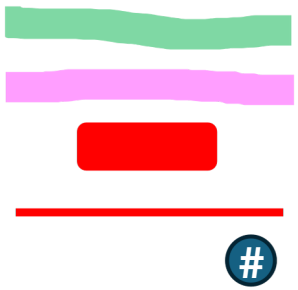
Homework 6 - Directions

1. For this homework you will be highlighting the datapath through the single cycle RISC-V design for 3 instructions. Note that I did Question 0 to show you want I want for the jr datapath. Notice how the data path itself is highlighted, the control signals are specified, and a brief writeup is provided showing the key steps that happen from fetching the jr instruction, all the way through the data path, until ultimately the PC is updated

2. You will be doing the same for the sw instruction, the jal instruction, and the lui instruction. Please consult the datasheet it will help you to setup the register file.

3. For each instruction you are allowed to cut wires, add any new wires, add new connections to existing wires, and to add additional components that we have used in Logisim – aka, multiplexors, adders, shifters, demux, etc. See the example I provided for jr as an example. Don't forget about the control signals, how should they be set? Does your instruction need new control signal(s)?

4. You can mark up and annotate each datapath any way that you want – on the computer, by hand, take pictures, etc. But your changes must be clear and easy to follow to receive full credit. On my sample I used:



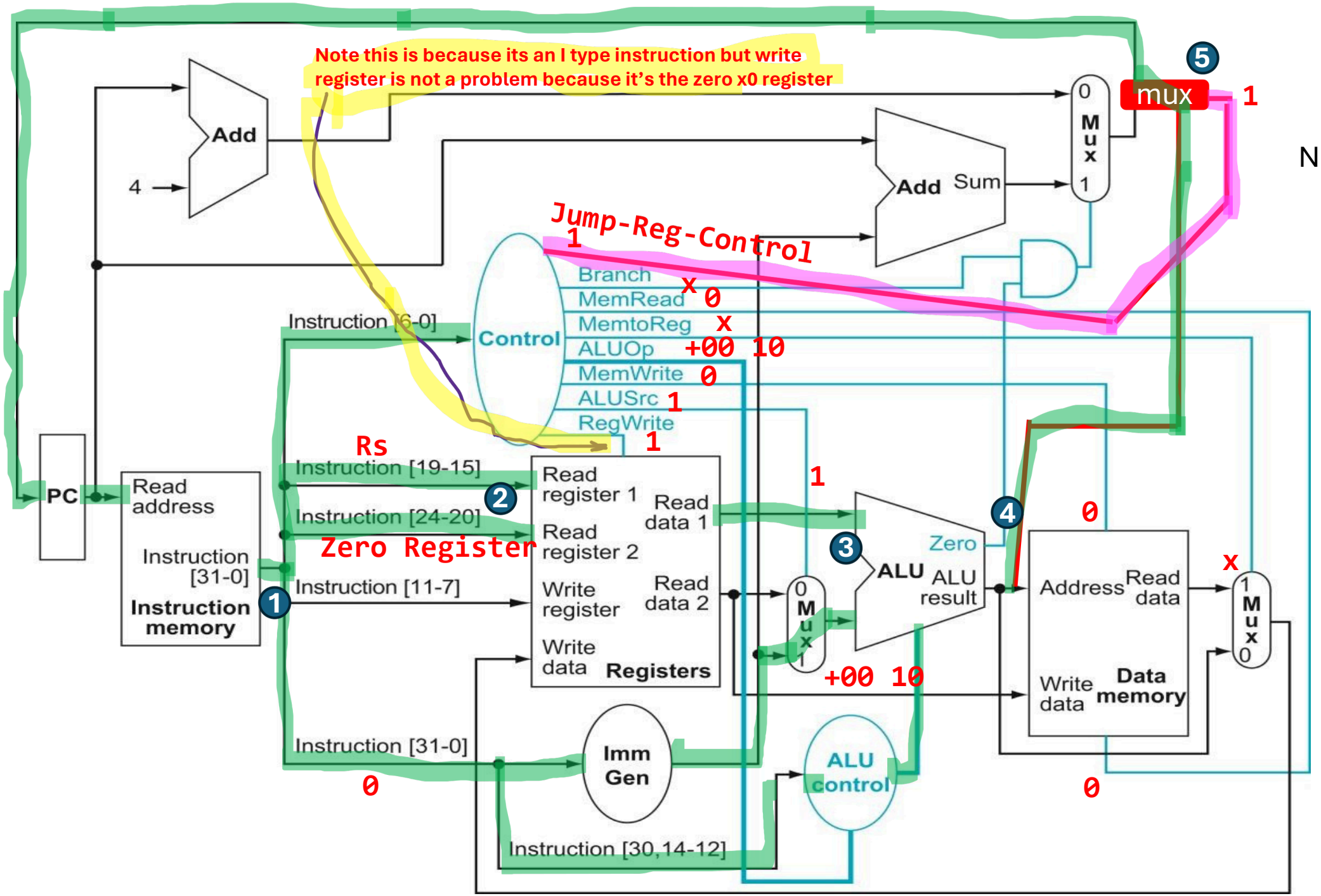
For the data path changes

For new control signals

For new components

For new wires

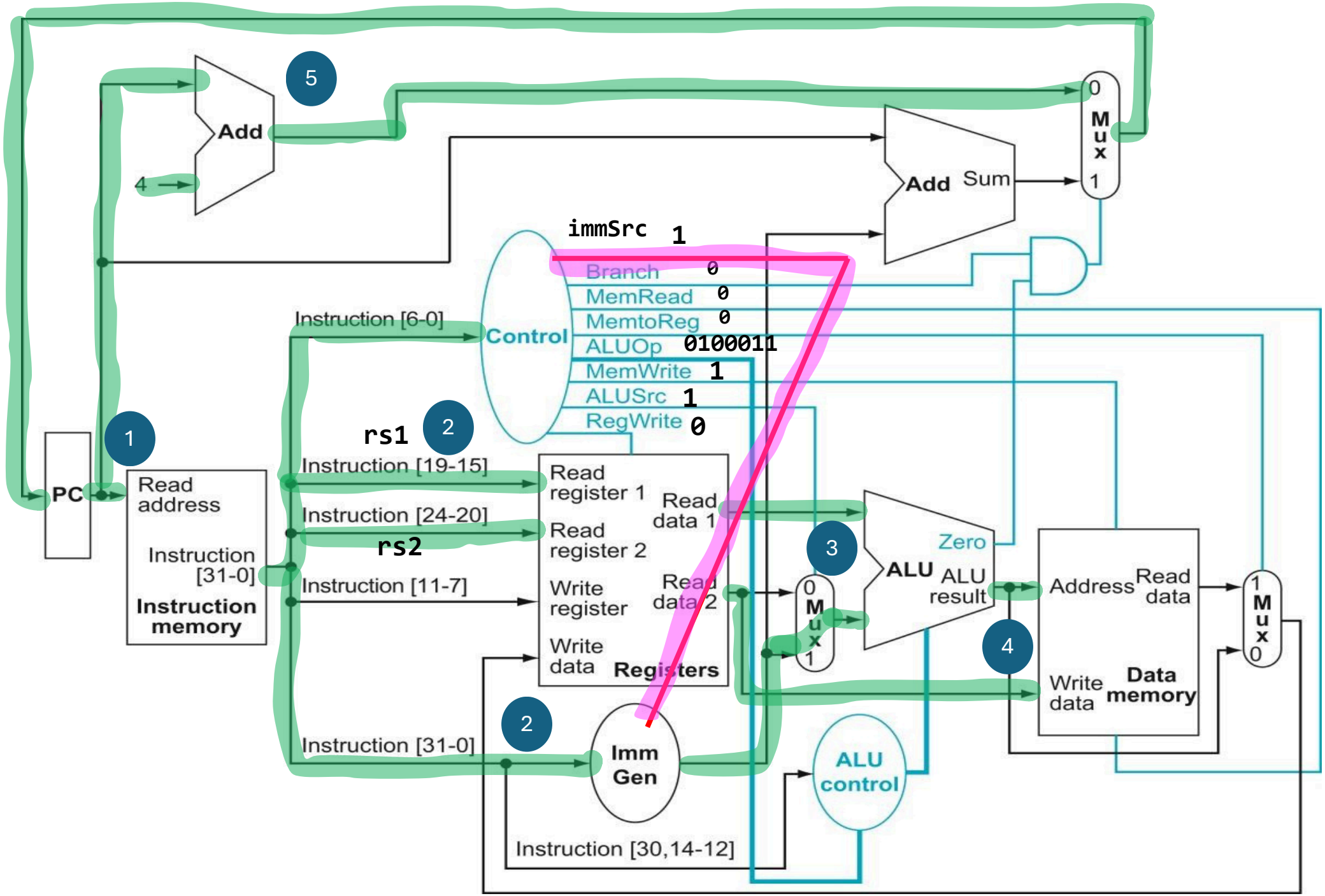
For annotations that support my description



Q1:
Jump Register
`jr`
Datapath
Note jr = jalr x0, Rs,0

1. PC points to jr instruction
2. Instruction is fetched and decoded – Rs1 and Rs2 placed on register file,Rs2 is zero register. Immediate value hard coded to 0 Control signals established
3. Immediate value added to Rs1
4. ALU result is new PC addresss
5. New wire routes PC address up to new mux that places PC value on output of upper right mux which will be the next PC address (on next clock cycle)

