

DEPARTMENT OF INFORMATION TECHNOLOGY, NITK SURATHKAL

Parallel Computing

LAB 1

Name: Chinmayi C. Ramakrishna

Roll No: 181IT113

CPU Details:

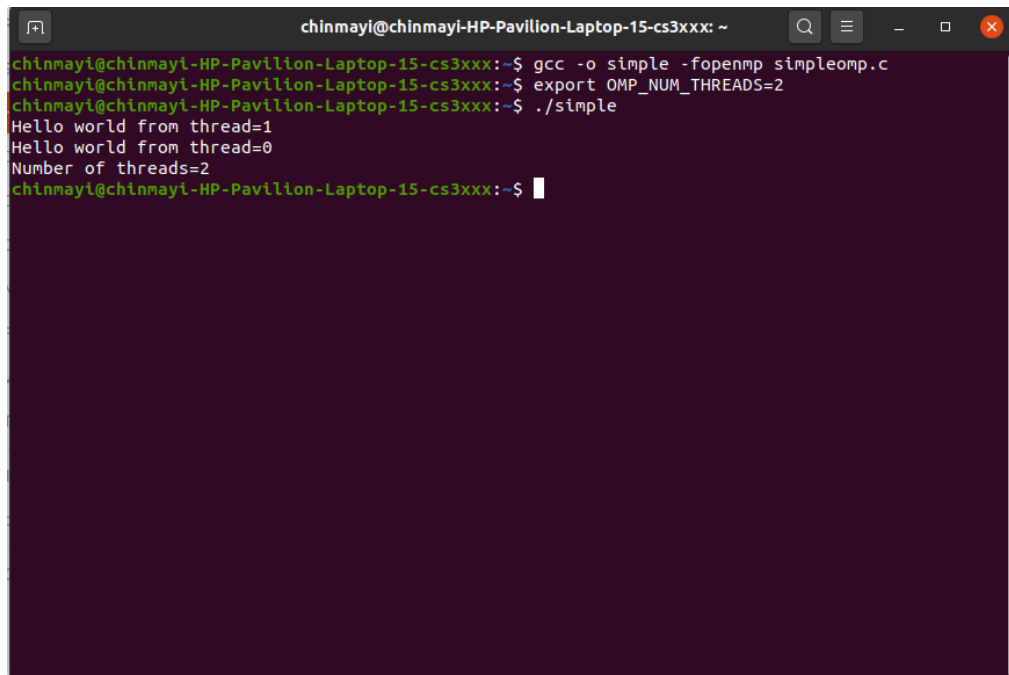
```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ lscpu
Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
Address sizes: 39 bits physical, 48 bits virtual
CPU(s): 8
On-line CPU(s) list: 0-7
Thread(s) per core: 2
Core(s) per socket: 4
Socket(s): 1
NUMA node(s): 1
Vendor ID: GenuineIntel
CPU family: 6
Model: 126
Model name: Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz
Stepping: 5
CPU MHz: 1211.456
CPU max MHz: 3900.0000
CPU min MHz: 400.0000
BogoMIPS: 2995.20
Virtualization: VT-x
L1d cache: 192 KiB
L1i cache: 128 KiB
L2 cache: 2 MiB
L3 cache: 8 MiB
NUMA node0 CPU(s): 0-7
Vulnerability Itlb multihit: KVM: Mitigation: Split huge pages
Vulnerability L1tf: Not affected
Vulnerability Mds: Not affected
Vulnerability Meltdown: Not affected
Vulnerability Spec store bypass: Mitigation: Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1: Mitigation: usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2: Mitigation: Enhanced IBRS, IBPB conditional, RSB filling
Vulnerability Tsx async abort: Not affected
Flags: fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_
_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf tsc_known_freq pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid s
se4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single ssbd lbrs lbrp stibp lbrs_enhanced
tpm_shadow_vmmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid avx512f avx512dq rdseed adx bnaop avx512ifma clflushopt intel_pt avx512cd
sha_ni avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp hwp_pkg_req avx512vbmi umip pku ospke avx512_vbmi2
gfni vaes vpclmulqdq avx512_vnni avx512_bitalg avx512_vpopcntdq rdpid md_clear flush_l3d arch_capabilities

chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ lscpu | egrep 'Model name|Socket|Thread|NUMA|CPU(s)'
```

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ lscpu -p
# The following is the parsable format, which can be fed to other
# programs. Each different item in every column has a unique ID
# starting from zero.
# CPU,Core,Socket,Node, ,L1d,L1i,L2,L3
0,0,0,0,,0,0,0,0
1,1,0,0,,1,1,1,0
2,2,0,0,,2,2,2,0
3,3,0,0,,3,3,3,0
4,0,0,0,,0,0,0,0
5,1,0,0,,1,1,1,0
6,2,0,0,,2,2,2,0
7,3,0,0,,3,3,3,0
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ nproc --all
8
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ echo "Threads/core: $(nproc --all)"
Threads/core: 8
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

