**getpinfo**

1. git clone git://github.com/mit-pdos/xv6-public.git xv6-getpinfo
2. Open syscall.h and add the following

#define SYS\_getpinfo 22

1. Open defs.h and add the following in //proc.c section

int getpinfo (void);

1. open user.h file

add in systemcall section

int getpinfo (void);

1. open proc.c and add the following function:

|  |
| --- |
| int |
|  | getpinfo () |
|  | { |
|  | struct proc \*p; |
|  | //Enables interrupts on this processor. |
|  | sti(); |
|  |  |
|  | //Loop over process table looking for process with pid. |
|  | acquire(&ptable.lock); |
|  | cprintf("name \t pid \t state \t \n"); |
|  | for(p = ptable.proc; p < &ptable.proc[NPROC]; p++){ |
|  | if(p->state == SLEEPING) |
|  | cprintf("%s \t %d \t SLEEPING \t \n ", p->name,p->pid); |
|  | else if(p->state == RUNNING) |
|  | cprintf("%s \t %d \t RUNNING \t \n ", p->name,p->pid); |
|  | else if(p->state == RUNNABLE) |
|  | cprintf("%s \t %d \t RUNNABLE \t \n ", p->name,p->pid); |
|  | } |
|  | release(&ptable.lock); |
|  | return 22; |
|  | } |

1. open sysproc.c and add following function:

|  |
| --- |
| int |
|  | sys\_getpinfo(void) |
|  | { |
|  | return getpinfo(); |
|  | } |

1. open usys.S and add the following

SYSCALL(getpinfo)

1. open the **syscall.c** file and add the following

//Add this where the other system calls are defined in syscall.c

extern int sys\_getpinfo(void);

//Add this inside static int (\*syscalls[])(void)

[SYS\_getpinfo] sys\_getpinfo,

1. Create user program getpinfo.c

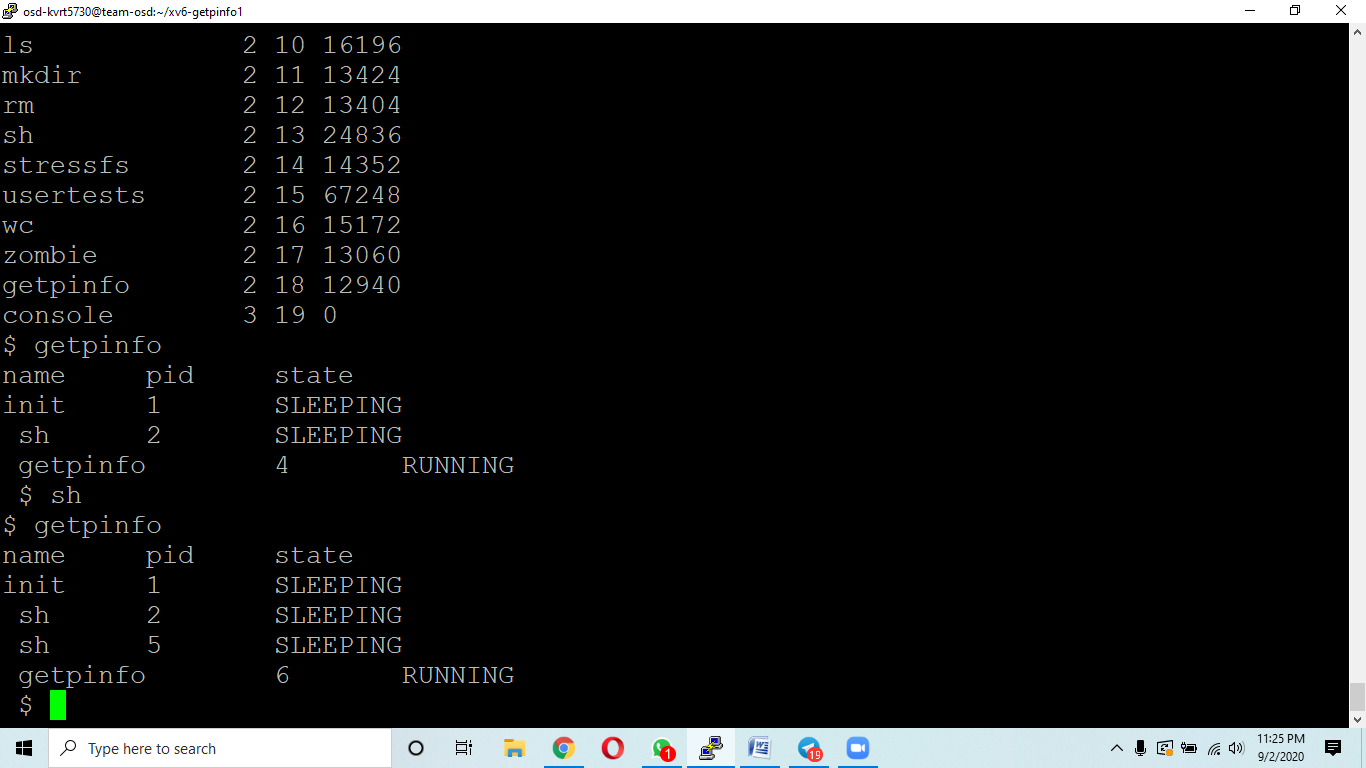
|  |
| --- |
| #include "types.h" |
|  | #include "stat.h" |
|  | #include "user.h" |
|  | #include "fcntl.h" |
|  |  |
|  | int main(void){ |
|  | getpinfo (); |
|  | exit(); |
|  | } |

1. nano Makefile

Add getpinfo\ to uprogs

Add getpinfo.c to Extra

1. make
2. make qemu-nox
3. $ getpinfo



1. $sh
2. $ getpinfo