# Control Unit

# IF/ID

# TABLE OF CONTENTS

# 1. IMPLEMENTATION OF CONTROL UNIT

| | Control Signals | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Instruction | OpCode | Reg Dst | ALU Src | Memto Reg | Reg Write | Mem Read | Mem Write | Branch | ALUOp |
| R-type | 000000 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 000 |
| lw | 000100 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 011 |
| sw | 000101 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 011 |
| addi | 000111 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 011 |
| beq | 000110 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 001 |
| slti | 000001 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 010 |

# 2. CONDITIONAL OPERATOR (NESTED)

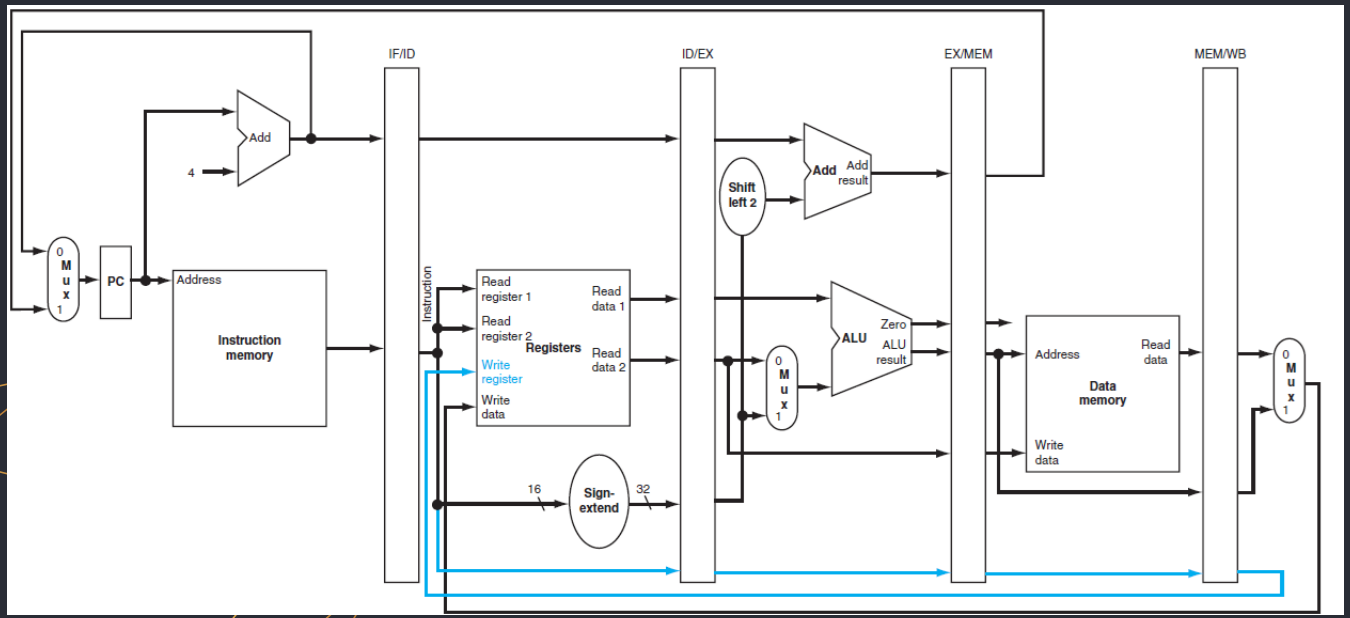
Just like the ternary operator in C (?:)

Example:

