CENG3420 Homework 2

Due: Apr. 07, 2020

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 `sw Rs, Im(Rt)`

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

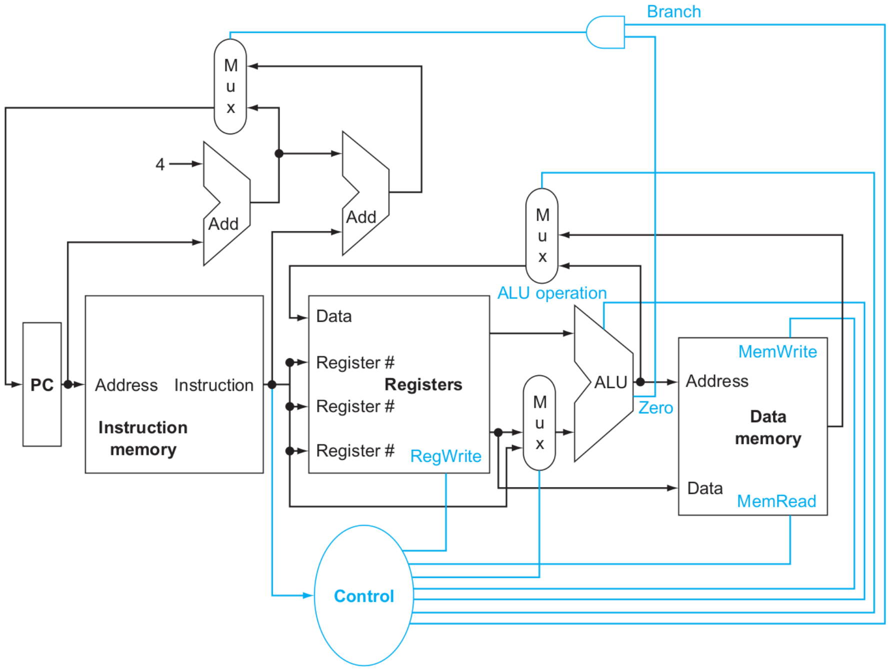

Figure 1: The basic implementation of the MIPS subset, including the necessary multiplexors and control lines.

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)	350	100	20	120	120	400	20	5

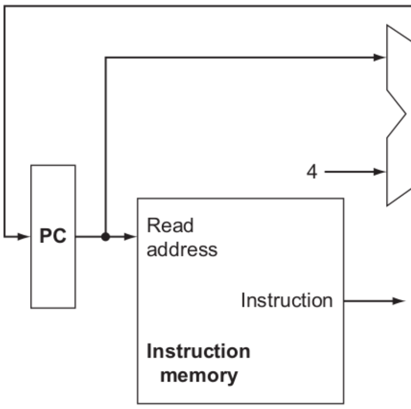

Figure 2: A portion of the datapath used for fetching instructions and incrementing the program counter.

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? Destribe the critical path and the function of each unit.

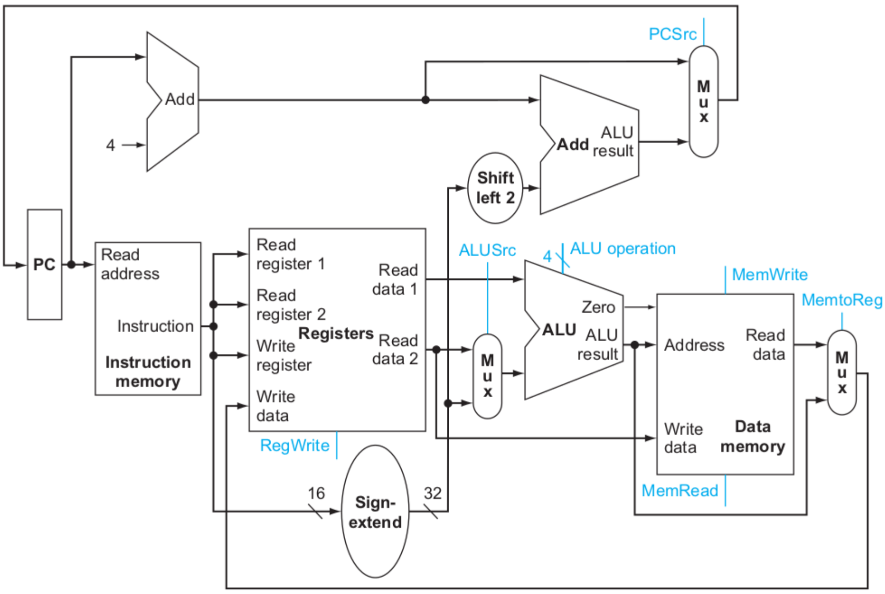

Figure 3: The simple datapath for the core MIPS architecture combines the elements required by different instruction classes.

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	160	250	200	140

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 (10%)

alu	beq	lw	sw
50%	25%	15%	10%

1. Assuming there are no stalls or hazards, what is the utilization of the data memory?
2. Assuming there are no stalls or hazards, what is the utilization of the write-register port of the "Registers" unit?

Q5 (15%) You are required to develop some simple measures of pipeline performance and relative speedup.

1. Let $T_{k,n}$ be the total time required for a pipeline with k stages to execute n instructions. Speedup of k stage pipeline is given by,

$$S_k = \frac{T_{1,n}}{T_{k,n}}. \tag{1}$$

   Determine S_k in terms of k and n.
2. Consider an instruction sequence of length n that is streaming through the instruction pipeline. Let p be the probability of encountering a conditional or unconditional branch instruction, and let q be the probability that execution of a branch instruction I causes a jump to a nonconsecutive address. Assume that each such jump requires the pipeline to be cleared, destroying all ongoing instruction processing, when I emerges from the last stage. Determine S_k in terms of k, n, p and q.

Q6 (15%) Given the following loop,

```
loop:  add R1, R2, R3
       sub  R4, R1, R3
       lw  R6, 0(R4)
       slt  R6, R7, R6
       beq  R1, R6, loop
```

1. 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. Show a pipeline execution diagram for the second 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.

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. Show a pipeline execution diagram by inserting stalls to resolve data hazards.

Q7 (15%)

Table 2: Instruction Category

R-type	beq	jmp	lw	sw
40%	25%	5%	25%	5%

Table 3: Predictor Accuracy

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

The importance of having a good branch predictor depends on how often conditional branches are executed. Together with branch predictor accuracy, this will determine how much time is spent stalling due to mispredicted branches. In this exercise, assume that the breakdown of dynamic instructions into various instruction categories is as Table 2. Also, assume that the branch predictor accuracies are as Table 3.

1. Stall cycles due to mispredicted branches increase the CPI. What is the extra CPI due to mispredicted branches with the always-taken predictor? Assume that branch outcomes are determined in the EX stage, that there are no data hazards, and that no delay slots are used.
2. Repeat 1 for the "always-not-taken" predictor.
3. Repeat 1 for for the 2-bit predictor.