI, AUIPC), unconditional jumps (JAL, JALR), the NOP-equivalent system instructions (FENCE, ECALL, EBREAK), with store-to-load round-trip check, and confirmation that register x0 remains hardwired to zero under a specified direct write attempt. Each test case assembles hand-encoded RV32I machine words directly into instruction memory via hierarchical path access, resets the core, runs a fixed number of clock cycles to let the pipeline-free datapath and final register-file write-back settle, and self-checks the resulting register-file and data-memory contents against expected values, with a watchdog timeout guarding against a stuck program counter.

The payoff is unambiguous, 49 directed test cases spanning every instruction format (R, I, S, B, U, and J), and every single one passed, with zero failures (as observed in Fig. 2).

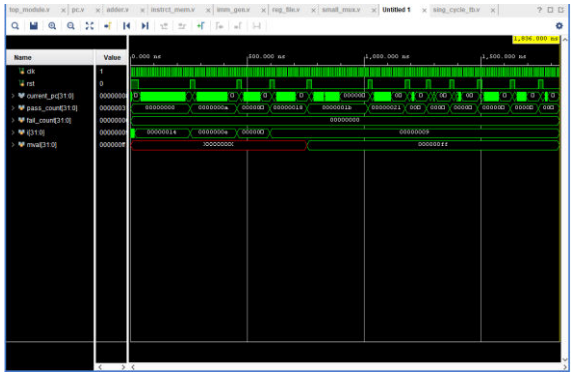


Fig 2. Representative simulation waveform from the functional-verification testbench, showing the program counter, pass/fail counters, and instruction and result values advancing across the directed test cases.

Passing a testbench in simulation is one kind of confidence; having a real toolchain place and route the design onto real hardware is another, and we wanted both before committing to the much longer ASIC flow. So alongside RTL simulation, we synthesized and implemented the same design for a Xilinx Artix-7 FPGA (part family XC7A35T, 256-pin package, speed grade -1) using Vivado, targeting a 12 ns clock period, and pulled the resource utilization and power numbers straight from Vivado's implemented-design reports (Fig. 3).

The numbers were reassuring: 744 look-up tables (LUTs) out of 20,800 available (3.58% utilization), 512 LUTs configured as distributed RAM (LUTRAM) out of 9,600 available (5.33%), 166 flip-flops (FFs) out of 41,600 available (0.40%), 98 of 170 available I/O pins (57.65%), and 1 of 32 available global clock buffers (BUFG, 3.13%). Total on-chip power came in at 0.137 W at a junction temperature of 25.7 °C, with a thermal margin of 59.3 °C (12.2 W) and a medium confidence rating from Vivado. That gave the confirmation, the RTL was synthesizable, and that we could turn to the Sky130 ASIC flow that occupies the rest of this section.

Fig 3. Vivado post-implementation resource utilization and on-chip power summary for the Xilinx Artix-7 (XC7A35T, 256-pin package, speed grade -1) FPGA target.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

