Uncached-Lightweight Tie Link CPU-Memory Communication System

Abstract

This project demonstrates a simple communication system between a CPU and memory module using an Uncached-Lightweight Tie Link in traditional Verilog. The CPU performs a write operation followed by a read operation, with the tie link transparently passing signals between the CPU and memory. The simulation validates correct protocol behavior through signal transitions and acknowledgments.

1. Introduction

In modern digital systems, modular communication between processing elements and memory units is crucial. This project implements a basic CPU-memory communication system connected by an Uncached-Lightweight Tie Link, a simple pass-through bridge. The objective is to verify the functional correctness of data transactions through simulation in Vivado.

2. Module Descriptions

2.1 CPU

The CPU module uses a finite state machine (FSM) to initiate a write operation to a specific memory address, wait for an acknowledgment, and then initiate a read from the same address. The result is stored in an internal register.

2.2 Uncached-Lightweight Tie Link

The Uncached-Lightweight Tie Link is a simple combinational bridge that forwards address, data, and control signals from the CPU to memory, and returns the data and acknowledgment signals from memory to the CPU.

2.3 Memory

The memory module is a single-port RAM model that accepts read and write requests. It responds with the appropriate data or acknowledgment signals based on the operation.

3. Simulation Results

The simulation was run in Vivado, and waveforms were analyzed for signal correctness. The CPU successfully wrote value 0xF0 to address 0x0A, and later read back the same value.

Acknowledgment signals (`ready`) were properly asserted after each transaction, and data paths through the tie link showed correct propagation.

4. Key Observations

- Correct handshaking using `write_en`, `read_en`, and `ready` signals.

- Accurate forwarding of address and data from CPU to memory through tie link.

- Valid data captured from memory back into the CPU after a read operation.

- Expected protocol behavior confirmed: PUT_FULL_DATA, GET, ACCESS_ACK, and ACCESS_ACK_DATA.

5. Conclusion

This project successfully demonstrates a fundamental CPU-memory transaction system using an Uncached-Lightweight Tie Link. The system is minimal yet functionally correct, and lays the groundwork for more advanced interconnects such as Heavy or Cache tie links. The simulation confirms data integrity and correct timing behavior.

6. Future Improvements

- Add support for PUT_PARTIAL_DATA using byte-enabled masking in memory.

- Introduce burst transfers or pipelined transactions.

- Extend the tie link to support arbitration or priority logic.

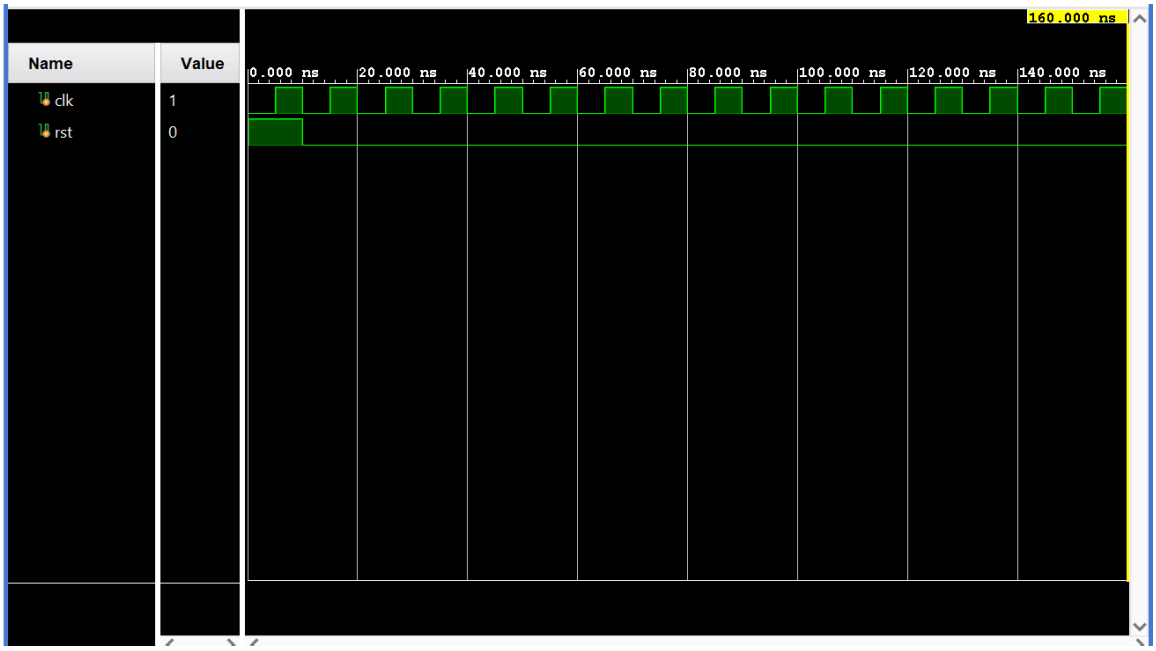
- Upgrade to SystemVerilog with UVM-style testbench automation.

