Computer Organization and Structure

Homework #4
Due: 2015/12/29

1. Assuming that logic blocks needed to implement a processor's datapath have the following latencies:

	I-Mem	**Add**	**Mux**	**ALU**	**Regs**	**D-Mem**	**Sign-extend**	**Shift-left-2**
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	2ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps

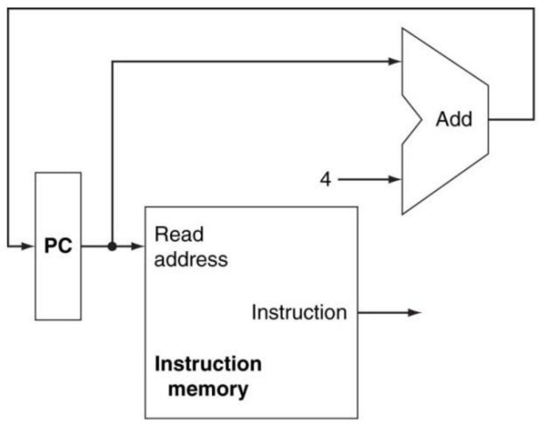

Figure 1: A portion of the datapath used for fetching instructions and incrementing the program counter. The fetched instruction is used by other parts of the datapath.

a. If the only thing we need to do in a processor is fetch consecutive instructions (Figure 1), what would the cycle time be?
b. Consider a datapath similar to the one in Figure 2, but for a processor that only has one type of instruction: unconditional PC-relative branch. What would the cycle time be for this datapath?
c. Repeat the above problem, but this time we need to support only *conditional* PC-relative branches.

The remaining three problems refer to the following logic block (resource) in the datapath:

	Resource
a.	Add 4 (to the PC)
b.	Data Memory

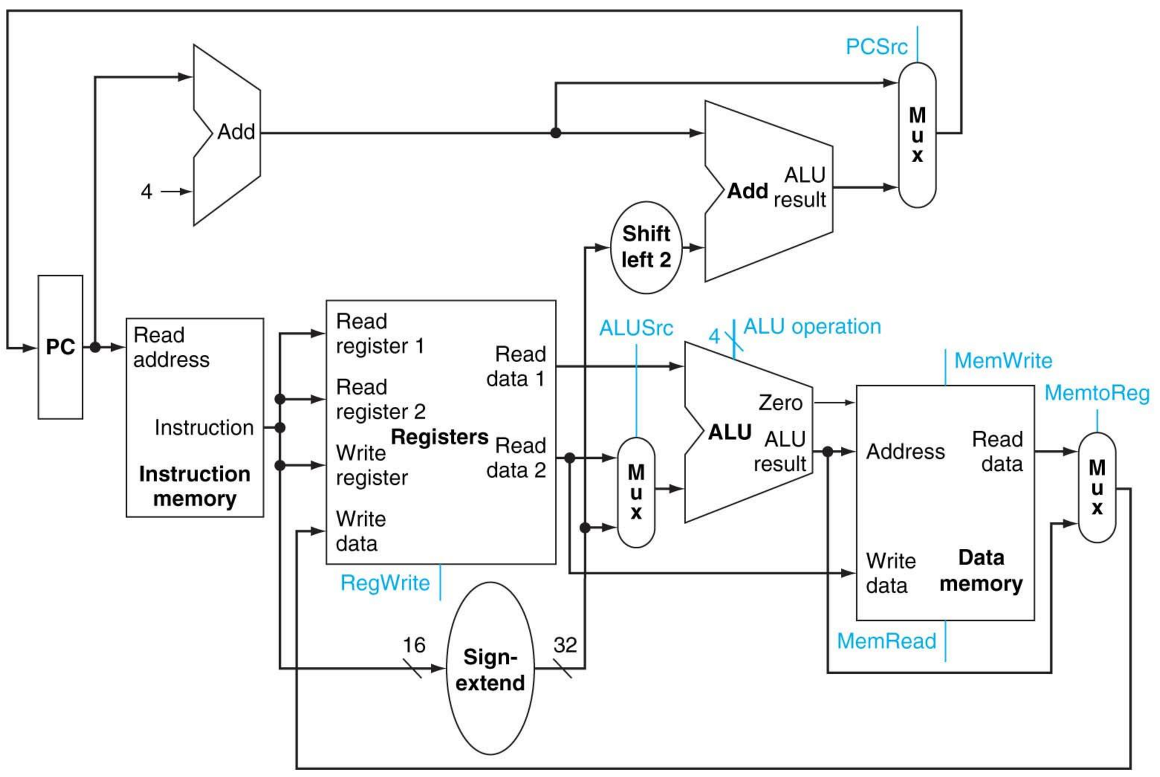

Figure 2: The simple datapath for the MIPS architecture combines the elements required by different instruction classes.This datapath can execute the basic instructions (load-store word, ALU operations, and branches) in a single clock cycle.

d. Which kinds of instructions require this resource?
e. For which kinds of instructions (if any) is this resource on the critical path?
f. Assuming that we only support `beq` and `add` instructions, discuss how changes in the given latency of this resource affect the cycle time of the processor. Assume that the latencies of other resources do not change.

