**National University of Computer & Emerging Sciences**

**Karachi Campus**



**Project Proposal**

**Operating Systems**

**Section: 4F**

­**Spin Lock (System Call)**

**Member 1: Muhammad Talha Waheed (Leader) - 21k-4656**  
**Member 2: Ayush Kumar - 21k-4654  
Member 3: Avinash - 21k-3413**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Goal:**

The goal is to test the spin lock mechanism that does kernel-level thread synchronization through multiple threads to access shared resources in a safe and efficient manner. Spin locks are a type of synchronization primitive used by the kernel to protect critical sections of code. They operate by blocking the execution of competing threads until the lock becomes available, at which point one of the threads can acquire it and proceed with the critical section. The challenge in implementing spin locks is to ensure that they are efficient, i.e., they do not consume too much CPU time, while also being effective, i.e., they prevent race conditions and other concurrency-related issues.

**Description:**

A system call project involving spin locks that aims to implement a mechanism for kernel-level thread synchronization. The project would involve creating a system call that enables user-level programs to request exclusive access to shared resources, using a spin lock mechanism. The system call would operate by invoking kernel-level functions that block the execution of competing threads until the lock becomes available. The challenge in implementing this project would be to design an efficient and robust spin lock mechanism that can prevent race conditions, deadlock, and other concurrency-related issues. The implementation would also need to be carefully tested to ensure that it is reliable and does not introduce any performance bottlenecks or security vulnerabilities. Overall, this project would require a deep understanding of Operating System internals, including kernel-level synchronization primitives, process and thread management, and system call handling.

**Actors/System User(s):**

Different Users and Admin

**List of Features:**

1. Spin lock creation: The ability to create a new spin lock object from user-level code, specifying the initial state (locked or unlocked) and any additional parameters.
2. Lock acquisition: The ability to request exclusive access to a shared resource protected by a spin lock. This operation should block until the lock becomes available.
3. Lock release: The ability to release a previously acquired spin lock, allowing other threads to access the shared resource.
4. Fairness: The ability to implement a fair spin lock mechanism that ensures that waiting threads are granted access in the order they requested it.
5. Spin lock destruction: The ability to destroy a spin lock object, releasing any associated resources.
6. Error handling: The ability to handle errors and exceptions that may occur during the creation, acquisition, or release of spin locks.
7. Performance optimizations: The ability to optimize the spin lock implementation for performance, by minimizing the amount of time that threads spend spinning and maximizing the throughput of concurrent operations.

**Tools & Techniques:**

* Programming languages: The implementation will be done in C, which is commonly used for Operating System development.
* Synchronization primitives: The implementation may require the use of various synchronization primitives, such as semaphores and mutexes, to ensure thread safety.
* Kernel-level programming: The implementation would need to interface with the kernel, which would require knowledge of kernel-level programming techniques and APIs.
* Integration testing: Integration testing can be used to verify that the spin lock implementation works correctly with other components of the Operating System.

GCC/GNU compiler will be used to compile and run the C codes on Linux (Ubuntu) using the text editor.

**Schedule:**

[To be submitted one week before the final exam of the spring 2023 semester]

**Course Teacher:** Dr. Ghufran Ahmed **Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_