ENSC 453/894 Lab Assignment 2

Lab Assignment Overview:

ENSC 453/894 is a programming-intensive course. You will work on five lab assignments on CPU, FPGA, and GPU programming, which will help you gradually master the highly demanded programming skills in the information and communications technology (ICT) industry sector. You need to form a 2-people team to work on lab 2 to 5 and will be evaluated per team. However, the detailed grading scheme for each lab (which often reveals the answers or clues to the answers) will NOT be released until your lab has been graded: imagine you are in a real interview, nobody will tell you what the detailed grading scheme is.

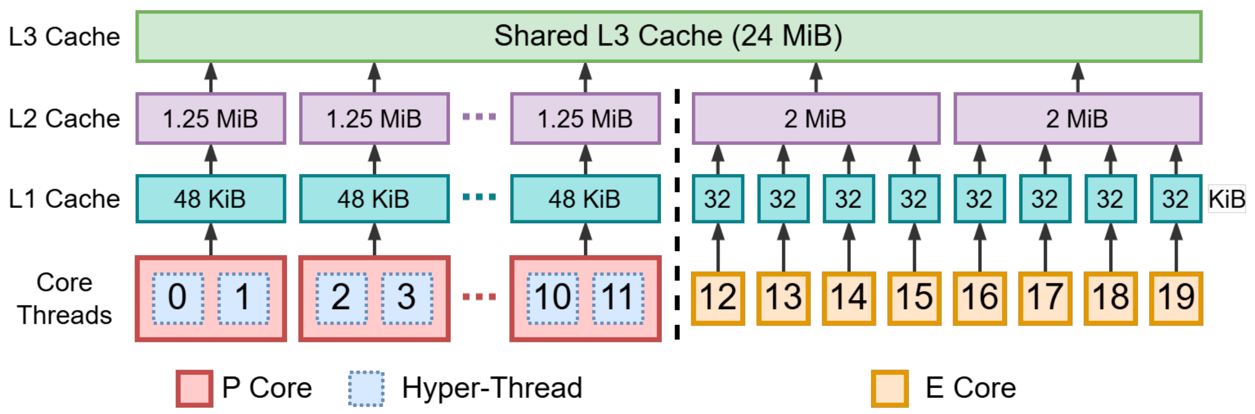
Lab Assignment 2 (10 marks):

In the first lab assignment, you have parallelized matrix multiplication on the Intel(R) Core(TM) i5-13500 CPU in our **FAS-LAB lab machines** using OpenMP. **In lab assignment 2, your task is to further optimize the same *kernel_gemm* function by exploring its cache locality in the FASLAB lab CPU server, as well as parallelizing and vectorizing the loops.**

CPU Configuration: The detailed CPU configurations are summarized in the following table.

# of CPU cores	6 performance cores (P-core, 2 hyper-threads per P-core) + 8 efficient cores (E-core). 20 hyper-threads in total.
SIMD support	It supports AVX2 (256-bit), but not AVX-512 (512-bit)

The i5-13500 CPU in **FAS-LAB lab** has 6 performance cores (P-cores) which are hyper-threaded and 8 efficient cores (E-cores) which are not hyper-threaded, resulting in 20 hyper-threads in total. It supports AVX2 (256-bit) instructions, but not AVX-512 (512-bit) instructions. Such information can be retrieved using the “cat /proc/cpuinfo” command on a Linux machine. Moreover, the “lscpu --all --extended” command can figure out which logical cores (i.e., hyper-threads, see the “CPU” column of the output) belong to which physical core (see the “CORE” column of the output), thus figure out which cores are P-cores (hyper-threaded) and which are cores are E-cores (not hyperthreaded). Specifically, for the i5-13500 CPU, CPU 0-11 (i.e., CORE 0-5) are P-cores, and CPU 12-19 (i.e., CORE 6-13) are E-cores, which is also shown in the figure below.



For the caches, our **FAS-LAB lab** machine includes a three-level cache, as shown in the figure above. Each P-core has a 48 KiB private L1 data cache and a 1.25 MiB private L2 cache. Each Ecore has a 32 KiB private L1 cache, and every 4 E-cores share a 2 MiB L2 cache. All the P-cores and E-cores share a 24 MiB L3 cache. The detailed configurations are shown in the table below and the figure above.

