



VIT[®]
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)

FALL – SEMESTER
Course Code: MCSE503L
Course-Title: – Computer Architecture and Organization
DIGITAL ASSIGNMENT - II

Name: Nidhi Singh
Reg. No:22MAI0015

Faculty : NARAYANAMOORTHY M - SCOPE

Slot- C1+TC1

(1) Write short notes on the following:

1.1. Super computing and high performance computing

Super computing:-

- Supercomputing refers to the processing of insanely complex or data-laden tasks using the combined resources of multiple computers working in parallel (which is a supercomputer). The term supercomputing is associated with supercomputers meaning computers systems which are powerful to handle high performance computing as opposed to general purpose computer.
- Supercomputers, in fact, are the fastest computers till date and hence the backbone of computational sciences. They are capable enough to process and generate huge amounts of data with unparalleled speed and efficiency.
- The hardware structure or architecture of supercomputers determines the efficiency of supercomputing systems. Today, supercomputers are used for highly intensive calculation tasks for projects ranging from quantum physics, weather forecasting, physical simulations, and molecular modeling.
- They can be used for simulations of airplanes in wind tunnels, splitting electrons, detonation of nuclear weapons, oil and gas exploration, and more.

High performance computing :-

- High-performance computing (HPC) is the use of supercomputers and parallel computing techniques to solve complex computational problems. Parallel computing is when a number of compute elements work in collaboration to solve a problem.
- All modern supercomputer architectures rely heavily on parallelism. In general, HPC is the practice of aggregating computing power in a way that delivers superior performance as opposed to a typical desktop computer. For years, high performance computing has demonstrated the ability to model and predict a wide range of physical properties and phenomena accurately.
- HPC architecture is influenced by the lowest-level technologies and circuit design, and how they can be most effectively employed in supercomputers. HPC is one of the applications of supercomputers that would allow them to solve computational problems that are too large for typical desktop computers to handle.

These HPC applications are driving continuous innovation in:

- **Healthcare, genomics and life sciences.** The first attempt to sequence a human genome took *13 years*; today, HPC systems can do the job in less than a day. Other HPC applications in healthcare and life sciences include drug discovery and design, rapid cancer diagnosis, and molecular modeling.
- **Financial services.** In addition to automated trading and fraud detection (noted above), HPC powers applications in Monte Carlo simulation and other risk analysis methods.
- **Government and defense.** Two growing HPC use cases in this area are weather forecasting and climate modeling, both of which involve processing vast amounts of historical meteorological data and millions of daily changes in climate-related data points. Other government and defense applications include energy research and intelligence work.
- **Energy.** In some cases, overlapping with government and defense, energy-related HPC applications include seismic data processing, reservoir simulation and modeling, geospatial analytics, wind simulation and terrain mapping.

Difference between Supercomputing and High-Performance Computing:-

| Supercomputing | High-Performance Computing |
|---|--|
| Supercomputing is the processing of extremely complex or data intensive problems by using supercomputers. | High-performance computing (HPC) is the recent version of what used to be called supercomputing. |
| Efficiently solves complex data intensive problems by concentrating the combined resources of multiple computers working in parallel. | HPC technology incorporates both administration and parallel computational techniques to develop parallel processing algorithms and systems. |
| A supercomputer is a really fast computer which is powerful enough to perform at or near the currently highest operational rate for computers. | HPC is the practice of aggregating computing power in a way that delivers superior performance as opposed to a typical desktop computer. |
| Applications include weather forecasting, aerospace engineering, automobile crash & safety modeling, quantum physics, physical simulations, oil and gas exploration, etc. | Applications include structural analysis, computational fluid dynamics, oil exploration, atmospheric sciences, and defense applications. |

