|  |  |  |
| --- | --- | --- |
| PROF. AKKARY | DEPT. OF ELECTRICAL AND COMPUTER ENGINEERING | November 10, 2011 |
|  | AMERICAN UNIVERSITY OF BEIRUT |  |
|  | **EECE421–COMPUTER Architecture** |  |
|  | **Quiz 1 – Fall 2011** |  |
|  |  |  |
| **NAME**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |
| **ID**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  |  |

**INSTRUCTIONS:**

* **The duration of the exam is 90 minutes (1.5 hours).No time extension.**
* **The exam is closed-book/closed-notes.**
* **Using Cell phones is not allowed in the examination room.**
* **Write your name and ID. NumBer in the space provided above.**
* **Circle only one answer.**
* **READ THE QUESTIONS CAREFULLY BEFORE ANSWERING.**
* **in some questions, more than one choice may be a valid answer. Circle the best choice you think is the most appropriate answer to the question.**
* **There is no penalty for wrong answers.**
* **Use the back pages for scratch if needed**
* **Check that you have a total of 5 pages.**
* **No questions are allowed.**
* **You cannot leave the exam room for any reason until you complete the exam.**
1. **[3 points]**Which of the following statements about pipelining is **FALSE**:
	1. Out-of-order issue causes out-of-order completion.
	2. Write-after-read hazards do not occur in in-order issue pipelines.
	3. Write-after-write hazards can occur in in-order issue pipelines.
	4. Optimizing compilers make write-after-write and write-after read hazards more probable.
	5. Read-after-write hazards can be avoided with register renaming.
2. **[3 points]**Which of the following statements is **TRUE** about the CDC computer that preformed dynamic scheduling with scoreboard:
	1. It used in-order issue and therefore did not encounter write-after-read hazards.
	2. It avoided write-after-read hazards by stalling some writes to the register file.
	3. It performed out-of-order completion and used reorder buffer to handle branch mispredictions.
	4. It avoided read-after-write hazards by stalling at the decode stage.
	5. It avoided write-after-write hazards by enforcing in-order completion using the scoreboard.
3. **[3 points]**Which of the following statements is **TRUE**:
	1. The reorder buffer is a buffer for instruction results.
	2. All computers that feature out-of-order execution use a reorder buffer file.
	3. Dynamic scheduling with scoreboard provides better performance than a reorder buffer.
	4. Instructions read operands or tags from the reorder buffer in program order.
	5. Instructions write results to the reorder buffer in-order.
4. **[3 points]**Select the **TRUE** statement:

A processor usesTomasulo’s algorithm to execute the following code segment. The latency of each operation is also shown.

Add R1, R2, R3 (2 cycles)

Mul R1, R1, R4 (20 cycles)

Ld R2, 0(R4) (3 cycles)

Sub R2, R1, R5 (1 cycle)