1. Write a C/C++ simple parallel program to display the *thread_id* and total number of threads.

a. Using OMP_NUM_THREADS



```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp simpleomp.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ export OMP_NUM_THREADS=2  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Hello world from thread=1  
Hello world from thread=0  
Number of threads=2  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

b. Using num_threads() method

```
simpleomp.c  x  ifparallel.c  x  num_threads.c  x  addarray.c
1  /*simpleomp.c*/
2  #include<omp.h>
3  #include<stdio.h>
4  int main(){
5  int nthreads,tid;
6  #pragma omp parallel private(tid) num_threads(2)
7  {
8  tid=omp_get_thread_num();
9  printf("Hello world from thread=%d\n",tid);
10 if(tid==0)
11 {
12     nthreads=omp_get_num_threads();
13     printf("Number of threads=%d\n",nthreads);
14 }
15 }
16 }
17
```

Output:

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp simpleomp.c
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple
Hello world from thread=0
Number of threads=2
Hello world from thread=1
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

c. Using omp_set_num_threads() Method:

```
simpleomp.c x ifparallel.c x num_threads.c x addarray.c
1  /*simpleomp.c*/
2  #include<omp.h>
3  #include<stdio.h>
4  int main(){
5  int nthreads,tid;
6  omp_set_num_threads(2);
7  #pragma omp parallel private(tid)
8  {
9  tid=omp_get_thread_num();
10 printf("Hello world from thread=%d\n",tid);
11 if(tid==0)
12 {
13     nthreads=omp_get_num_threads();
14     printf("Number of threads=%d\n",nthreads);
15 }
16 }
17 }
18
```

Output:

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp simpleomp.c
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple
Hello world from thread=0
Number of threads=2
Hello world from thread=1
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

2. ifparallel.c

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ subl ifparallel.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp ifparallel.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ export OMP_NUM_THREADS=2  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Enter 0: for serial 1: for parallel  
0  
Serial val=0 id= 0  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Enter 0: for serial 1: for parallel  
1  
Parallel val=1 id= 0  
Parallel val=1 id= 1  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

3. num_threads.c

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp num_threads.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Hello world from thread=0  
Hello world from thread=3  
Hello world from thread=2  
Hello world from thread=1  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

4. Write a C/C++ parallel program for adding corresponding elements of two arrays.

a. Chunk=2 n=20

Each thread id has 10 i values.

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp addarray.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Thread id= 0 i=0,c[0]=0  
Thread id= 0 i=1,c[1]=5  
Thread id= 0 i=4,c[4]=20  
Thread id= 0 i=5,c[5]=25  
Thread id= 0 i=8,c[8]=40  
Thread id= 0 i=9,c[9]=45  
Thread id= 1 i=2,c[2]=10  
Thread id= 1 i=3,c[3]=15  
Thread id= 1 i=6,c[6]=30  
Thread id= 1 i=7,c[7]=35  
Thread id= 1 i=10,c[10]=50  
Thread id= 1 i=11,c[11]=55  
Thread id= 1 i=14,c[14]=70  
Thread id= 1 i=15,c[15]=75  
Thread id= 1 i=18,c[18]=90  
Thread id= 1 i=19,c[19]=95  
Thread id= 0 i=12,c[12]=60  
Thread id= 0 i=13,c[13]=65  
Thread id= 0 i=16,c[16]=80  
Thread id= 0 i=17,c[17]=85  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

b. Chunk= 4 n=16

Each thread id has 4 i values.

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp addarray.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Thread id= 0 i=0,c[0]=0  
Thread id= 0 i=1,c[1]=5  
Thread id= 0 i=2,c[2]=10  
Thread id= 0 i=3,c[3]=15  
Thread id= 3 i=12,c[12]=60  
Thread id= 3 i=13,c[13]=65  
Thread id= 2 i=8,c[8]=40  
Thread id= 2 i=9,c[9]=45  
Thread id= 2 i=10,c[10]=50  
Thread id= 2 i=11,c[11]=55  
Thread id= 1 i=4,c[4]=20  
Thread id= 1 i=5,c[5]=25  
Thread id= 1 i=6,c[6]=30  
Thread id= 1 i=7,c[7]=35  
Thread id= 3 i=14,c[14]=70  
Thread id= 3 i=15,c[15]=75  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```

c. $\text{Chunk} = 5$ $n = 15$

Each thread id has 5 i values.

```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx: ~  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ gcc -o simple -fopenmp addarray.c  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ ./simple  
Thread id= 1 i=5,c[5]=25  
Thread id= 1 i=6,c[6]=30  
Thread id= 1 i=7,c[7]=35  
Thread id= 1 i=8,c[8]=40  
Thread id= 1 i=9,c[9]=45  
Thread id= 2 i=10,c[10]=50  
Thread id= 2 i=11,c[11]=55  
Thread id= 2 i=12,c[12]=60  
Thread id= 2 i=13,c[13]=65  
Thread id= 2 i=14,c[14]=70  
Thread id= 0 i=0,c[0]=0  
Thread id= 0 i=1,c[1]=5  
Thread id= 0 i=2,c[2]=10  
Thread id= 0 i=3,c[3]=15  
Thread id= 0 i=4,c[4]=20  
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$
```