Operating System Concepts

COP4610.02

Mini Project 4

Noah Baldwin

Cody Carroll

Paul Teleweck

**Work Breakdown:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Names** | **Code** | **Report** | **Documentation** | **Presentation** |
| Noah Baldwin | 33.33% | 33.33% | 33.33% | 33.33% |
| Cody Carroll | 33.33% | 33.33% | 33.33% | 33.33% |
| Paul Teleweck | 33.33% | 33.33% | 33.33% | 33.33% |

**Abstract:**

In this project, you'll be adding real kernel threads to xv6. First, define a new system call to create a kernel thread, called **clone()**, as well as one to wait for a thread called **join()**. Then, use **clone()** to build a little thread library, with a **thread\_create()** call and **lock\_aquire()** and **lock\_release()** functions.

**Changes Made**:

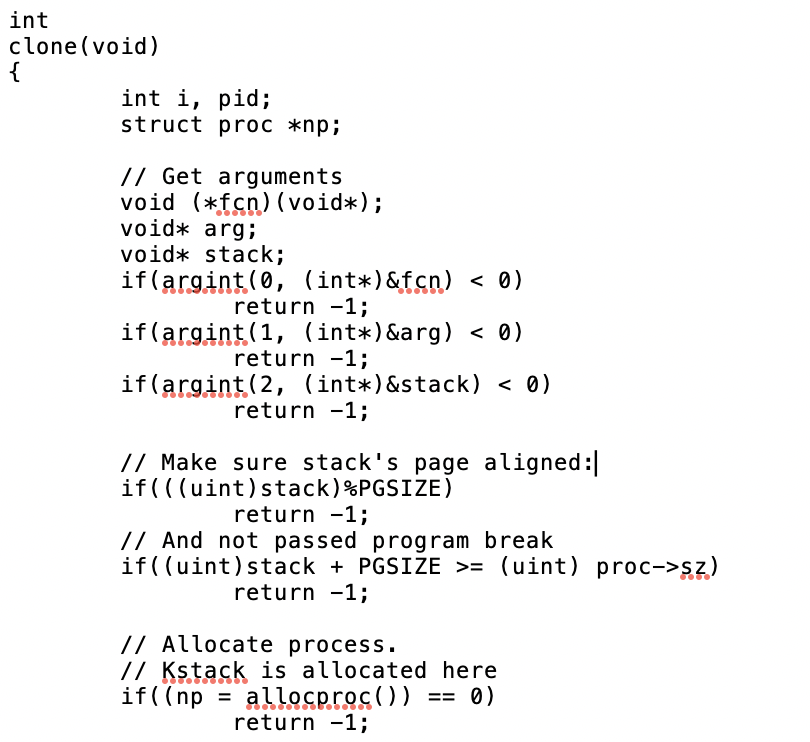
Proc.c

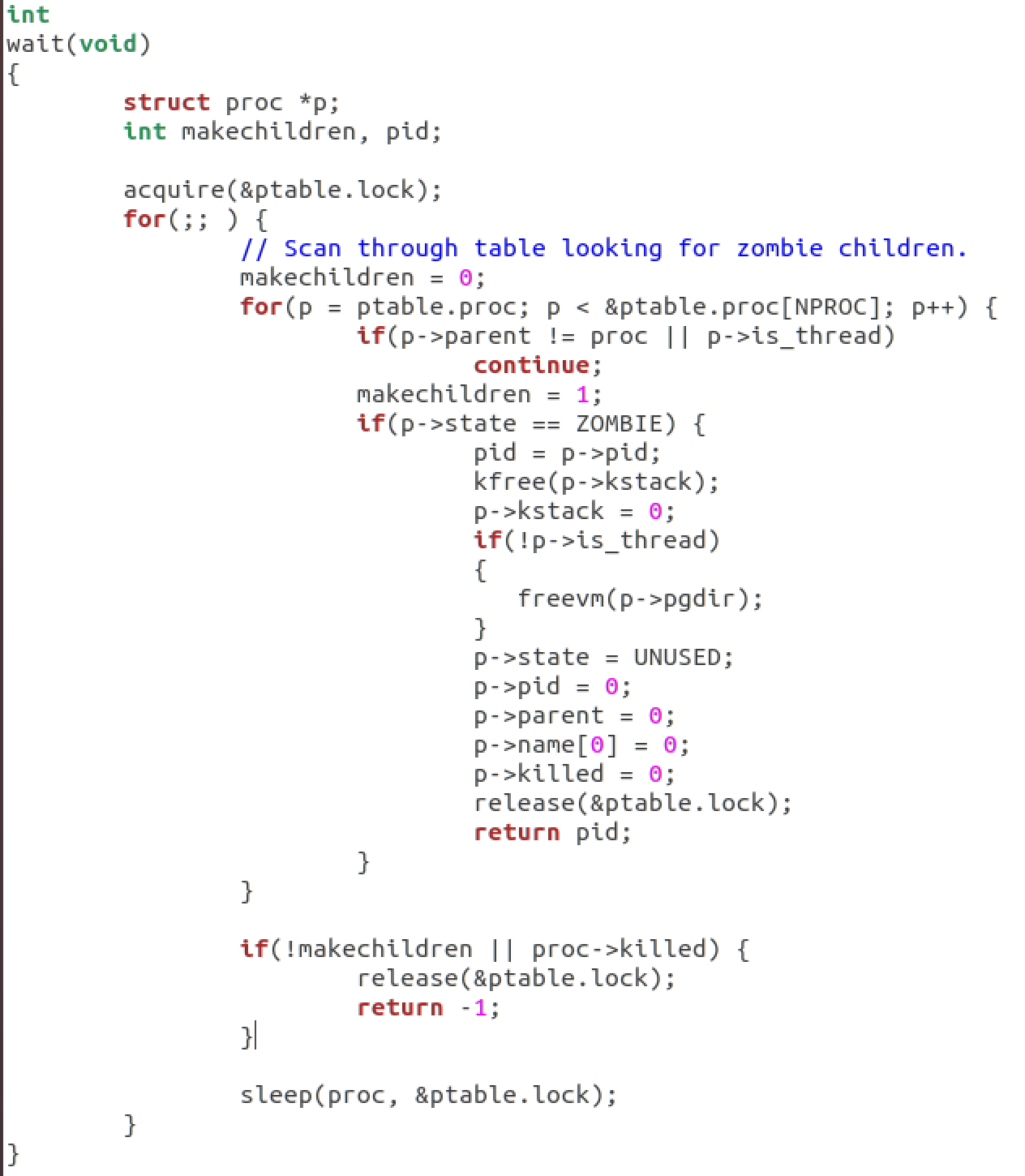
The structure **lock**: A lock or Mutex is a mechanism that is created for enforcing limits on access to a resource in an environment where there are many [threads of execution](https://en.wikipedia.org/wiki/Thread_(computer_science)).   
This is to alleviate CPU usage when there are lots of threads or processes active at the same time.

Define **clone()** system call which creates a new kernel thread which shares the calling process' address space.

Define **wait()** system call which should wait for a child process that does not share the address space with the process.

