**HPC Assignment No. 01**

**Batch:** B2

**PRN No:** 2019BTECS00033

**Name**: Teknath jha

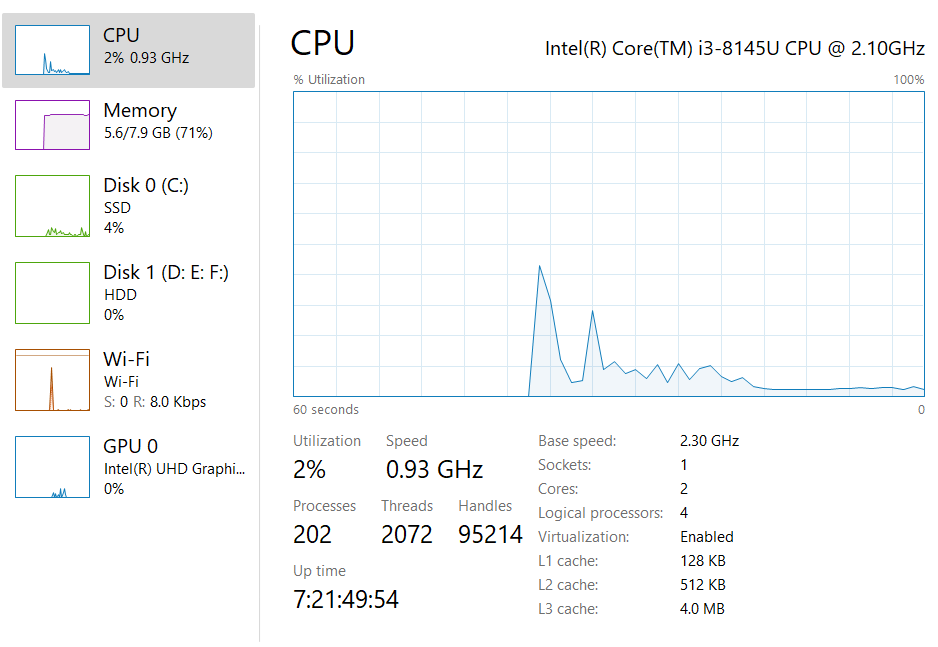
**Date**: 24th of August 2022

* **Title:**

**openMp program for :**

1. **Hello World**
2. **Squares from 1 to 100**

**MY SYSTEM CONFIGURATION :**



**Physical Vs Software threads :**

* + Software threads are threads of execution managed by the operating system.
  + A "hardware thread" is a physical CPU or core. So, a 4 core CPU can genuinely support 4 hardware threads at once - the CPU really is doing 4 things at the same time.
  + One hardware thread can run many software threads.
  + In modern operating systems, this is often done by time-slicing - each thread gets a few milliseconds to execute before the OS schedules another thread to run on that CPU

**ScreenShots OF Practical:**

1. **Hello World :**

**Parallel program :**

*/\**

*Author - Teknath jha*

*PRN    - 2019BTECS00033*

*HPC-Lab- 1*

*\*/*

*// PARALLEL*

#include <omp.h>

#include <time.h>

int main() {

  printf("Parallel program : \n");

  double start = omp\_get\_wtime();

#pragma omp parallel

  {

    printf("hello world from 2019BTECS00033 by thread %d \n",

           omp\_get\_thread\_num());

