CG2271 Real Time Operating Systems
Tutorial 8

Question 1

Describe what a race condition is.

Suppose two identical tasks update a shared variable *tmp* using this instruction:

```
tmp++;
```

Can a race condition occur? Why or why not?

Question 2

Using Google, the textbook or any other resource, describe what "atomicity" means. The kernel often disables interrupts to guarantee atomicity. Explain how this works. Explain further why it's a bad idea to allow user processes to achieve atomicity in the same way.

Question 3

For each of the following mechanisms:

- i) Is the mechanism is effective in enforcing mutal exclusions, and why or why not.
- ii) If the mechanism is able to enforce mutual exclusion, list and explain the advantages and disadvantages of the mechanism.

1) Disabling Interrupts.
2) Lock variables.
3) Strict alternation.
4) Peterson's Solution.
5) Sleep/wakeup.

Question 4

The condition variables in monitors work in a very similar way to "sleep" and "wake".

a. Explain how condition variables are the same and how they are different from "sleep" and "wake".
b. The "sleep" and "wake" operations are known to cause deadlock, resulting in the producer/consumer problem. If a buffer is implemented using monitors and condition variables, can the producer/consumer problem still occur? Why or why not?