

# Implementing SQL Server on the Virident FlashMAX Drive

Solution Guide

*January 2012*





## Table of Contents

Executive Summary .....	2
FlashMAX Benefits .....	2
Implementation Requirements .....	3
General High Performance Tunables .....	3
Placement of Databases on the Virident FlashMAX.....	6
Data Reliability .....	12
Data Availability.....	14
Conclusion .....	16
Contacting Virident .....	17



## Executive Summary

The Virident FlashMAX Drive delivers uncompromising performance to Microsoft SQL Server installations. A single drive provides up to 1.4TB of database space in an industry standard PCI Express half-height, half-length form factor. Multiple drives can be combined for larger datasets.

FlashMAX's incredibly high IOPS performance are a great match for OLTP type applications, while its great bandwidth allows it to replace racks of short-stroked disk drives in DSS type workloads. FlashMAX can increase the CPU core utilization on your database server, maximizing the investment in your hardware.

Best of all, FlashMAX is easy to deploy for the database administrator. Utilizing standard optimization techniques entire databases, specific indexes or tables, or even only portions of tables can be accelerated without any user-level application changes.

## FlashMAX Benefits

### Easy to Deploy

- Utilizes Microsoft standard STORPORT architecture
- Standard disk management utilities, both Microsoft and third-party, can be used
- Works in any industry standard, server class system
- Low memory overhead, leaving more room for SQL Server buffers

### Supports application classes with a single infrastructure

- DSS and BI applications with heavy read workloads
- OLTP applications with heavy mixed read and write workloads

### No need to overprovision

- No short-stroking required for highest performance (unlike HDD arrays or other flash)
- Best utilization of storage investment
- Predictable performance means meeting SLAs with less resources

### Enable new business intelligence

- Convert batch processing into interactive applications
- Faster, more informed decision making

## Implementation Requirements

The Virident FlashMAX supports most industry standard server installations, thanks to its small size and adherence to Microsoft and industry standards.

### Hardware

Hardware	A single half-height, half-length PCI Express x8 standard slot in any server-class chassis from 1-U to 4-U rackmount or blade form factor. No additional power is required for the drive to operate.
Processor	Modern Intel Xeon or AMD Opteron dual- or quad-socket server. Hyperthreading has been shown to be helpful in most workloads, but individual database administrators should verify this for their specific environments.
Memory	8GB DRAM or more. The FlashMAX device driver requires approximately one gigabyte for each terabyte of flash present in the system.

### Software

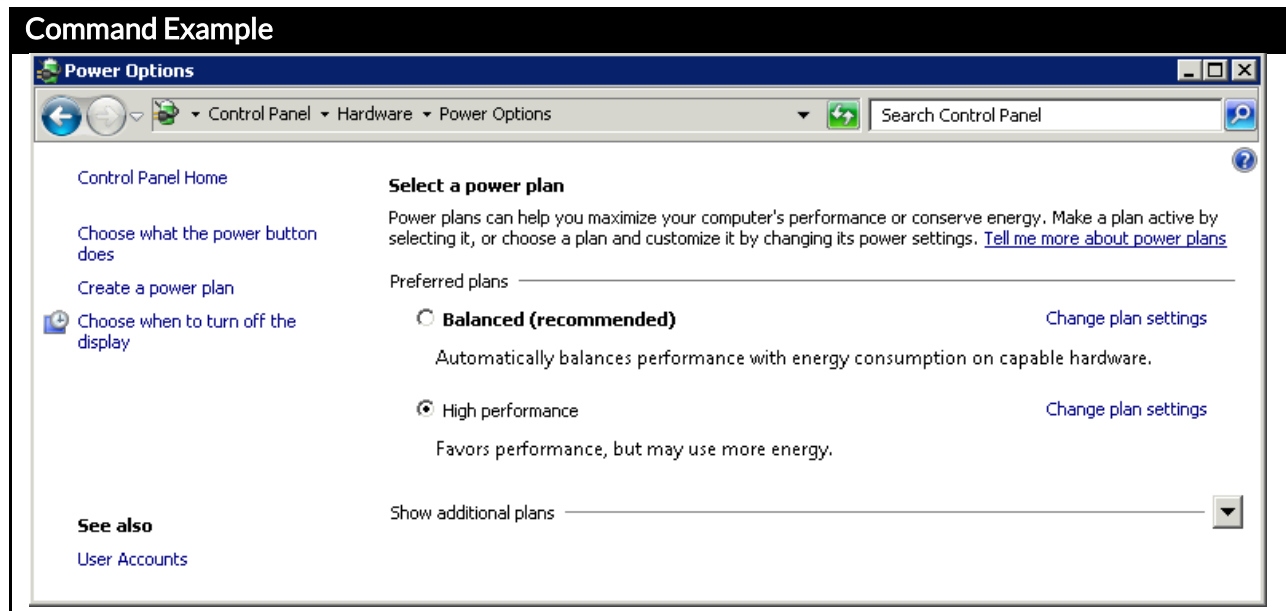
Operating System	Microsoft Windows Server 2008 R2 SP1, 64-bit edition
SQL Server	Microsoft SQL Server 2008 R2, SP1 or later. The latest service pack (in addition to the hotfix mentioned above) is required to ensure that SQL server is able to deliver the highest performance on the Virident FlashMAX's 4K physical sectors.

