Ex. No. 7

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:

Sender file

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <unistd.h>

#define SHM_SIZE 1024  // size of shared memory

int main() {
    key_t key = ftok("shmfile", 65);  // generate unique key
    int shmid = shmget(key, SHM_SIZE, 0666 | IPC_CREAT);  // create shared memor
y

    if (shmid == -1) {
        perror("shmget failed");
        exit(1);
    }

    char *str = (char *) shmat(shmid, NULL, 0);  // attach to shared memory

    if (str == (char *)-1) {
        perror("shmat failed");
        exit(1);
    }

    sprintf(str, "Hello from Sender Process!");
    printf("Sender: Data written to shared memory: %s\n", str);

    sleep(5);  // wait so receiver can read

    shmdt(str);  // detach from shared memory
    return 0;
}
```

Receiver file:

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <unistd.h>

#define SHM_SIZE 1024

int main() {
    key_t key = ftok("shmfile", 65);  // same key as sender
    int shmid = shmget(key, SHM_SIZE, 0666);  // access shared memory

    if (shmid == -1) {
        perror("shmget failed");
        exit(1);
    }

    char *str = (char *) shmat(shmid, NULL, 0);  // attach to shared memory

    if (str == (char *)-1) {
        perror("shmat failed");
        exit(1);
    }

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

    shmdt(str);  // detach

    shmctl(shmid, IPC_RMID, NULL);  // destroy the shared memory

    return 0;
}
```

Output: