Faster SQL Server Access with Solid State Disks

By Woody Hutsell
Texas Memory Systems

July 2008

Contents

Section 1

Executive Summary

This whitepaper discusses methods for improving SQL Server database performance.

It discusses methods for identifying I/O performance bottlenecks and points out components that are the best candidates for migration to a solid state disk. The paper also includes an in-depth explanation of solid state disk technology and possible implementations.

For more in-depth information, visit www.superSSD.com or contact one of the following:

-Existing customers contact support@superSSD.com.
-Potential customers contact sales@superSSD.com.

Section 2

Perspectives from Industry

From Chapter 6 of Tuning and Sizing NT Server by Curt Aubley, reprinted on Microsoft.com:

"No matter how well you sized and tuned your NT Server disk subsystem, the disk subsystem is still composed of physical devices that are slow when compared to RAM. There may be times, regardless of how well you have load balanced and tuned your disk subsystem, when 'hot spots' still exist. 'Hot spots' are defined here as areas of the disk subsystem that application(s) consistently request services from which greatly impact the application's performance. Consider another technique to alleviate hot spot situations; try implementing RAM disks."

From the Morgan Keegan System Area Network Conference, Solid State Caching, page 85-86:

"Since 1964 the capacity of disk drives has increased by over 6,000 times while the average seek latency has only seen about a tenfold improvement. The continued increase in capacity without corresponding improvement in access times has caused a steady decline in the access density of disk media, which is the ratio of I/Os per second divided by the capacity of the disk drive. As a result, many storage administrators are finding that in spite of the fact that they are adding storage capacity at an unprecedented pace, system performance does not improve. In other words, their storage infrastructures are I/O bound. One way to solve this dilemma is to spread the information access workload across a large number of systems. In many situations, however, a more efficient method of improving performance is the implementation of a solid-state disk (SSD) caching solution. SSD offers the fastest storage media available."

"However, we believe SSD caching will see increasing adoption in the system network as the optimal storage resource for certain data types requiring high levels of availability. The return on investment of SSD increases dramatically in a centralized storage configuration in which the cost may be amortized across a larger infrastructure and SSD can be added incrementally as needed…"

Section 3

The Problem of I/O Wait Time

Often, additional processing power alone does little or nothing to improve SQL Server performance. This is because the processor, no matter how fast, finds itself constantly waiting on mechanical storage devices for its data. While every other component in the "data chain" is solid state, hard drives are mechanical, relying on physical movement around a magnetic platter to access information.

In the last twenty years, processor speeds have increased at a geometric rate. At the same time, however, conventional storage access times have only improved marginally. The result is a massive performance gap, felt most painfully by database servers, which typically carry out far more I/O transactions than other systems. Super fast processors and massive amounts of bandwidth are often wasted as storage devices take several milliseconds just to access the requested data.

When servers wait on storage, users wait on servers. This is I/O wait time. Solid state disks solve the problem of I/O wait time by offering faster access times and more I/O transactions per second than monolithic RAID.[1] Figure 1 shows the performance of the RamSan-500 cached flash SSD and the RamSan-440 RAM SSD in sustained random I/O's per second versus other storage options.

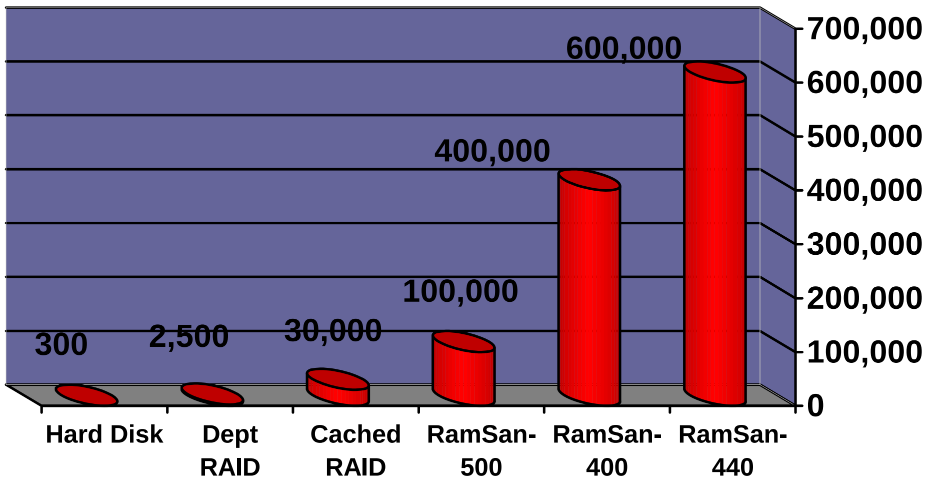


Figure 1: Chart-- Maximum I/Os per Second

[1] The performance numbers cited here are those of the RamSan-500, RamSan-400, and RamSan-440 SSD, sold by Texas Memory Systems.

