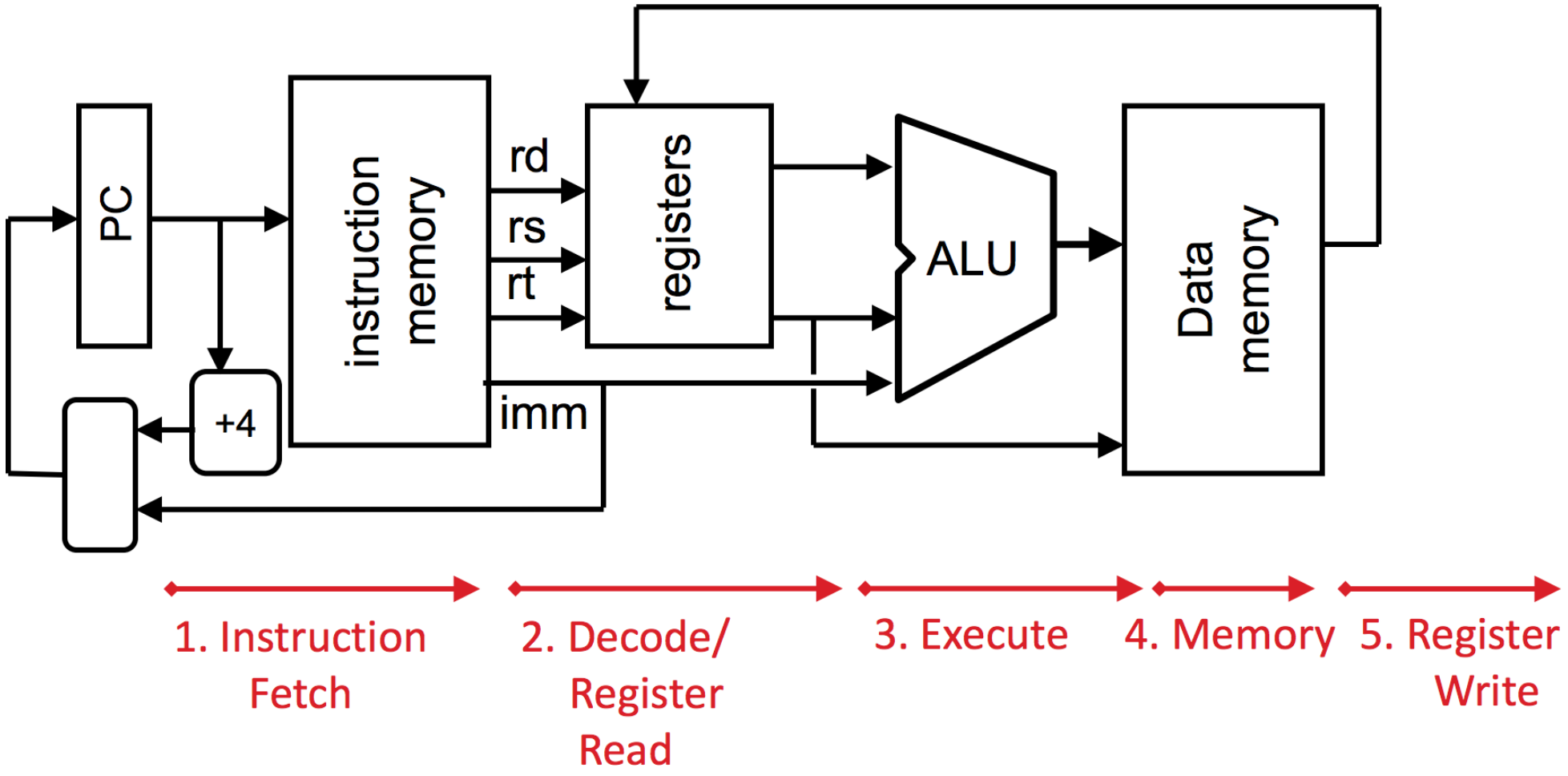
RISC-V Single-Cycle CPU Datapath

Discussion 7

Stages of Execution on Datapath



Stages of Execution on Datapath

IF Instruction Fetch
- Send address to the instruction memory, and read IMEM at that address.

