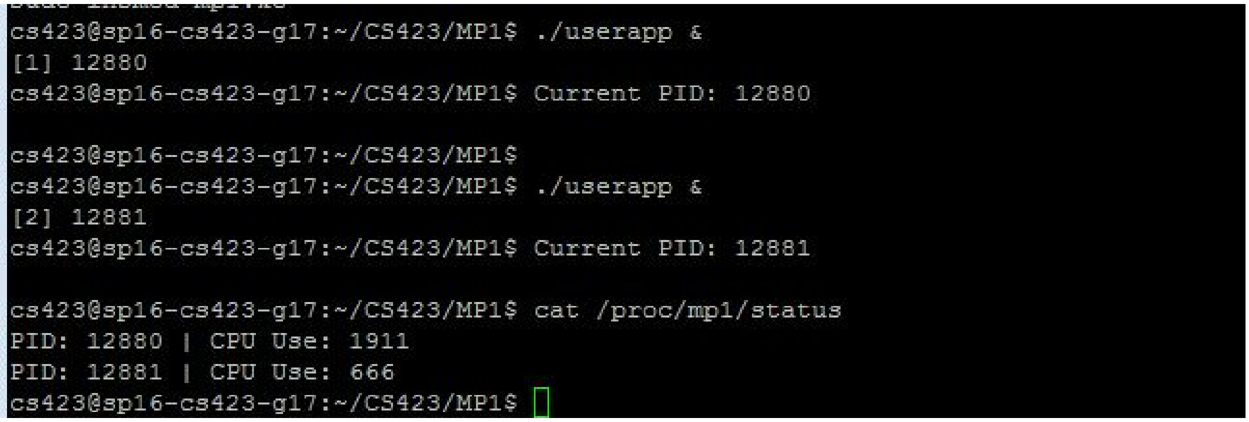
Group 17
Ahmed Shabban - shaaban2
Caleb Perkinson - perknsn2
Vibhakar Vemulapati - vemulpt2
Xiejia Zhang - xzhan121

MP1_Read: the read function is designed to be called twice with the return values being controlled by the global flag *read_end*. When read is called the first time, it copies the pid and cpu time stored in the linked list into the buffer provided by the user. The second time read is called, it returns 0 and copies nothing. This is done because *read* is continuously called until it returns 0, the EOF. But in the context of this MP, the entire list of PID/CPU time is to be returned to the user in a single *read* call. Therefore we designed our MP1_read to work this way.

MP1_Write: the write function simply takes the user input buffer containing the pid and appends it to the linked list. The new process struct is initialized with a CPU time of 0 and is added to the end of the process linked list in order to keep the head constant.

Work Queue and Timer: The timer and work queue worked served to implement the “Two-halves approach” to interrupt handling as follows. The Timer was configured to interrupt every 5 seconds and when its interrupt was fired it would schedule the work on a work queue. The work to be done was iterating through the linked list of process entries to update their cpu runtimes. This could be a lengthy process and so it should not be done in interrupt context, hence the workqueue. When the work queue was scheduled, it would run this work when it had the time to. It would first lock the data structure containing the linked list of processes and then iterate through and either update each cpu runtime or delete the entry if it had finished.

How to run:
Run “make” followed by “sudo insmod mp1.ko” . Afterwards, simply run “./userapp &” and userapp will print out its pid. Then “cat /proc/mp1/status” will print out whatever is registered on the linked list.



```
sudo insmod mp1.ko
cs423@sp16-cs423-g17:~/CS423/MP1$ ./userapp &
[1] 12880
cs423@sp16-cs423-g17:~/CS423/MP1$ Current PID: 12880

cs423@sp16-cs423-g17:~/CS423/MP1$
cs423@sp16-cs423-g17:~/CS423/MP1$ ./userapp &
[2] 12881
cs423@sp16-cs423-g17:~/CS423/MP1$ Current PID: 12881

cs423@sp16-cs423-g17:~/CS423/MP1$ cat /proc/mp1/status
PID: 12880 | CPU Use: 1911
PID: 12881 | CPU Use: 666
cs423@sp16-cs423-g17:~/CS423/MP1$
```