**Shared Memory:**

Shared memory is a memory shared between two or more processes.

**It is Asynchronous, thats we use semaphore with it.**

Why do we need a shared memory?

Each process has its own address space, if any process wants to communicate with some other information from its own address space to other process, then its possible with IPC.

We know that to communicate between two or more process, we use shared memory but before using shared memory what needs to be done with system calls,

> Create shared memory segment or use already created shared memory segment (shmget())

> Attach the process to already created shared memory ( shmat() )

> Detach the process from aleady attached shared memory segment ( shmctl() )

**To Create**:

*#include<sys/ipc.h>*

*#include<sys/shm.h>*

*int* ***shmget****(key\_t* ***key****, size\_t* ***size****, int* ***shmflg****);*

The above system call creates or allocates a system V shared memory segment.

**Key**

**>** It recognizes the shared memory segment.

**>** It can be a arbitrary value orit can be derived from the library function ftok().

> The key can be used as IPC\_PRIVATE means, running process as server and client.

> If client wants to use shared memory with this key, then client must be a child process of server. Child process needs to be created after the parent process obtains the shared memory.

**Size**

> The size of the shared memory segment rounded to multiple.

**Shmflg**

> It specify the required shared memory lag suzh as IPC\_CREAT or IPC\_EXCL.

**Return value:**

On success, This sysem call would return valid shared memory identifier and On failure, it return -1.

**To Attach: (Creating the path way to the shared memory)**

*#include<sys/tpes.h>*

*#include<sys/shm.h>*

*void\** ***shmat****( int* ***shmid****, const void\** ***shmaddr****, int* ***shmflg****)*

The shmat system call perform attaching a shared memory segment to address space of the calling process.

The arguments that needs to be passed are,

**shmid**

**>** It is the identifier to the shared memory segment.

> This id is the return value of shmget().

**shmaddr**

> It is to specify the attaching address.

> If shmaddr is NULL, the system by default chooses the suitable address to attach the segment.

> if shmaddr is not NULL and SHM\_RND is specified in shmflg, the attach is equal to the address of the nearest multiple of SHMLBA (lower boundary address).

> else shmaddr must be page alligned addressat which shared memory attachment occur.

**shmflg**

> It specifies the required shared memory flag SHM\_RND or SHM\_EXEC or SHM\_RDONLY or SHMREMAP.

**Return value:**

The call would return the address of attached shared memory on success

-1 on failure.

**To detach: (Destroying the path way to the shared memory)**

*#include<sys/types.h>*

*#include<sys/shm.h>*

*int shmdt(const void\* shmaddr)*

The above system call would perform etach of the shared memory for the address space of the calling process.

The arguments to be passed,

The address of the shared memory that has to detached. The to be detached segment must be the address returned by the shmat() system call.

Return value:

On Success: 0

On Failure: -1

**To control:**

*#include<sys/ipc.h>*

*#include<sys/shm.h>*

*int* ***shmctl****(int* ***shmid****, int* ***cmd****, struct shimd\_ds \*****buf****);*

The above system call is used to control the operation for the shared memory segment.

The arguments to be passed,

**shmid**

**>** It’s the identifier of the shared memory.

> It is the return value of shmget() system call.

**cmd**

> It’s the command to perform the required control operation on the shared memory segment.

> IPC\_STAT – Copies info of current value of each member to struct smid\_ds.

IPC\_SET – Set user id, group id of the owner, permission etc, pointed to by the struct buf.

IPC\_RMID – Marks the segment to be destroyed, after last process is detached.

IPC\_INFO – Returns the information about shared memory limit and parameter in structure.

SHM\_INFO – Returns a shm\_info structure containing information.

**buf**

A pointer to the shared memory structure named struct shmid\_ds.

**Return Value**

On success: Returns the index or identifier of the shared memory.

On Failure: -1 value.