CENG3420 Homework 2

Due: Mar. 02, 2019

Please submit PDF or WORD document directly onto blackboard.
DO NOT SUBMIT COMPRESSED ZIP or TARBALL.

Q1 (15%) The basic single-cycle MIPS implementation in Figure 1 can only implement some instructions. New instructions can be added to an existing Instruction Set Architecture. Following questions refer to the new instruction:

Instruction `lwi Rt, Rd(Rs)`

Interpretation `Reg[Rt] = Mem[Rd + Reg[Rs]]`

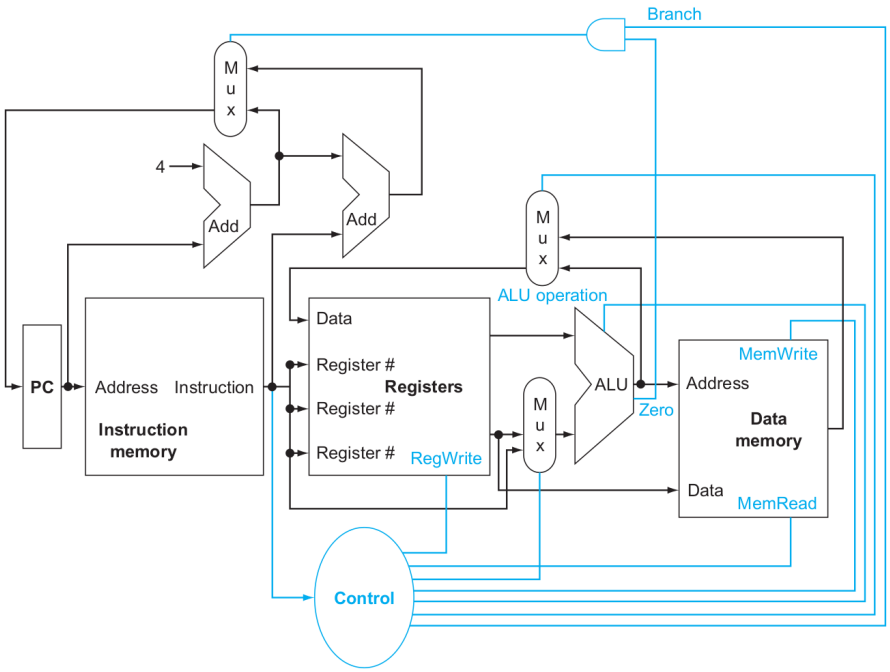

Figure 1: The basic implementation of the MIPS subset, including the necessary multiplexors and control lines.

1. Which existing blocks (if any) can be used for this instruction?
2. Which new functional blocks (if any) do we need for this instruction?
3. What new signals do we need (if any) from the control unit to support this instruction?

Q2 (15%) Following problems assume that logic blocks needed to implement a processor's datapath have the following latencies (Table 1):

1. If the only thing we need to do in a processor is fetch consecutive instructions (Figure 2), what would the cycle time be?

Table 1: Question 2

Item	I-Mem	Add	Mux	ALU	Regs	D-Mem	Sign-Extend	Shift-Left-2
Latency (ps)	750	200	50	250	300	500	100	0

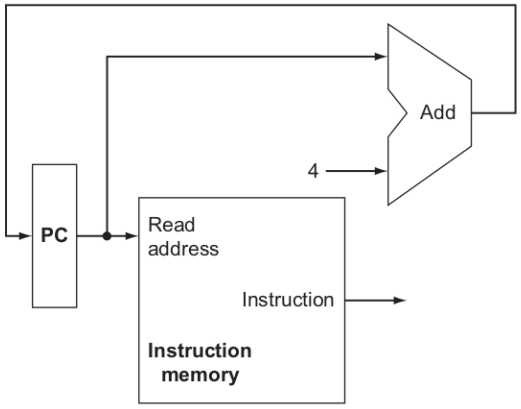

Figure 2: A portion of the datapath used for fetching instructions and incrementing the program counter.

