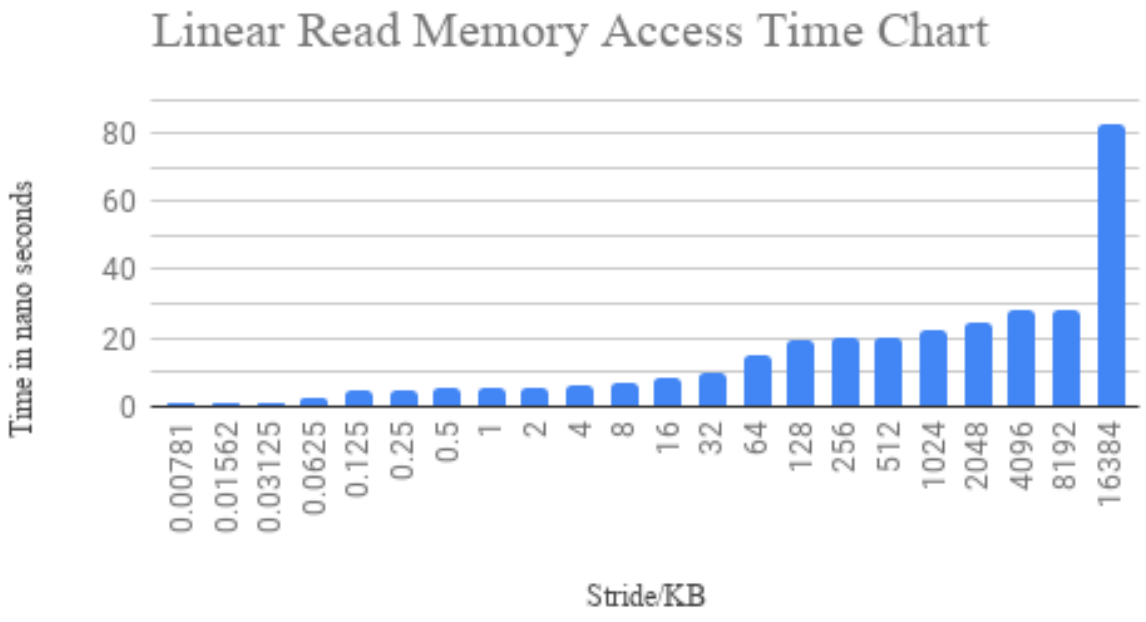
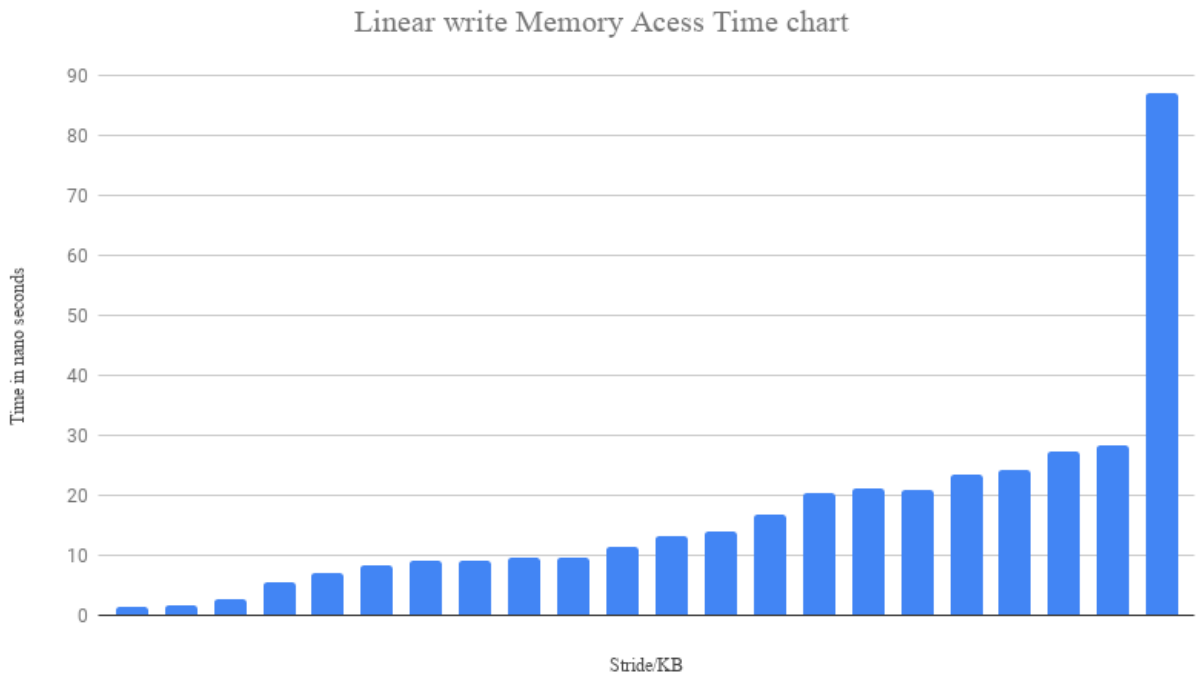
1.Introduction

