

# Homework #3: RISC-V Pipeline

Prof. Jae W. Lee ([jaewlee@snu.ac.kr](mailto:jaewlee@snu.ac.kr))

Department of Computer Science and Engineering

Seoul National University

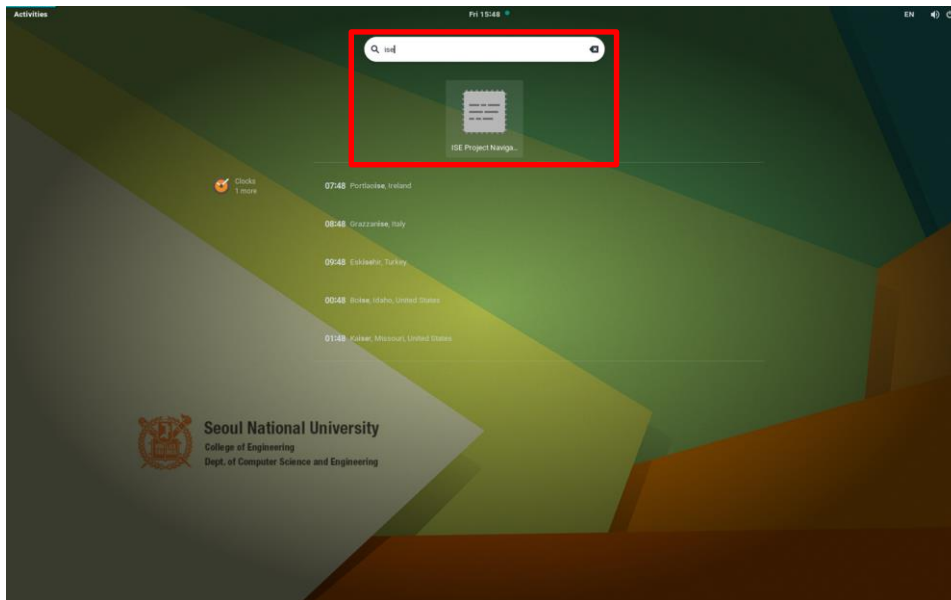
TA: Jeonghun Gong, Yunho Jin

# Goal of this project

- **You implement missing features in a RISC-V (64-bit) pipeline.**
  - Add additional bitwise operations (AND, OR, XOR, ANDi, ORi, XORi).
  - Implement a conditional branch (BEQ instruction).
  - Optimize the datapath by implementing:
    - Hazard detection unit
    - Forwarding unit

# Experimental setup

- You will use Xilinx ISE for Verilog simulation.
  - Same program that you used at last semester's logic design course.
  - Download link: Click [HERE](#).
  - You may use computers at 302-310-2 (Hardware Lab), where Xilinx ISE is pre-installed.



# Create a project

- **File -> New project**
  - Set “Top-level source type” to HDL.

New Project Wizard

Create New Project  
Specify project location and type.

Enter a name, locations, and comment for the project

Name: riscv-pipeline

Location: /jeonghun/arc/Lectures/2019Fall/ugradCA/PA3/riscv-pipeline

Working Directory: /jeonghun/arc/Lectures/2019Fall/ugradCA/PA3/riscv-pipeline

Description:

