**5. Methodology**

* **Modified HL5 Architecture Overview**:
  + Key changes from the original HL5.
* **Verification Strategy**:
  + Why instruction-level simulation was chosen.
* **Simulation Framework**:
  + Tools used (e.g., custom C++/Python simulator, Spike modifications).
* **Test Bench Design**:
  + Test cases (RISCV compliance tests, custom programs).
  + Coverage metrics (instruction coverage, edge cases).

**6. Implementation**

* **Processor Modifications**:
  + Detailed description of HL5 changes (pipeline, ISA extensions, etc.).
* **Simulator Setup**:
  + How the simulator models the modified HL5.
* **Test Programs**:
  + Assembly/RISCV-tests used for verification.

**7. Results & Analysis**

* **Functional Correctness**:
  + Pass/fail results for compliance tests.
* **Performance Metrics** (if applicable):
  + Simulation speed, instruction throughput.
* **Bug Detection**:
  + Errors found during simulation and fixes applied.

**8. Discussion**

* **Strengths of Instruction-Level Simulation**:
  + Faster than RTL simulation, good for early-stage verification.
* **Limitations**:
  + May miss timing/hardware-specific bugs.
* **Future Work**:
  + Extending to cycle-accurate simulation or FPGA validation.

**9. Conclusion**

* Summary of findings.
* Confirmation of the modified HL5’s correctness (if achieved).
* Broader implications for RISC-V verification.

**10. References**

* Cite relevant papers, RISC-V manuals, simulation tools, and verification methodologies.

**11. Appendices**

* Additional diagrams, test code, or logs.

**Key Tips for Writing**

1. **Use visuals**:
   * Block diagrams of the HL5 modifications, simulation workflow, and results.
2. **Leverage existing tools**:
   * If using Spike/QEMU, explain how you adapted them.
3. **Quantify results**:
   * "X% of RISCV compliance tests passed; Y bugs were found in Z modifications."