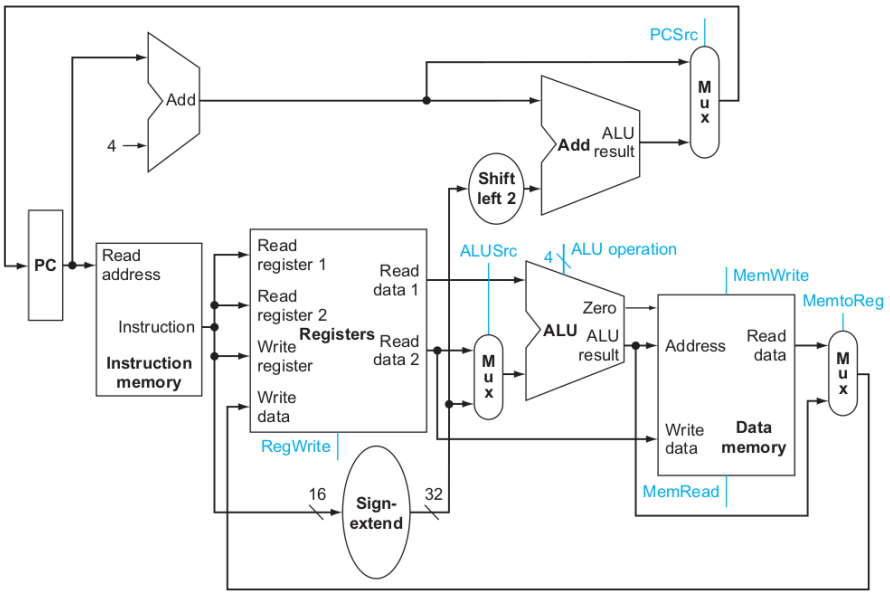

Figure 3: The simple datapath for the core MIPS architecture combines the elements required by different instruction classes.

2. Consider a datapath similar to the one in Figure 3, but for a processor that only has one type of instruction: unconditional PC-relative branch. What would the cycle time be for this datapath?
3. Repeat 2, but this time we need to support only conditional PC-relative branches.

Q3 (15%) Given the following specs of the datapath latencies:

1. What is the clock cycle time in a pipelined and non-pipelined processor?

Stages	IF	ID	EX	MEM	WB
Latencies (ps)	200	170	220	210	150

2. What is the total latency of an `LW` instruction in a pipelined and non-pipelined processor?
3. 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?

Q4 (15%) Regarding the following instructions:

```
I1:  lw  R1, 40(R2)
I2:  add R2, R3, R3
I3:  sw  R2, 50(R1)
```

1. Indicate dependencies and their type.
2. Assume there is no forwarding in this pipelined processor. Add `nop` instructions to eliminate hazards.
3. Assume there is full forwarding. Indicate hazards and add `nop` instructions to eliminate them.

Q5 (10%) Regarding the following MIPS instruction:

```
sw R16, -100(R6)
```

1. Which registers need to be read, and which registers are actually read?
2. What does this instruction do in the EX and MEM stages?

Q6 (15%) Given the following loop, assume that perfect branch prediction is used (no stalls due to control hazards), that there are no delay slots, and that the pipeline has full forwarding support. Also assume that many iterations of this loop are executed before the loop exits.

```
loop:  add R1, R2, R1
       lw  R2, 0(R1)
       lw  R2, 16(R2)
       slt R1, R2, R4
       beq R1, R9, loop
```

1. Show a pipeline execution diagram for the third iteration of this loop, from the cycle in which we fetch the first instruction of that iteration up to (but not including) the cycle in which we can fetch the first instruction of the next iteration. Show all instructions that are in the pipeline during these cycles (not just those from the third iteration).
2. How often (as a percentage of all cycles) do we have a cycle in which all five pipeline stages are doing useful work?
3. At the start of the cycle in which we fetch the first instruction of the third iteration of this loop, what is stored in the IF/ID register?

Q7 (15%) Regarding the following instruction sequences:

```
add R1, R2, R1
```

```
lw  R2, 0(R1)
lw  R1, 4(R1)
or  R3, R1, R2
```

1. Find all data dependences in this instruction sequence.
2. Find all hazards in this instruction sequence for a 5-stage pipeline with and then without forwarding.
3. To reduce clock cycle time, we are considering a split of the MEM stage into two stages. Repeat 2 for this 6-stage pipeline.