

**NAME:** Dev Sandip Shah

**ROLL NO:** 200123074

**Exercise 1.** Create a system call `int sys_draw(void *buf, uint size)`, which copies an ASCII art image (Use a buffer of any picture, google it for more information) of a picture to a user-supplied buffer, provided that the buffer is large enough. You are welcome to use an ASCII art generator, or draw your own by hand. (One such example is given with the assignment) If the buffer is too small, or not valid, return a negative value. If the call succeeds, return the number of bytes copied. You may find it helpful to review how other system calls are implemented and compiled into the kernel, such as `read`.

**Answer:**

An operating system supports two modes; the kernel mode and the user mode. When a program in user mode requires access to RAM or a hardware resource, it must ask the kernel to provide access to that particular resource. This is done via a system call. When a program makes a system call, the mode is switched from user mode to kernel mode. In order to define our own system, call in xv6, changes need to be made to 5 files. Namely, these files are as follows.

- (i) `syscall.h`
- (ii) `syscall.c`
- (iii) `sysproc.c`
- (iv) `usys.S`
- (v) `user.h`

We would start the procedure by editing `syscall.h` in which a number is given to every system call. This file already contains 21 system calls. In order to add the custom system call, the following line needs to be added to this file.

```
| 23 #define SYS Draw 22 |
```

Next, we need to add a pointer to the system call in the `syscall.c` file. This file contains an array of function pointers which uses the above-defined numbers (indexes) as pointers to system calls which are defined in a different location. In order to add our custom system call, add the following line to this file.

```
| 130 [SYS_Draw] sys_Draw,
```

When the system call with number 22 is called by a user program, the function pointer `sys_wolfie` which has the index `SYS_wolfie` or 22 will call the system call function. Hence, our next objective is to

implement a system call function. But we do not implement the system call function in syscall.c. We only add a prototype as shown below.

```
106 extern int sys_Draw(void);
```

We implement the system call function in sysproc.c.

```
93 int
94 sys_Draw(void)
95 {
96     void* buf;
97     uint size;
98
99     argptr(0, (void*)&buf, sizeof(buf));
100    argptr(1, (void*)&size, sizeof(size));
101
102    char text[] = "Say Hi to Draw: \n\
103                  *\n\
104                  ***\n\
105                  *****\n\
106                  *****\n\
107                  *****\n\
108                  ***\n\
109                  *\n";
110    if(sizeof(text)>size)
111        return -1;
112
113    strncpy((char *)buf, text, size);
114    return sizeof(text);
115 }
```

Now, to add an interface for a user program to call the system call, we add

```
| 32 SYSCALL(Draw)
```

to usys.S and

```
26 int Draw(void*, uint);
```

to user.h

**Exercise 2.** Write a user-level application, called Drawtest.c, that gets the image from the kernel, and prints it to the console. When the OS runs, your program's binary should be included in fs.img and listed if someone runs ls at the xv6 shell's command prompt. Study Makefile to figure out how to compile a usermode program and add it to fs.img

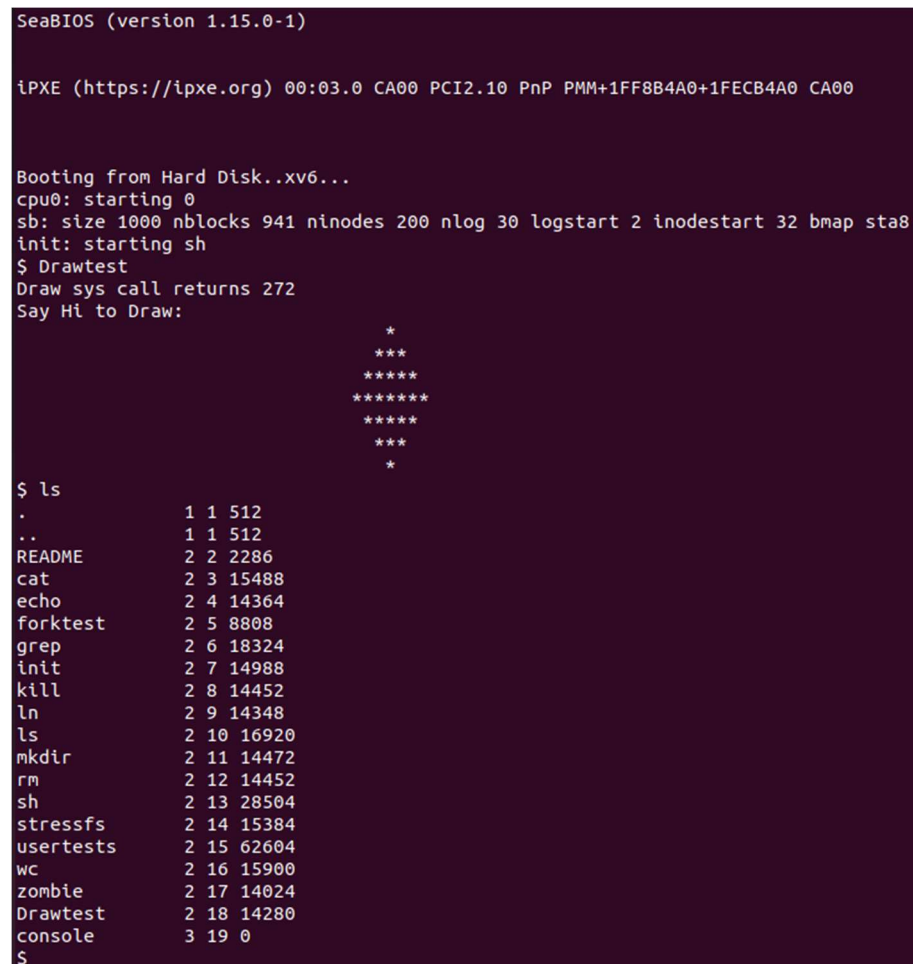
**Answer:**

Now the only task left is to add a user program to call the system call that we just made above. For this, I made a file Drawtest.c inside xv6 folder and wrote the following code in it.

```
1 #include "types.h"
2 #include "stat.h"
3 #include "user.h"
4
5 int
6 main(void)
7 {
8     static char buf[2000];
9     printf(1, "Draw sys call returns %d\n", Draw((void*) buf, 2000));
10
11     printf(1, "%s", buf);
12     exit();
13 }
```

After this, I added this file to the Makefile under UPROGS and EXTRA.

Then I used “make clean”, “make”, “make qemu-nox” and then entered Drawtest to get the following output. Also, my program is also listed when I use the ls command. Screenshot is attached below



```
SeaBIOS (version 1.15.0-1)

ipXE (https://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8B4A0+1FECB4A0 CA00

Booting from Hard Disk..xv6...
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap sta8
init: starting sh
$ Drawtest
Draw sys call returns 272
Say Hi to Draw:

          *
        ***
       *****
      *****
     *****
    ***
   *

$ ls
.          1 1 512
..         1 1 512
README    2 2 2286
cat        2 3 15488
echo       2 4 14364
forktest  2 5 8808
grep       2 6 18324
init       2 7 14988
kill       2 8 14452
ln         2 9 14348
ls         2 10 16920
mkdir     2 11 14472
rm         2 12 14452
sh         2 13 28504
stressfs  2 14 15384
usertests  2 15 62604
wc         2 16 15900
zombie    2 17 14024
Drawtest  2 18 14280
console   3 19 0
$
```