PWith the RTL verified twice over in simulation and on an FPGA it was time to take the design somewhere neither of those had: an actual manufacturable layout. For that, we used LibreLane, the actively maintained continuation of the OpenLane RTL-to-GDSII flow, which orchestrates Yosys for synthesis, OpenROAD for floorplanning, placement, clock tree synthesis, and routing, and (optionally) Magic for layout verification and stream-out, all targeting the SkyWater Sky130 open-source process design kit. Sky130 is a 130 nm mixed-signal/RF process for which SkyWater released a complete, foundry-qualified open PDK standard-cell libraries, device models, and design rules which is why it has become the de facto reference node for open-source ASIC work, and why it was the natural choice here. One detail of how we ran this is worth flagging up front rather than burying later: instead of the officially recommended Ubuntu-based environment, we deployed the flow on Ultramarine Linux, a Red Hat/Fedora-derived distribution, using Nix as the package and environment manager for LibreLane and its dependencies. We made that choice deliberately, and it came with a real consequence that shows up later in this paper a Magic sign-off run that first had to be debugged before it could be trusted so it is flagged here rather than glossed over. With the configuration decided, we let LibreLane run the flow stage by stage, in the order the tools expect. Logic synthesis came first: Yosys technology-mapped the behavioral RTL to Sky130 standard cells, with ABC handling boolean optimization to hit the clock-period constraint while keeping gate count down. Next came floorplanning and power-grid generation, where we fixed the die and core boundaries, placed the two SRAM macros at the floorplan edges, and built a VDD/GND metal mesh across the upper metal layers. Global and detailed placement followed: RePlace ran wirelength-driven global placement, and OpenDP legalized every cell's position onto the manufacturing grid without overlaps. Once cells were placed, TritonCTS built a buffered H-tree clock network from the clk root out to every sequential sink, using sky130_fd_sc_hd clock-buffer cells to keep skew and slew under control. Routing came next in two passes FastRoute for global routing, estimating congestion and laying out routing guides across metal layers met1 through met5, and TritonRoute for detailed routing, which performs the actual DRC-clean wiring and via insertion. From there the flow moved into signoff analysis: static timing analysis across process/voltage/temperature corners, and IR-drop analysis on the VPWR/VGND power nets, both discussed in the Results section below. The very last steps Magic-based DRC/LVS sign-off and antenna checking are where the one real snag in this whole project showed up: an initial pass reported 5.6 million DRC violations that turned out to be false positives from evaluating the raw GDSII stream, plus a genuine antenna violation on one over-length net.

## IV. RESULTS

SQuantitative results below are drawn directly from the LibreLane run logs and reports (synthesis, clock tree synthesis, detailed placement, global routing, static timing summary, and IR-drop analysis).

### A. Synthesis and Cell Area

Post-synthesis technology mapping produced 1,055 flip-flops in total: 1,024 clock-enabled scan D-flip-flops (mapped to sky130_fd_sc_hd__dfxtp_2-class cells) implementing the datapath's register file and pipeline-adjacent state, plus a small number of reset- and set-controlled flip-flops for control state. The synthesized top-level sequential logic reported a chip area of approximately 22,440 µm² for the register/flip-flop content alone, growing to approximately 65,239 µm² once the surrounding combinational logic for module top_module was included.

### B. Post-Placement and Post-CTS Cell Inventory

The floorplan that ultimately signed off was not the first one we tried. Two earlier iterations a larger die with the macros placed centrally, and a version with the macros close together but only a single-width keep-out halo both routed but failed downstream DRC, for reasons discussed in Section V. The floorplan that worked stacked both SRAM macros along the same left-hand die edge, doubled their keep-out halo to 40 µm, and let VDD/VSS straps run around the die perimeter instead of through the macro interior. That configuration settled at a 1,200×1,200 µm die (1,194.16×1,188.64 µm core), holding 5,095 placed instances the two SRAM macros plus standard-cell logic at an effective core utilization of 31.9%. That deliberately low utilization is the direct legacy of the routing failures in Section V: it is the headroom the tighter, centrally-placed floorplan did not have.

### C. Timing Summary Across PVT Corners

Static timing analysis was re-run on the final, sign-off floorplan across three representative process/voltage/temperature (PVT) corners typical, fast, and worst-case slow. Table I reports results in nanoseconds.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

I. TIMING SUMMARY ACROSS PVT CORNERS

| Corner | Hold WNS (ns) | Setup WNS (ns) | Setup TNS (ns) | Setup Viol. |
|---|---|---|---|---|
| nom_tt_025C_1v80 (typical) | 0.4720 | 5.9139 | 0.0000 | 0 |
| min_ff_n40C_1v95 (fast) | 0.2038 | 10.8402 | 0.0000 | 0 |
| max_ss_100C_1v60 (slow) | 1.2315 | -7.1070 | -70.2233 | 28 |

Hold timing is met with healthy positive slack in every corner tested. Setup timing is comfortably met at the typical corner (5.9139 ns WNS) and even more so at the fast corner (10.8402 ns WNS), confirming the tightened 40 ns clock period closes cleanly under nominal and best-case conditions. The worst-case slow corner (max_ss_100C_1v60) is a different story: setup slack goes negative there, with 28 failing paths and -70.2233 ns of total negative slack. That the slow corner is the one that fails is exactly consistent with Section I's framing and is not a new finding. This summary also reports two categories the earlier iteration of this flow did not surface: max slew violations (467 typical, 84 fast, 3,636 slow) and max cap violations (12 typical, 10 fast, 36 slow), both concentrated at the slow corner alongside the setup failures, which points to the same long combinational datapath rather than a separate problem.

