Procedure

For the given five tasks, an independent program and testing methodology was used to derive data for each task. These methods are listed below.

1. For Task 1, the Intel Memory Latency checker tool was used to find the latency of the main memory. The flag –idle_latency flag was used to determine the latency with a queue length of zero.
2. For task 2, the task2.c code was run twelve times with varying granularity and read/write ratio. Specifically, the program was run with read/write ratios of 100, 70, 50, 30. The granularity was tested at 64B, 256B, and 1025B. The Linux perf function (`perf stat ./test2 "granularity" "read%"`) was used to find the runtime of this program. This program functioned by allocating 1 GB of memory and performing both read and write tasks with it. The runtime of this process was taken and plugged into the formula 2GB / Runtime = Bandwidth (GB/s) in order to find how the output result changed.
3. To determine the trade-off between latency and throughput, task3.c was written to perform assorted read/write operations on an array stored in memory. Throughput is modulated by adjusting the number of threads used to execute this program, scaling from 1 to 8. The same perf command was used to determine the runtime of the overall program. (`perf stat ./test3`)
4. To determine the impact of cache miss ratio on performance, a program (task4.c) was written to perform cache operations as either “friendly” (Sequential array access), or “hostile” (accessing in random order). A modified perf command was run to analyze both the runtime of this program and the number of cache misses (`perf stat -e L1-dcache-loads,L1-dcache-load-misses,LLC-loads,LLC-load-misses ./test4 "friendly/hostile"`)
5. To determine the impact of TLB table miss ratio, the same exact program was used as in task 4, but the amount of data being processed was increased by a factor of 10. The same perf command was used to measure TLB loads and misses.(`perf stat -e L1-dcache-loads,L1-dcache-load-misses,LLC-loads,LLC-load-misses ./test5 "friendly/hostile"`)

Results

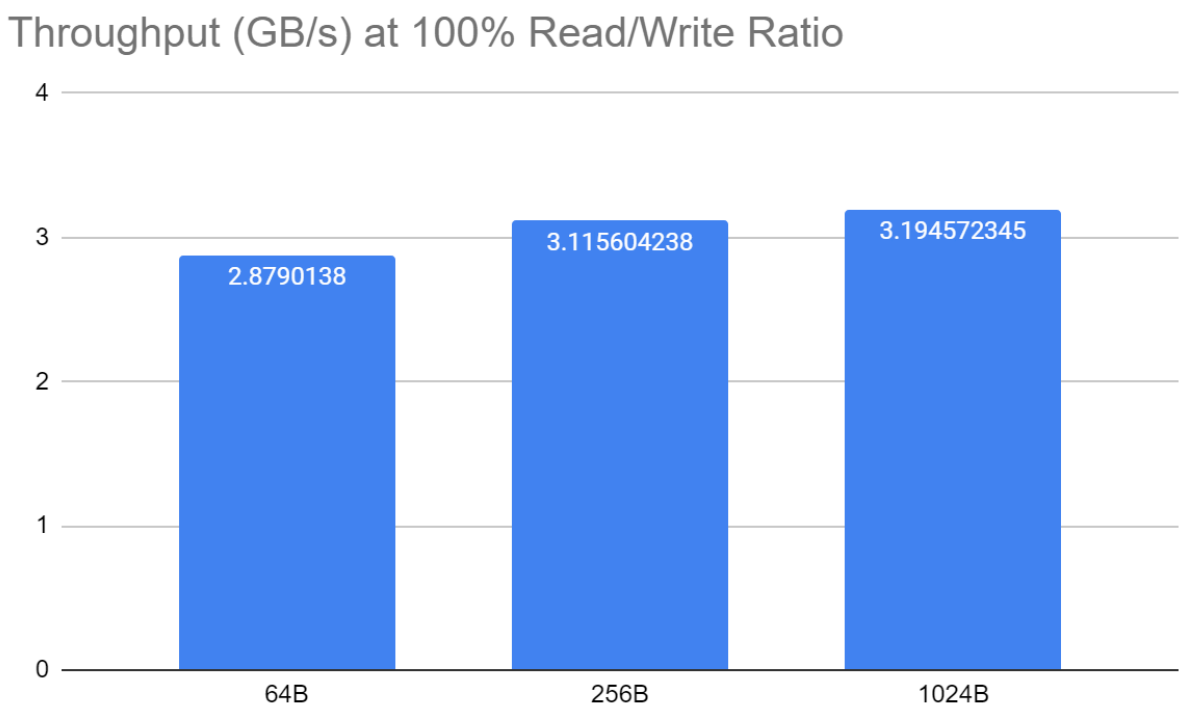
1. Intel Memory Latency Checker found the latency of the main memory to be 107.7ns

