

# Ateneo de Manila University

## IPC: Shared Memory



Department of Information Systems and  
Computer Science

S.Y. 2001-2002

<http://sysads.ateneo.net/wyu/>

[wyy@admu.edu.ph](mailto:wyy@admu.edu.ph)

## Interprocess Communications

- ★ mechanism in which processes communicate with one another
- ★ in order to facilitate communication among different processes
- ★ special mechanisms for process communications:
  - signals
  - semaphores
  - shared memory
  - pipes
  - message passing

## Shared Memory

- ★ a region in memory is shared between multiple processes
- ★ fastest form of IPC since data is not copied from process to process
- ★ are pre-specified by the system and defined in `sys/ipc.h` and `sys/shm.h`

## Issues with Shared Memory

- Mutual Exclusion Issues
- Synchronization Issues
- can be modeled as the readers/writers problem

## Initializing or Opening a Shared Memory Segment

★ `int shmget(key_t key, int size, int shmflg);`

- returns a shared memory identified corresponding to the share memory segment associated with the value of `key`
- `size` determines the amount of memory reserved
- `shmflg` determines modes of opening or initializing shared memory segments
- allowable `shmflags` are: `IPC_CREATE` and `IPC_EXCL`
- `shmflg` is set to 0 if an existing shared memory segment is to be opened

★ `int shmctl(int shmid, int cmd, struct shmid_ds *buf);`

- used to manipulate the `shmid_ds` data structure
- similar behavior with `semctl`

## Attaching/Detaching a Shared Memory Segment

```
★ void *shmat ( int shmid, const void *shmaddr,  
int shmflg );
```

- attaches a shared memory segment determined by `shmid`
- returns a pointer to that shared memory segment
- if `shmaddr` is zero it looks for an unmapped region for that shared memory segment
- if `shmaddr` is not zero `SHM_RND` is asserted to the `shmflg` and attaches the shared memory segment at that address
- `shmflg` can be set to `SHM_RDONLY` sets the area to read-only

```
★ int shmdt ( const void *shmaddr );
```

- detaches a shared memory segment
- `shmaddr` must be the address returned by the `shmat` call

## shmget Caveats

- `fork( )` After a `fork()` the child inherits the attached shared memory segments
- `exec( )` After an `exec()` all attached shared memory segments are detached (not destroyed)
- `exit( )` Upon `exit()` all attached shared memory segments are detached (not destroyed)

## Initializing Shared Memory

```
int shmid;
if ((shmid = shmget (key, sizeof (int),
    IPC_CREAT|IPC_EXCL|0666)) == -1) {
    perror ("shm");
    return 0;
}
```

## Opening Shared Memory

```
int shmidx;  
if ((shmidx = shmget (key, sizeof (int),  
    0)) == -1) {  
    perror ("shm");  
    return 0;  
}
```

## Attaching Shared Memory

```
int shmid;  
int *ptr;  
if ((ptr = shmat (shmid, (void *)0, 0))  
    == (int) -1) {  
    perror ("shm");  
    return 0;  
}
```

## Detaching Shared Memory

```
int shmid;  
int *ptr;  
if (shmdt (ptr) == -1) {  
    perror ("shm");  
    return 0;  
}
```

## Removing Shared Memory

```
shmctl(shmid, IPC_RMID, 0);
```

## Shared Memory for Related Processes

- ★ special mechanism provided by SysV that allows a shared memory region to be initiated and shared amongst related processes
- ★ also known as memory mapped shared memory
- ★ make use of the `/dev/zero` device in conjunction with the `mmap ( )` function
  - a unnamed memory are is created with `size` defined in `mmap`
  - the memory region is initialized to zero
  - multiple related processes can share this memory region as long as `MAP_SHARED` flag is set

## Shared Memory for Related Processes

```
caddr_t result;
int *ptr, fd;

if ((fd = open("/dev/zero", O_RDWR)) == -1)
    return 0;
result = mmap(0, sizeof (int),
    PROT_READ|PROT_WRITE, MAP_SHARED, fd, 0);
close (fd);

ptr = (int *)result;
```



Copyright © 2000-2001 by William Emmanuel S. Yu. This material may be distributed only subject to the terms and conditions set forth in the Open Content License, v1.0 or later (the latest version is presently available at <http://opencontent.org/opl.shtml>).