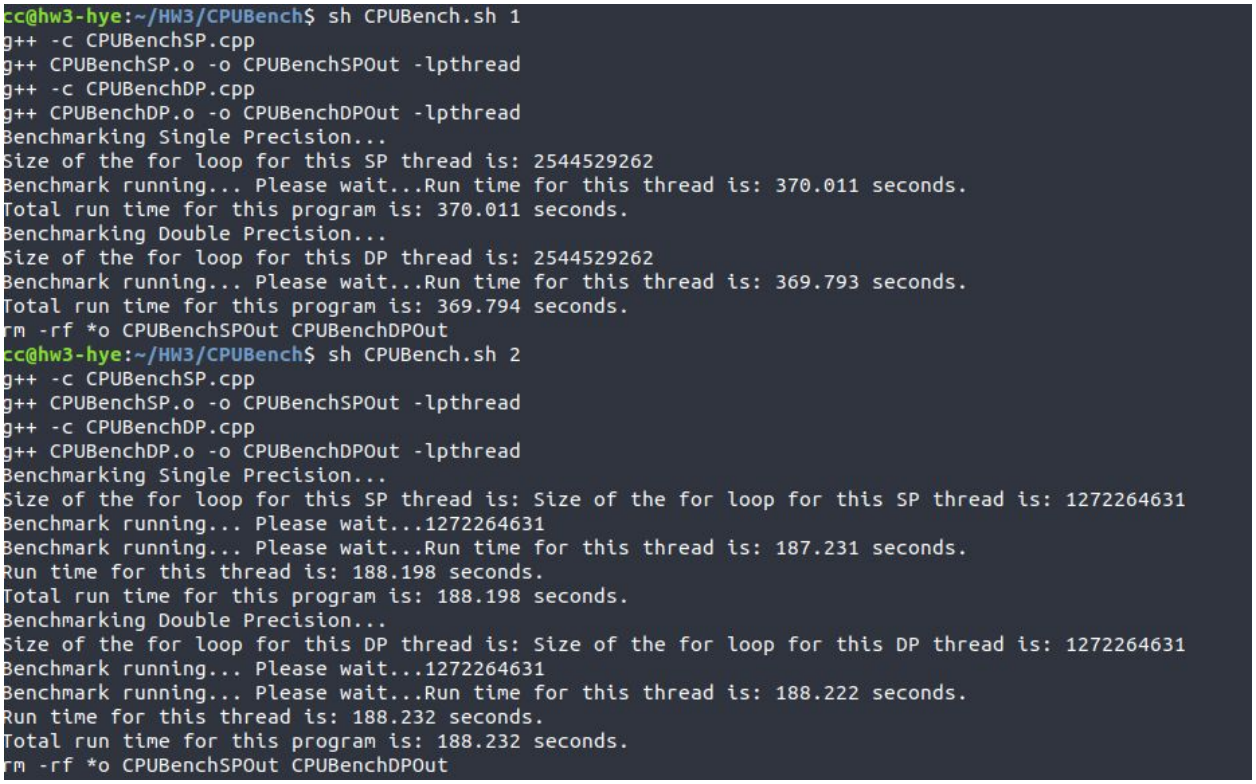
Part 1. CPUBench:

