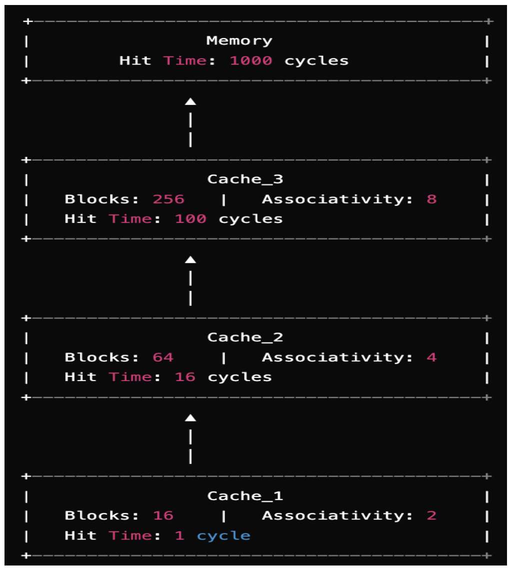
CSE 530: Fundamentals of Computer Architecture
Assignment - 1

Note : The Graphs are plotted such that Y-axis denotes Total Number of Cycles Taken, Cache Misses and AMAT. The X-axis denotes the variation in block sizes, associativity and number of levels. Also the miss rate, number of misses will have the same graphs and hit rate will have an inverted mirror graph. Due to complexity in visualization, I have not plotted it as all denote the same trends.

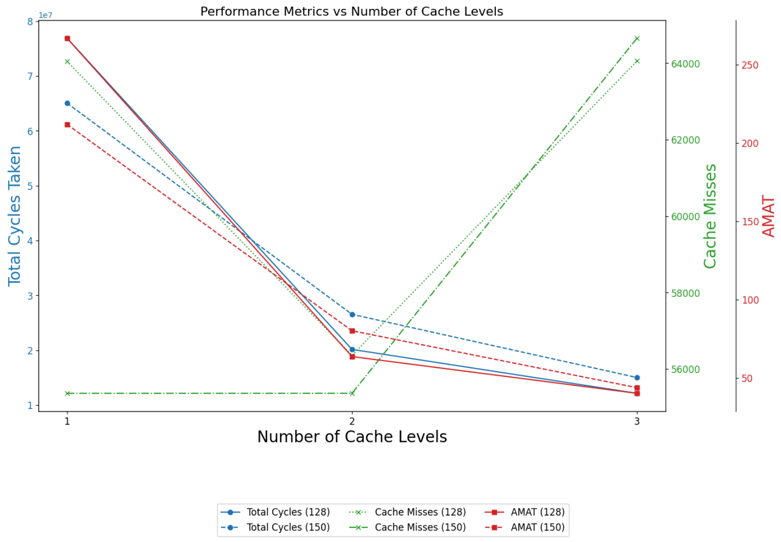
Bonus Question and Cache Architecture Block Diagram :

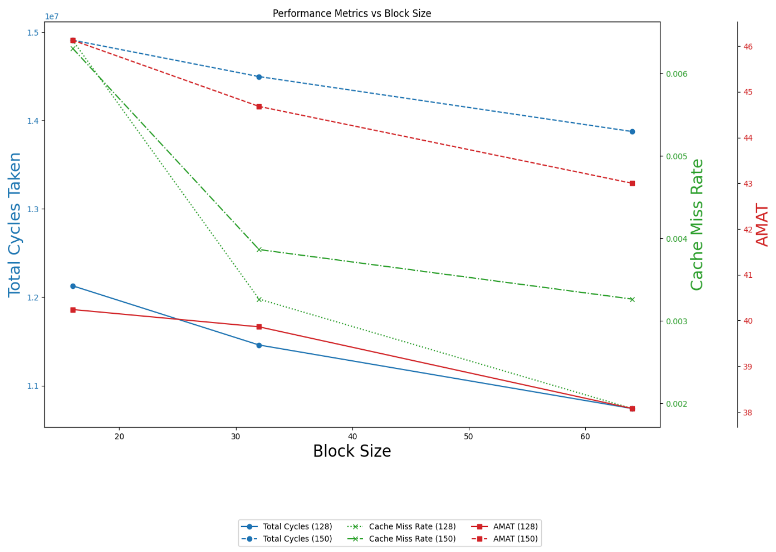
Operation	Writes	No. of Time Operations are Repeated (10^6 {Endurance}/ Number of writes)
Scatter(128)	96655	10.35 ~ 10
Scatter(150)	106155	9.42 ~ 9
Gather(128)	72821	13.73 ~ 14
Gather(150)	74491	13.42 ~ 13
Convolution(20)	66298	15.08 ~ 15
Convolution(32)	174607	5.73 ~ 6