The performance gap between the CPU and main memory system increased which compelled system designers to insert a small SRAM cache memory, called an L1 cache (Level 1 cache) been the CPU register file and main memory, The L1 cache can be accessed nearly as fast as the registers, typically in 2 to 4 clock cycles.continued to increase, L2 cache, between the L1 cache and main memory, that can be accessed in about 10 clock cycles. Some modern systems include an additional even larger cache, called an L3 cache, which sits between the L2 cache and main memory in the memory hierarchy and can be accessed in 30 or 40 cycles.

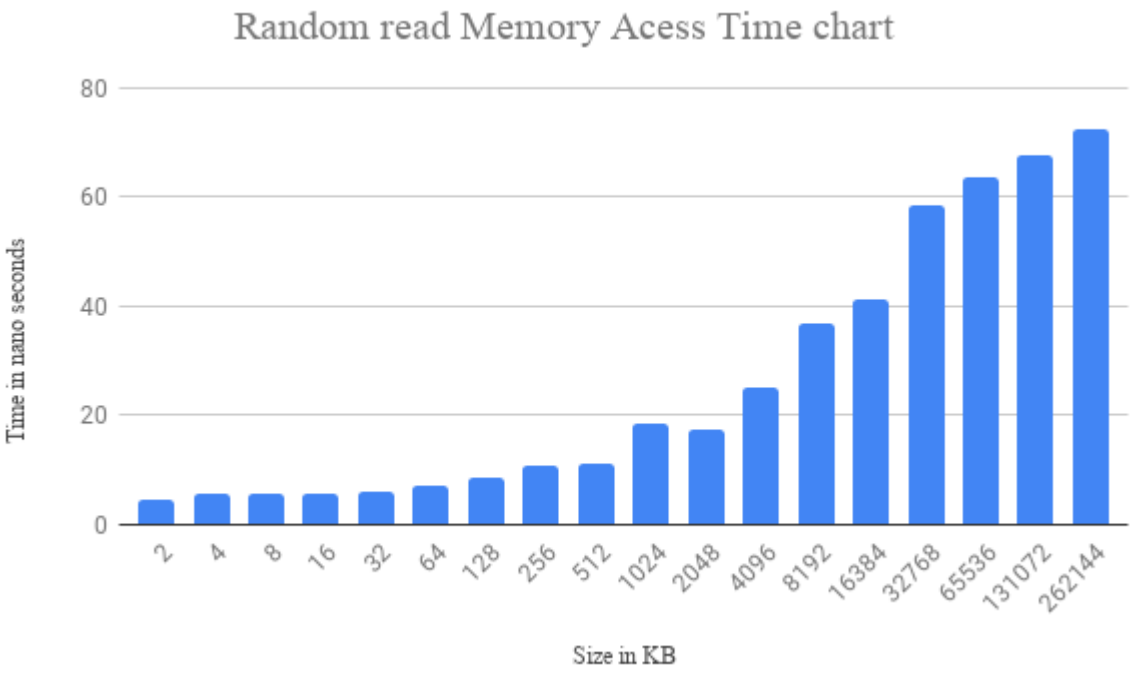
2: Part1:
Measuring the memory hierarchy performance in my PC(i7-8th Gen):
Basically, I have started testing the L1, L2, L3 cache access times by allocating an array of different sizes with a constant stride of 64bytes which gave me constant access times as the compiler optimization is so intelligent that it is recognizing the pattern and accessing the array value faster. Then I have allocated a larger array of size more than the L3 cache and by varying the stride per KB which will be constant from 0-32KB (L1 cache size), 32KB-256KB(L2 cache size) and 256KB-9MB(L3 cache size). This is because for every step of the stride there will be a miss in the cache at the cache sizes for every step and CPU tries to retrieve the data from previous level cache to the cache before it. So this procedure is followed for different patterns which resembles a memory mountain as follows.



