Waled Salem
CSE 460
Lab 5
Score: 20/20

1. Message Queues

Code:
msg1.cpp

```cpp
//modified msg1.cpp
/* Here's the receiver program. */

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

#define MAX_TEXT 512
struct my_msg_st {
   long int my_msg_type;
   char some_text[BUFSIZ];
};

int main()
{
   int running = 1;
   int msgid, msgid1;
   struct my_msg_st some_data;
   long int msg_to_receive = 0;
   char buffer[BUFSIZ];
   /* First, we set up the message queue. */
   msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
   msgid1 = msgget((key_t)1234, 0666 | IPC_CREAT);
   if (msgid == -1) {
       fprintf(stderr, "msgget failed with error: %d\n", errno);
       exit(EXIT_FAILURE);
   }
   /* Then the messages are retrieved from the queue, until an end message is encountered.
   Lastly, the message queue is deleted. */
    while(running) {
        if (msgrcv(msgid, (void *)&some_data, BUFSIZ,
        msg_to_receive, 0) == -1) {
            fprintf(stderr, "msgrcv failed with error: %d\n", errno);
            exit(EXIT_FAILURE);
        }
        printf("You wrote: %s", some_data.some_text);
        if (strncmp(some_data.some_text, "end", 3) == 0) {
            running = 0;
       }
       else{
       printf("Enter some text: ");
       fgets(buffer, BUFSIZ, stdin);
       some_data.my_msg_type = 1;
       strcpy(some_data.some_text, buffer);
       if (msgsnd(msgid1, (void *)&some_data, MAX_TEXT, 0) == -1) {
           exit(EXIT_FAILURE);
       }
       if (strncmp(buffer, "end", 3) == 0) {
           running = 0;
```

Waled Salem
CSE 460
Lab 5
Score: 20/20

```
51              exit(EXIT_FAILURE);
52          }
53          if (strncmp(buffer, "end", 3) == 0) {
54              running = 0;
55          }
56      }
57      }
58      if (msgctl(msgid, IPC_RMID, 0) == -1) {
59          fprintf(stderr, "msgctl(IPC_RMID) failed\n");
60          exit(EXIT_FAILURE);
61      }
62      exit(EXIT_SUCCESS);
63  }
```

msg2.cpp

Waled Salem
CSE 460
Lab 5
Score: 20/20

```c
//modified msg2.cpp
/* The sender program is very similar to msg1.cpp. In the main set up, delete the
msg_to_receive declaration and replace it with buffer[BUFSIZ], remove the message
queue delete and make the following changes to the running loop.
We now have a call to msgsnd to send the entered text to the queue. */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/msg.h>

#define MAX_TEXT 512
struct my_msg_st {
    long int my_msg_type;
    char some_text[MAX_TEXT];
};

int main()
{
    int running = 1;
    struct my_msg_st some_data;
    int msgid, msgid1;
    char buffer[BUFSIZ];
    long int msg_to_receive = 0;
    msgid = msgget((key_t)1234, 0666 | IPC_CREAT);
    msgid1 = msgget((key_t)1234, 0666 | IPC_CREAT);
    if (msgid == -1) {
        fprintf(stderr, "msgget failed with error: %d\n", errno);
        exit(EXIT_FAILURE);
    }
    while(running) {
        printf("Enter some text: ");
        fgets(buffer, BUFSIZ, stdin);
        some_data.my_msg_type = 1;
        strcpy(some_data.some_text, buffer);
        if (msgsnd(msgid, (void *)&some_data, MAX_TEXT, 0) == -1) {
            exit(EXIT_FAILURE);
        }else{
            if (msgrcv(msgid1, (void *)&some_data, BUFSIZ,
            msg_to_receive, 0) == -1) {
                fprintf(stderr, "msgrcv failed with error: %d\n", errno);
                exit(EXIT_FAILURE);
            }
            printf("You wrote: %s", some_data.some_text);
            if (strncmp(some_data.some_text, "end", 3) == 0) {
                running = 0;
            }
        }
        if (strncmp(buffer, "end", 3) == 0) {
```

```c
            if (strncmp(some_data.some_text, "end", 3) == 0) {
                running = 0;
            }
        }
        if (strncmp(buffer, "end", 3) == 0) {
            running = 0;
        }
    }
    if (msgctl(msgid, IPC_RMID, 0) == -1) {
        fprintf(stderr, "msgctl(IPC_RMID) failed\n");
        exit(EXIT_FAILURE);
    }
    exit(EXIT_SUCCESS);
}
```

