Name: Fan Feng
Assignment: 1
Course: CPSC 424/524

Modules:

```
[ff242@omega2 ~]$ module list

Currently Loaded Modules:
  1) StdEnv (S)    2) Langs/Intel/15.0.2

  Where:
   S:  Module is Sticky, requires --force to unload or purge
```

Environment:

```
[ff242@omega2 ~]$ which icc
/home/apps/fas/Langs/Intel/2015_update2/composer_xe_2015.2.164/bin/intel64/icc
[ff242@omega2 ~]$ icc --version
icc (ICC) 15.0.2 20150121
Copyright (C) 1985-2015 Intel Corporation.  All rights reserved.
```

Commands:

srun --pty -p interactive -c 1 -t 6:00:00 --mem-per-cpu=20gb bash
./run.sh

Output:

See appendix at the end of this report.

Exercise 1 (Approximation to Pi):

The approximation to Pi is given in the first line of each result in the output. We can demonstrate that this is a correct approximation by evaluating *sin(Pi)*, which is expected to be 0.

Here is an explanation to the difference between the four compiler options:

a. -g -O0 -fno-alias -std=c99
b. -g -O1 -fno-alias -std=c99
c. -g -O3 -no-vec -no-simd -fno-alias -std=c99
d. -g -O3 -xHost -fno-alias -std=c99

According to *man icc*, https://software.intel.com/sites/default/files/m/d/4/1/d/8/icc.txt and https://software.intel.com/en-us/articles/introduction-to-intel-advanced-vector-extensions:

(a) and (b) differ from the optimization option. -O0 disables optimization while -O1 optimizes to favor code size and code locality. As stated on the manual page: "-O1 may improve performance for applications with very large code size, many branches, and execution time not dominated by code within loops". Since none of the three situations applies to our code, (b) doesn't improve the performance much.

(c) uses -O3 optimization, which performs the default -O2 optimization, plus enabling more aggressive optimizations such as loop and memory access transformation, and prefetching. It is recommended for applications that have loops with heavy use of floating point calculations and process large data sets. Thus, (c) should improve the performance. However, since *-no-vec* and *-no-simd* effectively prevent vectorization and compiler interpretation of SIMD pragmas from happening, (c) ends up not improving performance much either.

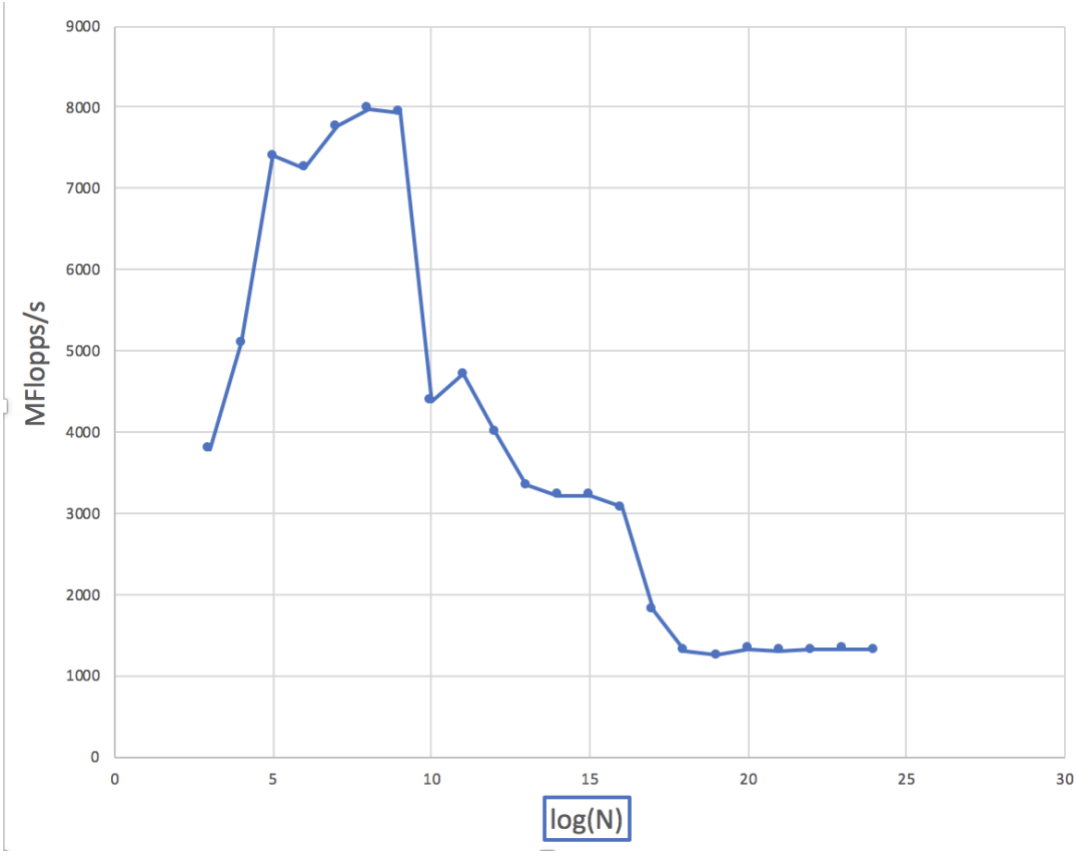
(d) fully takes advantage of -O3 optimization, and in addition -xHost tells compiler to generate instructions for the highest instruction set, thus option (d) improves the performance the most. Also note that the performance almost doubles, and that is because the Intel AVX expands the SIMD registers from 128 bits to 256 bits.