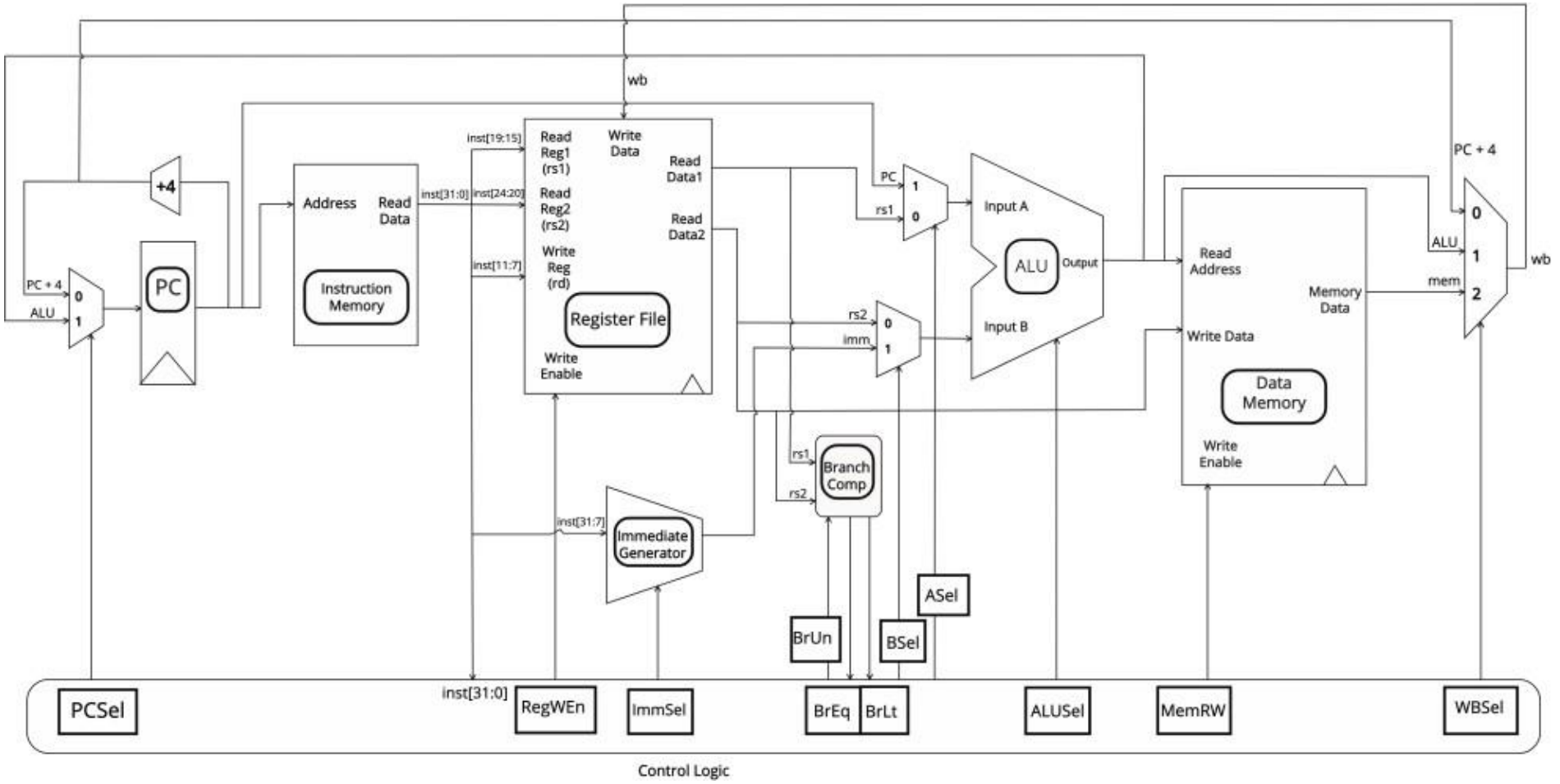
ID Instruction Decode
- Generate control signal from the instruction bits, generate the immediate, and read registers from the RegFile.

