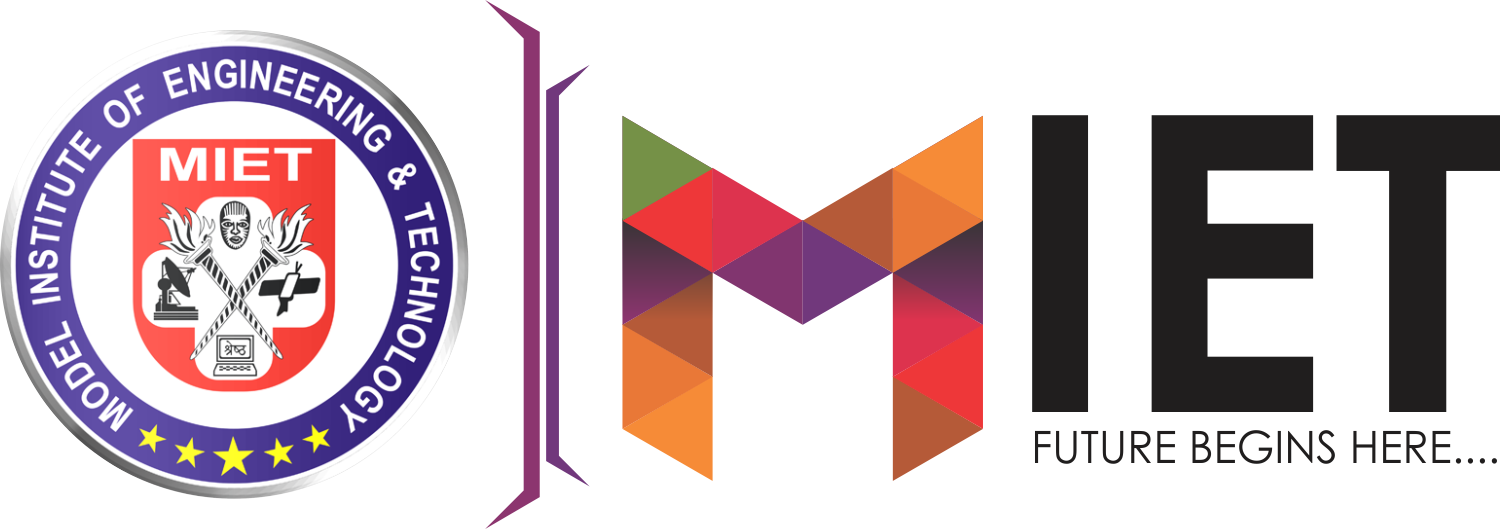
**MODEL INSTITUDE OF ENGINEERING AND TECHNOLOGY**



**A REPORT ON INTER PROCESS COMMUNICATION**

**BACHELOR’S OF ENGINEERING (Computer Science and Engineering)**

**SUBMITTED BY:**

NAMES :- GOKUL PANDEY, PRERIT MUJOO, SACHIN

DEV, ISHAN BHAT

ROLL NO. :- 2021A1R096, 2021A1R098, 2021A1R102,

2021A1R103

BRANCH :- CSE

SEMESTER :- 3RD

GROUP :- 6

**SUBMITTED TO** : MS. PRAGTI JAMWAL

**ACKNOWLEDGEMENT**

We are very thankful to everyone who supported us for this project and gives their guidance to complete our thesis work effectively and moreover on time.

We are equally grateful to our faculty Assistant Professor Ms. Pragti Jamwal, who gave us moral support and guided us in different matter and solutions regarding the topic.

We feel immensely proud in extending our heartiest thanks to Professor Dr. Ashok Kumar, HOD and Dean Academics Affair and Professor Dr. Ankur Gupta, Director MIET, for providing us a platform to improve in various Fields. They went out of their way and provides us with openhearted help and counsel. They have been a source of inspiration for us and their experience and knowledge have helped us in learning and giving this project the shape it has assumed.

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| S.NO. | TOPICS |
| 1. | PROJECT TITLE |
| 2. | ACKNOWLEDGEMENT |
| 3. | TABLE OF CONTENTS |
| 4. | INTRODUCTION |
| 5. | WHAT IS IPC |
| 6. | METHODS OF IPC |
| 7. | PROCESS |
| 8. | WHY WE NEED IPC |
| 9. | DIAGRAMMATIC REPRESENTATION |
| 10. | CODE |
| 11. | OUTPUTS |

**INTRODUCTION**

Inter process communication is a set of techniques for the exchange of data among multiple threads in one or more processes. Process may be running on one or more computers connected to the same network. Inter process communication through shared memory is a concept where two or more processes can access the common memory. And communication is done via this shared memory where changes can be made by one process and can be seen by another process.

Inter process communication (IPC) is used for exchanging data between multiple threads in one or more processes or programs. The Processes may be running on single or multiple computers connected by a network. The full form of IPC is Inter-process communication.

