MIPS32 Pipelined Processor - Module Descriptions

Detailed Module Explanations

Instruction Fetch (IF) Stage

1. pc.v Program Counter

Purpose: Holds the address of the current instruction.

Inputs: clk, reset, pc_next, pc_write

Outputs: pc

Logic: On every clock edge, if not stalled or reset, updates the PC.

2. instruction_memory.v

Purpose: Stores program instructions in a read-only format.

Inputs: address (from PC)

Outputs: instruction

Logic: Address is used as an index into memory array (word-aligned).

3. if_stage.v

Purpose: Combines PC increment logic and instruction fetch.

Outputs: pc_plus4, instruction

4. if_id.v Pipeline Register

Purpose: Holds instruction and PC from IF to ID stage.

Inputs: clk, reset, enable, pc, instruction

Outputs: pc_out, instruction_out

Instruction Decode (ID) Stage

MIPS32 Pipelined Processor - Module Descriptions

5. register_file.v

Purpose: Stores and provides register data.

Inputs: read_reg1, read_reg2, write_reg, write_data, reg_write, clk

Outputs: read_data1, read_data2

6. control_unit.v

Purpose: Decodes opcode to generate control signals.

Inputs: opcode

Outputs: reg_write, mem_read, alu_op, alu_src, etc.

7. sign_extender.v

Purpose: Sign-extends 16-bit immediate to 32-bit.

Input: 16-bit immediate

Output: 32-bit signed immediate

8. id_stage.v

Purpose: Wires together control unit, register file, and extender.

Outputs: All signals required by EX stage.

9. id_ex.v Pipeline Register

Purpose: Transfers all ID stage outputs to EX stage.

Includes: operand values, sign-extended immediate, control signals

Execute (EX) Stage

10. alu.v

Purpose: Executes arithmetic and logic operations.

Inputs: operand1, operand2, alu_control

Outputs: result, zero

11. alu_control.v

Purpose: Translates ALUOp + funct into actual ALU control signal.

Inputs: alu_op, funct

Output: alu_control

12. forwarding_unit.v

Purpose: Detects hazards and forwards data from later stages.

Inputs: Register numbers and write-back flags from EX/MEM and MEM/WB

Outputs: Control signals to select forwarded data

13. ex_stage.v

Purpose: Combines ALU, forwarding muxes, branch target calc

Outputs: ALU result, Branch target, Zero flag for branch

14. ex_mem.v Pipeline Register

Purpose: Passes ALU result and control signals to MEM stage.

Memory Access (MEM) Stage

15. data_memory.v

Purpose: Performs memory read/write operations.

Inputs: address, write_data, mem_read, mem_write, clk

Outputs: read_data

16. mem_stage.v

Purpose: Executes memory operations or passes ALU result forward.

Output: Read data or ALU result for write-back

17. mem_wb.v Pipeline Register

Purpose: Holds data for final write-back stage.

Write Back (WB) Stage

18. wb_stage.v

Purpose: Selects whether to write back from memory or ALU.

Inputs: mem_to_reg, mem_data, alu_result

Output: write_back_data

Auxiliary Modules

19. hazard_detection_unit.v

Purpose: Detects and stalls pipeline if a load-use hazard exists.

Inputs: Current instruction and EX/MEM reg usage

Output: pc_write, if_id_write, control_signal_flush

20. branch_control.v

Purpose: Compares registers for beq and decides whether to branch.

Inputs: Operand1, Operand2

Output: Branch decision (branch_taken)

21. top_module.v

Purpose: Integrates all pipeline stages.

Connects: Clock, reset, PC, memories, all stages

22. mux.v (Multiple versions)

Purpose: Used to switch between:

- Reg2 vs. Imm for ALU input
- MemData vs. ALU result for write-back
- Forwarded values in EX stage
- PC+4 vs. branch/jump targets

23. testbench.v

Purpose: Simulates and tests the processor.

Includes: Clock generation, Reset, Instruction memory initialization, Output dumping to waveform viewer