Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

B.Tech. Winter Semester 2023-24

School Of Computer Science and Engineering

(SCOPE)

Digital Assignment - IV

Operating System Lab

Apurva Mishra, 22BCE2791

20 September 2024

1. Questions

Problem 1.1.
Write a LINUX C programme to enable the inter process communication mechanism between the process writer and reader by utilising shared memory.

Note: Create two IPC programmes that use shared memory. Program 1 will create the shared segment, attach it to it, and write some content into it. Then, Program 2 will connect to the shared segment and read the value that Program 1 has written.

server.c (writer)

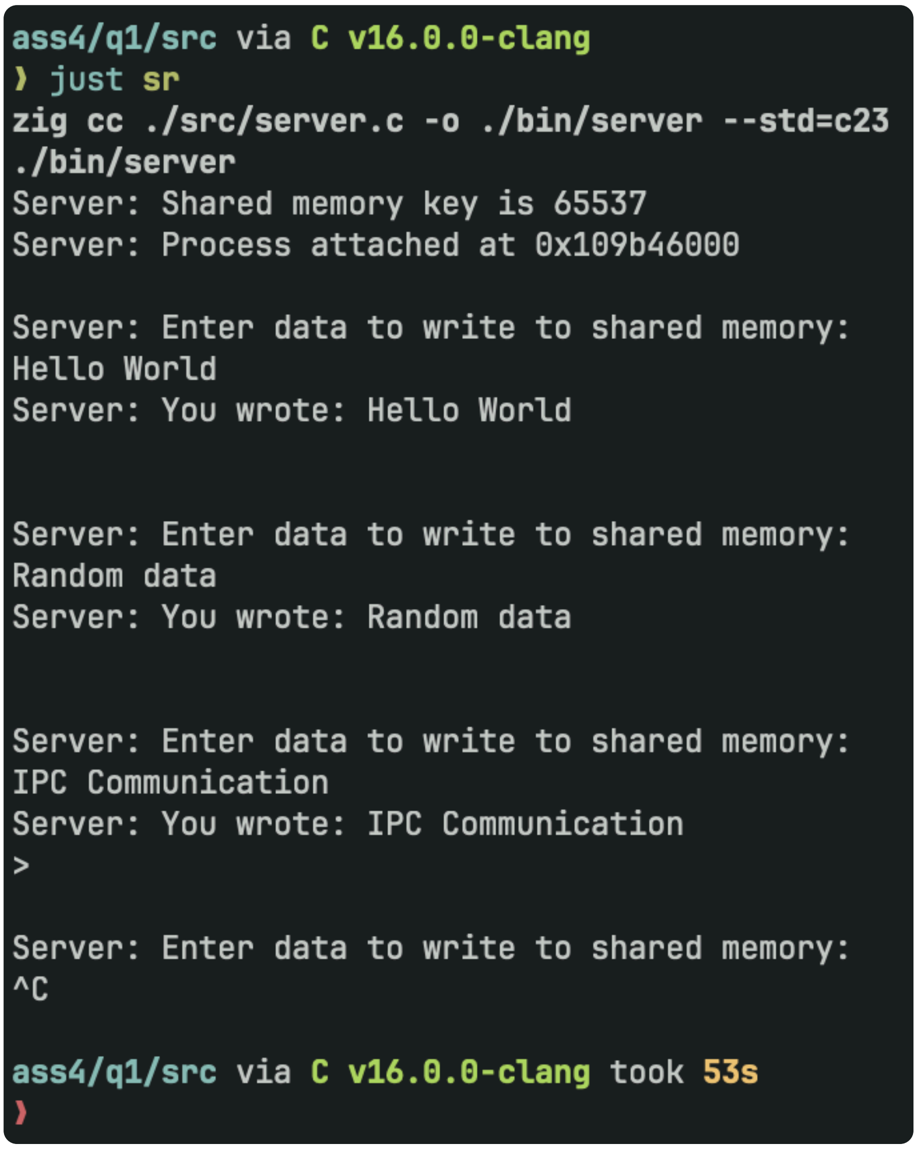
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/shm.h>
#include <unistd.h>

#define SHM_SIZE 1024

int main() {
  int i;
  void *shared_memory;
  char buff[100];
  int shmid;

  shmid = shmget((key_t)2323, SHM_SIZE, 0666 | IPC_CREAT);

  printf("Server: Shared memory key is %d\n", shmid);
  shared_memory = shmat(shmid, NULL, 0);

  printf("Server: Process attached at %p\n", shared_memory);

  while (true) {
    printf("\nServer: Enter data to write to shared memory: \n");

    read(0, buff, 100);
    strcpy(shared_memory, buff);
    printf("Server: You wrote: %s\n", (char *)shared_memory);
  }