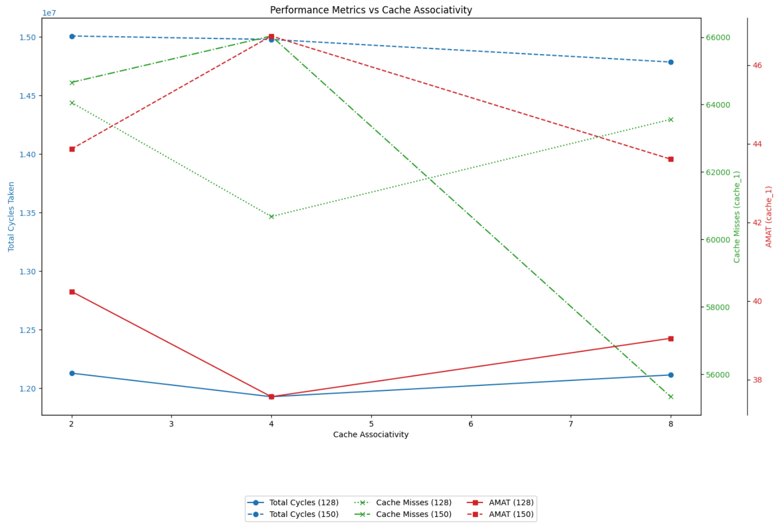
```
+----------------------------------------------------+
|                     Memory                         |
|            Hit Time: 1000 cycles                   |
+----------------------------------------------------+

                       ▲
                       |
                       |
+----------------------------------------------------+
|                     Cache_3                        |
|    Blocks: 256     |    Associativity: 8           |
|    Hit Time: 100 cycles                            |
+----------------------------------------------------+

                       ▲
                       |
                       |
+----------------------------------------------------+
|                     Cache_2                        |
|    Blocks: 64      |    Associativity: 4           |
|    Hit Time: 16 cycles                             |
+----------------------------------------------------+

                       ▲
                       |
                       |
+----------------------------------------------------+
|                     Cache_1                        |
|    Blocks: 16      |    Associativity: 2           |
|    Hit Time: 1 cycle                               |
+----------------------------------------------------+
```

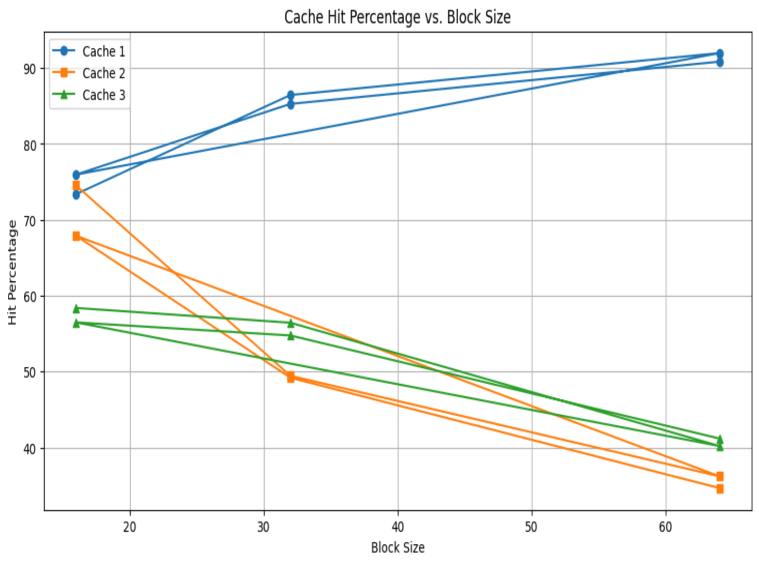
Scatter Operation :