## General High Performance Tunables

All standard tunables within both Microsoft Windows and Microsoft SQL Server can apply with a Virident FlashMAX.

### Software

At the operating system level, the most important tunable is the performance mode of the system. By default servers are configured in a "Balanced" or "Power Savings" mode (via the Control Panel -> Power Options panel). This mode trades off slower processor clock frequencies and longer latencies for a reduction in server power.



When paired with a low performing disk subsystem this performance tradeoff makes sense because SQL Server is often bottlenecked waiting for data from the slow disk. However, the Virident FlashMAX is able to provide data to SQL Server at such a high rate that this IO waiting time reduces almost to nothing. The idle assumption no longer applies, and as such it makes sense to use the “High Performance” setting to ensure the CPU is always running at the highest speed.

### Microsoft SQL Server Settings

Several general settings and techniques are configurable from within Microsoft SQL Server to get the highest performance possible with the Virident FlashMAX.

Multiple TempDB files should be used no matter where these files are stored, to avoid any bottlenecks on TempDB allocations.

SQL Server should be given as much memory as practicable. With the lower memory requirements of the Virident FlashMAX, it may be possible to increase the memory that is allocated to SQL Server.

Finally, because the FlashMAX performs best at the highest levels of IO parallelism, SQL Server should be set to run with a high setting for worker threads. The appropriate SP\_CONFIGURE settings to achieve this are “max worker threads” (set very high, in the range of 1000 to 3000) and “cost threshold for parallelism” (set to 0 to encourage SQL server to parallelize query execution).

## Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> sqlcmd
1> -- Enable the advanced configuration options
2> sp_configure 'show advanced options', 1;
3> reconfigure with override
4> go
Configuration option 'show advanced options' changed from 1 to 1. Run the RECONFIGURE statement
to install.
1> -- Make SQL server prefer parallelizing queries
2> sp_configure 'cost threshold for parallelism', 0
3> go
Configuration option 'cost threshold for parallelism' changed from 0 to 0. Run the RECONFIGURE
statement to install.
1> sp_configure 'lightweight pooling', 1
2> go
Configuration option 'lightweight pooling' changed from 1 to 1. Run the RECONFIGURE statement to
install.
1> sp_configure 'max worker threads', 3200
2> go
Configuration option 'max worker threads' changed from 3200 to 3200. Run the RECONFIGURE
statement to install.
1> sp_configure 'priority boost', 1
2> go
Configuration option 'priority boost' changed from 1 to 1. Run the RECONFIGURE statement to
install.
1> exit
PS C:\Users\Administrator>
```

## Optimization Note

It is often not necessary to change any SQL Server settings to see a significant performance impact with the FlashMAX drive. It is often worthwhile to start benchmarking without any changes to SQL configuration before individually tweaking specific settings.

## Virident FlashMAX Settings

No special settings are required on the Virident FlashMAX drive to achieve the highest performance. However, to ensure that the drive has been installed properly and the system properly configured the special “test.exe” utility provided by Virident should be run. This will allow performance configuration problems related to the drive, if any, to be isolated and fixed before any SQL server tests are performed.

## Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> U:\TEST.EXE

*** VIRIDENT FLASHMAX PERFORMANCE TEST ***
(C) Copyright 2011 Virident Systems, Inc.
-----
Checking for administrator permissions...OK
Checking the Virident Driver is loaded...OK
Checking power settings...OK
Checking CPU configuration...Intel(R) Xeon(R) CPU E5520 @ 2.27GHz...OK
Checking number of CPU cores...16...OK
Verifying drive letter is a FlashMAX card...OK
Creating 4GB test file...OK
-----
Running Max Read Bandwidth test... (20 seconds)
Max Read Bandwidth measured is 1318MB/s
Running Max Write Bandwidth test... (20 seconds)
Max Write Bandwidth measured is 1001MB/s
Running Max Read IOPS test... (20 seconds)
Max Read IOPS measured is 304K
Running Max Write IOPS test... (20 seconds)
Max Write IOPS measured is 282K
PS C:\Users\Administrator>
```