Section 4

Traditional Approaches to Improving SQL Server Performance

Decreasing application performance under heavy user loads is not a new story for most enterprises. As the number of concurrent users increase, the response time to users worsens. The knee-jerk reaction to this problem is to look at two solutions for database performance problems:

- **Server and processor performance**. When performance wanes, one of the first things that most IT shops do is add processors to servers or add servers to server farms.
- **SQL Statements**. Enterprises invest hundreds of thousands of dollars squeezing every bit of efficiency out of their SQL statements. The personnel required to painstakingly evaluate and iterate the code are a major cost for many IT shops.

Adding servers and processors has a minimal impact on database performance and compounds the resources wasted as even more processing power waits on the same slow storage. Tuning SQL may result in performance improvements, but even the best SQL cannot make up for poor storage I/O. In many cases, applications cannot support features that rely heavily on disk I/O, for example, developers often remove features that result in large queries and return large data sets in order to protect application performance.

When system administrators look to storage, they frequently try these different approaches to resolving performance problems:

- **Increase the number of disks**. Adding disks to JBOD or RAID is one way to improve storage performance. Increasing the number of disks spreads the I/O from a database across more physical devices. As with other approaches, this has a trivial impact on decreasing the problem.
- **Move the most frequently accessed files to their own disk**. This approach delivers the best I/O available from a single disk drive; however, the best I/O capability of a single hard disk drive is very limited.

The best solution to the performance gap is to implement solid state disks for the most frequently accessed database components.

Section 5

Introduction to Solid State Disks

Strictly, a solid state disk (or SSD) is any storage device that does not rely on mechanical parts to input and output data. Typically, however, the term refers to storage devices that use memory (RAM or Flash) as the primary storage media. Data is stored directly on memory chips and accessed from them. This generally results in storage speeds far greater than is even theoretically possible with conventional, magnetic storage devices. To fully make use of this speed, SSDs typically connect to servers or networks through multiple high-speed channels.

RAM Solid State Disks

What separates a RAM solid state disk from conventional memory is non-volatility. A RAM SSD typically includes internal batteries and backup disks so that, in the event of power loss or shutdown, the batteries keep the unit powered long enough for data to be written onto backup disks. Because of this, SSDs offer the raw speed of system memory without the disadvantage of losing data when powered down. Because of the lack of mechanical devices in the main data chain, SSDs typically have lower maintenance costs and higher reliability (including a higher MTBF) then conventional storage.

DDR-RAM memory, the primary storage media, fill the back of the unit. The front contains backup batteries, backup mechanical drives, and a front panel user interface.

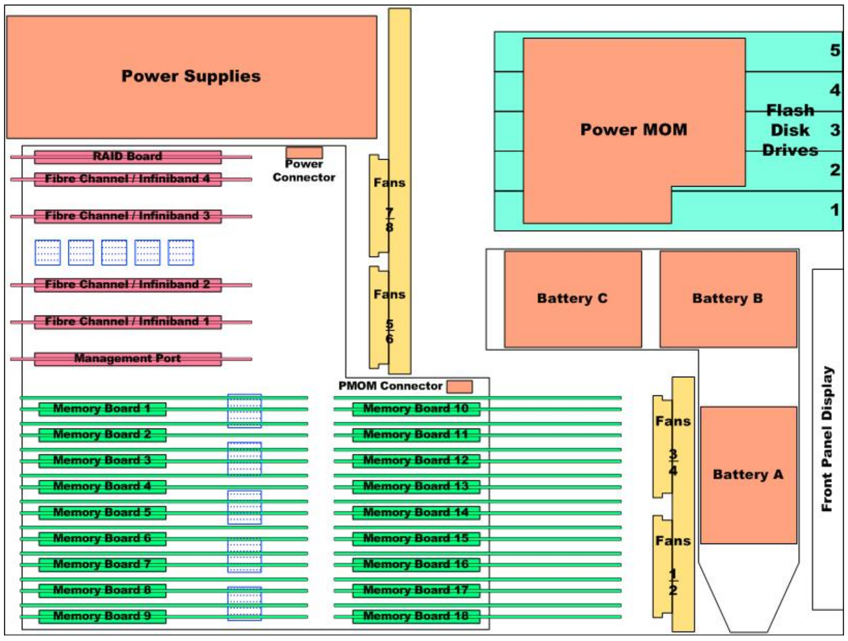


Figure 2: An Inside Look at the RamSan-400 Solid State Disk.

