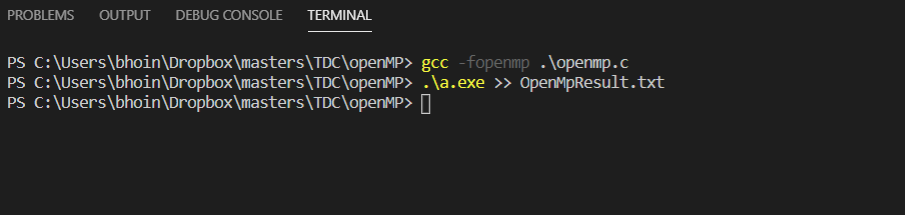
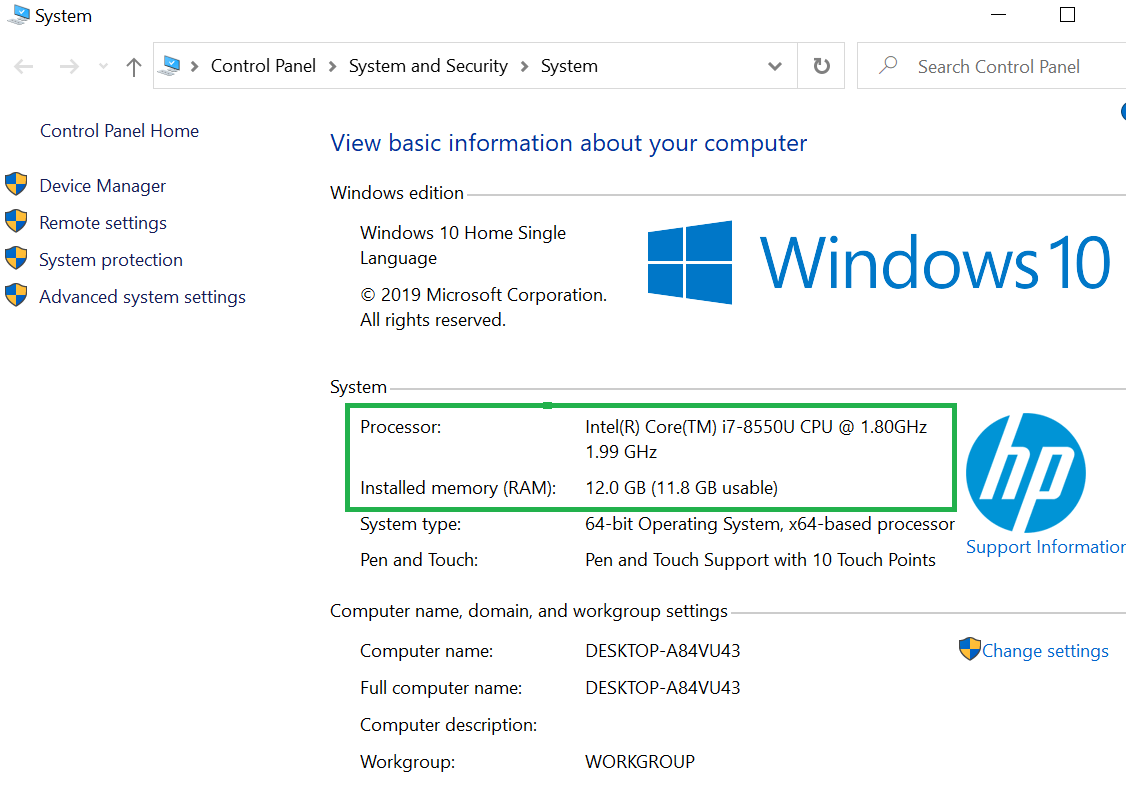
1. **Steps taken:**
2. Define serial matrix multiplication algorithm.
3. Define parallel matrix multiplication algorithm.
4. Check the accuracy of parallel matrix multiplication algorithm using serial matrix multiplication algorithm for 3 X 3 matrix multiplication
5. Execute parallel matrix multiplication algorithm using 2, 4 ,6 and 8 threads for 250 X 250, 500 X 500, 1000 X 1000, 2000 X 2000 matrix multiplication
6. Calculated the speed up for each multiplication operation.
7. **Command to execute openmp.c:**



You can code in openmp.c file and result in OpenMpResult.txt file

1. **Result – Report:**

a simple parallel algorithm for the matrix-matrix product using OpenMP. All executions, times and speed-ups are measured using the following machine.

****

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Matrix Size** | **Sequential Time** | **Parallel Time** | **Number of Threads** | **Speed Up** |
| 250 X 250 | 0.084000 | 0.063000 | 2 | 1.333335 |
| 250 X 250 | 0.084000 | 0.032000 | 4 | 2.625000 |
| 250 X 250 | 0.084000 | 0.028000 | 6 | 2.999999 |
| 250 X 250 | 0.084000 | 0.030000 | 8 | 2.800010 |
| 500 X 500 | 1.386000 | 0.724000 | 2 | 1.914365 |
| 500 X 500 | 1.386000 | 0.919000 | 4 | 1.508161 |
| 500 X 500 | 1.386000 | 0.646000 | 6 | 2.145510 |
| 500 X 500 | 1.386000 | 0.357000 | 8 | 3.882352 |
| 1000 X 1000 | 13.076000 | 6.022000 | 2 | 2.171372 |
| 1000 X 1000 | 13.076000 | 4.082000 | 4 | 3.203332 |
| 1000 X 1000 | 13.076000 | 3.583000 | 6 | 3.649456 |
| 1000 X 1000 | 13.076000 | 3.875000 | 8 | 3.374452 |
| 2000 X 2000 | 101.899000 | 61.812000 | 2 | 1.648531 |
| 2000 X 2000 | 101.899000 | 43.622000 | 4 | 2.335954 |
| 2000 X 2000 | 101.899000 | 35.517000 | 6 | 2.869020 |
| 2000 X 2000 | 101.899000 | 37.252000 | 8 | 2.735397 |

we get a maximum speed up near 3.6. OpenMP-enabled parallel code exploits coarse grain parallelism.

1. we have parallelized the outermost loop which drives the accesses to the result matrix in the first dimension.
2. Work-sharing among the threads is such that different threads will calculate different rows of the result matrix.
3. Accesses to matrices “matrixA” and “matrixB” are read-only and do not introduce any problems either.
4. Different threads will write different parts of the result in the array result, so we don’t get any problems during the parallel execution.

Source Code: <https://github.com/Nehabhoi/openMP>