

***BAHIR DAR UNIVERSTY***

***INSTITUTE OF TECHNOLOGY***

***FACULTY OF COMPUTING***

***DEPARTMENT OF SOFTWARE ENGINEERING***

*Operating system and system programming individual assignment*

*System call: int shmctl(int shmid, int cmd, struct shmid\_ds \*buf)*

*Name: Samrawit Fikremariam*

*Id: 1308042*

*Submitted to: Wendmu Baye*

*Submission Date: 17/11/14*

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

* ***shmid – shared memory segment id***
* ***cmd – command flag***
* ***buf – shmid\_ds structure buffer for return or set parameters***

***What is Shmctl?***

*The shmctl() function****allows the caller to control the shared memory segment specified by the shmid parameter****. A shared memory segment is controlled by setting the cmd parameter to one of the following values:*

*What is Shmctl in Linux?*

*shmctl()****performs the control operation specified by cmd on the System V shared memory segment whose identifier is given in shmid****. The buf argument is a pointer to a shmid\_ds structure, defined in <sys/shm.*

***Name :*** *shmctl - System V shared memory control.*

***Summary:*** *#include<sys/ipc.h>*

*#include <sys/shm.h>*

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

***What does shmctl do?***

*The shmctl() function****allows the caller to control the shared memory segment specified by the shmid parameter****. Remove the shared memory segment identifier shmid from the system and destroy the shared memory segment.*

***About shmctl system calls?***

*SYSTEM CALL: shmctl();*

*PROTOTYPE: int shmctl ( int shmqid, int cmd, struct shmid\_ds \*buf );*

*RETURNS: 0 on success*

*-1 on error: errno = EACCES (No read permission and cmd is IPC\_STAT)*

*EFAULT (Address pointed to by buf is invalid with IPC\_SET and*

*IPC\_STAT commands)*

*EIDRM (Segment was removed during retrieval)*

*EINVAL (shmqid invalid)*

*EPERM (IPC\_SET or IPC\_RMID command was issued, but*

*calling process does not have write (alter)*

*access to the segment)*

***Parametrs of system call shmctl***

*This system call have 3 parametrs which are shmid, cmd, and shmid\_ds \*buf .*

1. ***The buf parameter****:* i*s a pointer to a shmid\_ds data structure as defined in the sys/shm.h header file. The buf argument is a pointer to a shmid\_ds structure, defined in<sys/shm.h> as follows:*

*struct shmid\_ds {*

*struct ipc\_perm shm\_perm; /\* Ownership and permissions \*/*

*size\_t shm\_segsz; /\* Size of segment (bytes) \*/*

*time\_t shm\_atime; /\* Last attach time \*/*

*time\_t shm\_dtime; /\* Last detach time \*/*

*time\_t shm\_ctime; /\* Creation time/time of last modification via shmctl() \*/*

*pid\_t shm\_cpid; /\* PID of creator \*/*

*pid\_t shm\_lpid; /\* PID of last shmat(2)/shmdt(2) \*/*

*shmatt\_t shm\_nattch; /\* No. of current attaches \*/*

*};*

1. ***The cmd parameter***

*It is command flag.*

*The shared memory control operation to be performed, which is specified as one of the following:*

***IPC\_STAT***

*Use this operation to obtain status information for the shared memory identified by the****shmid****parameter. The current value of each field in the shmid\_ds data structure associated with the****shmid****parameter is placed in the data structure pointed to by the****buf****parameter. The contents of this structure are defined in the sys/shm.h header file. This operation requires read permission for the shared memory identified by the****shmid****parameter.*

***IPC\_SET***

*Use this operation to set the values of the following fields of the shmid\_ds data structure associated with the****shmid****parameter to the corresponding value in the structure pointed to by the****buf****parameter:*

* *shm\_perm.uid*
* *shm\_perm.gid*
* *shm\_perm.mode (only the low-order 9 bits)*

*This operation can be performed only by a process that has an effective user ID (UID) equal to one of the following:*

* *A process with the same privileges as the shared memory identified by the****shmid****parameter*
* *The value of shm\_perm.cuid or shm\_perm.uid in the shmid\_ds data structure associated with the****shmid****parameter.*

