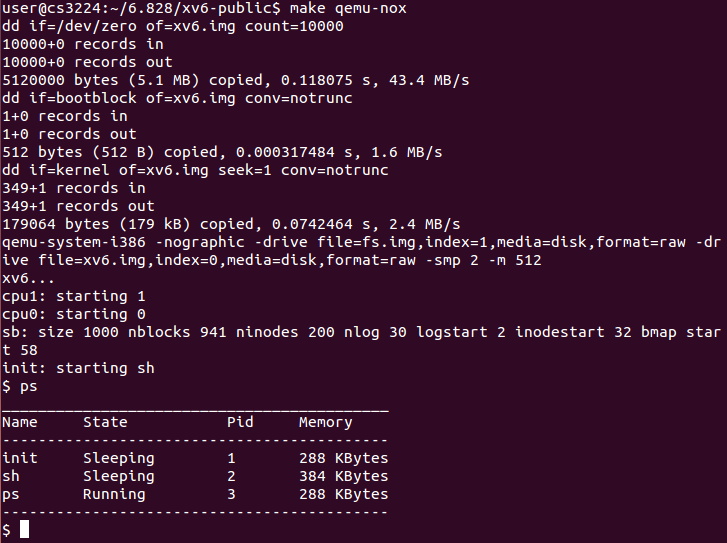
**PROCSTATE System Call**

**System Call:**

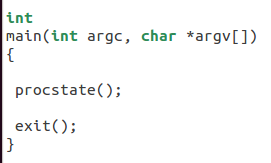
API of the operating system. Function that allows to access the properties of the CPU. Helps in raising the privilege level from the user mode to kernel mode.

* Enter the ”**ps**” command once the xv6 has started in the command prompt.

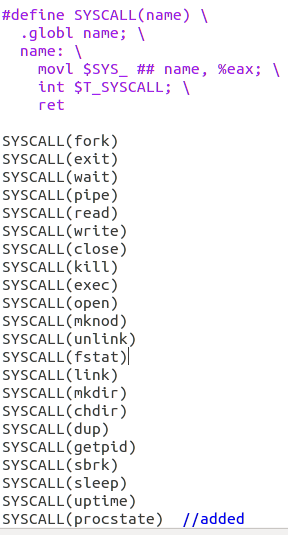


**Code Flow:**

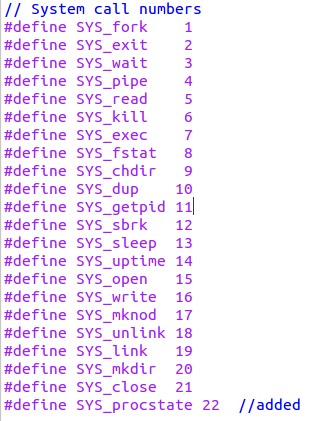
* Execution of the procstate() system call starts from “**ps.c**” file by calling the procstate() function.



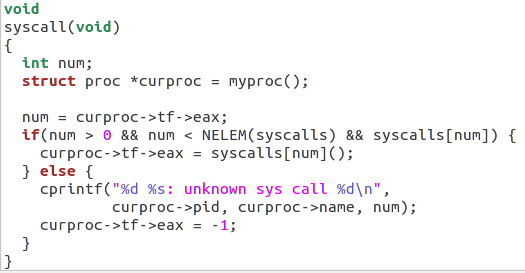
* When this procstate() system call is executed the control is transferred to **usys.S** filename.



* In the usys.S file #define SYSCALL definition will create (procstate) as a global variable and moves the SYS\_procstate to the eax register.
* Int T\_SYSCALL (Global interrupt) is called after moving the contents to the eax register. (Useful in interrupting the processor).
* The SYS\_procstate system call is defined in the **syscall.h** user file. It is given a unique number as 22.
* So, the value of 22 for the procstate system call is stored in the eax register.



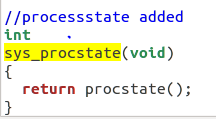
* After defining the system call and storing the system call number in the eax register. Now the control is transferred to the syscall() function defined in the **syscall.c** file.