D. Routing and Congestion

Detailed routing on the final, edge-placed floorplan completed with zero unresolved routing violations. Total routed wirelength came to 486,768 µm across the metal stack 187,192 µm on met1, 185,915 µm on met2, 90,823 µm on met3, 19,085 µm on met4, and 3,751 µm on met5 using 67,994 vias, concentrated on the lower layers (35,147 met1 vias, 27,256 li1 vias, 4,653 met2, 808 met3, 130 met4). That is roughly a third less wirelength than the earlier, larger floorplan needed, a direct consequence of the tighter 1,200×1,200 µm die: less area to cross means shorter nets, even with less routing headroom to spare.

E. Power Delivery and IR Drop

Power analysis at the nominal typical-typical, 25 °C, 1.8 V corner reported total power of 3.226 mW higher than the 1.96 mW reported for the earlier, larger floorplan, consistent with the tighter 1,200×1,200 µm die packing the same logic and switching activity into a smaller area. By mechanism, internal power dominates at 2.568 mW (79.6%), switching power contributes 0.638 mW (19.8%), and leakage is negligible at 0.020 mW (0.6%). By component, sequential logic draws the largest share at 1.102 mW (34.2%), followed closely by the two SRAM macros at 0.978 mW (30.3%), the clock network at 0.866 mW (26.8%), and combinational logic at just 0.280 mW (8.7%) a breakdown that makes the clock tree's cost legible for the first time in this project and is a natural target for the pipelined successor. IR drop remains comfortably within margin despite the higher power: worst-case drop on vccd1 (VDD) is $2.66\times10^{-4}$ V (0.01% of supply), and on vssd1 (GND) is $2.86\times10^{-4}$ V (0.02%), both roughly two orders of magnitude below typical 5–10% sign-off thresholds and confirming the perimeter-routed power mesh adequately supplies both the standard-cell fabric and the two edge-placed SRAM macros.

F. Sign-off Checks: DRC and Antenna

Design-rule checking was performed at two independent levels. TritonRoute's detailed routing stage reported no unresolved DRC violations, and the run's final routing report confirms zero routing violations outright. KLayout's geometric DRC pass, run independently against the streamed-out GDSII, also reported zero fatal errors that clean result is what exposed the real problem described in Section V: an initial Magic DRC pass against the same design reported 5.6 million violations, a number so far out of line with KLayout's zero that it was clearly not a real geometry problem. Reconfiguring Magic to extract and check against the design's native view rather than the raw GDSII stream (MAGIC_DRC_USE_GDS and MAGIC_EXT_USE_GDS both set false) eliminated the false positives outright. What remained after that fix, and after two rounds of floorplan and macro-halo tuning, was 795 Magic-reported "macro override" warnings all traceable to the same cause: Magic's connectivity check treats the deliberate 40 µm signal-free keep-out halo around each SRAM macro as a boundary condition and flags it every time a routed wire terminates at that edge, regardless of whether anything is actually wrong. That is a known, documented, and expected warning class rather than an unresolved defect, and KLayout's independent zero-violation result confirms the layout is geometrically clean. Antenna-rule checking, run with OpenROAD's antenna-repair step (GRT_REPAIR_ANTENNAS) enabled,

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

passed cleanly: zero net violations, zero pin violations, and a single diode inserted to resolve the one genuine antenna condition flagged earlier in the flow, down from an initial violation on a 486 µm net that exceeded the process's 400 µm antenna-length guideline.

