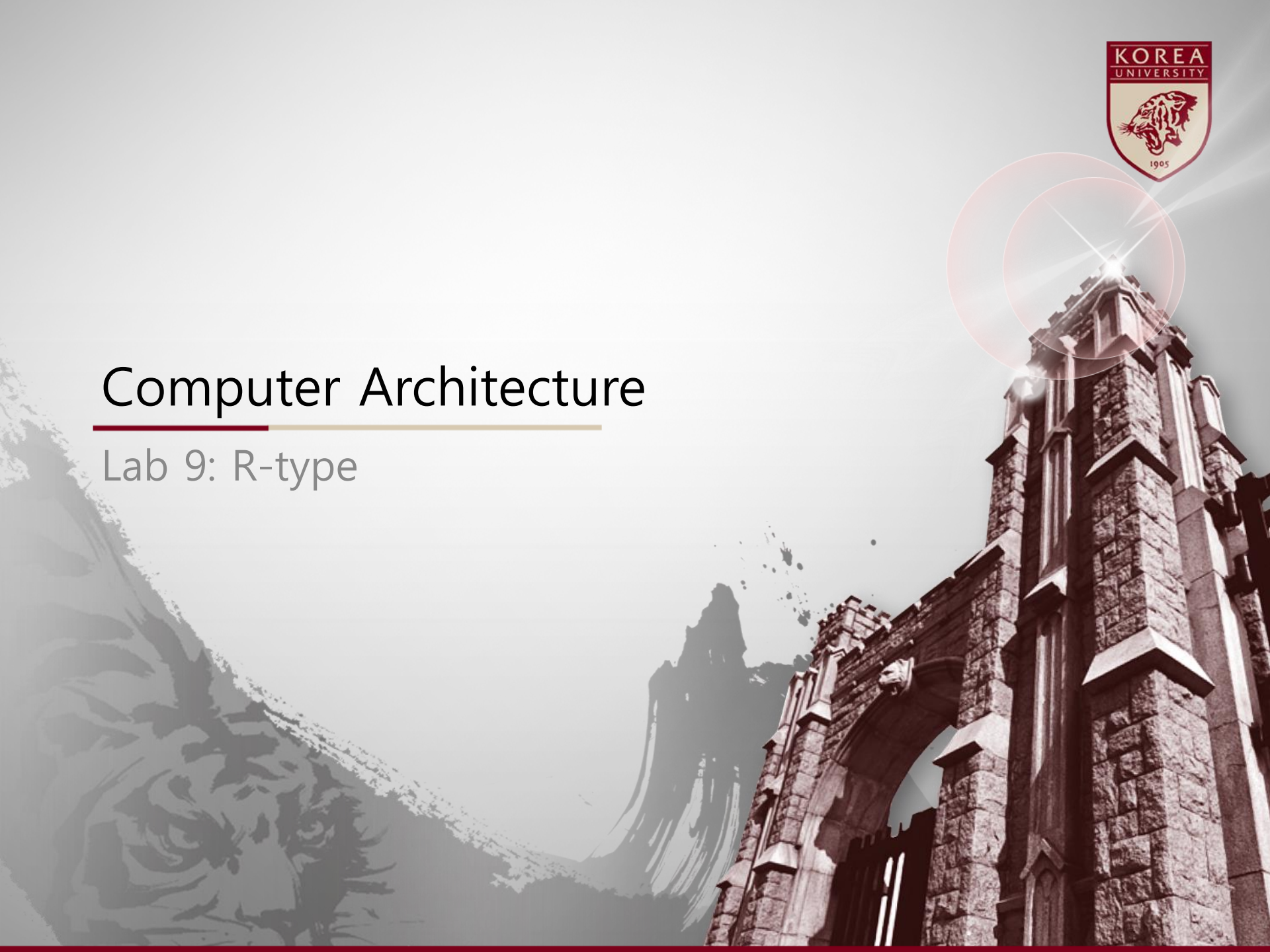


Computer Architecture

Lab 9: R-type

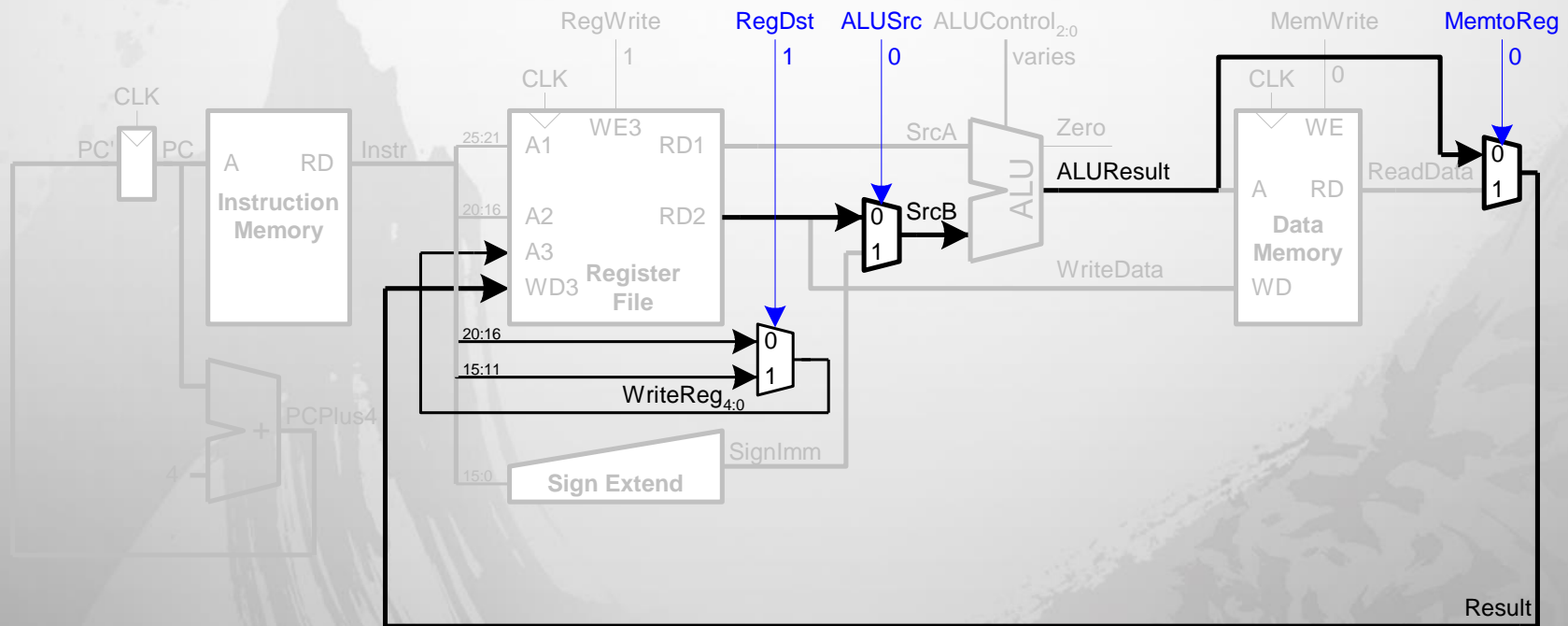


Getting Started

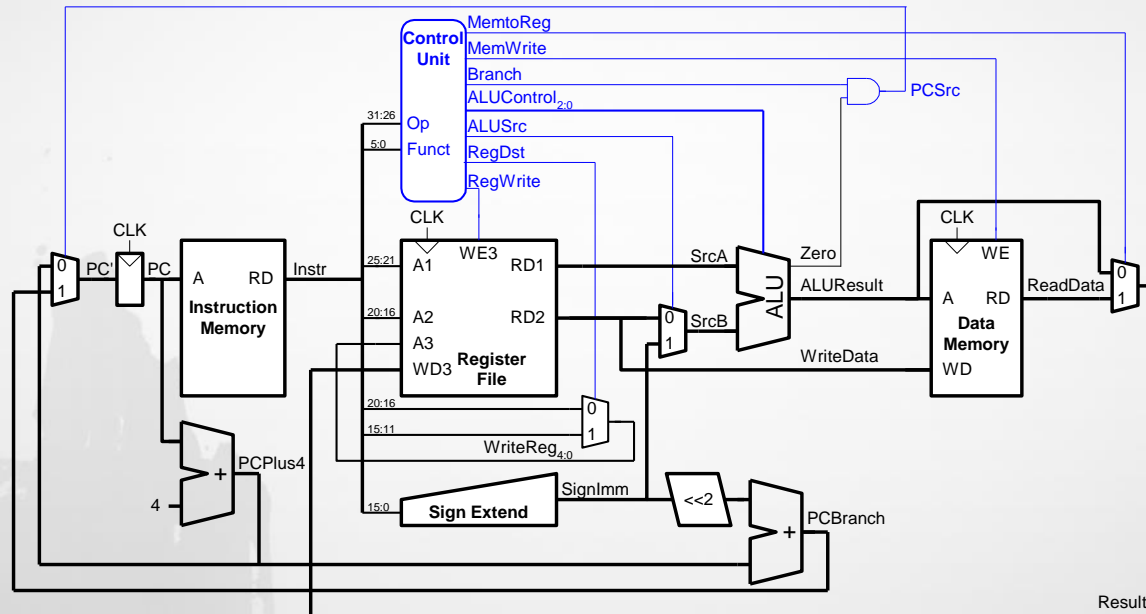
- Download alu.sv, regfile.sv, imem.sv, dmem.sv and controller.sv from Blackboard
- Open a web browser and go to <https://www.edaplayground.com/>

Single-Cycle Datapath: R-Type

- Read from rs and rt
- Write $ALUResult$ to register file
- Write to rd (instead of rt)



Control Unit: Main Decoder



Instruction	Op _{5:0}	RegWrite	RegDst	AluSrc	Branch	MemWrite	MemtoReg	ALUOp _{1:0}
R-type	000000	1	1	0	0	0	0	10
lw	100011	1	0	1	0	0	1	00
sw	101011	0	X	1	0	1	X	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; #20;
    #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] REG_WriteData;
  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(IMEM_Inst[20:16]),
    .iWaddr
      (IMEM_Inst[20:16]),
    .iWe
      (CTL_RegWrite),
    .iWdata
      (DMEM_ReadData),
    .oRdata1(REG_SrcA),
    .oRdata2(REG_WriteData)
  );

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

  dmem DMEM(
    .iClk
      (iClk),
    .iWe
      (CTL_MemWrite),
    .iAddr
      (ALU_ALUResult),
    .iWdata
      (REG_WriteData),
    .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

- Modify SystemVerilog modules to implement the R-type instructions
- 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

