EECS 112 & CSE 132, FALL 2017

Homework 3

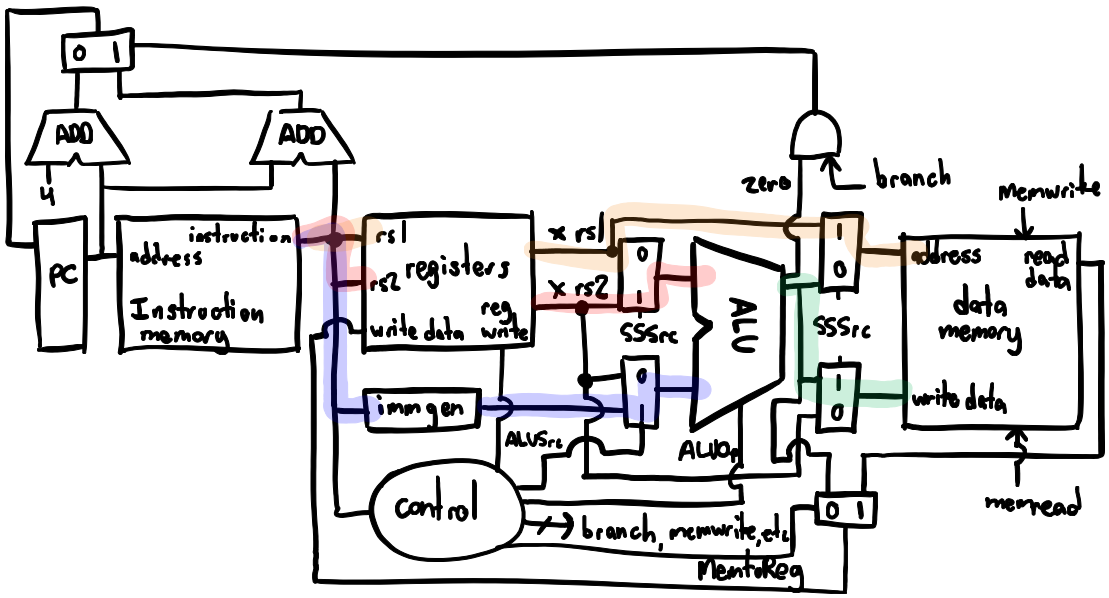
Due date: November, 22, 2017

Student ID:									Name:

1) Draw the required logic including register file, ALU and memory for the following instruction in a single cycle datapath.

`Ss rs1, rs2, imm` Mem[Reg[rs1]] = Reg[rs2] + immediate

SSSrc is 1 when SS instruction used



2) The individual stages of datapath have the following latencies:

IF	ID	EXE	MEM	WB
250 ps	350 ps	150 ps	300 ps	200 ps

Also, assume that instructions executed by the processor are broken down as follows:

ALU	Jump/Branch	Load	Store
45%	20%	20%	15%

a) What is the clock cycle time in a pipelined and non-pipelined datapath?

Pipelined:

$P_p = \max(250, 350, 150, 300, 200) = 350$ ps

Nonpipelined

$P_n = 250 + 350 + 150 + 300 + 200 = 1250$ ps

b) What is the total latency of an ld instruction in a pipelined and non-pipelined processor?

$t_p = 5 \text{ cycles} \cdot 350 \text{ ps} = 1750 \text{ ps}$

$t_{np} = 1250 \text{ ps}$

c) If we can split one stage of the pipelined datapath into two new stages, each with half the latency of the original stage, which stage would you split and what is the new clock cycle time of the processor?

Instruction Decode

$P_p = 300_{ps}$ ← MEM (next max)

d) Assuming there are no stalls or hazards, what is the utilization of the data memory? Utilization is the fraction of clock cycles in which data memory is used.

