Computer organization

Chapter 5

學號：　　姓名：

1. (18%) Describe the effect that a single stuck-at-0 fault (i.e., regardless of what it should be, the signal is always 0) would have for the signals shown below, in the multiple-cycle datapath in Figure 2. Which instructions, if any, will not work correctly? Explain why.
   A. RegWrite = 0
   B. MemRead = 0
   C. MemWrite = 0
   D. IRWrite = 0
   E. PCWrite = 0
   F. PCWriteCond = 0

   <sol>
   a.**RegWrite** = 0: All R-format instructions, in addition to 1 w, will not work because these instructions will not be able to write their results to the register file.
   b.**MemRead** = 0: None of the instructions will run correctly because instructions will not be fetched from memory.
   c. **MemWrite** = 0: s w will not work correctly because it will not be able to write to the data memory.
   d. **IRWrite** = 0: None of the instructions will run correctly because instructions fetched from memory are not properly stored in the IR register.
   e.**PCWrite** = 0: Jump instructions will not work correctly because their target address will not be stored in the PC.
   f.**PCWriteCond** = 0: Taken branches will not execute correctly because their target address will not be written into the PC.

2. (10%) Please add any necessary datapaths and control signals to include the instruction **beq (branch on equal)** to the single-cycle datapath shown in Figure 1.

3. (18%) According to the single-cycle datapath in Figure 1, please fill in the blanks in the following table of control signals

Instruction	RegDst	ALUSrc	MemtoReg	RegWrite	MemRead	MemWrite	Branch	ALUOp1	ALUOp0
R-format	1	0	0	1	0	0	0	1	0
lw	0	1	1	1	1	0	0	0	0

4. (18%) Consider the following machines, and compare their performance using the data below.

Instruction	**Frequency**
Loads	30%
Stores	20%
R-type	40%
Branch/Jump	10%

M1: The multicycle datapath of Chapter 5 with a **3.8 GHz** clock.

M2: A machine like the multicycle datapath of Chapter 5, except that register updates are done in the same clock cycle as a memory read or ALU operation. Thus in Figure 3, states 6 and 7 and states 3 and 4 are combined. This machine has a **3.5 GHz** clock, since the register update increase the length of the critical path.

M3: A machine like M2 except that effective address calculations are done in the same clock cycle as a memory access. Thus states 2, 3, and 4 can be combined, as can 2 and 5, as well as 6 and 7. This machine has a **2.8 GHz** clock because of the long cycle created by combining address calculation and memory access.

1. Please calculate the average CPI of each machine.

Find out which of the machine is fastest.

<sol>

1.

M1	
Instruction	**cycle**
Loads	5
Stores	4
R-type	4
Branch/Jump	3

M2	
Instruction	**cycle**
Loads	4
Stores	4
R-type	3
Branch/Jump	3

M3	
Instruction	**cycle**
Loads	3
Stores	3
R-type	3
Branch/Jump	3

M1: Average CPI = 4.2 the average time of excuting a instruction: 4.2/3.8=1.105

M2: Average CPI = 3.5 the average time of excuting a instruction:3.5/3.5=1

M3: Average CPI = 3 the average time of excuting a instruction: 3/2.8=1.071

2. The M2 machine is fastest!!

5. (22%) Note that some control signals are absent in the finite state machine of Figure 3. Please finish Figure 3 according to the multicycle datapath of Figure 2. In other word, please fill in the contents of state2, state4, state 9 in Figure 3.

6. (20%) We wish to design a method to detect exceptions, including *undefined instructions* and *arithmetic overflow*, and to transfer control to the appropriate state in the exception states. **Add any necessary datapaths and control signals to the multicycle datapath of Figure 2, and any necessary states and control signals to the finite state machine of Figure 3**. Not that *the exception address = 8000 0180*.

