Computer Architecture: Spring 2001 – Test 3 Solutions

The test is closed book, closed notes. You may only use the provided reference material. Please keep your discussion answers succinct.

1. (24 pts) For the code sequences below, identify if (a) there is a data hazard or not, (b) if there is a stall or not, (c) in the case of a data hazard, what forwarding path(s) are activated (an example forwarding path would be "fwd MEM stage result to EXE stage). Be sure to circle the register(s) that cause the hazard and show the linkage between the instructions that form the hazard.

a.
```
add $4, $6, $7
add $8, $4, $7
add $7, $6, $4
```
Data hazard, no stall, fwd WB to EXE

b.
```
sw  $4, ($6)4
lw  $3, ($4)8
add $4, $6, $8
```
No data hazard

c.
```
add $7, $9, $10
sw  $10, ($7)4
lw  $7, ($9)4
```
Data hazard, no stall, fwd MEM to EXE

d.
```
lw  $8, ($4),8
add $4, $5, $10
add $2, $8, $10
```
Data hazard, no stall, fwd WB to EXE

e.
```
add $4, $5, $10
and $10, $4, $4
sw  $5, ($11)8
```
Data hazard, no stall, fwd MEM to EXE

f.
```
lw  $4, ($5)8
add $7, $5, $4
add $5, $8, $5
```
Data hazard, STALL between lw and add, fwd WB to EXE

g.
```
lw  $4, ($5),8
sw  $4, ($3)4
add $5, $3, $2
```
Data hazard, no stall, fwd WB to MEM

2. (8 pts)
   a. In a pipelined machine, explain what constitutes a *control* hazard?
A control hazard occurs when branch is guessed wrong -- this require that instructions that are already in the pipeline that were fetched after the branch and before the branch outcome was known to be turned into NOPs.

   b. What hardware was added and where was it added to the MIPS pipelined implementation to reduce the impact of a control hazard? Explain how this reduced the impact of a control hazard.
A comparator was placed in the DECODE stage to determine the branch outcome of the BNE, BEQ branches. The sooner we know the outcome of a branch the less impact branch hazards will have on the pipeline.

3. (3 pts)What extra information needs to be passed to the execution stage from the decode stage in order to detect if a data hazard exists or not? This information was not passed in the multiple-clock cycle or single clock cycle MIPS implementations.
The register NUMBERs of the instruction operands were passed from Decode to Execute. These were compared with the destination numbers of instructions in the MEM, WB stages to determine if forwarding was needed.

4. (6 pts) Given a 256K byte cache, a maximum physical address size of 2 Gigabytes, and a cache block size of 64 bytes.

   a. For a direct mapped cache, how many bits are allocated to the TAG??
2 Gigbytes = 2^{31}, so 31 bits in address. 256K bytes = 2^{18}, so address bits needed for index/block select. 31 - 18 = 13 bits for tag.

   b. If the cache design was changed to an 8-way set associative cache of the same size and same block size, how many bits are allocated to the TAG?
8 way set associative = 2^3, means three bits are taken from index and added to tag. New tag = 16 bits.

5. (3 pts) What type of cache gives the lowest miss rate? *Fully associative cache.*

6. (6 pts) Two types of misses in a cache are called capacity misses and conflict misses. How would you change a cache design to reduce capacity misses? How would you change a cache design to reduce conflict misses? In both cases, only change one parameter of the cache design.
Increasing cache size reduces capacity misses. Increasing associativity reduces conflict misses.

7. (6 pts) What are the two principles of *locality* that form the rationale for using caches? Explain each.
temporal locality - an item referenced now will likely be referenced in the near future (temporary variables, instructions in loops).
spatial locality - if item A is referenced, then items near item A are likely to be referenced.

8. (3 pts) What extra status bit would I expect to find in a copyback cache that would not be needed in a write-through cache?
Dirty Bit -- set to a '1' if the block is written to.