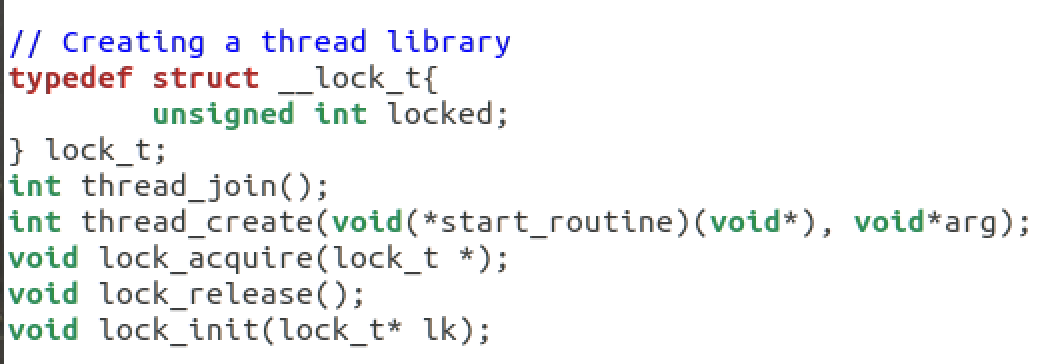
The **join()** sys call. This call waits for a child thread that shares the address space with the calling process. It returns the PID of waited for child or -1 if none.