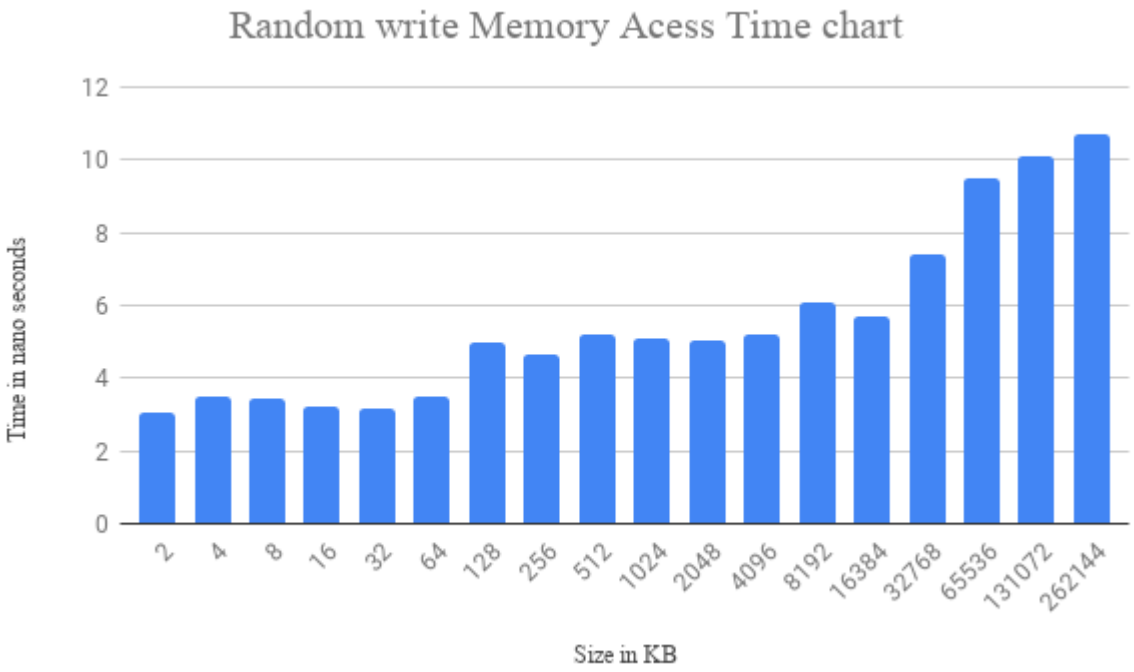
From the above graph the access times of the caches for linear read are as follows
L1 cache:2.27882ns
L2 cache : 5.678ns
L3cache : 10.0166ns
Main memory : 82.8952ns



From the above graph the access times of the caches for linear write are as follows
L1 cache:2.59497ns
L2 cache : 7.06401ns
L3cache : 9.66764ns
Main memory : 87.0303ns



From the above graph the access times of the caches for random read are as follows
L1 cache: 4.49595ns
L2 cache : 6.16069ns
L3cache : 10.9183ns
Main memory : 72.3485ns



