

# Computer Architecture

---

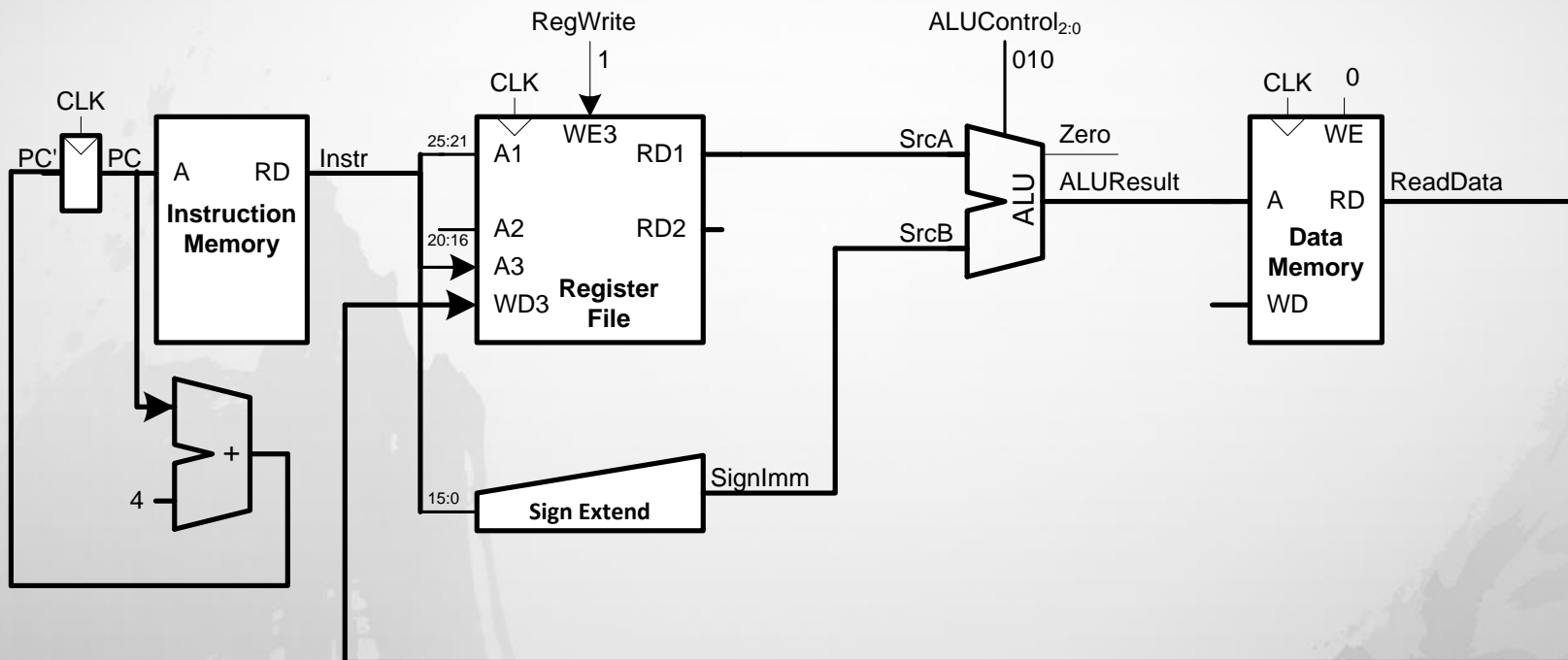
## Lab 7: Controller



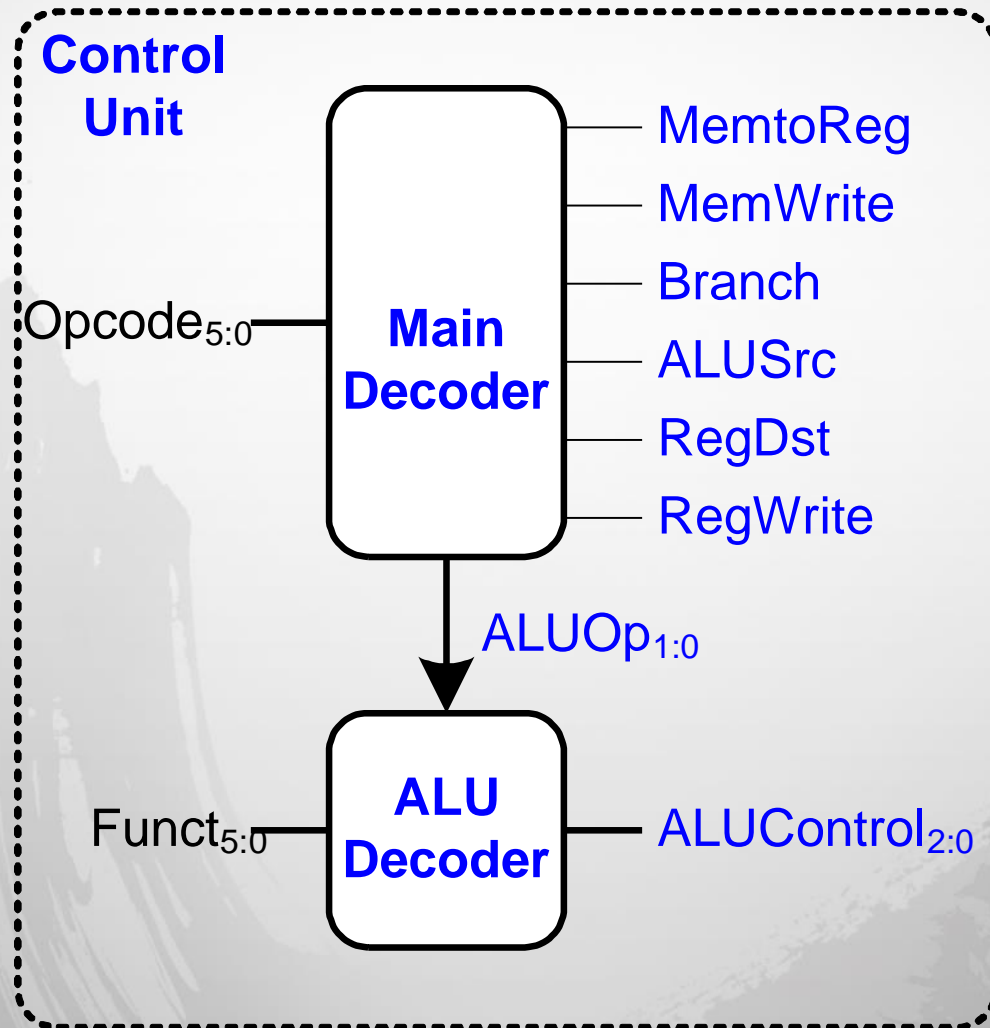
# Getting Started

- Download alu.sv, regfile.sv, imem.sv, dmem.sv and controller.sv from Blackboard
- Open a web browser and go to <https://www.edaplayground.com/>

