Faster Sybase Performance with Solid State Disks

By Woody Hutsell
Texas Memory Systems

July 2008

Contents

Section 1

Executive Summary

This whitepaper discusses methods for improving Sybase database performance, using solid state disks (SSD) to accelerate the most resource-intensive data that slows performance across the board.

To this end, 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. An in-depth explanation of SSD technology and possible implementations is also included.

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

Introduction

Sybase Corporation and numerous professionals have recommended improving database performance by placing the data on fast storage devices (in Sybase's words, "the fastest disks available"). Since solid state disks are the fastest storage devices on the market, they are a natural fit for this kind of use.

Database administrators or managers interested in solid state disk generally fit into two categories:

- They are designing a hardware infrastructure for a new database system and wish to possess maximum performance potential.
- They have achieved all possible performance improvements through software and could use additional multipliers on performance.

In either case, a solid state disk used in the manners prescribed here is an efficient use of available resources. Use of a solid state disk, obviously, does not replace the need to tweak settings and statements towards the end of greater performance. It is simply a cost-effective hardware solution meant to work alongside software improvements to achieve maximum customer satisfaction and ROI.

Section 3

The Problem of I/O Wait Time

Often, additional processing power alone will do little or nothing to improve Sybase 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" moves in terms of computation times and the raw speed of electricity through a circuit, hard drives move mechanically, 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, RamSan-400 RAM SSD and RamSan-440 RAM SSD in sustained random I/O's per second versus other storage options

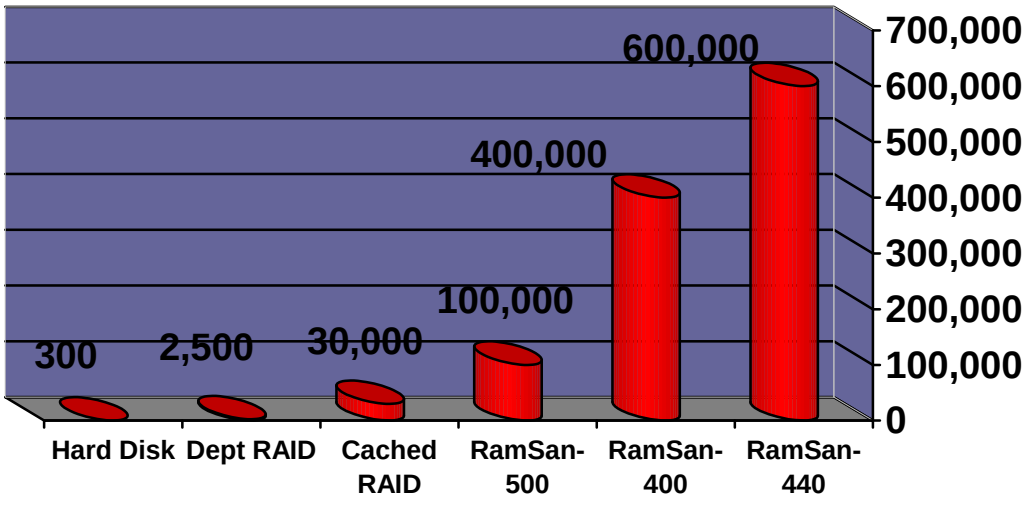

Figure 1: Chart-- Maximum I/Os per Second

Section 4

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 Flash drives, and a front panel user interface.