Waled Salem
CSE 460
Lab 5
Score: 20/20

Output:

```
[004893625@csusb.edu@jb359-29 Lab5]$ ./msg1
Hello
World
^C
[004893625@csusb.edu@jb359-29 Lab5]$ ./msg2
Enter some text: hello
You wrote: hello
Enter some text: world
You wrote: world
Enter some text: Great
You wrote: Great
Enter some text: ^C
[004893625@csusb.edu@jb359-29 Lab5]$ ./msg1
^C
[004893625@csusb.edu@jb359-29 Lab5]$
```

2. IPC Status Commands

Output:

```
[004893625@csusb.edu@jb359-29 Lab5]$ ipcs -s

------ Semaphore Arrays --------
key        semid      owner      perms      nsems

[004893625@csusb.edu@jb359-29 Lab5]$ ipcs -m

------ Shared Memory Segments --------
key        shmid      owner      perms      bytes      nattch     status
0x00000000 19300352   004893625@ 600        16777216   2          dest
0x00000000 19398657   004893625@ 600        524288     2          dest
0x00000000 19529730   004893625@ 600        7802880    2          dest
0x00000000 19857411   004893625@ 600        2887680    2          dest
0x00000000 19496964   004893625@ 600        7802880    2          dest
0x00000000 19955718   004893625@ 600        524288     2          dest
0x00000000 20086791   004893625@ 600        524288     2          dest
0x00000000 20217864   004893625@ 600        16777216   2          dest
0x00000000 20545545   004893625@ 600        458752     2          dest
0x00000000 20578314   004893625@ 600        16384      2          dest
0x00000000 20512779   004893625@ 600        471040     2          dest
0x00000000 20611084   004893625@ 600        16384      2          dest
0x00000000 20643853   004893625@ 600        2887680    2          dest

[004893625@csusb.edu@jb359-29 Lab5]$ ipcs -q

------ Message Queues --------
key        msqid      owner      perms      used-bytes   messages
```

$ipcs -s: identifies the processes that use semaphores

Waled Salem
CSE 460
Lab 5
Score: 20/20

$ipcs -m: identifies shared segments of memory
$ipcs -q: identifies IPC semaphores with messages in its queue.

3. Study of XV6

Output:

```
(gdb) target remote: 27030
Remote debugging using : 27030
warning: Remote gdbserver does not support determining executable automatically.
RHEL <=6.8 and <=7.2 versions of gdbserver do not support such automatic executable detection.
The following versions of gdbserver support it:
- Upstream version of gdbserver (unsupported) 7.10 or later
- Red Hat Developer Toolset (DTS) version of gdbserver from DTS 4.0 or later (only on x86_64)
- RHEL-7.3 versions of gdbserver (on any architecture)
warning: No executable has been specified and target does not support
determining executable automatically.  Try using the "file" command.
0x0000fff0 in ?? ()
(gdb) file kernel
A program is being debugged already.
Are you sure you want to change the file? (y or n) y
Reading symbols from kernel...done.
(gdb) break swtch
Breakpoint 1 at 0x8010469b: file swtch.S, line 11.
(gdb) continue
Continuing.

Thread 1 hit Breakpoint 1, swtch () at swtch.S:11
11        movl 4(%esp), %eax
(gdb) step
12        movl 8(%esp), %edx
(gdb) step
15        pushl %ebp
(gdb) step
swtch () at swtch.S:16
16        pushl %ebx
(gdb) step
swtch () at swtch.S:17
17        pushl %esi
(gdb) step
swtch () at swtch.S:18
18        pushl %edi
(gdb) step
swtch () at swtch.S:21
21        movl %esp, (%eax)
(gdb) step
22        movl %edx, %esp
(gdb) step
swtch () at swtch.S:25
25        popl %edi
(gdb) step
swtch () at swtch.S:26
26        popl %esi
(gdb) step
swtch () at swtch.S:27
27        popl %ebx
(gdb) step
swtch () at swtch.S:28
28        popl %ebp
(gdb) step
```