Select the type of top-level source for the project

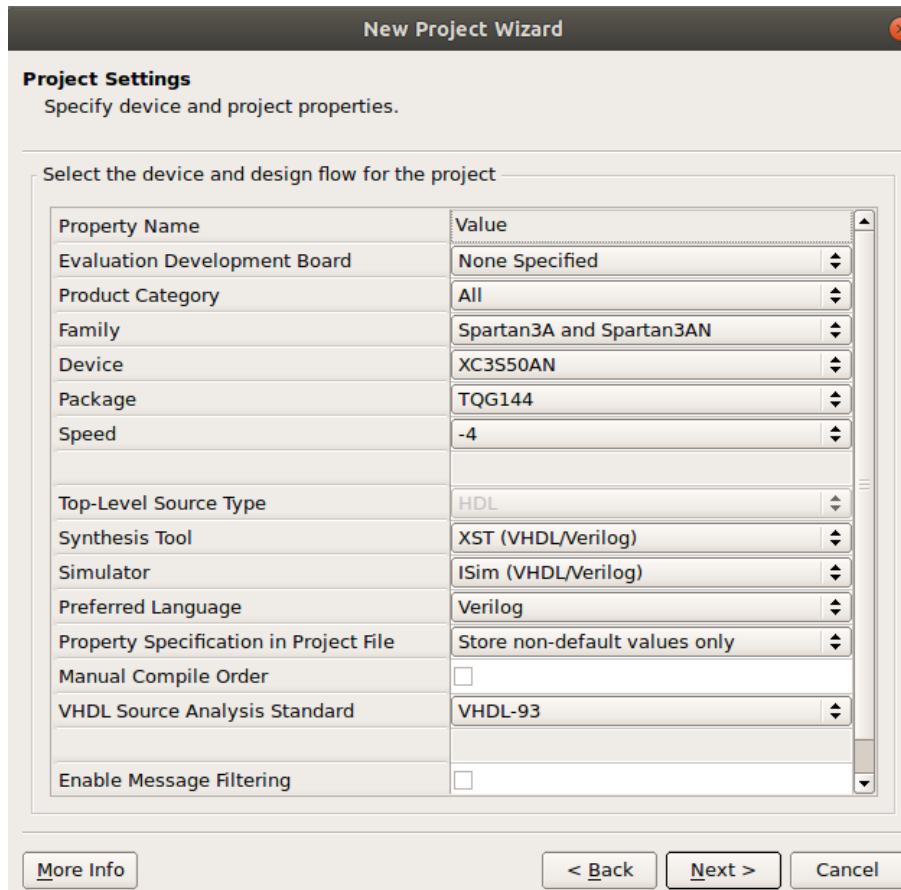
Top-level source type:  
HDL

More Info Next > Cancel

# Create a project

## ■ Project settings

- Set as shown below.

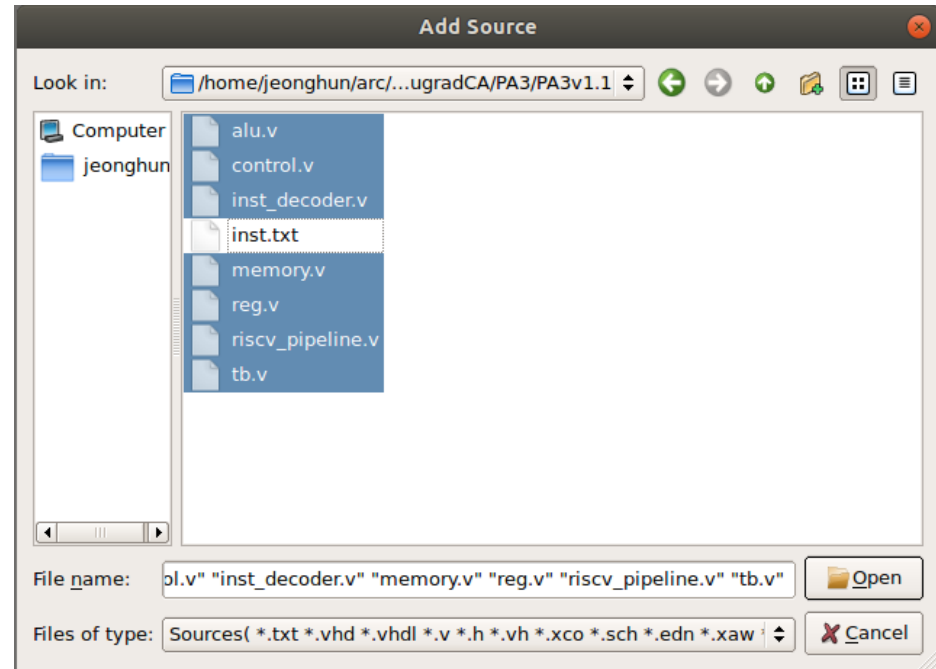
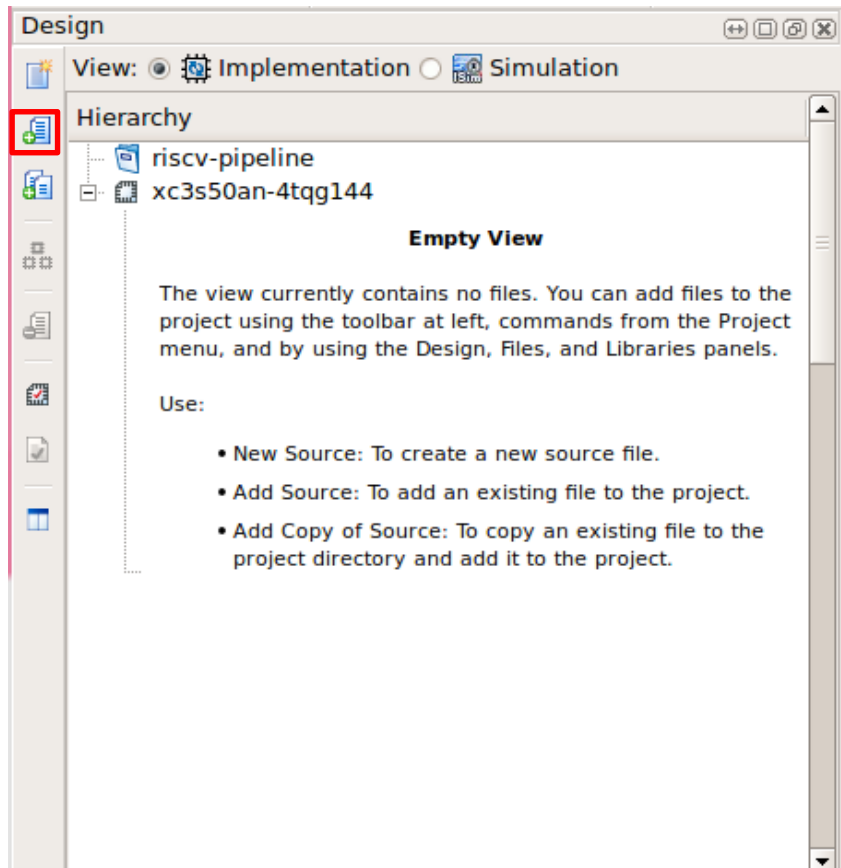