Cached Flash Solid State Disks

Cached flash solid state disks seek to balance the performance offered by a large RAM cache and the fast reads, high density and lower price per capacity of flash memory. A cached flash solid state disk is as fast as a RAM solid state disk for cache hits and still 20 times faster than typical hard disk based solutions if there is a cache miss (i.e. a read from the flash memory).

Section 6

Identifying I/O Subsystem Problems

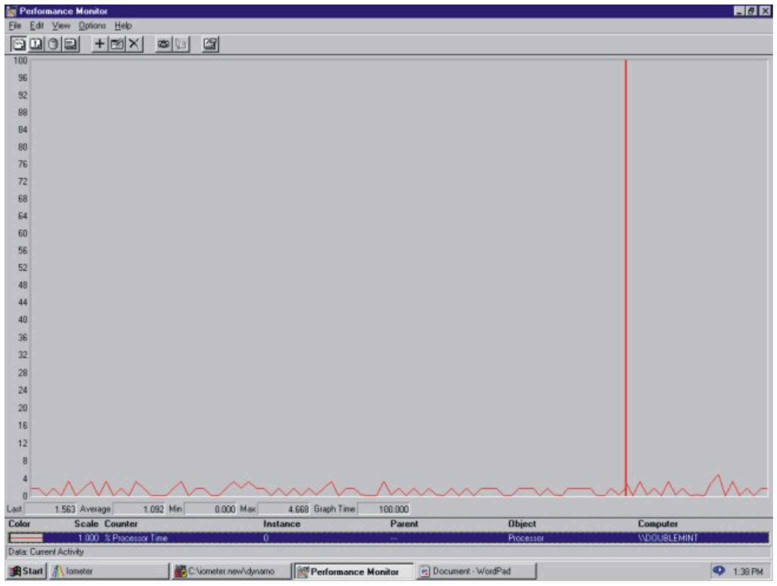
While a solid state disk can speed up almost any SQL Server database, it is most needed in installations where servers are experiencing I/O wait time. A number of elements are involved in server I/O. The PCI (or other) bus, host bus adapter, interface, storage network switch, RAID controller, and hard disk drives are all involved in every I/O between server and storage. Theoretically, any one of these points can cause an I/O bottleneck. In practice, however, the hard disk drives are the most likely culprit. Simply put, every component in the I/O process is solid state except for the hard disk drives. Therefore, when you identify I/O wait time, hard disk drives are the most likely cause. Adding a solid state disk drive can eliminate I/O wait time.

The best way to identify I/O wait time is to look at operating system performance. The tools to evaluate operating system performance vary by operating system.

For Microsoft Windows operating systems, the best tool for system performance analysis is the Performance Monitor. Unfortunately, the Performance Monitor does not provide the actual I/O Wait Time statistic, but it does include actual processor performance levels. The "Processor: % Processor Time" statistic shows the actual work being done by the processor. If transactions are hitting your system hard and yet your "% Processor Time" is well under 100%, it is possible to infer an I/O wait problem. Systems that implement solid state disk show high "% Processor Time" numbers.

For example, the following two figures show screen shots from Windows Performance Monitor.

