**ECE571: Advanced Microprocessor Design – Homework 7**
TLB – Fall 2024

**Due: Friday 1 November 2024, 12:00pm**

Create a document that contains the data and answers described in the sections below. A .pdf or .txt file is preferred but I can accept MS Office or Libreoffice format if necessary.

1. **TLB Behavior on the Haswell-EP Machine**

   For this section, log into the Haswell-EP machine just like in previous homeworks.

   (a) Run `perf list` on the Haswell-EP machine.
   How many TLB-related events are there in the "cache" and "virtual memory" sections? (Hint, in "less" (the pager used by perf list) you can use the / (slash) character to start a text search).

   (b) First run the traditional naive "matrix-matrix" code. For reference the core loop is listed below. The default size is 1024, which means a total of three arrays * 1024 * 1024 * 8 bytes (a 64-bit double) or roughly 24MB of memory.

```
/* Matrix multiply */
for(i=0;i<size;i++) {
   for(j=0;j<size;j++) {
      for(k=0;k<size;k++) {
         c[(i*size)+j]+=a[(i*size)+k]*b[(k*size)+j];
      }
   }
}
```

   Measure the STLB values; the "STLB" is the second-level TLB which is 1024 entries (8-way associative) on a Haswell-EP machine.

```
perf stat -e mem_uops_retired.stlb_miss_loads,\
mem_uops_retired.stlb_miss_stores,\
page-faults,major-faults,minor-faults \
/opt/ece571/matrix_multiply/matrix_multiply
```

   Report the STLB misses caused by stores and loads, as well as the pagefault types (major and minor).

   (c) Now run the swapped (worse) version where the "i" and "j" loops are switched so instead of walking linearly through memory the array access skip around with a stride of 8*size (8kB).

```
perf stat -e mem_uops_retired.stlb_miss_loads,\
mem_uops_retired.stlb_miss_stores,\
page-faults,major-faults,minor-faults \
/opt/ece571/matrix_multiply/matrix_multiply_swapped
```

   Report the STLB misses caused by stores and loads, as well as the pagefault types (major and minor).

   (d) Answer the following questions:

i. Did the number of TLB misses go up after switching the access order?

ii. How many TLB entries would be needed to cover 24MB of memory when using 4kb pages?

iii. It doesn't look like we see anywhere near the number of page faults we'd expect, which probably means the operating system or hardware is doing things to avoid expensive page faults. What might the OS or hardware be doing to avoid page faults?

2. **TLB and linear vs Random Access**

   (a) Let's now measure TLB misses when linearly walking through memory one byte at a time, using the provided example `memory_walk`

```
perf stat -e mem_uops_retired.stlb_miss_loads,\
mem_uops_retired.stlb_miss_stores,\
page-faults,major-faults,minor-faults \
/opt/ece571/matrix_multiply/memory_walk 240000
```

   (b) Now run a program that does the same amount of memory accesses but does them randomly instead:

```
perf stat -e mem_uops_retired.stlb_miss_loads,\
mem_uops_retired.stlb_miss_stores,\
page-faults,major-faults,minor-faults \
/opt/ece571/matrix_multiply/memory_walk_random 240000
```

   (c) Answer the following questions:

      i. Which way of accessing memory caused the most TLB misses?

      ii. What might explain this difference in behavior?

3. **Read Memory Paper**

   - There aren't any questions on this, but if you want some advance reading on DRAM that we will be covering (after the midterm) you can read this paper: *A Performance & Power Comparison of Modern High-Speed DRAM Architectures* by Li, Reddy, and Jacob.
     `https://user.eng.umd.edu/~blj/papers/memsys2018-dramsim.pdf`

4. **Submitting your work**

   - Create the document containing the data as well as answers to the questions asked.
   - Please make sure your name appears in the document.
   - e-mail the file to me by the homework deadline.