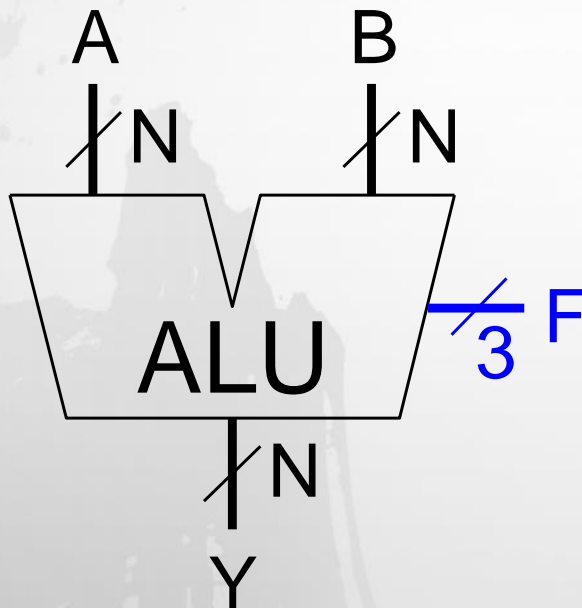
# Datapath for lw Instruction



# Single-Cycle Control



# Review: ALU



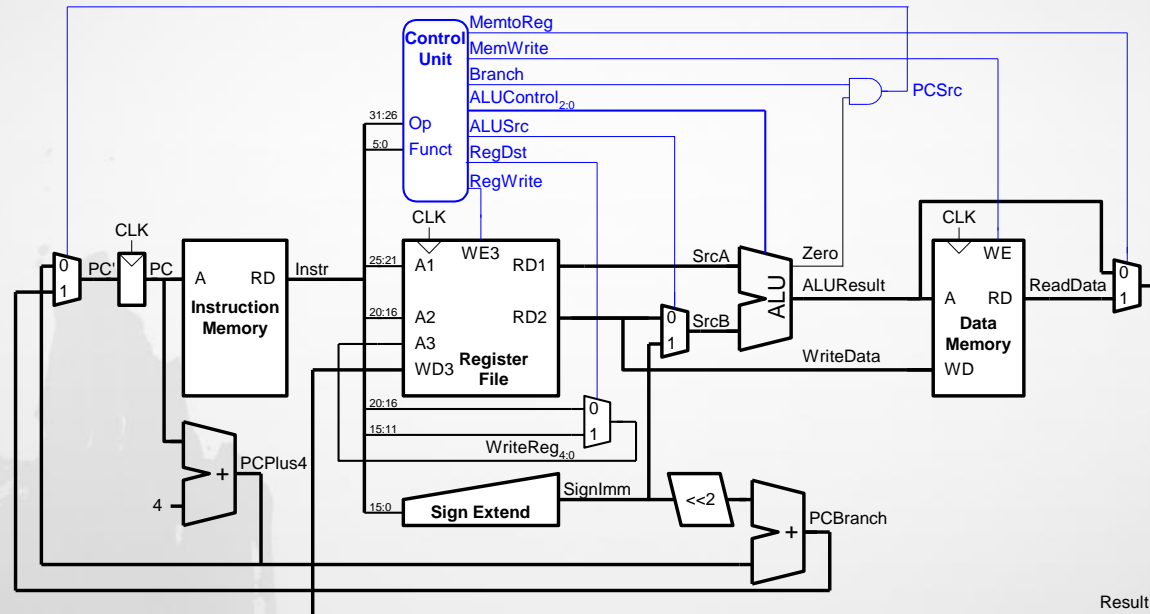
$F_{2:0}$	Function
000	$A \& B$
001	$A   B$
010	$A + B$
011	not used
100	$A \& \sim B$
101	$A   \sim B$
110	$A - B$
111	SLT

