**sem\_init() not work in OS X**

Unnamed semaphores are not supported in OS X(in Mac OS X Yosemite(10.10),at least,sem\_init(),sem\_destroy(),sem\_getvalue() are marked ‘deprecated’ and generate compiler warnings as a result),thus you need to use named semaphores.

TEST:

int test = sem\_init()

test == 1

A better solution (these days) than named semaphores on OS X is Grand Central Dispatch's dispatch\_semaphore\_t. It works very much like the unnamed POSIX semaphores:

Initialize the semaphore:

#include <dispatch/dispatch.h>

dispatch\_semaphore\_t semaphore;

semaphore = dispatch\_semaphore\_create(1); // init with value of 1

Wait & post(signal):

dispatch\_semaphore\_wait(semaphore, DISPATCH\_TIME\_FOREVER);

...

dispatch\_semaphore\_signal(semaphore);

Destroy:

dispatch\_release(semaphore);

Or **use sem\_open() instead**:

creates a new POSIX semaphore or opens an existing semaphore.

The semaphore is identified by **name**:

POSIX semaphores allow processes and threads to synchronize their actions.

A semaphore is an integer whose value is never allowed to fall below zero.

Two operations can be performed on semaphores:

Increment the semaphore value by one **sem\_post(3)**

Decrement the semaphore value by one **sem\_wait(3)**,if the value of a semaphore is currently zero,then it will block until the value becomes greater.

POSIX semaphores comes in two forms:

**named semaphores**:

a named semaphore is identified by a name of the form **/somename** that is,a **null-terminated string** of up to NAME\_MAX characters consisting of **an initial slash** ,followed

by one or more characters,none of which are slashes. Two processes can operate on the same named semaphore by passing the same name to sem\_open();

The sem\_open() creates a new named semaphore or open an existing named semaphore,after it has been opened,it can be operated using sem\_post() and sem\_wait.

When a process finishes using the semaphore,it can use sem\_close() to close it.

When all processes have finished using it,it can be removed from the system using **sem\_unlink()**.

**unnamed semaphores(memory based semaphores)**:

 Does not have a name

 Placed in an area of memory shared between the threads of a process(thread-shared semaphore),e.g. a global variable.

Placed in a shared memory region(e.g. a system V shared memory segment created using **shmget()**, or a POSIX shared memory object built created using **shm\_open()**).

Unnamed semaphore must be initialized using **sem\_init()**,then operate using **sem\_post()** and **sem\_wait()**,

When the semaphore is no longer required,and **before the memory is deallocated**,it should be destroy using **sem\_destroy().**

**sem\_t \*sem\_open(const char \****name***, int** *oflag***,** **mode\_t** *mode***, unsigned int** *value***);**