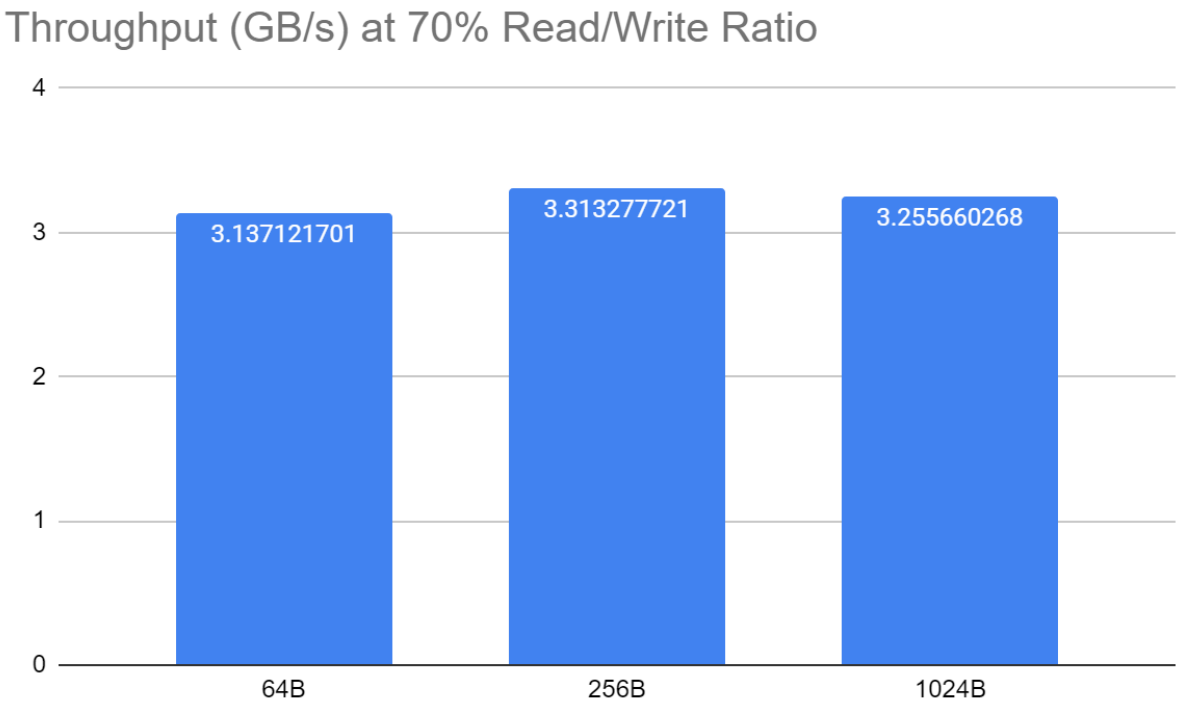
```
muizza@muizza-ThinkPad-T14s-Gen-1:~/Desktop/AdvCompSys/proj1/Linux$ ./mlc --idle_latency
Intel(R) Memory Latency Checker - v3.11b
Command line parameters: --idle_latency

Using buffer size of 1800.000MiB
Each iteration took 225.8 base frequency clocks (       107.7    ns)
```

