As seen in the above output of the program, once the priority raises for the communication task, it stays at a priority of 4. It reaches 1504 ms early on in the program and the task priority changes to 4 just as the program requirement requests. As provided in the code, I used vApplicationTickHook for setting a tick counter for both the matrix task and comms task.

1. Why is matrixtask using most of the CPU utilization?

The matrix task takes more execution time becdause it is computationally more expensive and has a higher priority time than the other tasks

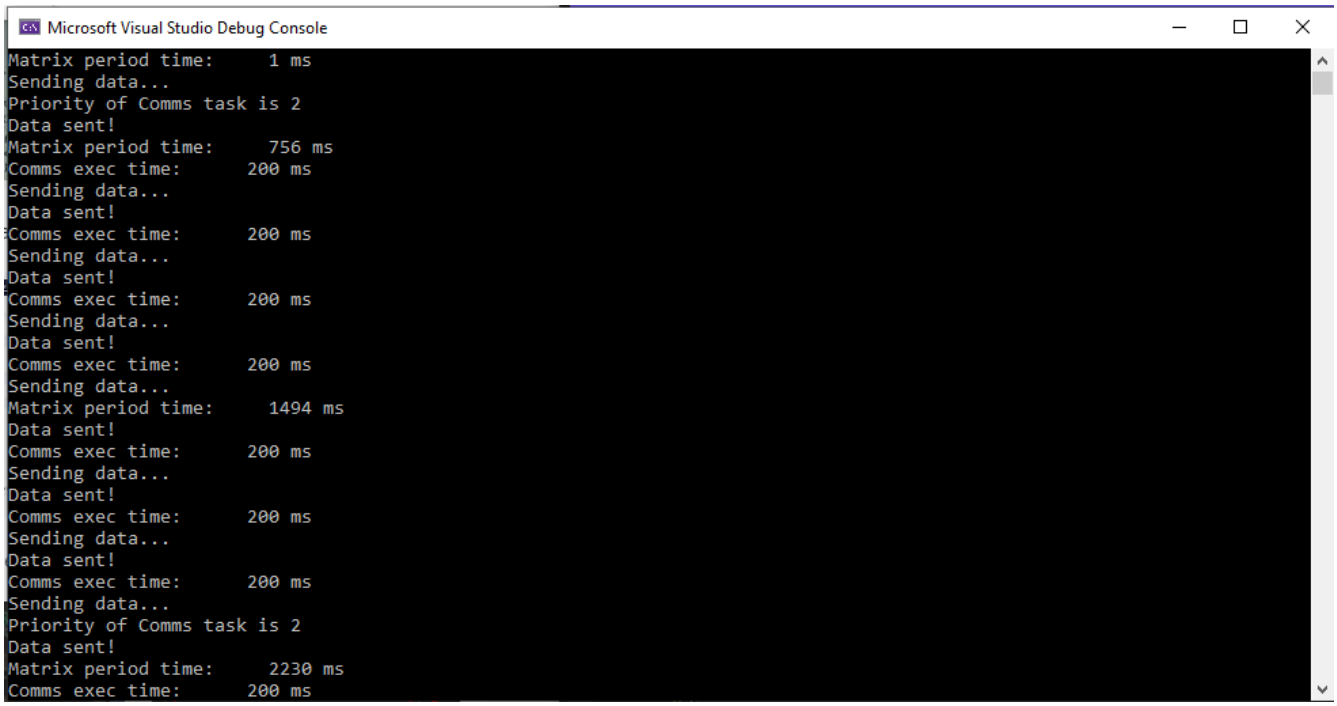
2. Why must the priority of communicationtask increase in order for it to work properly?

The matrix task does have a higher execution time than the communication task at the start of the program. This causes the matrix task to block the communication task due to its expensive computation. By raising the priority of the communication task, we can make sure it runs properly. This would be an example of pre-empting.

3. What happens to the completion time of matrixtask when the priority of communication task is increased?

There is little difference in the execution time for the matrix task. It generally stays around the 630 ms execution time range.

4. How many seconds is the period of matrixtask?



This was calculated to be about 0.756 seconds. This was done by taking the printf for finding the task time and placing it at the beginning of the task instead of the bottom of the task. In this case, I can see the next time the matrix task starts again, getting the period.