Ld+Sd = 35%

e) Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

ALU+Ld = 65%

3) Assume that X_{11} is initialized to 11 and X_{12} is initialized to 22. Suppose that the following code is executed on a pipeline datapath that does not handle data hazards (The programmer has to insert NOPs in the code where necessary). What would the final values of registers X_{13} and X_{14} be?

```
1 addi X11, X12, 5
2 add  X13, X11, X12
3 NOP
4 addi X14, X11, 15
```

$X_{11} = 22 + 5 = 27$ @ 3

$X_{13} = 11 + 22 = 33$ @ 4

NOP

$X_{14} = 27 + 15 = 42$

$X_{13} = 33$ $X_{14} = 42$

4) Consider the following sequence of instructions that are executed on a five-stage pipelined datapath.

```
add X15, X12, X11
ld  X13, 4(X15)
ld  X12, 0(X2)
or  X13, X15, X13
sd  X13, 0(X15)
```

a) If there is no forwarding or hazard detection, insert the minimum number of NOPs to ensure correct execution.

```
add x15 x12 x11
nop
nop
ld  x13 4(x15)
ld  x12 0(x2)
nop
or  x13 x15 x13
nop
nop
sd  x13 0(x15)
```

b) Now, change and/or rearrange the code to minimize the number of NOPs needed. You can assume register X_{17} can be used to hold temporary values in your modified code.

5) Suppose that we are having the following sequence of instructions:

```
ld  X1, 40(X6)
add X6, X2, X2
sd  X6, 50(X1)
```

a) Indicate dependencies and their type.

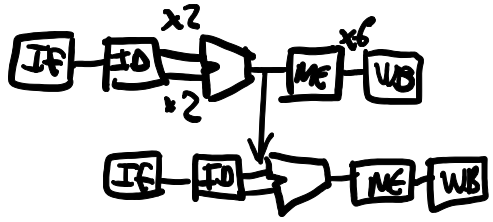
x1 data dependency (flow, RAW)
x6 data dependency (flow, RAW)

b) Assume there is no forwarding in this pipelined processor. Insert the minimum number of NOPs to ensure correct execution.

```
ld x1 40(x6)
add x6 x2 x2
nop
nop
sd x6 50(x1)
```

c) Add minimum number of NOPs to this code to eliminate hazards if there is ALU-ALU forwarding only (no forwarding from the MEM to the EX stage)?

```
ld x1 40(x6)
add x6 x2 x2
sd x6 50(x1)
```



6) The breakdown of instruction categories are as follows:

R-Type	BEQZ/BNEZ	JAL/JALR	LD	SD
40%	25%	5%	25%	5%

Also, assume the following branch predictors accuracies:

Always-Taken	Always-Not-Taken	2-Bit
45%	55%	85%

a) What is the extra CPI due to mispredicted branches with the always-taken predictor? (Assume that branch outcomes are determined in the EXE stage.)

$$\text{Extra CPI} = \frac{\text{mispredict}}{\text{instruction}} \cdot \frac{\text{penalty}}{\text{misprediction}}$$

$25\% \cdot 55\% \cdot 2 = 27.5\%$

b) What is the extra CPI for the always-not-taken predictor?

$25\% \cdot 45\% \cdot 2 = 22.5\%$

c) What is the extra CPI for the 2-bit predictor?

$25\% \cdot 15\% \cdot 2 = 7.5\%$

7) Suppose the repeating pattern (in a loop) of branch outcomes: NT, NT, NT, T, NT

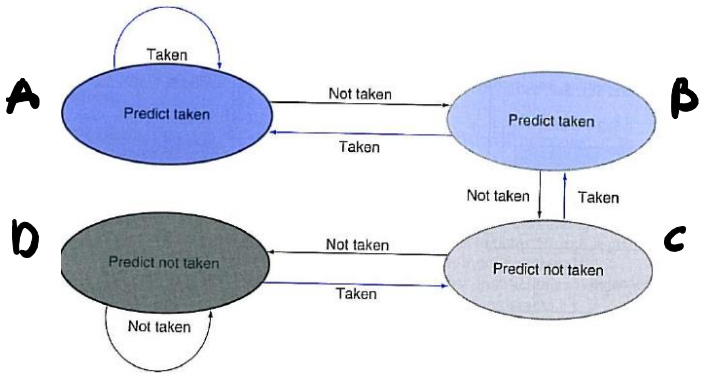
a) What is the accuracy of always-taken and always-not-taken predictors for this sequence of branch outcomes?

AT: 20%

ANT: 80%

b) What is the accuracy of the 2-bit predictor for this pattern? First, write the output of prediction then calculate accuracy.

Assume that the predictor starts off in the bottom left of the following figure (predict-not-taken).



NT	NT	NT	T	NT	?
D	D	D	D	C	D
✓	✓	✓	✗	✓	?

2-BIT: 80%