DR. D. Y. PATIL
EDUCATIONAL FEDERATIONS

Practical No.4.

Aim:- Write a CUDA program For:
1. Addition of two large vectors.
2. Matrix multiplication using CUDA C.

Objective:- The objective of this task is to write a CUDA program to perform vector addition & matrix multiplication using CUDA C.

Theory:-

CUDA programming model:-

The CUDA programming model is a parallel computing platform & programming model designed to use NVIDIA GPUs to accelerate computation. It allows programmers to write parallel code that runs on the GPU, which is designed to handle massive parallelism & compute-intensive tasks.

CUDA Kernel:-

A CUDA kernel is a function that runs on the GPU & is executed by many threads in parallel. It is written in CUDA C & is designed to perform a specific task on the GPU, such as vector addition or matrix multiplication.

DR. D. Y. PATIL
EDUCATIONAL FEDERATIONS

Vector addition:

Vector addition is a simple operation that adds two vectors element-wise to produce a new vector. In CUDA, the operation ~~that adds two vectors~~ can be parallelized by dividing the vectors into smaller chuncks & assigning each chunk to a different thread. The threads can then perform the addition operation in parallel, resulting in faster computation times.

Matrix Multiplication:

Matrix multiplication is a computationally intensive operation that multiplies two matrices together to produce a new matrix. In CUDA, the operation can be parallelized by dividing the matrices into smaller chuncks & assigning each chunk to a different thread block. The threads within each block can then perform the multiplication operation in parallel, resulting in faster computation times.

The syntax used in CUDA C code for vector addition & matrix multiplication includes:

Kernel definition: The kernel function is defined using the "global" keyword, which indicates that the function runs on the GPU.

DR. D. Y. PATIL
EDUCATIONAL FEDERATIONS

Thread Indexing: ThreadIdx & blockDim variables are used to calculate the index of each thread & the total number of threads in the block.

~~Memory Al~~
Memory Allocation: The "cudaMalloc" function is used to allocate memory on the GPU, & the "CudaMemcpy" function is used to transfer data between the CPU & GPU.

Execution configuration: The "<<<....>>>" syntax is used to specify the number of thread blocks & threads per block to be used for the kernel execution.

Reduction: For matrix multiplication, a reduction operation is used to combine the results of the thread blocks.

Conclusion: In conclusion, writing a CUDA program for vector addition & matrix multiplication using CUDA C can significantly accelerate computation by leveraging the massive parallelism of NVIDIA GPUs.