The image shows the 'New Project Wizard' dialog box, specifically the 'Project Settings' step. The title bar reads 'New Project Wizard'. Below the title bar, the text 'Project Settings' is followed by the instruction 'Specify device and project properties.' The main area is titled 'Select the device and design flow for the project' and contains a table of properties. The properties are listed in two columns: 'Property Name' and 'Value'. The values are set as follows: Evaluation Development Board (None Specified), Product Category (All), Family (Spartan3A and Spartan3AN), Device (XC3S50AN), Package (TQG144), Speed (-4), Top-Level Source Type (HDL), Synthesis Tool (XST (VHDL/Verilog)), Simulator (ISim (VHDL/Verilog)), Preferred Language (Verilog), Property Specification in Project File (Store non-default values only), Manual Compile Order (unchecked), VHDL Source Analysis Standard (VHDL-93), and Enable Message Filtering (unchecked). At the bottom, there are three buttons: 'More Info', '< Back', and 'Next >', and a 'Cancel' button.

Property Name	Value
Evaluation Development Board	None Specified
Product Category	All
Family	Spartan3A and Spartan3AN
Device	XC3S50AN
Package	TQG144
Speed	-4
Top-Level Source Type	HDL
Synthesis Tool	XST (VHDL/Verilog)
Simulator	ISim (VHDL/Verilog)
Preferred Language	Verilog
Property Specification in Project File	Store non-default values only
Manual Compile Order	<input type="checkbox"/>
VHDL Source Analysis Standard	VHDL-93
Enable Message Filtering	<input type="checkbox"/>