Experiment screenshots:



```
cc@hw3-hye:~/HW3/CPUBench$ sh CPUBench.sh 1
g++ -c CPUBenchSP.cpp
g++ CPUBenchSP.o -o CPUBenchSPOut -lpthread
g++ -c CPUBenchDP.cpp
g++ CPUBenchDP.o -o CPUBenchDPOut -lpthread
Benchmarking Single Precision...
Size of the for loop for this SP thread is: 2544529262
Benchmark running... Please wait...Run time for this thread is: 370.011 seconds.
Total run time for this program is: 370.011 seconds.
Benchmarking Double Precision...
Size of the for loop for this DP thread is: 2544529262
Benchmark running... Please wait...Run time for this thread is: 369.793 seconds.
Total run time for this program is: 369.794 seconds.
rm -rf *o CPUBenchSPOut CPUBenchDPOut
cc@hw3-hye:~/HW3/CPUBench$ sh CPUBench.sh 2
g++ -c CPUBenchSP.cpp
g++ CPUBenchSP.o -o CPUBenchSPOut -lpthread
g++ -c CPUBenchDP.cpp
g++ CPUBenchDP.o -o CPUBenchDPOut -lpthread
Benchmarking Single Precision...
Size of the for loop for this SP thread is: Size of the for loop for this SP thread is: 1272264631
Benchmark running... Please wait...1272264631
Benchmark running... Please wait...Run time for this thread is: 187.231 seconds.
Run time for this thread is: 188.198 seconds.
Total run time for this program is: 188.198 seconds.
Benchmarking Double Precision...
Size of the for loop for this DP thread is: Size of the for loop for this DP thread is: 1272264631
Benchmark running... Please wait...1272264631
Benchmark running... Please wait...Run time for this thread is: 188.222 seconds.
Run time for this thread is: 188.232 seconds.
Total run time for this program is: 188.232 seconds.
rm -rf *o CPUBenchSPOut CPUBenchDPOut
```

```
cc@hw3-hye:~/HW3/CPUBench$ sh CPUBench.sh 4
g++ -c CPUBenchSP.cpp
g++ CPUBenchSP.o -o CPUBenchSPOut -lpthread
g++ -c CPUBenchDP.cpp
g++ CPUBenchDP.o -o CPUBenchDPOut -lpthread
Benchmarking Single Precision...
Size of the for loop for this SP thread is: Size of the for loop for this SP thread is: 636132315636132315
Benchmark running... Please wait...Size of the for loop for this SP thread is: 636132315
Benchmark running... Please wait...
Benchmark running... Please wait...Size of the for loop for this SP thread is: 636132315
Benchmark running... Please wait...Run time for this thread is: 97.1143 seconds.
Run time for this thread is: 97.1535 seconds.
Run time for this thread is: 97.2022 seconds.
Run time for this thread is: 97.2471 seconds.
Total run time for this program is: 97.2491 seconds.
Benchmarking Double Precision...
Size of the for loop for this DP thread is: Size of the for loop for this DP thread is: 636132315636132315
Benchmark running... Please wait...
Benchmark running... Please wait...Size of the for loop for this DP thread is: 636132315
Benchmark running... Please wait...Size of the for loop for this DP thread is: 636132315
Benchmark running... Please wait...Run time for this thread is: 97.0049 seconds.
Run time for this thread is: 97.0133 seconds.
Run time for this thread is: 97.0314 seconds.
Run time for this thread is: 97.0726 seconds.
Total run time for this program is: 97.0731 seconds.
rm -rf *o CPUBenchSPOut CPUBenchDPOut
```

```
cc@hw3-hye:~/HW3/CPUBench$ sh CPUBench.sh 8
g++ -c CPUBenchSP.cpp
g++ CPUBenchSP.o -o CPUBenchSPOut -lpthread
g++ -c CPUBenchDP.cpp
g++ CPUBenchDP.o -o CPUBenchDPOut -lpthread
Benchmarking Single Precision...
Size of the for loop for this SP thread is: Size of the for loop for this SP thread is: 318066157318066157
Size of the for loop for this SP thread is: 318066157
Benchmark running... Please wait...Benchmark running... Please wait...Size of the for loop for this SP thread is: 318066157
Size of the for loop for this SP thread is: Size of the for loop for this SP thread is: 318066157
Benchmark running... Please wait...Benchmark running... Please wait...
Benchmark running... Please wait...Size of the for loop for this SP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this SP thread is: 318066157
Benchmark running... Please wait...318066157
Benchmark running... Please wait...Run time for this thread is: 96.7762 seconds.
Run time for this thread is: 96.8304 seconds.
Run time for this thread is: 96.9096 seconds.
Run time for this thread is: 96.939 seconds.
Run time for this thread is: 96.9622 seconds.
Run time for this thread is: 97.0518 seconds.
Run time for this thread is: 97.1214 seconds.
Run time for this thread is: 97.1165 seconds.
Total run time for this program is: 97.1288 seconds.
Benchmarking Double Precision...
Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Size of the for loop for this DP thread is: 318066157
Benchmark running... Please wait...Run time for this thread is: 96.5988 seconds.
Run time for this thread is: 96.6016 seconds.
Run time for this thread is: 96.7445 seconds.
Run time for this thread is: 96.7647 seconds.
Run time for this thread is: 96.7948 seconds.
Run time for this thread is: 96.9234 seconds.
Run time for this thread is: 96.9084 seconds.
Run time for this thread is: 96.9508 seconds.
Total run time for this program is: 96.9601 seconds.
rm -rf *o CPUBenchSPOut CPUBenchDPOut
```

