# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

This project aims to benchmark different parts of a computer system, from the CPU, memory and disk.

This assignment ran on Amazon ec2 t2.microinstance with CPU specifications as follow.

OS: Ubuntu 14.04

```
ubuntu@ip-172-31-37-205:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 62
model name      : Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz
stepping        : 4
microcode       : 0x416
cpu MHz         : 2500.040
cache size      : 25600 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
```

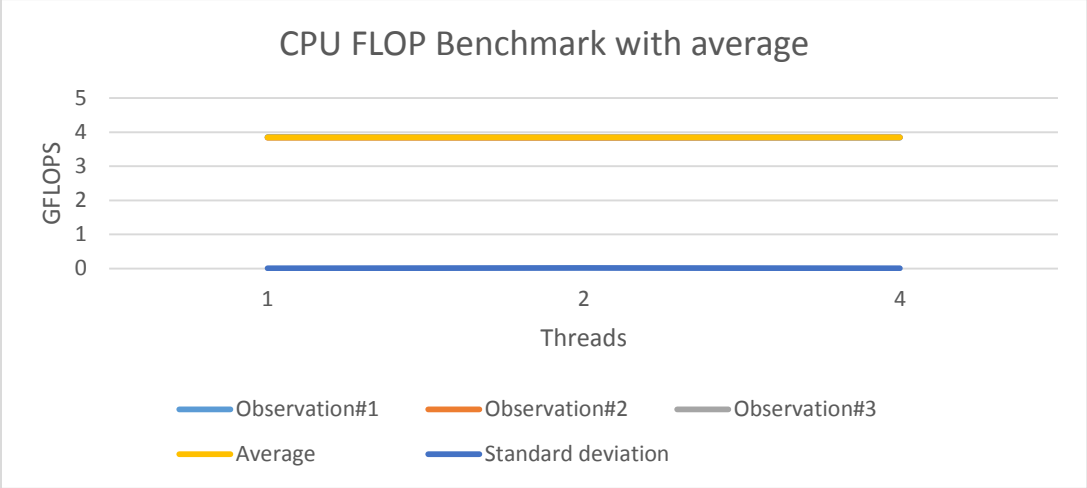
1. CPU
   a. Measure the processor speed, in terms of floating point operations per second (Giga FLOPS, $10^9$ FLOPS) and integer operations per second (Giga IOPS, $10^2$ IOPS).
   b. Measure the processor speed at varying levels of concurrency (1 thread, 2 threads, 4 threads)

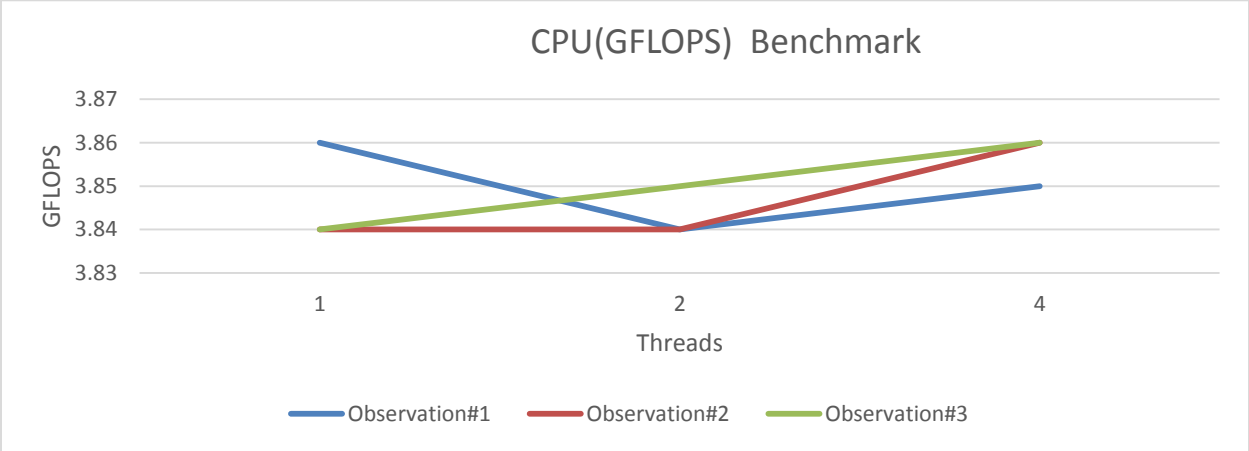
   Floating point operation readings are as follows.

GFLOPS

| Threads | Observation#1 | Observation#2 | Observation#3 | Average | Standard deviation |
|---|---|---|---|---|---|
| 1 | 3.86 | 3.84 | 3.85 | 3.85 | 0.008164966 |
| 2 | 3.84 | 3.84 | 3.86 | 3.846666 | 0.00942809 |
| 4 | 3.84 | 3.85 | 3.86 | 3.85 | 0.008164966 |

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

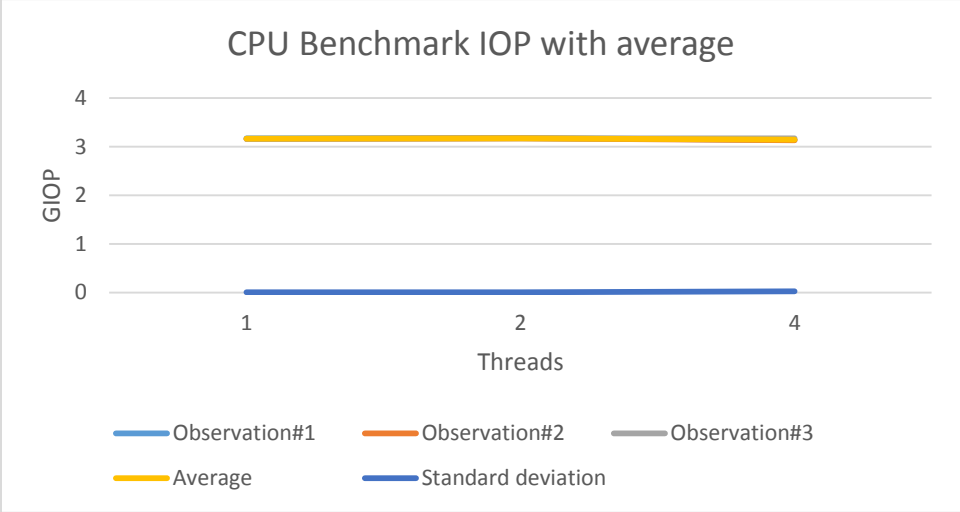


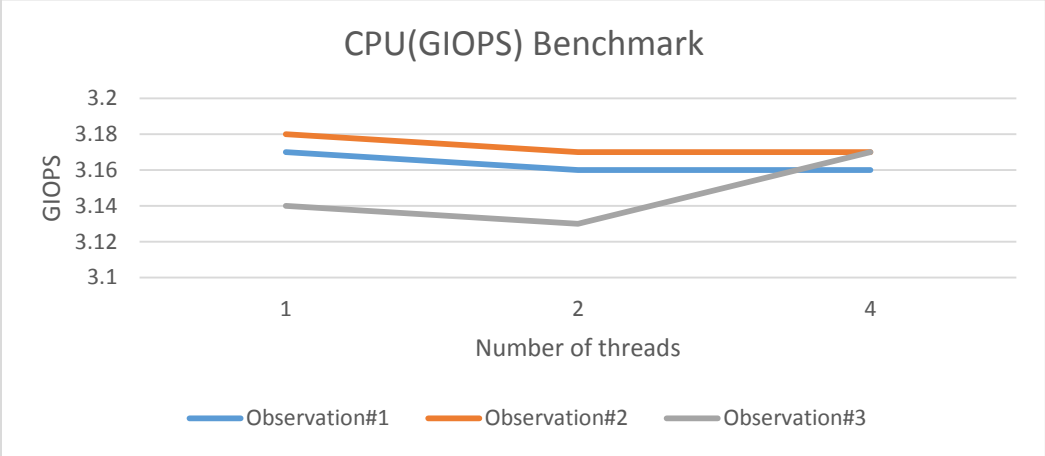




IOP:

| Threads | Observation#1 | Observation#2 | Observation#3 | Average | Standard deviation |
|---|---|---|---|---|---|
| 1 | 3.17 | 3.16 | 3.16 | 3.16333 | 0.005773503 |
| 2 | 3.18 | 3.17 | 3.17 | 3.17333 | 0.005773503 |
| 4 | 3.14 | 3.13 | 3.17 | 3.14666 | 0.02081666 |

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation




c. Compute the theoretical peak performance of your processor in flops/sec.

Intel ® Xeon® CPU ES-2600 series have 8 instructions per cycle.

Theoretical peak performance can be calculated by below formula:
**=Number of Cores ∗ CPU frequency ∗ Instructions per cycle * CPU per node**
= 1 * 2.50 Ghz * 8 * 1= 20 GFLOPS.

d. What efficiency do you achieve compared to the theoretical performance?
Max GFLOPS achieved through my assignment is 4.55 GFLOPS.
Theoretical GFLOPS is 20 GFLOPS
So 4.55/20=22.75 % of total peak performance
Note: As I increases complexity of instructions, GFLOPS also started increasing.