  printf("\nServer: Exiting...\n");
  return 0;
}
```

client.c (reader)

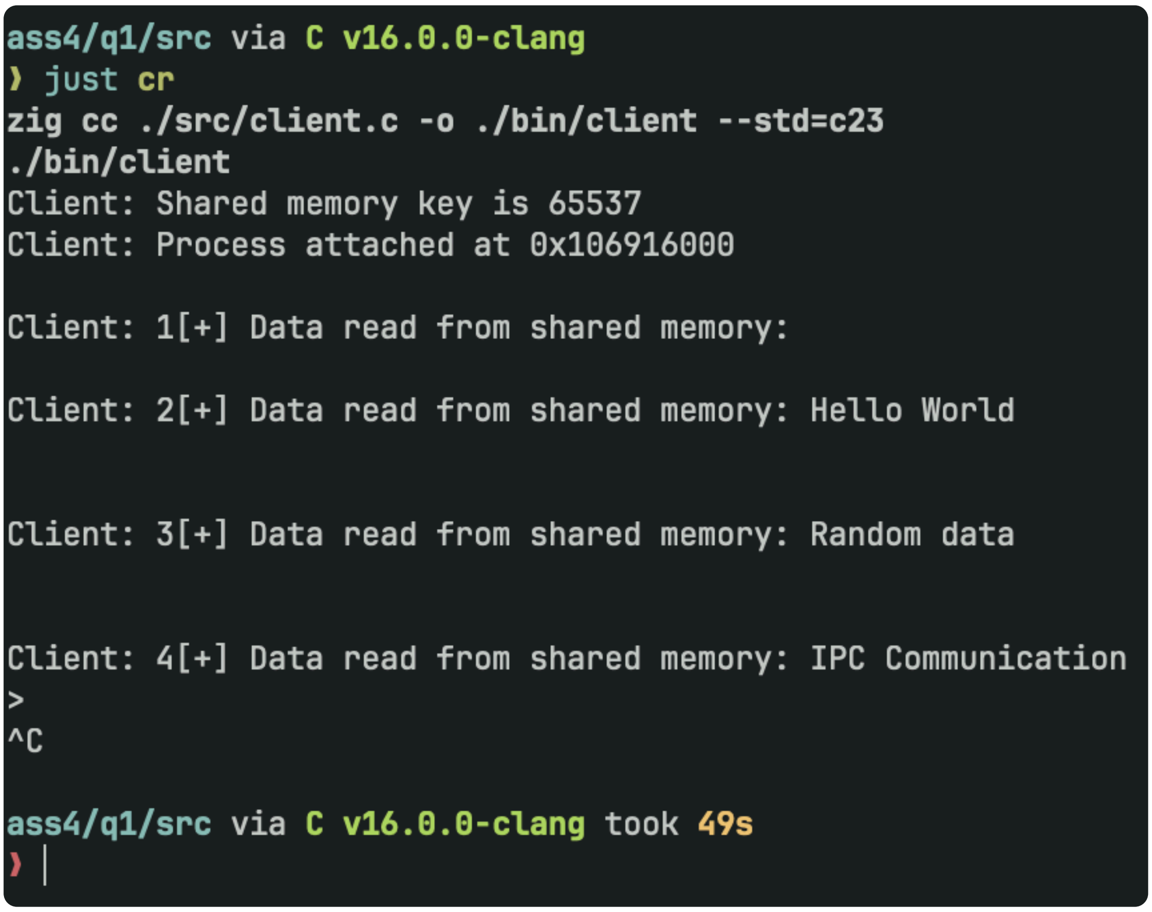
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/shm.h>
#include <unistd.h>

#define SHM_SIZE 1024

int main() {
  int i;
  void *shared_memory;
  char buff[100];
  int shmid;

  shmid = shmget((key_t)2323, SHM_SIZE, 0666);
  printf("Client: Shared memory key is %d\n", shmid);

  shared_memory = shmat(shmid, NULL, 0);
  printf("Client: Process attached at %p\n", shared_memory);

  int count = 1;
  while (true) {
    printf("\nClient: %d[+] Data read from shared memory: %s\n", count,
           (char *)shared_memory);
    sleep(15);
    count += 1;
  }

  printf("\nClient: Exiting...\n");
  return 0;
}
```

Output: Sender and Receiver

Here `server.c` is writing to the shared memory and `client.c` reads from the shared memory. Also, `just sr` and `just cr` are build scripts for server and client respectively. zig cc is used as the C compiler on MacOS.



```
ass4/q1/src via C v16.0.0-clang
❯ just sr
zig cc ./src/server.c -o ./bin/server --std=c23
./bin/server
Server: Shared memory key is 65537
Server: Process attached at 0x109b46000

Server: Enter data to write to shared memory:
Hello World
Server: You wrote: Hello World


Server: Enter data to write to shared memory:
Random data
Server: You wrote: Random data


Server: Enter data to write to shared memory:
IPC Communication
Server: You wrote: IPC Communication
>

Server: Enter data to write to shared memory:
^C

ass4/q1/src via C v16.0.0-clang took 53s
❯
```



```
ass4/q1/src via C v16.0.0-clang
❯ just cr
zig cc ./src/client.c -o ./bin/client --std=c23
./bin/client
Client: Shared memory key is 65537
Client: Process attached at 0x106916000

Client: 1[+] Data read from shared memory:

Client: 2[+] Data read from shared memory: Hello World


Client: 3[+] Data read from shared memory: Random data


Client: 4[+] Data read from shared memory: IPC Communication
>
^C

ass4/q1/src via C v16.0.0-clang took 49s
❯ |
```