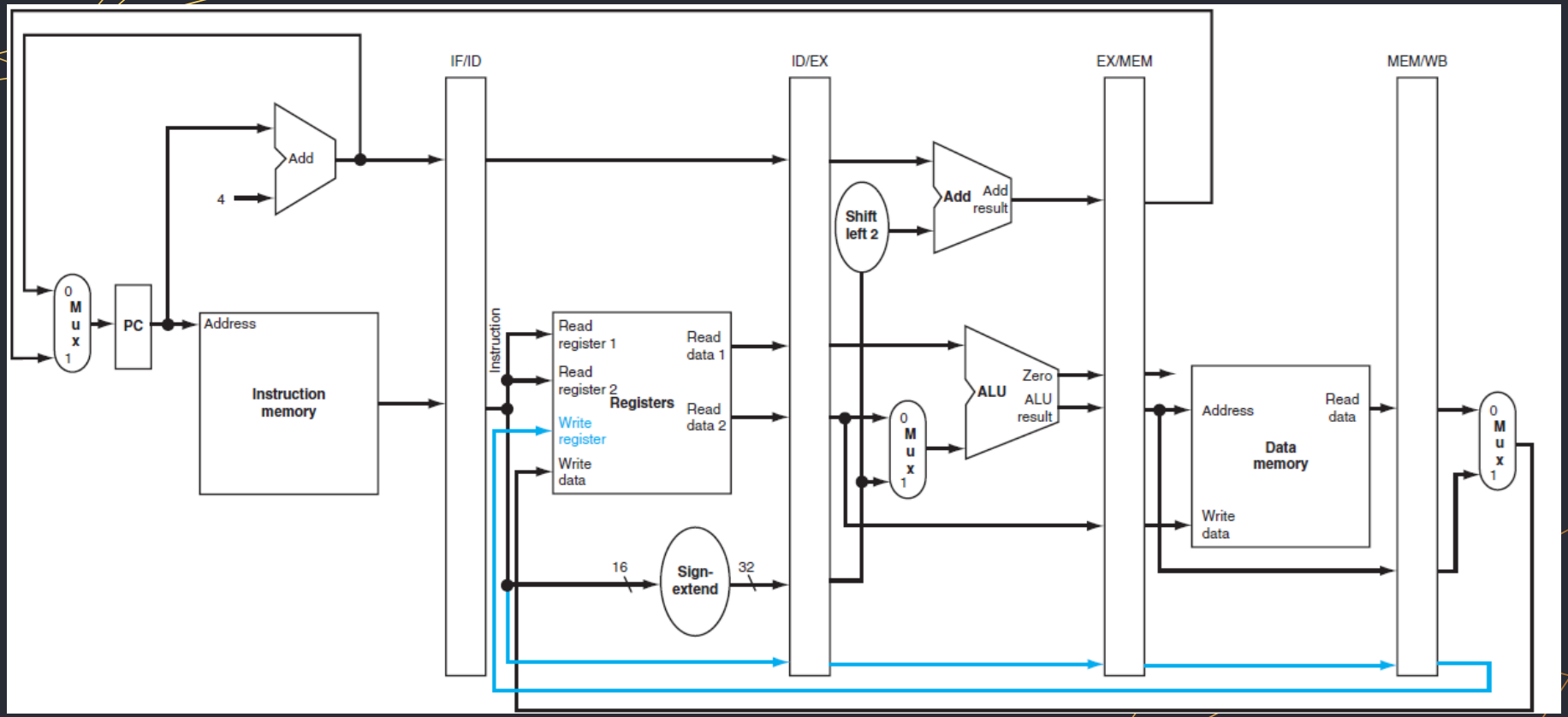
```
assign out = sel == 2'b00 ? a :
             sel == 2'b01 ? b :
             sel == 2'b10 ? c:
             sel == 2'b11 ? d : 1'b0;
```

# 3. INTRODUCTION TO PIPELINE REGISTERS

- Placed between Fetch & Decode stages
- The outputs are produced on the positive edge of the clock in the stages, therefore,  to make sure the data from pipeline registers are reliable, they need to output on the negative edge of the clock



# 4. INTEGRATING ALL MODULES



# 5. INPUTS & OUTPUTS

Finally, it's time to instantiate the modules and connect them together. Modules which need to be instantiated and connected are as follows:

**Control Unit**:

- Inputs: Opcode (6 bits)
- Outputs: RegDst (1), ALUSrc (1), MemtoReg (1), RegWrite (1), MemRead (1), MemWrite (1), Branch (1), ALUOp (3)

**IF/ID Register**:

- Inputs:clk(1),  next_pc (32), instruction (32), hit (1)
- Outputs: instruction_out (32), next_pc_out (3), hit_out (1)

**MIPS: (Instantiate Fetch, Decode, Control Unit)**