Name : Ved Naik
PSU ID: 980091315



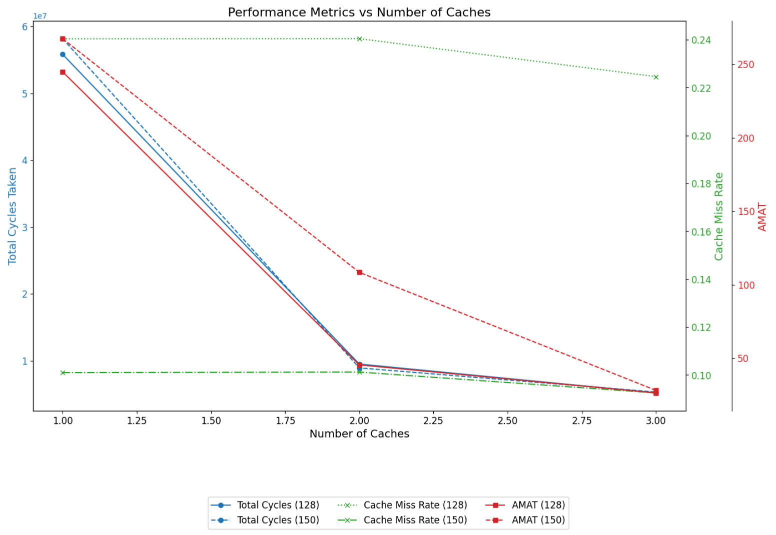


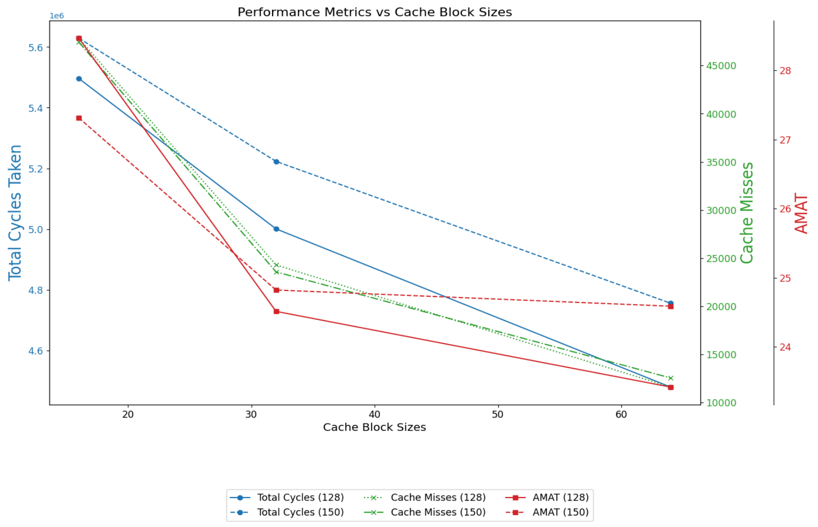
As we vary the Block size for L1 cache from 16 to 64 for matrix size 128x128, the total number of cycles decrease from 12.1M to 10.7M. Larger the Block size the more will be the hits for cache L1. Cache L1 has better miss rates when compared to L2 and L3 caches as L1 is more closer to CPU as compared to L2 and L3 caches. As the block size increases L2 and L3 caches miss rates drop significantly indicating improved data locality with larger block sizes.
Increasing cache associativity improves performance by reducing miss rates across all cache levels. Increasing associativity from 2 way to 4 way the associativity reduces misses in cache 3 from 5,355 to 5,199 and in cache 2 from 14,140 to 12,562. Improved associativity also lowers the average memory access time (AMAT), especially for cache 1. AMAT decreases as block size increases and associativity improves. Cache 3 shows a much higher AMAT indicating it is slower to access data compared to L1 and L2 cache. AMAT improvements are more seen with higher block sizes and increased associativity.

As associativity increases the improvements in cache performance start to show diminishing returns. While increasing associativity from 2 to 4, cache misses are reduced,but the reduction is minor because increasing associativity allows the cache to hold more blocks per set but not all workloads benefit equally from this increased flexibility. Higher associativity increases the complexity of cache lookup as the cache has to search through more possible blocks when the associativity is increased. This can lead to longer latency.

