

Final Year B. Tech., Sem VII 2022-23

High Performance Computing Lab

PRN/ Roll No: 2020BTECS00206

Full Name: SAYALI YOGESH DESAI

Batch: B4

Assignment No. 1

Que 1. Difference between Software and Hardware threads

Hardware Thread:

- i. 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.
- ii. One hardware thread can run many software threads.
- iii. 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.
- iv. Since the OS switches back and forth between the threads quickly, it appears as if one CPU is doing more than one thing at once, but in reality, a core is still running only one hardware thread, which switches between many software threads.

Software Thread:

- i. Software threads are threads of execution managed by the operating system.
- ii. Software threads are abstractions to the hardware to make multi-processing possible.
- iii. If you have multiple software threads but there are not multiple resources then these software threads are a way to run all tasks in parallel by allocating resources for limited time (or using some other strategy) so that it appears that all threads are running in parallel.

Que 2. Which type of threads are supported by the processor?

- i. Threads are the virtual components or codes, which divides the physical core of a CPU into virtual multiple cores. A single CPU core can have up-to 2 threads per core.
- ii. For example, if a CPU is dual core (i.e., 2 cores) it will have 4 threads. And if a CPU is Octal core (i.e., 8 core) it will have 16 threads and vice-versa.
- iii. There is single thread on the core which when gets the information from the user, creates another thread and allocates the task to it. Similarly, if it gets another

instruction, it forms second thread and allocates the task to it. Making a total of two threads.

- iv. Generally, the Hardware Threads are supported by the processor. The hardware threads are mostly based on the multi-core architecture which is latest architecture to achieve high performance.

Que 3. Simple Hello World Program with 4 threads

- **Code**

```
#include<omp.h>

#include<stdio.h>

#include<stdlib.h>

int main(int argc, char* argv[]){

    #pragma omp parallel

    {

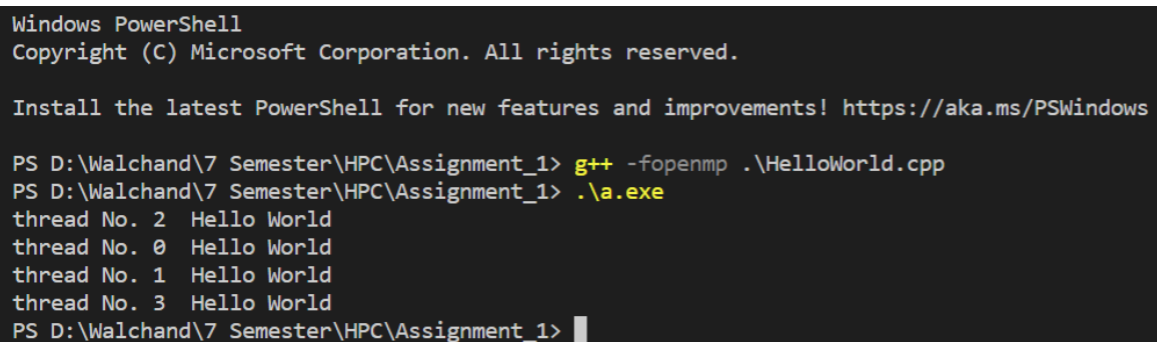
        printf("thread No. %d Hello World\n", omp_get_thread_num());

    }

    return 0;

}
```

- **Output**



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\Walchand\7 Semester\HPC\Assignment_1> g++ -fopenmp .\HelloWorld.cpp
PS D:\Walchand\7 Semester\HPC\Assignment_1> .\a.exe
thread No. 2 Hello World
thread No. 0 Hello World
thread No. 1 Hello World
thread No. 3 Hello World
PS D:\Walchand\7 Semester\HPC\Assignment_1> █
```

- fopenmp is a flag which is required to execute openmp parallel programs on GCC compiler
- omp.h - header file in C which have all openmp functions defined
- omp_get_thread_num() - function to get thread ID/ number

Que 4. Calculate squares and sum of first 100 numbers using serial processing. Compare speed in parallel and serial processing.

- **Code:**

```
#include<omp.h>
#include<stdio.h>
#include<stdlib.h>
#include<bits/stdc++.h>
using namespace std;

static int sum =0;
int main()
{
    double itime, ftime, exec_time;
    itime = omp_get_wtime();
    for(int i=1; i<=100;i++)
    {
        printf("Thread No. %d Number : %d Square : %d \n", omp_get_thread_num(), i, i
* i);
        sum += i * i;
    }
    printf("\nSum is %d ", sum);
    cout << endl;
    ftime = omp_get_wtime();
    exec_time = (ftime - itime);
    printf("\nTime taken is %f\n", exec_time);
    return 0;
}
```

- **Output:**

```
Thread No. 0 Number : 88 Square : 7744
Thread No. 0 Number : 89 Square : 7921
Thread No. 0 Number : 90 Square : 8100
Thread No. 0 Number : 91 Square : 8281
Thread No. 0 Number : 92 Square : 8464
Thread No. 0 Number : 93 Square : 8649
Thread No. 0 Number : 94 Square : 8836
Thread No. 0 Number : 95 Square : 9025
Thread No. 0 Number : 96 Square : 9216
Thread No. 0 Number : 97 Square : 9409
Thread No. 0 Number : 98 Square : 9604
Thread No. 0 Number : 99 Square : 9801
Thread No. 0 Number : 100 Square : 10000
```

```
Sum is 338350
```

```
Time taken is 0.052000
```

```
PS D:\Walchand\7 Semester\HPC\Assignment_1> █
```

Que 5. Calculate squares and sum of first 100 numbers using parallel processing. Compare speed in parallel and serial processing.

- Code:**

```
#include<omp.h>
#include<stdio.h>
#include<stdlib.h>
#include<bits/stdc++.h>
using namespace std;

static int sum =0;
int main(int argc, char *argv[])
{
    double itime, ftime, exec_time;
    itime = omp_get_wtime();
    #pragma omp parallel
        for(int i=1; i<=100;i++)
        {
            if(i%4==omp_get_thread_num())
            {
                printf("\nThread No. %d Number : %d Square : %d", omp_get_thread_num(), i, i
                * i);
                sum+=i*i;
            }
        }
    printf("\n\nSum is %d",sum);
    cout << endl;
    ftime = omp_get_wtime();
    exec_time = (ftime - itime);
    printf("\nTime taken is %f\n", exec_time);
    return 0;
}
```

- Output:**

```
Thread No. 0 Number : 64 Square : 4096
Thread No. 0 Number : 68 Square : 4624
Thread No. 0 Number : 72 Square : 5184
Thread No. 3 Number : 55 Square : 3025
Thread No. 3 Number : 59 Square : 3481
Thread No. 3 Number : 63 Square : 3969
Thread No. 3 Number : 67 Square : 4489
Thread No. 3 Number : 71 Square : 5041
Thread No. 2 Number : 74 Square : 5476
Thread No. 2 Number : 78 Square : 6084
Thread No. 1 Number : 81 Square : 6561
Thread No. 0 Number : 100 Square : 10000
Thread No. 3 Number : 99 Square : 9801
Thread No. 2 Number : 98 Square : 9604

Sum is 338350

Time taken is 0.059000
PS D:\Walchand\7 Semester\HPC\Assignment_1> █
```