From the above graph the access times of the caches for random write are as follows
L1 cache: 3.04307ns
L2 cache : 4.9754ns
L3cache : 6.10042ns
Main memory : 10.72415ns

ANSWERS TO THE QUESTIONS IN PART 1:

a) yes, latency can be represented by the access time of the memory hierarchy as it the time taken to access each element from that memory block. I took care of the overhead by removing the time taken by the base program from the time taken by the program with read/write methodology. Also, we used rdtsc to calculate the ticks properly to measure the access time.

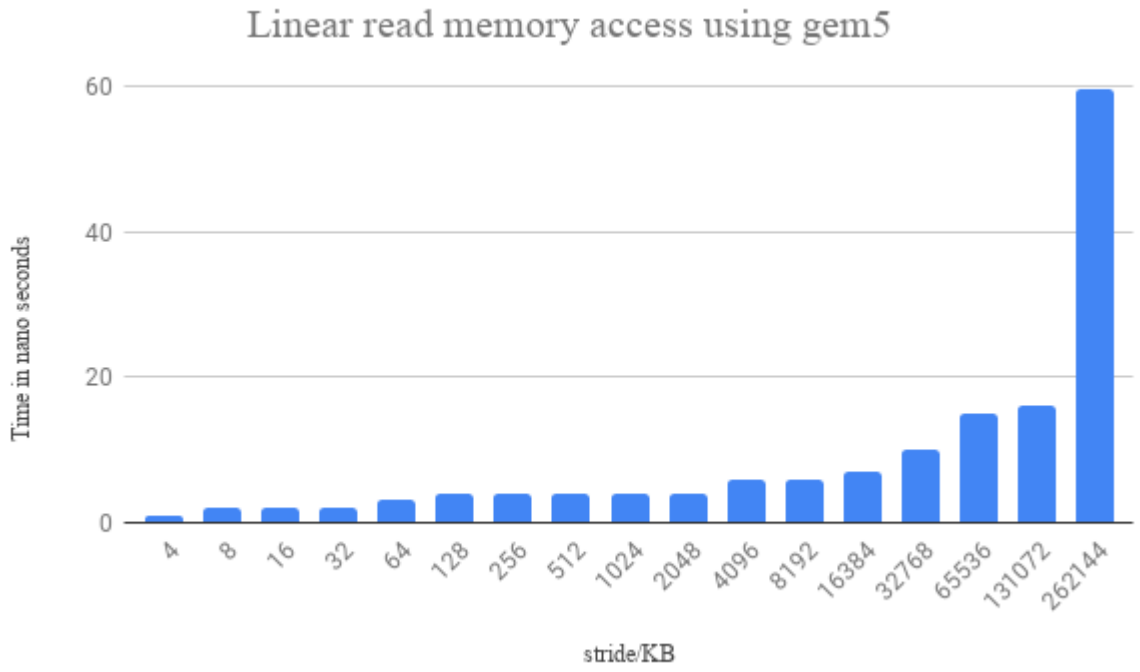
b)In linear access the compiler is so optimized that if we access a constant amount of block every iteration in a loop the prefetcher takes the block of same size which are contiguous so that making the linear access random.

But in random access the prefetcher cannot calculate which block to get in the next instruction beforehand. The write takes more time than read because of the write-through and write-back mechanisms of the cache where it need to write into the cache as well as it lower level memory.