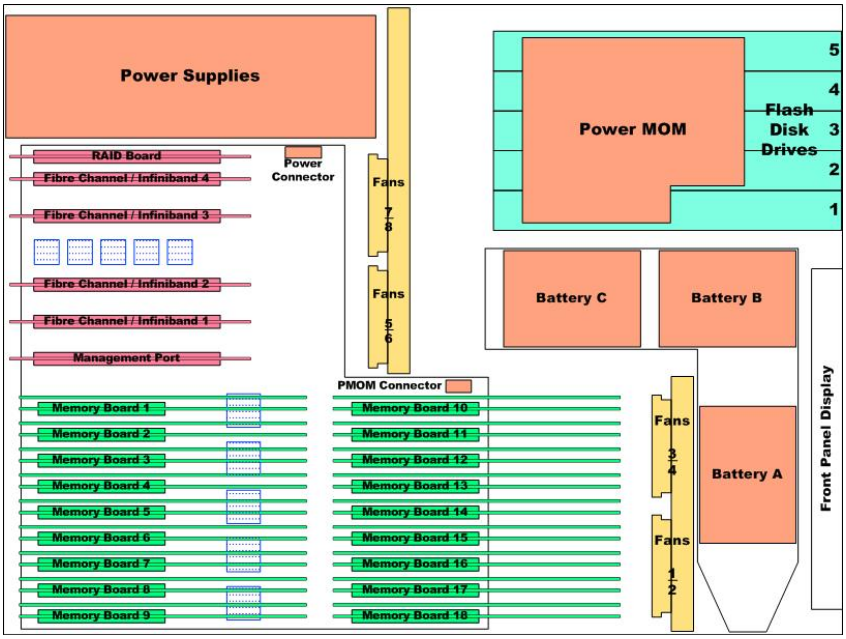

Figure 2: An Inside Look at the RamSan-440 Solid State Disk.

Cached Flash Solid State Disks

Cached Flash SSD seek to balance the performance offered by a large RAM cache and the fast reads, high density and lower price per capacity provided by Flash memory. A Cached Flash SSD 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 5

Identifying I/O Wait Time

There are a variety of tools for monitoring the degree of I/O Wait Time in a system. Here are three different approaches, one using tools from Sybase Adaptive Server and two others using OS tools for Windows and UNIX.

Sybase

The `sp_sysmon` procedure in Sybase can give specific data on I/O wait time and its effects on database performance. To use it, execute the following using isql:

```
sp_sysmon [interval] diskio
```

This command will execute **`sp_sysmon`** in the [*`interval`*] specified (in the form *hh:mm:ss*) and return only the "Disk I/O Management" section of the report.

Figure 3: A sample report from `sp_sysmon`.

```
Disk I/O Management
-------------------

  Max Outstanding I/Os         per sec   per xact    count  % of total
  -------------------------  ---------  ---------  -------  ----------
    Engine 0                       n/a        n/a     1220       n/a
    Engine 1                       n/a        n/a     1421       n/a
    Engine 2                       n/a        n/a     1860       n/a
……
  Total Requested Disk I/Os        n/a        n/a     4501       n/a

  Completed Disk I/O's
    Engine 0                      25.0        0.2      952      28.9 %
    Engine 1                      21.1        0.2     1101      33.4 %
    Engine 2                      18.4        0.2     1242      37.7 %
……
    Total Completed I/Os          64.5        0.6     3295
……
```

By comparing "Requested Disk I/Os" to "Completed Disk I/Os," the degree of literal I/O wait time in the system can be measured. Any significant difference in these numbers indicate I/O wait time of some kind.

Note that Sybase specifies `sp_sysmon` contributing 5 to 7% overhead while it runs on a single CPU server, and more on multiple CPU servers. Also, since `sp_sysmon` uses the same internal counters as the Adaptive Server Monitor and resets them with use, the Adaptive Server Monitor should not be used simultaneously. Further details on the use of `sp_sysmon` are available in the Sybase Performance and Tuning Manual.

Windows Server OS Performance Monitor

For Microsoft Windows operating systems the best tool for system performance analysis is Performance Monitor. Unfortunately, Performance Monitor does not provide actual I/O Wait Time statistics. It does, however, include realtime processor performance levels. "Processor: % Processor Time" measures the actual work being done by the processor. If a system is hit hard by transactions and yet "% Processor Time" is well under 100% it is possible to infer sever I/O wait time. Systems that implement solid state disks will typically show high "% Processor Time" numbers.

For example, the following two figures (Figure 3 and Figure 4) show screen shots from Windows Performance Monitor.

Figure 3 below 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.

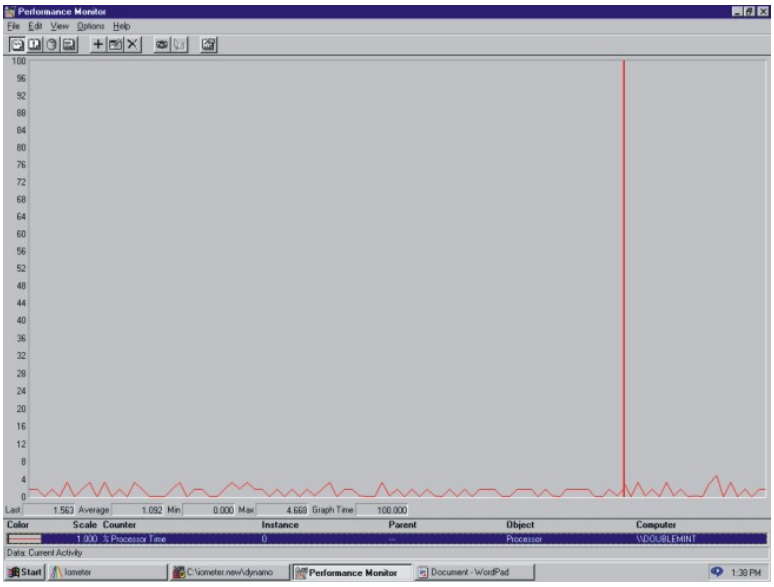

Figure 3: Processor Performance when Writing to Hard Disk

On the following page, **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.

