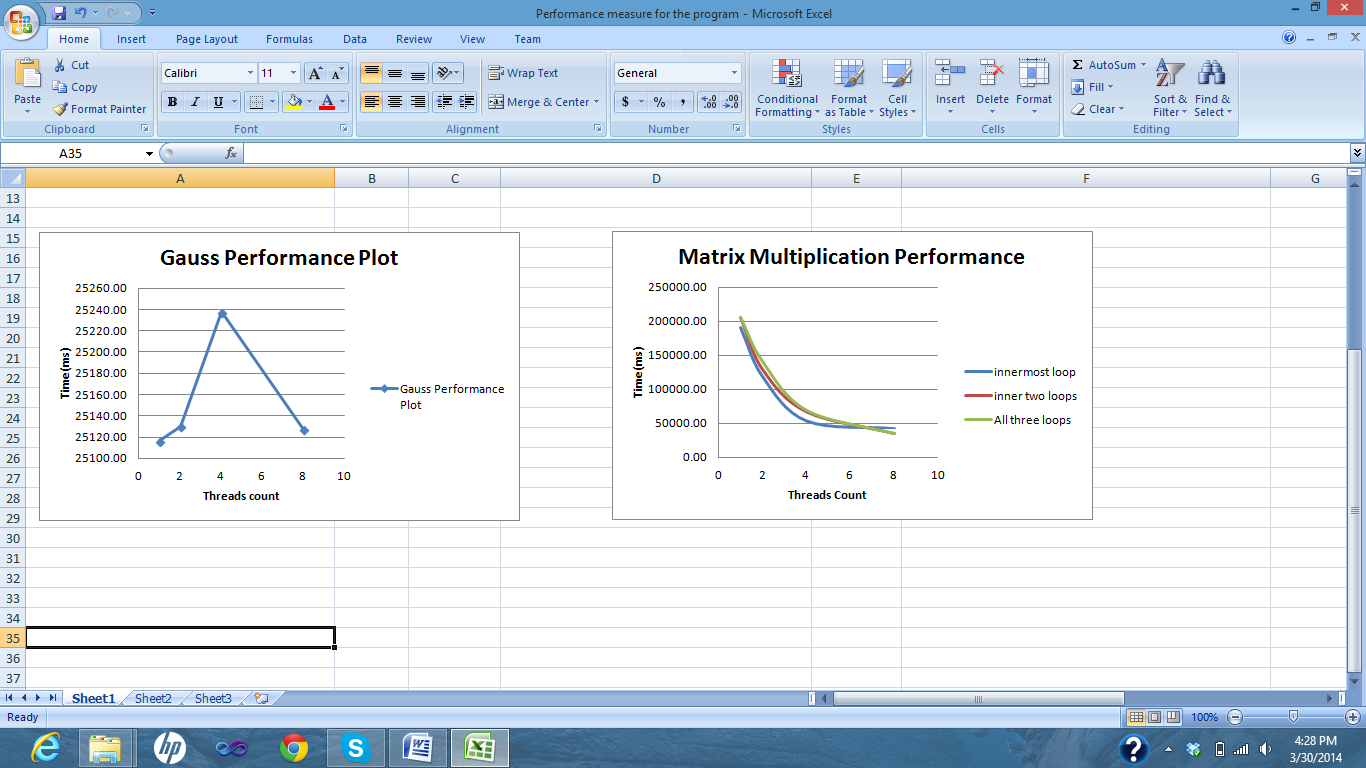
## Programs & its performance measurements :(Submission Document)

