ICS 152, Problem Set 3

- Please show your work.
- Bottom line answers without proper explanation are worth **zero** points.

1. No changes are needed in the datapath. The new variant is just like lw except that the ALU will use the Read data 2 input instead of the sign-extended immediate. However, the instruction format will need to change. The data will be written to register specified by the rd field (instead of the rt field). To modify the control, we simply need to add a new row the existing control table, with RegDst = 1, ALUSrc = 0, MemtoReg = 1, RegWrite = 1, MemRead = 1, MemWrite = 0, ALUOp = 00

2. We need to perform two add instructions, one in the lw instruction and one in the addi instruction – we could use the ALU for one and the adder that computes PC + 4 for the other. However, we cannot write to two registers simultaneously, and hence (without changing the register file), it is impossible to perform the above operations in a single cycle.

3. a) Using a temp register (add or addi)

```
add $t1, $rs, $zero
add $rs, $rt, $zero
add $rt, $t1, $zero
```

b) Without a temp register

```
sw $rs, offset(baseaddr)
add $rs, $rt, $zero
lw $rt, offset(baseaddr)
```

c) Let s the number of swap instructions and x the rest
$1.1 (x+s) <= (x+3s)$
Therefore, if more than 5% of the instructions are swaps, the implementation is recommended.

4. The load instruction is the critical path. Thus, only the two adders can tolerate up to 400 ps delay. All the other components would benefit from further improvements.

5. Obviously M2 is the fastest one with the given instruction mix. However, if we increase the number of load instructions in the mix to 45%, M3 becomes the fastest one.

6. a) The single memory access CPI is $5(.25)+4(.1)+4(.52)+3(.11)+3(.02) = 4.12$.
The two-cycle memory access CPI is $6(.25)+5(.1)+4(.52)+3(.11)+3(.02) = 4.47$.
Thus, the machine with the two-cycle memory accesses is 7% faster than the machine with a single memory access.
b) Splitting instruction fetch adds one extra cycle to every instruction and will not increase performance.