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

e. As a separate experiment, run your benchmark on floating point and integer instructions and 4 threads for a 10-minute period for each one, and take samples every second on how many instructions per second were achieved during the experiment; plot the data for the two experiments (FLOPS and IOPS) with time (0to 10 min) on the x-axis and FLOPS/IOPS on the y-axis, with 1-second samples (you will have 600 samples for FLOPS and 600 samples for IOPS to plot)

Data files with 600 samples are placed in current folder.

600 GFLOPS sample values are as follows in the order of 1 to 600 seconds.

Threads count 4

3.747727 ,3.762418 ,3.731740 ,3.759607 ,3.750117 ,3.766586 ,3.780819 ,3.752881 ,3.780479 ,3.762037
,3.775413 ,3.781755 ,3.760508 ,3.777875 ,3.763766 ,3.761791 ,3.763042 ,3.740995 ,3.744871 ,3.727967
,3.731830 ,3.741043 ,4.669968 ,3.735837 ,3.713170 ,3.702730 ,3.726917 ,3.718395 ,3.752931 ,3.723261
,3.748088 ,3.744126 ,3.727324 ,3.749120 ,3.724675 ,3.735490 ,3.753588 ,3.741054 ,3.744405 ,3.727035
,3.739878 ,3.746921 ,3.762547 ,3.756916 ,3.718156 ,3.743129 ,3.748116 ,3.760683 ,3.761022 ,3.718658
,3.723798 ,3.742074 ,3.726205 ,3.741992 ,3.696995 ,3.711592 ,3.714736 ,3.721720 ,3.735558 ,3.696174
,3.710899 ,3.725614 ,3.687804 ,3.716526 ,3.687160 ,3.691806 ,3.686337 ,3.675524 ,3.677686 ,3.672612
,3.696862 ,3.743959 ,3.709168 ,3.734920 ,3.716689 ,3.732735 ,3.720132 ,3.720561 ,3.732566 ,3.713857
,3.720478 ,3.707944 ,3.744691 ,3.749163 ,3.742143 ,3.744396 ,3.708393 ,3.701066 ,3.714122 ,3.706279
,3.725858 ,3.698224 ,3.721889 ,3.724193 ,3.695671 ,3.712588 ,3.719072 ,3.710946 ,3.710477 ,3.715153
,3.735619 ,3.717788 ,3.738571 ,3.745917 ,3.725736 ,3.740824 ,3.736822 ,3.718627 ,3.729047 ,3.722337
,3.737020 ,3.702359 ,3.708331 ,3.731694 ,3.707834 ,3.714797 ,3.721916 ,3.716092 ,3.717741 ,3.696019
,3.719851 ,3.708110 ,3.694446 ,3.711391 ,3.722304 ,3.676821 ,3.720510 ,3.691746 ,3.718074 ,3.671096
,3.714196 ,3.702716 ,3.693657 ,3.708207 ,3.718236 ,3.696727 ,3.706142 ,3.685457 ,3.696828 ,3.681399
,3.715154 ,3.718645 ,3.712940 ,3.726979 ,3.721413 ,3.705698 ,3.722706 ,3.701559 ,3.712675 ,3.717999
,3.730916 ,3.731578 ,3.720101 ,3.725911 ,3.718636 ,3.714666 ,3.733294 ,3.711326 ,3.730831 ,3.711656
,3.759138 ,3.750870 ,3.740710 ,3.743489 ,3.737129 ,3.739043 ,3.741181 ,3.714471 ,3.723082 ,3.698355
,3.723414 ,3.731473 ,3.711065 ,3.722165 ,3.704283 ,3.712826 ,3.718611 ,3.702641 ,3.748060 ,3.726924
,3.703963 ,3.672192 ,3.677530 ,3.704846 ,3.699636 ,3.719030 ,3.721264 ,3.687621 ,3.704390 ,3.706764
,3.712663 ,3.731737 ,3.713633 ,3.731213 ,3.720163 ,3.718889 ,3.726728 ,3.730431 ,3.732673 ,3.732368
,3.719230 ,3.736373 ,3.712764 ,3.760919 ,3.736589 ,3.726461 ,3.736525 ,3.718599 ,3.716155 ,3.725479
,3.724862 ,3.729371 ,3.710204 ,3.720208 ,3.715851 ,3.709942 ,3.745370 ,3.750498 ,3.705928 ,3.726705
,3.712190 ,3.714955 ,3.709592 ,3.738349 ,3.717637 ,3.697514 ,3.727089 ,3.734218 ,3.721894 ,3.750478
,3.731587 ,3.736030 ,3.710211 ,3.749483 ,3.746595 ,3.719804 ,3.728126 ,3.732032 ,3.728325 ,3.717987
,3.703988 ,3.728169 ,3.710979 ,3.707288 ,3.754849 ,3.715352 ,3.726193 ,3.725329 ,3.711838 ,3.730877
,3.693813 ,3.713286 ,3.709075 ,3.721484 ,3.725887 ,3.705232 ,3.731569 ,3.709191 ,3.735088 ,3.741789
,3.727088 ,4.601570 ,3.748450 ,3.708448 ,3.735943 ,3.723252 ,3.714283 ,3.732644 ,3.724688 ,3.735416
,3.719557 ,3.720868 ,3.756722 ,3.724163 ,3.714546 ,3.731216 ,3.713361 ,3.716358 ,3.710238 ,3.719535
,3.717009 ,3.714712 ,3.723148 ,3.726155 ,3.729912 ,3.746266 ,3.719156 ,3.726592 ,3.735270 ,3.710507
,3.708789 ,3.700123 ,3.710372 ,3.725966 ,3.723130 ,3.732137 ,3.701978 ,3.731453 ,3.733941 ,3.733217
,3.700174 ,3.719470 ,3.725113 ,3.718177 ,3.711973 ,3.732102 ,3.701137 ,3.723987 ,3.727412 ,3.701258
,3.706319 ,3.718514 ,3.740363 ,3.697650 ,3.721402 ,3.749757 ,3.712643 ,3.728681 ,3.741150 ,3.720658

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

,3.740113 ,3.703695 ,3.733996 ,3.727306 ,3.730135 ,3.727892 ,3.717689 ,3.733313 ,3.737702 ,3.720319
,3.720702 ,3.706013 ,3.722088 ,3.710902 ,3.734336 ,3.744958 ,3.733198 ,3.732650 ,3.749070 ,3.710959
,3.718691 ,3.724427 ,3.696884 ,3.736622 ,3.726177 ,3.728245 ,3.693682 ,3.713307 ,3.738389 ,3.711757
,3.724725 ,3.726532 ,3.713117 ,3.724831 ,3.713823 ,3.737247 ,3.722852 ,3.740337 ,3.737253 ,3.706420
,3.706021 ,3.704190 ,3.682107 ,3.724861 ,3.694281 ,3.708894 ,3.729737 ,3.746953 ,3.732515 ,3.722552
,3.732455 ,3.739771 ,3.731821 ,3.724987 ,3.715004 ,3.746924 ,3.718866 ,3.729132 ,3.753577 ,3.712334
,3.734874 ,3.730990 ,3.713454 ,3.742215 ,3.712575 ,3.732399 ,3.710430 ,3.737712 ,3.749802 ,3.715102
,3.730588 ,3.736583 ,3.728177 ,3.727439 ,3.700147 ,3.726214 ,3.726128 ,3.714253 ,3.726930 ,3.711552
,3.737554 ,3.726811 ,3.704904 ,3.727515 ,3.701414 ,3.718961 ,3.729376 ,3.693783 ,3.720380 ,3.708279
,3.731815 ,3.719066 ,3.727510 ,3.726650 ,3.708032 ,3.728075 ,3.725471 ,3.709632 ,3.742457 ,3.697721
,3.704863 ,3.699252 ,3.698371 ,3.686986 ,3.710245 ,3.710817 ,3.702903 ,3.711006 ,3.744639 ,3.727647
,3.709771 ,3.734428 ,3.713793 ,3.739984 ,3.719968 ,3.709908 ,3.723345 ,3.717870 ,3.755559 ,3.734723
,3.734931 ,3.749036 ,3.721397 ,3.740003 ,3.741386 ,3.718333 ,3.742600 ,3.688894 ,3.702282 ,3.716173
,3.713460 ,3.731816 ,3.718835 ,3.710485 ,3.713641 ,3.726018 ,3.741562 ,3.722855 ,3.739517 ,3.705381
,3.689295 ,3.735165 ,3.733120 ,3.747885 ,3.742284 ,3.706372 ,3.735112 ,3.718745 ,3.717453 ,3.714060
,3.716118 ,3.726212 ,3.702466 ,3.723692 ,3.732869 ,3.721093 ,3.725088 ,3.719555 ,3.728084 ,3.727692
,3.729535 ,3.753211 ,3.745523 ,3.735706 ,3.714379 ,3.691962 ,3.706566 ,3.702974 ,3.701199 ,3.713019
,3.714050 ,3.722743 ,3.706432 ,3.706718 ,3.710519 ,3.709789 ,3.743536 ,3.737007 ,3.732429 ,3.690174
,3.723785 ,3.750725 ,3.732102 ,3.748595 ,3.745062 ,3.728224 ,3.771779 ,3.778062 ,3.759259 ,3.770714
,3.755283 ,3.775942 ,3.773899 ,3.756197 ,3.762919 ,3.766556 ,3.777513 ,3.720603 ,3.743557 ,3.738269
,3.711821 ,3.731169 ,3.740941 ,3.743031 ,3.747051 ,3.726588 ,3.728646 ,3.734734 ,3.709769 ,3.742516
,3.705835 ,3.714560 ,3.721679 ,3.714301 ,3.722730 ,3.697808 ,3.750780 ,3.767625 ,3.734102 ,3.735386
,3.700743 ,3.729049 ,3.697627 ,3.701521 ,3.742126 ,3.705233 ,3.739447 ,3.737723 ,3.738346 ,3.755471
,3.715133 ,3.738206 ,3.737056 ,3.715720 ,3.739054 ,3.709950 ,3.725566 ,3.720447 ,3.719877 ,3.730207
,3.711000 ,3.726412 ,3.726719 ,3.711255 ,3.717046 ,3.698486 ,3.723432 ,3.714709 ,3.716229 ,3.730330
,3.698693 ,3.715579 ,3.725154 ,3.716884 ,3.736985 ,3.718406 ,3.724782 ,3.691635 ,3.706014 ,3.702191
,3.676141 ,3.705454 ,3.715730 ,3.705691 ,3.701325 ,3.703182 ,3.708339 ,3.697623 ,3.728714 ,3.720974
,3.684420 ,3.708972 ,3.698270 ,3.731629 ,3.720624 ,3.680438 ,3.699198 ,3.680366 ,3.707035 ,3.701620