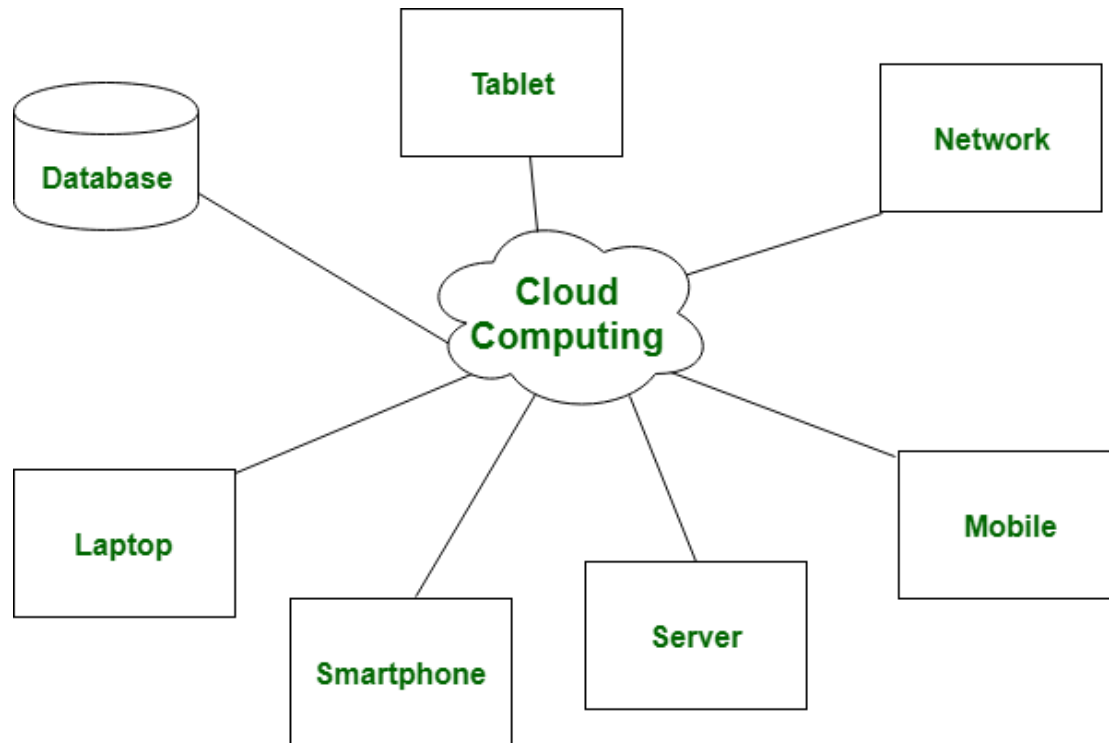
1.2. Cloud computing and Grid computing

Cloud Computing :-

- In its most basic form, a cloud is nothing more than a network used for downloading or uploading data. Cloud computing is defined as serverless computing, which entails the user extracting information from the internet through web-related searches without really having to meddle with the management of data. It is made up of a sizable IT infrastructure that allows for quick and accurate information retrieval from the internet.
- Cloud computing is essentially the collection of remotely managed, configured, and accessed hardware and software components. Although it is a more recent development in technology, it is extremely precise and satisfies all needs. Because it

gives platform freedom and does not require local installation, it is accountable for making the application more collaborative.

- Cloud Computing is a Client-server computing architecture. In cloud computing, resources are used in centralized pattern and cloud computing is a high accessible service. It is a pay and use business means, in cloud computing, the users pay for the use.



Cloud Computing Architecture: Cloud computing architecture refers to the components and sub-components required for cloud computing. These components typically refer to:

1. Front end(fat client, thin client)
2. Back-end platforms(servers, storage)
3. Cloud-based delivery and a network(Internet, Intranet, Intercloud).

Types of cloud computing:

Public Cloud

Public cloud is a type of cloud computing in which a cloud service provider makes computing resources—anything from SaaS applications, to individual virtual machines (VMs), to bare metal computing hardware, to complete enterprise-grade infrastructures and development platforms—available to users over the public internet. These resources might be accessible for free, or access might be sold according to subscription-based or pay-per-usage pricing models.

Private cloud

Private cloud is a cloud environment in which all cloud infrastructure and computing resources are dedicated to, and accessible by, one customer only. Private cloud combines many of the benefits of cloud computing—including elasticity, scalability, and ease of service delivery—with the access control, security, and resource customization of on-premises infrastructure.

Hybrid cloud

Hybrid cloud is just what it sounds like—a combination of public and private cloud environments. Specifically, and ideally, a hybrid cloud connects an organization's private cloud services and public clouds into a single, flexible infrastructure for running the organization's applications and workloads.

1. **Amazon Web Services (AWS):** One of the most successful cloud-based businesses is Amazon Web Services (AWS), which is an Infrastructure as a Service (IaaS) offering that pays rent for virtual computers on Amazon's infrastructure.
2. **Microsoft Azure Platform:** Microsoft is creating the Azure platform which enables the .NET Framework Application to run over the internet as an alternative platform for Microsoft developers. This is the classic Platform as a Service (PaaS).
3. **Google:** Google has built a worldwide network of datacenters to service its search engine. From this service, Google has captured the world's advertising revenue. By using that revenue, Google offers free software to users based on infrastructure. This is called Software as a Service (SaaS).

