Lab 13

For this lab, you will explore the basic principles of cache by using the cache simulator to derive simple measures about performance. The cache_summary.pdf file attached to this Blackboard post provides valuable details needed to successfully complete this lab.

You must perform this lab on systems4.cs.uic.edu with X11 forwarding enabled. The provided files most likely will not work on another system, so please complete these labs using systems4. ssh -Y username@systems4.cs.uic.edu

1. The labwk13.tar.gz file attached to this Blackboard post provides the cache simulator and files needed to compete the lab. Download and copy this file to systems4.cs.uic.edu. Connect to systems4 using SSH and extract the file using the following command: tar -xzf labwk13.tar.gz. The files within this extracted directory should include cache (executable), blocking.cc, histogram.cc, inst_compact.h, inst_legible.h, inst_none.h, and sort.cc
2. Review section 2 of the cache_summary.pdf, which explains the theory of caches.
3. Perform some simulations using the "instrumented" C++ code histogram.cc. Open the file in your favorite editor and examine what the code is doing. There are two macros defined INST_R and INST_W. These macros are used to provide the addresses of the data being accessed. The macro is included using the preprocessing directive #include using one of the following files (each file provides different functionality).
   inst_legible.h
   Defines the macros to display the addresses to the console for you to view.
   inst_none.h
   Defines the macros to do nothing.
   inst_compact.h
   Defines the macros to output the address for use by the cache simulator.
   Compile the histogram.cc file.
   g++ histogram.cc
   Run the executable.
   ./a.out
   Examine the addresses accessed and the resulting histogram produced.
   The addresses are displayed on the screen. The histogram is provided in the file histogram.txt. Display by using the command: cat histogram.txt

4. Explore the cache simulator. The cache simulator provides a text and graphic mode to visualize cache usage. The cache simulator binary is included in the tar file and is named cache
   Examine the options available by running the help
   ./cache -h
   Make note of the options to enable the graphical display (-g) and the description of the key and mouse inputs accepted by the graphical display
   IMPORTANT: If you receive an error of "permission denied", run chmod +x cache to fix this error.

5. Next, update to use the inst_compact.h header instead, so the cache simulator is able to process the output. This is accomplished by editing histogram.cc and replacing inst_legible.h with inst_compact.h. **Save** and **recompile** histogram.cc.
   Run the below command to ensure the output of cache is shown correctly in the terminal (disables automatic margins to ensure text is not wrapped incorrectly).
   tput rmam
   Run the updated histogram program with the output piped through cache (default text only).
   ./a.out | ./cache
   View the histogram: cat histogram.txt
   Answer the following questions. You must submit your answers in gradescope.

   1. Define spatial locality of reference and temporal locality of reference. (10 points, 5 points per definition)

   2. a) Given the fact that the cache size is 4096 bytes, the block size is 16 bytes, and address is 32 bits, what is the size of Block Offset for the cache organizations given below? (5 points)

      b) Complete the below table for direct mapped, 2-way associative, and fully associative caches. (30 points, 5 points per blank)

	Direct Mapped	2-Way Associative	Fully Associative
Size of Index			
Size of Tag			

      c) Given the address 0xABCDEF01, provide the Block Offset for this address and complete the table given below(34 points, 5 points per blank, 4 points for block offset)

	Direct Mapped	2-Way Associative	Fully Associative
Index			
Tag			

   3. What is the miss rate using the histogram example and the cache organizations from question 2? (21 points, 4 points per command and 3 points per miss rate)

	Direct Mapped	2-Way Associative	Fully Associative
Command Used			
Miss Rate (%)			