**Lab Exercise 6.3 – Print Additional Identifiers Like blockDim.x or gridDim.x in a CUDA Program**

**Objective:**  
To modify a CUDA kernel to print additional thread and block identifiers such as blockDim.x (the number of threads per block) and gridDim.x (the number of blocks in the grid), which are useful for understanding how the CUDA execution model works.

**Sample CUDA Program**

**Filename:** block\_grid\_identifiers.cu

#include <iostream>

// CUDA kernel

\_\_global\_\_ void identify\_thread\_and\_block() {

int global\_id = blockIdx.x \* blockDim.x + threadIdx.x;

// Print block, thread, and additional identifiers

printf("Block %d, Thread %d, Global ID: %d, BlockDim.x: %d, GridDim.x: %d\n",

blockIdx.x, threadIdx.x, global\_id, blockDim.x, gridDim.x);

}

int main() {

std::cout << "Launching CUDA kernel with additional identifiers...\n";

int numBlocks = 4;

int threadsPerBlock = 8;

identify\_thread\_and\_block<<<numBlocks, threadsPerBlock>>>();

cudaDeviceSynchronize();

std::cout << "Execution complete.\n";

return 0;

}

**Explanation:**

* blockIdx.x: The index of the block in the grid.
* threadIdx.x: The index of the thread within its block.
* blockDim.x: The number of threads per block.
* gridDim.x: The number of blocks in the grid.
* global\_id: The unique thread ID across the entire grid.

The kernel prints the block and thread identifiers along with the additional information about the block and grid dimensions.

**How to Compile and Run:**

nvcc block\_grid\_identifiers.cu -o block\_grid\_identifiers

./block\_grid\_identifiers

**Expected Output (shortened):**

Launching CUDA kernel with additional identifiers...

Block 0, Thread 0, Global ID: 0, BlockDim.x: 8, GridDim.x: 4

Block 0, Thread 1, Global ID: 1, BlockDim.x: 8, GridDim.x: 4

...

Block 3, Thread 7, Global ID: 31, BlockDim.x: 8, GridDim.x: 4

Execution complete.