ECE 612 – Real Time Embedded Systems

Assignment 1

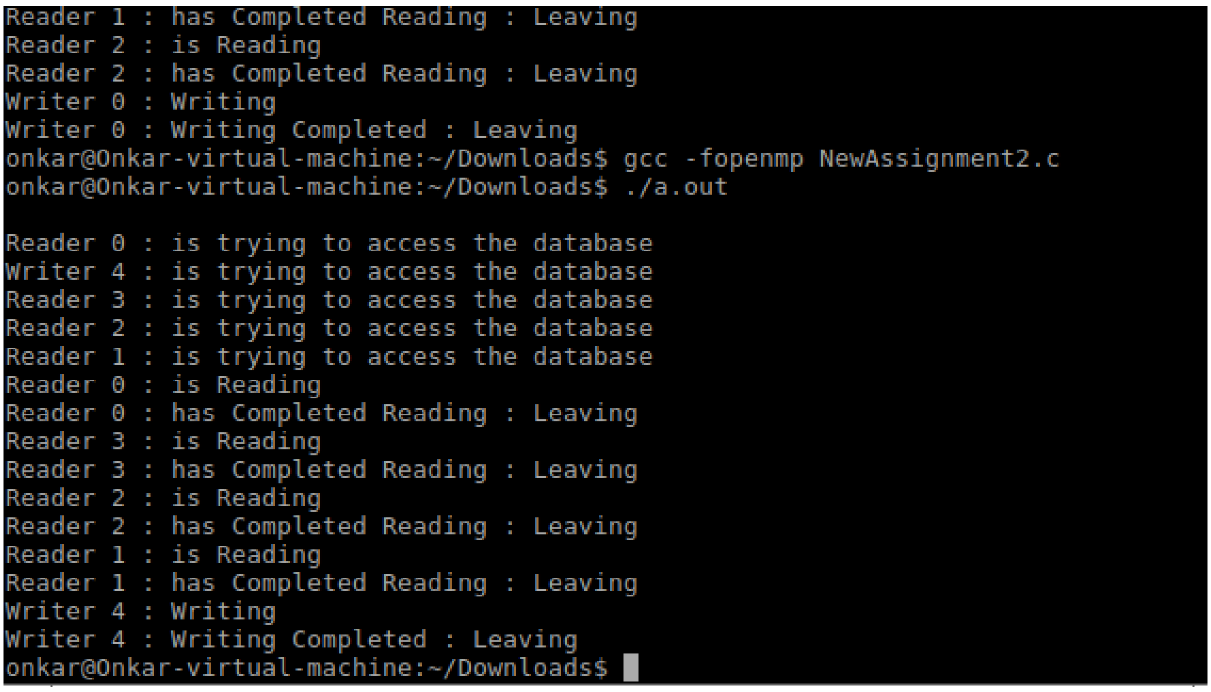
Name: Onkar Randive

G01036553

Problem Definition: This problem is a classic example of synchronisation problem in which multiple users try to access a shared resource simultaneously. In this particular case, multiple threads try to access a shared resource leading to deadlock and starvation. Reader threads try to read whereas the writer threads try to write/change the data. The solution tries to avoid deadlock and starvation by implementing locks such that no other thread are given access when a writer thread is writing in to the memory.

Solution: The problem is solved in the C programming language using OpenMP (a cross environmental platform used for shared memory parallel programming) to create parallel threads to create a Readers Writers scenario. The omp.h library file is used to use the OpenMP functions such as omp_set_lock(),omp_init_lock() etc. There are 5 threads (the number of threads can be changed inside the program in-order to increase the number of threads)that are randomly selected to be Readers or writers by generating random binary bits using the rand () function.

Output:



```
Reader 1 : has Completed Reading : Leaving
Reader 2 : is Reading
Reader 2 : has Completed Reading : Leaving
Writer 0 : Writing
Writer 0 : Writing Completed : Leaving
onkar@Onkar-virtual-machine:~/Downloads$ gcc -fopenmp NewAssignment2.c
onkar@Onkar-virtual-machine:~/Downloads$ ./a.out

Reader 0 : is trying to access the database
Writer 4 : is trying to access the database
Reader 3 : is trying to access the database
Reader 2 : is trying to access the database
Reader 1 : is trying to access the database
Reader 0 : is Reading
Reader 0 : has Completed Reading : Leaving
Reader 3 : is Reading
Reader 3 : has Completed Reading : Leaving
Reader 2 : is Reading
Reader 2 : has Completed Reading : Leaving
Reader 1 : is Reading
Reader 1 : has Completed Reading : Leaving
Writer 4 : Writing
Writer 4 : Writing Completed : Leaving
onkar@Onkar-virtual-machine:~/Downloads$
```

Two separate locks are defined for Readers and Writers respectively. OMP locks are used to lock Readers from reading while a Writer is writing onto the shared resource. Similarly, writer locks are used to prevent a writer from writing data while a reader is accessing the data. Also, starvation is avoided by running the threads in the order they are invoked. The shared resource used in the code is the STDOUT.

References:

1) OpenMP – Livermore Computing

   https://computing.llnl.gov/tutorials/openMP/

2) https://gcc.gnu.org/wiki/openmp

Note : Code file(.C file has been included)

Please compile as gcc G01036553.c -fopenmp