Home Assignment - 2

Name - Rajeev Kumar
ID - 12341700

- **Step 1 –Make procinfo.h**

  File Name: procinfo.h
  Code Added:

```
struct proc_info{

  int pid;

  int ppid;

  int sz;

  char state[16];

  char name[16];

};
```

- **Step 2 – proc.c**

  File Name: `proc.c`
  Code Added:

```
int get_proc_info(int pid, struct proc_info *info){
  struct proc *p;
  acquire(&ptable.lock);
  for(p = ptable.proc; p < &ptable.proc[NPROC]; p++){
    if(p->pid == pid){
      info->pid = p->pid;
      info->ppid = (p->parent) ? p->parent->pid : 0;
      info->sz = p->sz;
      safestrcpy(info->name, p->name, sizeof(info->name));
     switch(p->state){
       case UNUSED: safestrcpy(info->state,"UNUSED", sizeof(info->state));
break;
```

```
      case EMBRYO: safestrcpy(info->state,"EMBRYO", sizeof(info->state));
break;
      case SLEEPING: safestrcpy(info->state,"SLEEPING",
sizeof(info->state)); break;
      case RUNNABLE: safestrcpy(info->state,"RUNNABLE",
sizeof(info->state));break;
      case RUNNING: safestrcpy(info->state,"RUNNING",
sizeof(info->state)); break;
      case ZOMBIE: safestrcpy(info->state, "ZOMBIE", sizeof(info->state));
break;
      default: safestrcpy(info->state, "UNKNOWN", sizeof(info->state));
break;
      }
     release(&ptable.lock);
     return 0;
    }
  }
  release(&ptable.lock);
  return -1;
}
```

- **Step 3 : sysproc.c**

  File Name: `sysproc.c`

  Code Added:

```
int
sys_getprocinfo(void){
  int pid;
  char *uaddr;
  struct proc_info info;
  if(argint(0, &pid) < 0 || argptr(1, &uaddr, sizeof(info)) < 0)
      return -1;
  if(get_proc_info(pid, &info) < 0)
      return -1;
  if(copyout(myproc()->pgdir, (uint)uaddr, (char*)&info, sizeof(info)) < 0)
      return -1;
  return 0;}
```

- **Step 4 : pinfo.c**

  File Name: `pinfo.c`
  Code Added:

```
#include "user.h"

int main(int argc, char *argv[]){
  if(argc != 2){
  printf(1,"Usage: pinfo <pid>\n");
  exit();
  }

  int pid = atoi(argv[1]);
  struct proc_info info;
  if(getprocinfo(pid, &info) < 0){
    printf(1,"Error: invalid PID %d\n", pid);
    exit();
  }

  printf(1,"PID: %d\n", info.pid);
  printf(1,"PPID: %d\n", info.ppid);
  printf(1,"Name: %s\n", info.name);
  printf(1,"State: %s\n", info.state);
  printf(1,"Size: %d\n", info.sz );
  exit();
}
```

- **Step 5 : testproc.c**

  File Name: `testproc.c`
  Code Added:

```
#include "user.h"
```

```
int main(void) {
  int i;
  int num_children = 5;
  for(i = 0; i < num_children; i++) {
    int pid = fork();
    if(pid < 0) {
      printf(1, "Fork failed\n");
      exit();
    }

    if(pid == 0) {
      printf(1, "Child process %d started with PID %d\n", i+1, getpid());
      while(1);
    }
  }

  exit();
}
```

- **Step 6 : proc.h**

  File Name: `proc.h`
  Code Added:

```
#include "procinfo.h"

int get_proc_info(int pid, struct proc_info *info);
```

- **Step 7 : syscall.h**

  File Name: `syscall.h`
  Code Added:

```
#define SYS_getprocinfo 26
```

- **Step 8 : syscall.c**

  File Name: `syscall.c`
  Code Added:

  ```
  extern int sys_getprocinfo(void);
  ```

  And inside `syscalls[ ]` table:

  ```
  [SYS_getprocinfo] sys_getprocinfo
  ```

- **Step 9 : user.h**

  File Name: `user.h`
  Code Added:

  ```
  #include "procinfo.h"
  #include "types.h"

  int getprocinfo(int pid, struct proc_info *info);
  ```

- **Step 10 : usys.S**

  File Name: `usys.S`
  Code Added:

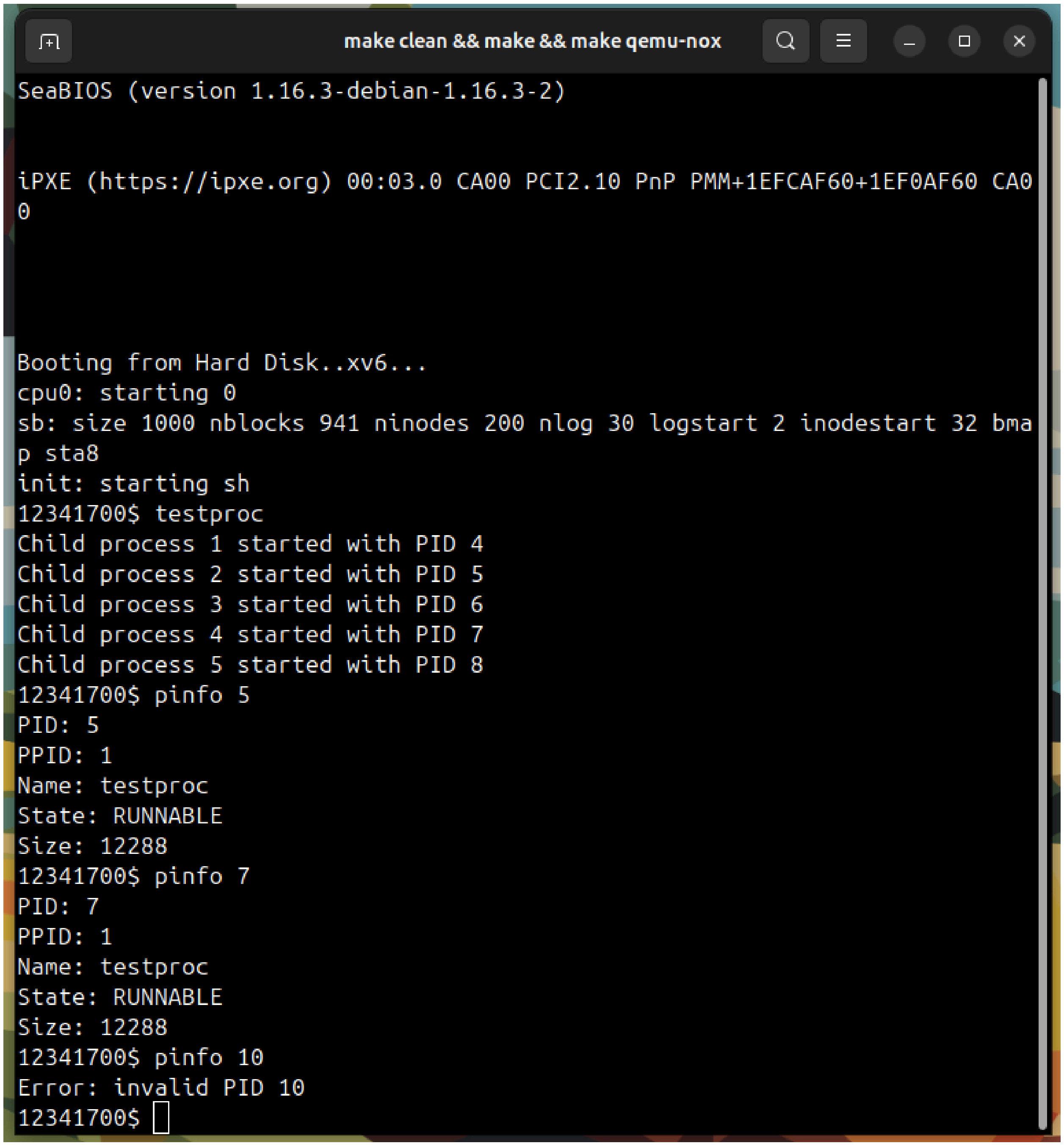
  ```
  SYSCALL(getprocinfo)
  ```

- **Step 11 : Makefile**

  File Name: `Makefile`
  Code Added:

  ```
  _pinfo\
  _testproc\
  ```

OUTPUT :



```
SeaBIOS (version 1.16.3-debian-1.16.3-2)


iPXE (https://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1EFCAF60+1EF0AF60 CA0
0





Booting from Hard Disk..xv6...
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bma
p sta8
init: starting sh
12341700$ testproc
Child process 1 started with PID 4
Child process 2 started with PID 5
Child process 3 started with PID 6
Child process 4 started with PID 7
Child process 5 started with PID 8
12341700$ pinfo 5
PID: 5
PPID: 1
Name: testproc
State: RUNNABLE
Size: 12288
12341700$ pinfo 7
PID: 7
PPID: 1
Name: testproc
State: RUNNABLE
Size: 12288
12341700$ pinfo 10
Error: invalid PID 10
12341700$ 
```