# Computer Organization
# Lab 4 - Pipelined CPU

教授:蔡文錦
助教:劉益先、鄭吉呈、黃芷柔、林彧頎

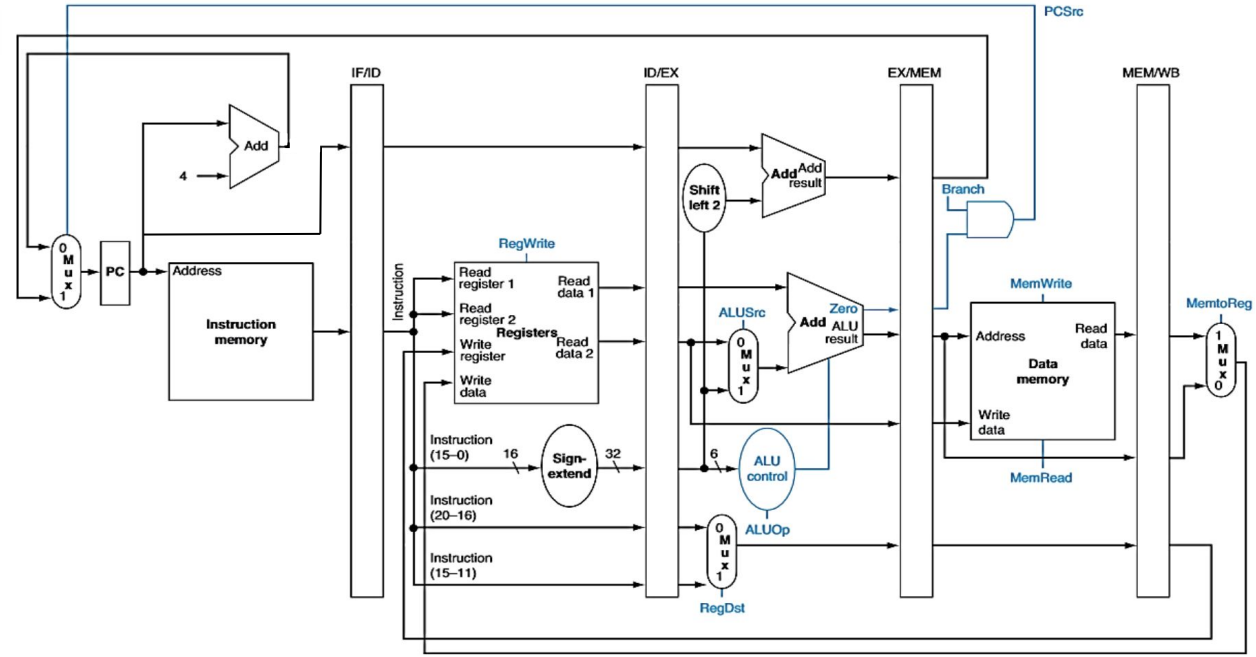
# Objectives

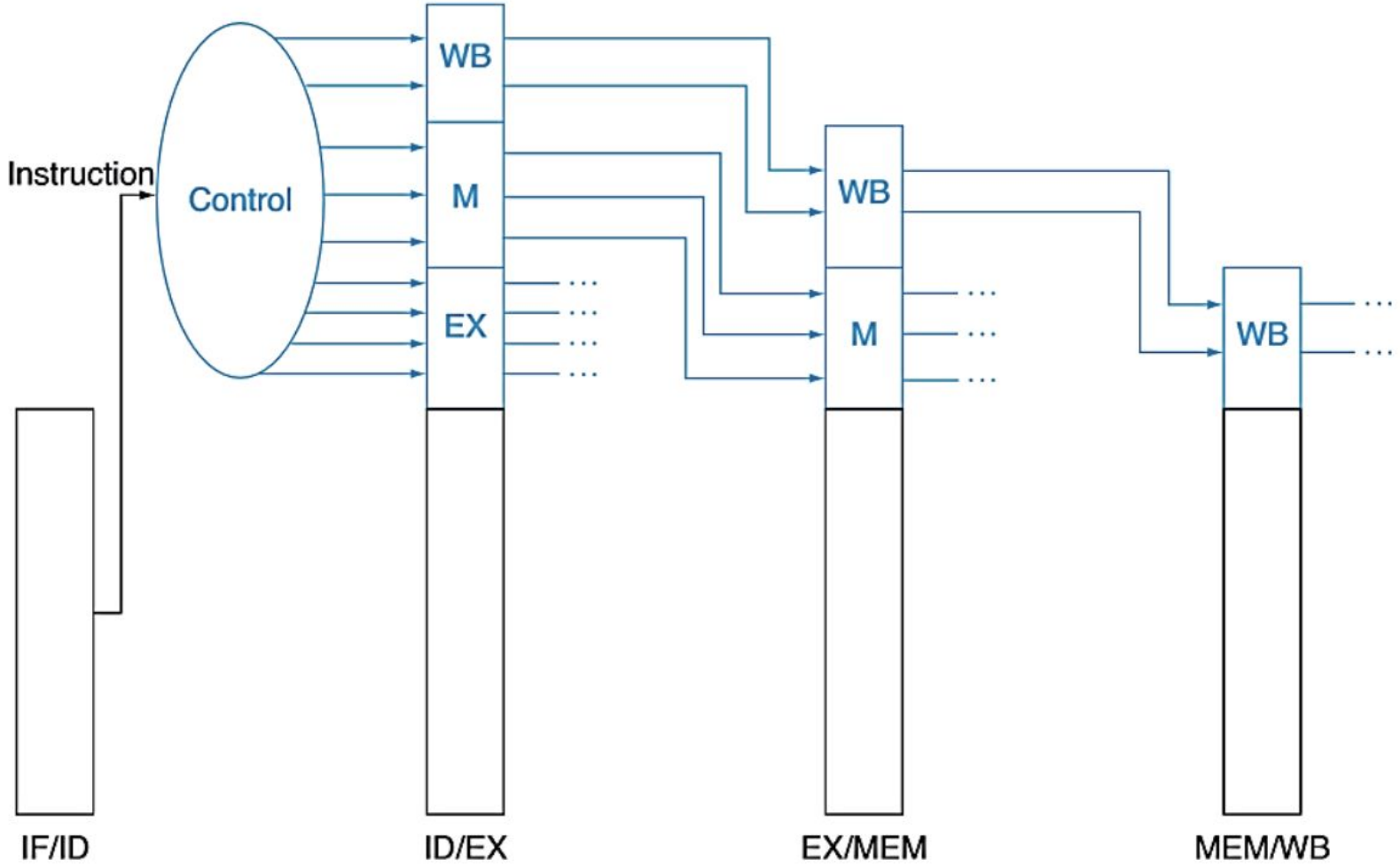
In this lab, you are going to implement a **Pipelined CPU** with memory unit, which can run **R-type** and **I-type** instructions.

