Experiment No: 4 Group A

Aim: To write a CUDA program for, find.

1. Addition of two large vectors
2. Matrix Multiplication using CUDA C

Objective: To study and implement the CUDA program using vectors.

Pre-requisites:
64-bit Open source Linux or its derivative
Programming Languages: C/C++, CUDA

Theory:

CUDA:

CUDA programming is especially well-suited to address problems that can be expressed as data-parallel computations. Any applications that process large data sets can use a data-parallel model to speed up the computations. Data-parallel processing maps data elements to parallel threads.

The first step in designing a data parallel program is to partition data across threads, with each thread working on a portion of the data. The first step in designing a data parallel program is to partition data across threads, with each thread working on a portion of the data.

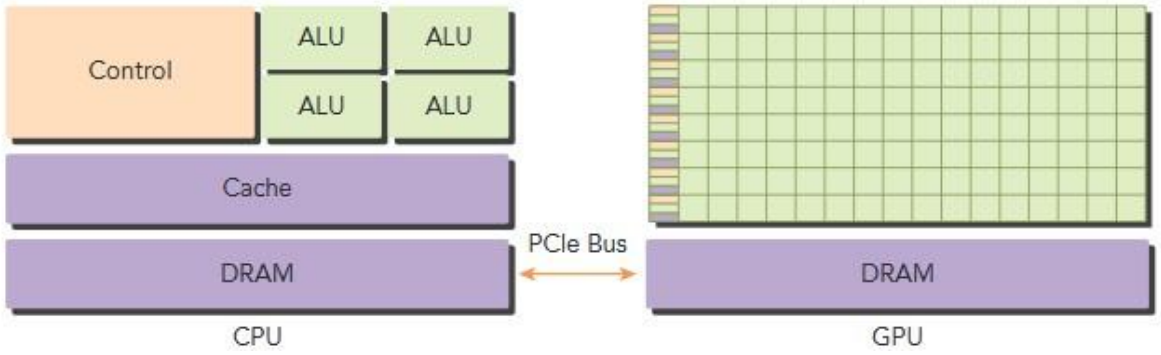
CUDA Architecture:

A heterogeneous application consists of two parts:

- Host code
- Device code

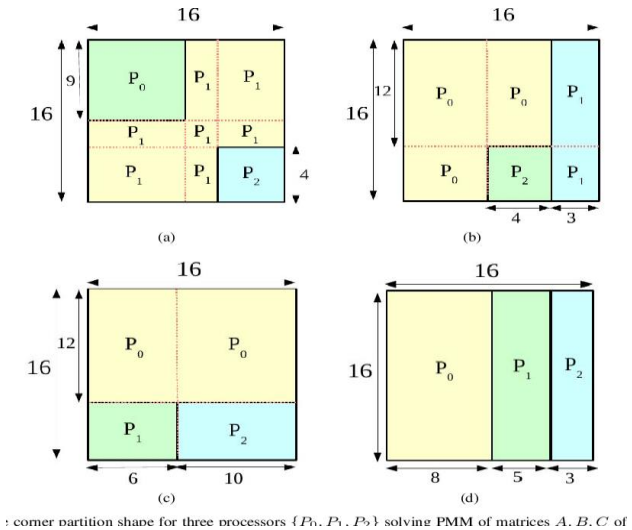
Host code runs on CPUs and device code runs on GPUs. An application executing on a heterogeneous platform is typically initialized by the CPU. The CPU code is responsible for managing the environment, code, and data for the device before loading compute-intensive tasks on the device. With computational intensive applications, program sections often exhibit a rich amount of data parallelism. GPUs are used to accelerate the execution of this portion of data parallelism. When a hardware component that is physically separate from the CPU is used to accelerate computationally intensive sections of an application, it is referred to as a hardware

accelerator. GPUs are arguably the most common example of a hardware accelerator. GPUs must operate in conjunction with a CPU-based host through a PCI-Express bus, as shown in Figure.



Matrix-Matrix Multiplication

- Consider two *n* x *n* matrices *A* and *B* partitioned into *p* blocks $A_{i,j}$ and $B_{i,j}$(0≤ *i*<j) of size each. $(n/\sqrt{p}) \times (n/\sqrt{p})$
- Process $P_{i,j}$ initially stores $A_{i,j}$ and $B_{i,j}$ and computes block $C_{i,j}$ of the result matrix.
- Computing submatrix $C_{i,j}$ requires all submatrice $A_{i,k}$ and $B_{k,j}$ for $0 \leq k < \sqrt{p}$
- All-to-all broadcast blocks of *A* along rows and *B* along columns.
- Perform local submatrix multiplication



: corner partition shape for three processors $\{P_0, P_1, P_2\}$ solving PMM of matrices A, B, C of

Conclusion: Thus, we have successfully implemented the Addition of two large vectors and Matrix Multiplication using CUDAC Programming.