Figure 1

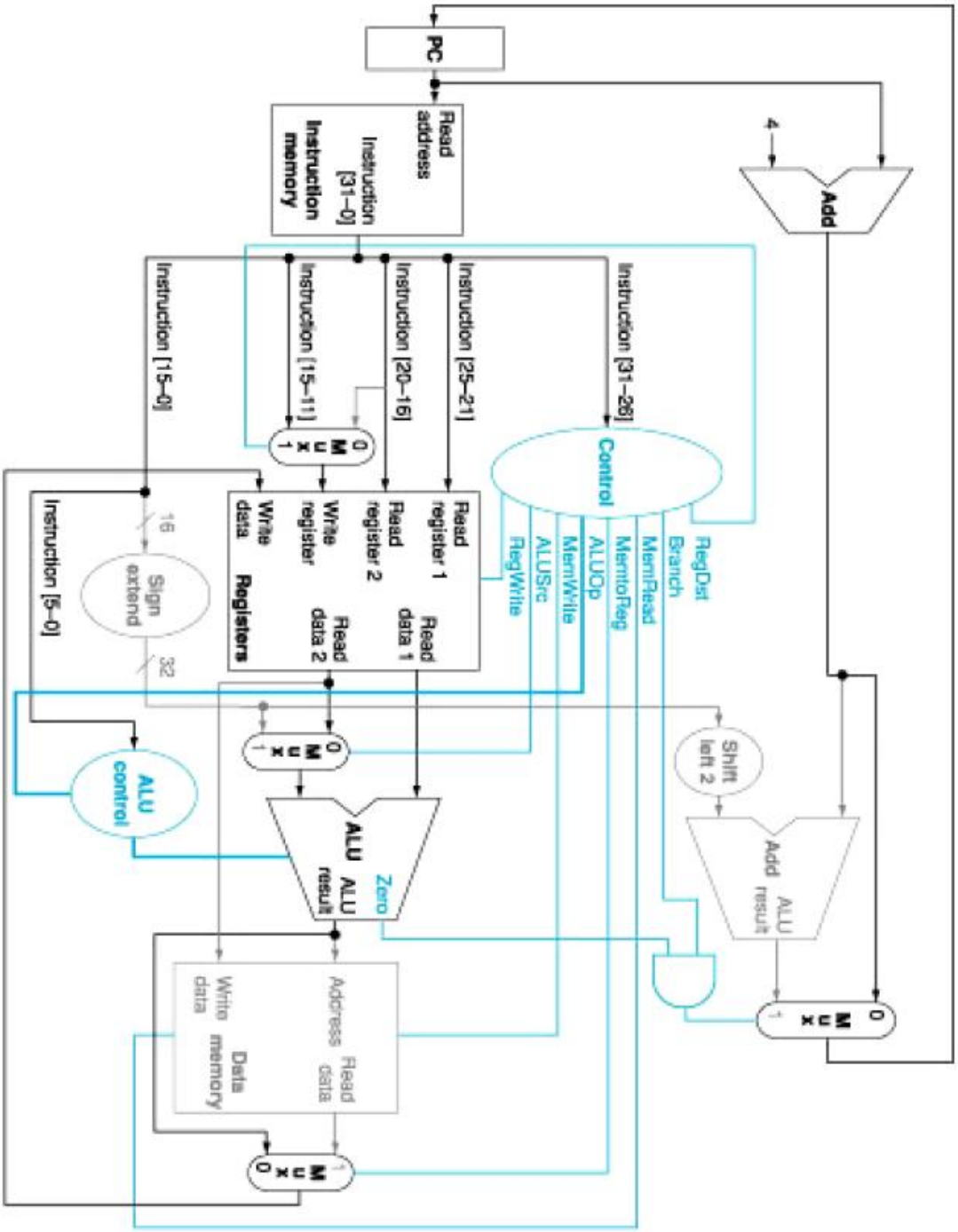

PC
Read address
Instruction [31–0]
Instruction memory
4
Add
Instruction [31–26]
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
Instruction [15–0]
Instruction [5–0]
Control
RegDst
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
16
Sign extend
32
ALU control
ALU
ALU result
Zero
Shift left 2
Add
ALU result
Address
Read data
Write data
Data memory
M u x
0
1