Part 2. Matrix Multiplication:

For this part, I almost finished the code but there is problem with the how to set the code to get correct number of instruction that ran in the program in order to get the correct GigaOPS result. This is the part that I failed to figure out about how to count the number of instruction:

cout << "Processor speed is: " << pow(m, 3)/ 1000000000 / elapsed << " GigaOPS." <<endl;

The run time scales on multiple threads correctly according to the 4 core CPU.
But since I cannot get a correct number of instructions, I cannot get a precise result for CPU speed.

```
cc@hw3-hye:~/homework3/MatrixBench$ ./spout 4
Please enter side length of the matrices (has to be less than 23170):1200
Matrix1 has been initialized.
Matrix2 has been initialized.
Total run time for this program is: 6.28142 seconds.
Processor speed is: 0.00429839 GigaOPS.
cc@hw3-hye:~/homework3/MatrixBench$ ./spout 1
Please enter side length of the matrices (has to be less than 23170):1200
Matrix1 has been initialized.
Matrix2 has been initialized.
Total run time for this program is: 23.5981 seconds.
Processor speed is: 0.0732263 GigaOPS.
```

I think there are some problems with my code that needs to solved.
And for CPUBench part, the performance is bit higher than theoretical value which is 2.3 Ghz. There could be some problem about the code, too.

Part 3. HPLBench

I have successfully installed and built mHPL and tuned the HPL.data file with 4 cores and required ram size.
But the shpl/xhpl did not run successfully.

Output:

cc@hw3-hye:~/mHPL-1.0/bin/ORION$ mpirun -np 4 shpl

[[28129,1],1]: A high-performance Open MPI point-to-point messaging module
was unable to find any relevant network interfaces:

Module: OpenFabrics (openib)
  Host: hw3-hye

Another transport will be used instead, although this may result in
lower performance.

==
HPLinpack 1.0a -- High-Performance Linpack benchmark --   January 20, 2004
Written by A. Petitet and R. Clint Whaley, Innovative Computing Labs., UTK
==

An explanation of the input/output parameters follows:
T/V   : Wall time / encoded variant.
N   : The order of the coefficient matrix A.
NB   : The partitioning blocking factor.
P   : The number of process rows.
Q   : The number of process columns.
Time   : Time in seconds to solve the linear system.
Gflops : Rate of execution for solving the linear system.

The following parameter values will be used:

N   :   25344
NB   :   192
PMAP   : Row-major process mapping
P   :   2
Q   :   2
PFACT :   Right
NBMIN :   4
NDIV   :   2
RFACT :   Crout
BCAST : 1ringM
DEPTH :   1
SWAP   : Mix (threshold = 64)
L1   : transposed form
U   : transposed form
EQUIL : yes
ALIGN : 8 float precision words

- The matrix A is randomly generated for each test.
- The following scaled residual checks will be computed:

```
   1) ||Ax-b||_oo / ( eps * ||A||_1  * N             )
   2) ||Ax-b||_oo / ( eps * ||A||_1  * ||x||_1  )
   3) ||Ax-b||_oo / ( eps * ||A||_oo * ||x||_oo )
- The relative machine precision (eps) is taken to be           5.960464e-08
- Computational tests pass if scaled residuals are less than            16.0

[hw3-hye:01519] 3 more processes have sent help message help-mpi-btl-base.txt / btl:no-nics
[hw3-hye:01519] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error
messages
--------------------------------------------------------------------------
mpirun noticed that process rank 1 with PID 1522 on node hw3-hye exited on signal 11
(Segmentation fault).
--------------------------------------------------------------------------
```

Workload	Concurrency	CPUBench Measured Ops/Sec (GigaOPS)	MatrixBench Measured Ops/Sec (GigaOPS)	HPL Measured Ops/Sec (GigaOPS)	Theoretical Ops/Sec (GigaOPS)	CPUBench Efficiency (%)	MatrixBench Efficiency (%)	HPL Efficiency (%)
SP	1	2.70				117%		
SP	2	5.31				115%		
SP	4	10.28				108%		
SP	8	10.30				112%		
DP	1	2.70				117%		
DP	2	5.31				115%		
DP	4	10.30				108%		
DP	8	10.31				117%		

I cannot fill in the other parts of the table because I got stuck with figuring out the number of instructions run so to get the CPU speed.