Waled Salem
CSE 460
Lab 5
Score: 20/20

```
27        popl %ebx
(gdb) step
swtch () at swtch.S:28
28        popl %ebp
(gdb) step
swtch () at swtch.S:29
29        ret
(gdb) step
forkret () at proc.c:398
398     {
(gdb) step
forkret () at proc.c:401
401       release(&ptable.lock);
(gdb) step
release (lk=0x80112d20 <ptable>) at spinlock.c:48
48      {
(gdb) step
49        if(!holding(lk))
(gdb) continue
Continuing.

Thread 1 hit Breakpoint 1, swtch () at swtch.S:11
11        movl 4(%esp), %eax
(gdb) clear
Deleted breakpoint 1
(gdb) break exec
Breakpoint 2 at 0x80100a10: file exec.c, line 12.
(gdb) continue
Continuing.
[Switching to Thread 2]

Thread 2 hit Breakpoint 2, exec (path=0x1c "/init", argv=0x8dfffed0)
    at exec.c:12
12      {
(gdb) continue
Continuing.

Thread 2 hit Breakpoint 2, exec (path=0x816 "sh", argv=0x8dffeed0) at exec.c:12
12      {
(gdb) continue
Continuing.
^[[A
^[[A
[Switching to Thread 1]

Thread 1 hit Breakpoint 2, exec (path=0x1880 "ls", argv=0x8dfbeed0)
    at exec.c:12
12      {
(gdb) continue
Continuing.

Thread 1 hit Breakpoint 2, exec (path=0x1880 "ls", argv=0x8df23ed0)
    at exec.c:12
```

Waled Salem
CSE 460
Lab 5
Score: 20/20

```
[Switching to Thread 1]

Thread 1 hit Breakpoint 2, exec (path=0x1880 "ls", argv=0x8dfbeed0)
    at exec.c:12
12      {
(gdb) continue
Continuing.

Thread 1 hit Breakpoint 2, exec (path=0x1880 "ls", argv=0x8df23ed0)
    at exec.c:12
12      {
(gdb) continue
Continuing.

Thread 1 hit Breakpoint 2, exec (path=0x1880 "ls", argv=0x8dfc6ed0)
    at exec.c:12
12      {
(gdb) print argv[0]
$1 = 0x1880 "ls"
(gdb) print argv[1]
$2 = 0x0
(gdb) print argv[2]
$3 = 0x0
(gdb) backtrace
#0  exec (path=0x1880 "ls", argv=0x8dfc6ed0) at exec.c:12
#1  0x801053a0 in sys_exec () at sysfile.c:420
#2  0x80104879 in syscall () at syscall.c:139
#3  0x80105835 in trap (tf=0x8dfc6fb4) at trap.c:43
#4  0x8010564f in alltraps () at trapasm.S:20
#5  0x8dfc6fb4 in ?? ()
Backtrace stopped: previous frame inner to this frame (corrupt stack?)
(gdb) up
#1  0x801053a0 in sys_exec () at sysfile.c:420
420       return exec(path, argv);
(gdb)
```

A breakpoint is made at swtch, then I proceeded to step through each line of code. The breakpoint was eventually cleared.

Waled Salem
CSE 460
Lab 5
Score: 20/20

```
- Red Hat Developer Toolset (DTS) version of gdbserver from DTS 4.0 or later (only on x86_64)
- RHEL-7.3 versions of gdbserver (on any architecture)
warning: No executable has been specified and target does not support
determining executable automatically.  Try using the "file" command.
0x0000fff0 in ?? ()
(gdb) file kernel
A program is being debugged already.
Are you sure you want to change the file? (y or n) y
Reading symbols from kernel...done.
(gdb) break scheduler
Breakpoint 1 at 0x80103ab0: file proc.c, line 324.
(gdb) step
Cannot find bounds of current function
(gdb) step
Cannot find bounds of current function
(gdb) continue
Continuing.
[Switching to Thread 2]

Thread 2 hit Breakpoint 1, scheduler () at proc.c:324
324     {
(gdb) step
326       struct cpu *c = mycpu();
(gdb) step
mycpu () at proc.c:42
42        if(readeflags()&FL_IF)
(gdb) step
readeflags () at x86.h:98
98        asm volatile("pushfl; popl %0" : "=r" (eflags));
(gdb) step
mycpu () at proc.c:42
42        if(readeflags()&FL_IF)
(gdb) step
45        apicid = lapicid();
(gdb) step
lapicid () at lapic.c:103
103       if (!lapic)
(gdb) step
102     {
(gdb) step
lapicid () at lapic.c:103
103       if (!lapic)
(gdb) step
105       return lapic[ID] >> 24;
(gdb) step
106     }
(gdb) step
mycpu () at proc.c:48
48        for (i = 0; i < ncpu; ++i) {
(gdb) step
49          if (cpus[i].apicid == apicid)
(gdb) step
48        for (i = 0; i < ncpu; ++i) {
(gdb) step
49          if (cpus[i].apicid == apicid)
```

A breakpoint was placed at scheduler and I proceeded to step through each line of code. I then cleared the breakpoint.

I believe i deserve a **score of 20/20** for this lab since i completed all required tasks.