Ex.No:7(i) INTER PROCESS COMMUNICATION USING SHARED

Date: MEMORY

**AIM**:

To write a c program for inter process communication using shared memory.

**ALGORITHM:**

1. Create a child process using fork() command.
2. Create shared memory for parent process shmget() system call.
3. The current process writes the content in shared memory using shmptr pointer.
4. Attach the same shared memory to the child process.
5. The data in shared memory is read by the child process using shmptr() pointer.
6. Detach and rebase the shared memory.

**PROGRAM:**

#include<stdio.h>

#include<sys/shm.h>

#include<sys/ipc.h>

int main()

{

int child,shmid,i;

char \*shmptr;

child=fork();

if(!child)

{

shmid=shmget(2041,32,0666|IPC\_CREAT);

shmptr=shmat(shmid,0,0);

printf("\n PARENT PROCESS WRITING \n");

for(i=0;i<10;i++)

{

shmptr[i]='a'+i;

putchar(shmptr[i]);

}

printf("\n\n %s",shmptr);

wait(NULL);

}

else

{

shmid=shmget(2041,32,0666);

shmptr=shmat(shmid,0,0);

printf("\n CHILD PROCESS READING\n");

for(i=0;i<10;i++)

putchar(shmptr[i]);

shmdt(NULL);

shmctl(shmid,IPC\_RMID,NULL);

}

return 0;

}

**SAMPLE INPUT AND OUTPUT:**

[root@local host~]# ./a.out

parent process writing

abcdefghij

child process reading

abcdefghij

**CONCLUSION:**

The above program for inter process communication using shared memory was executed successfully and output was verified.

**Ex.No:7(ii) INTER PROCESS COMMUNICATION USING PIPES**

Date:

**AIM:**

To write a c program for implementation of inter process communication using pipes.

**ALGORITHM:**

1. Create the child process using fork() command
2. Create the pipe structure using pipe() command.
3. Close the read end of the pipe using close() command.
4. Write the data in the pipe using write() command.
5. Close the writer end of the child process using close() command.
6. Display the string in the pipe using read system call.
7. Stop the process.

**PROGRAM:**

#include<stdio.h>

int main()

{

int fd[2],child;

char a[10];

printf("\n enter the string to store into the pipe");

scanf("%s",a);

pipe(fd);

child=fork();

if(!child)

{

close(fd[0]);

write(fd[1],a,5);

wait(0);

}

else

{

close(fd[1]);

read(fd[0],a,5);

printf("\n the string reterived from the pipe is%s\n",a);

}

return 0;

}

**SAMPLE INPUT AND OUTPUT:**

[root@local host~]# ./a.out

Enter The String To Store Into The Pipe

Temple

The String Reterived From The Pipe Is: Temple

**CONCLUSION:**

The above program for inter process communication using pipes was executed successfully and output was verified.