Cache config	Cache for P-cores	Cache for E-cores
Level 1 cache size	48 KiB private, associativity = 12	32 KiB private, associativity = 8
Level 2 cache size	1.25 MiB private, associativity = 10	2 MiB shared, associativity = 16
Level 3 cache size	24 MiB shared cache shared by all cores, associativity = 12	
Cache line size	64 bytes per cache line for all three levels of cache	

Your Tasks: You will decide how to tile the loops based on the loop tiling/blocking technique introduced in Lecture 3 and 5. Also you will decide which loop to parallelize and which loop to vectorize using OpenMP pragmas. For the number of threads, you only need to choose and report the number of threads (one number) that gives you the best performance.

You are also required to write a simple lab report to report the performance speedup over the original single-thread code (from lab assignment 1), including:

- [3 marks] Performance speedups for the single-thread tiled version, under different tiling strategies and tile sizes, and why.
- [1 marks] Performance speedup by adding vectorization to your best single-thread tiled version, and why.
- [3 marks] Performance speedups by the combination of tiling (under different tiling strategies and tile sizes), vectorization, and parallelization (choose your best number of threads), and why. You may have to change the tiling strategies and tile sizes to get the best performance.

In your final code, please only submit the configuration that achieves the best performance. For the performance ranking, please use a matrix size of 4096 * 4096 in your submission. **TA will run all submitted lab 2 codes on the FAS-RLA lab CPU server, and grade based on the performance of your code: the faster your code runs, the higher mark you get. 3 marks (out of total 10 marks) are reserved for performance competition.**

To be eligible for the performance competition, your final program must give the correct result, i.e., the sum of C array should have the same result of the original single-threaded version. Otherwise, you will get 0 point for the performance competition.

Note you CANNOT change the Makefile; otherwise, you will get 0 point for the performance competition. The intention is to test your manual optimizations, not the compiler optimizations.

You can include other manual optimizations to further push forward your program performance. For example, you are allowed to use the X86 SIMD intrinsics to do loop vectorization: https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html. Please document them in your report if you decide to use other optimizations. But keep in mind, you need to make sure your final result is correct; otherwise, you will get 0 point for the performance competition.

For the ranking, the competition marks are given as below:

- Top 2 groups: get 3 marks
- Top 3-6 groups: get 2.5 marks
- Top 7-10 groups: get 2 marks
- Top 11-15 groups: get 1.5 mark
- Top 16-20 groups: get 1 mark
- Top 21-25 groups: get 0.5 marks
- Top 26-29 groups: get 0 mark

Please make sure your code can compile and run correctly on the lab computer. If your code cannot compile, then your group can get at most 5 marks for lab 2. If your code can compile, but cannot run, then your group can get at most 6 marks for lab 2. Always note that in addition to the actual coding for implementation, you need to provide good commenting and coding styles, which count in your lab assignment marking (negative marking).

Assignment Submission:

Your lab assignment 2 will be submitted electronically through Canvas. You will need to submit a single ***YOUR_LAB_GROUP.zip*** file. This format makes it easy for TA to do auto-grading. Please double check the format correctness. Failure to comply with this format will result in a **ZERO** point for this assignment. To zip your files in Linux,

Go to your lab2 directory

1. *Go to your lab2 folder*
2. *make clean //**make sure you clean up your files***
3. *cd .. //go one level up*
4. *mv lab2 **YOUR_LAB_GROUP** //rename your lab2 folder using your lab group*
5. *zip -r YOUR_LAB_GROUP.zip YOUR_LAB_GROUP //zip all your lab2 files into a single .zip*
6. **Note: in addition to all the code, you also need a simple lab report in PDF (described earlier) in the .zip file.**

Submission Deadline:

Your lab assignment 2 is due at **11:59:59pm on Friday, Jan 30, 2026**. You need to meet the deadline: every 10 minutes late submission, you lose 1 mark; that is, 100 minutes late, you will get 0 mark on this lab.

Lab Demonstration:

You will have to demo your lab assignment 2 to the TA in the lab sessions on Tuesday Jan 27, and Thursday, Jan 29, according to your scheduled demo time. **The demo has to be done on the FAS-RAL lab servers**. Only code from your Canvas submission is allowed in the lab demo. Each student group has around 10 minutes to explain your code to the TA. If you fail to do the demo (without a medical note), or if it is determined that you do not understand the code being evaluated, you will be awarded zero on lab assignment 2 and considered as cheating. Also please show up on the demo day on time (TA will send out your scheduled time), otherwise you will lose 0.5 mark.