# Control Unit: ALU Decoder

ALUOp <sub>1:0</sub>	Meaning
00	Add
01	Subtract
10	Look at Funct
11	Not Used

ALUOp <sub>1:0</sub>	Funct	ALUControl <sub>2:0</sub>
00	X	010 (Add)
X1	X	110 (Subtract)
1X	100000 (add)	010 (Add)
1X	100010 (sub)	110 (Subtract)
1X	100100 (and)	000 (And)
1X	100101 (or)	001 (Or)
1X	101010 (slt)	111 (SLT)

# Control Unit: Main Decoder



Instruction	Op <sub>5:0</sub>	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	ALUOp <sub>1:0</sub>
lw	100011	1	0	1	0	0	1	00

# Testbench

- Copy the following testbench code to "testbench.sv" on the left

```
module testbench_mips();
  logic clk;
  logic reset;

  mips dut(
    .iClk          (clk),
    .iReset        (reset)
  );

  always
  begin
    clk = 1; #5; clk = 0; #5;
  end

  initial begin
    $dumpfile("dump.vcd"); $dumpvars;
    reset = 0; #21;
    reset = 1; #10;
    reset = 0; #10;
    #10; $stop;
  end
endmodule
```



# Datapath

- Copy the following code to "design.sv" on the right

```

`include "alu.sv"
`include "regfile.sv"
`include "imem.sv"
`include "dmem.sv"
`include "controller.sv"

module mips(
  input logic iClk,
  input logic iReset
);

  logic [31:0] ALU_ALUResult;
  logic [31:0] REG_SrcA;
  logic [31:0] IMEM_Inst;
  logic [31:0] DMEM_ReadData;
  logic [31:0] pc;

  logic CTL_RegWrite;
  logic CTL_MemWrite;
  logic [2:0] CTL_ALUControl;

  alu ALU(
    .iA      (REG_SrcA),
    .iB      ({16{IMEM_Inst[15]}}, IMEM_Inst[15:0]),
    .iF      (CTL_ALUControl),
    .oY      (ALU_ALUResult),
    .oZero   ()
  );

  regfile REG(
    .iClk      (iClk),
    .iReset    (iReset),
    .iRaddr1   (IMEM_Inst[25:21]),
    .iRaddr2   (),
    .iWaddr    (IMEM_Inst[20:16]),
    .iWe       (CTL_RegWrite),
    .iWdata    (DMEM_ReadData),
    .oRdata1   (REG_SrcA),
    .oRdata2   ()
  );

  imem IMEM(
    .iAddr      (pc),
    .oRdata     (IMEM_Inst)
  );

  dmem DMEM(
    .iClk      (iClk),
    .iWe       (CTL_MemWrite),
    .iRaddr    (ALU_ALUResult),
    .iWdata    (),
    .oRdata    (DMEM_ReadData)
  );

  controller CTL(
    .iOp      (IMEM_Inst[31:26]),
    .oRegWrite (CTL_RegWrite),
    .oMemWrite (CTL_MemWrite),
    .oALUControl (CTL_ALUControl)
  );

  always_ff@(posedge iClk, posedge iReset)
  if(iReset)
    pc <= 0;
  else
    pc <= pc + 4;


endmodule

```

# Building Blocks

- Click "+" right after "design.sv"
- Upload "alu.sv"
- Upload regfile.sv, imem.sv, dmem.sv, and controller.sv in the same way

# Lab Assignment

- Finish the implementation of the controller module in controller.sv
- Save and submit the link of your design to the Blackboard.
  - Click  in the bottom window to copy the URL of your design.

# Expected Result

