# ENGN 2910A Homework 03 (140 points) – Due Date: Oct 15th 2015

Professor: Sherief Reda

School of Engineering, Brown University

---

1. [10 points] Assume the following code snippet is executed on the classical 5-stage pipeline design.

I1: LW R1, 0 (R1)
I2: AND R1, R1, R2
I3: LW R2, 0 (R1)
I4: LW R1, 0 (R3)

a. Find all data dependences in this instruction sequence.

b. Find all hazards in this instruction sequence if executed on the classical MIPS 5-stage pipeline with forwarding and then without forwarding

2. [from Debois *et al.* – 25 points] The following program is executed on the classical 5-stage pipeline design. The program searches an area of memory and counts the number of times a memory work is equal to a key word:

```
SEARCH:    LW R5, 0(R3)          /I1 Load item
           SUB R6, R5, R2        /I2 Compare with Key
           BNEZ R6, NOMATCH      /I3 Check for match
           ADDI R1, R1, #1       /I4 Count for matches
NOMATCH:   ADDI R3, R3, #4       /I5 Next item
           BNE R4, R3, SEARCH    /I6 Continue until all items
```

Branches are predicted untaken always and are taken in EX if needed. Hardware support for branches is included in all cases. Consider several possible pipeline interlock designs for data hazards and answer the following questions for each loop iteration, except for the last iteration.

a. Assume first that the pipeline has no forwarding unit and no hazard detection units. Values are not forwarded inside the register file. Re-write the code by inserting NOPs wherever needed so that the code will execute correctly.

b. Next, assume no forwarding at all, but a hazard detection unit the stalls instructions in ID to avoid hazards. How many clock cycles does it take to execute one iteration of the loop (1) on a match and (2) on a no match?

c. Next, assume internal register forwarding and a hazard detection unit that stalls instructions in ID to avoid hazards. How many cycles does it take to execute one iteration of the loop (1) on a match and (2) on a no match?

d. Next, assume full forwarding and a hazard detection unit that stalls instructions in ID to avoid hazards. How many clocks does it take to execute one iteration of the loop (1) on a match and (2) on a no match?

e. A basic block is a sequence of instructions with one entry point and one exit point. Identify basic blocks (using instruction numbers) in the code. Is it safe for the compiler to move I5 up across the BNEZ. Does this help? How? Is it always safe to move instructions across basic block boundaries?

3. [from Patterson – 30 points] For a direct-mapped cache design with 32-bit address, the following bits of the address are used to access the cache. Tag: 31-12, Index: 11-6, and Offset: 5-0.

a. What is the cache line size (in words)?

b. How many entries does the cache have?

c. What is the ratio between the total bits required for such a cache implementation over the data storage bits?

Starting from the power on, the following byte-addressed cache references are recorded Address: 0, 4, 16, 132, 232, 160, 1024, 30, 140, 3100, 180, 2180.

d. How many blocks are replaced?

e. What is the hit ratio?

f. List the final state of the cache, with each valid entry represented as a record of <index, tag, data>.

4. [from Patterson – 15 points] Assume the following address stream for a word-addressable memory:

3, 180, 43, 2, 191, 88, 190, 14, 181, 44, 186, 253

a. Show the final cache contents for a three-way set associate cache with two-word blocks and a total size of 24 words. Use the LRU replacement. For each reference identify the index bits, the tag bits, the block offset bits, and if it is a hit or a miss.

b. Show the final cache contents for a fully associative cache with one-word blocks and a total size of 8 words. Use LRU replacement. For each reference, identify the index bits, the tag bits, and if it is a hit or a miss.

c. What is the miss rate for a fully associate cache with two-word blocks and a total size of 8 words, using LRU replacement? What is the miss rate using MRU (most recently used) replacement? Finally what is the best possible miss rate for this cache, given any replacement policy?

5. [from Harris – 40 points] Consider a virtual memory system that can address a total of $2^{32}$ bytes but limited to 8 MB of physical memory. Assume that virtual and physical pages are each 4 KB in size.

a. How many bits is the physical address?

b. What is the maximum number of virtual pages in the system?

c. How many physical pages are in the system?

d. How many bits are the virtual and physical page numbers?

e. Suppose that you come up with a direct mapped scheme that maps virtual pages to physical pages. The mapping uses the least significant bits of the virtual page number to determine the physical page number. How many virtual pages are mapped to each physical page? Why is this ”direct mapping” a bad plan?

f. Clearly, a more flexible and dynamic scheme for translating virtual addresses into physical addresses is required than the one described in part (d). Suppose you use a page table to store mappings (i.e., translations from virtual page number to physical page number). How many page table entries will the page table contain?

g. Assume that, in addition to the physical page number, each page table entry also contains some status information in the form of a valid bit (V) and a dirty bit (D). How many bytes long is each page table entry? (Round to an integer number of bytes.)

h. Sketch the layout of the page table. What is the total size of the page table in bytes?

6. [from Harris – 20 points] You decide to speed up the virtual memory system of the previous exercise by using a translation look aside buffer (TLB). Suppose your memory system has the characteristics shown in the given table. The TLB and cache miss rates indicate how often the requested entry is not found. The main memory miss rate indicates how often page faults occur.

a. What is the average memory access time of the virtual memory system before and after adding the TLB? Assume that the page table is always resident in the physical memory and is never held in the data cache.

b. If the TLB has 64 entries, how big (in bits) is the TLB? Give numbers for data (physical page number), tag (virtual page number), and valid bits of each entry. Show your work clearly.

c. Sketch the TLB. Clearly label all fields and dimensions.

| Memory unit | Access Time (Cycles) | Miss rate |
|---|---|---|
| TLB | 1 | 0.05% |
| cache | 1 | 2% |
| main memory | 100 | 0.0003% |
| disk | 1,000,000 | 0% |

d. What size SRAM would you need to build the TLB described in part (c)? Given your answer in terms of depth $\times$ width.