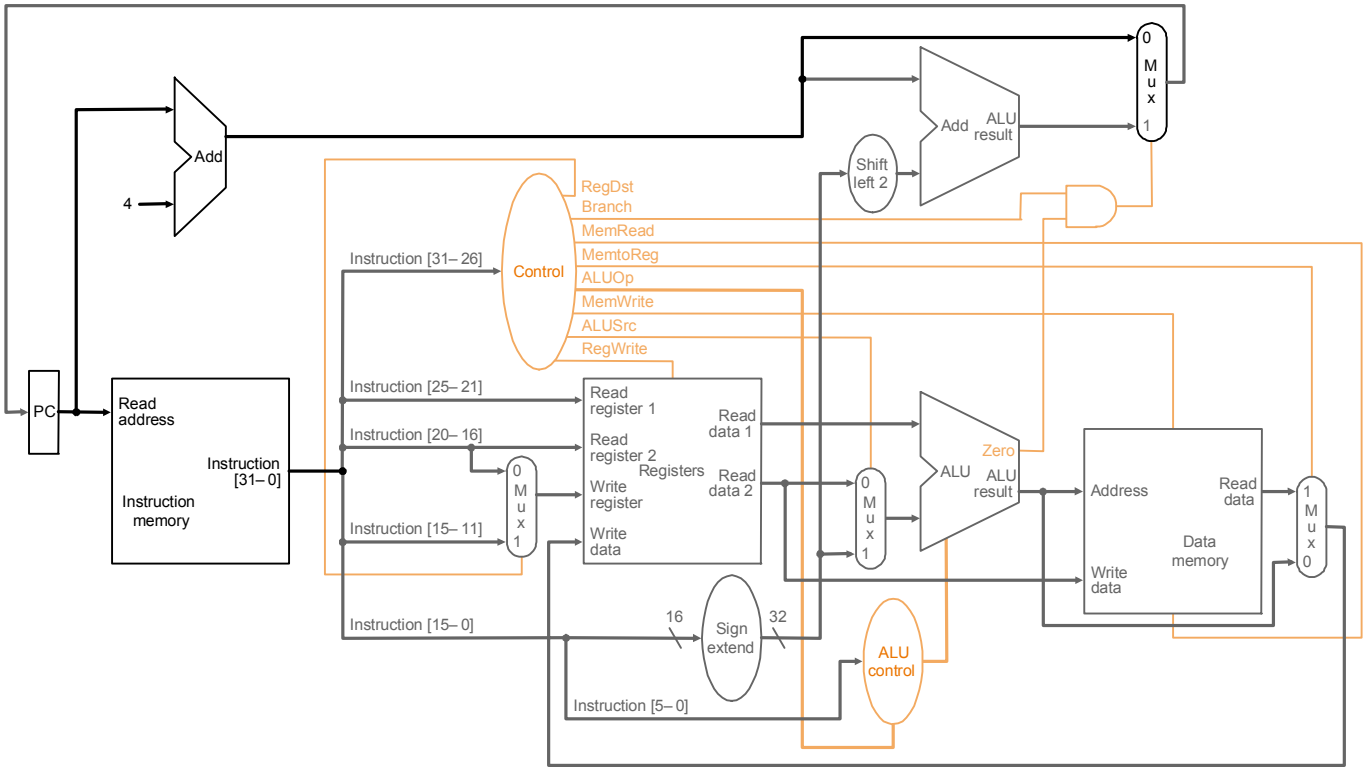
微算機系統第四次隨堂考 姓名_____ 學號_____

Answer below problems based on the instruction set, {`lw`, `sw`, `beq`, `add`, `sub`, `and`, `or`, `slt`}.

1. (40%) Describe the effect that a single stuck-at-0 fault would have on the multiplexor in the single-cycle datapath. Which instruction, if any, would still work? Consider each of the following faults separately: RegDst=0, ALUSrc=0, MemtoReg=0, Zero=0.
2. (40%) Repeat the problem 1, but this time consider stuck-at-1 fault.
3. (20%) Repeat the problem 1, but this time consider the multiple-cycle datapath: RegDst=0, ALUSrcA=0, MemtoReg=0, IorD=0. (Please refer to the finite state machine on the back.)



Answers: 1.

(a) RegDst = 0: Write register number can only come from Instruction[20-16]; hence, all R-format instructions, including `add`, `sub`, `and`, `or`, `slt`, will not work because these instructions will not be able to write their results to the register file. However, all I-format instructions, such as `lw`, `sw` and `beq`, can still work properly.

(b) ALUSrc = 0: The second ALU source cannot come from sign-extension output; hence, all I-format instructions will not work correctly, but all R-format instructions function well.

(c) MemtoReg = 0: Write data in register file can only come from ALU result; hence, all instructions except `lw` works properly.

(d) Zero = 0: All instructions except `beq` functions well.

2.

(a) RegDst = 1: Write register number can only come from Instruction[15-11]; hence, all instructions except `lw` work properly.

(b) ALUSrc = 1: The second ALU source cannot come from the register files; hence, all R-format instructions will not work correctly, but all I-format instructions function well.

(c) MemtoReg = 1: Write data in the register file can only come from the Memory; hence, all R-format instructions do not work. However, `lw`, `sw` and `beq` work properly.

(d) Zero = 1: Even `beq` cannot work properly because condition test in `beq` may sometimes fail.