Appendix A: Verilog Source Code (Top Level)

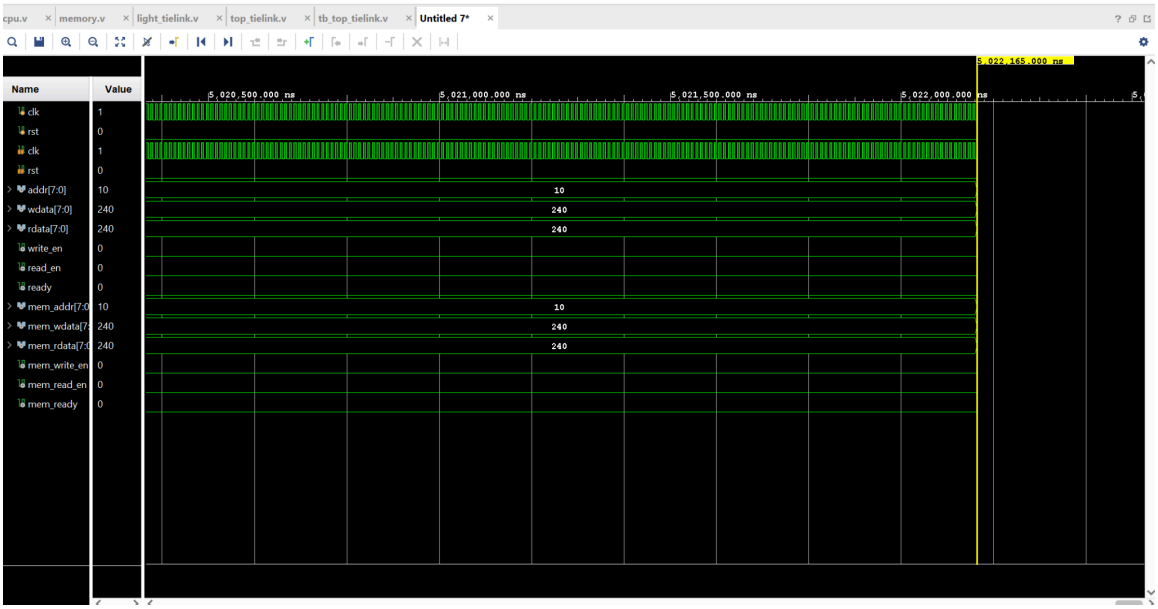
Refer to the files: top_tielink.v, cpu.v, memory.v, light_tielink.v, tb_top_tielink.v

