计算机组成第四章作业

4.1

考虑如下指令：

指令：and rd, rs1, rs2

解释：Reg[rd] = Reg[rs1] AND Reg[rs2]

4.1.1

RegWrite	ALUSrc	ALUoperation	MemWrite	MemRead	MemToReg
true	0	"and"	false	false	0

4.1.2

Registers, ALUsrc mux, ALU, and the MemToReg mux.

4.1.3

All blocks produce some output. Th e outputs of DataMemory and Imm Gen are not used.

4.3

根据下述指令组合回答问题。

R-type	I-type(non-ld)	Load	Store	Branch	Jump
24%	28%	25%	10%	11%	2%

4.3.1

25+10 = **35%**, only **Load** and **Store** use Data Memory.

4.3.2

100% Every instruction must be fetched from instruction memory before it can be executed.

4.3.3

100 - 24 = **76%**. Only R-type instructions do not use the Sign extender.

4.3.4

Th e sign extend produces an output during every cycle. If its output is not needed, it is **simply ignored**.

4.4

在制造硅芯片时，材料（例如硅）缺陷和制造错误会导致电路故障。一个非常常见的故障是，信号线发生“断路”，导致总是保持逻辑“0”。这被称为“固定为0” 故障。

4.4.1

Load is broken.

4.4.2

I-type, loads , stores are all broken.

4.5

本题中，我们将仔细讨论单周期数据通路中之心指令的细节。假设本周起处理器取来指令：0x00c6_fa23.

4.5.1

原机器码翻译成汇编指令为：

sd x12, 20(x13)

ALUop	ALU Control Lines
00	0010

4.5.2

Th e new PC is the old **PC + 4**. Th is signal goes from the PC, through the “PC + 4” adder, through the “branch” mux, and back to the PC.

4.5.3

ALUsrc :

Inputs: Reg[x12] and 0x0000000000000014 ;

Output: 0x0000000000000014

MemToReg :

Inputs: Reg[x13] + 0x14 and ；

output:

Branch:

Inputs: PC+4 and 0x000000000000000A

4.5.4

ALU inputs: Reg[x13] and 0x0000000000000014
PC + 4 adder inputs: PC and 4
Branch adder inputs: PC and 0x0000000000000028

4.5.5

Inst_field[19:15] = 5'b01101;

Inst_field[24:20] = 5'b01100;

Inst_field[11:7] = 5'b10100.

4.6

4.4节中没有讨论型指令，如addi或者andi.

4.6.1

No additional logic blocks are needed.

4.6.2

Branch: false
MemRead: false (See footnote from solution to problem 4.1.1.)
MemToReg: 0
ALUop: 10 (or simply saying “add” is sufficient for this problem)
MemWrite: false
ALUsrc: 1
RegWrite: 1

4.7

假设用来实现处理器数据通路的各功能模块延迟如下所示：

I-Mem/D-Mem	Register File	Mux	ALU	Adder	Single gate	Register Read	Register Setup	Sign extend	Control
250ps	150ps	25ps	200ps	150ps	5ps	30ps	20ps	50ps	50ps

4.7.1

30+250+150+25+200+25+20=700ps

4.7.2

30+250+150+25+200+250+25+20=950ps

4.7.3

30+250+150+200+25+250=905ps

4.7.4

30+250+150+25+200+5+25+20=705ps

4.7.5

30+250+150+25+200+25+20=700ps

4.7.6

950ps

4.11

4.11.1

No new function blocks are needed.

4.11.2

Only the control unit needs modification.

4.11.3

No new data paths are needed.

4.11.4

No new signals are needed.

4.12

4.12.1

No new functional blocks are needed.

4.12.2

The register file needs to be modified so that it can write to two registers in the same cycle. The ALU would also need to be modified to allow read data 1 or 2 to be passed through to write data 1.