# Add sources to project

- Use a design tab at the upper left corner



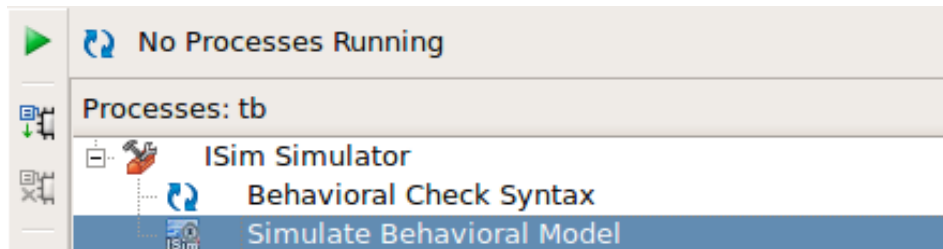
# Running simulation

## ■ Specify your input program image at memory.v

- File's location is relative to the project folder.

```
module rom64
(
    input    [63:0] address,
    output   [31:0] data_out
);
    parameter FILE = "inst.dat";
    parameter ROM64_BITMASK = 64'h1fc;
```

## ■ Click “Simulate Behavioral Model” to start simulation.



## ■ Check the console below to check simulation results

```
Finished circuit initialization process.
reg[ 1] = 0x000000000000000a0
reg[ 2] = 0x000000000000000ff
reg[ 3] = 0x0000000000000007b
reg[ 4] = 0x00000000000000011b
reg[ 2] = 0x00000000000000021a
reg[ 5] = 0x000000000000000010
M[ 2] = 0x00000000000000011b
reg[ 7] = 0x00000000000000011b
Stopped at time : 180 ns : File "/home/jeonghun/arc/Lectures/2019Fall/ugradCA/PA3/PA3v2.0/src/riscv_pipeline.v" Line 230
ISim>
```

# Baseline pipeline given

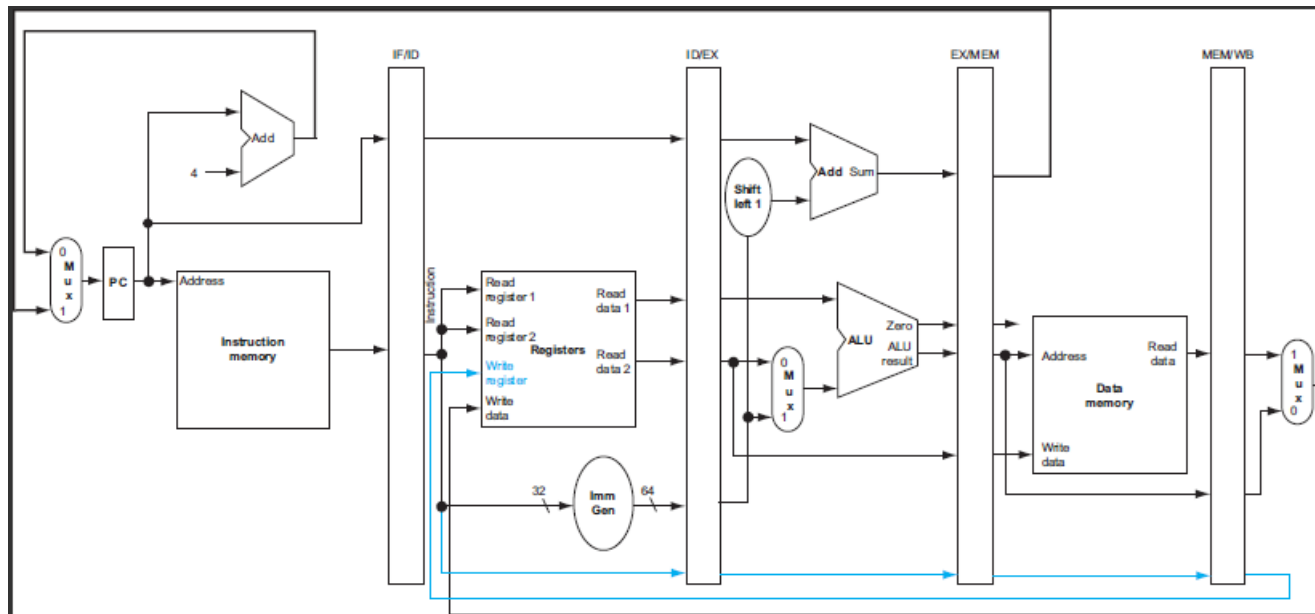
## ■ Supported instructions

Arithmetic instructions: ADD, SUB, ADDi

Memory instructions: LD, SD

Other instruction: HALT (Stops simulation)