2. Assuming the following latencies for logic blocks in the datapath:

	I-Mem	**Add**	**Mux**	**ALU**	**Regs**	**D-Mem**	**Sign-extend**	**Shift-left-2**
a.	400ps	100ps	30ps	120ps	200ps	350ps	20ps	0ps
b.	500ps	150ps	100ps	180ps	220ps	1000ps	90ps	20ps

a. What is the clock cycle time if the only type of instructions we need to support are ALU instructions (`add`, `and`, etc.)?
b. What is the clock cycle type if we only had to support `lw` instructions?
c. What is the clock cycle time if we must support `add`, `beq`, `lw`, and `sw` instructions?

For the remaining problems, assume that there are no pipeline stalls and that the breakdown of executed instructions is as follows:

	`add`	`addi`	`not`	`beq`	`lw`	`sw`
a.	30%	15%	5%	20%	20%	10%

b.	25%	5%	5%	15%	35%	15%

d. In what fraction of all cycles is the data memory used?
e. In what fraction of all cycles is the input of the sign-extend circuit needed? What is this circuit doing in cycles in which its input is not needed?
f. If we can improve the latency of one of the given datapath components by 10%, which component should it be? What is the speed-up from this improvement?

3. Consider a 5-stage (`IF`, `ID`, `EX`, `MEM`, `WB`) MIPS pipeline processor with hazard detection unit. Suppose the processor has instruction memory for `IF` stage, and data memory for `MEM` stage so that the structural hazard for memory references can be avoided.

   a. Assume *no forwarding unit* is employed for the pipeline. We are given a code sequence shown below.

```
LD    R1, 10(R2);        #R1 ← MEM[R2 + 10]
SUB   R4, R1, R6;        #R4 ← R1 - R6
ADD   R5, R1, R6;        #R5 ← R1 + R6
```

   Show the *timing of each instruction* of the code sequence. Your answer may be in the following form.

Instruction		Clock Cycle									
		1	2	3	4	5	6	7	8	9	10
LD	R1, 10(R2)	IF	ID	EX	MEM	WB					
SUB	R4, R1, R6										
ADD	R5, R1, R6										

   b. Consider another code sequence shown below.

```
SUB   R1, R3, R8;        #R1 ← R3 - R8
SUB   R4, R1, R6;        #R4 ← R1 - R6
ADD   R5, R1, R6;        #R5 ← R1 + R6
```

   Suppose both hazard detector and forwarding unit are employed. Show the timing of each instruction of the code sequence as the above table.

4. The following problems refer to the following sequence of instructions:

	Instruction sequence

a.	`lw $1, 40($6)`
	`add $6, $2, $2`
	`sw $6, 50($1)`

b.	`lw $5, -16($5)`
	`sw $5, -16($5)`

	`add $5, $5, $5`

a. Indicate dependences and their type.
b. Assume there is no forwarding in this pipelined processor. Indicate hazards and add `nop` instructions to eliminate them.
c. Assume there is full forwarding. Indicate hazards and add `nop` instructions to eliminate them.

The remaining problems assume the following clock cycle times:

	Without forwarding	**With full forwarding**	**With ALU-ALU forwarding only**
a.	300ps	400ps	360ps
b.	200ps	250ps	220ps

d. What is the total execution time of this instruction sequence without forwarding and with full forwarding? What is the speed-up achieved by adding full forwarding to a pipeline that had no forwarding?
e. Add `nop` instructions to this code to eliminate hazards if there is ALU-ALU forwarding only (no forwarding from the MEM to the EX stage)?
f. What is the total execution time of this instruction sequence with only ALU-ALU forwarding? What is the speed-up over a no-forwarding pipeline?

5. Identify all of the data dependencies in the following code. Which dependencies are data hazards that will be resolved via forwarding? Which dependencies are data hazards that will cause a stall?

```
add  $2, $5, $4
sub  $4, $2, $5
lw   $5, 100($2)
add  $3, $5, $2
```