600 GIOP values in the order of 1 to 600 sec are as follows.

Threads count 4

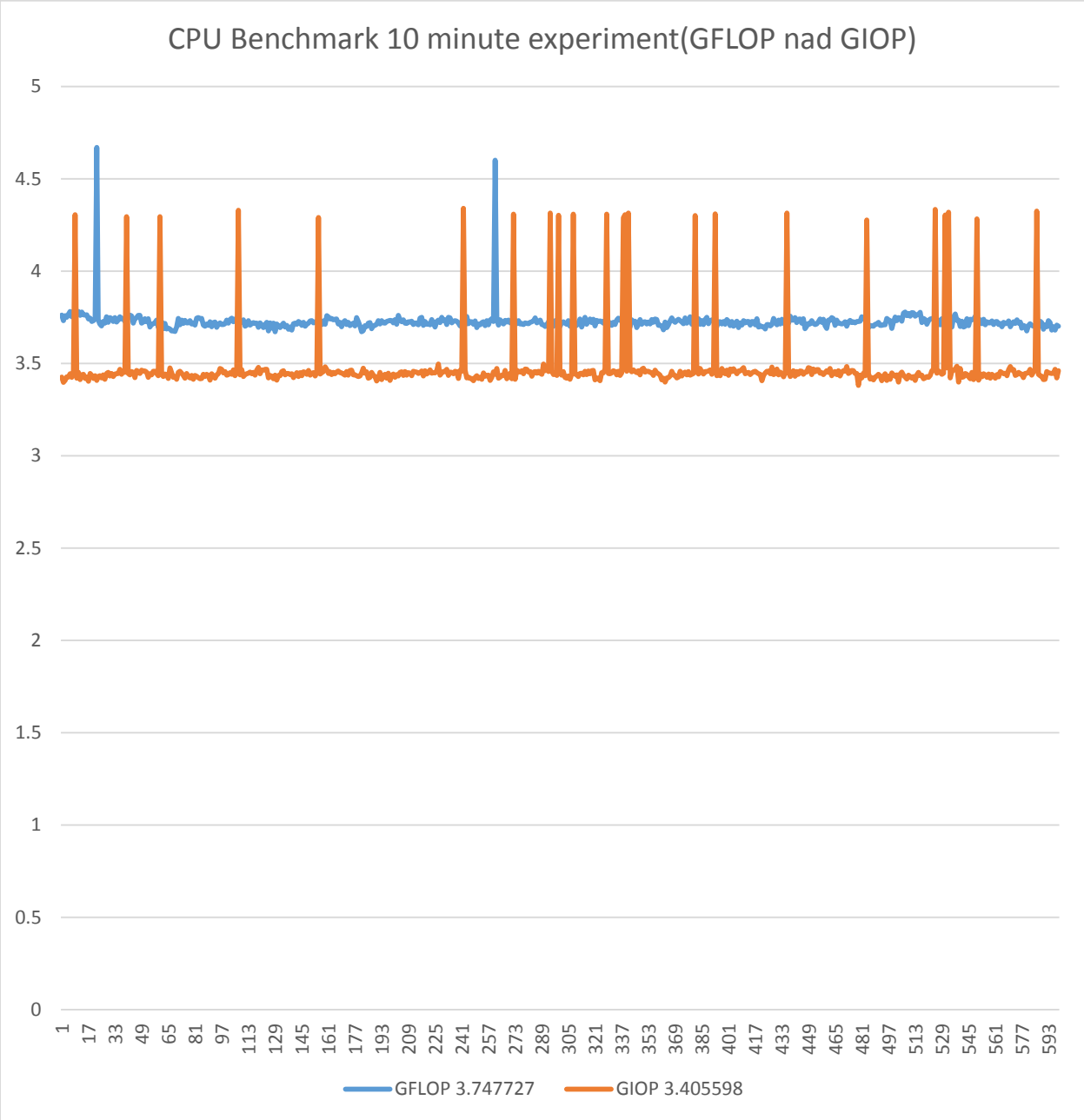
3.405598 ,3.425411 ,3.397916 ,3.408322 ,3.427081 ,3.426834 ,3.444529 ,3.425313 ,3.454451 ,4.305624
,3.420757 ,3.451109 ,3.412842 ,3.437546 ,3.441167 ,3.418314 ,3.425702 ,3.403665 ,3.446065 ,3.434533
,3.418084 ,3.433651 ,3.407685 ,3.429187 ,3.433716 ,3.421616 ,3.438720 ,3.414660 ,3.445899 ,3.452153
,3.432287 ,3.446325 ,3.429325 ,3.442343 ,3.444372 ,3.445977 ,3.467545 ,3.441028 ,3.459894 ,3.451843
,4.295765 ,3.445616 ,3.437515 ,3.455991 ,3.452353 ,3.423434 ,3.461882 ,3.456446 ,3.445226 ,3.463264
,3.457284 ,3.459768 ,3.453630 ,3.425154 ,3.443595 ,3.438126 ,3.458399 ,3.457813 ,3.445330 ,3.434227
,4.295653 ,3.443096 ,3.428369 ,3.442392 ,3.439352 ,3.418414 ,3.475363 ,3.449913 ,3.424345 ,3.428850
,3.413596 ,3.452158 ,3.457856 ,3.447604 ,3.434990 ,3.420197 ,3.446106 ,3.429166 ,3.421605 ,3.434723
,3.413562 ,3.434471 ,3.431014 ,3.423507 ,3.417078 ,3.419250 ,3.441588 ,3.429541 ,3.441369 ,3.434748
,3.410262 ,3.443793 ,3.433776 ,3.419878 ,3.438171 ,3.444211 ,3.473441 ,3.468110 ,3.448420 ,3.456086