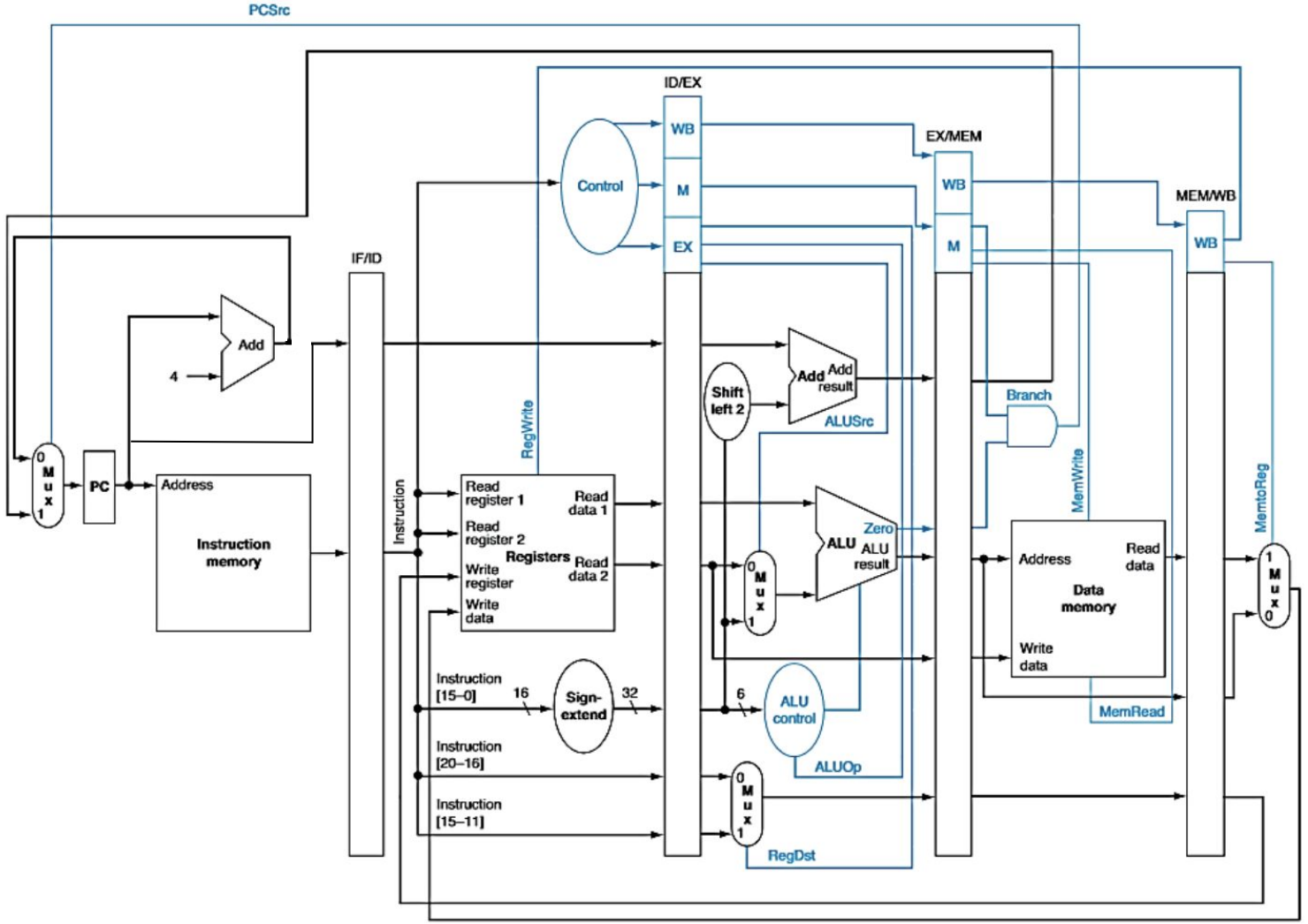
- To realize how to set the control signal in different instruction type.
- To learn how to follow the datapath to form a pipelined CPU

# Overview

The following diagram is the datapath of pipelined CPU.




