**Homework Assignment #4**

***Due Date: 11/17, 11:59 p.m. Please submit via Blackboard. Late submissions are accepted till 11/21, 11:59 p.m, with 10% penalty each day. For all questions, please note that you need to show the steps how you obtain your result and please do NOT just provide the final answer.***

***Please name your submission file starting with “LastName\_FirstName\_HW4”.***

**Q1. (8 points)** Please briefly explain what a vector architecture is.

**Q2. (10 points)** Please briefly explain what the below two RV64V vector instructions do.

vadd v3, v1, v2

vld v1, r1

**Q3. (12 points)** Please use a vector add example to explain what multiple lanes is in a vector architecture.

**Q4. (8 points)** Please briefly explain what grid and thread block are in the GPU architecture.

**Q5.** **(15 points)** In a CUDA program doing vector addition, let’s assume that we would like each thread to calculate one output element of the addition. How would you map the data index to thread id and/or block id? Which expression provides the best mapping below? Please explain and justify your choice in your own words.

a) i = blockIdx.x + threadIdx.x;

b) i = threadIdx.x + threadIdx.y;

c) i = blockIdx.x \* threadIdx.x;

d) i = blockIdx.x \* blockDim.x + threadIdx.x;

**Q6. (20 points)** In the following loop, find all the true dependences, output dependences, and anti-dependences. Eliminate the output dependences and anti-dependences by renaming.

for (i=0; i<100; i++) {

A[i] = A[i] \* B[i]; /\* S1 \*/

B[i] = A[i] + c; /\* S2 \*/

A[i] = C[i] \* c; /\* S3 \*/

C[i] = D[i] \* A[i]; /\* S4 \*/

}

**Q7. (15 points)** Consider the following loop:

for (i=0; i < 100; i++) {  
 A[i] = A[i] + B[i]; /\* S1 \*/

B[i+1] = C[i] + D[i]; /\* S2 \*/

}

Are there dependences between S1 and S2? Is this loop parallel? If not, show how to make it parallel.

**Q8.** **(12 points)** Assume a hypothetical GPU with the following characteristics:

* Clock rate 1.5 GHz
* Contains 16 SIMD processors, each containing 16 single-precision floating-point units
* Has 100 GB/s off-chip memory bandwidth

Without considering memory bandwidth, what is the peak single-precision floating-point throughput for this GPU in GLFOP/s, assuming that all memory latencies can be hidden?

Assuming each single precision operation requires four-byte two operands and outputs one four-byte result, is this throughput sustainable given the memory band-width limitation?

THE END.