Build Instructions

To build the code run cmd ‘make’

Note:

- makefile uses the nvcc compiler.
- makefile uses ‘-O’ compiler flag compiling the file.
- The generated output file is called ‘*lab5_cuda’.*
- Make sure you load the cuda environment (module load cuda) before building.

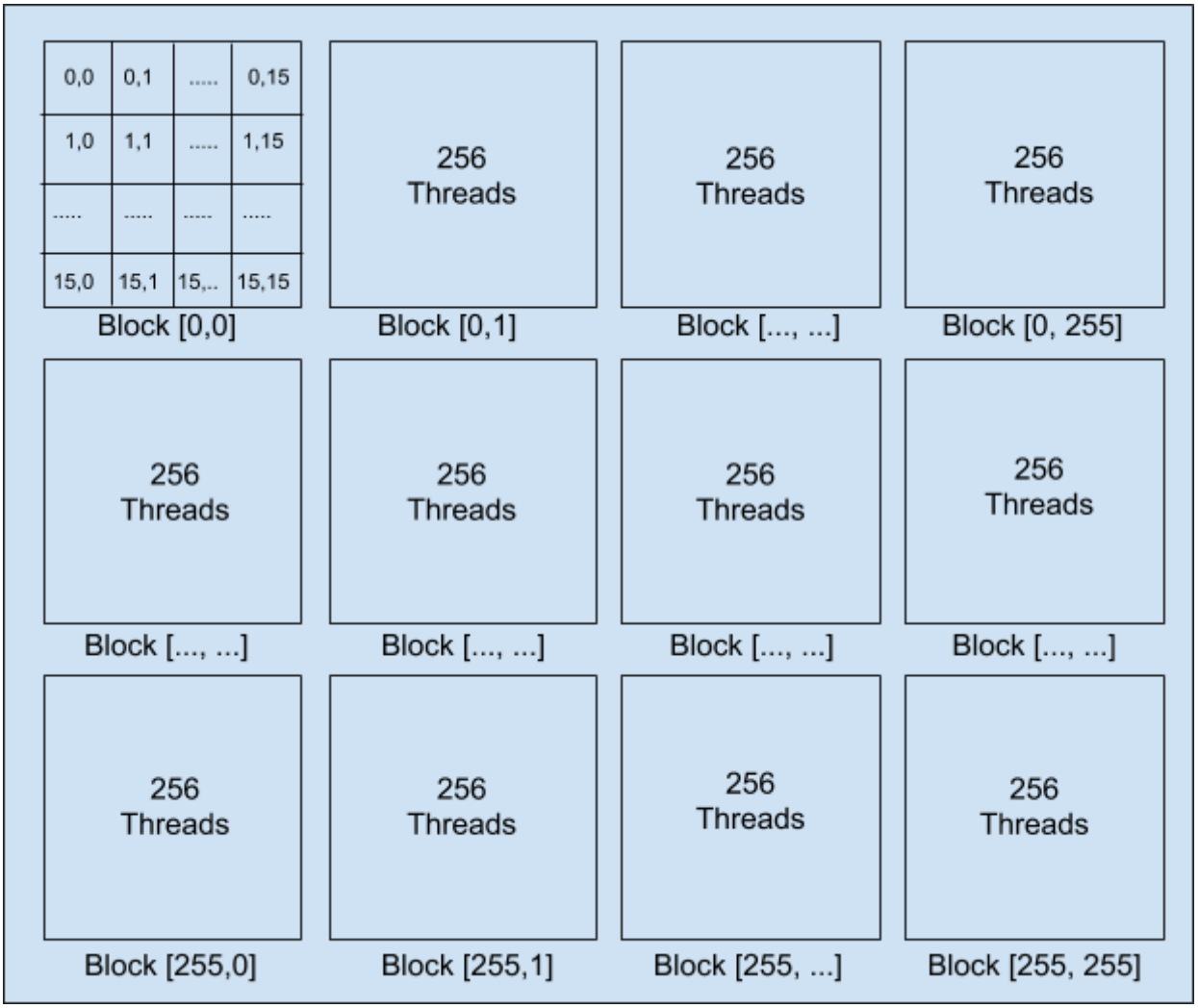
Execution Instructions

- Reserve 1 gpu node 28 ppn using qsub command (*-l nodes=1:ppn=28:gpus=1*) on owens.
- Execute: *“time ./lab5_cuda”*

Implementation

In this lab, we use CUDA to perform a matrix operation. The input matrix is initialized on the host and then sent to the device where it is processed to generate the desired output. Finally, once the entire matrix is processed on the device, its result is sent back to the host.

I am using a block size of (256 X 256) and a grid size of (16 X 16) and 256 threads per block, to process my input matrix with the dimension of (4096 X 4096).



Performance

The computation structure does (4096 * 2 floating point) operations for every cell in the matrix.

- **Operation 1:** float temp = inputMatrix[k*n + i] * inputMatrix[k*n + j]
- **Operation 2:** result[i*n + j] += temp

So total number of floating point operation is (4096*4096*4096*2) = 137438953472.

	Lab 5 (CUDA)					
	CUDA Parallel			**Serial**		
	Real	**User**	**GFlops/sec**	**Real**	**User**	**GFlops/sec**
Run 1	00:01.105	00:00.668	124.37914	27:52.976	27:52.472	0.00141
Run 2	00:01.086	00:00.649	126.55520	28:06.995	28:06.615	0.00136
Run 3	00:01.029	00:00.645	133.56555	23:43.449	23:41.804	0.00165
Run 4	00:01.033	00:00.624	133.04835	24:25.045	24:24.825	0.00159
Run 5	00:01.050	00:00.650	130.89424	24:10.372	24:09.753	0.00159
Average	00:01.061	00:00.647	129.68850	25:39.767	25:39.094	0.00152

NOTE: Real/User time is in mm:ss.ms format.