Grid Computing :-

- Grid computing is ideal for sorting or spacing out vast volumes of data that have either been improperly packed together or need to be. It operates on the idea of many hundreds of connected and networked computers that handle the task like a supercomputer and operate like stacked software.
- It basically eliminates the otherwise tiresome chore of using individual software and front end - back end links for carrying out a particular task by acting as a lender of effective software machines that one may use for a set period of time, even paying if necessary.
- A top-notch method for those without technical backgrounds to do their work quickly and accurately is provided by grid computing. Since several computers are involved in the advancement, the run time of each task and the time it takes to produce the desired result are greatly decreased to save resources.

- In order to accomplish the main goal, grid computing essentially combines and employs the efficiency of many hands. It is based on a grid computer with processors that utilizes conjugated networks.
- Grid Computing is a Distributed computing architecture. In grid computing, resources are used in collaborative pattern, and also in grid computing, the users do not pay for use.

Types:-

Grid computing is generally classified as follows.

1). Computational grid

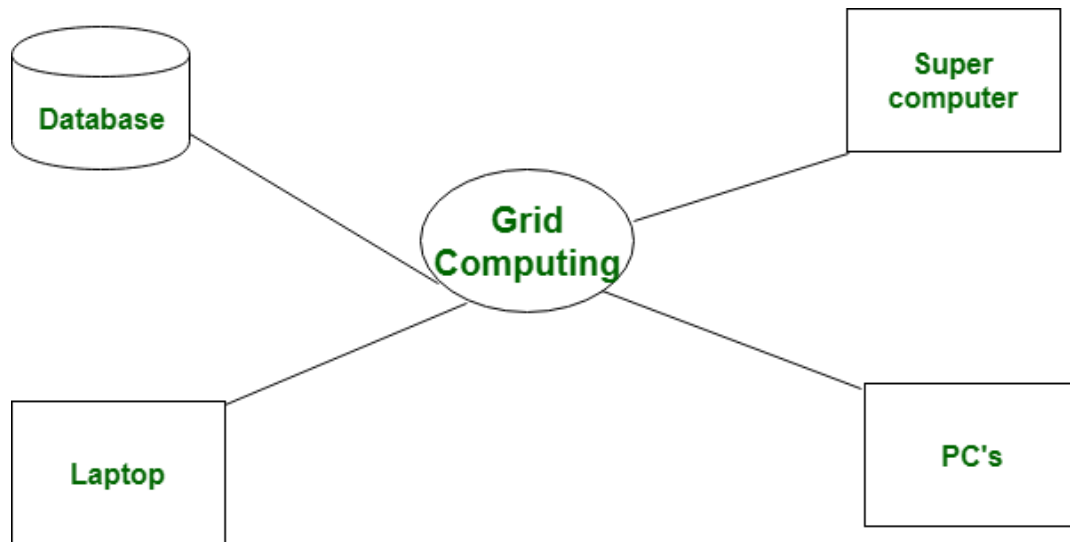
A computational grid consists of high-performance computers. It allows researchers to use the combined computing power of the computers. Researchers use computational grid computing to perform resource-intensive tasks, such as mathematical simulations.

2). Scavenging grid

While similar to computational grids, CPU scavenging grids have many regular computers. The term *scavenging* describes the process of searching for available computing resources in a network of regular computers. While other network users access the computers for non-grid-related tasks, the grid software uses these nodes when they are free. The scavenging grid is also known as CPU scavenging or cycle scavenging.

3). Data grid

A data grid is a grid computing network that connects to multiple computers to provide large data storage capacity. You can access the stored data as if on your local machine without having to worry about the physical location of your data on the grid.



Difference between Cloud Computing and Grid Computing:-

| Key | Cloud Computing | Grid Computing |
|---------------|--|--|
| Architecture | Cloud Computing follows a client-server computing architecture. | Grid Computing follows a distributed computing architecture. |
| Resource | In Cloud Computing, resources are centrally managed. | In Grid Computing, resources are managed on a collaboration pattern. |
| Flexibility | Cloud Computing is more flexible than Grid Computing. | Grid Computing is less flexible. |
| Payment | Users pay for using the cloud computing resources. They need not to set up anything. They use the platform as a service. | Grid computing needs to be set up first. Users need not pay anything once the set-up is done. |
| Accessibility | Cloud Computing is a highly accessible service. It can be accessed using conventional web protocols. | Grid Computing is low on accessibility as compared to cloud computing. It can be accessed using grid middleware. |
| Scalability | When compared to grid computing, it is extremely scalable. | Grid computing, on the other hand, is not as scalable as cloud computing. |