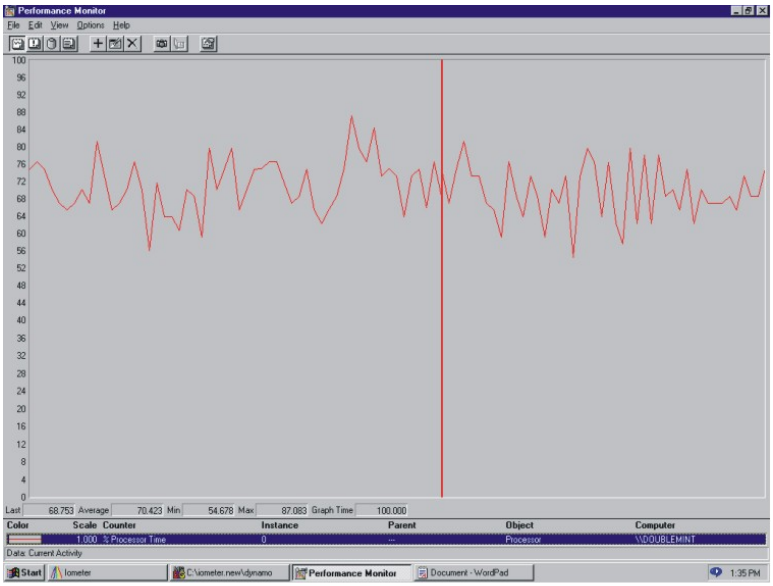

Figure 4: Processor Performance when Writing to Solid State Disk

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. Once 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.

UNIX

For UNIX operating systems, the following commands are useful: `top`, `iostat`, and `sar`. Depending on the command you will receive slightly different output.

The `top` command, when executed on a Solaris system, produces results that have the following format:

```
load averages:  0.09,  0.04,  0.03
16:31:09
66 processes:  65 sleeping, 1 on cpu
CPU states: 69.2% idle, 18.9% user, 11.9% kernel, 0.0% iowait, 0.0% swap
Memory: 128M real, 4976K free, 53M swap in use, 542M swap free
```

The key is that this command provides the "% iowait" for the system. It is important to note, that `top` provides a snapshot of performance. To examine I/O performance over a set period of time, use the SyBase tools described above.

It is also reasonable to look at the `vmstat` command. This command will tell you how frequently your system is paging to virtual memory (disk). If you have frequent paging, it makes sense to consider adding RAM to your system or using a solid state disk for paging. Paging to disk is another way that hard disk drives can introduce bottlenecks into system performance.

Section 6

Components to Move to a 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 Sybase 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-440, for example, can provide 512GB 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 aspect 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.

Tempdb

Up to 25% of read/write activity can be in *tempdb*—the volatile "work area" database where temporary or intermediate data is created and read by users. Query-intensive servers, databases with locked tables, or simple high-traffic databases are only some scenarios that result in an overburdened *tempdb*. Because of, among other things, its tendency to thrash host storage, *tempdb* should reside separately on a solid state disk or on a LUN (Logical Unit) existing on a high-I/O solid state disk.

Quoted from Chapter 35 of the Sybase Performance and Tuning Guide:

"Keep *tempdb* on separate physical disks from your critical application databases at all costs.

Use the fastest disks available. If your platform supports **solid state** devices and your *tempdb* use is a bottleneck for your applications, use those devices."

Note that, since the publication of the Sybase Performance and Tuning Guide, virtually all enterprise platforms support solid state disks.

Like other parts of the server, *tempdb* contains data and a log. Just as regular logs and indices are extremely I/O intensive, *tempdb*'s log sees a disproportionate amount of action. It is unlikely that even intense I/O activity in the log and data could saturate the I/O potential of a quality (>100,000 I/Os per second) solid state disk. Since *tempdb* is, by definition, temporary, it is not vital to separate the log from its related database.

"Hot" Tables

The small portion of frequently accessed ("hot") tables typically account for a large percentage of database I/O activity. 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. Since solid state disks perform in identical fashions regardless of the random or sequential nature of the data, they are well suited for storing "hot" tables.