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

,3.436606 ,3.439593 ,3.449690 ,3.440873 ,3.465245 ,3.434122 ,3.455244 ,4.329433 ,3.438273 ,3.466030
,3.428878 ,3.440340 ,3.444480 ,3.446525 ,3.452904 ,3.444454 ,3.451788 ,3.437034 ,3.463683 ,3.479165
,3.444517 ,3.465744 ,3.463166 ,3.467786 ,3.468383 ,3.421803 ,3.434817 ,3.431134 ,3.416643 ,3.436461
,3.409671 ,3.434965 ,3.434205 ,3.456942 ,3.460320 ,3.437867 ,3.445555 ,3.445364 ,3.440102 ,3.443922
,3.423312 ,3.436091 ,3.450020 ,3.429303 ,3.449586 ,3.434333 ,3.454006 ,3.452410 ,3.440748 ,3.461784
,3.436389 ,3.447242 ,3.450843 ,3.432838 ,3.449488 ,4.290218 ,3.443305 ,3.450589 ,3.456834 ,3.481581
,3.466305 ,3.445144 ,3.454615 ,3.439970 ,3.452074 ,3.443567 ,3.452550 ,3.458837 ,3.449512 ,3.448228
,3.443031 ,3.436939 ,3.456730 ,3.437266 ,3.462432 ,3.463912 ,3.437432 ,3.438208 ,3.434048 ,3.429034
,3.431853 ,3.448573 ,3.474536 ,3.447568 ,3.454963 ,3.419672 ,3.424667 ,3.451941 ,3.437243 ,3.428991
,3.405621 ,3.419383 ,3.452229 ,3.412329 ,3.434791 ,3.441888 ,3.419914 ,3.446676 ,3.408352 ,3.435534
,3.442258 ,3.425801 ,3.440355 ,3.431737 ,3.451555 ,3.454120 ,3.434722 ,3.455875 ,3.436389 ,3.450669
,3.447379 ,3.433393 ,3.457685 ,3.454860 ,3.453236 ,3.445643 ,3.449869 ,3.459079 ,3.432524 ,3.460912
,3.441924 ,3.449466 ,3.464770 ,3.447306 ,3.450155 ,3.448946 ,3.461265 ,3.496444 ,3.436451 ,3.449243
,3.455976 ,3.456160 ,3.467699 ,3.435702 ,3.452561 ,3.451059 ,3.444485 ,3.457771 ,3.454376 ,3.419062
,3.462389 ,3.457338 ,4.339757 ,3.464035 ,3.421282 ,3.425457 ,3.419180 ,3.418103 ,3.406501 ,3.425315
,3.433891 ,3.418878 ,3.424673 ,3.413602 ,3.448582 ,3.430728 ,3.434982 ,3.436607 ,3.408126 ,3.424594
,3.454318 ,3.441019 ,3.473242 ,3.419802 ,3.425870 ,3.436805 ,3.433450 ,3.462311 ,3.416731 ,3.433926
,3.447693 ,3.416362 ,4.308095 ,3.414544 ,3.462619 ,3.457022 ,3.453251 ,3.460796 ,3.443702 ,3.470066
,3.468784 ,3.453296 ,3.456075 ,3.430256 ,3.452664 ,3.456064 ,3.446882 ,3.457829 ,3.447612 ,3.440870
,3.496545 ,3.460595 ,3.479051 ,3.455897 ,4.315306 ,3.456421 ,3.439235 ,3.450692 ,3.434896 ,4.302499
,3.452465 ,3.436338 ,3.456108 ,3.424249 ,3.419793 ,3.431556 ,3.413608 ,3.441762 ,4.308771 ,3.442636
,3.439764 ,3.429119 ,3.445399 ,3.447705 ,3.437730 ,3.457834 ,3.439863 ,3.458880 ,3.443907 ,3.460761
,3.462959 ,3.410926 ,3.421188 ,3.419136 ,3.406632 ,3.449722 ,3.448427 ,3.470215 ,4.308677 ,3.446183
,3.444364 ,3.448665 ,3.460267 ,3.455973 ,3.437970 ,3.457457 ,3.435665 ,3.449050 ,4.289853 ,4.305032
,3.458435 ,4.315351 ,3.466027 ,3.455131 ,3.465108 ,3.455638 ,3.450291 ,3.460869 ,3.437570 ,3.468162
,3.473621 ,3.453187 ,3.457542 ,3.457991 ,3.451014 ,3.449923 ,3.434675 ,3.464474 ,3.448937 ,3.444041
,3.436538 ,3.410590 ,3.419598 ,3.399405 ,3.422276 ,3.423569 ,3.431738 ,3.457268 ,3.438264 ,3.434863
,3.421141 ,3.426608 ,3.457683 ,3.432808 ,3.446699 ,3.452761 ,3.438166 ,3.456263 ,3.436119 ,3.441325
,3.446404 ,4.301920 ,3.455531 ,3.442332 ,3.472132 ,3.453711 ,3.442496 ,3.456783 ,3.455944 ,3.458833
,3.465610 ,3.441512 ,3.453216 ,4.310672 ,3.433819 ,3.458321 ,3.433582 ,3.473321 ,3.456576 ,3.443807
,3.464466 ,3.437308 ,3.469557 ,3.456421 ,3.470535 ,3.457466 ,3.443992 ,3.459708 ,3.459016 ,3.467558
,3.477060 ,3.444190 ,3.449053 ,3.452385 ,3.439453 ,3.448660 ,3.455222 ,3.457257 ,3.450218 ,3.444261
,3.440668 ,3.406554 ,3.434649 ,3.448277 ,3.457374 ,3.457356 ,3.444002 ,3.479823 ,3.440215 ,3.445553
,3.469613 ,3.438247 ,3.452851 ,3.442648 ,3.427526 ,3.450818 ,4.314848 ,3.454187 ,3.429235 ,3.441798
,3.460282 ,3.431131 ,3.456584 ,3.441834 ,3.458611 ,3.457034 ,3.440404 ,3.453404 ,3.453020 ,3.477291
,3.473730 ,3.439177 ,3.469648 ,3.454352 ,3.467214 ,3.466590 ,3.447399 ,3.449680 ,3.440150 ,3.449797
,3.460251 ,3.451692 ,3.472587 ,3.433605 ,3.451018 ,3.456360 ,3.440054 ,3.460081 ,3.432452 ,3.452536
,3.461054 ,3.458196 ,3.483561 ,3.446050 ,3.451104 ,3.462915 ,3.460283 ,3.445920 ,3.446153 ,3.381598
,3.437335 ,3.420844 ,3.447263 ,3.434259 ,4.277496 ,3.445492 ,3.415376 ,3.423735 ,3.411069 ,3.426027
,3.432811 ,3.440958 ,3.432175 ,3.406913 ,3.430689 ,3.444283 ,3.407777 ,3.432688 ,3.414015 ,3.448806
,3.438003 ,3.430988 ,3.438785 ,3.398457 ,3.442322 ,3.453573 ,3.436091 ,3.425726 ,3.415592 ,3.437357
,3.429387 ,3.420784 ,3.428431 ,3.408620 ,3.438395 ,3.451616 ,3.433071 ,3.436297 ,3.416843 ,3.422175
,3.428632 ,3.424109 ,3.434704 ,3.460059 ,3.460396 ,4.333751 ,3.447537 ,3.459034 ,3.451555 ,3.441866

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

,3.447678 ,4.303699 ,3.475603 ,4.318962 ,3.422179 ,3.459285 ,3.463415 ,3.470456 ,3.484833 ,3.398483
,3.472251 ,3.424344 ,3.427084 ,3.445340 ,3.441282 ,3.427945 ,3.417615 ,3.452134 ,3.412243 ,3.427368
,4.283678 ,3.410058 ,3.442435 ,3.427159 ,3.444676 ,3.437780 ,3.424732 ,3.441943 ,3.420282 ,3.441544
,3.432663 ,3.419494 ,3.439840 ,3.427887 ,3.455527 ,3.449980 ,3.444340 ,3.446501 ,3.434055 ,3.466412
,3.483462 ,3.463651 ,3.466076 ,3.435053 ,3.439207 ,3.451790 ,3.443273 ,3.443244 ,3.423598 ,3.445032
,3.447161 ,3.450035 ,3.468390 ,3.436626 ,3.452977 ,3.463630 ,4.324897 ,3.441405 ,3.434021 ,3.431017
,3.412572 ,3.413766 ,3.455813 ,3.450512 ,3.447309 ,3.445290 ,3.446766 ,3.467657 ,3.422126 ,3.461142
,



# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

f. Extra Credit (3.3%): Run the Linpack benchmark (http://en.wikipedia.org/wiki/LINPACK) and report the best performance achieved; what efficiency do you achieve compared to the theoretical performance?

```
Current date/time: Thu Feb 11 03:17:31 2016

CPU frequency:    2.881 GHz
Number of CPUs: 1
Number of cores: 1
Number of threads: 1

Parameters are set to:

Number of tests: 15
Number of equations to solve (problem size) : 1000  2000  5000  10000 15000 18000 20000 22000 25000 26000 27000 30000 35000 40000 45000
Leading dimension of array                  : 1000  2000  5008  10000 15000 18008 20016 22008 25000 26000 27000 30000 35000 40000 45000
Number of trials to run                     : 4     2     2     2     2     2     2     2     2     2     1     1     1     1     1
Data alignment value (in Kbytes)            : 4     4     4     4     4     4     4     4     4     4     4     1     1     1     1

Maximum memory requested that can be used=800204096, at the size=10000

================== Timing linear equation system solver ==================

Size   LDA    Align. Time(s)    GFlops   Residual     Residual(norm) Check
1000   1000   4      0.037      17.9337  9.900691e-13 3.376390e-02   pass
1000   1000   4      0.035      19.1465  9.900691e-13 3.376390e-02   pass
1000   1000   4      0.037      18.0474  9.900691e-13 3.376390e-02   pass
1000   1000   4      0.036      18.6005  9.900691e-13 3.376390e-02   pass
2000   2000   4      0.271      19.7209  4.053480e-12 3.526031e-02   pass
2000   2000   4      0.264      20.2172  4.053480e-12 3.526031e-02   pass
5000   5008   4      3.928      21.2261  2.336047e-11 3.257429e-02   pass
5000   5008   4      3.940      21.1648  2.336047e-11 3.257429e-02   pass
10000  10000  4      31.312     21.2975  1.124127e-10 3.963786e-02   pass
10000  10000  4      30.815     21.6409  1.124127e-10 3.963786e-02   pass

Performance Summary (GFlops)

Size   LDA    Align.  Average  Maximal
1000   1000   4       18.4320  19.1465
2000   2000   4       19.9691  20.2172
5000   5008   4       21.1954  21.2261
10000  10000  4       21.4692  21.6409

Residual checks PASSED

End of tests
```

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

Theoretical performance is 20 GFLOP, LIMPACK is 21.4692. Since Intel E5-2670 series have Max turbo frequency, it may increase more than 20 GFLOPS, So LIMPACK achieved almost 100% of theoretical performance.