  }

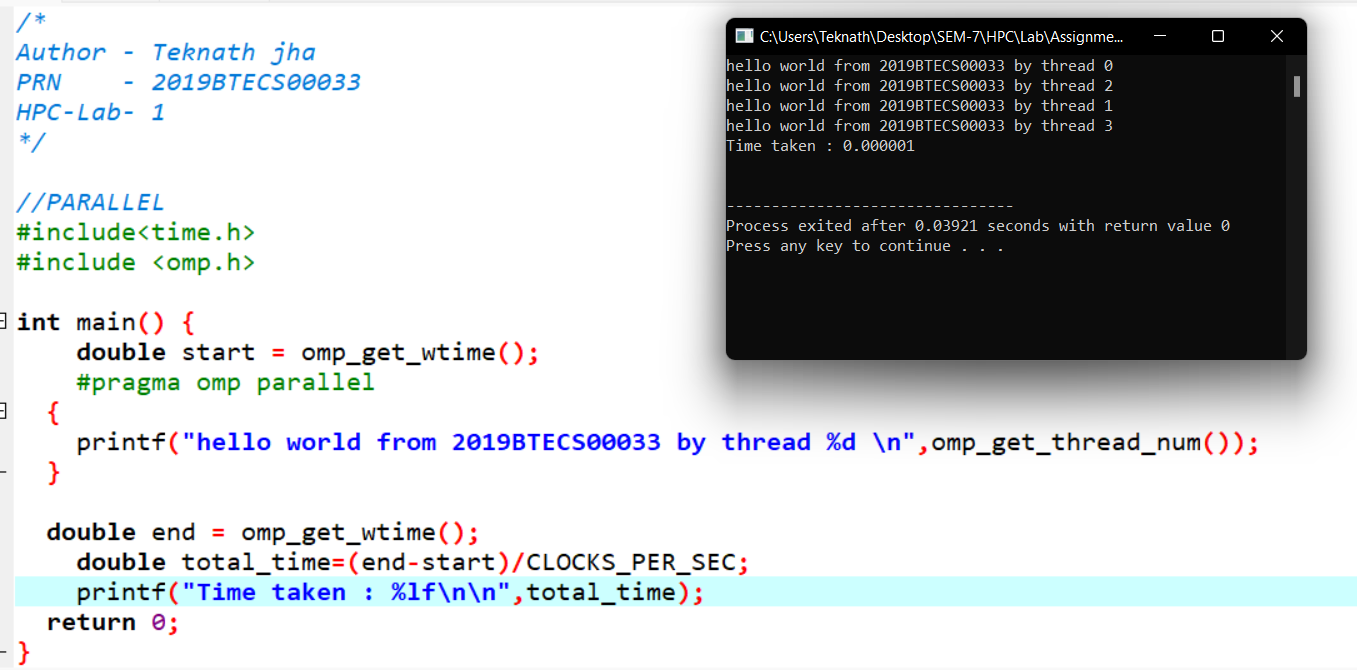
  double end = omp\_get\_wtime();

  double total\_time = (end - start) / CLOCKS\_PER\_SEC;

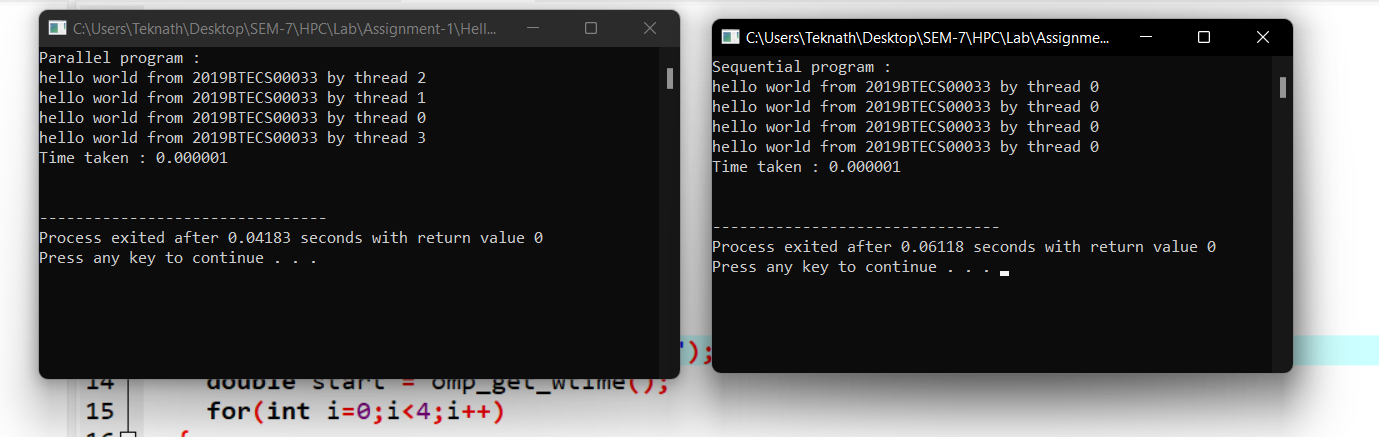
  printf("Time taken : %lf\n\n", total\_time);