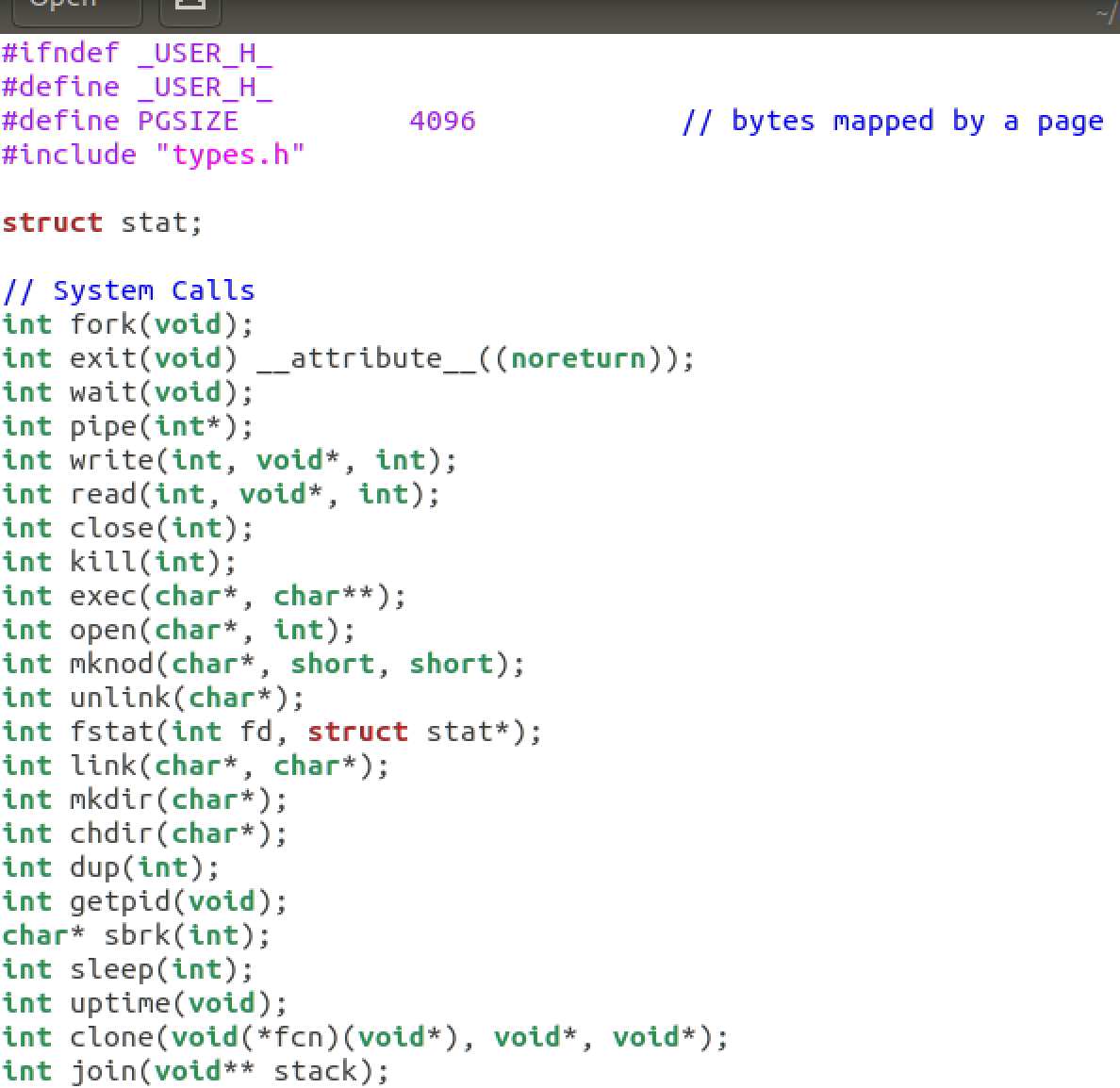




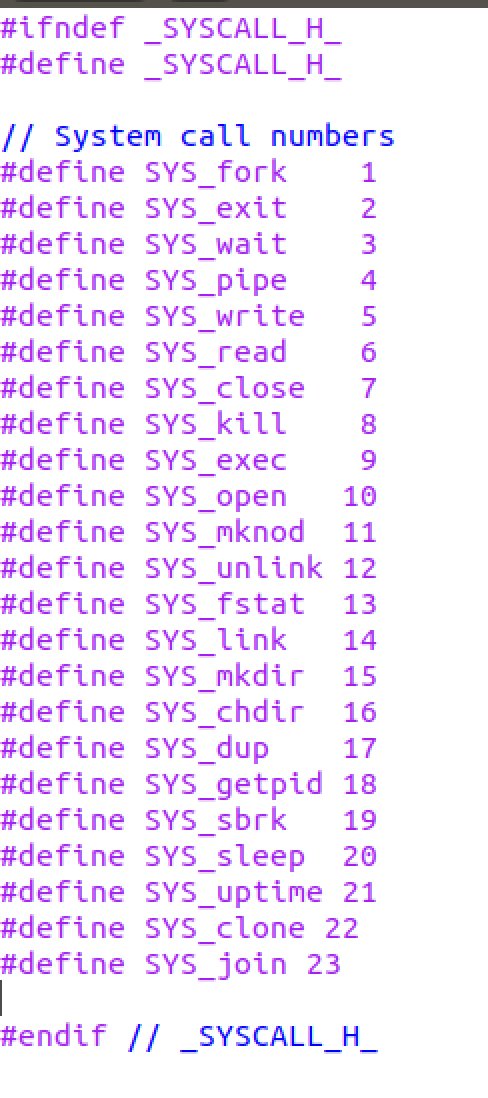


User.h  
  
The **clone()**, **join()**, **lock()**, and **thread\_create()** functions were added to the user.h file. This creates functionality within the user files.

