Rules of the Examination

Closed book, closed notes. No calculators, cell phones, PDAs or other electronic devices. Just you, your writing instrument, and the exam paper. Answer all questions. Please keep you answers concise and confined to the area provided for each question. **Any answer not contained in the space immediately following the question will not be graded!** Be sure to address the actual question asked.

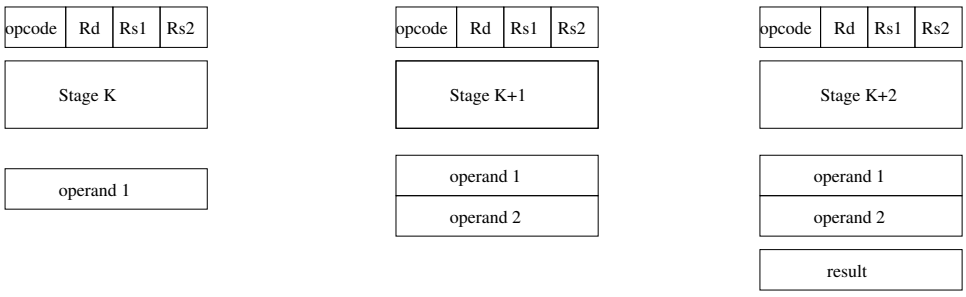
Questions

1. **(2 points)** True/False: The 90/10 rule states that 90 percent of branches are taken and 10 percent of branches are not taken.
2. **(2 points)** True/False: A write no-allocate policy permits a cache memory to hold a data write for virtual memory from disk that is not currently present in the main memory/cache hierarchy.
3. **(2 points)** True/False: While local branch prediction can generally correctly predict 40% of branches, non-local branch predictors can greatly improve this to 80-90%.
4. **(2 points)** True/False: In addition to architectural attacks on hazards, effectively compiler optimization can have dramatic impact on program performance. Sometimes compilers can improve program performance by up to 90%.
5. **(2 points)** True/False: Increasing the associativity reduces conflict misses, but has the cost of increasing the hit time.
6. **(2 points)** True/False: Conflict misses cannot occur in set associate caches.
7. **(2 points)** True/False: In measures of the effectiveness of dynamic branch prediction, Patterson & Hennessy show that a branch predictor with 4095 entries and 2-bits per entry is sometimes more effective thank a predictor with an infinite number of 2-bit entries (for the benchmark programs studied).
8. **(2 points)** True/False: In general, the average memory access time performance for memory reads is far more important than it is for memory writes.

9. **(10 points)** Assume branches are predicted to occur with a frequency of 20% of all instructions executed and that you are evaluating the potential speedup of a 5 stage pipeline that must stall 3 cycles for each branch outcome. Show quantitatively the expected speedup (from a non-pipelined machine) from pipelining in the presence of of these branch stall costs.

10. **(10 points)** Show the average memory access time for the following memory design. A two level cache system with a hit time for each of: L1 (3 cycles) and L2 (9 cycles). Assume that the design can instantiate an L2 lookup one machine cycle after the L1 lookup begins.

11. **(10 points)** Consider the following pipeline structure:



Assume that all operands are specified to registers by the instruction fields Rd for the result, Rs1 for the first operand and Rs2 for the second operand. Assume further that Stage K fetches operand 1 for instruction $i + 2$, Stage $K + 1$ fetches operand 2 for instruction $i + 1$, and Stage $K + 2$ produces a result for instruction i. Add the circuits needed to implement data forwarding to overcome RAW hazards. You can use as primitives mutiplexers, demultiplexers, encoders, decoders, comparators, and any logic gates you require.

12. **(10 points)** Assume you are evaluating a memory system design where you are considering the trade-off between space for a larger TLB. At its current design, the TLB is expected to be effective for 85% of all memory references. It is expected that the increased in size will increase its effectiveness to 95%. However, the larger size will add an additional one cycle delay in address translation which will also increase the hit time to L1 by one clock cycle. Derive a quantitative assessment of the performance impacts of each.

13. **(10 points)** Consider the original Pentium design that is able to schedule two integer operations for simultaneous execution. Outline the main building blocks of the general solution they applied to use interleaved memory to ensure independence of the memory accesses by the concurrently executing integer units (do not worry about the details of the execute units, simply provide a block diagram of the circuit that can guarantee that the two integer units do not simultaneously access the same data). As with the original Pentium design, your solution must use only a 1-bit of the memory addresses to determine conflict. Be sure to show how the address bits are used to control access to the memory spaces.