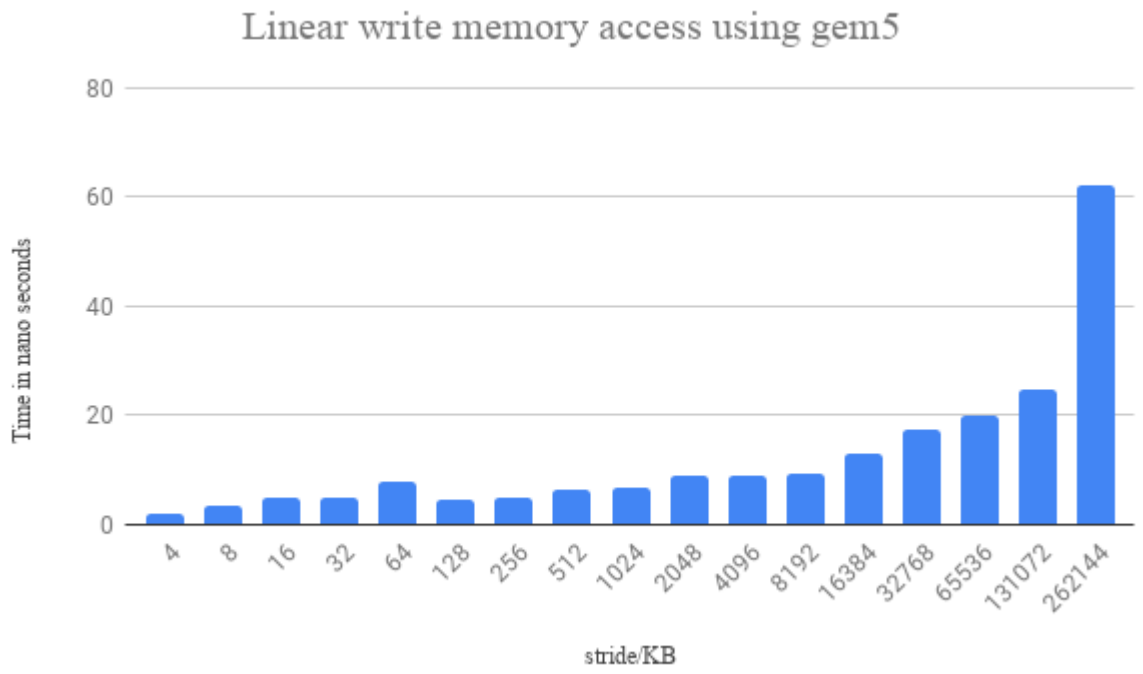
PART2:

Simulation of read/write patterns using gem5:

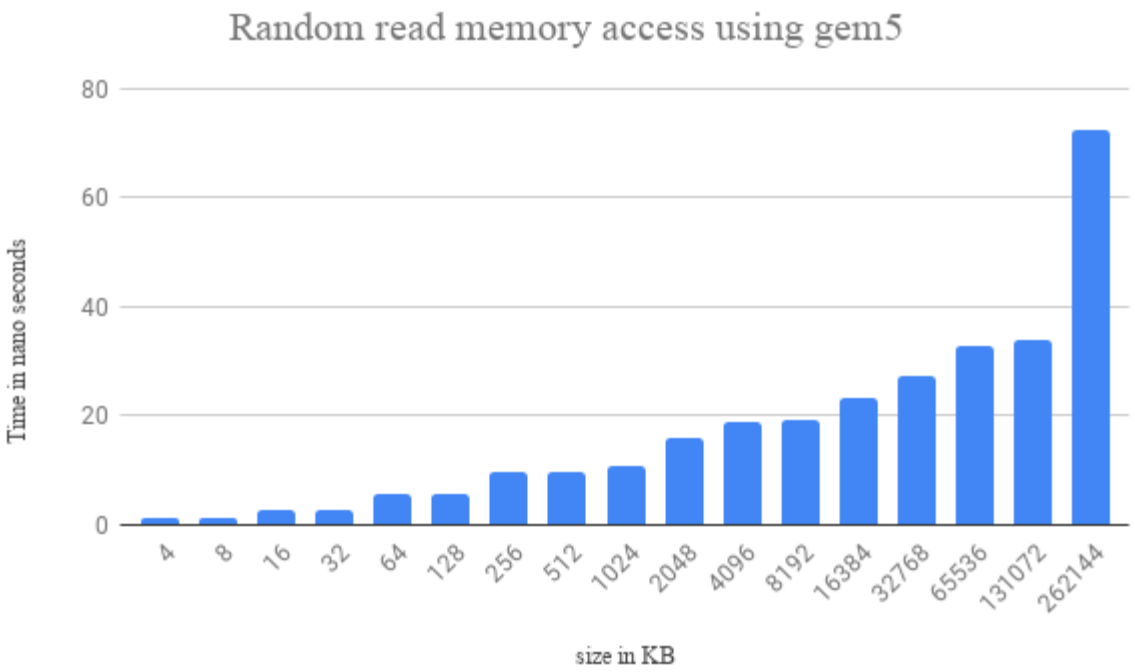
In this we used the gem5 simulator to measure the access time of the caches by limiting the L2 cache size to 256 KB and other cache adjustments.



