A diagram of a machine

AI-generated content may be incorrect.

**Figure 1. Datapath of 5-Stage RV32I Pipelined Processor with Full Hazard Detection, Forwarding, and Control Logic**

A diagram of a computer hardware

AI-generated content may be incorrect.

**Figure 2. 5-Stage RV32I Pipelined Processor with External Instruction and Data Memory Interfaces**

**A diagram of a green square

AI-generated content may be incorrect.**

**Figure 3. Netlist View of 5-Stage RV32I Pipelined Processor with External Instruction and Data Memory Interfaces**

**A diagram of a computer

AI-generated content may be incorrect.**

**Figure 4. Netlist View of 5-Stage RV32I Pipelined Processor**

**A diagram of a computer

AI-generated content may be incorrect.**

**Figure 7. Netlist View of Controller of 5-Stage RV32I Pipelined Processor**

**A computer screen shot of a diagram

AI-generated content may be incorrect.**

**Figure 8. Netlist View of Hazard Unit of 5-Stage RV32I Pipelined Processor**

**A computer screen shot of a computer

AI-generated content may be incorrect.**

**Figure 8. Overall Netlist View of Data Path of 5-Stage RV32I Pipelined Processor**

**A computer screen shot of a computer

AI-generated content may be incorrect.**

**Figure 5. Netlist View of Data Memory**

**A computer program with text

AI-generated content may be incorrect.**

**Figure 6. Netlist View of Instruction Memory**

**Table 1. Categorized Signal and Bus Lines in a 5-Stage RV32I Pipelined Processor**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Signal Name** | **Bit Width** | **Source** | **Destination** | **Description** |
| **Global Signals** | | | | |
| clk | 1 | Global | All synchronous logic | Clock signal |
| rst | 1 | Global | All reset logic | Asynchronous reset |
| **Fetch Stage (F)** | | | | |
| pcF | 32 | Freg | Instruction Memory | Program Counter (current) |
| pcF0 | 32 | PCmux | Freg | Next PC value |
| InstrF | 32 | Instruction Memory | Dreg | Fetched instruction |
| PCPlus4F | 32 | PCplus4 adder | Dreg | PC + 4 for sequential flow |
| StallF | 1 | Hazard Unit | Freg | Stall PC update |
| **Decode Stage (D)** | | | | |
| InstrD | 32 | Dreg | Controller | Decoded instruction |
| pcD | 32 | Dreg | Ereg | PC value in decode |
| Rs1D, Rs2D | 5 | InstrD | Hazard Unit | Source register IDs |
| Read1D, Read2D | 32 | reg\_file | Ereg | Register read data |
| ImmExtD | 32 | extender | Ereg | Extended immediate |
| PCPlus4D | 32 | Dreg | Ereg | PC+4 forwarded |
| RdD | 5 | InstrD | Ereg | Destination register |
| RegWriteD | 1 | Controller | Ereg | Register write enable |
| MemWriteD | 1 | Controller | Ereg | Memory write enable |
| JumpD, BranchD | 1 | Controller | Ereg | Jump/Branch control |
| ALUSrcD | 1 | Controller | Ereg | ALU source select |
| ALUControlD | 3 | Controller | Ereg | ALU operation control |
| ResultSrcD | 2 | Controller | Ereg | Result source select |
| ImmSrcD | 2 | Controller | extender | Immediate format select |
| **Execute Stage (E)** | | | | |
| Read1E, Read2E | 32 | Ereg | Forwarding muxes | Register data |
| pcE | 32 | Ereg | PCplusbranch | PC in execute |
| ImmExtE | 32 | Ereg | srcBmux2 | Extended immediate |
| PCPlus4E | 32 | Ereg | Mreg | PC+4 forwarded |
| PCTargetE | 32 | PCplusbranch | PCmux | Branch/Jump target |
| SrcAE, SrcBE | 32 | Muxes | ALU | ALU operands |
| ALUResultE | 32 | ALU | Mreg | ALU result |
| WriteDataE | 32 | srcBmux1 | Mreg | Store data |
| RegWriteE | 1 | Ereg | Mreg | Register write enable |
| MemWriteE | 1 | Ereg | Mreg | Memory write enable |
| ALUSrcE | 1 | Ereg | srcBmux2 | ALU source B select |
| ResultSrcE | 2 | Ereg | Mreg | Result select |
| ZeroE | 1 | ALU | Branch logic | Zero flag |
| OverflowE | 1 | ALU | - | Overflow flag |
| **Memory Stage (M)** | | | | |
| ALUResultM | 32 | Mreg | Data Memory/Wreg | Memory address/ALU result |
| WriteDataM | 32 | Mreg | Data Memory | Store data |
| PCPlus4M | 32 | Mreg | Wreg | PC+4 forwarded |
| RegWriteM | 1 | Mreg | Wreg | Register write enable |
| MemWriteM | 1 | Mreg | Data Memory | Memory write enable |
| ResultSrcM | 2 | Mreg | Wreg | Result select |
| RdM | 5 | Mreg | Wreg | Destination register |
| **Writeback Stage (W)** | | | | |
| ALUResultW | 32 | Wreg | rsltmux | ALU result |
| ReadDataW | 32 | Wreg | rsltmux | Memory read data |
| PCPlus4W | 32 | Wreg | rsltmux | PC+4 value |
| ResultW | 32 | rsltmux | reg\_file | Write data |
| RegWriteW | 1 | Wreg | reg\_file | Register write enable |
| ResultSrcW | 2 | Wreg | rsltmux | Result select |
| RdW | 5 | Wreg | reg\_file | Destination register |

