Name: _____

CIS 351 Practice Final

Monitor Piazza for details about the exam. None of these problems are due for credit.

1. Review the cache homework
2. Review the not-for-credit cache activities on PL
3. Describe the reasoning behind including cache memory in a system versus just a CPU, RAM, and hard drive.
4. Caches are designed to realize temporal and spatial locality. Give context for each of the localities and describe how a cache is configured to make use of them.
5. Explain the relationship between memory and a cache. How do we map memory to the cache? How do we check if an item is in the cache (What items are checked?). Be detailed.
6. Be prepared to calculate the miss rate for a given piece of assembly code and a specified cache.
7. What effect(s) can changing the block size have on a cache? (Hint: Pick one type of cache and explain for it.)
8. For pipelining, review the Week 14 activities from number 8 onwards, the Week 15 activities, and the Pipelining lab.
9. Explain the benefits of a pipelined CPU over a single cycle CPU. Explain possible drawbacks of pipelining.
10. Consider playing a video game with two friends. The three of you decide you want to start a magic-sword business. One of you is a strong miner who can get ore, another is good at turning ore into refined metal, and the third can turn metal into magic swords. If the miner takes 5 minutes to gather enough ore, the refiner takes 10 minutes to purify the ore into metal, and the blacksmith takes 15 minutes to make a sword from metal, how long will it take you to make 1 sword? 5 swords? What is the long-term rate at which you can make swords?
11. The code below does not have any data hazards. Make a small modification to the code so that it has a data hazard that can be solved without a stall. Add another hazard that can be solved only by stalling the pipeline.

```
add $t0 $t1 $t2
add $t3 $t4 $t5
add $t6 $t7 $t8
```

12. Consider the code below running on our five-stage MIPS pipeline. Assume the pipeline solves hazards by forwarding whenever possible and stalls only when absolutely necessary. Which instruction is in each stage of the pipeline during Cycle 7 (0-indexed)? What are the values on each of the wires of our microarchitecture diagram?

Name: _____

```
add $t0 $t1 $t2
addi $t3 $t4 $t5
sub $s0 $s1 $s2
lw $s4, 0($t3)
slt $a0 $0 $t2
and $s3 $s0 $t1
ori $v0 $t0 12
xori $v1 $t1 18
and $s1 $s2 $s3
```

13. Explain why hazards caused by a load-word instruction cannot be solved without stalling.

14. Given assembly code, identify hazards in the code running on a pipelined CPU. Explain why they are a problem and how they are mitigated.

15. For each instruction below, explain where the operands to the ALU come from when that instruction in in the Execute stage (i.e., whether they are forwarded and, if so, from which stage).

```
add $t0 $t1 $t2
add $t0 $t4 $t5
add $t6 $t1 $t8
add $t1 $t0 $t6
add $t3 $t0 $t6
```

16. Explain what branch prediction is and why it is "safe" – that is, why it does not result in changing the outcome of a program if we guess wrong.

17. Explain any benefits and drawbacks of branch prediction compared to stalling.

18. Explain why branch prediction often has better than 50 percent accuracy.

19. Explain the relationship between the stage in which a branch decision is made and the branch-misdirection penalty.

20. Consider the following code running on a machine that computes its branch decision in the Decode stage (such as the final version of our pipelined microarchitecture from the slides). If the branch instruction enters the pipeline during Cycle 0, what instruction, if any, is in the Decode stage during Cycle 2?

```
beq $0 $0 done
add $t6 $t1 $t8
add $t1 $t0 $t6
add $t3 $t0 $t6
done:
sub $t3 $t0 $t6
sub $t1 $t0 $t6
```