Layout-versus-schematic (LVS) verification, unavailable in the earlier iteration of this flow for the reasons discussed in Section V, was completed in this run once Magic's extraction was fixed. The Magic-extracted layout was compared against the synthesized netlist with Netgen and passed cleanly: circuits match uniquely, with the layout and schematic device classes for top_module confirmed equivalent. Combined with the DRC and antenna results above, this paper now reports a fully sign-off-checked layout DRC-clean, antenna-clean, LVS-verified, and timing-characterized closing the one gap the previous iteration of this project had left open.

G. Layout Composition

The final layout (visualized in KLayout, Fig. 4) shows the two SRAM macros instruction memory (insmem.inst_ram_block, placed at (15 µm, 50 µm)) and data memory (datamem.data_ram_block, placed at (15 µm, 650 µm)) both un-rotated (orientation N) and stacked one above the other along the left-hand edge of the 1,200×1,200 µm die, each surrounded by a doubled, 40 µm signal-free keep-out halo. This is a deliberate change from the earlier floorplan, where the macros sat side by side nearer the die interior: pushing both to one edge frees the rest of the die for standard-cell placement and lets the VDD/VSS straps run cleanly around the perimeter rather than through the macro region. Dense met2/met3 routing is visible to the right of the macros, corresponding to the address, data, and control interconnect running between the memories and the surrounding datapath logic.

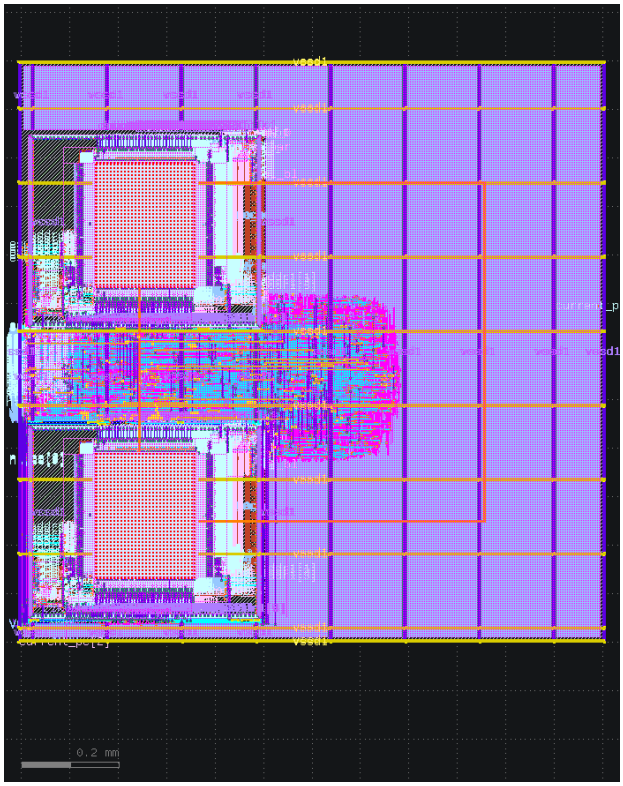



Fig 4. Full-chip GDSII layout in KLayout, showing the full routing stack, the perimeter-routed VDD/VSS mesh, and the two SRAM macros stacked along the left edge of the 1,200×1,200 µm die.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

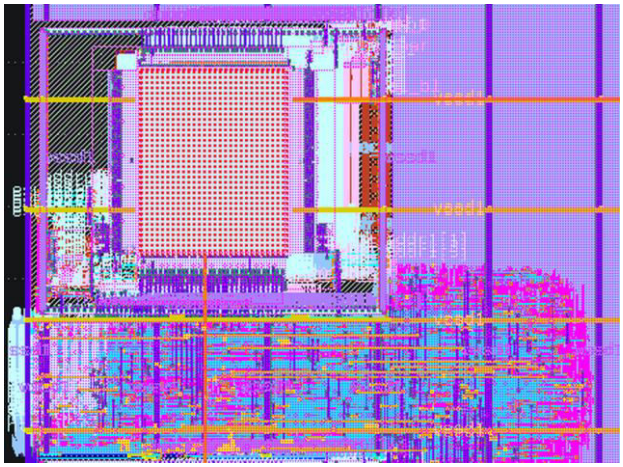


Fig 5. Zoomed-in detail of the instruction-memory SRAM macro, its doubled keep-out halo, and the dense met2/met3 routing congestion in the region between the two macros.