## Placement of Databases on the Virident FlashMAX

There are two general implementation strategies for using the Virident FlashMAX to accelerate SQL Server: either placing all tables and indexes on the drive, or selecting individual “hot” tables or indexes to place on the drive while still employing a slower and larger storage tier to hold the bulk of the database that is relatively less active.

### All Tables and Indexes on the FlashMAX

The fastest and lowest effort way of implementing the Virident FlashMAX into a database application is to place everything onto the FlashMAX drive. In the case where the database is larger than the capacity of an individual drive, multiple drives (up to eight in a single server) can be installed and striped using standard Windows Disk Manager tools.

If the entire database (tables, indexes, logs) fits, including planned growth, this option is as simple as possible for the database administrator and delivers the greatest performance gain possible. Management and

maintenance is no different in this case from that of any other direct attached database store.

It can reduce up-front investment by completely eliminating hardware such as battery-backed RAID drives and large, short-stroked disk arrays. Ongoing costs, such as power, rack space, and cooling, are also reduced the most in this configuration.

### Optimization Note

One often overlooked benefit, beyond simply increasing transaction throughput with a FlashMAX drive, is stability of performance. Thanks to its superior design, at any workload level the FlashMAX can deliver a steadier, much more predictable level of service than any other HDD or SSD solution. This in turn translates into a reduction in storage capacity overprovisioning with resultant CAPEX and OPEX savings.

## Individually Tiering Tables and Indexes

In some cases it is not practical to place the entire database on Virident FlashMAX volumes. The dataset itself could be too large to fit practically, there could be an existing SAN infrastructure that needs to be retained, or high performance may only be needed for individual tables (or sometimes even just parts of tables). In these cases a more involved effort can be undertaken to place only portions of the database on a Virident FlashMAX while still deriving quantifiable benefits.

### 1. Placing Specific Indexes on FlashMAX

#### Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> SQLCMD
1> ALTER DATABASE test ADD FILEGROUP flashmax_index;
2> GO

1> ALTER DATABASE test ADD FILE ( NAME='flashidx', FILENAME='V:\TEST\flashidx.mdf',
    SIZE=1000MB, FILEGROWTH=1000MB) TO FILEGROUP flashmax_index;
2> GO

1> USE test
2> GO
Changed database context to 'test'.

1> CREATE INDEX birthday_idx ON usertable(birthday) WITH ( FILLFACTOR=100,
    SORT_IN_TEMPDB=ON ) ON flashmax_index;
2> GO
```

### Optimization Note

Indexes may be placed onto the FlashMAX via the use of standard FILEGROUPS in SQL Server. Care should be taken when moving clustered indexes to the FlashMAX, since moving a clustered index to a filegroup will also move the data into that filegroup as well.

Choosing which indexes to place on a FlashMAX drive is a manual process, but the inputs to that process can be generated either by database administrator intuition and application knowledge, or via Microsoft SQL Server Dynamic Management Views: “sys.dm\_db\_index\_usage\_stats” and “sys.dm\_db\_index\_operational\_stats.”

### Command Example

(This example modified from [http://sqlblog.com/blogs/louis\\_davidson/archive/2007/07/22/sys-dm-db-index-usage-stats.aspx](http://sqlblog.com/blogs/louis_davidson/archive/2007/07/22/sys-dm-db-index-usage-stats.aspx))