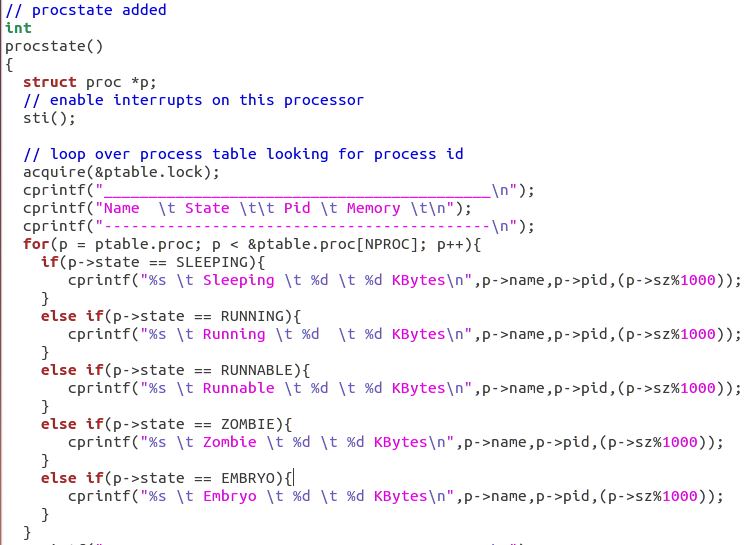
* In syscall.c file syscall(num) function is called with num as 22 for procstate system call and it puts the contents in the eax register and it is returned to the user. *(curproc->tf->eax=syscalls[num]())*
* Here the system call number is mapped to sys\_procstate function.



Now the control is transferred to sys\_procstate function in the **sysproc.c** file.

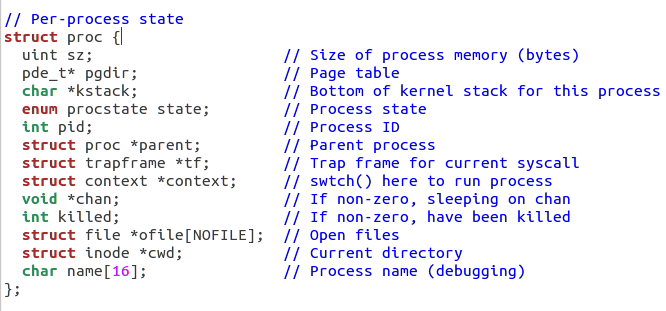


Implementation of the procstate() function is defined in the **proc.c** file.

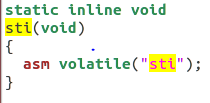




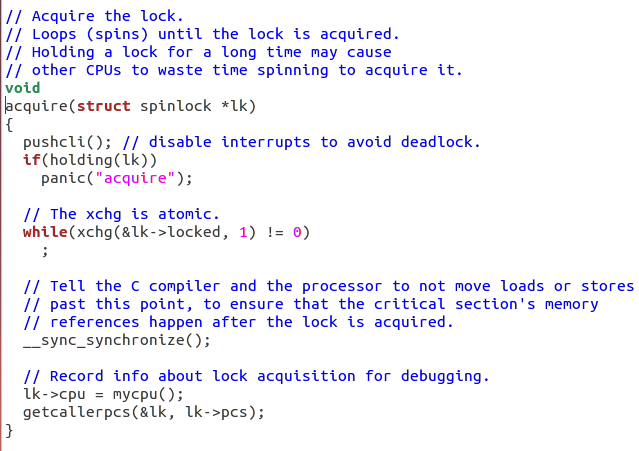
* In the procstate() function, initially a pointer \*p of type proc is created. Proc structure has the following attributes.



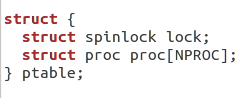
* Sti() function is used to enable interrupts on this processor.



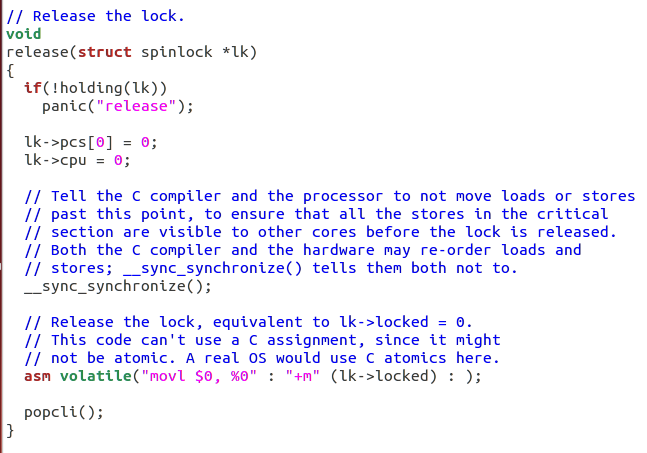
* Acquire(&ptable.lock) is used to acquire access to the process table and loop it to find the processes executing on the cpu.



* The process table is initialized as shown below.



* After acquiring the process table, we can loop it to display the number of processes running in the cpu.
* After displaying the processes, the process table lock is released.



Thus the flow of the system call (Procstate) can be represented as ps.c->usys.S->syscall.h->syscall.c->sysproc.c->proc.c.

**OUTPUT:**