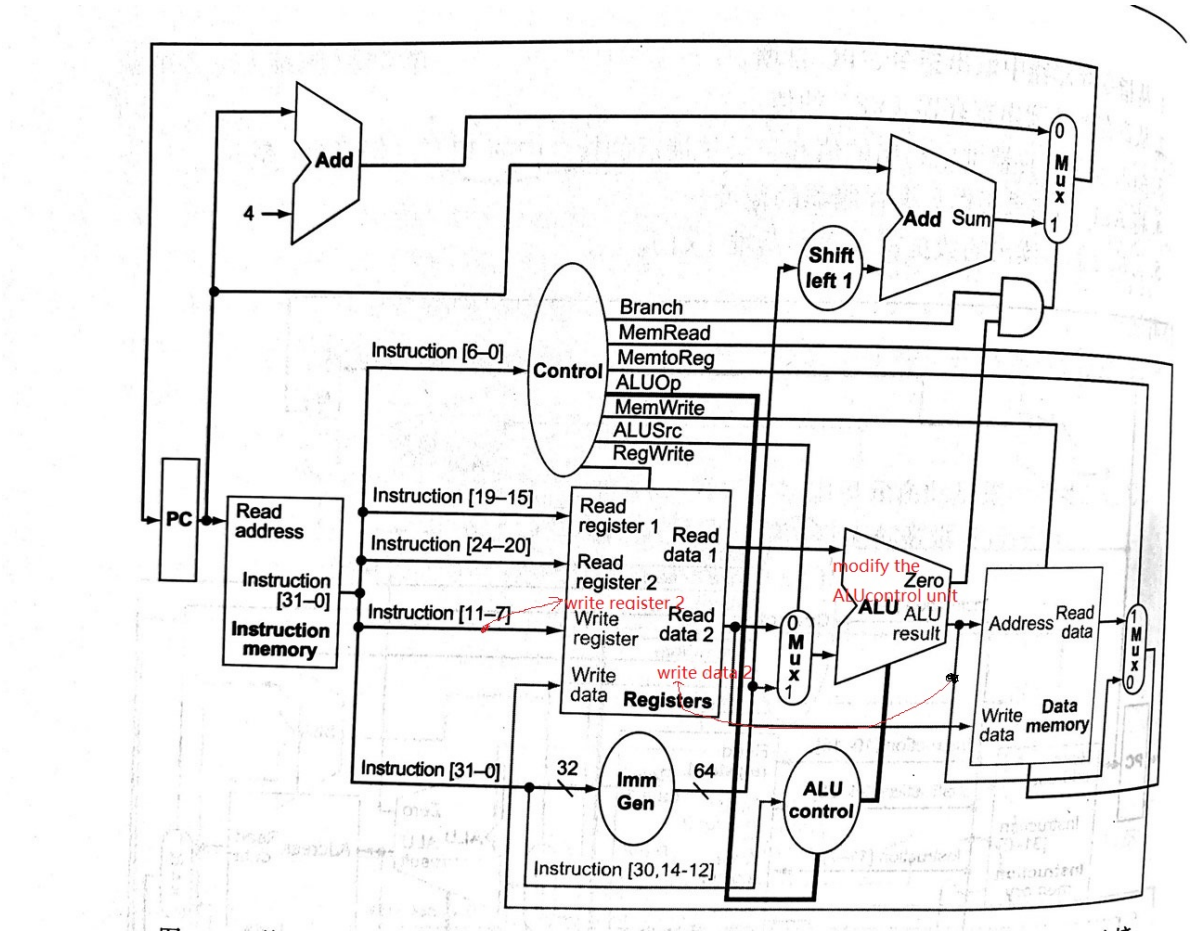
4.12.3

The answer depends on the answer given in 4.12.2: whichever input was not allowed to pass through the ALU above must now have a data path to write data 2.

4.12.4

There would need to be a second RegWrite control wire.

4.12.5



4.13

4.13.1

We need some additional muxes to drive the data paths discussed in 4.13.3.

4.13.2

No functional blocks need to be modified.