(2) Write short notes on the following:

2.1. Power efficient processors

- The best processors are the most essential part of any PC. And, given that they are the brains of the entire operation, it's very important that you choose the best performing, most power-efficient one you can afford.
- In addition to performing the vital operations that actually make the machine run, the best Intel processors and the best AMD processors even come with integrated graphics processors, which offer massive savings since the best graphics cards can cost hundreds of dollars. If you do not need a dedicated GPU, that is.
- With the release of Intel Raptor Lake and the AMD Ryzen 7000-series processors, the very best processors from the previous generations are getting some healthy price cuts, which means more savings for you. Although, if you are looking for high-end builds, you'll find some excellent high-end CPUs are going to provide outstanding levels of performance when it comes to content creation or playing the best PC games at high frame rates, making them worth the splurge.
- We've tested the best CPUs on the market, putting them through their paces to see how they perform in the real world. And, we're here to help you make the right choice for your needs and budget. Whether you want something to power your family desktop PC or are looking for a powerful chip for content creation, we have a terrific option for you below.

1. Intel Core i9-13900K (The best processor overall)

1. SPECIFICATIONS

- i. **Cores:** 16
- ii. **Threads:** 32
- iii. **Base clock:** 4.5GHz
- iv. **Boost clock:** 5.7GHz
- v. **L3 cache:** 80MB
- vi. **TDP:** 170W

2. REASONS TO BUY

- i. Dominates both single- and multi-core workloads
- ii. Same pricing as the Core i9-12900K
- iii. Top-tier gaming performance
- iv. Surprisingly low power consumption on average

3. REASONS TO AVOID

- i. Runs hot and hungry at maximum load
- ii. Lacks AMD's 3D V-cache for gaming
- iii. Basically overkill for most users

2. AMD Ryzen 5 7600X(The perfect mix of performance and value)

1. SPECIFICATIONS

- i. **Cores:** 6
- ii. **Threads:** 12
- iii. **Base clock:** 4.7GHz
- iv. **Boost clock:** 5.3GHz
- v. **L3 cache:** 38MB
- vi. **TDP:** 105W

2. REASONS TO BUY

- i. Outstanding performance for the price
- ii. Very energy efficient
- iii. DDR5 and PCIe 5.0 support

3. REASONS TO AVOID

- i. Requires AM5 motherboard
- ii. Multicore performance lags somewhat

3. AMD Ryzen 9 7950X(The best AMD processor available right now)

1. SPECIFICATIONS

- i. **Cores:** 16
- ii. **Threads:** 32
- iii. **Base clock:** 4.5GHz
- iv. **Boost clock:** 5.7GHz
- v. **L3 cache:** 80MB
- vi. **TDP:** 170W

2. REASONS TO BUY

- i. Best-in-class performance
- ii. Very energy efficient
- iii. DDR5 and PCIe 5.0 support

3. REASONS TO AVOID

- i. Requires AM5 motherboard
- ii. Expensive
- iii. Professional content creators might want something better

4. Intel Core i5-12600K(The best Intel chip by value)

1. SPECIFICATIONS

- i. **Cores:** 10
- ii. **Threads:** 16
- iii. **Base clock:** 3.7GHz
- iv. **Boost clock:** 4.9GHz
- v. **L3 cache:** 20MB
- vi. **TDP:** 125W

2. REASONS TO BUY

- i. Excellent performance
- ii. Affordable

3. REASONS TO AVOID

- i. Will need all new hardware

5. AMD Ryzen 7 5800X3D (The best processor for gaming on AM4 motherboards)

1. SPECIFICATIONS

- i. **Cores:** 8
- ii. **Threads:** 16
- iii. **Base clock:** 3.7GHz
- iv. **Boost clock:** 4.7GHz
- v. **L3 cache:** 100MB
- vi. **TDP:** 105W

2. REASONS TO BUY

- i. Outstanding gaming performance
- ii. Uses current-gen AM4 Socket
- iii. New 3D V-Cache technology

3. REASONS TO AVOID

- i. No DDR5 or PCIe 5.0
- ii. Non-gaming performance lags a little

6. AMD Ryzen 7 7700X(High-performance at an accessible price)

1. SPECIFICATIONS

- i. **Cores:** 8
- ii. **Threads:** 16
- iii. **Base clock:** 4.5GHz
- iv. **Boost clock:** 5.4GHz
- v. **L3 cache:** 32MB
- vi. **TDP:** 105W

