REPORT

*AIM

The aim is to get ourselves accustomed with the concepts of processes and shared memory.

*New Functions & Libraries Used

Ref:- Operating Systems Concepts.(#include <fcntl.h>,#include <sys/shm.h>,#include <sys/stat.h>, #include <sys/mman.h>)

```
shm_open(name,O_CREAT | O_RDWR | O_TRUNC,0666);

ftruncate(fd,SIZE);

sprintf(ptr,"%s",str1);
```

Ref:-Google(#include <sys/time.h> ,#include <sys/types.h> ,#include <sys/wait.h> ,#include <unistd.h>)

```
fork()

gettimeofday(&start,NULL);

execvp(temp[0],temp)
```

*Low Level Design Of My Program

Input handling:

To attain inputs i had to change my standard main design to the design shown below:

```
int main(int argc,char** argv)

{

}
```

It’s convenient to use this design as the input will be given after ./a.out.argc represents number of arguments of input and argv is a 2D char array capable of storing all the input strings given.I loaded all input strings apart from ./a.out into a 2D temp array.temp[0] has input’s bash command which lies in bin and temp contains the whole input strings.

I also created shared memory using memory mapped files.This associates the region of shared memory with a file.Using shm_open() I created a shared memory object named "Shared_memory"

Then I used ftruncate(fd,SIZE) function to allocate the size of object in bytes.(in my prog size =4096). Now I called fork() to create a child process which executes the input command using execvp() function. Before calling execvp() I recorded the starting time stamp using gettimeofday(&start,NULL) function.Then i used the mmap() function to establish a memory mapped file containing the shared memory object.I created struct timeval start to store the start time sec and micro sec.For the sake of my convenience I concatenated secs and micro secs into a string with a comma in between them and then loaded it into the shared memory to a pointer ptr using sprintf() function.Now I called execvp(temp[0],temp) to execute the input.

Parent waits till the child completes its process and then it records the time stamp using gettimeofday(&end,NULL) function.Again I used the mmap() function for the same purpose.Now ptr has the concatenated string.I used strtok(ptr,","),strtok(NULL,",") to get starting time stamp sec’s and micro-sec’s but these are stored in strings so I converted them into integers.End has the end time stamp hence I obtained time taken by using simple arithmetic operations.I also used munmap(ptr,SIZE) & shm_unlink(name) for unmapping the memory and to unlink the shared object.

*Output Analysis

My output basically executes the input command given and shows the time taken to execute the the command in terms of seconds.I also used error handling wherever possible like:

->If fork returns -1;my output is (“fork error\n”)

->if mmap fails then ,my output is(“mmap error\n”)

->if execvp returns -1,my output is(“exec error\n”)

Some of my major observations of output analysis are: even if I executed the same command using my code I am getting different elapsed times with slight deviation.my output is not capable of executing the history command as I am not storing the all previous command .My code also obviously can’t run cd <location> command the displayed elapsed time in case of history and cd are not correct as they are not executing.

-----------------------------THE END-------------------------------