Project 2.B Detailed Solution

(1) Configuration: I have configured xv6-public on my Ubuntu Linux, but I do not have build-essentials and gawk, so install it using command

```
sudo apt-get install build-essential gawk qemu expect
```

(2) Kernel Programming: I mainly worked on 8 files step by step

a) syscall.h

i. #define SYS_getreadcount22

b) defs.h

i. intgetreadcount(void); //this is a declaration

c) user.h

i. intgetreadcount(void);//this direction will be directly interfaced with kernel users

d) sysproc.c

```
int sys_getreadcount(void)
{
   return  getreadcount();
}//it is directly interfaced to the system calling action
```

e) syscall.c

extern int sys_getreadcount(void); //linked to sysproc.c, thus enabling calling system process getreadcount () when needed

[SYS_getreadcount] sys_getreadcount, //Add the system call of getreadcount() to the system call table.

f) Usys.S

SYSCALL(getreadcount) //I imagine it like bind getreadcount to system call table, too

g) syscall.c:

```
extern int readcalledcount ;
int getreadcount()
{
   return readcalledcount;
}
```

The most important part is the ***extern int readcalledcount*** ; readcalledcount is defined in the same file of int sysread(void), which will be called once a user call read() system call indirectly or directly, so I define a global variable ***readcalledcount*** to record the call of this system call. int getreadcount() is very easy, it is just returning a value.

h) create getreadcount.c

This will be the kernel level application, and the implementation is easy.

```
int main(int argc, char *argv[])
{
      getreadcount(); //just call the system call
      exit();
}
```

i) make some change in the make file

```
UPROGS = _getreadcount\
EXTRA=getreadcount.c
```