## ■ Hazard detection, forwarding are NOT supported.





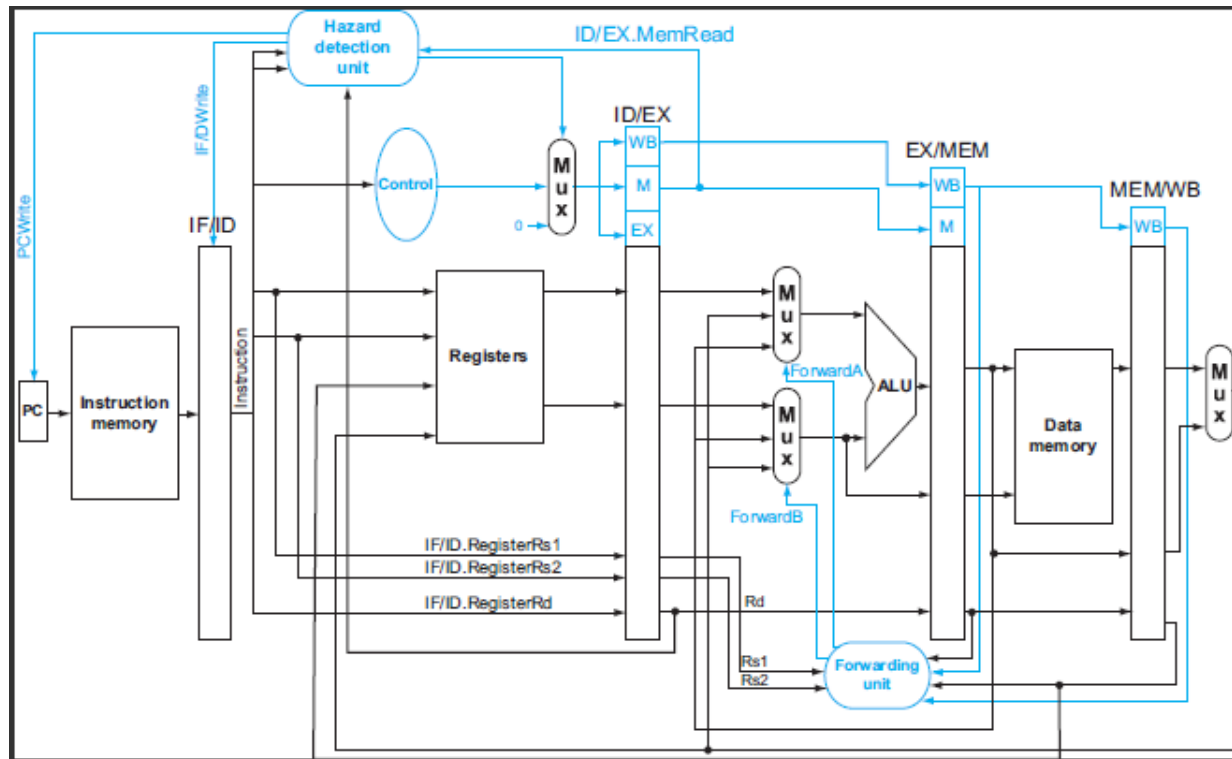
# Problem 1. Bit instructions

- You should add AND, OR, XOR, ANDi, ORi, and XORi instructions.
  - Refer to RISC-V reference sheet for instruction encoding.
  - Use given bit.dat file for test.

