CO502 - Part 2 (Hardware Units)

Group 1

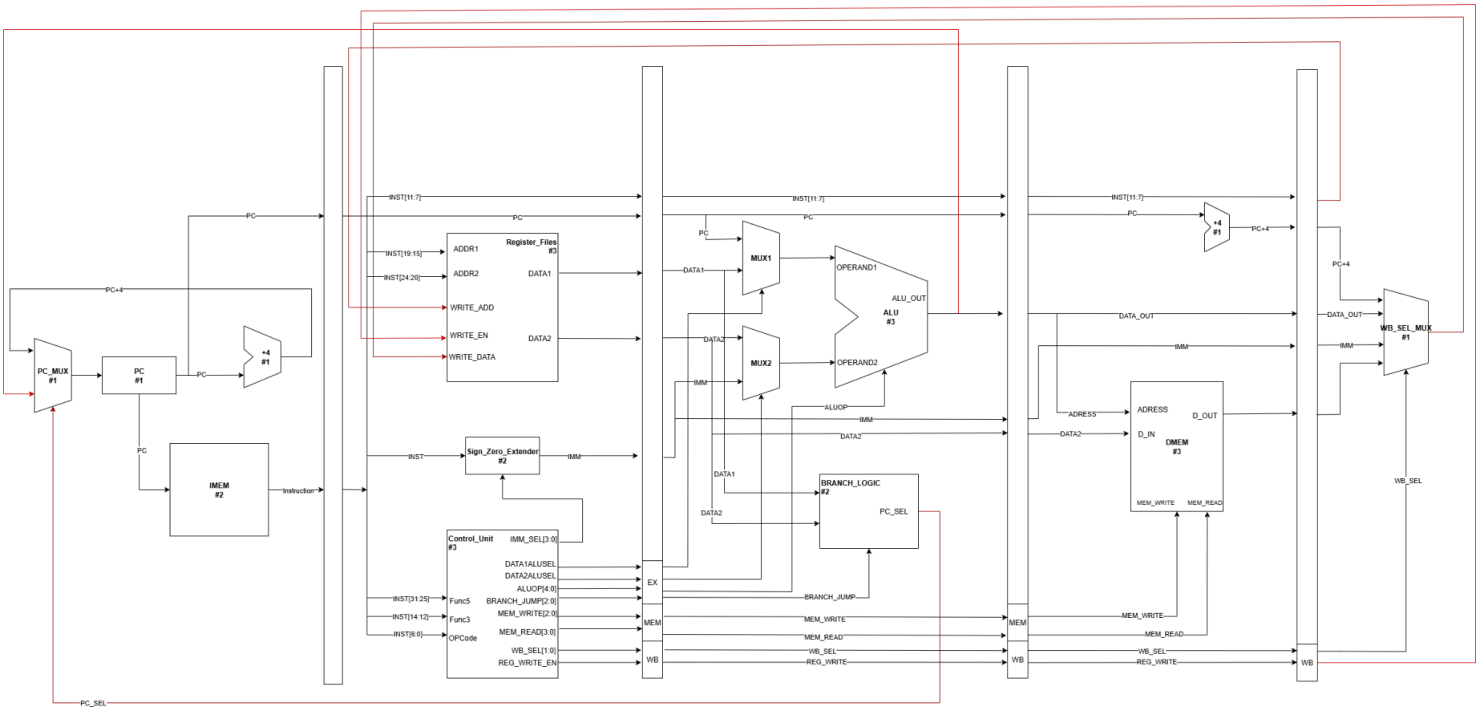
Table of Contents

Overview

This part of the project focuses on implementing the core hardware units of a 5-stage pipelined RISC-V processor. The processor is designed according to the classic RISC pipeline model, consisting of the following stages:

- Instruction Fetch (IF)
- Instruction Decode (ID)
- Execute (EX)
- Memory Access (MA)
- Write Back (WB)

Each stage includes specific hardware components responsible for its operation. The diagram below illustrates the overall architecture of the processor across these stages.