2. Memory

a. Measure the memory speed of your host.

b. Your parameter space should include read+write operations (e.g. memcpy),sequential access, random access, varying block sizes (1B, 1KB, 1MB), and varying the concurrency (1 thread & 2 threads)

c. The metrics you should be measuring are throughput (Megabytes per second, MB/sec) and latency (milliseconds, ms)

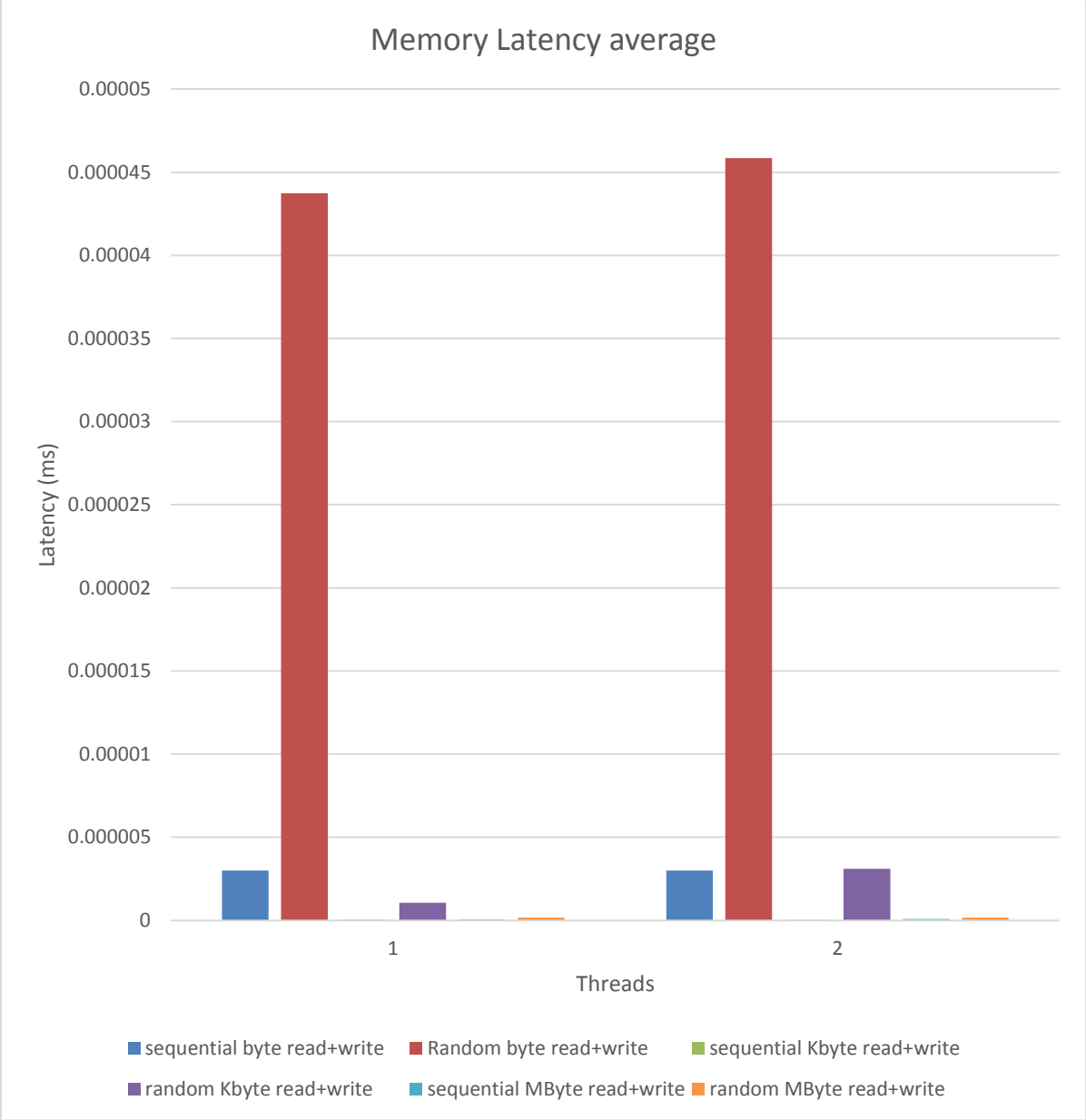
Reading are as below.

| **Sequential Byte transfer latency (ms)** | | | | | |
|---|---|---|---|---|---|
| Threads | Observation#1 | observation#2 | observation#3 | average | standard deviation |
| 1 | 0.000003 | 0.000003 | 0.000003 | 0.000003 | 0 |
| 2 | 0.000003 | 0.000003 | 0.000003 | 0.000003 | 0 |
| **Sequential byte transfer throughput (MB/sec)** | | | | | |
| 1 | 370.3 | 375.86 | 373.7373 | 373.2991 | 2.430538139 |
| 2 | 375.29 | 375.612 | 369.457569 | 373.453 | 4.124151511 |
| **Random Byte transfer latency (ms)** | | | | | |
| 1 | 0.0000417 | 0.00004624 | 0.00004324 | 4.37267E-05 | 2.30879E-06 |
| 2 | 0.0000408 | 0.00004593 | 0.00005079 | 0.00004584 | 4.99561E-06 |
| **Random byte transfer throughput (MB/sec)** | | | | | |
| 1 | 23.98 | 21.63 | 23.128 | 22.91266667 | 1.189706406 |
| 2 | 24.54 | 21.7723 | 19.849346 | 22.053882 | 2.357970537 |
| **Sequential Kbyte transfer latency (ms)** | | | | | |
| 1 | 4.35E-08 | 2.68E-08 | 4.56E-08 | 3.86333E-08 | 1.03016E-08 |
| 2 | 2.82E-08 | 2.76E-08 | 2.79E-08 | 2.79E-08 | 3E-10 |
| **Sequential Kbyte transfer throughput (MB/sec)** | | | | | |
| 1 | 22995.09 | 37382.39 | 21913.85 | 27430.44333 | 8635.577678 |
| 2 | 35484.81 | 36282.907 | 35848.752 | 35872.15633 | 399.5629205 |
| **Random Kbyte transfer latency (ms)** | | | | | |
| 1 | 2.35E-07 | 0.000002675 | 2.799E-07 | 1.0633E-06 | 1.39595E-06 |
| 2 | 2.901E-07 | 3.326E-07 | 3.098E-07 | 3.10833E-07 | 2.12688E-08 |
| **Random Kbyte transfer throughput (MB/sec)** | | | | | |
| 1 | 4255.5844 | 3738.91 | 3572.6609 | 3855.718433 | 356.1309932 |
| 2 | 3446.9954 | 3006.23853 | 3227.38033 | 3226.871419 | 220.3788747 |

