ELEC 377 Lab 3 Description

The program developed by this group for this lab implements a simple hardware algorithm which uses the shared memory capability of modern operating systems to allow for synchronization between multiple processes. Specifically, the lab focused on the synchronization of consumer and producer programs that use shared memory. To accomplish this, two producers and one consumer were created. The producer program writes to the shared memory one character at a time from a text file if it has access to the lock. The consumer then reads one character at a time to an output file, but should only do so when the producer is not making changes and if there are characters to be read in the shared memory. The implementation of the producer and consumer programs can be found in files `producer.c` and `consumer.c` respectively. Files `common.h` and `common.c` include methods and the data structure that are common to both the producer and consumer programs.

To ensure synchronization between the consumer and producers, `meminit` and common were used. `meminit` allows for initialization of shared memory and was called before each producer and consumer read/wrote to memory. `Common.h` contained the structure that allowed for communication between the consumer and producers. The struct contains integer variables `lock`, `in`, `out`, `count`, and `numProducers`. `lock` was used to ensure mutual exclusion. The structure also contains an array of characters called '`buffer`' that is 5 bytes in size. This character array is where the characters are added too by the producers and where characters are read from by consumers. It is only 5 bytes in size to ensure that a single producer cannot fill it and exit in the same time slice. This array has been designed in this way such that testing with multiple producers can be done more easily. Specific difficulties faced while building the algorithm can be found in the appropriate documentation, `377Lab3_Testing.pdf`.