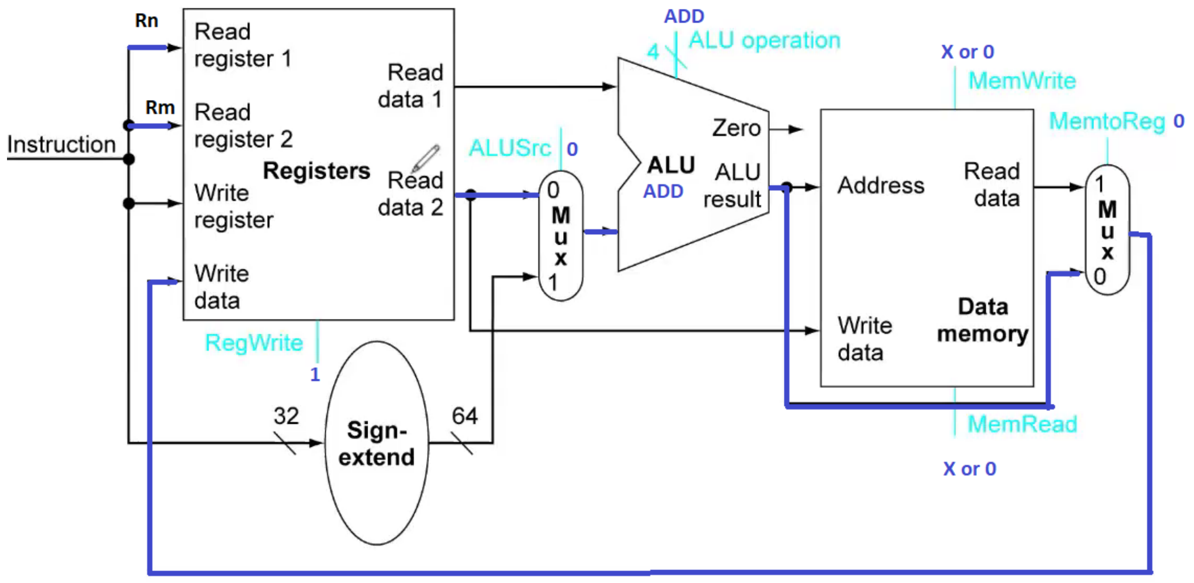
CSCI 4591: Computer Architecture
HW Assignment # 6
Due Date: April 4, 2021 @ 11:55 PM

1) Consider the following instruction:
Instruction: AND Rd, Rn, Rm
Interpretation: [Rd] ← [Rn] AND [Rm]

1. What are the values of control signals generated by the control in figure shown on slide 26 (The datapath for the memory instructions and the R-type instructions) for this instruction?
   Control signals are written in diagram

PCSrc = 0 or x
MemWrite = x
MemtoReg = 0
MemRead = x
ALUSrc = 0
RegWrite = 1 output of ALU and write it back
Reg2Loc = 0
ALUop = 10 / ADD

R-Type/Load/Store Datapath



2. Which resources (blocks) perform a useful function for this instruction?
   Read Registers, Read Data, ALU, Multiplexors
   All except branch ADD unit

Control Signal: RegWrite

3. Which resources (blocks) produce no output for this instruction? Which resources produce output that is not used?
   Sign extended converter and Data Memory Block
   ADD branch and write ports
   Control Blocks: PCSrc, MemRead, MemWrite, Reg2Loc

2) Consider the following instruction mix:

R-Type	**I-Type**	**LDUR**	**STUR**	**CBZ**	**B**
24%	28%	25%	10%	11%	2%

1. What fraction of all instructions use data memory?

   LDUR and STUR use data memory so 25% + 10% = 35%

2. What fraction of all instructions use instruction memory?
   All use instruction memory = 100%

3. What fraction of all instructions use the sign extend?
   I type, LDUR, STUR, CBZ and B use sign extend so
   28% + 25% + 10% + 11% = 74%

4. What is the sign extend doing during cycles in which its output is not needed?
   If sign extend is not needed, it just doesn't turn on.

3) When silicon chips are fabricated, defects in materials (e.g., silicon) and manufacturing errors can result in defective circuits. A very common defect is for one signal wire to get "broken" and always register a logical 0. This is often called a "stuck-at-0" fault.