RISC-V 5 Stage Pipelined Processor

As summarized in the table below:

Stage	Main Components
IF	Program Counter (PC), Instruction Memory
ID	Control Unit, Sign/Zero Extender, Register File
EX	ALU, Branch Unit, Multiplexers
MA	Data Memory, Multiplexers
WB	Multiplexer (for write-back value selection)

The following sections describe the design and implementation of each hardware unit, explaining their purpose, design decisions, limitations, timing behavior, and any utility modules used.

Control Unit

Decodes the instruction word into necessary control signals.

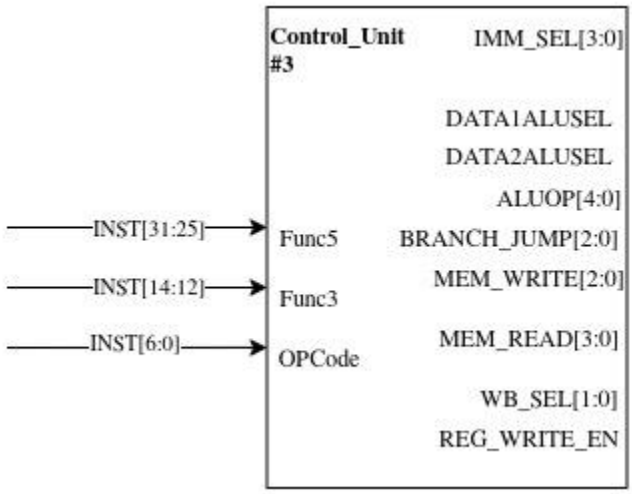


Figure 1: Control Unit

Input:

- OPCode: First 7 bits of the instruction that contains the information on the type of operation to be performed.
- Func3: Specifies the sub-operation under the subset of operations possible under the same opcode.
- Fun5: Provides additional information required to perform the operation.

Outputs:

- REG_WRITE_EN: Signals whether the data can be written onto the register.
- WB_SEL: 2-bit signal determines data source for register write-back:
    - 00: ALU result

    - 01: Memory data (for loads)
    - 10: PC+4 (for JAL/JALR)
- MEM_READ: Memory read control (bit 3: enable, bits 2:0: operation type)
    - MEM_READ[3]: Read Enable Bit
        - 1: Memory read operation is active (LOAD instruction)
        - 0: No memory read operation
    - MEM_READ[2:0]: Load Type Encoding (complete funct3 field)
        - 000: LB (Load Byte, sign-extended)
        - 001: LH (Load Halfword, sign-extended)
        - 010: LW (Load Word)
        - 011: Reserved
        - 100: LBU (Load Byte Unsigned)
        - 101: LHU (Load Halfword Unsigned)
        - 110: Reserved
        - 111: Reserved
- MEM_WRITE: Memory write control (bit 2: enable, bits 1:0: operation type)
    - MEM_WRITE[2]: Write Enable Bit
        - 1: Memory write operation is active (STORE instruction)
        - 0: No memory write operation
    - MEM_WRITE[1:0]: Store Size Encoding (directly from funct3[1:0])
        - 00: SB (Store Byte) - 8-bit write
        - 01: SH (Store Halfword) - 16-bit write
        - 10: SW (Store Word) - 32-bit write
        - 11: Reserved/Invalid
- BRANCH_JUMP: Branch/jump operation control and condition encoding
    - BRANCH_JUMP[3] - Branch/Jump Enable Bit
        - 1: Branch or jump instruction is active
        - 0: No branch or jump instruction (normal sequential execution)
    - BRANCH_JUMP[2:0] - Operation Type Encoding
- ALUOP:
    - Bits [2:0]: Operation type (derived from funct3 or hardcoded)
    - Bit [3]: Multiplication/LUI indicator
    - Bit [4]: Arithmetic shift/subtraction/LUI modifier
- DATA1ALUSEL: ALU operand 1 source (register vs PC)
- DATA2ALUSEL: ALU operand 2 source (register vs immediate)
- IMM_SEL: Immediate generation format selection. The control unit supports 6 different immediate types:
    - IMM_TYPE1: U-type (LUI, AUIPC)
    - IMM_TYPE2: J-type (JAL)
    - IMM_TYPE3: I-type (LOAD, JALR, I-type ALU)
    - IMM_TYPE4: B-type (BRANCH)
    - IMM_TYPE5: S-type (STORE)
    - IMM_TYPE6: Shift immediate (I-type shifts)