It is a set of programming interface which allow a programmer to coordinate activities among various program processes which can run concurrently in an operating system. This allows a specific program to handle many user requests at the same time.

Since every single user request may result in multiple processes running in the operating system, the process may require to communicate with each other. Each IPC protocol approach has its own advantage and limitation, so it is not unusual for a single program to use all of the IPC methods.

**What is Inter Process Communication**

IPC is a mechanism which allows the exchange of data between processes. It enables resource and data sharing between the processes without interference.

Processes that execute concurrently in the operating system may be either independent processes or cooperating processes.

A process is independent and it may or may not be affected by other processes executing in the system. Any process that does not share data with any other process is independent.

Suppose if a process is cooperating then, it can be affected by other processes that are executing in the system. Any process that shares the data with another process is called a cooperative process.

**METHODS OF IPC**

These are a few different approaches for Inter- Process Communication:

1. **Pipes**
2. **Shared Memory**
3. **Message Queue**
4. **Direct Communication**
5. **Indirect communication**
6. **Message Passing**
7. **FIFO**

**Pipe:-**

The pipe is a type of data channel that is unidirectional in nature. It means that the data in this type of data channel can be moved in only a single direction at a time. Still, one can use two-channel of this type, so that he can able to send and receive data in two processes. Typically, it uses the standard methods for input and output. These pipes are used in all types of POSIX systems and in different versions of window operating systems as well.

**Shared Memory:-**

It can be referred to as a type of memory that can be used or accessed by multiple processes simultaneously. It is primarily used so that the processes can communicate with each other. Therefore the shared memory is used by almost all POSIX and Windows operating systems as well.

**Message Queue:-**

In general, several different messages are allowed to read and write the data to the message queue. In the message queue, the messages are stored or stay in the queue unless their recipients retrieve them. In short, we can also say that the message queue is very helpful in inter-process communication and used by all operating systems.

**Message Passing:-**

It is a type of mechanism that allows processes to synchronize and communicate with each other. However, by using the message passing, the processes can communicate with each other without restoring the hared variables.

Usually, the inter-process communication mechanism provides two operations that are as follows:

* send (message)
* received (message)

**Direct Communication:-**

* In this type of communication process, usually, a link is created or established between two communicating processes. However, in every pair of communicating processes, only one link can exist.

**Indirect Communication**

* Indirect communication can only exist or be established when processes share a common mailbox, and each pair of these processes shares multiple communication links. These shared links can be unidirectional or bi-directional.

**PROCESS**

• SERVER READS FROM THE INPUT FILE.

• THE SERVER WRITES THE DATA EITHER USING PIPES OR MESSAGE QUEUE, OR FIFO.

• THE CLIENT READS THE DATA FROM THE IPC CHANNEL, AGAIN REQUIRING THE IPCBUFFER TO THE CLIENT PROCESS.

• FINALLY, THE DATA IS COPIED.

The operating system should always manage the message queue of the data which we have to input. Data should be received by another process at the same time. This whole task is carried out by the concept of shared memory.

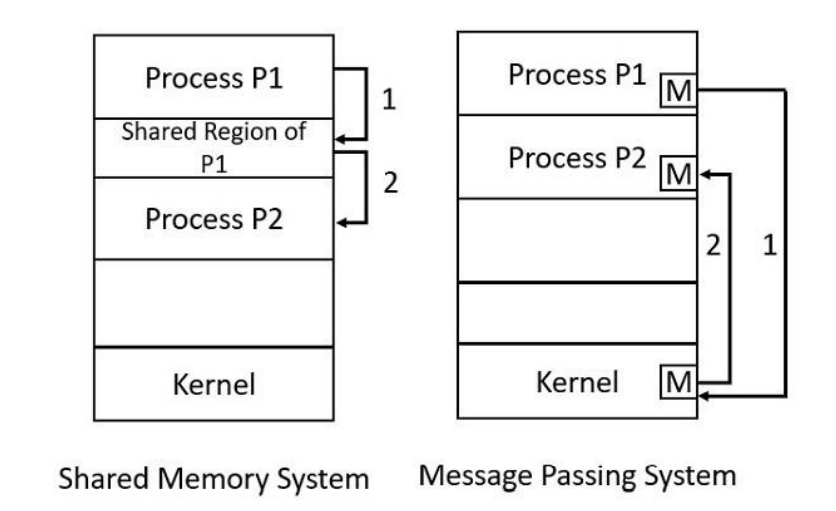
**WHY WE NEED IPC**

There are numerous reasons to use inter-process communication for sharing the data. Here are some of the most important reasons that are given below:

* It helps to speedup modularity
* Computational
* Privilege separation
* Convenience
* Helps operating system to communicate with each other and synchronize their actions as well.