EX Execute
- Perform ALU operations, and do branch comparison.

MEM Memory
- Read from or write to the data memory.

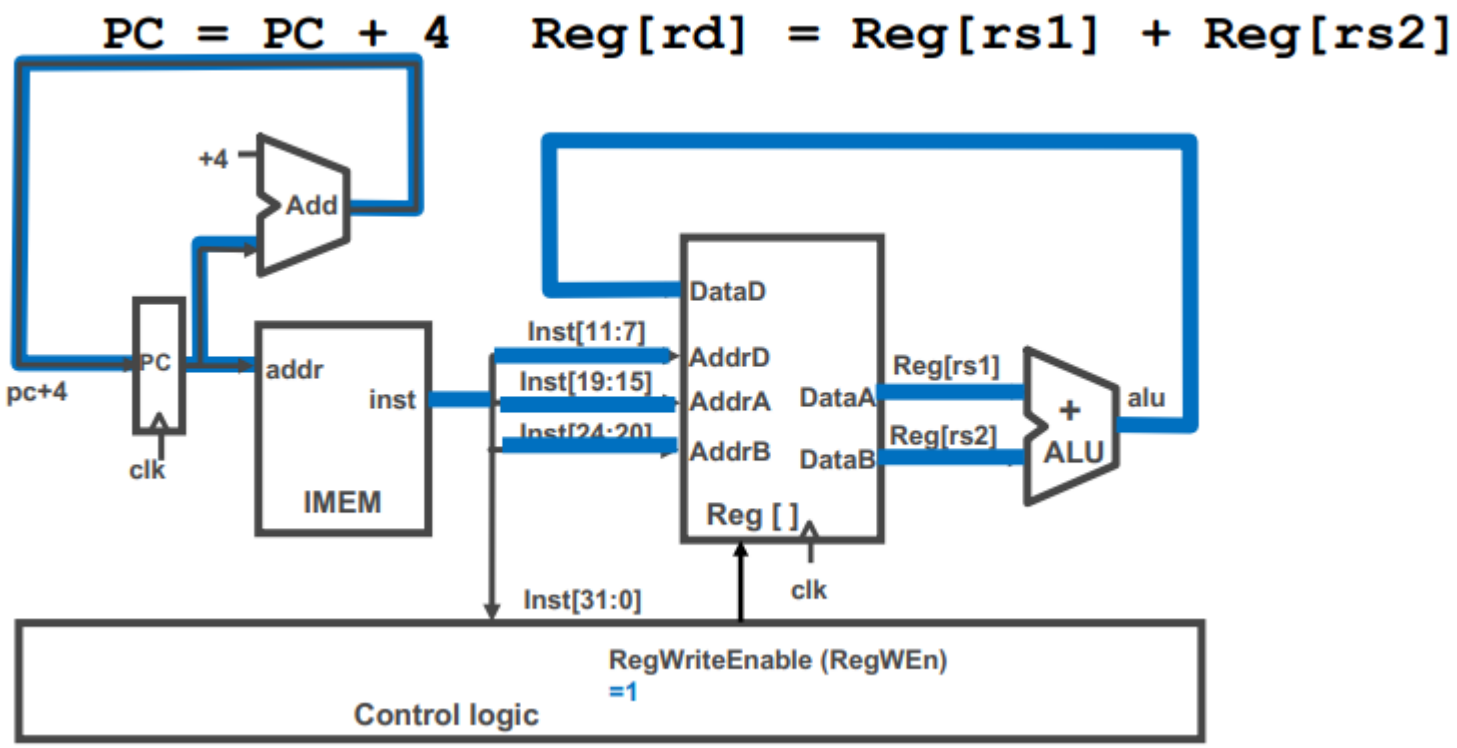
WB Writeback
- Write back the PC+4, the result of the ALU operation, or data from memory to the RegFile.