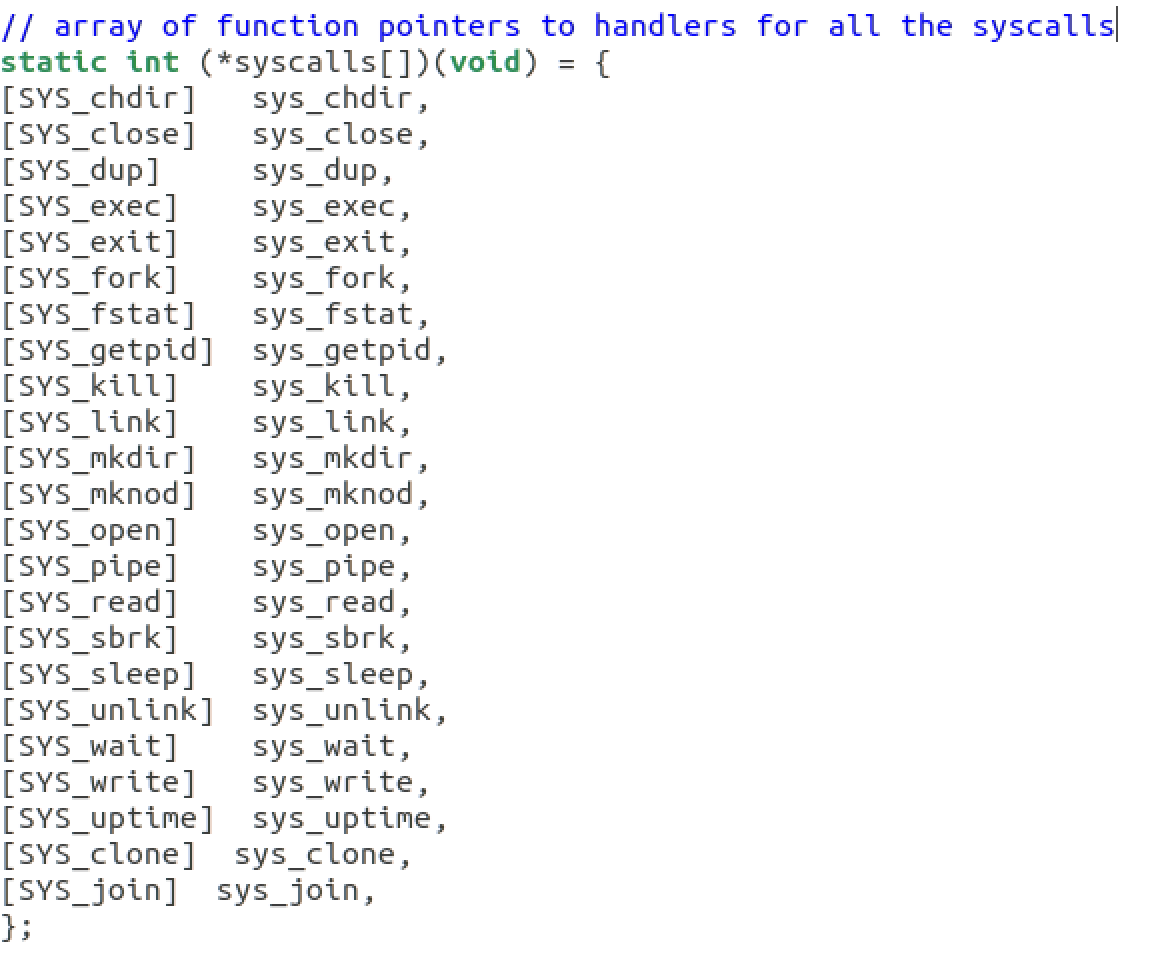




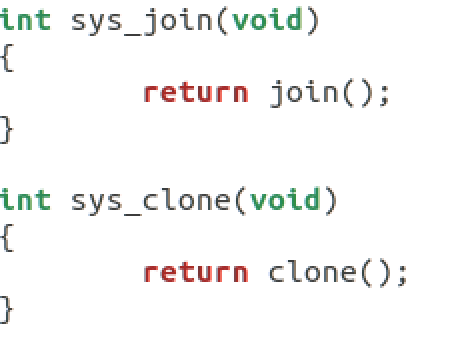
Syscall.h  
  
System calls are defined for **clone()** and **join()**.



Syscall.c  
  
System calls are added for **clone()** and **join()**.