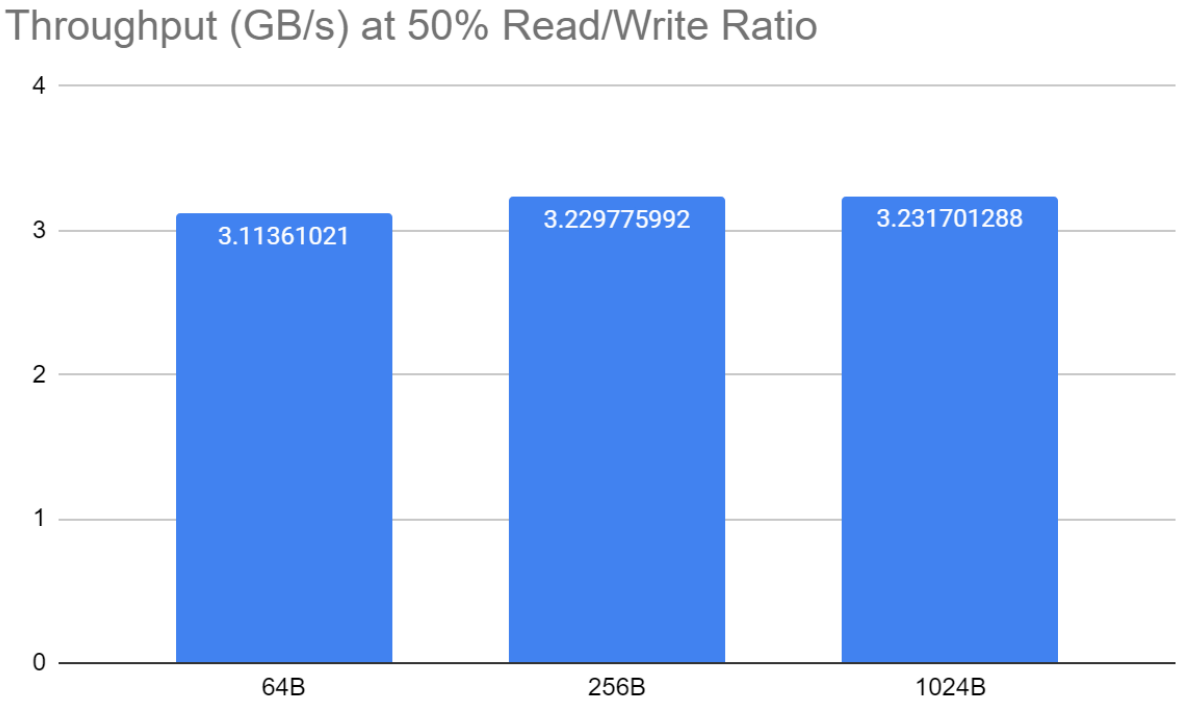
2. The throughput result across the three granularity levels (x-axis) for each of the four read/write ratios is shown below. The general result from this test shows that as the granularity increases, the throughput generally increases (and thus runtime would decrease) but this bonus will plateau. The greatest throughput occurred when the read/write ratio was 70:30, and the lowest throughput by far occurred when the ratio was 0:100 (meaning write-only, all reads would error). The worst throughput was at read-only.

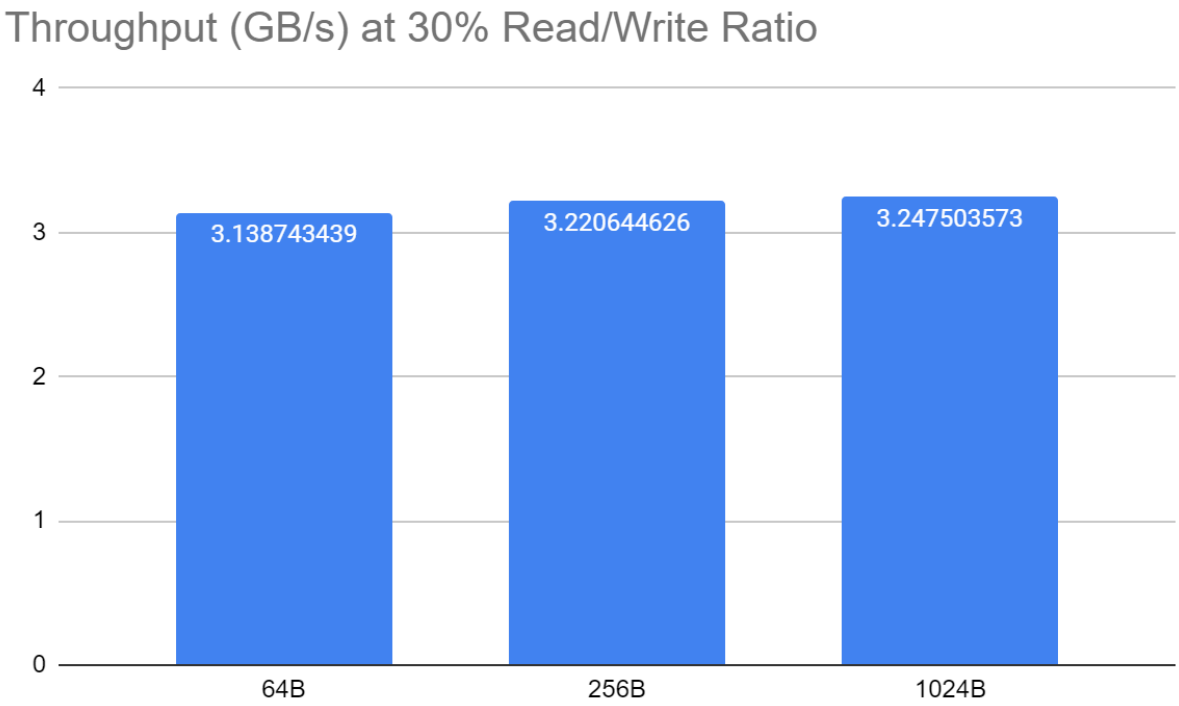
Table 1: Throughput (GB/s) for varying Read/Write Ratio and Granularity

Read/Write	100/0	70/30	50/50	30/70
64B	2.8790138	3.137121701	3.11361021	3.138743439
256B	3.115604238	3.313277721	3.229775992	3.220644626
1024B	3.194572345	3.255660268	3.231701288	3.247503573




Throughput (GB/s) at 70% Read/Write Ratio
4
3
2
1
0
3.137121701
3.313277721
3.255660268
64B
256B
1024B


Throughput (GB/s) at 50% Read/Write Ratio
4
3
2
1
0
3.11361021
3.229775992
3.231701288
64B
256B
1024B



3. Table 2 shows throughput and runtime for the program with 8 different threading levels. It can be seen that as the number of threads increases, the runtime did initially decrease, but then plateaus around 4 threads. The throughput also initially started to decrease but also plateaued around 4 threads.

