Experiment No. 7

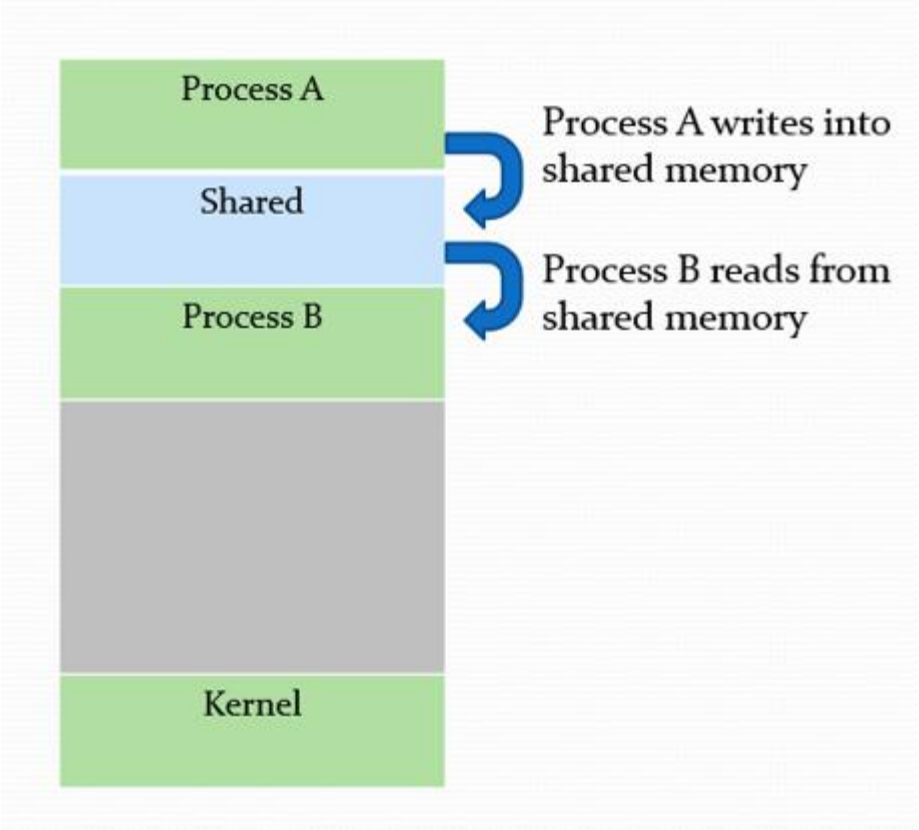
Write a program to implement interprocess communication using shared memory (SHM) in linux.

By

Harshali Bhuwad

IPC through shared memory

- Inter Process Communication through shared memory is a concept where two or more process can access the common memory.
- And communication is done via this shared memory where changes made by one process can be viewed bv another process.



The problem with pipes, fifo and message queue-

- The two process to exchange information. The information has to go through the kernel.
- Server reads from the input file.
- The server writes this data in a message using either a pipe, fifo or message queue.
- The client reads the data from the IPC channel, again requiring the data to be copied from kernel's IPC buffer to the client's buffer.
- Finally the data is copied from the client's buffer.A total of four copies of data are required (2 read and 2 write). So, shared memory provides a way by letting two or more processes share a memory segment. With Shared Memory the data is only copied twice – from input file into shared memory and from shared memory to the output file.

SYSTEM CALLS USED ARE:

ftok(): is use to generate a unique key.
shmget(): int shmget(key_t,size_tsize,intshmflg); upon successful completion, shmget() returns an identifier for the shared memory segment.
shmat(): Before you can use a shared memory segment, you have to attach yourself to it using shmat(). void *shmat(int shmid ,void *shmaddr ,int shmflg);
shmid is shared memory id. shmaddr specifies specific address to use but we should set
it to zero and OS will automatically choose the address.
shmdt(): When you're done with the shared memory segment, your program should detach itself from it using shmdt(). int shmdt(void *shmaddr);
shmctl(): when you detach from shared memory,it is not destroyed. So, to destroy shmctl() is used. shmctl(int shmid,IPC_RMID,NULL);

SHARED MEMORY FOR WRITER PROCESS

```
#include <iostream>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
using namespace std;
int main()
{
        // ftok to generate unique key
        key_t key = ftok("shmfile",65);
        // shmget returns an identifier in shmid
        int shmid = shmget(key,1024,0666|IPC_CREAT);
        // shmat to attach to shared memory
        char *str = (char*) shmat(shmid,(void*)0,0);
        cout<<"Write Data : ";
        gets(str);
        printf("Data written in memory: %s\n",str);
        //detach from shared memory
        shmdt(str);
        return 0;
}
```

SHARED MEMORY FOR READER PROCESS

```
#include <iostream>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>
using namespace std;

int main()
{
        // ftok to generate unique key
        key_t key = ftok("shmfile",65);
        // shmget returns an identifier in shmid
        int shmid = shmget(key,1024,0666|IPC_CREAT);
        // shmat to attach to shared memory
        char *str = (char*) shmat(shmid,(void*)0,0);
        printf("Data read from memory: %s\n",str);
        //detach from shared memory
        shmdt(str);
        // destroy the shared memory
        shmctl(shmid,IPC_RMID,NULL);
        return 0;
}
```