**Basic Structure of a CUDA Program**

A CUDA program consists of two types of code:

* **Host code** – Runs on the CPU
* **Device code (kernel)** – Runs on the GPU

**General Structure**

#include <stdio.h>

// 1. Device (GPU) code – Kernel function

\_\_global\_\_ void kernelFunction() {

// Code executed by each GPU thread

}

// 2. Host (CPU) code – Main function

int main() {

// a. Allocate memory on device (if needed)

// b. Transfer data from host to device (optional)

// c. Launch kernel on GPU

kernelFunction<<<number\_of\_blocks, threads\_per\_block>>>();

// d. Synchronize to ensure GPU work is done

cudaDeviceSynchronize();

// e. Transfer results back to host (optional)

// f. Free device memory (if allocated)

return 0;

}

**Explanation of Components**

| **Part** | **Description** |
| --- | --- |
| \_\_global\_\_ | Declares a kernel function that runs on the GPU |
| kernelFunction<<<...>>>(); | Launches the kernel with specified blocks and threads |
| cudaDeviceSynchronize() | Makes the CPU wait until the GPU finishes execution |
| Memory functions (cudaMalloc, cudaMemcpy, cudaFree) | Used for managing memory between host and device |

**Example Launch**

myKernel<<<2, 4>>>();

* Launches 2 blocks, each with 4 threads (total 8 threads)