

Ex No: 7 IPC USING SHARED MEMORY**Date: 19.02.2025****Aim:**

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

Algorithm:**Sender Process**

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 Process

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 <sys/ipc.h>

#include <sys/shm.h>

#include <unistd.h>
```

```
#include <string.h>
```

```
int main() {
```

```
    key_t key = ftok("shmfile",65); // Generate unique key
```

```
    int shmid = shmget(key, 1024, 0666|IPC_CREAT); // Create shared
```

```
    memory char *str = (char*) shmat(shmid, (void*)0, 0); // Attach to  
    shared memory
```

```
    sprintf(str, "Welcome to Shared Memory");
```

```
    printf("Message Sent: %s\n", str);
```

```
    sleep(5); // Delay to allow receiver to read
```

```
    shmdt(str); // Detach from shared memory
```

```
    return 0;
```

```
}
```

receiver.c

```
#include <stdio.h>
```

```
#include <sys/ipc.h>
```

```
#include <sys/shm.h>
```

```
#include <unistd.h>
```

```
int main() {
```

```
    key_t key = ftok("shmfile",65); // Generate same key
```

```
    int shmid = shmget(key, 1024, 0666|IPC_CREAT); // Access shared
```

```
    memory char *str = (char*) shmat(shmid, (void*)0, 0); // Attach to  
    shared memory
```

```
    printf("Message Received: %s\n",str);
```

```
    shmdt(str); // Detach from shared memory

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

    return 0;
}
```

SampleOutput:

Terminal 1:

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

```
[root@localhost student]# ./sender
```

Message Sent: Welcome to Shared Memory

Terminal 2:

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

```
[root@localhost student]# ./receiver
```

Message Received: Welcome to Shared Memory

Result:

Thus, the C program for Inter Process Communication (IPC) using shared memory was successfully executed, and the message was successfully passed from the sender process to the receiver process.