

CS 344: Operating Systems Lab

ASSIGNMENT 0B

Name: Pragyaa Banerjee

Roll no.: 200123080

Question 1

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 implement the system call function in sysproc.c



```
19
20 // return how many clock tick interrupts have occurred
21 // since start.
22 int
23 sys_tick(void)
24 {
25     uint ticks;
26
27     acquire(&tickslock);
28     ticks = Ticks;
29     release(&tickslock);
30     return ticks;
31 }
32
33 int
34 sys_draw(void)
35 {
36     char buf[1024];
37     uint size;
38
39     argptr(0, (void*)buf, sizeof(buf));
40     argptr(1, (void*)&size, sizeof(size));
41
42     char text[] = "Hello xv6 is fine! id= %d\n";
43     int i;
44     for (i = 0; i < size; i++)
45         buf[i] = text[i % sizeof(text)];
46
47     if (size > 0)
48         printf(buf, size);
49
50     if (size > 0)
51         return size;
52     return 0;
53 }
54
55 int
56 sys_test(void)
57 {
58     return 0;
59 }
60
61 int
62 sys_exit(int code)
63 {
64     return code;
65 }
66
67 int
68 sys_wait(int pid)
69 {
70     return 0;
71 }
72
73 int
74 sys_kill(int pid)
75 {
76     return 0;
77 }
78
79 int
80 sys_chdir(char *path)
81 {
82     return 0;
83 }
84
85 int
86 sys_mkdir(char *path, mode_t mode)
87 {
88     return 0;
89 }
90
91 int
92 sys_rmdir(char *path)
93 {
94     return 0;
95 }
96
97 int
98 sys_open(char *path, int flags)
99 {
100     return 0;
101 }
102
103 int
104 sys_close(int fd)
105 {
106     return 0;
107 }
108
109 int
110 sys_read(int fd, void *buf, int size)
111 {
112     return 0;
113 }
114
115 int
116 sys_write(int fd, void *buf, int size)
117 {
118     return 0;
119 }
120
121 int
122 sys_fstat(int fd, struct stat *stat)
123 {
124     return 0;
125 }
```

Question 2

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.

