Last assignment!

- Individual assignment, not a group assignment
- You can choose to do the design in any language you wish, C, C++, python, Verilog etc
- Submit your code (well commented) and report (pdf) on LMS.
- Rename the filename of your code to <roll_number>_filename.< >
- All codes will run through a plagiarism check. Files found similar with get a 0 for the assignment. Repeat offence will attract Grade penalty on the overall grade
  - Excuses such as “I shared my code with a friend. I was not aware that he/she will upload the same file”, “ I looked at my friends code on zoom, but did not copy”, will not be encouraged.
- Submit by Nov 22 2020, 11:59pm
- Marks: 20 (code) + 10 (results/report)

Problem statement

a. Design a direct mapped cache of size 256kilobytes. Block size: 4 bytes. Assume a 32 bit address. You do not need to implement a main memory, which means you do not need to implement data in the cache. (Implementing a 2^32 memory will be impossible!)

b. Design a 4-way Set associative cache of the same size (256kB)

Output: You need to report hit/miss rates of the two caches for the input memory trace files (5 traces) provided in the next slide.

[Hint: For reporting hit/miss, all you need to check is a tag match, so there is no need to have data in the cache]

Input to your code

- Use the memory trace files at this location (https://cseweb.ucsd.edu/classes/fa07/cse240a/proj1-traces.tar.gz) as input.
- The trace file will specify all the memory accesses/addresses that occur in a certain program. Each line in the trace file will specify a new memory reference and has the following fields:
  - Access Type: A single character indicating whether the access is a load ('l') or a store ('s'). You can ignore this field. For reporting hit/miss, it does not matter whether it is a Load/Store
  - Address: A 32-bit integer (in unsigned hexadecimal format) specifying the memory address that is being accessed. This is the only field you need.
  - Instructions since last memory access: Indicates the number of instructions of any type that executed between since the last memory access (i.e. the one on the previous line in the trace). For example, if the 5th and 10th instructions in the program's execution are loads, and there are no memory operations between them, then the trace line for with the second load has "4" for this field. You can safely ignore this field

Report (pdf format):

- You need to report hit/miss rates of the Direct mapped and Set associative cache for the five memory trace files (5 marks). [You can report it in the form of a table]

- What are your observations from the above experiment (5 marks)

- No need to write anything else in the report