Instruction
Control
WB
M
EX
WB
M
WB
IF/ID
ID/EX
EX/MEM
MEM/WB


PCSrc
ID/EX
EX/MEM
MEM/WB
WB
M
EX
Control
IF/ID
Add
4
0
M
u
x
1
PC
Address
Instruction memory
Instruction
RegWrite
Read register 1
Read register 2
Registers
Write register
Write data
Read data 1
Read data 2
Shift left 2
Add
Add result
ALUSrc
Branch
MemWrite
MemtoReg
ALU
Zero
ALU result
Address
Read data
Data memory
Write data
MemRead
Instruction [15–0]
16
Sign-extend
32
6
ALU control
ALUOp
Instruction [20–16]
Instruction [15–11]
RegDst
1
M
u
x
0

# Attached Files

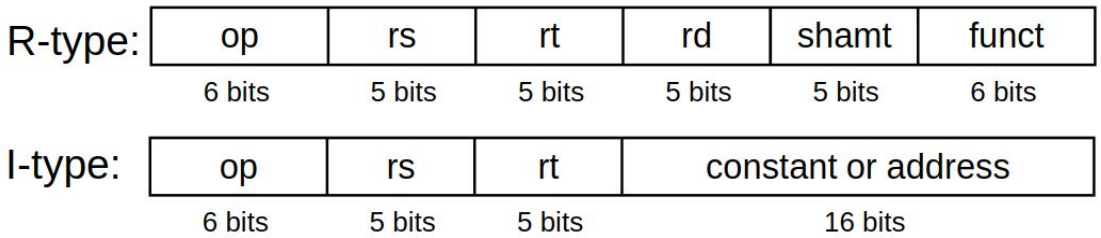
- **TO DO**
  - Adder.v
  - ALU_Ctrl.v
  - ALU.v
  - Decoder.v
  - MUX_2to1.v
  - Pipe_CPU.v
  - Shift_Left_Two_32.v
  - Sign_Extend.v
- **DO NOT modify**
  - Data_Memory.v
  - Instruction_Memory.v
  - Pipe_Reg.v
  - ProgramCounter.v
  - Reg_File.v
- **For validation - DO NOT modify**
  - testbench.v
- **Testcase - YOU CAN modify the instructions in it.**
  - *.txt

# Instruction Set

You are going to implement these instructions :

- **R-type :** add, sub, AND, OR, NOR, slt
- **I-type :** lw, sw, beq, bne, addi

Instruction Format:



# Instruction Set

| R-type | | |
|---|---|---|
| **Function** | **Op Field** | **Function Field** |
| add | 6’b000000 | 6’b100011 |
| sub | 6’b000000 | 6’b100001 |
| AND | 6’b000000 | 6’b100110 |
| OR | 6’b000000 | 6’b100101 |
| NOR | 6’b000000 | 6’b101011 |
| slt | 6’b000000 | 6’b101000 |

| I-type | |
|---|---|
| **Function** | **Op Field** |
| addi | 6b’001001 |
| lw | 6b’101100 |
| sw | 6b’100100 |
| beq | 6b’000110 |
| bne | 6b’000101 |

# Compile & Run

- **Compile**
  - $ iverilog -o lab4 testbench.v
- **Run**
  - $ ./lab4
  - (windows)$ vvp lab4

**Wrong results:**

```
************************************************
* Register Error! [Register 31]                *
* Correct result: 00000050                     *
* Your result:    00000000                     *
************************************************
************************************************
*               You have 19 error !            *
************************************************
```

**Correct results:**

```
************************************************
*          Congratulation. ALL PASS !          *
************************************************
```

# Grading Policy

- There are **3 hidden cases with serial several instructions**, and you will get **33** points for each correct testcase, with **an additional point** for submitting, totally **100** points.
- **Any assignment work by fraud will get a zero point !**
- **No late submission !**

# Submission

- **Please attach student IDs as comments at the top of each TO DO file.**
- The files you should hand in include:
  - **all *.v files excluding testbench.v**
- Compress all file ***.v** into one zip file **without any extra folder layer**, and **make sure do not add unnecessary files or folders** (like .DS_Store, __MACOSX).
- Name your zip file as **HW4_{studentID}.zip**
  - **e.g.**
    - **HW4_123456789.zip**
      - **{*.v}**
- **Wrong format will have 20% penalty！**
- **Deadline: 8/18 23:55**