* 1. Issue order is Mul, Add, Ld, Sub
	2. Issue order is Add, Sub, Ld, Mul
	3. Issue order is Ld, Sub, Mul, Add
	4. Issue order is Mul, Ld, Sub, Add
	5. None of the above
1. **[3 points]**Which of the following statements is **TRUE**:
	1. Tomqsulo’s algorithm avoids write-after-write hazards by register renaming.
	2. Tomasulo’s algorithm avoids write-after-write hazards by stalling at the decode stage.
	3. Tomasulo’s algorithm avoids write-after-write hazards by using a reorder buffer.
	4. Tomasulo’s algorithm avoids write-after-write hazards by writing results into the register file in-order.
	5. Tomasulo’s algorithm avoids write-after-write hazards using tag matching in the RF.
2. **[3 points]**Which of the following statements is **TRUE**:
	1. Tomasulo’s algorithm avoids write-after-read hazards by writing results into the register file in-order
	2. Tomasulo’s algorithm avoids write-after-read hazards by stalling at the decode stage.
	3. Tomasulo’s algorithm avoids write-after-read hazards using tag matching in the register file.
	4. Tomasulo’s algorithm avoids write-after-read hazards by using a reorder buffer.
	5. None of the above is TRUE.
3. **[3 points]**Choose the one **FALSE** statement about addressing modes.
	1. Indirect branches get the upper 4 bits of the address from the PC.
	2. Base addressing adds an offset to a base address.
	3. Immediate addressing gets its operand from the instruction.
	4. PC relative addressing computes the target from the PC and a displacement.
	5. Indirect branches can jump to any address in memory.
4. **[3 points]**Which of the following is **TRUE**?
	1. The most significant factor in computer performance is always the performance of its CPU.
	2. The length of out-of-order superscalar pipeline has no impact on performance.
	3. Adding processors to a computer system that uses multiple processors for separate tasks usually increases throughput.
	4. Benchmarks are programs specifically chosen to debug the dapapath and control of a microprocessor.
	5. The Millions of Instructions per Second (MIPS) is the only valid metric to measure performance of a computer.
5. **[3 points]**Dynamic CPU power is given by: P = ½ CV2freq.
	1. True
	2. False
6. **[3 points]**The relative performance of two processors with the same instruction set architecture (ISA) can be judged by the number of instructions required to execute a program.
	1. True
	2. False
7. **[3 points]**Which of the following statements about pipelining is **FALSE**:
	1. Pipelining improves performance by increasing instruction throughput.
	2. Pipelining does not decrease the execution time of an individual instruction.
	3. The ideal CPI of a pipelined processor is at most one.
	4. The ideal speed-up of a *k*-stage pipelined processor over a single-cycle processor is 1/*k*.
	5. Sophisticated addressing modes that update registers can complicate hazard detection.
8. **[3 points]**A machine may be enhanced by adding vector hardware. When a computation is run in vector mode, it runs at 4 times faster than in normal mode, assuming the same execution frequency. However, the vector hardware increases the circuit complexity and consequently increase the cycle time by 6%.

 The machine designer evaluates a benchmark suite of 10 applications and finds out that 5 applications can benefit from the vector hardware for 16% of the computation and 5 applications do not benefit from the vector hardware at all.

* 1. Calculate the overall speedup for each of the 10 applications.

For the machines that benefit: speedup = (1/(0.84 +0.04))/1.06 = 1.072

For the machines that do not benefit: speedup = 1/1.06 = 0.943

* 1. Calculate the arithmetic mean of the speedup over all 10 applications.

Arithmetic mean: 0.5\*1.072 + 0.5\*0.943 = 1.0075

1. **[7 points]**Assume that we make an enhancement to a computer that improves some mode of execution by a factor of 5. Enhanced mode is used 80% of the time, measured as a percentage of the execution time *when the enhanced mode is in use.* Recall that Amdahl’s Law depends on the fraction of the original, *unenhanced* execution time that could make use of the enhanced mode. Thus, we cannot directly use this 80% measurement to compute speedup with Amdahl’s Law.
	1. What is the speedup we have obtained from fast mode?

Let T be new execution time.

Old execution time = 5 \* 0.8T + 0.2T = 4T + 0.2T = 4.2 T

Speedup obtained = 4.2T / T = 4.2

* 1. What percentage of the original execution time has been converted to fast mode?

Percentage of execution converted to fast mode: 4T/4.2T = 0.952

1. **[7 points]**Assume the following machine and application parameters:
* 1 out of 8 instructions on average is a conditional branch.
* Branch misprediction rate: 10%.
* Average fetch to branch execution pipeline time: 30 cycles.
* Cache miss rate to DRAM: 2 misses per 1000 instructions.
* Load cache miss to data return from DRAM: 300 cycles.
* Unlimited Instruction Level Parallelism, i.e. no pipeline stalls due to data hazards.
* Cycle time of the 4-wide superscalar is 90% of the cycle time of the 8-wide processors.
	1. Calculate the relative performance of 4 wide and 8 wide superscalars.

4-wide processor CPI = 0.25 + 0.1\*30/8 + 2\*300/1000 = 1.225

8-wide processor CPI = 0.125 + 0.1\*30/8 + 2\*300/1000 = 1.1

Relative performance 0.9\*1.225/1.1 = 1.002

8-wide processor is only 0.2% faster than the 4-wide processor!