2. REASONS TO BUY

- i. Phenomenal gaming performance
- ii. Accessible price
- iii. Energy efficient

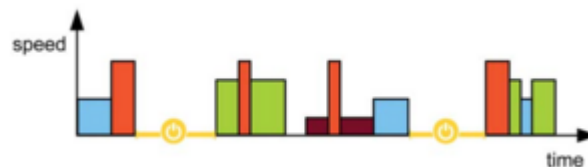
3. REASONS TO AVOID

- i. Needs new hardware to use
- ii. Ryzen 5 7600X offers better value

2.2. Energy efficient algorithms

- The efficiency of an algorithm is typically determined by the quality of the solution it returns, along with the amount of resources that it requires to return such a solution. Traditional analysis of algorithms mostly involves evaluating the performance of an algorithm in terms of the resources, running time, and storage, i.e., the number of steps it requires in order to produce the desired result, or the space it takes up in the storage device during execution.
- However, as energy is a limited and expensive resource, it is of critical importance to analyze algorithms (or the solutions returned by the algorithms) for this resource as well. To quote former Google CEO, Eric Schmidt: “What matters most to computer designers at Google is not speed, but power, low power, because data centers can consume as much electricity as a city”.

- To address the aim of power saving, there has been a significant upsurge in the study of energy-efficient algorithms in recent years. The most obvious type of algorithmic problem that considers energy as a resource is the question of how to manage power directly at the processor level.
- As an example, most modern portable computers are equipped with a processor that is speed scalable and is supplied with a sleep state. Suppose we want to process a number of programs on such a computer, where each program has to be processed within a specific time frame and requires a particular amount of CPU-cycles. Then, the operating system has to schedule these programs on the processor, that is, it has to decide when to transition the processor to the sleep state, and furthermore, at which times and at what speed to process each program.
- These decisions can have a huge impact on the energy consumption of the processor and, in turn, on the battery life of the computer. It has been observed that it is sometimes beneficial to operate the processor faster than necessary in order to reside to the sleep state for longer periods of time, a technique commonly referred to as race to idle.
- However, operating the processor at its highest possible speed whenever there is work to be done, and transitioning it to the sleep state otherwise (as is commonly done in practice) is also suboptimal.



A schedule

- Together with Chien-Chung Huang, we have developed an algorithm for the aforementioned setting. The schedules that our algorithm produces have an energy consumption that is guaranteed to be arbitrarily close to the optimal one.
- Additionally, the algorithm can compute such schedules in a running time that is polynomial in the input size, which is considered efficient. The main idea of our algorithm is to divide each program into smaller, unsplittable subprograms and identify a specific set of orderings in which these subprograms could potentially be processed.
- We prove that one of these orderings produces a sufficiently good solution, and that we can efficiently find the best ordering in the set by using a well-known technique called dynamic programming. Our result is the best possible, given universally accepted complexity-theoretic assumptions.
- Cloud computing is an internet-based computing which provides metering-based services to consumers. It means accessing data from a centralized pool of compute resources that can be ordered and consumed on demand. It also provides computing resources through virtualization over internet.

- Data center is the most prominent in cloud computing which contains collection of servers on which Business information is stored and applications run. Data center which includes servers, cables, air conditioner, network etc. consumes more power and releases huge amount of Carbon-di-oxide (CO₂) to the environment. One of the most important challenges faced in cloud computing is the optimization of Energy Utilization. Hence the concept of green cloud computing came into existence.

There are multiple techniques and algorithms used to minimize the energy consumption in cloud.

Techniques include:

1. Dynamic Voltage and Frequency Scaling (DVFS)
2. Virtual Machine (VM)
3. Migration and VM Consolidation

Algorithms are:

1. Maximum Bin Packing
2. Power Expand Min-Max and Minimization Migrations
3. Highest Potential growth

The main expectation of cloud service consumer is to have a reliable service. To satisfy consumer's expectation several Data centres are established all over the world and each Data centre contains thousands of servers. Small amount of workload on server consumes 50% of the power supply. Cloud service providers ensure that reliable and load balancing services to the consumers around the world by keeping servers ON all the time. To satisfy this SLA provider has to supply power continuously to data centres leads to huge amount of energy utilization by the data centre and simultaneously increases the cost of investment.

The major challenge is utilization of energy efficiently and hence develops an eco-friendly cloud computing.