An artificial 3 time unit delay is taken up when the control signals are generated. Following outlines the control signal outputs for each instruction:

Instruction	ALUOP	REG_WRITE_EN	MEM_WRITE	MEM_READ	BRANCH_JUMP	IMM_SEL	DATA1ALUSEL	DATA2ALUSEL	WB_SEL
R-Type									
ADD	00000	1	000	0000	0000	1xxx*	0	0	00
SUB	10000	1	000	0000	0000	1xxx	0	0	00
SLL	00001	1	000	0000	0000	0xxx	0	0	00
SLT	00010	1	000	0000	0000	1xxx	0	0	00
SLTU	00011	1	000	0000	0000	1xxx	0	0	00
XOR	00100	1	000	0000	0000	0xxx	0	0	00
SRL	00101	1	000	0000	0000	0xxx	0	0	00
SRA	10101	1	000	0000	0000	0xxx	0	0	00
OR	00110	1	000	0000	0000	0xxx	0	0	00
AND	00111	1	000	0000	0000	0xxx	0	0	00
MUL	01000	1	000	0000	0000	1xxx	0	0	00
MULH	01001	1	000	0000	0000	1xxx	0	0	00
MULHSU	01010	1	000	0000	0000	1xxx	0	0	00
MULHU	01011	1	000	0000	0000	1xxx	0	0	00
DIV	01100	1	000	0000	0000	0xxx	0	0	00
DIVU	01101	1	000	0000	0000	0xxx	0	0	00
REM	01110	1	000	0000	0000	1xxx	0	0	00

REMU	01111	1	000	0000	0000	1xxx	0	0	00
I-Type									
ADDI	00000	1	000	0000	0000	0010	0	1	00
SLTI	00010	1	000	0000	0000	0010	0	1	00
SLTIU	00011	1	000	0000	0000	1011	0	1	00
XORI	00100	1	000	0000	0000	0010	0	1	00
ORI	00110	1	000	0000	0000	0010	0	1	00
ANDI	00111	1	000	0000	0000	0010	0	1	00
SLLI	00001	1	000	0000	0000	0101	0	1	00
SRLI	00101	1	000	0000	0000	0101	0	1	00
SRAI	10101	1	000	0000	0000	0101	0	1	00
LB	00000	1	000	1000	0000	0010	0	1	01
LH	00000	1	000	1001	0000	0010	0	1	01
LW	00000	1	000	1010	0000	0010	0	1	01
LBU	00000	1	000	1100	0000	1010	0	1	01
LHU	00000	1	000	1101	0000	1010	0	1	01
S-Type									
SB	00000	0	100	0000	0000	0100	0	1	xx
SH	00000	0	101	0000	0000	0100	0	1	xx
SW	00000	0	110	0000	0000	0100	0	1	xx

U-Type									
LUI	11000	1	000	0000	0000	0000	0	1	00
AUIPC	00000	1	000	0000	0000	0000	1	1	00
B-Type									
BEQ	00000	0	000	0000	1000	0011	1	1	xx
BNE	00000	0	000	0000	1001	0011	1	1	xx
BLT	00000	0	000	0000	1100	0011	1	1	xx
BGE	00000	0	000	0000	1101	0011	1	1	xx
BLTU	00000	0	000	0000	1110	0011	1	1	xx
BGEU	00000	0	000	0000	1111	0011	1	1	xx
J-Type									
JAL	00000	1	000	0000	1010	0001	1	1	10

Following outlines the limitations of the design:

- Hardcoded delay values may not match actual hardware
- Hazard handling is required.
- No exception and interrupt support

Register File

Implements the 32 general-purpose registers (x0-x31) specified by the RISC-V ISA. It operates as a 3-port memory (2 read ports, 1 write port) that enables:

- Dual simultaneous reads for source operands (rs1, rs2)
- Single write operation for destination register (rd)
- Pipeline data supply during the Instruction Decode (ID) stage
- Write-back data storage from the Write-Back (WB) stage

Each register is 32 bits wide for a total storage of 1024 bits (32 x 32 bits). Read operations are done asynchronously with 1 time unit delay, The write operations are synchronized to the

negative clock edge with a 2 time unit delay (as shown in Figure 3) to coordinate with the write-back stage timing.

