**CUDA Cores in Popular NVIDIA GPUs**

**Table: CUDA Cores in Popular NVIDIA GPUs**

| **GPU Model** | **CUDA Cores** | **Architecture** |
| --- | --- | --- |
| **NVIDIA Tesla K80** | 4992 | Kepler |
| **NVIDIA Tesla V100** | 5120 | Volta |
| **NVIDIA Tesla T4** | 2560 | Turing |
| **NVIDIA GTX 1080 Ti** | 3584 | Pascal |
| **NVIDIA RTX 2080 Ti** | 4352 | Turing |
| **NVIDIA RTX 3060** | 3584 | Ampere |
| **NVIDIA RTX 3070** | 5888 | Ampere |
| **NVIDIA RTX 3090** | 10496 | Ampere |
| **NVIDIA RTX 4090** | 16384 | Ada Lovelace |

**CUDA Cores vs. Threads**:

* **1 thread** = 1 **CUDA core** (usually).
* If your kernel (program) has **1024 threads in a block**, those **1024 threads** will be distributed among **CUDA cores**.
* If there are **more threads than cores**, the threads will be scheduled and processed **in batches**.

dim3 blockDim(8, 8, 16); // 8 \* 8 \* 16 = 1024 threads

You **cannot exceed 1024** total threads in one block.  
For example:

dim3 blockDim(40, 40); // 40 \* 40 = 1600 threads

**Formula**

When using dim3(x, y, z) for blocks:

Total threads per block = x \* y \* z ≤ 1024

**Summary Table**

| **Component** | **Limit** |
| --- | --- |
| Max Threads Per Block | 1024 |
| Max Threads Per 1D Block dimension (x) | 1024 |
| Max Threads Per 2D Block (x \* y) | 1024 |
| Max Threads Per 3D Block (x \* y \* z) | 1024 |

| **blockDim.x** | **blockDim.y** | **blockDim.z** | **Total Threads** | **Allowed?** |
| --- | --- | --- | --- | --- |
| 1024 | 1 | 1 | 1024 | ✅ |
| 32 | 32 | 1 | 1024 | ✅ |
| 16 | 16 | 4 | 1024 | ✅ |
| 1024 | 2 | 1 | 2048 | ❌ (Too much) |