Inter process communication (IPC) is one of the key mechanisms used by operating systems to achieve these goals. IPC helps processes communicate with each other without having to go through user-level routines or interfaces. It allows different parts of a program to access shared data and files without causing conflicts among them. In inter-process communication, messages are exchanged between two or more processes. Processes can be on the same computer or on different computers. In this article, we will discuss IPC and its need, and different approaches for doing IPC.

**DIAGRAMATIC REPRESENTATION**



**CODE**

**READERS PROCESS**

#include <sys/ipc.h>

#include <sys/shm.h>

#include <stdio.h>

int main()

{

key\_t key = ftok("shmfile",65);

int shmid = shmget(key,1024,0666|IPC\_CREAT);

char str = (char) shmat(shmid,(void\*)0,0);

printf("Data read from memory: %s\n",str);

shmdt(str);

shmctl(shmid,IPC\_RMID,NULL);

return 0;

}

**WRITERS PROCESS**

#include <sys/ipc.h>

#include <sys/shm.h>

#include <stdio.h>

int main()

{

key\_t key = ftok("shmfile",65);

int shmid = shmget(key,1024,0666|IPC\_CREAT);

char str = (char) shmat(shmid,(void\*)0,0);

printf("Write Data : ");

gets(str);

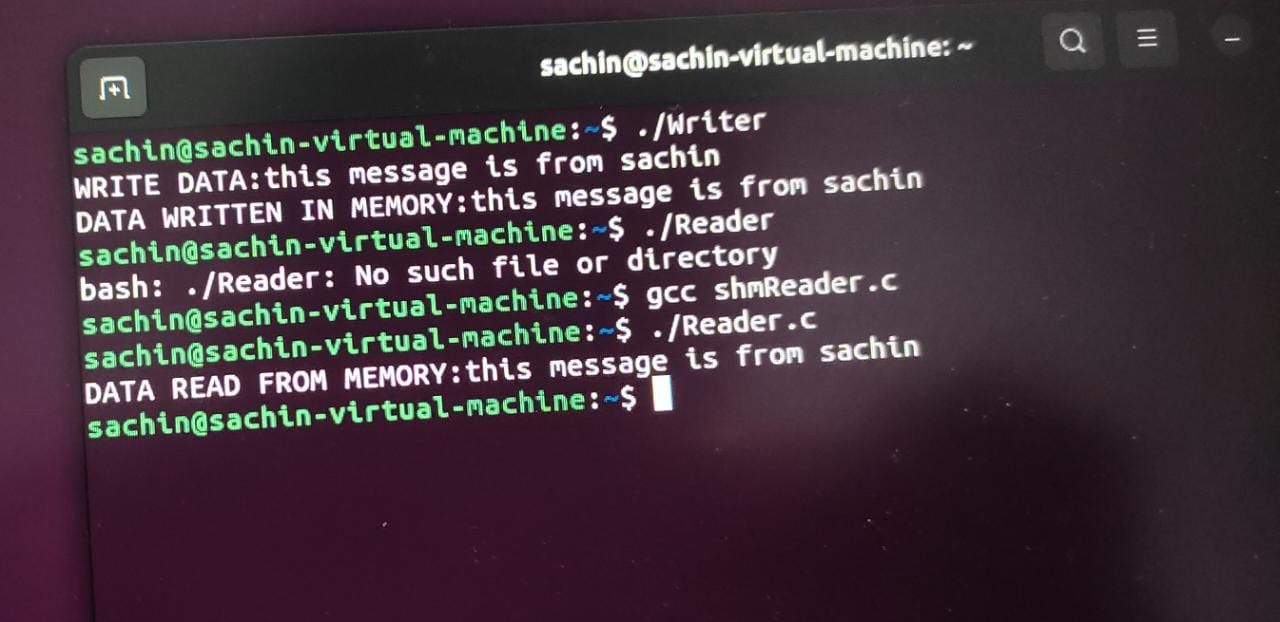
printf("Data written in memory: %s\n",str);

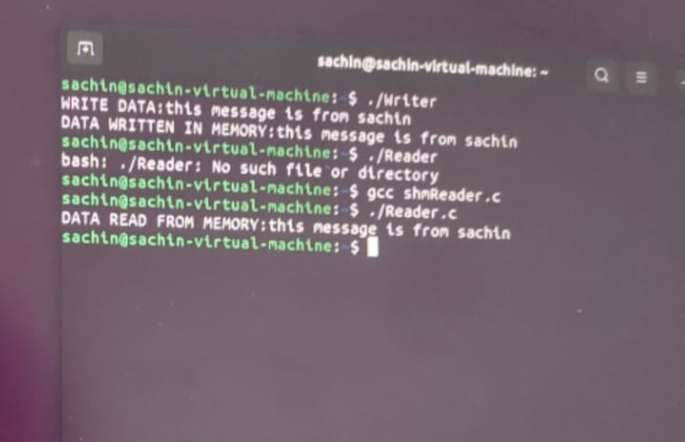
shmdt(str);

return 0;

}

**OUTPUTS**



****