Exercise 1 (Latency of the Divide Operation):
One billion division operations take about 7.2 seconds.

(2.8 * 1,000,000,000 * 7.2) / (1,000,000,000) = 20.16

The latency of each division operation is about 20.16 cycles.

Exercise 2 (Vector Triad Performance):



Initially, as N increases, MFlops/s increases because the CPUs have not been fully utilized. During this phase, neither CPU nor cache is the bottleneck, the number of operations is.

[ff242@c31n15 ff242_ps1_cpsc424]$ lscpu
Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit

Byte Order: Little Endian
CPU(s): 8
On-line CPU(s) list: 0-7
Thread(s) per core: 1
Core(s) per socket: 4
Socket(s): 2
NUMA node(s): 2
Vendor ID: GenuineIntel
CPU family: 6
Model: 26
Model name: Intel(R) Xeon(R) CPU X5560 @ 2.80GHz
Stepping: 5
CPU MHz: 1600.000
CPU max MHz: 2800.0000
CPU min MHz: 1600.0000
BogoMIPS: 5600.28
Virtualization: VT-x
L1d cache: 32K
L1i cache: 32K
L2 cache: 256K
L3 cache: 8192K
NUMA node0 CPU(s): 1,3,5,7
NUMA node1 CPU(s): 0,2,4,6
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 rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf pni dtes64 monitor ds_cpl vmx est tm2 ssse3 cx16 xtpr pdcm dca sse4_1 sse4_2 popcnt lahf_lm spec_ctrl ibpb_support tpr_shadow vnmi flexpriority ept vpid dtherm ida

The first decrease happens between N = 794 and N = 1667. Respectively, the space needed for 4 vectors are 4*794*8 = 25 KB, and 4*1667*8 = 52 KB. The decrease happens because L1 cache has size 32 KB, which is between 25 KB and 52 KB.

The second decrease happens between N = 7355 and N = 15447. Respectively, the space needed is 229 KB and 483 KB. The decrease happens because the L2 cache has size 256 KB, which is between 229 KB and 483 KB.

The third decrease happens between N = 143056 and N = 300419.

Respectively, the space needed is 4.4 MB and 9.2 MB. The decrease happens because the L3 cache has size 8 MB, which is between 4.4 MB and 9.2 MB.

Appendix:

```
[ff242@c15n02 pa1]$ ./run.sh

==================== make ================================================
icc -g -O0 -fno-alias -std=c99 -o ex1_a ex1_pi.c timing.o
icc -g -O1 -fno-alias -std=c99 -o ex1_b ex1_pi.c timing.o
icc -g -O3 -no-vec -no-simd -fno-alias -std=c99 -o ex1_c ex1_pi.c timing.o
icc -g -O3 -xHost -fno-alias -std=c99 -o ex1_d ex1_pi.c timing.o
icc -g -O0 -fno-alias -std=c99 -o ex1_div ex1_div.c timing.o
icc -g -O3 -xHost -fno-alias -std=c99 -o ex2 ex2.c dummy.c timing.o

==================== Exercise 1 - Compiler Option (a) ====================
Pi = 3.141593, sin(Pi) = 0.000000
Time spent = 7.208700
MFlops/s = 693.606319

==================== Exercise 1 - Compiler Option (b) ====================
Pi = 3.141593, sin(Pi) = 0.000000
Time spent = 7.208866
MFlops/s = 693.590353

==================== Exercise 1 - Compiler Option (c) ====================
Pi = 3.141593, sin(Pi) = 0.000000
Time spent = 7.208311
MFlops/s = 693.643760

==================== Exercise 1 - Compiler Option (d) ====================
Pi = 3.141593, sin(Pi) = 0.000000
Time spent = 3.603819
MFlops/s = 1387.417112

==================== Exercise 1 - Division Operation Latency =============
Time spent = 7.207881
Number of division operations = 1000000000

==================== Exercise 2 - Vector Triad Performance ===============
N = 9, MFlops/s = 3662.686654
N = 19, MFlops/s = 5042.825211
N = 40, MFlops/s = 7398.651117
N = 85, MFlops/s = 7253.176954
N = 180, MFlops/s = 7792.955632
N = 378, MFlops/s = 7982.839889
N = 794, MFlops/s = 7936.389164
N = 1667, MFlops/s = 4389.718685
N = 3502, MFlops/s = 4753.765841
N = 7355, MFlops/s = 4100.780358
```

N = 15447, MFlops/s = 3367.075034
N = 32439, MFlops/s = 3226.444891
N = 68122, MFlops/s = 3217.002601
N = 143056, MFlops/s = 3079.587605
N = 300419, MFlops/s = 1776.934042
N = 630880, MFlops/s = 1309.882236
N = 1324849, MFlops/s = 1301.287279
N = 2782184, MFlops/s = 1315.433999
N = 5842587, MFlops/s = 1309.898488
N = 12269432, MFlops/s = 1327.067479
N = 25765808, MFlops/s = 1324.045796
N = 54108198, MFlops/s = 1324.441703
N = 113627216, MFlops/s = 1334.188112

========== Clean up ==========
rm ex1_a ex1_b ex1_c ex1_d ex1_div ex2