Stepping back from the individual numbers, the overall picture is a design that is physically realizable and, for the first time in this project, fully sign-off checked: placement is legal, the final route reports zero unresolved violations, LVS confirms the layout matches the schematic, and IR drop on both power rails sits roughly two orders of magnitude below typical signoff thresholds. The one place that still needs real attention is setup timing under worst-case slow-process conditions, where the summary shows 28 failing paths and roughly -70.22 ns of total negative slack. That result did not surprise us  it is exactly what we expected from Section I's framing: a single-cycle datapath has an inherently long combinational path running from instruction fetch through the ALU and back to the register file, and that path is what gives out under the pessimistic ss_100C_1v60 corner. Because hold timing is met everywhere with healthy margin and setup is comfortably met at typical and fast corners, this remains a frequency ceiling rather than a functional-correctness risk, and it is the clearest, most concrete argument this project could hand to its own pipelined successor: relax the clock period, shift SYNTH_STRATEGY toward delay optimization, or better yet, break the critical path across pipeline stages, which is exactly the plan for what comes next.

The decision to use hard SRAM macros instead of inferred memory, made back in Section III-A, still shows up clearly here, though its footprint moved with the floorplan. The two macros anchor the routing congestion pattern visible in Fig. 5, where met2/met3 usage clusters in the region between them rather than spreading uniformly across the die. Their final placement  stacked along the die's left edge rather than side by side nearer the interior, each behind a doubled 40 µm halo  is what let the design reach a 31.9% effective core utilization at a smaller 1,200×1,200 µm die than the original 1,500×1,500 µm floorplan: with both macros out of the central routing region and given real breathing room, the router had space to work even at a tighter overall footprint, which is the opposite of what the first, cramped attempt at this floorplan achieved (Section V).

H. Comparison with Prior RV32I Implementations

Table II places this design's key metrics alongside the two most directly comparable prior single-cycle RV32I implementations from Section II, refs. [1] and [5]. Both prior works are FPGA prototypes reporting an achieved maximum clock frequency, whereas this work is an ASIC static-timing result at a fixed clock-period constraint; the two are not strictly equivalent metrics (FPGA fMAX reflects routed programmable-fabric delay, while the ASIC figures reflect standard-cell timing at a fixed target period across PVT corners), so the comparison below is intended to situate this design relative to prior work rather than to claim a like-for-like speed advantage. Ref. [9], also cited in Section II for its RTL-to-GDSII methodology, is omitted from the numeric comparison because it implements a telecommunication receiver core rather than a RISC-V processor and is not an architectural comparison point.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

II. COMPARISON WITH PRIOR SINGLE-CYCLE RV32I IMPLEMENTATIONS

| Reference | Platform | Clock / Power | Functional Verification |
|---|---|---|---|
| [1] La Gala et al., 2024 | Xilinx Spartan-7 FPGA | 12 MHz target; power not reported | Passed full official RISC-V integer test suite |
| [5] Dennis et al., 2017 | Xilinx Spartan 3E FPGA | 32 MHz max; 7.9 mW | Not reported |
| This work | Sky130 130 nm ASIC | 25 MHz (40 ns) target, met at nominal/fast, unmet at worst-case slow corner; 3.226 mW nominal | 49/49 directed RV32I tests passed (all 40 instructions) |

Despite the FPGA/ASIC metric mismatch, two points are worth drawing out. First, this work is the only one of the three to report a physical-design-stage area and IR-drop characterization alongside timing, since refs. [1] and [5] stop at FPGA place-and-route. Second, this work is the only one of the three to report hard-macro SRAM integration rather than treating memory as an inferred or platform-provided (BRAM) resource, which is what drives its comparatively large post-placement area footprint (Section IV-B) and is a direct consequence of targeting a manufacturable ASIC flow rather than an FPGA prototype.