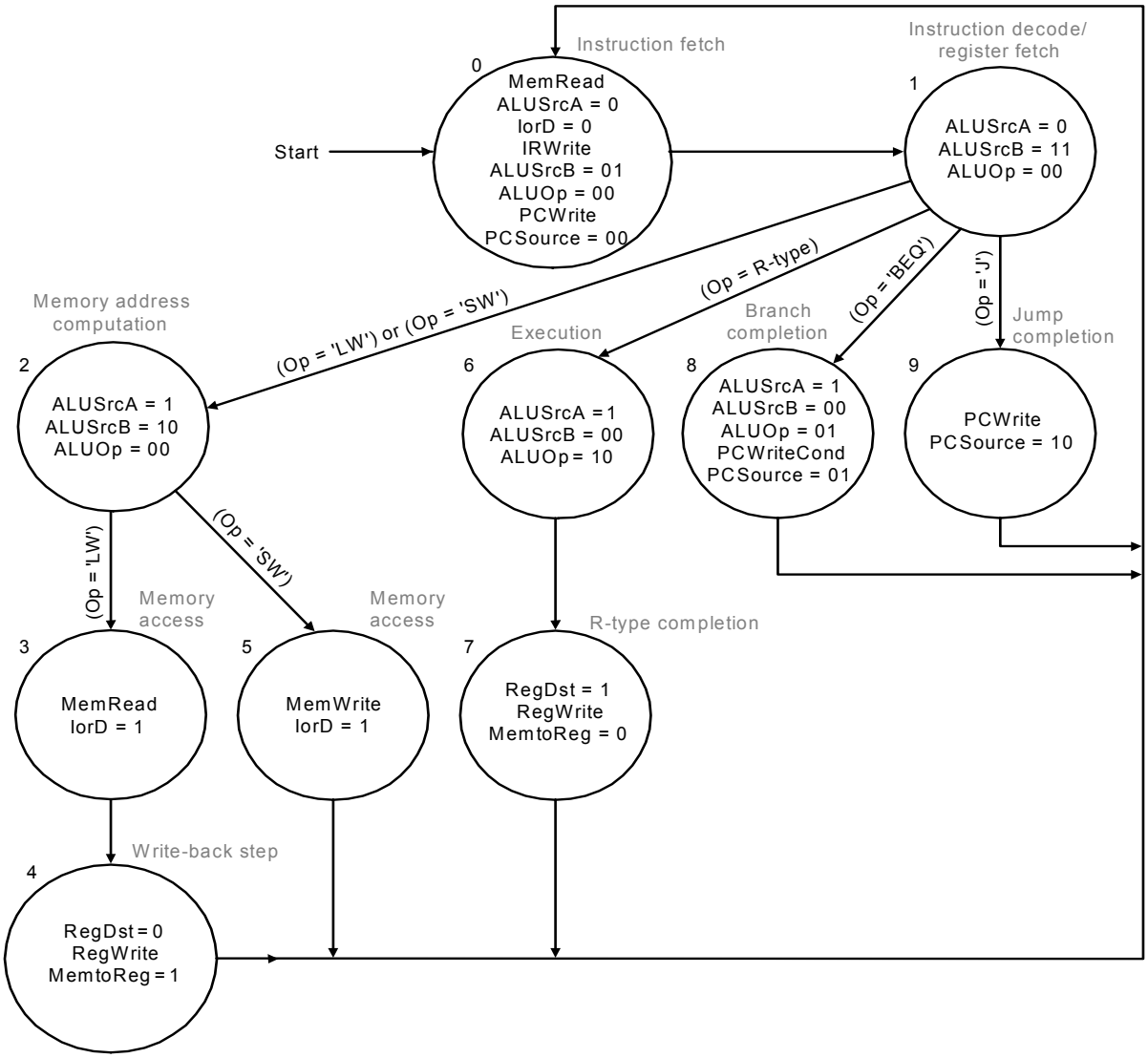
3.

(a) RegDst = 0: Any instruction that goes through state 7 (which requires RegDst = 1) fail to work.

(b) ALUSrcA = 0: Any instruction that goes through states 2, 6 and 8 (which require ALUSrcA = 1) fail to work. Therefore, only `beq` functions well.

(c) MemtoReg = 0: Any instruction that goes through state 4 (which requires MemtoReg = 1) fail to work.

(d) IorD = 0: Any instruction that goes through states 3 and 5 (which requires IorD = 1) fail to work.


Start
0
Instruction fetch
MemRead
ALUSrcA = 0
IorD = 0
IRWrite
ALUSrcB = 01
ALUOp = 00
PCWrite
PCSource = 00
1
Instruction decode/
register fetch
ALUSrcA = 0
ALUSrcB = 11
ALUOp = 00
(Op = 'LW') or (Op = 'SW')
(Op = R-type)
(Op = 'BEQ')
(Op = 'J')
2
Memory address
computation
ALUSrcA = 1
ALUSrcB = 10
ALUOp = 00
6
Execution
ALUSrcA =1
ALUSrcB = 00
ALUOp= 10
8
Branch
completion
ALUSrcA = 1
ALUSrcB = 00
ALUOp = 01
PCWriteCond
PCSource = 01
9
Jump
completion
PCWrite
PCSource = 10
(Op = 'LW')
(Op = 'SW')
3
Memory
access
MemRead
IorD = 1
5
Memory
access
MemWrite
IorD = 1
7
R-type completion
RegDst = 1
RegWrite
MemtoReg = 0
4
Write-back step
RegDst = 0
RegWrite
MemtoReg = 1