CRUZ, AIRON JOHN R.
CEPARCO-S11

Project Specification

Question 1 — 100 pts

Write the kernel in (1) C program (2) an x86-64 assembly language (3) x86 SIMD using YMM register. The kernel is to perform dot product between vector *A* and vector *B* and place the result in *sdot*.

As bonus, (4) CUDA kernel as well. Use the best CUDA implementation.

Input: Scalar variable *n* (integer) contains the length of the vector; Vectors *A* and *B* are both **64-bit integer**.

Input: Memory location *n* (integer) contains the length of the vector; memory location *A* and memory location *B* (both **64-bit** integer) are vectors with size *n*.

Process: $sdot = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$

Output: store the result in memory location *sdot*. Display the result for all versions of kernel (i.e., C, x86-64, x86 SIMD using YMM register and CUDA if ever).

Note:

1a.) Write a C main program to call the kernels of C version, non-SIMD x86-64 assembly language and SIMD YMM x86 assembly language version.

1b.) For the bonus CUDA program, C main program to call CUDA kernel.

2.) Time the kernel portion only. Include any overhead time as well (i.e., CUDA data transfer time, page faults etc.)

3.) For each kernel version, time the process for vector size $n = \{2^{20}, 2^{24}, \text{and } 2^{30}\}$. If 2^{30} is not possible, you may reduce it to 2^{29} or at the least up to 2^{28}.

4.) For each version, you need to run at least 30 times and get the average execution time.

5.) For the data, you may initialize each vector with the same or different random value.

6.) You will need to check the correctness of your output. Thus, if C version is your "sanity check answer key", then the output of the x86-64 version and SIMD version have to be checked with the C version and output correspondingly (i.e., x86-64 kernel output is correct, etc.).

7.) output in Github (make sure that I can access your Github):

a.) Github readme containing the following:

- i.) comparative execution time as well as analysis of the performance of different kernels (how many times faster, why is it faster, overheads in calling the kernel, etc.)
- ii.) screenshot of the program output with correctness check (C)
- iii.) screenshot of the program output including correctness check (x86-64)
- iv.) screenshot of the program output including correctness check (SIMD YMM register)
- v.) screenshot of the program output including correctness check (CUDA, optional)

b.) Visual studio project file folder containing **complete** files (source code: x86-64 non-SIMD, x86-64 SIMD version and all other required files) for others to load and execute your program.

c.) Link to your colab CUDA program. Make sure I can access it.