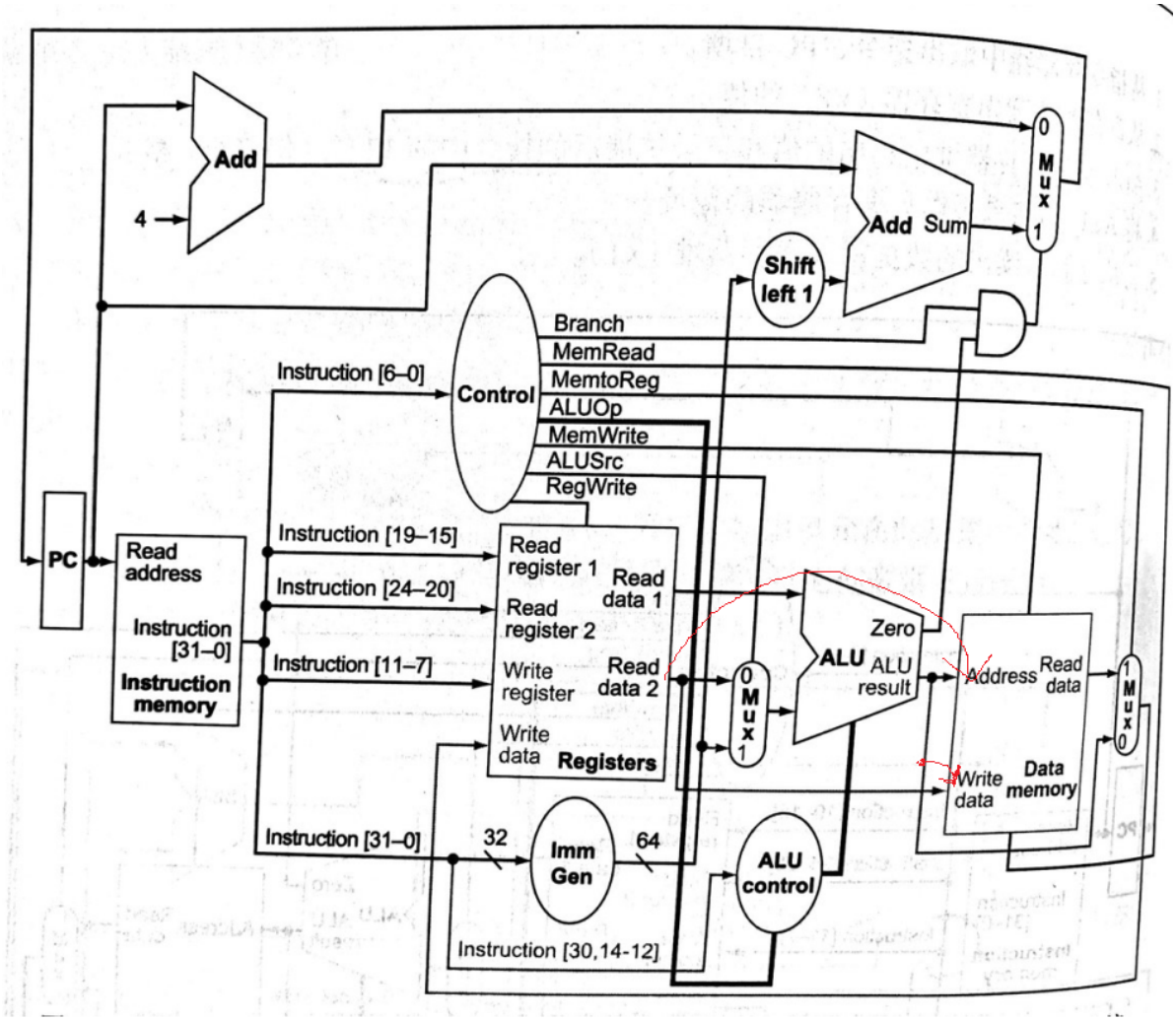
4.13.3

There needs to be a path from the ALU output to data memory's write data port. There also needs to be a path from read data 2 directly to Data memory's Address input.

4.13.4

These new data paths will need to be driven by muxes. These muxes will require control wires for the selector.

4.13.5


Add
4
Add Sum
Shift left 1
0 M u x 1
Branch
MemRead
MemtoReg
ALUOp
MemWrite
ALUSrc
RegWrite
Control
Instruction [6–0]
PC
Read address
Instruction [31–0]
Instruction memory
Instruction [19–15]
Instruction [24–20]
Instruction [11–7]
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
ALU
Zero
ALU result
Address
Read data
Write data
Data memory
1 M u x 0
Instruction [31–0]
32
Imm Gen
64
ALU control
Instruction [30,14-12]