Figure 2

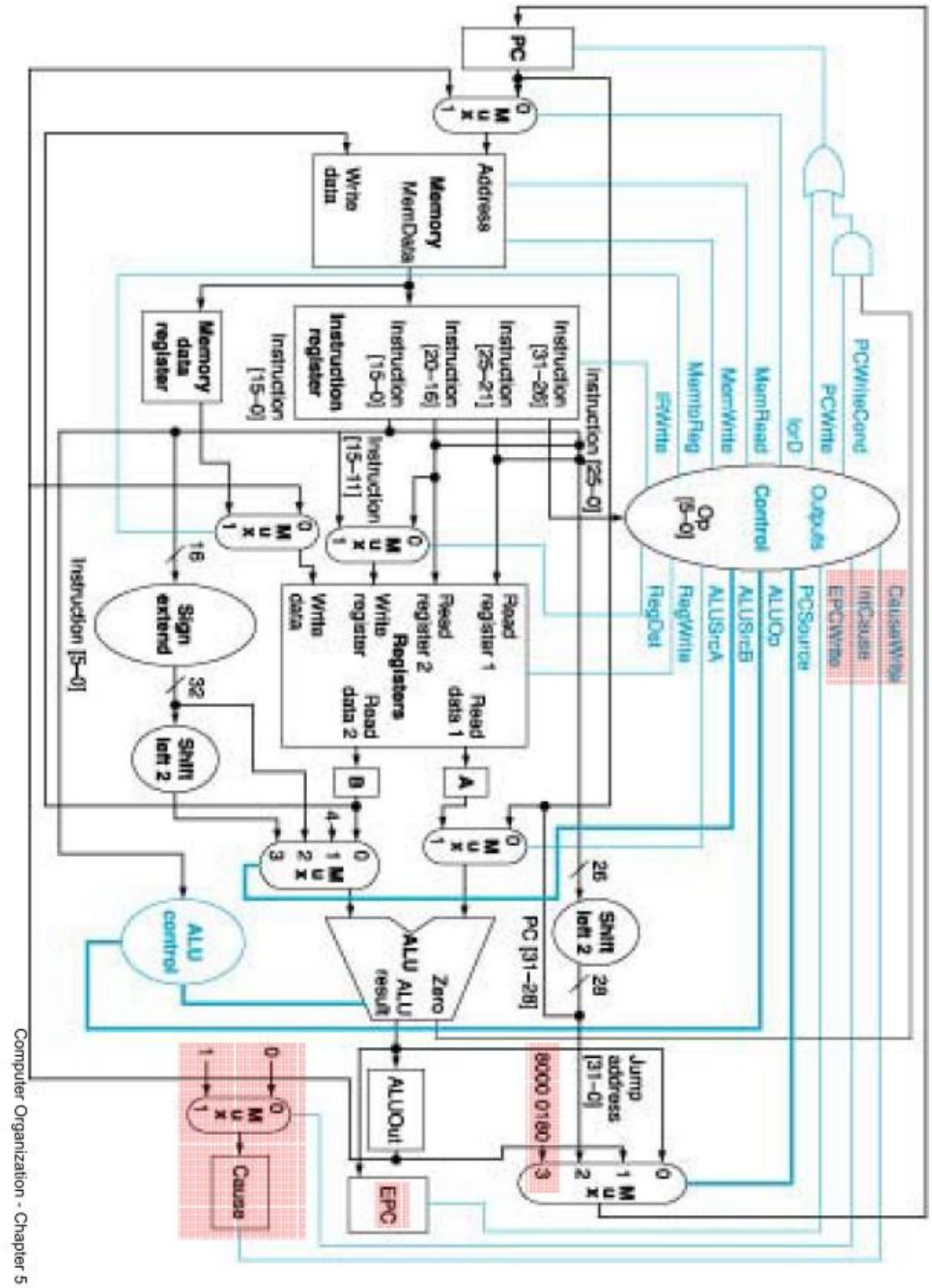
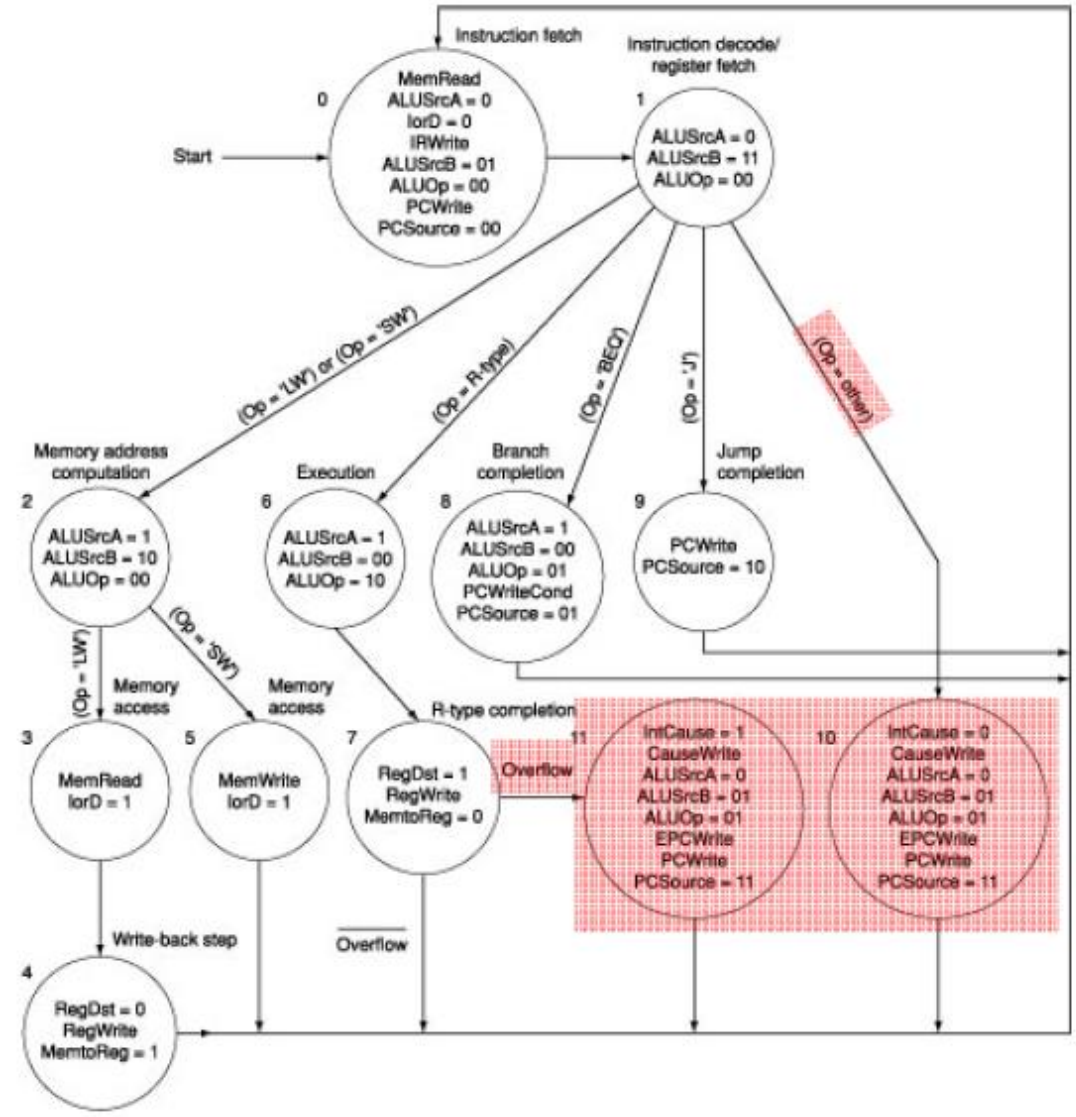

PC
Address
Memory
MemData
Write data
Memory data register
Instruction register
Instruction [31–26]
Instruction [25–21]
Instruction [20–16]
Instruction [15–0]
Instruction [25–0]
Instruction [15–11]
Instruction [5–0]
Sign extend
16
32
Shift left 2
Registers
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
A
B
ALU
ALU result
Zero
ALU control
PC [31–28]
Shift left 2
26
28
Jump address [31–0]
0000 0100
ALUOut
EPC
Cause
Op [5–0]
Outputs
Control
PCWriteCond
PCWrite
IorD
MemRead
MemWrite
MemtoReg
IRWrite
PCSource
ALUOp
ALUSrcB
ALUSrcA
RegWrite
RegDst
CauseWrite
IntCause
EPCWrite
M u x

Figure 3


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