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 Code:

SENDER.C

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

#define SHM_SIZE 1024 // Size of the shared memory segment
```

```
int main() { key_t key; int shmid; char *shm;

// Generate a unique key for shared memory
key = ftok("shmfile", 65);

// Allocate shared memory segment
shmid = shmget(key, SHM_SIZE, 0666|IPC_CREAT);
if (shmid == -1) {
    perror("shmget failed");
    exit(1);
}

// Attach the shared memory segment to sender's address space
shm = (char*) shmat(shmid, NULL, 0);
if (shm == (char*) -1) {
    perror("shmat failed");
    exit(1);
}

// Write message to shared memory
printf("Sender Process: Writing to shared memory...\n");
sprintf(shm, "Hello from Sender!");

// Set delay for simulation
sleep(2);

// Detach the shared memory segment
shmdt(shm);
printf("Sender Process: Data written and detached from shared memory.\n");

return 0;


}
```

RECEIVER.C

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

#define SHM_SIZE 1024  // Size of the shared memory segment

int main() {
    key_t key;
    int shmid;
    char *shm;

    // Generate the same key used by sender for shared memory
    key = ftok("shmfile", 65);

    // Allocate shared memory segment
    shmid = shmget(key, SHM_SIZE, 0666);
    if (shmid == -1) {
        perror("shmget failed");
        exit(1);
    }

    // Attach the shared memory segment to receiver's address space
    shm = (char*) shmat(shmid, NULL, 0);
    if (shm == (char*) -1) {
```

```
        perror("shmat failed");
        exit(1);
    }

    // Read message from shared memory
    printf("Receiver Process: Reading from shared memory...\n");
    printf("Received message: %s\n", shm);

    // Detach the shared memory segment
    shmdt(shm);
    printf("Receiver Process: Detached from shared memory.\n");

    // Deallocate shared memory segment
    shmctl(shmid, IPC_RMID, NULL);
    printf("Receiver Process: Shared memory deallocated.\n");

    return 0;
}
```

OUTPUT:

```
Sender Process: Writing to shared memory...
Sender Process: Data written and detached from shared memory.
Receiver Process: Reading from shared memory...
Received message: Hello from Sender!
Receiver Process: Detached from shared memory.
Receiver Process: Shared memory deallocated.
```

RESULT:

C program to do Inter Process Communication (IPC) using shared memory between sender process and receiver process is executed successfully.