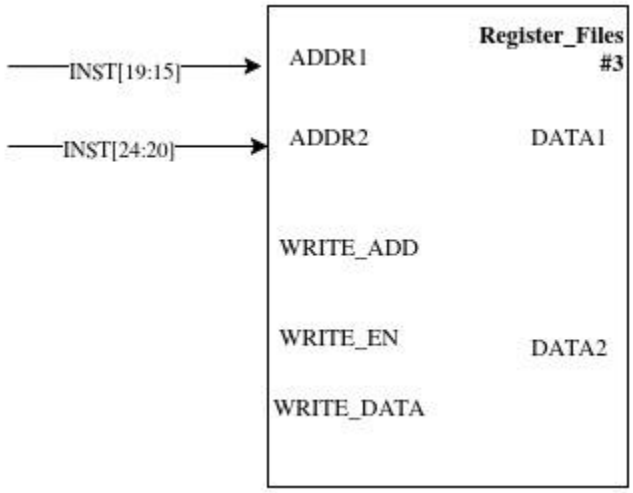


Figure 2: Register File

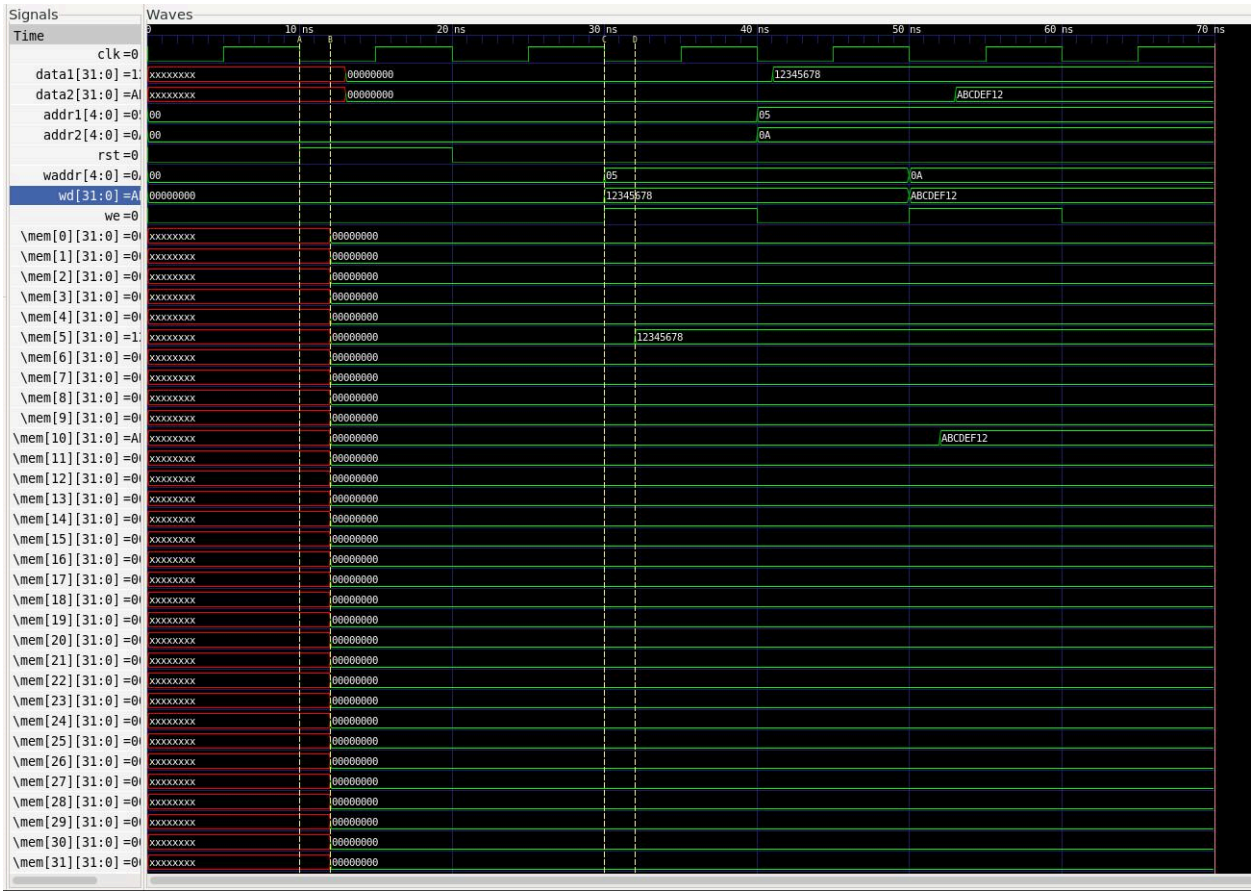


Figure 3: Register File Timing Diagram

Following outlines the limitations of the design:

- Only one register write per cycle
- Must rely on external hazard handling

Integer ALU

Performs all arithmetic, logical, and comparison operations. It operates in the Execute (EX) stage of the pipeline and implements the complete RV32IM instruction set.

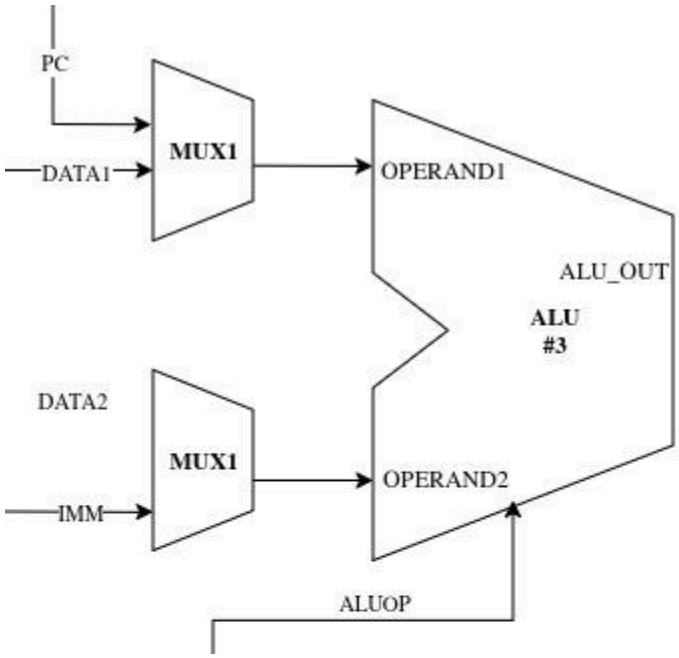


Figure 4: ALU

Inputs:

- DATA1: Register data or PC
- DATA2: Register data or immediate value
- SELECT: 5-bit ALUOP from control unit

Output Usage:

- Memory address calculation: For load/store instructions
- Branch target calculation: For branch/jump instructions
- Arithmetic results: For register write-back
- Forwarding: For hazard resolution

Design Choices:

- Conditional computation: Only active operation units consume power
- Reduced switching activity: Unused functional units remain inactive
- Operand isolation: Multiplication and division operands are gated

- Power-conscious approach: Optimized for low-power operation
- RISC-V Division Compliance:
    - Division by zero: Returns -1 for signed, max value for unsigned
    - Overflow handling: Proper handling of most negative ÷ -1
    - Remainder by zero: Returns dividend unchanged
    - Safe operation: No exceptions, always returns valid result

Limitations:

- Multiplication and division operations create long delays
- Large area overhead due to multiple parallel functional units
- Only integer operations supported
- No vector or parallel data operations