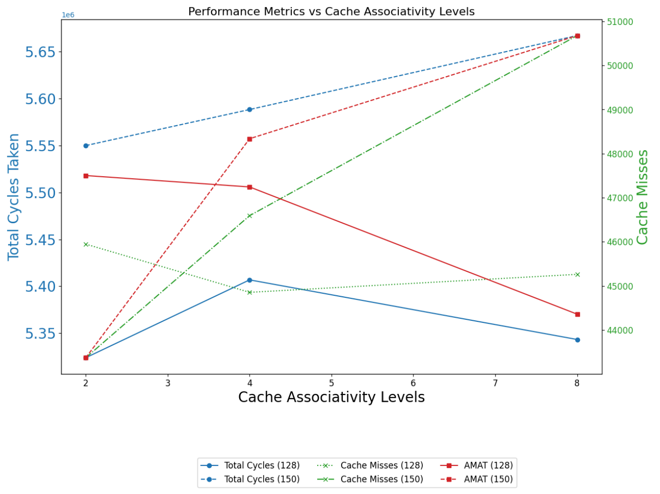
Increasing block size and associativity leads to reducing misses in L1 cache and performance gains whereas L2 and L3 cache experience increased transfer latency. This is because moving larger blocks of data between caches takes more time.
Matrices of power 2 (128x128) are more efficient due to better alignment with system architecture. Matrices not in power of 2 (150x150) have higher cycle counts and miss rates which leads to reduced performance. The 150x150 matrix leads to more fragmented cache access causing a higher rate of cache misses which in turn results in additional memory fetches and higher latency.

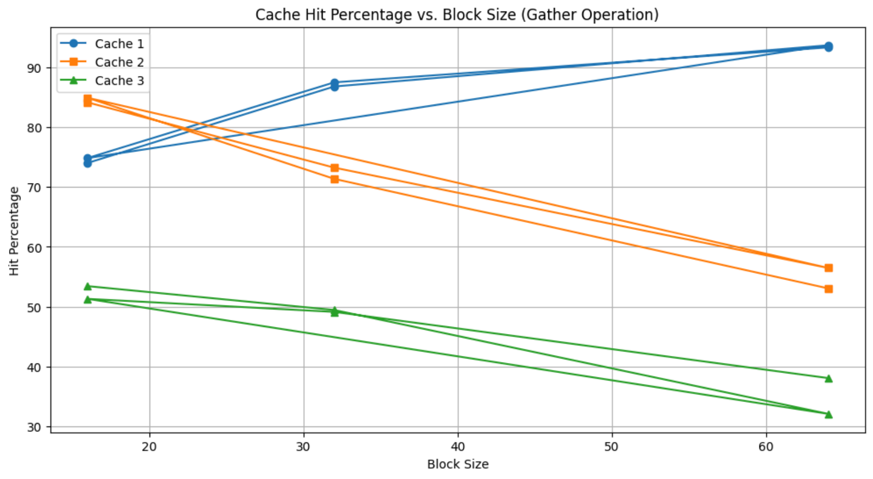
Gather Operation:





Name : Ved Naik
PSU ID: 980091315





Increasing associativity from 2 to 4 results in higher hit rates and fewer cache misses across levels. For 128x128 matrix when block size is 16, increasing associativity from 2 to 4 reduces L1 cache misses from 45,946 to 44,853 showing a moderate improvement. When increasing associativity from 4 to 8 the performance gains from increased associativity starts to diminish for L3 and higher order cache. Increasing cache associativity reduces conflict misses where multiple blocks compete for the same cache location. Higher associativity allows more blocks to be stored in the same cache set reducing the chance of miss.

With increase in block size, the total number of cycles and cache misses start to decrease. Increasing the block size from 16 to 64 for a 128x128 matrix size results in a reduction in total number of cycles from 5.5M to 4.48M which denotes a significant improvement in execution efficiency. Larger block sizes allow more data to be fetched from memory per cache line which increases spatial locality reducing the number of cache misses.

L1 Cache experiences the reduction in miss with misses dropping from 47,845 to 11,617 when increasing the block size from 16 to 64. While cache L1 benefits the most from larger block sizes L2 and L3 cache shows diminishing returns due to their slower nature. Even though fewer cache L2 and cache L3 misses occur with larger block sizes the AMAT for cache L2 and cache L3 increases due to the longer latency for accessing larger blocks. Due to the hierarchical nature of Cache, higher-level caches are smaller and faster, while lower-level caches are larger but slower.