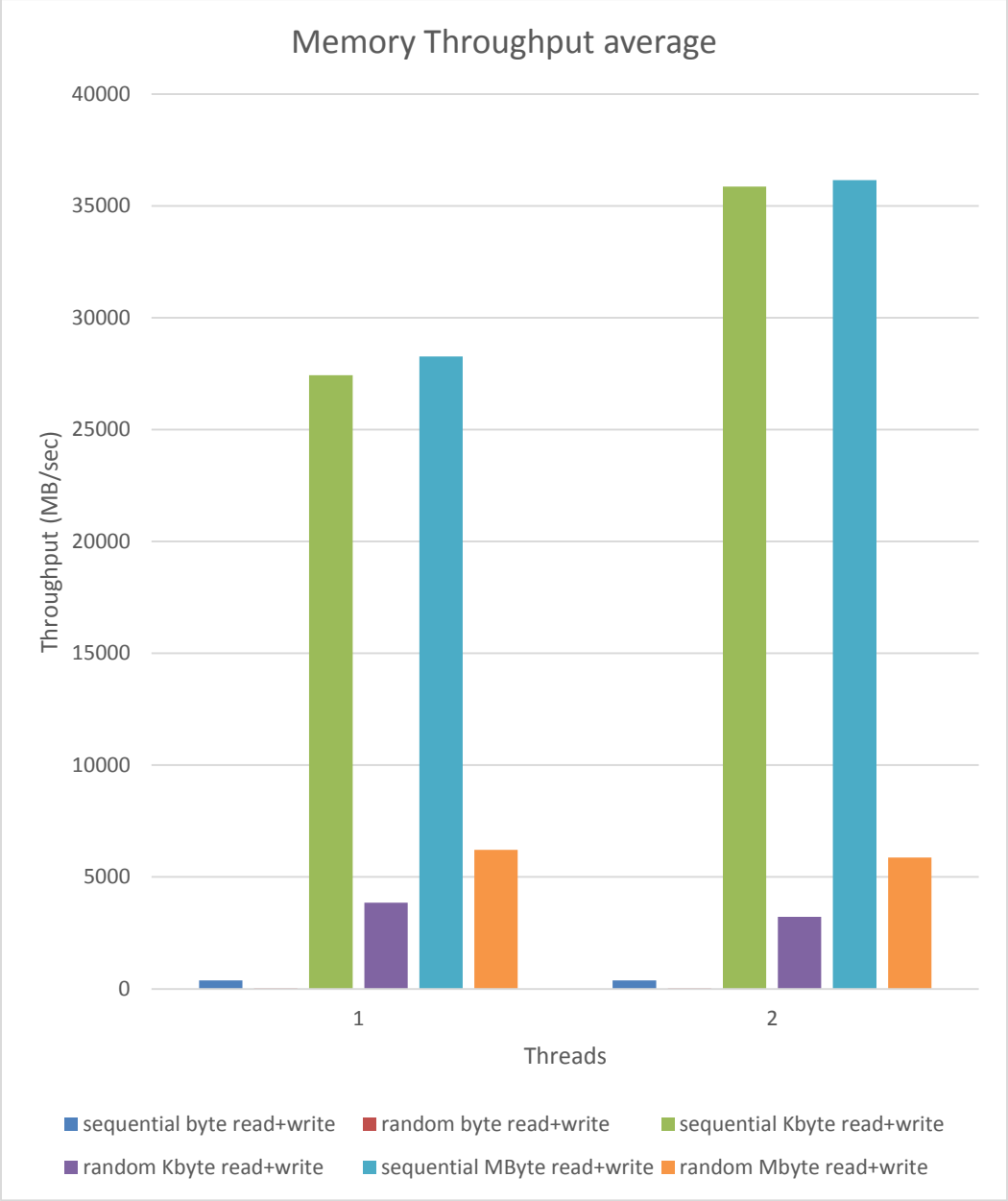
# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

| | | | | | |
|---|---|---|---|---|---|
| **Sequential Mbyte transfer latency (ms)** | | | | | |
| 1 | 5.97E-08 | 0.000000056 | 5.81E-08 | 5.79333E-08 | 1.85562E-09 |
| 2 | 7.58E-08 | 7.93E-08 | 0.000000073 | 7.60333E-08 | 3.15647E-09 |
| **Sequential Mbyte transfer throughput(MB/sec)** | | | | | |
| 1 | 26763.81 | 27863.31 | 27203.872 | 27276.99733 | 553.3855331 |
| 2 | 36189.64 | 35603.0769 | 36697.9229 | 36163.5466 | 547.8892133 |
| **Random Mbyte transfer latency (ms)** | | | | | |
| 1 | 1.524E-07 | 1.672E-07 | 1.633E-07 | 1.60967E-07 | 7.67094E-09 |
| 2 | 1.729E-07 | 1.793E-07 | 1.596E-07 | 1.706E-07 | 1.00494E-08 |
| **Random Mbyte transfer throughput (MB/sec)** | | | | | |
| 1 | 6559.75 | 5979.91 | 6124.8598 | 6221.5066 | 301.7599366 |
| 2 | 5782.06 | 5576.0489 | 6263.894863 | 5874.001254 | 353.0197061 |

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation



# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation




d. Compute the theoretical memory bandwidth of your memory, based on the type of memory and the speed.

Formula to calculate Memory bandwidth is the product of:=

**Base DRAM clock frequency * Number of data transfers per clock * Memory bus (interface) width: * Number of interfaces**:

Since there is no mention DRAM type. With reference to https://en.wikipedia.org/wiki/DDR3_SDRAM for CPU E5-2670 v2. I am considering DDR3-1333* memory type with 10666.67 MB/sec.

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

e. Extra Credit (3.3%): Run the Stream benchmark (http://www.cs.virginia.edu/stream/) and report the best performance achieved; what efficiency do you achieve compared to the theoretical performance?

```
-------------------------------------------------------------
STREAM version $Revision: 5.10 $
-------------------------------------------------------------
This system uses 8 bytes per array element.
-------------------------------------------------------------
Array size = 20000000 (elements), Offset = 0 (elements)
Memory per array = 152.6 MiB (= 0.1 GiB).
Total memory required = 457.8 MiB (= 0.4 GiB).
Each kernel will be executed 10 times.
 The *best* time for each kernel (excluding the first iteration)
 will be used to compute the reported bandwidth.
-------------------------------------------------------------
Your clock granularity/precision appears to be 1 microseconds.
Each test below will take on the order of 50971 microseconds.
   (= 50971 clock ticks)
Increase the size of the arrays if this shows that
you are not getting at least 20 clock ticks per test.
-------------------------------------------------------------
WARNING -- The above is only a rough guideline.
For best results, please be sure you know the
precision of your system timer.
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           6173.1     0.052940     0.051838     0.054501
Scale:          6134.5     0.053769     0.052164     0.055194
Add:            8851.5     0.055673     0.054228     0.056868
Triad:          8107.0     0.060925     0.059208     0.061939
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three arrays
```

Assumed theoretical peak memory speed is 10666.67 MB/sec. with stream max speed is 8851.5 MB/sec so 8851.5/10666.67= 82.98% of theoretical peak performance.

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

3. Disk

a. Measure the disk speed; Hint: there are multiple ways to read and write to disk, explore the different APIs, and pick the fastest one out of all them

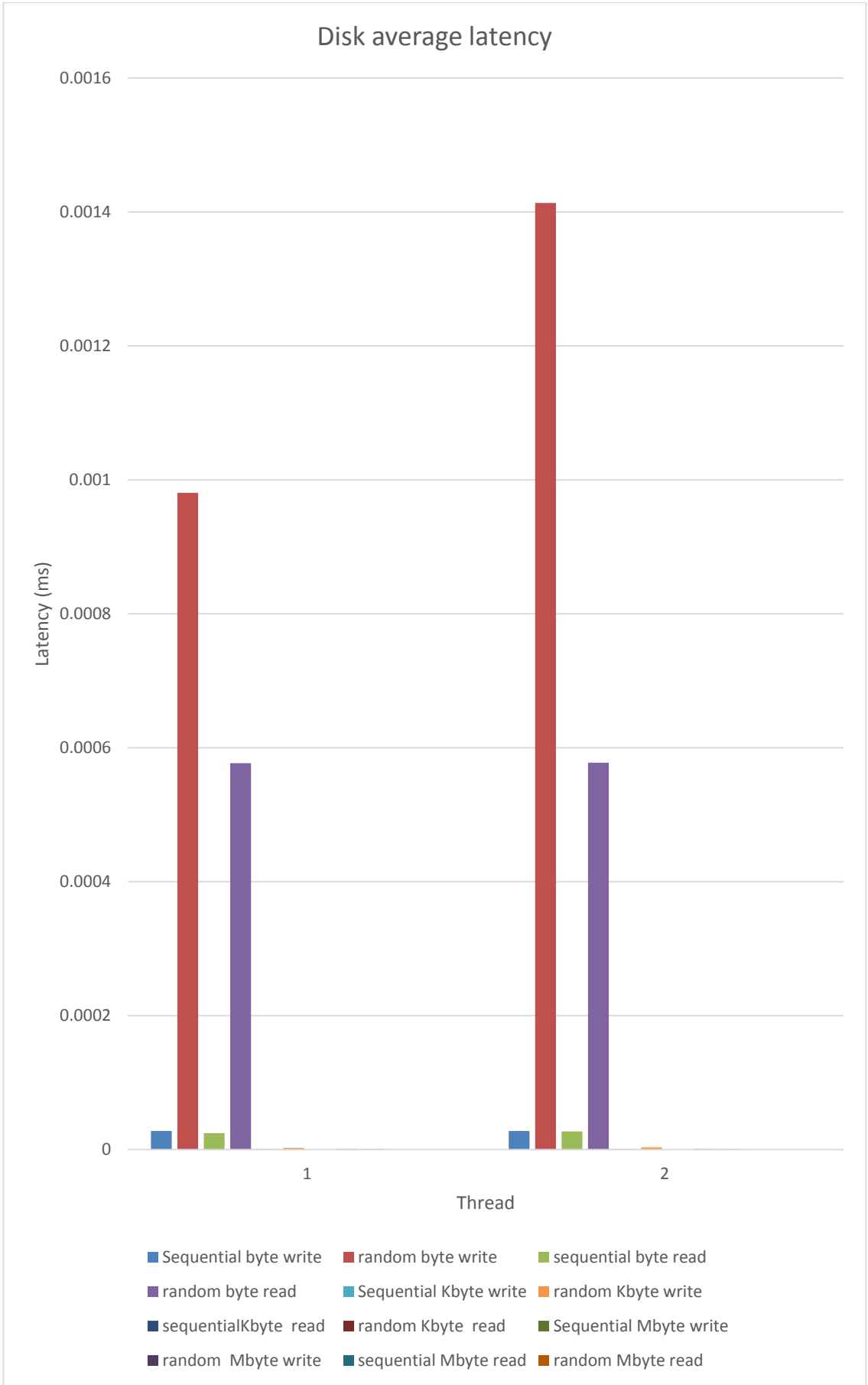
b. Your parameter space should include read operations, write operations, sequential access, random access, varying block sizes (1B, 1KB, 1MB), and varying the concurrency (1 thread, 2 threads) c. The metrics you should be measuring are throughput (MB/sec) and latency (ms)