Number of Threads	Runtime (microseconds)	Throughput (Ops/sec)
1	1.36	735219.14
2	1.53	652019.97
3	1.58	632178.03
4	1.60	624067.43
5	1.58	632897.13
6	1.58	631191.82
7	1.58	634368.71
8	1.58	634542.88

4. The results for both the friendly and hostile cache tests are shown below (L1-dcache-loads only, LLC loads and misses are not supported).
   For the friendly test, the miss ratio was 0.66% and runtime was 0.332468946 sec.
   For the hostile test, the miss ratio was 2.69% and the runtime was 0.519800570 sec.

   It is clear that the friendly cache test shows the optimal behavior. Sequential reads allow the miss ratio to drastically decrease and reduce the required runtime. In reality, the LLC Cache Miss % would be more important than the L1 Cache Miss %. This is because there is a far greater penalty for accessing RAM than there is for L1 cache to access L2.

```
muizza@muizza-ThinkPad-T14s-Gen-1:~/Desktop/AdvCompSys/proj1$ perf stat
-e L1-dcache-loads,L1-dcache-load-misses,LLC-loads,LLC-load-misses
./test4 friendly
The cache-friendly sum: -2014260032
The execution time: 0.033057 seconds

 Performance counter stats for './test4 friendly':

       548,734,485      L1-dcache-loads
         3,648,865      L1-dcache-load-misses             #    0.66% of
all L1-dcache accesses
   <not supported>      LLC-loads
   <not supported>      LLC-load-misses

       0.332468946 seconds time elapsed

       0.251128000 seconds user
       0.081041000 seconds sys


muizza@muizza-ThinkPad-T14s-Gen-1:~/Desktop/AdvCompSys/proj1$ perf stat
-e L1-dcache-loads,L1-dcache-load-misses,LLC-loads,LLC-load-misses
./test4 hostile
The cache-hostile sum: 1350474461
The execution time: 0.222974 seconds

 Performance counter stats for './test4 hostile':

       577,469,487      L1-dcache-loads
        13,802,927      L1-dcache-load-misses             #    2.39% of
all L1-dcache accesses
   <not supported>      LLC-loads
   <not supported>      LLC-load-misses

       0.519800570 seconds time elapsed

       0.444965000 seconds user
       0.074994000 seconds sys
```

5. To determine the impact of TLB miss ratio, the same program as task 4 was run with 10 times the amount of data so that the TLB would overflow.

   For the friendly test, the miss ratio was 0.65% and runtime was 3.157238277 sec.
   For the hostile test, the miss ratio was 2.35% and the runtime was 4.586369378 sec.

   While the miss ratio did not change much, it is clear that having frequent TLB misses will greatly degrade speed and runtime.

```
muizza@muizza-ThinkPad-T14s-Gen-1:~/Desktop/AdvCompSys/proj1$ perf stat
-e L1-dcache-loads,L1-dcache-load-misses,LLC-loads,LLC-load-misses
./test5 friendly
The cache-friendly sum: 887459712
The execution time: 0.330489 seconds

 Performance counter stats for './test5 friendly':

     5,496,208,138      L1-dcache-loads
        35,601,668      L1-dcache-load-misses             #    0.65% of
all L1-dcache accesses
   <not supported>      LLC-loads
   <not supported>      LLC-load-misses

       3.157238277 seconds time elapsed

       2.494487000 seconds user
       0.653127000 seconds sys


muizza@muizza-ThinkPad-T14s-Gen-1:~/Desktop/AdvCompSys/proj1$ perf stat
-e L1-dcache-loads,L1-dcache-load-misses,LLC-loads,LLC-load-misses
./test5 hostile
The cache-hostile sum: 1917244113
The execution time: 1.770739 seconds

 Performance counter stats for './test5 hostile':

     5,810,246,890      L1-dcache-loads
       136,431,603      L1-dcache-load-misses             #    2.35% of
all L1-dcache accesses
   <not supported>      LLC-loads
   <not supported>      LLC-load-misses

       4.586369378 seconds time elapsed

       3.914789000 seconds user
       0.670963000 seconds sys
```