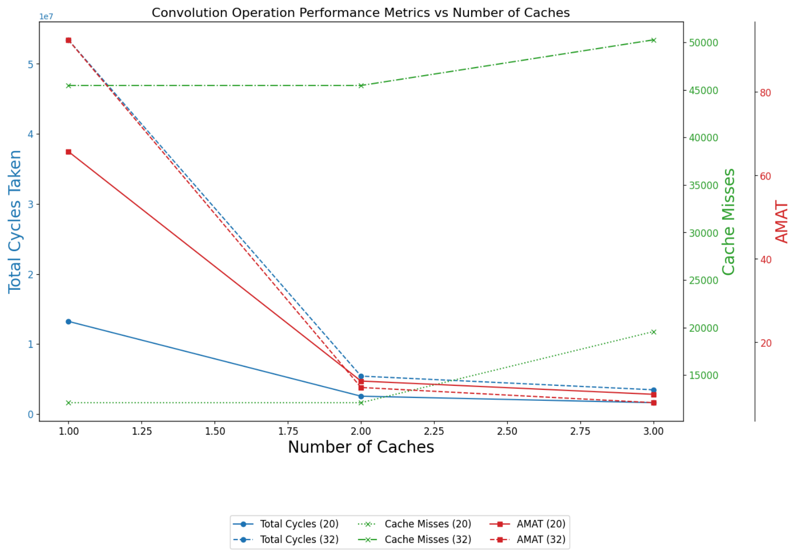
Convolution operation :

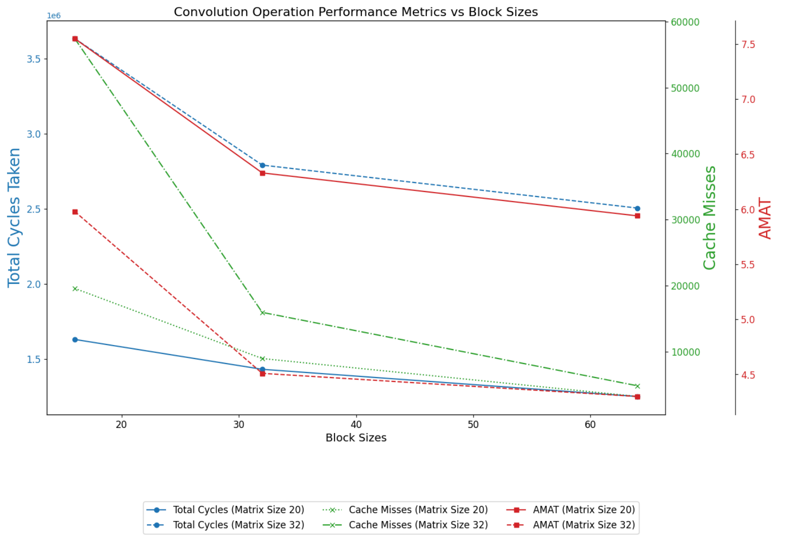
Block Size 64 appears optimal for minimizing total cycles and improving performance at Cache L1. But leads to higher misses and AMAT at Cache L2 and Cache L3. The increase in total cycles and AMAT with larger matrix size with smaller block sizes, shows memory access patterns and cache configuration leads to decrease in performance. There is a trade-off, increasing block sizes in Cache L1 can boost performance, but it might lead to decreased efficiency in higher level caches. Therefore, a balanced cache configuration is important.