| Sequential write Byte transfer latency | | | | | |
|---|---|---|---|---|---|
| Threads | Observation#1 | observation#2 | observation#3 | average | standard deviation |
| 1 | 0.000028 | 0.000028 | 0.000028 | 0.000028 | 0 |
| 2 | 0.000028 | 0.000028 | 0.000028 | 0.000028 | 0 |
| Sequential write byte transfer throughput | | | | | |
| 1 | 36.02 | 36.8034 | 36.096 | 36.30646667 | 0.432031311 |
| 2 | 35.25 | 35.7 | 35.82 | 35.59 | 0.300499584 |
| Random write Byte transfer latency | | | | | |
| 1 | 0.00139799 | 0.00014364 | 0.001400252 | 0.000980627 | 0.000724853 |
| 2 | 0.0014189 | 0.001404 | 0.00141785 | 0.001413583 | 8.316E-06 |
| Random write byte transfer throughput | | | | | |
| 1 | 0.7153 | 0.69618 | 0.714171 | 0.708550333 | 0.010727885 |
| 2 | 0.764766 | 0.712186 | 0.72529 | 0.734080667 | 0.027370074 |
| Sequential Kbyte write transfer latency | | | | | |
| 1 | 9.413E-07 | 8.181E-07 | 8.562E-07 | 8.71867E-07 | 6.30765E-08 |
| 2 | 1.1781E-07 | 9.608E-07 | 9.325E-07 | 6.7037E-07 | 4.7874E-07 |
| Sequential Kbyte write throughput | | | | | |
| 1 | 1062.38 | 1322.4014 | 1167.939407 | 1184.240269 | 130.7748856 |
| 2 | 1048.843 | 1040.77 | 1072.3829 | 1053.998633 | 16.42495899 |
| Random Kbyte write latency | | | | | |
| 1 | 2.6156E-06 | 2.3976E-06 | 2.4363E-06 | 2.48317E-06 | 1.16311E-07 |
| 2 | 3.8269E-06 | 3.2183E-06 | 3.2993E-06 | 3.44817E-06 | 3.30484E-07 |
| Random Kbyte write throughput | | | | | |
| 1 | 382.315 | 417.078 | 410.4658 | 403.2862667 | 18.4601142 |
| 2 | 261.30785 | 310.726 | 303.0916 | 291.7084833 | 26.6030054 |
| Sequential Mbyte write latency | | | | | |
| 1 | 7.199E-07 | 5.749E-07 | 7.174E-07 | 6.70733E-07 | 8.30035E-08 |
| 2 | 6.559E-07 | 6.525E-07 | 6.789E-07 | 6.62433E-07 | 1.43615E-08 |
| Sequential Mbyte write throughput | | | | | |
| 1 | 1389.0236 | 1739.5089 | 1393.8269 | 1507.453133 | 200.980539 |
| 2 | 1524.5362 | 1513.255 | 1479.576 | 1505.789067 | 23.39145178 |
| Random Mbyte write latency | | | | | |

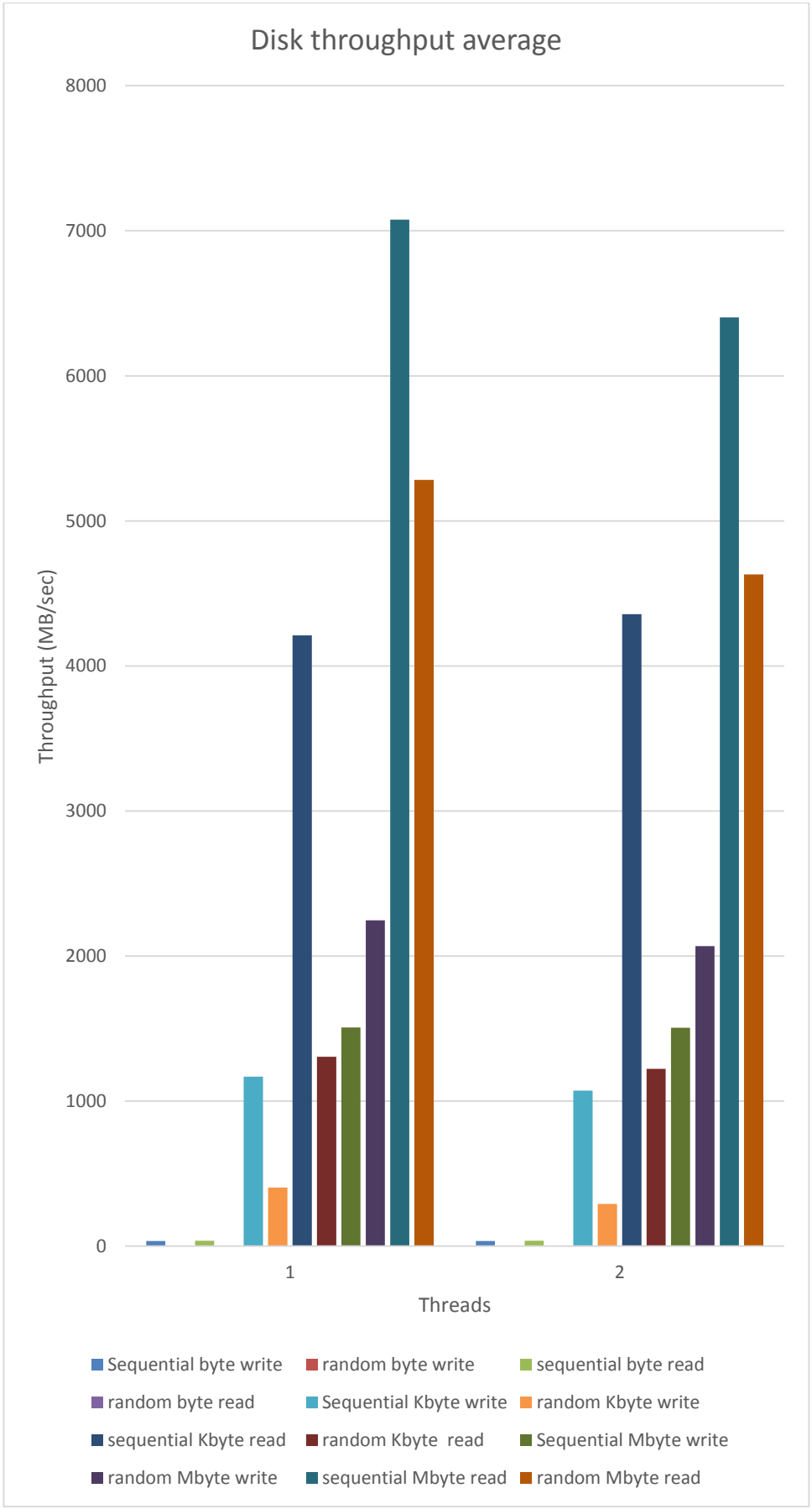
# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