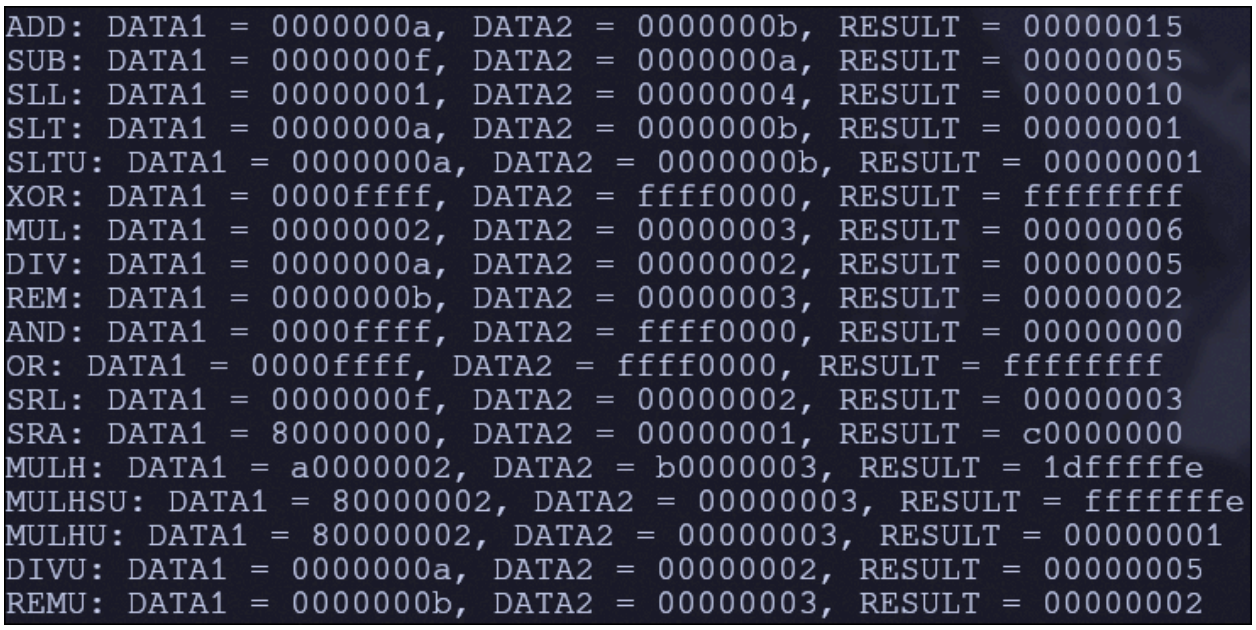


Figure 5: Testbench output for ALU

Branch Logic

The branch logic serves as the control flow decision unit in the RISC-V pipeline processor, determining whether branches and jumps should be taken. It operates in the Execute (EX) stage and provides the critical PC multiplexer select signal that controls program flow. The module implements:

- Conditional branch evaluation: For all RISC-V branch instructions
- Unconditional jump support: For JAL and JALR instructions
- Pipeline control: Generates PC selection signal for instruction fetch
- Comparison operations: Signed and unsigned comparisons for branch conditions

Output Operation Encoding (4-bit op signal)

op[3]: Branch/Jump enable bit (1 = active, 0 = no branch)

op[2:0]: Branch/Jump type encoding:

op[2:0]	Operation	Condition	Description
000	**BEQ**	`data1 == data2`	Branch if equal
001	**BNE**	`data1 != data2`	Branch if not equal
010	**JAL/JALR**	Always true	Unconditional jump
100	**BLT**	`$signed(data1) < $signed(data2)`	Branch if less than (signed)
101	**BGE**	`$signed(data1) >= $signed(data2)`	Branch if greater/equal (signed)
110	**BLTU**	`$unsigned(data1) < $unsigned(data2)`	Branch if less than (unsigned)
111	**BGEU**	`$unsigned(data1) >= $unsigned(data2)`	Branch if greater/equal (unsigned)

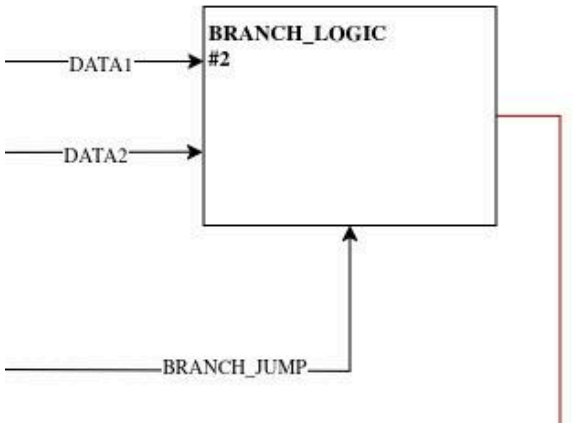


Figure 6: Branch Logic

Design Decisions:

- Parallel computation: All comparison results available simultaneously
- Fast selection: Multiplexer selects appropriate result based on operation
- Combinational logic: No clock dependency for comparison operations

- Enable gating: Branch evaluation only when instruction requires it
- Timing control: 2ns delay models realistic branch decision timing

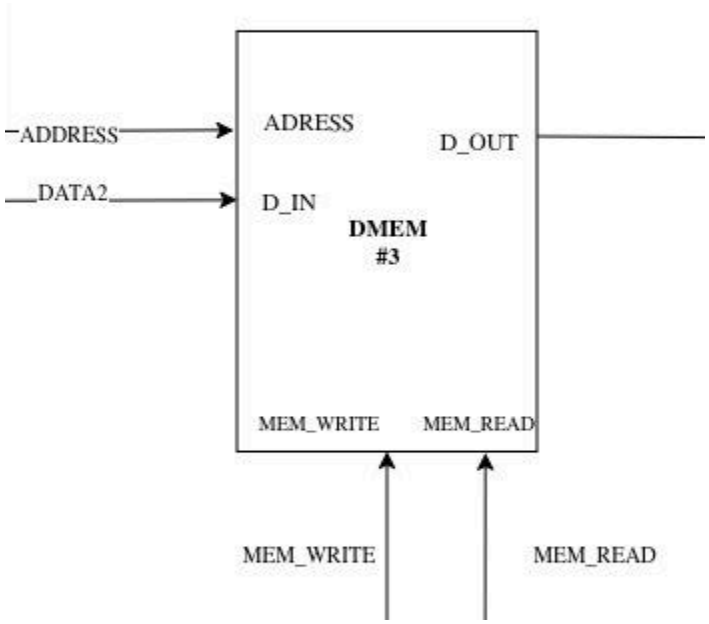
Limitations:

- No speculation: Cannot begin fetching target instructions early
- No early resolution: Branch condition cannot be determined earlier in pipeline
- Always-on comparisons: All comparison operations computed regardless of need

Data Memory

The data memory (dmem) serves as the primary data storage component in the RISC-V pipeline processor, handling all load and store operations. It operates in the Memory Access (MA) stage of the pipeline and provides:

- Load operation support: All RISC-V load instructions (LB, LH, LW, LBU, LHU)
- Store operation support: All RISC-V store instructions (SB, SH, SW)
- Variable data width access: Byte, halfword, and word operations
- Sign extension handling: Automatic sign extension for signed loads
- Pipeline integration: Interfaces with CPU pipeline through control signals



Design Decisions:

- Storage: 256 bytes (64 words) byte-addressable array
- Endianness: Little-endian byte ordering
- Address range: 0x00 to 0xFF
- Access latency: 2 time units for both read and write operations

Limitations:

- Small memory size: Only 256 bytes total capacity
- No caching: Direct memory access without cache optimization

Utilities and Supporting Components

Multiplexers (MUXes)

32-bit 2-to-1 Multiplexer

- PC Multiplexer: Selects between PC+4 (sequential) and branch target
- ALU Input Multiplexers: Select between register data and PC/immediate values
- Various control path selections throughout the pipeline

32-bit 3-to-1 Multiplexer

- Forwarding Multiplexers: Select between current register data, forwarded ALU result, or forwarded memory data
- Write-back Multiplexer: Selects between memory data, ALU result, and PC+4 for register write-back

3-bit 2-to-1 Multiplexer

- Control Unit ALU Operation Selection: Chooses between funct3 field and hardcoded 3'b000 for specific instructions
- Used for JAL, AUIPC, STORE, LOAD, BRANCH instructions where ALU operation is predetermined

Adders

The implementation includes specialized adder modules for program counter arithmetic and address calculation.

Conclusion

This report covered the design and implementation of key hardware components for a 5-stage pipelined RISC-V processor. Each unit is modular, reusable, and designed to match the RISC-V ISA specifications. The current design does not include pipeline hazard handling, forwarding, or stall logic, which will be added in future iterations.