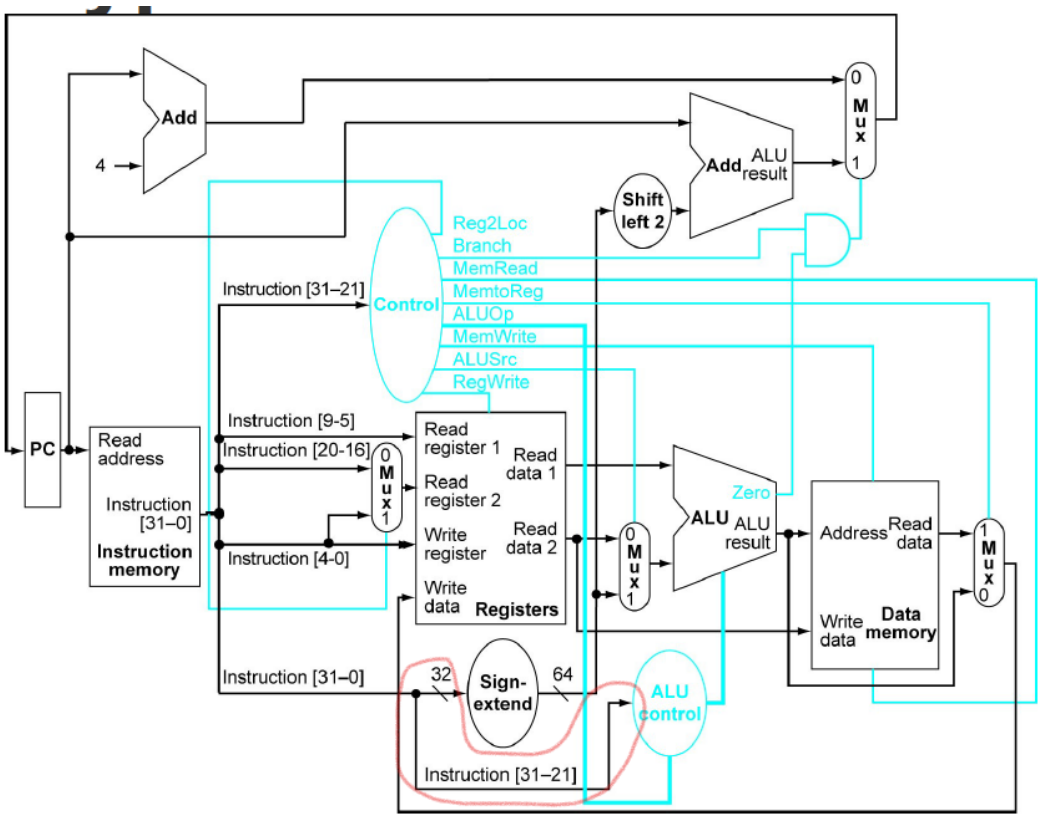
1. Which instructions fail to operate correctly if the MemtoReg wire is stuck at 0?
   If MemToReg is stuck at 0, then write data back to register will not operate
   R type and LDUR will not move forward

2. Which instructions fail to operate correctly if the ALUSrc wire is stuck at 0?
   Sign extend will not operate properly. Part of the instructions will not be read.
   I instruction type will not work and LDUR/STUR

3. Which instructions fail to operate correctly if the Reg2Loc wire is stuck at 0?
   slide 34
   STUR and CBZ will not work

