Ex. No.: 7
Date:

IPC USING SHARED MEMORY

Aim:

To write a C program to do Inter Process Communication (IPC) using shared memory between sender process and receiver process.

Algorithm:

sender

1. Set the size of the shared memory segment
2. Allocate the shared memory segment using shmget
3. Attach the shared memory segment using shmat
4. Write a string to the shared memory segment using sprintf
5. Set delay using sleep
6. Detach shared memory segment using shmdt

receiver

1. Set the size of the shared memory segment
2. Allocate the shared memory segment using shmget
3. Attach the shared memory segment using shmat
4. Print the shared memory contents sent by the sender process.
5. Detach shared memory segment using shmdt

Program Code:
sender.c

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/ipc.h>
#include<sys/shm.h>
#include<unistd.h>
#include<string.h>
 #define SHM_SIZE 1024
 int main(){
 key_t key = ftok("shmfile",65);
 int shmid = shmget(key, SHM_SIZE, 0666 | IPC_CREAT);
 char *shm = (char *)shmat(shmid, NULL, 0);
 int *flag = (int *)shm;

 while(1){
    printf("Sender: Enter a Message: ");
    fgets(shm + sizeof(int), SHM_SIZE - sizeof(int), stdin);
    shm[strcspn(shm + sizeof(int), "\n") + sizeof(int)] = 0;
```

```c
        *flag = 1;
        while(*flag==1) sleep(1);
        printf("Sender: Received response: %s\n", shm + sizeof(int));
        memset(shm + sizeof(int), 0, SHM_SIZE - sizeof(int));
        }
 shmdt(shm);
 return 0;
```

receiver.c

```c
#include<stdio.h>
#include<stdlib.h>
#include<sys/ipc.h>
#include<sys/shm.h>
#include<unistd.h>
#include<string.h>
#define SHM_SIZE 1024
int main(){
 key_t key = ftok("shmfile",65);
 int shmid = shmget(key, SHM_SIZE, 0666 | IPC_CREAT);
 char *shm = (char *)shmat(shmid, NULL, 0);
 int *flag = (int *)shm;
while(1){
    while(*flag==0)sleep(1);
        printf("Receiver: Received message: %s\n", shm + sizeof(int));
        printf("Receiver: Enter a response: ");
        fgets(shm + sizeof(int), SHM_SIZE - sizeof(int), stdin);
        shm[strcspn(shm + sizeof(int), "\n") + sizeof(int)] = 0;
        *flag = 0;
        }
 shmdt(shm);
 return 0;
}
```

Sample Output

Terminal 1

```
[root@localhost student]# gcc sender.c -o sender
[root@localhost student]# ./sender
```

Terminal 2

```
[root@localhost student]# gcc receiver.c -o receiver
[root@localhost student]# ./receiver
Message Received: Welcome to Shared Memory
[root@localhost student]#
```

Result:

Inter-process communication using shared memory has been successfully implemented, allowing processes to efficiently share and exchange data.