The idle servers and resources in data center wastes huge amount of energy. Energy also wasted when the server is overloaded. Few techniques such as load balancing, VM virtualization, VM migration, resource allocation and job scheduling etc. are used to solve the problem. It is also found that transporting data between data centers and home computers can consume even larger amounts of energy than storing it.

(3) Write a c program using OpenMP, MPI and CUDA to perform arithmetic and logical operations between two numbers?

OpenMP:-

```
#include<stdio.h>
#include<omp.h>
int main()
{
    double start_time, end_time, total_time;
    start_time = omp_get_wtime();
    int tid;
    int n1 = 40, n2 = 8;
    omp_set_num_threads(5);
#pragma omp parallel private(tid)
    {
        tid = omp_get_thread_num();
        if (tid==0)
        {
            printf("\n%d+%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 + n2, tid);
            printf("\n%d%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 | n2, tid);
        }

        else if (tid==1)
            printf("\n%d-%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 - n2, tid);

        else if (tid==2)
        {
            printf("\n%d*%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 * n2, tid);
            printf("\n%d&%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 & n2, tid);
        }

        else if (tid==3)
            printf("\n%d/%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 / n2, tid);

        else if (tid==4)
            printf("\n%d^%d=%d\tFROM THREAD NUMBER %d\n", n1, n2, n1 ^ n2, tid);

        end_time = omp_get_wtime();

        total_time = end_time - start_time;
        printf("\nTOTAL TIME BY THE THREAD %d : %f\n\n", tid, total_time);
    }
    return 0;
}
```

Output :-

```
40^8=32 FROM THREAD NUMBER 4
TOTAL TIME BY THE THREAD 4 : 0.000785

40/8=5 FROM THREAD NUMBER 3
TOTAL TIME BY THE THREAD 3 : 0.001102

40*8=320 FROM THREAD NUMBER 2
40&8=8 FROM THREAD NUMBER 2
TOTAL TIME BY THE THREAD 2 : 0.001474

40-8=32 FROM THREAD NUMBER 1
TOTAL TIME BY THE THREAD 1 : 0.001731

40+8=48 FROM THREAD NUMBER 0
40|8=40 FROM THREAD NUMBER 0
TOTAL TIME BY THE THREAD 0 : 0.001954
```

MPI program code

```
#include <stdio.h>
#include <mpi.h>
int main(int argc, char** argv) {
// Initialize MPI
MPI_Init(NULL, NULL);

// Get the rank of the current process
int rank;
```

```

MPI_Comm_rank(MPI_COMM_WORLD, &rank);

// Get the total number of processes
int size;
MPI_Comm_size(MPI_COMM_WORLD, &size);

// Set the two numbers to be used in the operations
int num1 = 80;
int num2 = 10;

// Perform arithmetic operations
int sum = num1 + num2;
int difference = num1 - num2;
int product = num1 * num2;
int quotient = num1 / num2;

// Perform logical operations
int and_result = num1 & num2;
int or_result = num1 | num2;
int xor_result = num1 ^ num2;

// Print the results
printf("Rank %d: Sum = %d\n", rank, sum);
printf("Rank %d: Difference = %d\n", rank, difference);
printf("Rank %d: Product = %d\n", rank, product);
printf("Rank %d: Quotient = %d\n", rank, quotient);
printf("Rank %d: AND result = %d\n", rank, and_result);
printf("Rank %d: OR result = %d\n", rank, or_result);
printf("Rank %d: XOR result = %d\n", rank, xor_result);

// Finalize MPI
MPI_Finalize();
}

```

Output :-

```
Rank 1: Sum = 90
Rank 1: Difference = 70
Rank 1: Product = 800
Rank 1: Quotient = 8
Rank 1: AND result = 0
Rank 1: OR result = 90
Rank 1: XOR result = 90
Rank 0: Sum = 90
Rank 0: Difference = 70
Rank 0: Product = 800
Rank 0: Quotient = 8
Rank 0: AND result = 0
Rank 0: OR result = 90
Rank 0: XOR result = 90
```

CUDA Program:

Addition:-

```
#include<stdio.h>
#include<cuda.h>
#include<cuda_runtime_api.h>

__global__ void add(int* a, int* b, int* c)
{
    *c = *a + *b;
}

int main(void)
{
    int a, b, c;
    int* d_a, *d_b, *d_c;
    int size = sizeof(int);

    cudaMalloc((void**)&d_a, size);
    cudaMalloc((void**)&d_b, size);
    cudaMalloc((void**)&d_c, size);

    a = 54;
    b = 18;
    cudaMemcpy(d_a, &a, size, cudaMemcpyHostToDevice);
    cudaMemcpy(d_b, &b, size, cudaMemcpyHostToDevice);

    add << <1, 1 >> > (d_a, d_b, d_c);

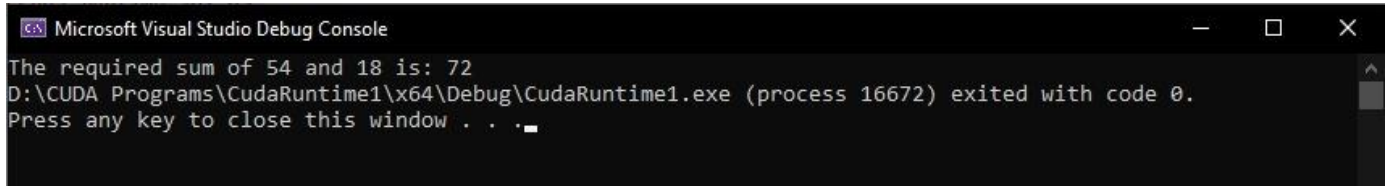
    cudaMemcpy(&c, d_c, size, cudaMemcpyDeviceToHost);
```



```
cudaFree(d_a); cudaFree(d_b); cudaFree(d_c);

printf("The required sum of %d and %d is: %d", a,b,c);
return 0;
}
```

Output :-

A screenshot of the Microsoft Visual Studio Debug Console window. The window title is "Microsoft Visual Studio Debug Console". The output text is: "The required sum of 54 and 18 is: 72", "D:\CUDA Programs\CudaRuntime1\x64\Debug\CudaRuntime1.exe (process 16672) exited with code 0.", and "Press any key to close this window . . .".

```
Microsoft Visual Studio Debug Console
The required sum of 54 and 18 is: 72
D:\CUDA Programs\CudaRuntime1\x64\Debug\CudaRuntime1.exe (process 16672) exited with code 0.
Press any key to close this window . . .
```

Subtraction:-

```
#include<stdio.h>
#include<cuda.h>
#include<cuda_runtime_api.h>

__global__ void diff(int* a, int* b, int* c)
{
    *c = *a - *b;
}

int main(void)
{
    int a, b, c;
    int* d_a, * d_b, * d_c;
    int size = sizeof(int);

    cudaMalloc((void**)&d_a, size);
    cudaMalloc((void**)&d_b, size);
    cudaMalloc((void**)&d_c, size);

    a = 54;
    b = 18;
    cudaMemcpy(d_a, &a, size, cudaMemcpyHostToDevice);
    cudaMemcpy(d_b, &b, size, cudaMemcpyHostToDevice);

    diff << <1, 1 >> > (d_a, d_b, d_c);
```

```

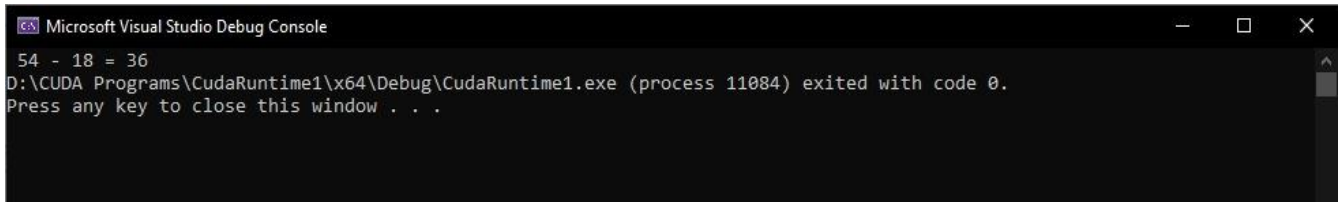
    cudaMemcpy(&c, d_c, size, cudaMemcpyDeviceToHost);

    cudaFree(d_a); cudaFree(d_b); cudaFree(d_c);

    printf("The required difference of %d and %d is: %d", a,b,c);
    return 0;
}

```

Output:



```

Microsoft Visual Studio Debug Console
54 - 18 = 36
D:\CUDA Programs\CudaRuntime1\x64\Debug\CudaRuntime1.exe (process 11084) exited with code 0.
Press any key to close this window . . .