**Table 2. ImmSrc Encoding**

|  |  |  |
| --- | --- | --- |
| **ImmSrc** | **ImmExt** | **Type** |
| **000** | {{20{Instr[31]}}, Instr[31:20]} | **I** |
| **001** | {{20{Instr[31]}}, Instr[31:25], Instr[11:7]} | **S** |
| **010** | {{20{Instr[31]}}, Instr[7], Instr[30:25], Instr[11:8], 1’b0} | **B** |
| **011** | {{12{Instr[31]}}, Instr[19:12], Instr[20], Instr[30:21], 1’b0} | **J** |
| **100** | {Instr[31:12], 12’b0} | **U** |

**Table 3. Main Decoder Truth Table**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **op** | **Reg-Write** | **Imm-Src** | **ALU-Src** | **Mem-Write** | **Result-Src** | **Branch** | **ALU-Op** | **Jump** | **Jumplr** | **Instruction** |
| 0110011 | 1 | xxx | 0 | 0 | 00 | 0 | 10 | 0 | 0 | R-type |
| 0010011 | 1 | 000 | 1 | 0 | 00 | 0 | 10 | 0 | 0 | I-type ALU |
| 0000011 | 1 | 000 | 1 | 0 | 01 | 0 | 00 | 0 | 0 | I-type Load |
| 0100011 | 0 | 001 | 1 | 1 | x0 | 0 | 00 | 0 | 0 | S-type |
| 1100011 | 0 | 010 | 0 | 0 | x0 | 1 | 01 | 0 | 0 | B-type |
| 1101111 | 1 | 011 | x | 0 | 10 | 0 | xx | 1 | 0 | J-type |
| 1100111 | 1 | 000 | 1 | 0 | 10 | 0 | 10 | 0 | 1 | I-type jalr |
| 0110111 | 1 | 100 | 1 | 0 | 00 | 0 | 11 | 0 | 0 | U-type lui |
| 0010111 | 1 | 100 | x | 0 | 11 | 0 | xx | 0 | 0 | U-type auipc |
| 1110011 | 0 | xxx | x | 0 | x0 | 0 | xx | 0 | 0 | I-type ecall, ebreak |

**A table with numbers and symbols

AI-generated content may be incorrect.A table with numbers and symbols

AI-generated content may be incorrect.**

**A table with numbers and text

AI-generated content may be incorrect.A table with numbers and letters

AI-generated content may be incorrect.**

**Table 4. ALU Decoder Truth Table**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ALU-Op** | **funct3** | **{op5, funct75}** | **ALU-Control** | **Operation** | **Instruction** |
| 00 | x | xx | 0000 | add | I-type Load, S-type |
| 01 | xxx | xx | 0001 | subtract | B-type |
| 10 | 000 | 00, 01, 10 | 0000 | add | add, addi, jalr |
| 11 | 0001 | subtract | sub |
| 001 | xx | 0110 | shift left logical | sll, slli |
| 010 | xx | 0101 | set less than | slt, slti |
| 011 | xx | 1001 | set less than (U) | sltu, sltui |
| 100 | xx | 0100 | xor | xor, xori |
| 101 | x0 | 0111 | shift right logical | srl, srli |
| x1 | 1000 | shift right arithmetic | sra, srai |
| 110 | xx | 0011 | or | or, ori |
| 111 | xx | 0010 | and | and, andi |
| 11 | x | xx | 1010 | take immediate only | lui |

**Table 5. LSU Truth Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **funct3** | **LSType** | **Type** | **Instruction** |
| 000 | 10000 | Byte | lb, sb |
| 001 | 01000 | Half | lh, sh |
| 010 | 00100 | Word | lw, sw |
| 100 | 00010 | Byte (U) | lbu |
| 101 | 00001 | Half (U) | lhu |

**A table with numbers and symbols

AI-generated content may be incorrect.A table with numbers and symbols

AI-generated content may be incorrect.A table with numbers and numbers

AI-generated content may be incorrect.A table with numbers and symbols

AI-generated content may be incorrect.**

**Table 5. BRU Truth Table**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Branch** | **Jump** | **Jumplr** | **funct3** | **Flag**  **= {V, C, N, Z}** | **PCSrc1** | **PCSrc0** | **Instruction** |
| 1 | 0 | 0 | 000 | xxxx | 0 | Z | beq |
| 001 | xxxx | 0 | ~ Z | bne |
| 100 | xxxx | 0 | N ^ V | blt |
| 101 | xxxx | 0 | ~ (N ^ V) | bge |
| 110 | xxxx | 0 | ~ C | bltu |
| 111 | xxxx | 0 | C | bgeu |
| 0 | 1 | 0 | xxx | xxxx | 0 | 1 | jal |
| 0 | 0 | 1 | xxx | xxxx | 1 | 0 | jalr |

**Table 6. EU Truth Table**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **op** | **funct12** | **Ecall** | **Ebreak** | **Instruction** |
| 1110011 | 000000000000 | 1 | x | ecall |
| 1110011 | 000000000001 | x | 1 | ebreak |