4) In my lecture I did not discussed the I-type instructions like ADDI or ANDI

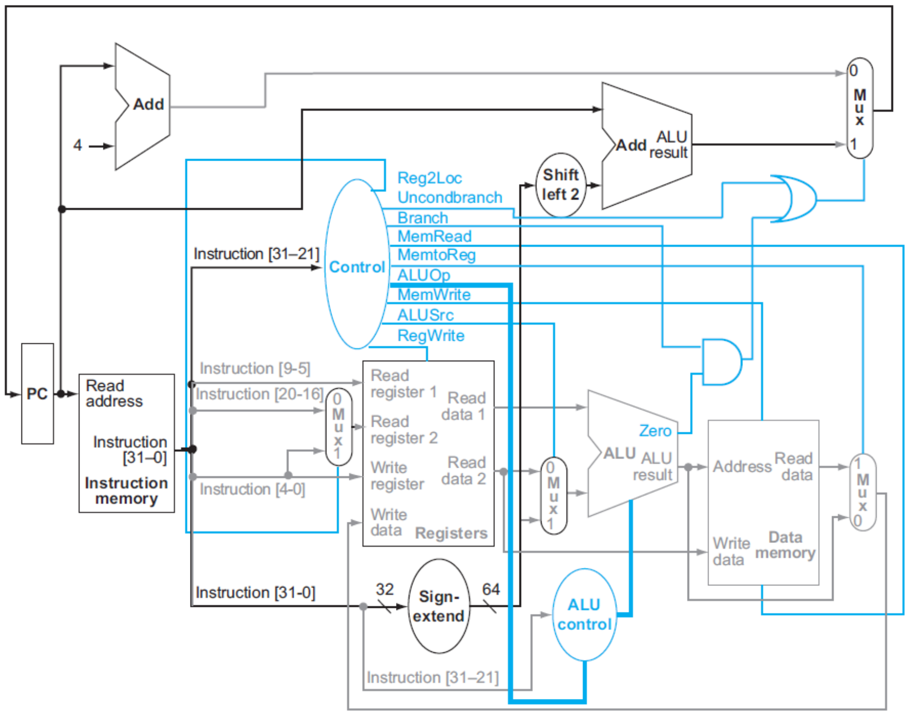
1. What additional logic blocks, if any, are needed to add I-type instructions to the figure shown on slide 36? Add any necessary logic blocks to the figure and explain their purpose.
   https://www.cs.fsu.edu/~zwang/files/cda3101/Fall2017/Lecture5_cda3101.pdf
   No need to add any extra logic blocks, but we do need to change the sign extended to immed. The immed field is zero-extended if it is a logical operation. Otherwise, it is sign extended. The path is extended to accommodate the different instructions for the offset value.



2. List the values of the signals generated by the control unit for ADDI. Explain the reasoning for any "don't care" control signals.
   Reg2Loc, the first multiplexor isnt needed for 2nd register, dont care
   MemRead also isn't needed since only LDUR is the only one that uses it, dont care
   Jump =0
   Branch = 1
   MemWrite = 0
   MemRead = 0
   RegWrite = 0
   AluOP = 00

5) In my class lecture I had mentioned that the figure for Datapath With Branch Added shown on slide 41 may not work for unconditional branch instruction. (Jump/J Format)

1. Explain why or why not?
   The AND Gate will not generate PC source signal. Branch is just a simple path, from instruction, to sign extend to shift left Add ALU result. PCSrc = 1, MemWrite = 0, RegWrite = 0, rest is don't care.
   Unconditional Branch looks like branch instruction with longer offset but is not conditional. 2 Bits of branch address = 00. Next bits are sign extended and then shift it. We Need an additional OR gate with the control signal to select branch target.
   **An
   additional OR-gate (at the upper right) is used to control the multiplexor that chooses between the branch target and the sequential instruction
   following this one. One input to the OR-gate is the Uncondbranch control signal. Although not shown, the Sign-extend logic would recognize
   the unconditional branch opcode and sign-extend the lower 26 bits of the branch instruction to form a 64-bit address to be added to the PC.**
   quoted from figure 4.23
2. How will you fix the datapath to fix the problem.
   There is no fix since it is in the hardware



6) We have only one sign extend unit but multiple input format (R, D, I B, CB). How will you design the sign extend unit to accommodate all the different possible formats?

   Extend the bits, look at opcode and feed all the bits into it

The sign extend shouldn't need to be redesigned. Multiple input formats can run, the opcode determines what happens to the input. The opcode differentiates between different instructions and from there decides what the extension should be.

R- Does Not go through sign extend, so no need to do anything extra
D- Bit 20-12(9) → sign extend to 64 bits

Opcode	Address	Op2	Rn	Rt
31-21	20-12	11-10	9-5	4-0
11 bits	9 bits	2 bits	5 bits	5 bits

I → extend to 64

Opcode	Rs	Rt	Address
31-26	11-10	9-5	25-16
6 bits	5 bits	5 bits	16 bits

Branch → extend to 64 bits

Opcode	Address
31-26	25-0
6 bits	26 bits

CB → extend to 64 bits

Opcode	Address	Rt
31- 24	23-5	4-0
8 bits	19 bits	5 bits