```

Multiplication:-

```

#include<stdio.h>
#include<cuda.h>
#include<cuda_runtime_api.h>

__global__ void prod(int* a, int* b, int* c)
{
    *c = *a * *b;
}

int main(void)
{
    int a, b, c;
    int* d_a, * d_b, * d_c;
    int size = sizeof(int);

    cudaMalloc((void**)&d_a, size);
    cudaMalloc((void**)&d_b, size);
    cudaMalloc((void**)&d_c, size);

    a = 54;
    b = 18;
    cudaMemcpy(d_a, &a, size, cudaMemcpyHostToDevice);
    cudaMemcpy(d_b, &b, size, cudaMemcpyHostToDevice);

    prod << <1, 1 >> > (d_a, d_b, d_c);
}

```

```

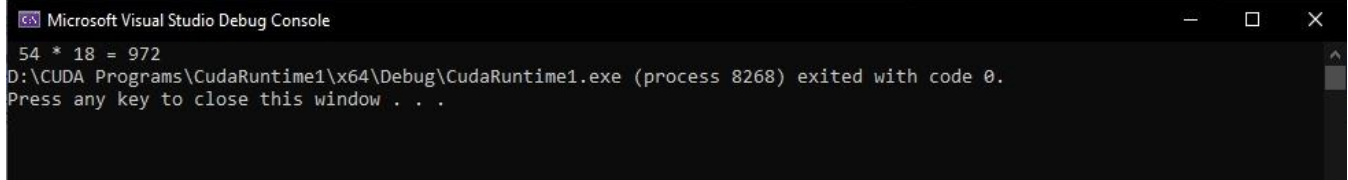
    cudaMemcpy(&c, d_c, size, cudaMemcpyDeviceToHost);

    cudaFree(d_a); cudaFree(d_b); cudaFree(d_c);

    printf("The required multiplication of %d and %d is: %d", a,b,c);
    return 0;
}

```

Output:-



```

Microsoft Visual Studio Debug Console
54 * 18 = 972
D:\CUDA Programs\CudaRuntime1\x64\Debug\CudaRuntime1.exe (process 8268) exited with code 0.
Press any key to close this window . . .

```

Division:

```

#include<stdio.h>
#include<cuda.h>
#include<cuda_runtime_api.h>

__global__ void div(int* a, int* b, int* c)
{
    *c = *a / *b;
}

int main(void)
{
    int a, b, c;
    int* d_a, * d_b, * d_c;
    int size = sizeof(int);

    cudaMalloc((void**)&d_a, size);
    cudaMalloc((void**)&d_b, size);
    cudaMalloc((void**)&d_c, size);

    a = 54;
    b = 18;
    cudaMemcpy(d_a, &a, size, cudaMemcpyHostToDevice);
    cudaMemcpy(d_b, &b, size, cudaMemcpyHostToDevice);

    div << <1, 1 >> > (d_a, d_b, d_c);
}

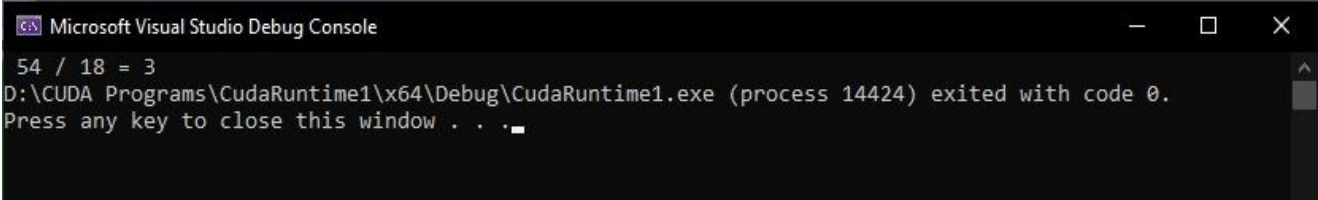
```

```
cudaMemcpy(&c, d_c, size, cudaMemcpyDeviceToHost);

cudaFree(d_a); cudaFree(d_b); cudaFree(d_c);

printf("The required division of %d and %d is: %d", a,b,c);
return 0;
}
```

Output:

A screenshot of the Microsoft Visual Studio Debug Console window. The window has a title bar with the text "Microsoft Visual Studio Debug Console" and standard minimize, maximize, and close buttons. The console output shows the result of a division: "54 / 18 = 3". Below this, it states "D:\CUDA Programs\CudaRuntime1\x64\Debug\CudaRuntime1.exe (process 14424) exited with code 0." and "Press any key to close this window . . .".

```
Microsoft Visual Studio Debug Console
54 / 18 = 3
D:\CUDA Programs\CudaRuntime1\x64\Debug\CudaRuntime1.exe (process 14424) exited with code 0.
Press any key to close this window . . .
```