***IPC\_RMID***

*Use this operation to remove the following from the z/TPF system:*

* *The shared memory identifier specified by the****shmid****parameter*
* *The shared memory associated with the****shmid****parameter*
* *The shmid\_ds data structure associated with the****shmid****parameter.*

*This operation can be performed only by a process that has an effective UID equal to one of the following:*

* *A process with the same privileges as the shared memory identified by the****shmid****parameter*
* *The value of shm\_perm.cuid or shm\_perm.uid in the shmid\_ds data structure associated with the****shmid****parameter.*

*Removal is completed asynchronously to the return from the shmctl function when the last attached shared memory segment is detached. When IPC\_RMID is processed, no more attaches for the shared memory identified by the* **shmid** parameter are allowed.

1. ***The shmid parameter***

*The shared memory identifier for which this operation is to take place. The shared memory identifier is returned by the****shmget****function. The fields of the shmid\_ds structure are as follows:*

* ***shm\_perm***

*This is an ipc\_perm structure (see below) that specifies*

*the access permissions on the shared memory segment.*

* ***shm\_segsz***

*Size in bytes of the shared memory segment.*

* ***shm\_atime***

*Time of the last* [*shmat(2)*](https://man7.org/linux/man-pages/man2/shmat.2.html) *system call that attached this*

*segment.*

* ***shm\_dtime***

*Time of the last* [*shmdt(2)*](https://man7.org/linux/man-pages/man2/shmdt.2.html) *system call that detached this*

*segment.*

* ***shm\_ctime***

*Time of creation of segment or time of the last* ***shmctl****()*

***IPC\_SET*** *operation.*

* ***shm\_cpid***

*ID of the process that created the shared memory segment.*

* ***shm\_lpid***

*ID of the last process that executed a* [*shmat(2)*](https://man7.org/linux/man-pages/man2/shmat.2.html) *or*

[*shmdt(2)*](https://man7.org/linux/man-pages/man2/shmdt.2.html) *system call on this segment.*

* ***shm\_nattch***

*Number of processes that have this segment attached.*

*The ipc\_perm structure is defined as follows (the highlighted*

*fields are settable using IPC\_SET):*

*struct ipc\_perm {*

*key\_t \_\_key; /\* Key supplied to shmget(2) \*/*

*uid\_t* ***uid****; /\* Effective UID of owner \*/*

*gid\_t* ***gid****; /\* Effective GID of owner \*/*

*uid\_t cuid; /\* Effective UID of creator \*/*

*gid\_t cgid; /\* Effective GID of creator \*/*

*unsigned short* ***mode****; /\** ***Permissions*** *+ SHM\_DEST and*

*SHM\_LOCKED flags \*/*

*unsigned short \_\_seq; /\* Sequence number \*/*

*};*

*Successful IPC\_INFO or SHM\_INFO syscalls return index of highest used entry in the kernel’s array of shared memory segments. Successful SHM\_STAT syscalls return id of memory segment provided in shmid. Everything else returns zero upon success*

***Source code that implements this system call***

*The work of this system call is to get or set control details on shared memory segment. The following code attaches shared memory to the address space of the calling process by the shmget function. The shared memory identifier is removed from the system by using the shmctl function when it is no longer needed.*

*#include <sys/ipc.h>*

*#include <sys/shm.h>*

*void main(void)*

*{*

*Key\_t i;*

*int shm;*

*void \*addr;*

*struct shmid\_ds buf;*

*i = ftok("/usr",3);*

*shm = shmget(i,8000,IPC\_CREAT+S\_IRUSR+S\_IWUSR);*

*addr = shmat(shm,NULL,0);*

*i = shmctl(shm,IPC\_RMID,&buf);*

*}*

***Normal return***

*If successful, the shmctl function returns a value of 0.*

***Error return***

*If successful, the shmctl function returns a value of -1 and sets errno to one of the following:*

***EACCES ;*** *the cmd parameter is specified as IPC\_STAT, but the calling process doesn’t have read permission for the shared memory.*

***EINVAL;*** *the value of the shmid parameter is not a valid share memory identifier or the value of cmd parameter is not a valid operation.*

***EPEREM;*** *the effective UID of the calling process is not equal to that of process*