  return 0;

}

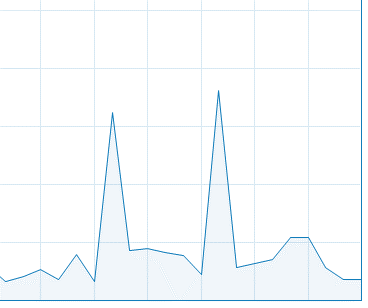
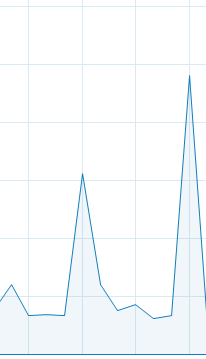
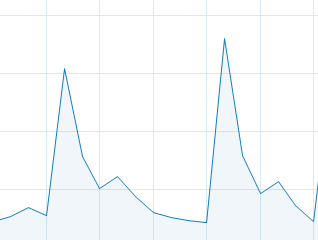


**Comparison with sequential :**

****

**In below images 1st peak is of sequential and later is of parallel program :**

**Images from CPU Utilization Task Manager :**

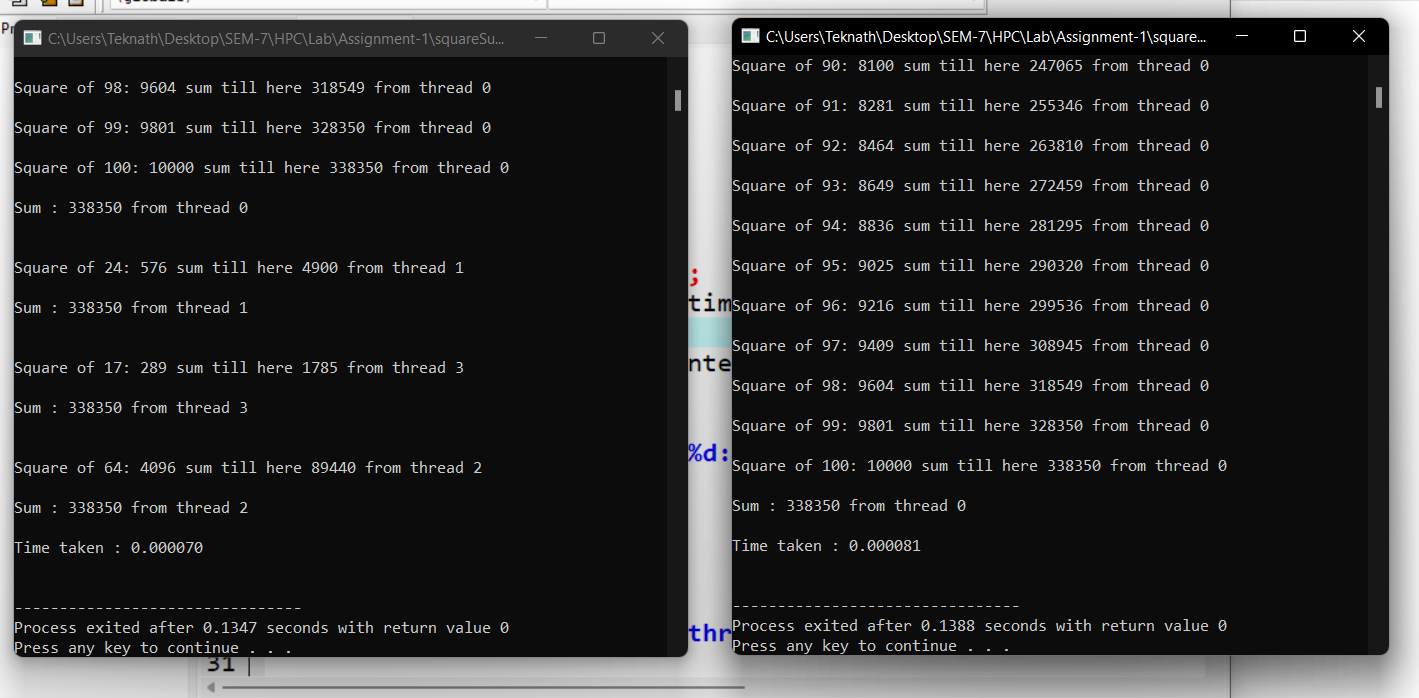
1. 
2. ****
3. ****

**Conclusion : my sequential program uses less CPU and parallel program uses more UPU for same program and same number of instruction Sets.**

**Although time calculation is negligible as it is small program .**

**B) Squares from 1 to 100**

**Output Screenshots :**

****

**Parallel :70 Sequential: 81**

**So here parallel is faster than sequential.**

*/\**

*Author - Teknath jha*

*PRN    - 2019BTECS00033*

*HPC-Lab- 1*

*\*/*

*//PARALLEL*

*//finding sum of squares of 1 to 100*

#include<omp.h>

#include<time.h>

int main(){

  int id;

  long long int sum=0;

  long long  int square=0;

  long long  int pointer=0;

  double start = omp\_get\_wtime();