# Problem 2. Hazard detection

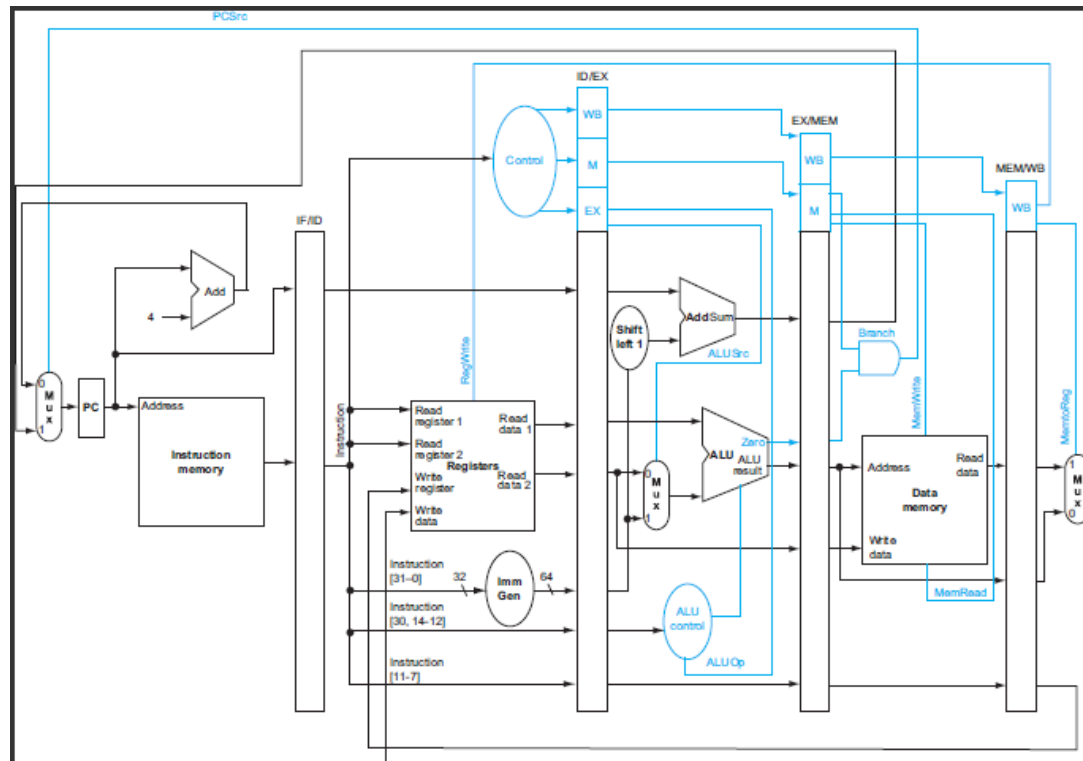
## ■ Implement a hazard detection unit

- It should stall your pipeline when needed.
- The baseline needs explicit NOP instructions to resolve RAW hazard.
- Test your implementation for correctness using hazard\_ctl.dat.



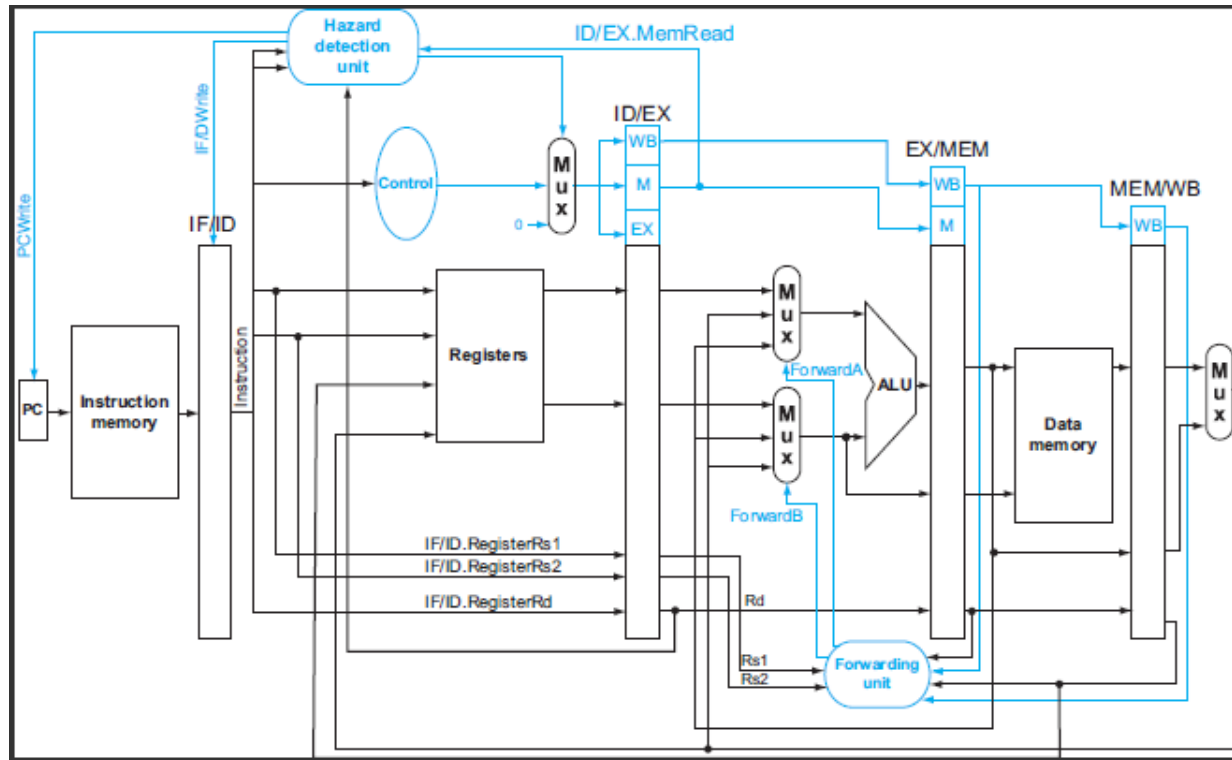
# Problem 3. Branch instruction (BEQ)