## V. IMPLICATIONS

Every real project runs into at least one thing that does not go according to plan, and this one is worth telling straight rather than smoothing over. We deliberately ran the entire physical design flow on Ultramarine Linux, a Red Hat/Fedora-derived distribution, using Nix to manage dependencies, rather than the Ubuntu-based environment LibreLane officially targets. That choice did not stop the flow outright, but it did make Magic's sign-off checks behave in a way that took real effort to track down. The first full run reported the design as unmanufacturable: 5.6 million Magic DRC violations against roughly zero from KLayout's independent check, plus one genuine antenna violation on a 486 µm net that exceeded the process's 400 µm guideline. The scale mismatch between the two DRC engines was the tell  a real geometry problem does not produce a 5.6-million-to-zero split  and the root cause turned out to be that Magic was evaluating the raw, streamed-out GDSII rather than an extracted view of the design, which it treats far more conservatively. Pointing Magic's DRC and extraction at the native design view instead fixed the false positives outright. That still left real, if smaller, problems to chase: an initial tight floorplan crammed routing against the SRAM macro edges and left standard cells too spread out elsewhere, which briefly reintroduced both antenna and DRC failures (down to roughly 2,700, then 900, reported violations across successive iterations) before we identified the actual cause as insufficient macro-to-macro and macro-to-die-edge spacing. Shrinking the die to 1,200×1,200 µm, doubling the macro keep-out halo to 40 µm, enabling OpenROAD's antenna repair, and finally relocating both SRAM macros to stack along a single die edge  freeing the interior for routing and letting the power mesh run around the perimeter brought the design to a clean sign-off: 795 residual Magic warnings, all attributable to the same deliberate 40 µm halo Magic flags unconditionally wherever a wire terminates at it, and a passing LVS run against Netgen. The most concrete, reusable lesson here is less about the Nix/Ultramarine environment specifically and more general: a DRC engine that disagrees with an independent checker by six orders of magnitude is telling you something about its inputs, not about your layout, and it is worth chasing that disagreement down rather than working around the tool that raised it.

The other limitation is one we already went looking for and found on purpose: the single-cycle datapath's long combinational path is what fails setup timing at the worst-case slow-process corner  28 failing paths and roughly -70.22

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

ns of total negative slack at the 40 ns clock period and we knew going in that this was the price of keeping the first version of this core simple. It is not a defect so much as a known boundary of the design we chose to build, and it points directly at the fix breaking that path into pipeline stages rather than at anything that needs debugging. Beyond that, this is also a single-author project without an independent team re-running the flow on a second machine, so the repository is public precisely so that claim can be checked rather than taken on faith.

None of these are surprises we stumbled into; they are the exact kind of friction the literature in Section II led us to expect from an open-source RTL-to-GDSII flow, and knowing where they were let us report them honestly instead of designing around them quietly. If anything, they sharpen what this project set out to show: that the open-source path from RTL to physical layout works, that it can be pushed hard enough to expose its own rough edges, and that those edges are specific and fixable rather than fundamental.

## VI. CONCLUSION

We set out to see whether a RISC-V processor could go the whole distance from a blank Verilog file to a physically realistic, fully sign-off-checked layout using nothing but open-source tools, and it did. We built a 32-bit RV32I single-cycle core one block at a time, proved it correct against 49 passing directed tests before synthesis ever touched it, and then made it more honest by swapping inferred memory for real OpenRAM SRAM macros. We double-checked that the RTL was genuinely implementable by putting it on a Xilinx Artix-7 FPGA before committing to the longer road, and then carried it through a complete LibreLane physical design flow on the Sky130 node our own floorplanning, our own macro placement, our own clock tree, our own routing and signoff configuration, not defaults borrowed from someone else's project. What came out the other end is routed with zero unresolved violations, DRC-clean, antenna-clean, LVS-verified against the synthesized netlist, comfortably within IR-drop margins, and precisely characterized on timing: hold met everywhere, setup met at nominal and fast corners, and a clearly localized ceiling at the worst-case slow corner that tells us exactly what a pipelined version of this core needs to fix. Along the way we hit a real sign-off snag when Magic's DRC engine first reported 5.6 million violations against a design KLayout scored as clean; tracking that down to a raw-GDSII evaluation setting, then iterating the floorplan tighter die, doubled macro halo, edge-stacked SRAM placement, antenna repair enabled is what took the design from "routes, but fails sign-off" to fully clean, and we wrote down what that took rather than stepping over it. Taken together, this project is less a finished chip and more a complete, honest lap around the open-source RTL-to-GDSII track proof that the loop closes end to end, including the LVS check the previous iteration of this project had to leave open, a map of exactly where it gets tight, and a running start for the pipelined 5-stage (IF, ID, EX, MEM, WB) core we're building next, still entirely on the open-source Sky130/OpenROAD toolchain.