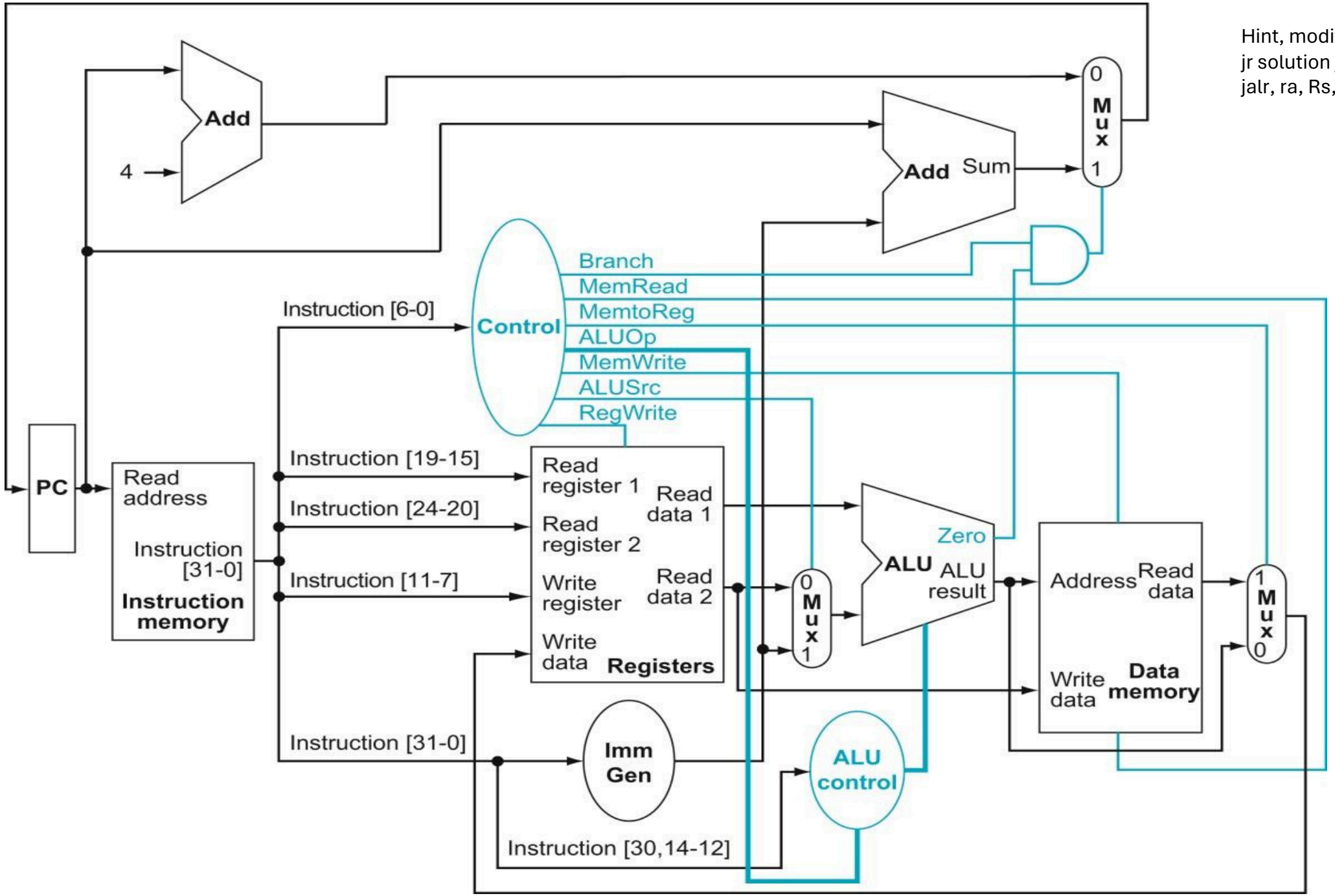
Q1: Store Word (sw) Datapath



1.PC points to sw instruction. Instruction is fetched and decoded.
2.rs1, and rs2 are placed on register file. Immediate is placed on Imm Gen – new control signal immSrc determines where immediate value is located (e.g., if immSrc is 1, Imm Gen will output bits [0:4] and [5:11] combined and sign extended; will find correct bits for immediate.
3. rs1 is added with the immediate.
4. Sum result is fed into the address input to memory. rs2 is fed into the write data input to memory.rs2 is now written to memory.
5. PC is incremented by 4 on next clock cycle.

Hint, modify
jr solution jal =
jalr, ra, Rs, imm

Q2:
Jump Link **`jal`**
Datapath



Q3:
Load Upper Immediate
lui Datapath