9. (4 pts) What is the difference between a non-allocate write policy and an allocate write policy?
On a write miss, a non-write allocated policy writes only to the block in main memory. An allocate write policy will the write the block in main memory, but then bring the block into the cache after the write (will allocate space for it in the cache). The allocate policy means that subsequent reads/writes to that block will hit in the cache.

10. (4 pts) Why is a translation lookaside buffer needed to support virtual memory system? What data is kept in a TLB?

A TLB is a speedup mechanism for virtual to physical address calculation. The TLB caches the Page Table entries that are normally kept in the Page table in main memory. The virtual to physical translation mechanism would be too slow if the PTE had to be located in the page table for each translation. NOTE -- the TLB can be DISABLED and the virtual to physical translation will still occur - the TLB is only there to speed up the translation.

11. (4 pts) What is the difference between a virtual cache and a physical cache? How does the TLB figure into this difference?

A virtual cache is referenced by virtual addresses, i.e. the virtual to physical translation has not been performed yet. In this case, the TLB is placed after the cache in the path to main memory. A physical cache is referenced by physical addresses, i.e., the virtual to physical translation takes place before the cache is accessed. In this case, the TLB is placed between the CPU and cache.

(4 pts) In a virtual memory system, assume that I have a 4 Kbyte page size and a 32-bit virtual address. The page table is a linear page table with each page table entry being 4 bytes. Give the address IN HEX of the page table entry that corresponds to the virtual address 0x0008C10F0 .

*4K = 2^{12} (lowest 3 hex digits is page offset) The page offset is 0x 0F0, the Virtual page number is 0x008C1. Each PTE is 8 bytes, so the address of a PTE is 0x008C1 * 8 = 0x 02304.*

12. (6 pts) Draw the finite state machine of a 2-bit dynamic branch predictor and label each state.

See supplemental notes on Chapter 6.

13. (3 pts)Why is a branch address target cache often included on most processors? What is the benefit of a BTAC? (Do not just tell me the function of a BTAC - I want to know why this functionality is important).

Calculation of the branch address adds extra clock cycles to branch execution in most modern CPUs. Remembering this branch address via a cache can speed up branch execution. The fastest branch execution comes from a hit in the branch target address cache plus a correct branch prediction.

14. (3 pts) What is data speculation on the IA-64 architecture?
Because of load latencies, the compiler for the IA-64 will move load instructions several instructions ahead of where the result is actually needed. If the load instruction is moved ahead of a store instruction for which the store address is not known (an ambigious store), then the load becomes a speculative load because the store may affect the load data.

15. (4 pts) What is the ALAT table and how is it used in data speculation on the IA-64?
The address of a speculative load is saved in the ALAT table (Advanced Load Address Table). Before the value of the load is used by another instruction, a check instruction is executed which checks to see if the load address is still in the ALAT table. If it is, then the data value is ok. If the load address has been removed, the load is re-executed along with the instruction that needs the load data. A load address can be removed from the ALAT table by a store instruction whose address range overlaps the load address range.

16. (3 pts) What is predicated execution on the IA-64?
Predicate registers are 1-bit registers in the IA-64. Each instruction on the IA-64 can be tied to a predicate register. If the Predicate register bit is '1', the instruction is executed normally. If the predicate register bit is '0', the instruction is treated as a NOP.

17. (3 pts) Why is predicated execution supported on the IA-64?
Predicated execution is included because it can remove branch instructions. Short sequences of instructions that are protected by a branch can be executed more efficiently via predicated execution since the instructions are fetched as a group in a VLIW machine and already present in the pipeline. Also, the compiler is free to place other instructions between the instruction that sets the predicate register and the instructions that use the predicate register, giving more freedom to the compiler to structure code more effectively.

18. (3 pts) Why is it usually necessary to be able to mark some pages of memory as uncacheable?
Cache locations that map to I/O devices usually needed to be marked as uncacheable since the I/O device can change the contents of the page without the knowledge of the CPU.