Trenton Bergdahl - 11768322

11/15/2023

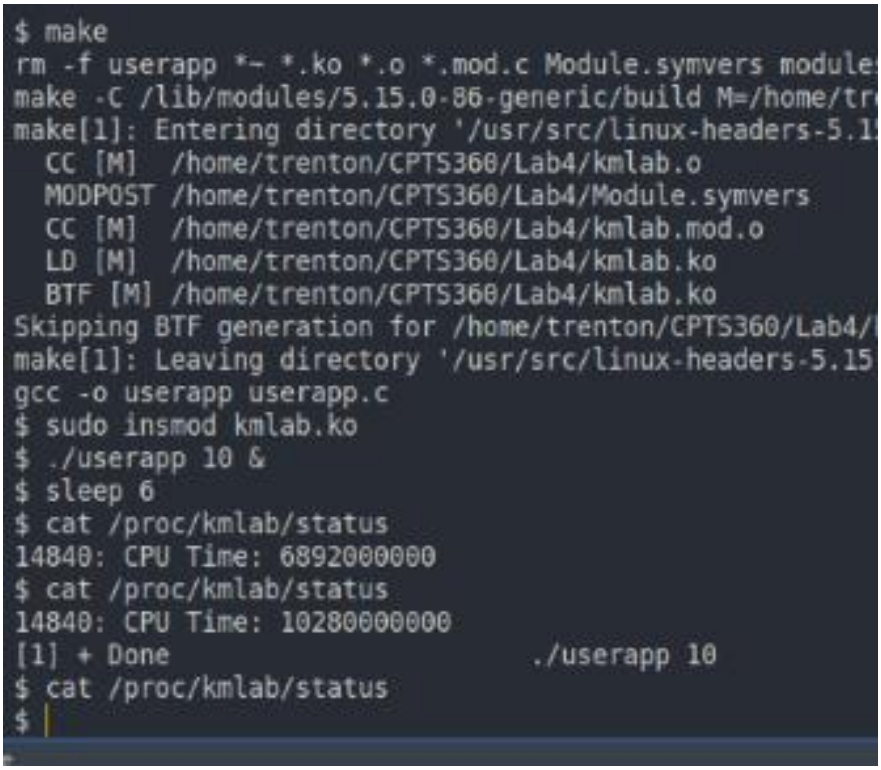
Lab 4 Report

The implementation of my module was to refactor proc read and write. Proc read was refactored to display all process entries in the process list to the /proc/kmlab/status file. Proc write is designed to accept a string pid and convert it to an integer, then use the pid to get the CPU time of the process, and store both in a list entry. There is also a timer configured that will schedule a function called work_function that will traverse the list and update the list based off process completion. If a process isn't complete the work function will update its CPU time and if it is complete it will remove the process from the list. Screenshots of the outputs are provided below.

Output with multiple processes:

After inserting the two processes, they are both in the list, as shown by the first cat command. Then, process 1 completes, shown by the [1] + Done. The next call to cat only shows the second process, which makes sense, as the work function should have removed the first process from the list. Then the 2nd process finishes, and the third and final call to cat reflects this as the process is complete, so there should be no processes left in the list.

Output with single processes:



After inserting the process, a similar thing happens. It runs and is in the linked list, shown by the first call to cat. Next it completes ([1] + Done) and while the process is still in the list, the kernel timer hasn't interrupted it yet to remove the process. On the final call to cat, you can see the timer has interrupted the module and removed the process from the list since it is complete.