| | | | | | |
|---|---|---|---|---|---|
| 1 | 4.743E-07 | 3.746E-07 | 5.106E-07 | 4.53167E-07 | 7.04199E-08 |
| 2 | 4.753E-07 | 4.737E-07 | 5.022E-07 | 4.83733E-07 | 1.60126E-08 |
| Random Mbyte write throughput | | | | | |
| 1 | 2108.5381 | 2669.4908 | 1958.4908 | 2245.506567 | 374.7672887 |
| 2 | 2103.8844 | 2111.08516 | 1991.219 | 2068.72952 | 67.22256518 |
| Sequential read Byte transfer latency | | | | | |
| 1 | 0.000027 | 0.000027 | 0.00002 | 2.46667E-05 | 4.04145E-06 |
| 2 | 0.000027 | 0.000027 | 0.000027 | 0.000027 | 4.1496E-21 |
| Sequential read byte transfer throughput | | | | | |
| 1 | 36.71 | 36.95 | 37.105 | 36.92166667 | 0.199018425 |
| 2 | 36.5674 | 36.85 | 36.8916 | 36.76966667 | 0.176398677 |
| Random read Byte transfer latency | | | | | |
| 1 | 0.000588186 | 0.000583444 | 0.000559448 | 0.000577026 | 1.54067E-05 |
| 2 | 0.000576244 | 0.00058197 | 0.0005747 | 0.000577638 | 3.82991E-06 |
| Random read byte transfer throughput | | | | | |
| 1 | 1.6983 | 1.71396 | 1.7874 | 1.73322 | 0.047570087 |
| 2 | 1.7353 | 1.718303 | 1.740012 | 1.731205 | 0.011419148 |
| Sequential Kbyte read transfer latency | | | | | |
| 1 | 2.486E-07 | 2.378E-07 | 2.271E-07 | 2.37833E-07 | 1.075E-08 |
| 2 | 2.198E-07 | 0.000000242 | 2.278E-07 | 2.29867E-07 | 1.12434E-08 |
| Sequential Kbyte read throughput | | | | | |
| 1 | 4022.1557 | 4204.394 | 4403.9311 | 4210.160267 | 190.9530083 |
| 2 | 4549.664 | 4131.605 | 4389.183 | 4356.817333 | 210.9004129 |
| Random Kbyte read latency | | | | | |
| 1 | 7.913E-07 | 7.398E-07 | 7.667E-07 | 7.65933E-07 | 2.57586E-08 |
| 2 | 8.121E-07 | 8.461E-07 | 0.000000798 | 8.18733E-07 | 2.47266E-08 |
| Random Kbyte read throughput | | | | | |
| 1 | 1263.801 | 1351.78 | 1304.361 | 1306.647333 | 44.03403911 |
| 2 | 1231.301 | 1181.8935 | 1253.07839 | 1222.090963 | 36.47520696 |
| Sequential Mbyte read latency | | | | | |
| 1 | 1.513E-07 | 1.241E-07 | 1.524E-07 | 1.426E-07 | 1.60309E-08 |
| 2 | 6.559E-07 | 0.000000144 | 1.801E-07 | 3.26667E-07 | 2.85695E-07 |
| Sequential Mbyte read throughput | | | | | |
| 1 | 6611.4501 | 8059.769 | 6561.8022 | 7077.673767 | 850.8816099 |
| 2 | 6713.034571 | 6944.211921 | 5553.898 | 6403.714831 | 744.9847153 |
| Random Mbyte read latency | | | | | |
| 1 | 1.798E-07 | 1.975E-07 | 1.914E-07 | 1.89567E-07 | 8.99129E-09 |
| 2 | 2.143E-07 | 2.198E-07 | 2.139E-07 | 0.000000216 | 3.29697E-09 |
| Random Mbyte read throughput | | | | | |
| 1 | 5562.7374 | 5063.1385 | 5224.5939 | 5283.489933 | 254.9535689 |
| 2 | 4666.559 | 4549.1366 | 4674.8818 | 4630.192467 | 70.31967997 |

# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation



# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation



# CS553 Programming Assignment #1
# Benchmarking
# Performance Evaluation

d. Extra Credit (3.3%): Run the IOZone benchmark (http://www.iozone.org/) and report the best performance achieved; what efficiency do you achieve compared to the theoretical erformance? Hint: The theoretical performance is generally advertised by the manufacturer.

```
Run began: Thu Feb 11 06:31:21 2016

Auto Mode
Command line used: ./iozone -a
Output is in Kbytes/sec
Time Resolution = 0.000001 seconds.
Processor cache size set to 1024 Kbytes.
Processor cache line size set to 32 bytes.
File stride size set to 17 * record size.
                                                     random  random    bkwd   record   stride
      KB  reclen   write rewrite    read    reread    read   write    read  rewrite     read   fwrite frewrite   fread  freread
      64       4  666414 2278628  5860307 10821524 7940539 2467108 3791156  3958892  6421025  2133730  2298136 3363612 10821524
      64       8 1357066 1638743  7100397 1597288510821524 2561267 4564786  4564786  7940539  2298136  1638743 4018152  7940539
      64      16 1330167 1638743  9318832 1597288512902017 2892445 3791156  2278628  6421025  2467108  1933893 5283570 15972885
      64      32 1518251 2561267  7940539 1597288512310336 2662899 3791156  2662899  4018152  1599680  2278628 5283570 15972885
      64      64 1279447 2052169 15972885 15972885 3738358 2561267 3958892  2772930  5283570  2923952  2662899 5389653 12902017
     128       4 1294270 2211113  5074121 11720614 9129573 2608629 4557257  4135958  7082197  2286447  2326073 4407601  9977956
     128       8 1440082 2511022  5545860 1636517311470204 2066432 4233807  5784891  7917784  2843510  2608629 3759450 12842051
     128      16 1243315 2166499  6114306 1284205111720614 1967960 5379161  4135958  7082197  2132084  2608629 5122535 11720614
     128      32 1183041 2621367  6406138 1172061411470204 2248149 2784517  3895854  5784891  2464907  2420455 4557257 12842051
     128      64 1598754 2770150  4717434 1801235912842051 2453642 5847904  3359523  4267461  2367096  2905056 4934216 14200794
     128     128 1543594 2326073 16365173 15881078 9977956 3380677 4444086  3380677  4934216  1579934  2985839 4889281 14200794
     256       4 1235882 2509882  4496299  9114515 8024634 2118645 3823789  4350555  5938650  2557711  1855098 5117791  8271916
     256       8 1405778 2726577  6719046 1416439511091721 2325094 4054829  5347168  8024634  2607399  2639446 5320671  7314033
     256      16 1375171 2582316  3454704 1416439512228612 2726577 4197489  5569035 10651598  2242541  2692393 4572895 13454450
     256      32 1497953 2672291  5455847 1281227712812277 2486631 5569035  5347168  8024634  3078337  3009318 4422226 11207494
     256      64 1489641 2812272  7314033 1416439510758322 3326279 4404088  3935921  9192546  3159869  3705040 4819184 14353744
     256     128 1629830 2607399  7791708 1345445010758322 2975955 5717301  4350555  6437081  2943325  5022044 4197489 13454450
     256     256 1552085 2935279 12812277 1281227710245071 2911402 4496299  2943325  5938650  2880164  3159869 5938650 12228612
     512       4 1210664 2350044  6018636 12215097 8665997 2337255 5019764  5067142  7540170  2169601  2263354 5566231 10434519
     512       8 1492545 2180616  6736026 14240069 8701110 2625909 4650188  5760329 10235583  2613128  2584820 4384358 11877300
     512      16 1500891 2958758  6843354 13438092 8665997 2815243 3905895  5278892  9468384  2678309  2926501 3849877 12499490
     512      32 1547395 2979282  5227492 12499490 8701110 2811557 4916336  5398323 11374040  2725905  2768068 6571132 13109944
     512      64 1467055 2922519  6821616 1462806712146009 2879414 4411377  6104175  8528336  2958758  2681653 6472112 10694336
     512     128 1679288 3099691  6414119 14240069 9468384 2651850 4916336  4701087  7988981  3634839  2910635 5019764 12797441
     512     256 1684557 2438090  7540170 1310994411374040 3104171 4973263  3415178  7022379  2750343  4271001 5624545 13872122
     512     512 1695195 2497638  9814569 12499490 9638369 2548017 5384786  3710197  5278892  3659616  2782414 6319739 12499490
    1024       4 1369216 1957770  6128613 11614118 8970167 2659742 4428688  5303701  8680108  2443368  2082099 5593820  8895850
    1024       8 1556257 2534194  6489770 11645609 8822754 3003881 6128613  6431462 10557785  2432298  2859868 6441107 14044758
    1024      16 1638171 2409105  6974549 1347205310158253 2461573 6137371  7020149  9569770  2910252  2509027 6317933 13643232
    1024      32 1607514 2875184  6327241 1217374712639479 3425544 4614246  7054742 10039528  2674649  2375790 5390231 12790037
    1024      64 1468964 2646630  6327241 1529515713102174 2774874 6393168  5758828 10662628  2789291  2798378 6819511 14422045
    1024     128 1726407 3179559  7319232 1399898113472053 2859868 5885083  6489770 11520658  2659742  3261656 6163794 14044758
    1024     256 1651398 2375790  7644933 1294422412173747 3189002 5983467  4654248  9142007  3894582  3603747 5720477 13818817
    1024     512 1812371 2392998  7369466 1364323212037272 3778101 5720477  4789183  8391792  2852271  4195100 7020149 12492426
    1024    1024 1687075 3083680  9402175 1177330112492426 3725664 6692005  3369115  5444898  3261656  2934110 6489770 10454984
```

I have attached total output of above tool in XLS format.