**ELE709 - Real-Time Computer Control Systems Lab 4 - Resource Sharing and Coordination**

# Name: Alex Hagos Student ID: 500763787

1. **Exercise 4.1:**
   1. Explain why the example program lab4.c didn’t work correctly?

When lab4.c is executed. It was noticed that the printed output consisted of a series of printed values which do not follow the order shown in the lab manual (0123456789). It is possible to say that the program lab4.c did not work correctly because the code does not include a way of controlling which and when threads can access the share resource. No semaphores or mutexes are present in the base code of lab4.c. As a result, as the code is initiated, all threads try to access the shared resource at the same time, and the output shows overlapping and apparent disorder in the printing output of the string. The execution of the strings was purely controlled by the OS scheduler.

* 1. A mutex was suggested as a way to make the example program work correctly. What is the purpose for using the mutex in this case? In particular, what is being protected by this mutex?

The purpose for using the mutex in this case would be control which thread can access the shared resource, and making sure that only one thread can make use of the content of the shared resource at a time. This is because a mutex can only be unlocked by the task that locked it in the first place, which means that external tasks or processes cannot interfere or manipulate the content of the shared resource until the task holding the mutex releases it. Therefore, what is being protected by the mutex is the critical section of the shared resource code is, which is printing the character string

“0123456789”.

1. **Exercise 4.2:** Explain the logic of your program for this exercise. In particular, what are the predicates (associated with the condition variable) for Thread A and Thread B?

In order to coordinate the two threads (Thread A and Thread B) in order to print the sequence “A0 B1 A2 B3 A4 B5 A6 B7 A8 B9” it was required to make use of a mutex and a conditional variable. In the code shown in ex4.2, for thread A there is an if statement that indicates the case in which the string index equals an even number, which results in the printing of A%d, this predicate in the if statement is used as the logic expression that triggers the sending of the signal by the conditional variable “condvar”. In the thread A function there is also another if statement, whose condition is the case in which the string index is an odd number. This condition serves as the logic expression that will trigger the “wait” function for the conditional variable, as odd string index numbers is, by design. The condition used to trigger the “send signal” condition in the thread B function.