**CUDA Programming Model: Threads, Blocks, and Grids**

In CUDA, programs are executed in parallel by **many threads** organized in a hierarchy of:

1. **Threads**
2. **Blocks**
3. **Grids**

**1. Thread**

* A single execution instance of a kernel.
* Each thread has a unique ID: threadIdx
* Threads are grouped inside **blocks**

**2. Block**

* A group of threads.
* Each block has its own ID: blockIdx
* Threads in a block can share data using **shared memory**
* Maximum threads per block typically: **1024**

**3. Grid**

* A group of blocks.
* The entire grid runs the kernel.
* Each grid has an ID: gridDim
* Block IDs are accessed via blockIdx

**Coordinate Variables**

| **Variable** | **Description** |
| --- | --- |
| threadIdx.x | Thread's index within the block |
| blockIdx.x | Block's index within the grid |
| blockDim.x | Number of threads per block |
| gridDim.x | Number of blocks in the grid |

**Execution Example**

\_\_global\_\_ void kernel() {

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

printf("Thread global ID: %d\n", global\_id);

}

int main() {

kernel<<<2, 4>>>(); // 2 blocks, each with 4 threads

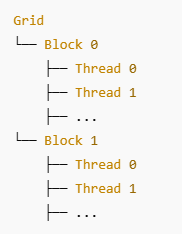
cudaDeviceSynchronize();

return 0;

}

**Output:** This will launch **8 threads** in total and print global thread IDs 0 to 7.

**Visualization**



**Summary**

| **Component** | **Represents** | **Access with** |
| --- | --- | --- |
| Thread | Smallest unit | threadIdx |
| Block | Group of threads | blockIdx |
| Grid | Group of blocks | gridDim |