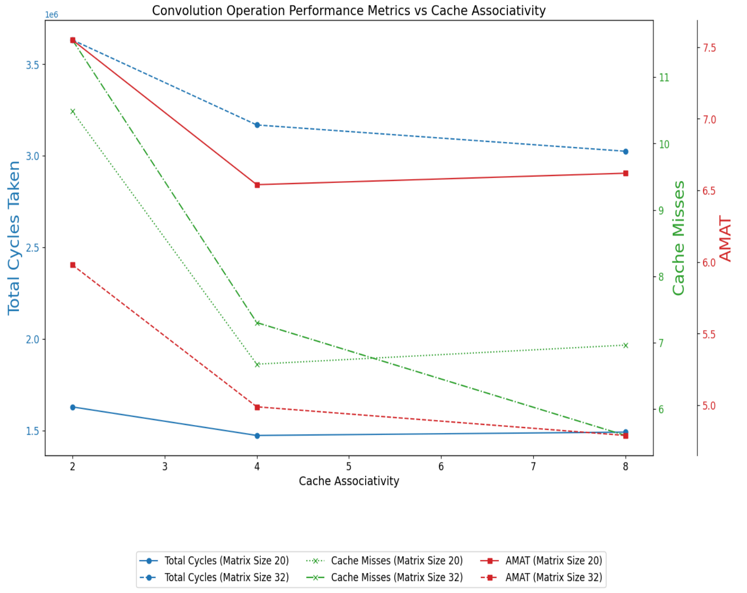
Higher associativity AMAT decreases the miss rates at each cache level. For the 20x20 matrix with block size 16 increasing the associativity from 2 to 4 lowers the AMAT for L1 cache from 7.549 to 6.544, for L2 cache from 62.470 to 83.034, and for L3 cache from 457.734 to 475.084. With increasing the associativity overall system performance is improved due to decrease in AMAT as average time to access memory decreases. Increasing cache associativity from 4 to 8 further reduces misses leading to better cache performance and fewer total number of cycles taken.

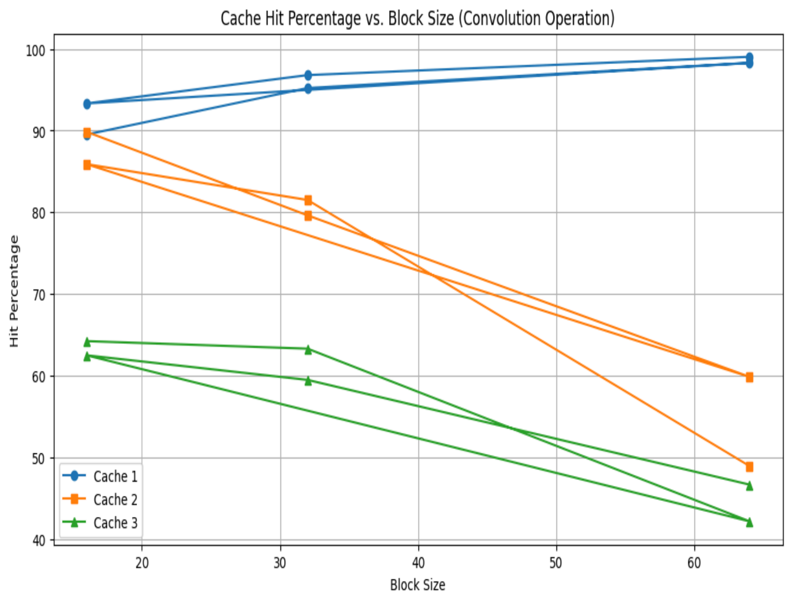
Matrix size of power 2, here 32x32, aligns well with typical cache line sizes and memory pages resulting in more efficient memory access patterns and fewer conflict misses cache utilization and performance. This alignment allows cache architectures to fully use their design efficiencies, reducing wasted space within cache lines and enhancing the hit rate especially in L1 cache. Matrix sizes which are not a power of 2, here 20x20 has less efficient cache utilization because of higher conflict misses and cache line fragmentation which leads to more frequent replacements and poor performance. Increasing block sizes from 16 to 64 is more useful in improving the performance of the 20x20 matrix as it removes initial inefficiencies more effectively and increases the hit rates.

Name : Ved Naik
PSU ID: 980091315









Comparison between three operations:

Scatter operation distributes data from a source to non-contiguous memory location. Scatter operation has non-sequential writes which results in inefficient use of cache lines as the data is being written to different locations, caches cannot predict which memory locations will be accessed next thus increasing the chances of cache misses.

In the gather operation the data is fetched from non contiguous memory locations and collected into a single structure. This creates irregular memory access patterns, which can lead to a higher number of cache misses, especially in lower L1 cache. Here the data is spread across memory so the cache cannot effectively use spatial locality which leads to poor cache line utilization causing frequent cache misses.

Convolution operation accesses data in a highly regular and predictable manner and has better cache utilization than scatter and gather operations. Due to the sliding of filters over input data, memory is accessed in a contiguous fashion. As a result, convolution operations benefit from both spatial and temporal locality which allows the cache to effectively prefetch data into cache lines. Caches can reuse previously loaded data thus minimizing cache misses and improving performance.

Name : Ved Naik
PSU ID: 980091315