## REFERENCES

[1] A. La Gala, M. Chiariello, M. Malanchini, M. Tambaro, and M. De Matteis, "Design and Test-Verification of a Single-Cycle RISC-V Microprocessor on FPGA," in Proc. 2024 31st IEEE Int. Conf. Electronics, Circuits and Systems (ICECS), Nancy, France, Nov. 2024.

[2] Keerthi M., Lakshmi Girish, Priyadarshini N. M., and K. Bailey, "Implementation of Single Cycle RISC-V Processor Using Verilog," Int. J. Emerging Technologies and Innovative Research (JETIR), vol. 12, no. 1, pp. f283–f288, Jan. 2025.

[3] T. Priya, P. Ushasri, and V. B. Jyothirmai, "Design of Single Cycle RISC-V Processor," Int. J. Advance Research and Innovative Ideas in Education (IJARIIE), vol. 10, no. 2, pp. 527–533, 2024.

[4] Anjana K. M., Anusha S. B., Dikshitha U., and Shilpa V., "Implementation of RISC-V Single Cycle Core," Int. J. Advanced Research in Computer and Communication Engineering (IJARCCE), vol. 14, no. 1, 2025, doi: 10.17148/IJARCCE.2025.14143.

[5] D. K. Dennis, A. Pundir, S. S. Virk, S. Agarwal, T. Sharma, A. Rajwar, and K. C. Ramkumar, "Single Cycle RISC-V Micro Architecture Processor and Its FPGA Prototype," in Proc. 2017 7th Int. Symp. Embedded Computing and System Design (ISED), 2017.

[6] E. I. Baungarten-Leon, S. Ortega-Cisneros, G. Leyva, H. E. Muñoz Zapata, E. Guzmán-Quezada, F. J. Alvarado-Rodríguez, and J. J. Raygoza-Panduro, "Comprehensive RTL-to-GDSII Workflow for Custom Embedded FPGA Architectures Using Open-Source Tools," Electronics, vol. 14, no. 19, Art. 3866, Sep. 2025, doi: 10.3390/electronics14193866.

[7] D. Acharya and U. S. Mehta, "Performance Analysis of RTL to GDS-II Flow in Opensource Tool Qflow and Commercial Tool Cadence Encounter for Synchronous FIFO," in Proc. 2022 IEEE Int. Conf. Electron Devices Society Kolkata Chapter (EDKCON), 2022, pp. 199–204.

| e-ISSN: 2278 – 8875, p-ISSN: 2320 – 3765| www.ijareeie.com | Impact Factor: 9.867 | A Monthly Peer Reviewed & Refereed Journal |

