

Aim: To implement inter process communication using shared memory

Description: In this program we use shmget(),shmat(). Concept of shared memory is used.in the server.c we create a segment and attach it to our data space.Then in the client.c that segment is accessed and attached to the memory and the message is read. A * character is attached so as to notify that it has accessed the memory.

shm_server.c

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

#define SHMSZ 27
main()
{
    char c;
    int shmid;
    key_t key;
    char *shm, *s;

    /*
     * We'll name our shared memory segment
     * "5678".
     */
    key = 5678;

    /*
     * Create the segment.
     */
    if ((shmid = shmget(key, SHMSZ, IPC_CREAT | 0666)) < 0) {
        perror("shmget");
        exit(1);
    }

    /*
     * Now we attach the segment to our data space.
     */
    if ((shm = shmat(shmid, NULL, 0)) == (char *) -1) {
        perror("shmat");
        exit(1);
    }

    /*
     * Now put some things into the memory for the
     * other process to read.
     */
}
```

```
s = shm;

for (c = 'a'; c <= 'z'; c++)
    *s++ = c;
*s = NULL;

/*
 * Finally, we wait until the other process
 * changes the first character of our memory
 * to '*', indicating that it has read what
 * we put there.
 */
while (*shm != '*')
    sleep(1);

exit(0);
}
```

shm_client.c

```
/*
 * shm-client - client program to demonstrate shared memory.
 */
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>

#define SHMSZ 27

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

    /*
     * We need to get the segment named
     * "5678", created by the server.
     */
    key = 5678;

    /*
     * Locate the segment.
     */
    if ((shmid = shmget(key, SHMSZ, 0666)) < 0) {
        perror("shmget");
        exit(1);
    }
}
```

```

/*
 * Now we attach the segment to our data space.
 */
if ((shm = shmat(shmid, NULL, 0)) == (char *) -1) {
    perror("shmat");
    exit(1);
}

/*
 * Now read what the server put in the memory.
 */
for (s = shm; *s != NULL; s++)
    putchar(*s);
putchar('\n');

/*
 * Finally, change the first character of the
 * segment to '*', indicating we have read
 * the segment.
 */
*shm = '*';

exit(0);
}

```

Output:

```

week-7 -- -bash -- 69x17
shmserver.c:7:1: warning: type specifier missing, defaults to 'int'
      [-Wimplicit-int]
main()
^
shmserver.c:39:4: warning: incompatible pointer to integer conversion
      assigning to 'char' from 'void *' [-Wint-conversion]
*s = NULL;
   ^
shmserver.c:48:1: warning: implicit declaration of function 'sleep'
      is invalid in C99 [-Wimplicit-function-declaration]
sleep(1);
^
3 warnings generated.
Rohits-MacBook-Pro:week-7 rohitdandamudi$ ./a.out
Rohits-MacBook-Pro:week-7 rohitdandamudi$ ./a.out
*bcdefghijklmnopqrstuvwxyz
Rohits-MacBook-Pro:week-7 rohitdandamudi$ _

week-7 -- -bash -- 67x17
main()
^
shmclient.c:24:1: warning: implicitly declaring library function
      'exit' with type 'void (int) __attribute__((noreturn))'
      [-Wimplicit-function-declaration]
exit(1);
^
shmclient.c:24:1: note: include the header <stdlib.h> or explicitly
      provide a declaration for 'exit'
shmclient.c:36:18: warning: comparison between pointer and integer
      ('char' and 'void *')
for (s = shm; *s != NULL; s++)
               ^
3 warnings generated.
Rohits-MacBook-Pro:week-7 rohitdandamudi$ ./a.out
abcdefghijklmnopqrstuvwxyz
Rohits-MacBook-Pro:week-7 rohitdandamudi$ 

```

Analysis: Thus we see that interprocess communication is achieved using shared memory.