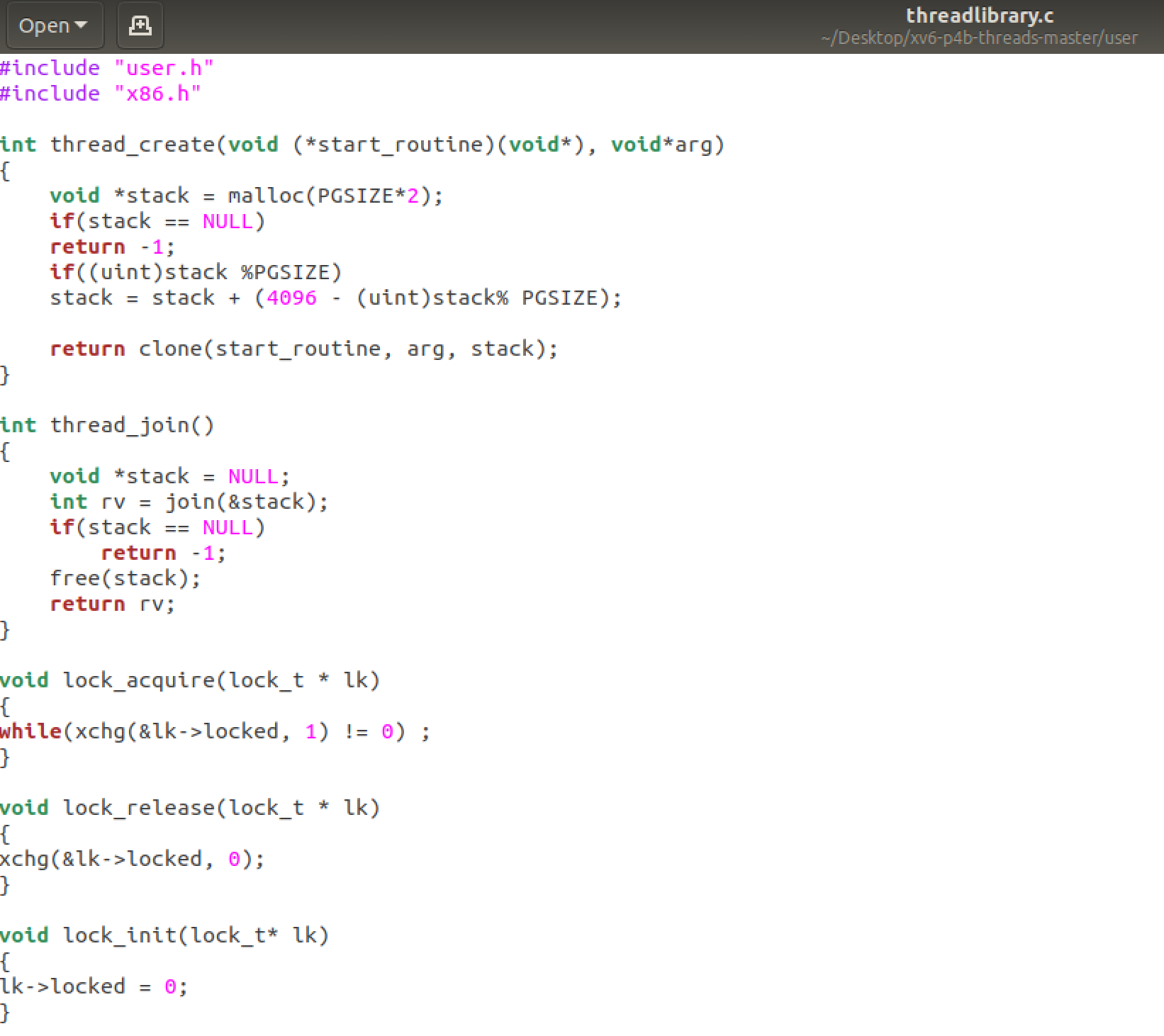
Sysproc.c  
  
Created **sys\_clone**, and **sys\_join** to initialize the system calls.



Threadlibrary.c

The updated lock structure prevents threads from overwriting performance critical tasks. The processes within create a thread library using the **Thread\_create()** routine. This routine should call **malloc()** to create a new user stack, use **clone()** to create the child thread and get it running.

The **Thread\_join()** call is also used, which calls the underlying **join()** system call, frees the user stack, and then returns.



Output