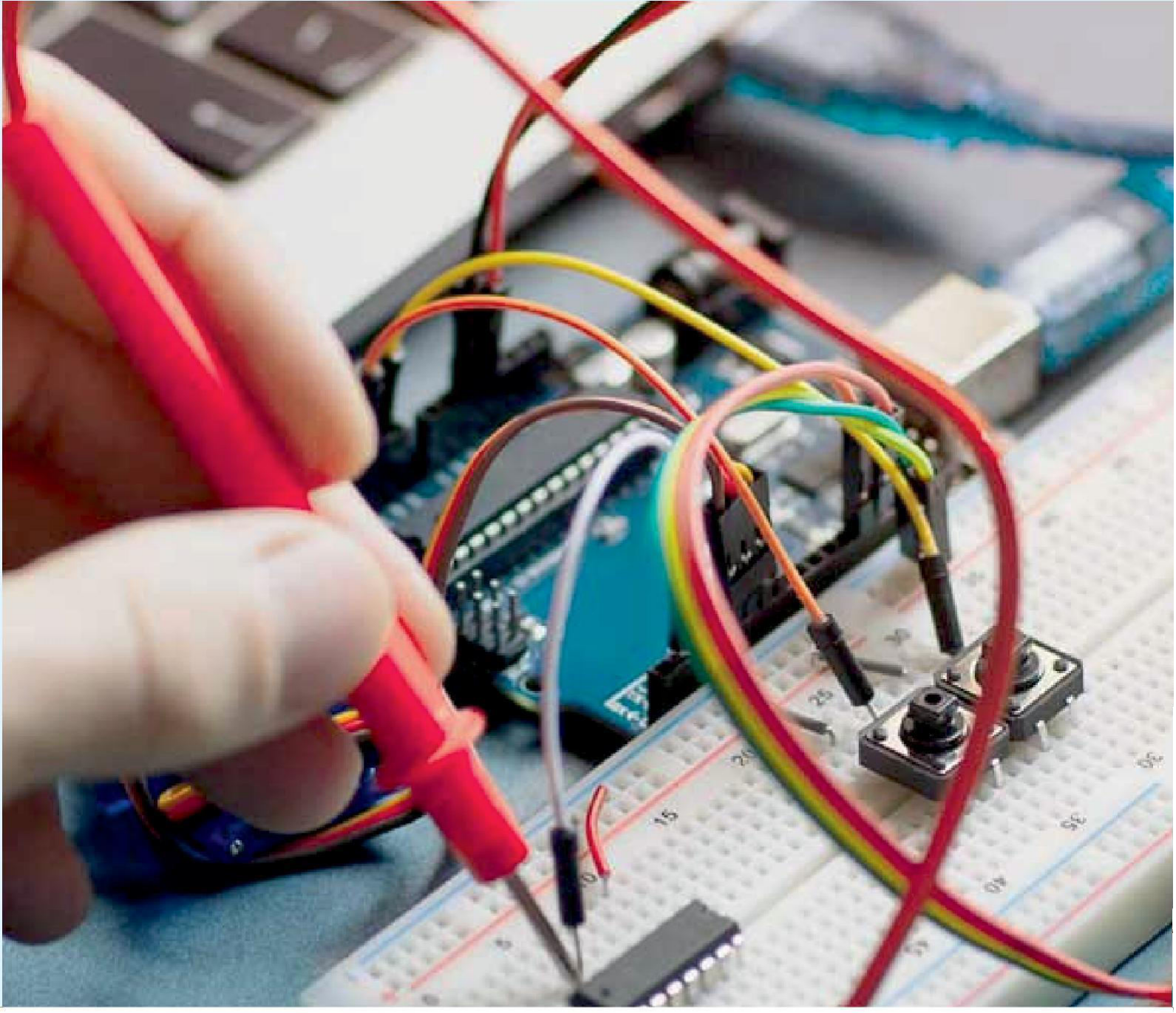
|| Volume 15, Issue 7, July 2026 ||

| DOI:10.15662/IJAREEIE.2026.1507009 |

[8] Jayasurya K., Ajith R., and Premananda B. S., "Area and Power Optimized RTL to GDS II Flow of a Telecommunication Receiver Core," in Proc. 2024 5th Int. Conf. Circuits, Control, Communication and Computing (I4C), 2024, pp. 207–212.

[9] M. D. Rao, G. Srilekha, G. Teja, K. Anusha, and G. Ganesh, "32-bit RISC-V Processor Architecture Design Using FPGA," Int. J. Creative Research Thoughts (IJCRT), vol. 13, no. 4, Apr. 2025.

[10] K. Kırali and C. B. Fidan, "Implementation of FPGA based 32-bit RISC-V processor," Engineering Science and Technology, an International Journal, vol. 70, Art. 102139, 2025, doi: 10.1016/j.jestch.2025.102139.



INNO SPACE
SJIF Scientific Journal Impact Factor

doi cross ref

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

# International Journal of Advanced Research

## in Electrical, Electronics and Instrumentation Engineering

9940 572 462 6381 907 438 ijareeie@gmail.com



Scan to save the contact details