Stages of Execution on Datapath



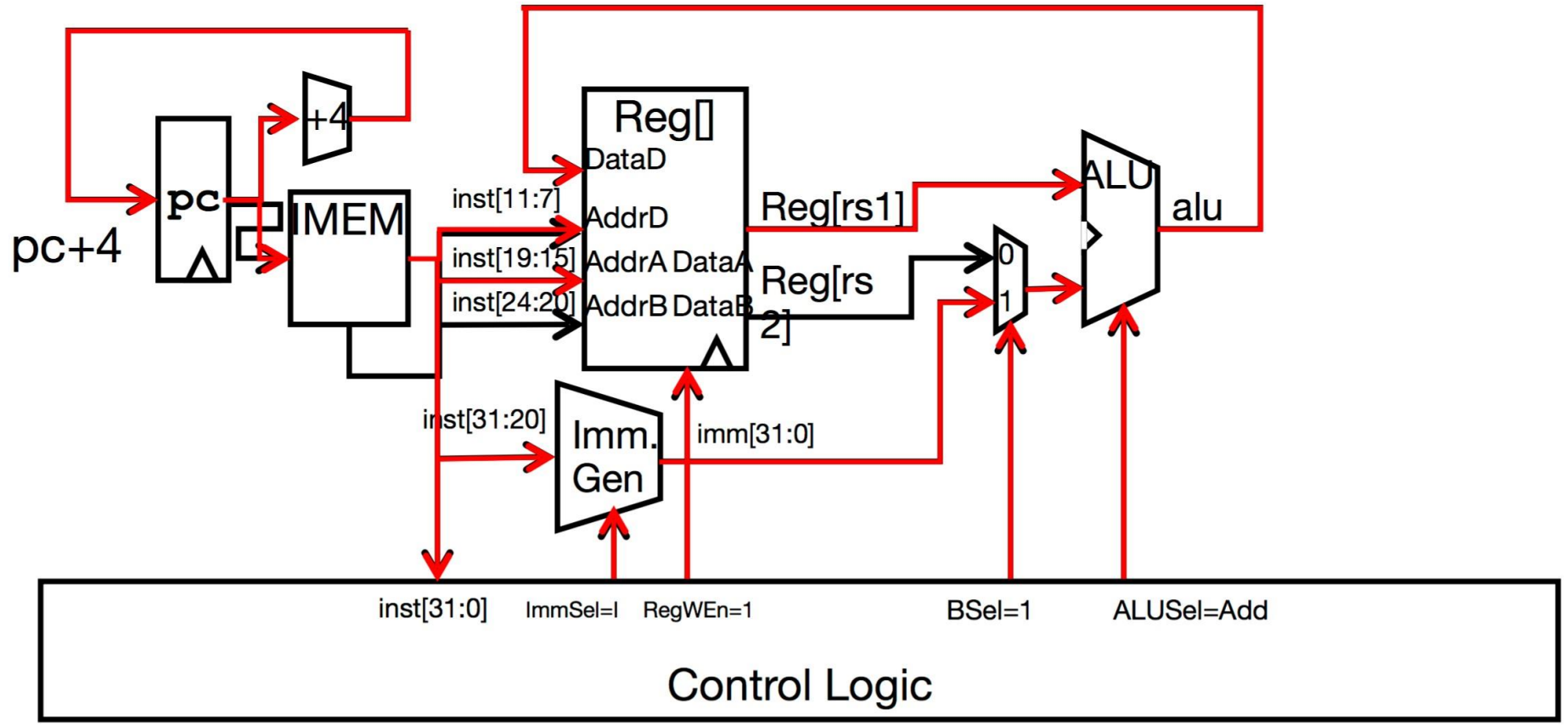
Example 1

- add R R[rd] = R[rs1] + R[rs2], PC = PC + 4



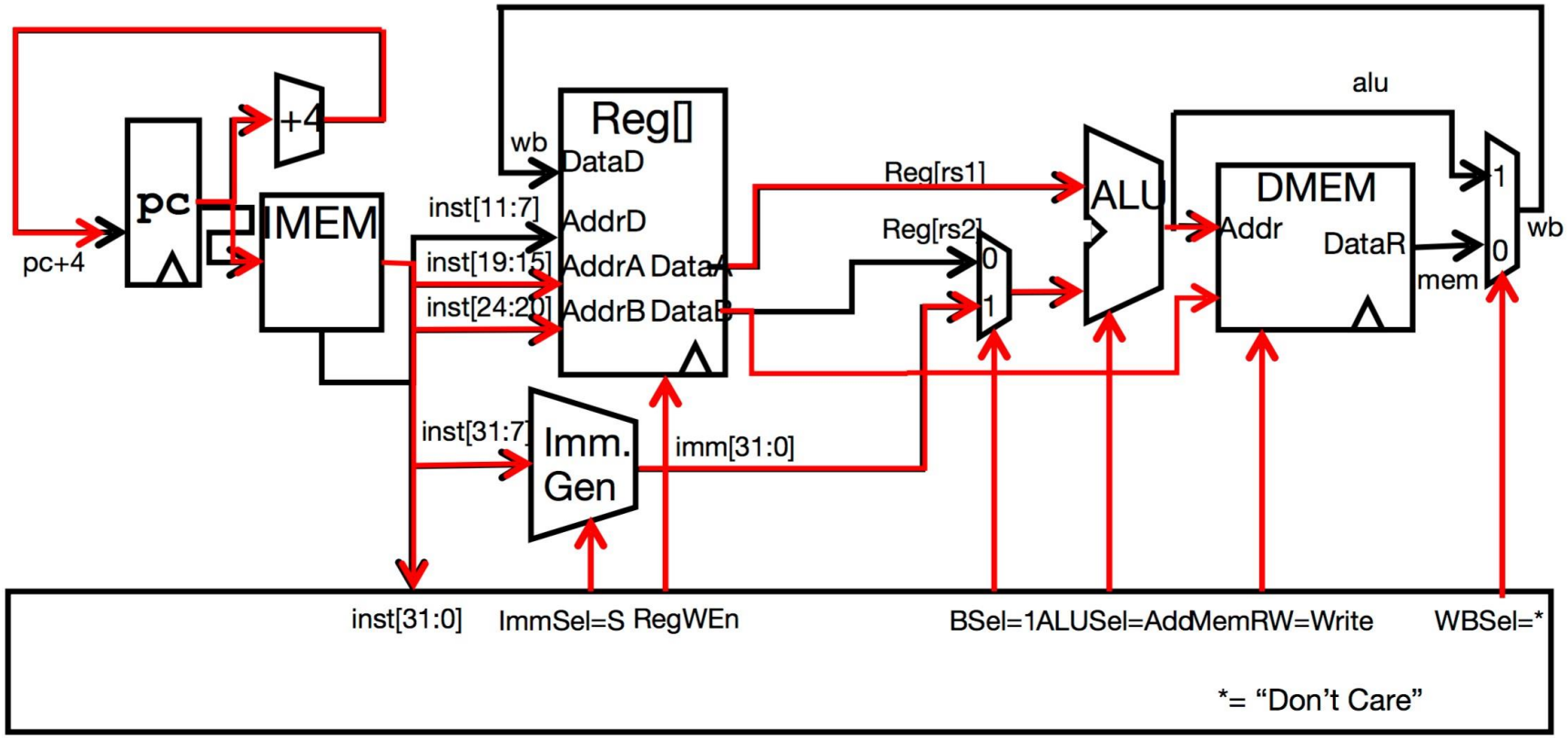
Example 2

- addi I R[rd] = R[rs1] + imm, PC = PC + 4



Example 3

- sw S M[R[rs1]+imm](31:0) = R[rs2](31:0), PC = PC + 4



Some questions

- Which instruction exercise the critical path?
- Why is the single cycle datapath inefficient?
- How can you improve its performance? What is the purpose of pipelining?

- Load word(lw), which uses all 5 stages.
- At any given time, most of the parts of the single cycle datapath are sitting unused. Also, even though not every instruction exercises the critical path, the datapath can only be clocked as fast as the slowest instruction.
- Performance can be improved with pipelining, or putting registers between stages so that the amount of conditional logic between registers is reduced, allowing for a faster clock time.