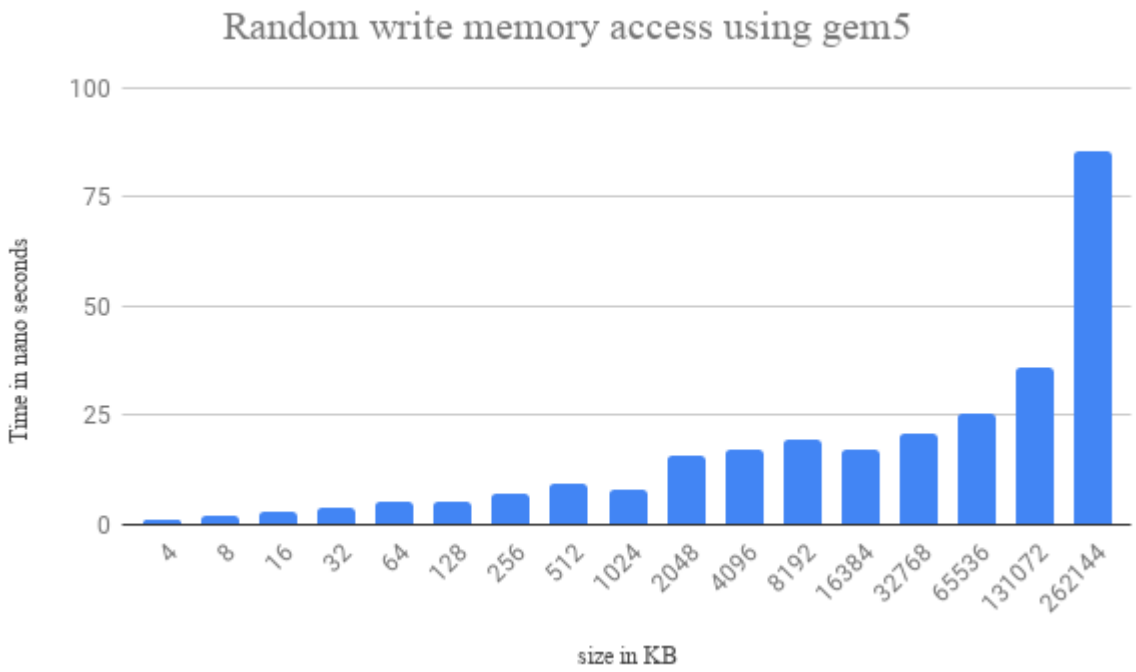
From the above graph the access times of the caches for random read are as follows
L1 cache: 2..06077ns
L2 cache : 6.03511ns
L3cache : 10.03461ns
Main memory : 59.48439ns



From the above graph the access times of the caches for random read are as follows
L1 cache: 2.15559ns
L2 cache : 4.5684ns
L3cache : 9.047715ns
Main memory : 62.34954ns



From the above graph the access times of the caches for random read are as follows
L1 cache: 1.148071ns
L2 cache : 5.687247ns
L3cache : 10.35419ns
Main memory : 72.39015ns



From the above graph the access times of the caches for random read are as follows
L1 cache: 1.99595ns
L2 cache : 3.64069ns
L3cache : 9.21818ns
Main memory : 85.67359ns