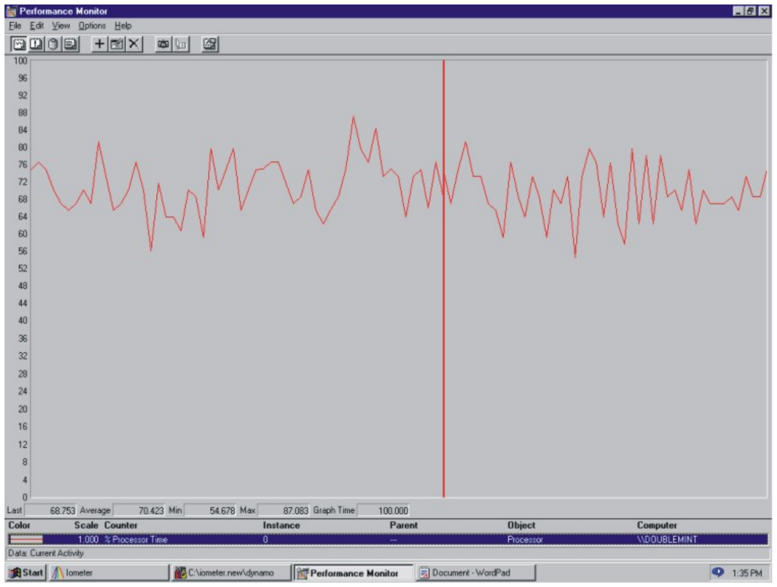
On the following page, **Figure 3** shows the "Processor: % Processor Time" for a Windows system running Intel's IOMeter performing 100% random writes to a hard disk drive. In this exhibit, the processor utilization averages around 1.8%. Running additional applications on this system would only increase the processor utilization marginally, because the processor is waiting on I/O from the hard disk drive. In this example, IOMeter shows that on average there were 150 writes per second (150 IOPS) to the disk drive.



```
Last 1.563  Average 1.092  Min 0.000  Max 4.668  Graph Time 100.000
Color   Scale  Counter            Instance   Parent   Object      Computer
        1.000  % Processor Time   0          ---      Processor   \\DOUBLEMINT
Data: Current Activity
```

Figure 3: Processor Performance when Writing to Hard Disk

Below, **Figure 4** shows the exact same system, exact same access specifications in IOMeter running against a Texas Memory Systems RamSan solid state disk. In this example, the processor averages 68% utilization. The IOMeter shows that 27,000 writes per second are going to the RamSan (27,000 IOPS). As it turns out, this IOPS number is a limitation of the host bus adapter used in the demonstration. Nonetheless, this example clearly illustrates how a solid state disk can improve processor utilization for a Windows based system.



```
Last 68.753  Average 70.423  Min 54.678  Max 87.083  Graph Time 100.000
Color   Scale  Counter            Instance   Parent   Object      Computer
        1.000  % Processor Time   0          ---      Processor   \\DOUBLEMINT
Data: Current Activity
```

Figure 4: Processor Performance when Writing to Solid State Disk

In addition to processor indicators, Microsoft recommends looking at the "Physical Disk: Average Queue Length" and "Physical Disk: Disk Bytes Per Second" to detect bottlenecks in the disk subsystem. Please visit our **Diagnosing Windows I/O Bottlenecks Using Performance Monitor** for more details on setting up Perfmon testing.

If the "Physical Disk: Average Disk Queue Length" is consistently high, consider moving files that are located on that disk to the solid state disk. A "Disk Queue Length" greater than 3 (per disk in the RAID set) indicates a problem.

The Physical Disk: Disk Bytes Per Second" indicator helps visibly determine the level of performance you are getting when your disk array levels off. At the point disk bytes per second levels off and physical disk queue lengths increase, is the point that your system is demonstrating an I/O bottleneck. In addition, if you simultaneously graph Processor Utilization, you will notice that processor time tends to decrease when your system experiences an I/O bottleneck. One you have narrowed down the time range causing I/O problems, further refine your search for the problem by using Perfmon's ability to narrow details based on reads/writes and by drive arrays. In this way, you can quickly determine which array is causing I/O problems.

These tools allow you to observe the I/O wait time for a Windows based system.

Section 7

SQL Server Components That Should Be Moved to Solid State Disk

Once you determine that your system is experiencing I/O subsystem problems, the next step is to determine which components of your SQL Server database are experiencing the highest I/O and in turn causing I/O wait time. Examine the following database components:

Entire Database. There are some databases that should have all of their files moved to solid state disk. These databases tend to have at least one of the following characteristics:

- High concurrent access. Databases that are being hit by a large number of concurrent users should consider storing all of their data on solid state disk. This will make sure that storage is not a bottleneck for the application and maximize the utilization of servers and networks. I/O wait time will be minimized and servers and bandwidth will be fully utilized.
- Frequent random accesses to all tables. For some databases, it is impossible to identify a subset of files that are frequently accessed. Many times these databases are effectively large indices.
- Small to medium size databases. Given the fixed costs associated with buying RAID systems, it is often economical to buy a solid state disk to store small to medium sized databases. A RamSan-400, for example, can provide 128GB of database storage for the price of some enterprise RAID systems.
- Large read-intensive databases. Given the fixed costs associated with architecting a RAID system for performance (buying a large cache, buying a lot of spindles for striping), it is economical and much faster to buy a RamSan-500 cached Flash solution in order to accelerate large read-intensive databases. A single RamSan-500 can scale to 2TB of capacity.
- Database performance is key to company profitability. There is some subset of databases that help companies make more money, lose less money, or improve customer satisfaction if they process faster. Solid state disks can help make these companies more profitable.

