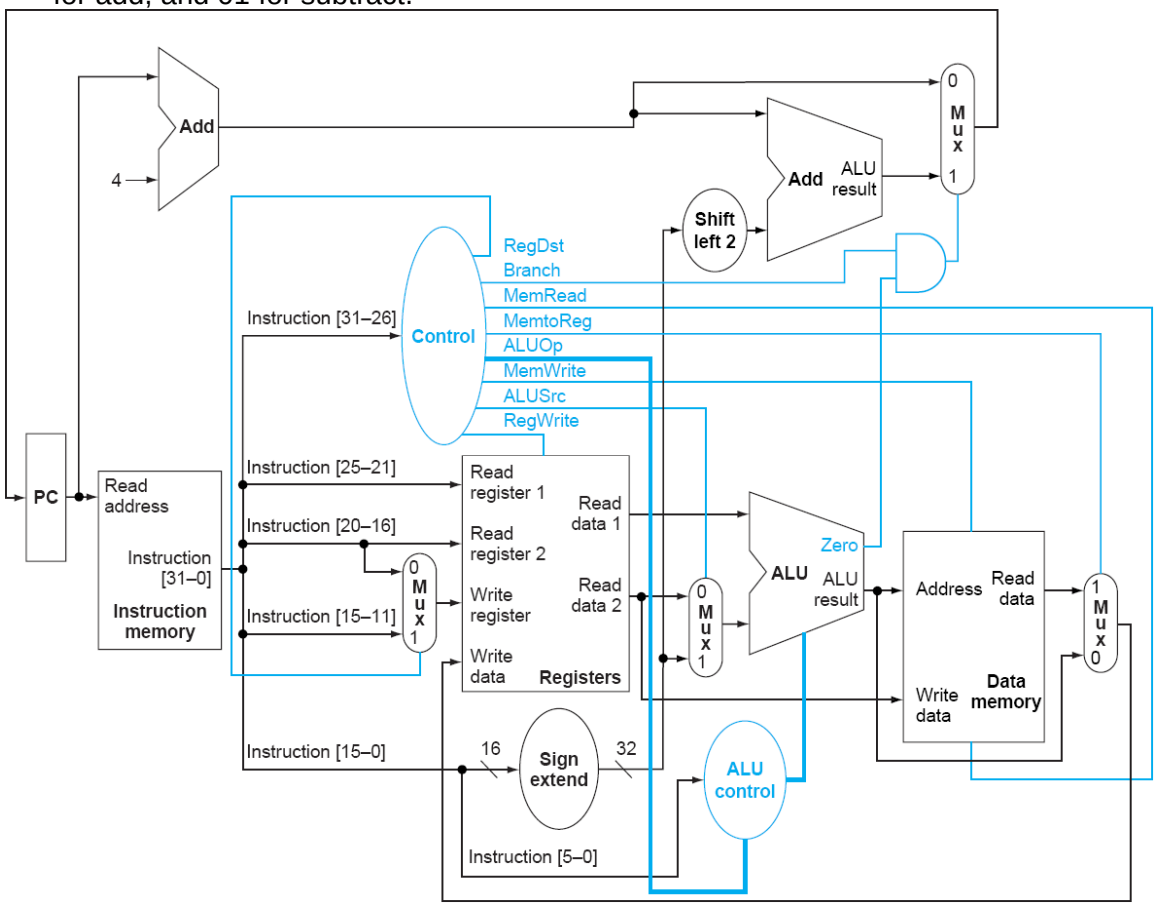
1. Provide all control signals for a ADD instruction using the following processor design. Assume ALUOp is 10 for interpretation of instruction function code, 00 for add, and 01 for subtract.



RegDst	__1__	MemWrite	__0__
Branch	__0__	ALUSrc	__0__
MemRead	__0__	RegWrite	__1__
MemtoReg	__0__		
ALUOp	_00__		

2. Describe how the above design would need to be modified in order to add the **JR** instruction. Recall that this instruction moves the contents of the register addressed by the "*rs*" field (bits 25 down to 21) into the program counter (PC). It is an R-type instruction.

The "read data 1" output from the register file would need to feed into the PC. Since the PC is already receiving its input from the "branch" mux, another mux would need to be inserted in front of the PC.

3. Describe briefly why the MIPS designers use a separate ALU controller rather than centralize all control in the main control unit?

Because all R-type instructions share the same control state, except for the ALU operation.

4. Draw the design of a simple datapath that multiplies the contents of register **A** by 3 and latches the value into another register **B**. This datapath can be represented by the following RTL:

```
B <- (A) * 3
```

You are provided with a library of the following components:

- **register**
- a combinational logic block named "**shift-left by one**", that provides an output whose value is the input value shifted to the left by one bit
- **adder**

Assume these components may be scaled to any arbitrary bit width, so you may abstract away the size of each of the signals.