**Programs & code :** gaussomp.c, matrixcpy.c, matrixin.c, matrixin2.c

**Performance Measurements:** Performance measure for the program.xlsx

**Solution 1:**



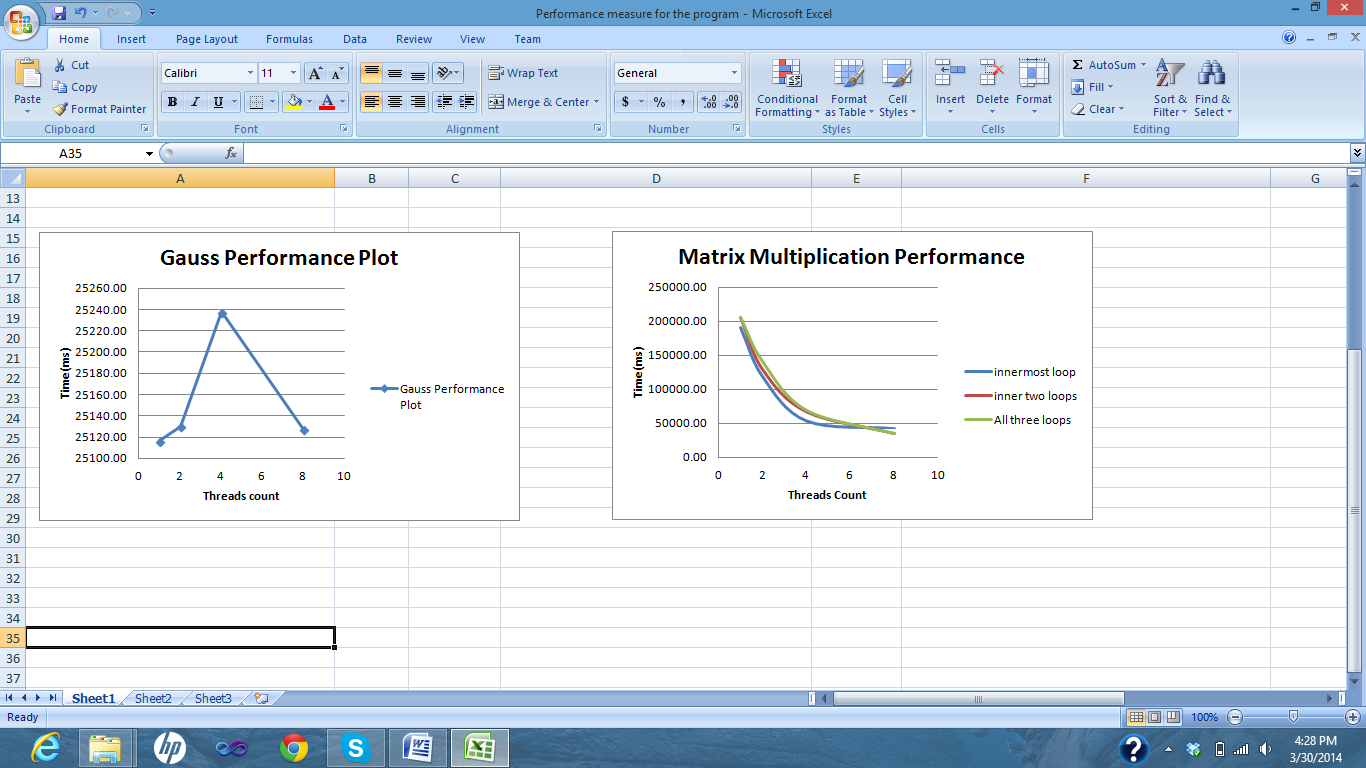
**Solution 2 : Deliverables**

1.The synchronization is all taken care by the joining of threads in the code as the two inner for loops are independent (no loop dependencies),those can be parallelized.

2.The outermost for loop cannot be parallelized as the values which are updated in the first for loop will be updated again the second or so forth loops. To avoid this dependency, the outer loop is not parallelized in my code.

3.Please find the below graph for the plot for performance plot.

4.I have tried parallelizing the outer loop with the help of barrier but I am not able to come up with the consistent results at all the runs. Hence I felt this would be a better option. I had tried two versions of the code but with the same technique parallelizing the inner loops. Attached document has two versions. One with barrier gave poor performance. Hence I feel the better performance would be without using barriers for the inner loop. My conclusion is purely based on my experiments with the code files generated by me.



**Reference :**

www.google.com - for few programming syntax and the semantics.