Appendix B: Simulation Waveforms & Schematic

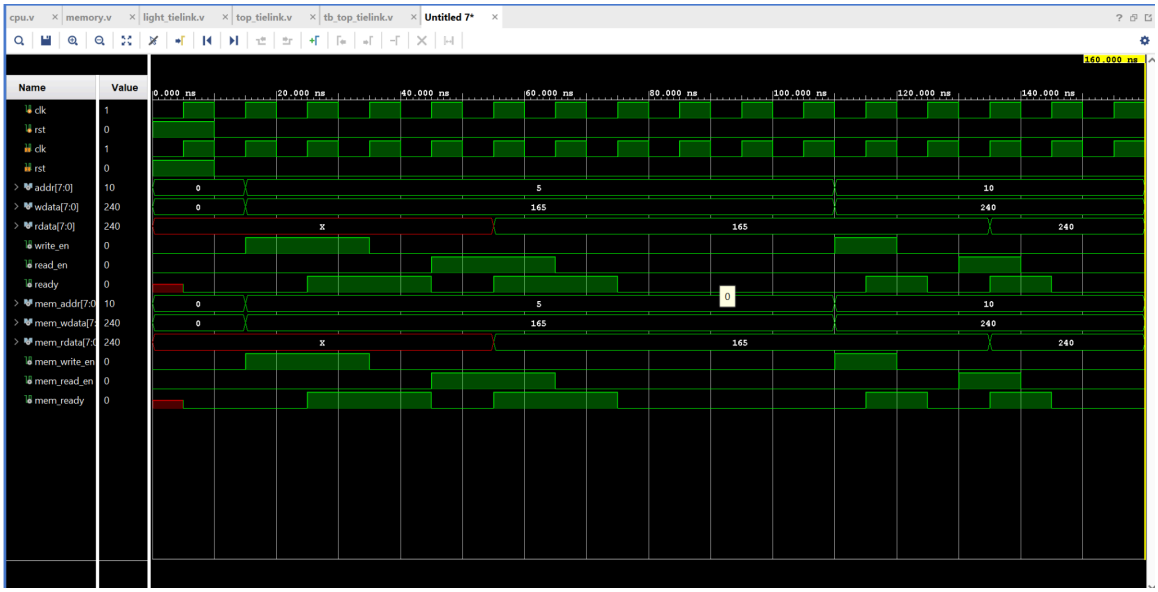
This section presents the waveform outputs and schematic generated from Vivado simulation. They demonstrate the functionality of the lightweight tie link system from signal timing and structural perspectives.



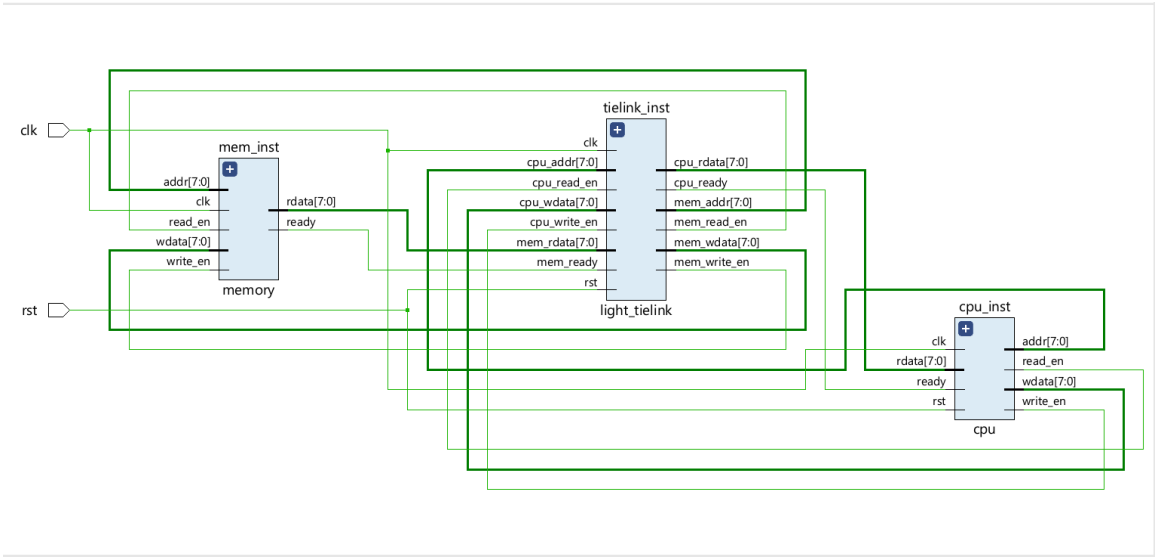
Waveform 1: Clock and Reset Signal Behavior



Waveform 2: CPU Writes and Reads with Correct Data Handshaking



Waveform 3: Final Transaction Verification (0xF0 Written and Read)



Block Diagram: RTL Hierarchy (CPU ↔ Tie Link ↔ Memory)

by: Ragul Ganesh Anitha Palanivel