- Your pipeline should support BEQ instruction.
  - Refer to RISC-V reference sheet for instruction encoding.
  - Pipeline should be stalled properly to resolve control hazards.
  - Use given branch.dat file for test.



# Problem 4. Forwarding unit

- Implement forwarding unit for EX/MEM result.
  - The pipeline should operate as fast as or faster than TA's implementation.



# Test with your own code

## ■ Generate assembly code with RISC-V compiler

- Write your code in C
- Compile with -S (uppercase) option to generate assembly code.  
`$> riscv64-unknown-elf-gcc -S [input_file]`
- Assembly code (\*.s) is generated.
- However, note that our pipeline only supports a subset of the RISC-V instruction set.

# Test with your own code

## ■ Use provided assembler (G++ is required).

- It generates binary for your pipeline.
- List of supported instructions
  - Arithmetic operations: ADD(i), SUB
  - Bitwise operations: AND(i), OR(i), XOR(i)
  - Memory operations: LD, SD
  - Branch operation: BEQ
  - Others: NOP, HALT
- Branch to label is not supported
  - Only branch to offset is supported.
- Build: `$> make`

```
jeonghun@NEETProduction:~/riscv-assembler$ make
g++ -Wall -Werror -I. -c main.cpp -o main.o
g++ -Wall -Werror -I. -c riscv-assembler.cpp -o riscv-assembler.o
g++ main.o riscv-assembler.o -o riscv-assembler
```

- Usage: `$> ./riscv-assembler [input_file] > [output_file]`

```
jeonghun@NEETProduction:~/riscv-assembler$ ./riscv-assembler input.s > inst.dat
```

# Grading policy

- **Problem 1: 15%**
- **Problem 2: 20%**
- **Problem 3: 25%**
- **Problem 4: 30%**
- **Writeup: 10%**
- **For late submission:**
  - Within next 24 hours: 10% deduction
  - Within next 48 hours: 30% deduction
  - Within next 72 hours: 50% deduction
  - After next 72 hours: Submission not accepted

# Submission Guidelines

- **What you have to submit:**

- riscv\_pipeline.v, control.v, inst\_decoder.v

- **You may add new files to your project.**

- If that's the case, please specify it on your writeup. Otherwise, they might be ignored.

- **Do not modify these files:**

- tb.v, alu.v, reg.v, memory.v,
- Also, do not include them in your submission.



# Submission Guidelines

## ■ Writeup

Briefly describe your implementation (no more than 5 pages).

Filename: [student\_id].pdf (example: 2019-12345.pdf)

**Please** submit it in **PDF** format. Other formats are not accepted.

## ■ Compress your source code and report into single zip file.

Refer to the previous slide for files to submit.

Filename should be [student\_id].zip (example: 2019-12345.zip).

**Please** submit it in **ZIP** format. Other formats are not accepted.

## ■ Submission deadline: By 2019. 11. 01 23:59 KST