Transaction Logs. Transaction logs are one of the most important factors in the write performance for SQL Server databases. Whenever a database write occurs, SQL Server creates a transaction log entry. Each SQL Server database has at least one transaction log (*.ldf). SQL Server considers operations committed once the write to the transaction logs is complete.

Because the transaction logs are a source of constant I/O during database operation, the transaction logs should be stored on the fastest disk possible. Writing a transaction log to a solid state disk is a natural way to improve overall database performance.

The Microsoft web site: MSDN Home > MSDN Library > Microsoft SQL Server > Database Design > Physical Database Design has the following to say about optimizing performance of transaction logs:

> "Create the transaction log on a physically separate disk or RAID (redundant array of independent disks) device."

Texas Memory Systems recommends that the separate disk be a RamSan solid state disk.

Temporary Database (tempdb). SQL Server uses tempdb to support temporary data during certain operations. The tables support complex queries, joins, and index creations. Because tempdb supports many kinds of operations, it is essential that they be located on the fastest possible storage.

When complex operations occur, they complete more quickly if the tempdb is moved to solid state disk. Because the I/O to the tempdb can be frequent, disk drives cannot handle them easily.

The Microsoft web site: MSDN Home > MSDN Library > Microsoft SQL Server > Database Design > Physical Database Design says the following about optimizing performance of tempdb:

> "Place the **tempdb** database on a fast I/O subsystem to ensure good performance. Stripe the **tempdb** database across multiple disks for better performance. Use filegroups to place the **tempdb** database on disks different from those used by user databases."

Indexes. An index is a data structure that speeds up access to database records. An index is usually created for each table in a database. SQL Server updates these indexes when adding records and modifying records' identifying data. When a read occurs, SQL Server consults an index to access the correct record quickly. Furthermore, many concurrent users may read any index simultaneously. This activity to the disk drive creates frequent, small, and random transactions, which makes the disk drives unable to keep up with demand and I/O wait time results.

Storing indexes on a solid state disk can increase the entire application's performance. For on-line transaction processing (OLTP) systems with a high number of concurrent users, this results in faster database access. Historically, indexes have been a common SQL Server component to be moved to solid state disk because they can be recreated from the existing data.

The Microsoft web site: MSDN Home > MSDN Library > Microsoft SQL Server > Database Design > Physical Database Design > Data Placement Using Filegroups says the following about allocating indexes to their own Filegroup:

> "By default, indexes are created on the same filegroup as the base table on which the index is created. However, it is possible to create nonclustered indexes on a filegroup other than the filegroup of the base table. By creating the index on a different filegroup, you can realize performance gains if the filegroups make use of different physical drives with their own controllers."

In this case, Texas Memory Systems recommends moving the filegroups that contain the indexes to solid state disk.

Frequently Accessed Tables. Sometimes, just 5-10% of tables account for a large percentage of all database activity and thus I/O to storage. When a large number of users hit a table, they are likely going after different records and different attributes. As a result, the activity on that table is random. Disk drives are notoriously bad at servicing random requests for data. In fact, the peak performance of a disk drive drops as much as 95% when servicing random transactions. When a table experiences frequent access, transaction queues develop where other transactions are literally waiting on the disk to service the next request. These queues are another sign that the system is experiencing I/O wait time.

Moving frequently accessed tables to a solid state disk makes sense. Unlike traditional hard drives, random transactions do not affect solid state disk performance. Additionally, solid state disks have faster access times than disk drives. Therefore, moving frequently accessed tables to solid state disk can improve application performance up to 10x.

SQL Server supports moving specific tables to specific filegroups, which enables you to move the most frequently accessed tables to solid state disk easily.

Section 8

Identifying the Most Frequently Accessed SQL Server Tables

The SQL Server Profiler tool allows database administrators to monitor events in SQL Server. The profiler monitors SQL Server activities by tracing selected events, including database objects. Database objects include databases, tables, indexes, views, and stored procedures. The SQL Server Profiler specifically supports monitoring when a table is opened. Tables are opened during "Select," "Insert," and "Delete" statements. By using the profiler, a database administrator can monitor the number of times all tables are opened during the trace period. The results of the trace can be sent to a table for further analysis to determine which tables were most frequently opened during the trace period and thus which tables are the best candidates for placement onto a solid state disk.