  #pragma omp parallel

  {

    while(pointer <101){

       square = pointer\*pointer;

      sum+=square;

      printf("\nSquare of %d: %d sum till here %d from thread %d \n",pointer,square,sum ,omp\_get\_thread\_num() );

      pointer++;

    }

    printf("\nSum : %d from thread %d \n\n",sum ,omp\_get\_thread\_num() );

  }

  double end = omp\_get\_wtime();

  double total\_time=(end-start)/CLOCKS\_PER\_SEC;

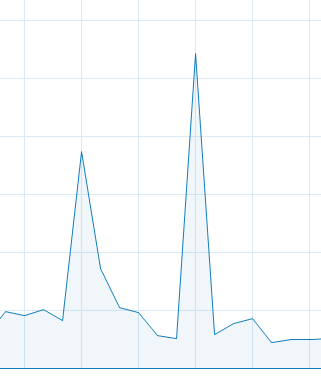
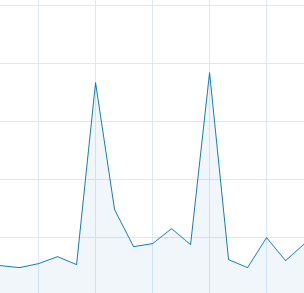
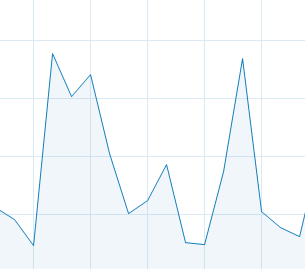
  printf("Time taken : %lf\n\n",total\_time);

  return 0;

}

**In below images 1st peak is of sequential and later is of parallel program :**

**CPU Graphs :**

1. ****
2. ****
3. ****

**Here most of time width of sequential is more than parallel which shows time difference.**

**Conclusion :**

**In execution : sequential taken 0.000081 while parallel taken 0.000070 which is considerable difference , further observation of CPU cycles also proves this that parallel is faster than sequential .**