Steele Muchemore-Allen
April 18, 2019
ECE 331

Introduction

It is necessary to manage which process can use system resources at a given time in a mutli-core system. In the edge kernel module, this was achieved by encorporating mutex locks. The mutex lock is a simple way to lock the system resources to one process until the process is complete and releases the lock.

Locking: Purpose, Design, and Testing

Without proper locking, a process could interfere with another process, possibly causing data corruption or loss. In the edge kernel module, the mutex lock was selected for locking purposes.

Designing the kernel module with mutex locks was done with the functions in linux/mutex.h. Locks were initialized, tested, aquired, and relaesed using mutex function calls.

Testing the mutex lock was done using a C program which called fork() to create child processes. The processes attempted to write to the edge device, each waiting util the resources were unlocked by the current process. Testing the locking ensured that edge was a concurrent multi-threaded kernel module.

Conclusion

Mutex locking was the selected method of locking for the kernel module due to its ease of initializing, checking, and getting locks. The design of the kernel module used mutex funtions defined in linex/mutex.h. Testing the mutex lock was done with a C program that created parent and child processes to test the effectiveness and correctness of locking in the kernel.