Microsoft cautions that "Object" related traces could impose a significant overhead on database performance because of the frequency that objects are accessed. If you have a test environment that models production activity and transaction mixes, this may be the best place to use an SQL Server Profiler "Object" trace. Otherwise, make sure you do not use the Profiler during peak transaction periods unless you are willing to take the performance hit.

More information on SQL Server Profiler and database objects can be found at: MSDN Home > MSDN Library > Microsoft SQL Server > Monitoring Server Performance and Activity > Monitoring with SQL Profiler > Monitoring with SQL Profiler Event Classes > Objects Event Category

Section 9

Customer Case Study: Pilgrim A|S

PILGRIM A|S is a Danish brand producing fashion jewellery and accessories – including a range of eyewear and watches - for men and women. All products are designed exclusively in Denmark. PILGRIM's designs continually evolve in step with the latest fashion trends, yet, remains distinct from similar products available on today's market. PILGRIM differentiates itself with its courage to explore new paths – for example, by combining unique materials and colors in new and unexpected ways. Behind the scenes, PILGRIM's competitive advantage also lies in its courage to explore new paths – for example, its use of the RamSan solid state disk to eliminate performance bottlenecks for its central IT application.

The Challenge: Finding the source of terrible performance during peak periods

In the Spring of 2006, PILGRIM was facing problems with the IT application at the core of its business – Microsoft Dynamics AX (formerly Microsoft Axapta) in a 3-tiered environment with Microsoft SQL Server 2005 Enterprise. PILGRIM used Dynamics AX for everything from manufacturing and inventory control to accounting and business intelligence. Users began to complain of terrible response times during peak usage periods. The source turned out to be the high number of data intensive updates made to keep the system up to date. The result was widespread performance degradation across the company. "Our existing SAN provided a continuous flow of IOs," said Anders Schack Petersen of PILGRIM, "but since we perform a lot of data intensive updates, it couldn't cope, resulting in long response times."

PILGRIM started investigating the source of the application performance issues. Their configuration was a two-tiered SQL server run by Axapta. Furthermore, they were able to eliminate the network from the investigation after discovering the Fibre Channel SAN had bandwidth to spare. Finally, the culprit was discovered – the storage subsystem, currently an IBM DS4300. While the system was designed to be fast, it was unable to keep up with the intense, random IO of burst periods.

According to Petersen, he first considered expanding the lagging storage subsystem. Fortunately, Petersen was also "closely following the development of online massive multiplayer game EVE-Online." Its developers, CCP Games, had used a RamSan solid state disk to eliminate performance bottleneck associated with up to 30,000 concurrent users on a SQL Server database. "Their excitement with the RamSan led us to contact them about their experiences, which in turn lead us to try a demo version of the RamSan."

The Solution: A Texas Memory Systems RamSan Solid State Disk

PILGRIM's first observation upon receiving their evaluation unit was "The RamSan is amazingly easy to install, much easier than other Fibre Channel storage devices we have tested," said Petersen. The demonstration unit was quickly integrated into the test environment and immediately the team began to see the potential of the RamSan.

Result: "The World's Fastest Storage" delivers unparalleled, cost-effective performance

After some testing of the product in the production environment, PILGRIM was able to conclude that the RamSan could easily handle the bursts of activity caused by their growing business. PILGRIM is able to handle 20 times as many I/O-per second - from 1,200 I/O-per second with its old system to 24,000 I/O-per second with the RamSan-400 at its peak.

More importantly, PILGRIM decided on the RamSan because of the systems performance scalability. "Even considering that we can never use the full potential of the RamSan, the cost per IO was still better than other SAN solutions that we looked at", said Petersen, "We are excited that as we grow we can expand the system's capacity and still have plenty of performance headroom."

PILGRIM decided to start with a 64GB RamSan-400 system with plans to upgrade to a 128GB unit within one or two years. The RamSan system is field upgradeable to higher capacities allowing companies to grow their solution as their database grows. The system was purchased directly from Texas Memory Systems and includes 24x7 telephone support to handle inquiries any time of the year.

When asked what advice they would offer to potential buyers of solid state disk, Petersen said "Don't be scared. The technology has proven itself superior in many circumstances. You just have to know when to apply it. For applications with smaller databases but heavy load, it's a life saver".