HDL Complexity Analysis Report

Project: RISC-V SoC (rv32im_soc) | Generated: February 27, 2026 at 22:55

Analysis Mode: **Verilator AST**

1. Executive Summary

68	D	22 modules across 14 files
Score (0-100)	Grade	Scope

Key Findings:

- Found 2 module(s) with critical complexity (CC > 20)
- Found 3 module(s) with high complexity (CC > 10)
- Found 4 module(s) with deep nesting (depth > 4)
- Found 1 module(s) with many ports (> 50)
- Found 3 module(s) with large size (> 300 LOC)

2. Metrics Dashboard

8.4	34	3.1	142
Avg CC	Max CC	Avg Nesting	Avg LOC

47	28	19	5
Always Blocks	Always FF	Always Comb	High CC Modules

3. Module Complexity Breakdown

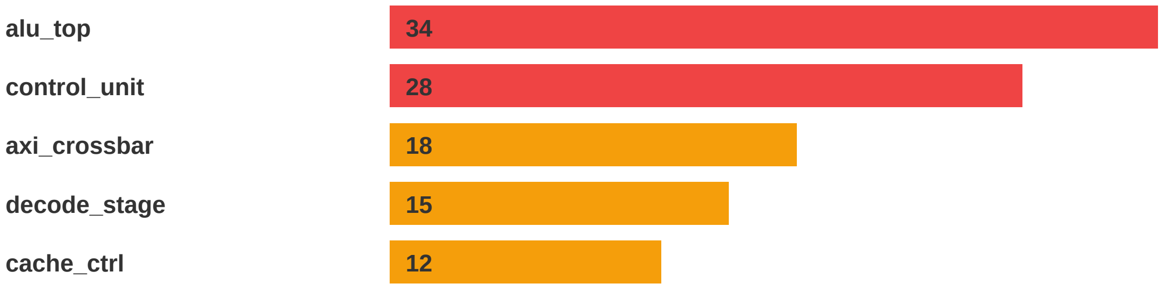
Analyzed 22 modules. Modules exceeding thresholds are highlighted below.

Module	File	Line	CC	Severity	Issue
alu_top	rtl/core/alu_top.sv	12	34	CRITICAL	alu_top: critical complexity (34), deep nesting (depth 7), large module (520 LOC)

Module	File	Line	CC	Severity	Issue
control_unit	rtl/core/control_unit.sv	1	28	CRITICAL	control_unit: critical complexity (28), large module (410 LOC)
axi_crossbar	rtl/bus/axi_crossbar.sv	5	18	HIGH	axi_crossbar: high complexity (18), many ports (64), large module (380 LOC)
decode_stage	rtl/core/decode_stage.sv	8	15	HIGH	decode_stage: high complexity (15), deep nesting (depth 6)
cache_ctrl	rtl/mem/cache_ctrl.sv	1	12	HIGH	cache_ctrl: high complexity (12), deep nesting (depth 5)
uart_rx	rtl/periph/uart_rx.sv	47	0	MEDIUM	uart_rx: deep nesting (depth 5)
spi_master	rtl/periph/spi_master.sv	22	0	MEDIUM	spi_master: deep nesting (depth 5)

4. Complexity Distribution

Horizontal bars show cyclomatic complexity for flagged modules. Thresholds: green (≤5), blue (6-10), orange (11-20), red (>20).



5. Analysis Methodology

This report was generated using **Verilator AST analysis** (--xml-only mode). Verilator compiles the RTL into an XML abstract syntax tree, from which CARE extracts precise structural metrics:

Metric	Method
Cyclomatic Complexity	Counted from *if*, *case*, *caseitem*, *while*, and *for* AST nodes within always blocks
Nesting Depth	Measured by walking the AST tree depth through *begin*, *if*, *else*, *case*, and loop structures
Port Count	Extracted from *var* elements with dir=input/output/inout attributes
Signal Count	Total *var* elements in the module AST subtree
Expression Depth	Max nesting of operator nodes (add, mul, and, or, etc.) in assignments
Operator Count	Total arithmetic, logical, comparison, and shift operator nodes

Metric	Method
Instantiation Count	Number of *instance* nodes (sub-module instantiations)

Complexity Thresholds

Metric	Threshold	Action
Cyclomatic Complexity	> 10	Review for refactoring (HIGH)
Cyclomatic Complexity	> 20	Mandatory refactoring (CRITICAL)
Nesting Depth	> 4	Extract nested logic into submodules
Port Count	> 50	Consider interface bundling
Module LOC	> 300	Split into smaller modules

6. Recommendations

• **Mandatory refactoring:** 2 module(s) have critical cyclomatic complexity (>20). Decompose complex always blocks into separate combinational and sequential sub-modules.

• **Recommended refactoring:** 3 module(s) have high complexity (CC 11-20). Review case statements for simplification and extract FSM logic into dedicated state modules.

• **Reduce nesting:** 4 module(s) exceed nesting depth 4. Flatten nested if/case chains using early-return patterns or extract inner logic into tasks/functions.

• **Reduce module size:** 3 module(s) exceed 300 LOC. Split into hierarchical sub-modules with clear interfaces.

• **Bundle interfaces:** 1 module(s) have >50 ports. Use SystemVerilog interfaces or structs to group related signals.

Scoring Methodology

The complexity score starts at 100 and deducts points for each flagged module: -10 per critical CC module, -5 per high CC module, -3 per large module, -2 per deep nesting or many-port module. The final score is clamped to [0, 100] and mapped to a letter grade (A: 90-100, B: 80-89, C: 70-79, D: 60-69, F: 0-59).