```
SELECT object_schema_name(indexes.object_id)+'.'+object_name(indexes.object_id) AS objectName,
       indexes.name,
       CASE WHEN is_unique = 1 THEN 'UNIQUE ' ELSE '' END + indexes.type_desc,
       ddius.user_seeks,
       ddius.user_scans,
       ddius.user_lookups,
       ddius.user_updates
FROM sys.indexes
     LEFT OUTER JOIN sys.dm_db_index_usage_stats ddius
       ON indexes.object_id = ddius.object_id
        AND indexes.index_id = ddius.index_id
        AND ddius.database_id = db_id()
WHERE object_schema_name(indexes.object_id)='dbo'
ORDER BY ddius.user_seeks + ddius.user_scans + ddius.user_lookups DESC
```

	objectName	name	(No column name)	user_seeks	user_scans	user_lookups	user_updates
1	dbo.stock	pkey_stock	UNIQUE CLUSTERED	5116831	0	0	1277881
2	dbo.orders	pkey_orders	UNIQUE CLUSTERED	1675353	1	0	256716
3	dbo.item	pkey_item	UNIQUE CLUSTERED	1279163	0	0	0
4	dbo.customer	pkey_customer	UNIQUE CLUSTERED	838626	7	7758	256389
5	dbo.district	pkey_district	UNIQUE CLUSTERED	654066	0	0	256567
6	dbo.warehouse	pkey_warehouse	UNIQUE CLUSTERED	384687	0	0	128120
7	dbo.order_line	pkey_order_line	UNIQUE CLUSTERED	282162	0	0	1406150
8	dbo.new_orders	pkey_new_orders	UNIQUE CLUSTERED	256608	0	0	256426
9	dbo.customer	idx_customer	NONCLUSTERED	169458	0	0	0
10	dbo.orders	idx_orders	NONCLUSTERED	12812	0	0	128447
11	dbo.history	NULL	HEAP	0	0	0	128120

## 2. Placing Portions of Tables on FlashMAX

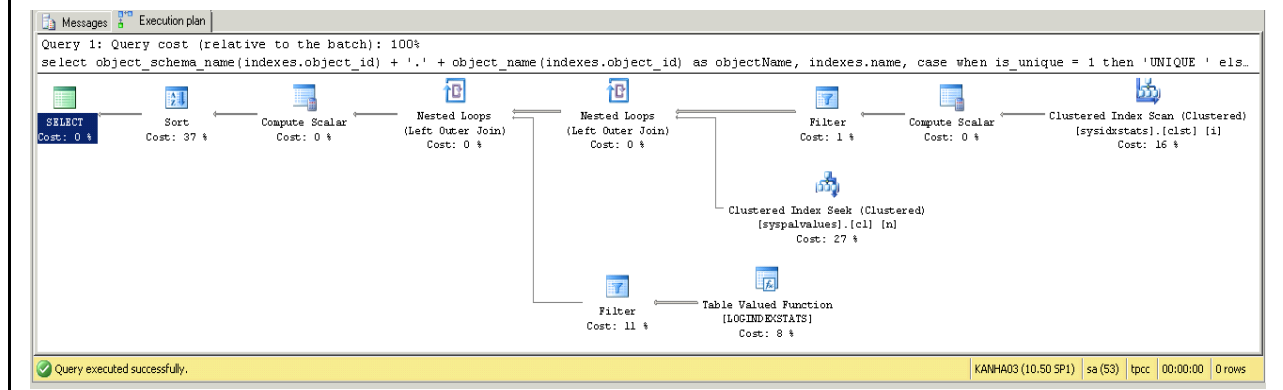
Even more finely grained tuning of database placement is possible via the use of partitioned tables. By partitioning a table (often done by date or sequence number), historic data can be manually tiered to slower rotating media while recent data can be stored on FlashMAX locally for frequent access and updates.

This method requires significant, ongoing participation of the database administrator as the PARTITION function and FILEGROUPS need to change tiered over time. For example, a partitioning scheme based on order date would need to be updated monthly or yearly to move the FILEGROUPS of older parts of the table to slower SAN storage.

### Optimization Note

Partitioning tables and placing frequently accessed partitions on the Virident FlashMAX can significantly speed up application performance, but be sure that queries accessing these tables are not also accessing indexes, other tables, or other partitions of the same table, that are not stored on the FlashMAX.

A graphical way of verifying this is to select “Display Estimated Execution Plan” for a representative set of queries in the Management Studio and examining the resultant plans for access to indexes or table scan against portions of the database not on FlashMAX.



## Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> SQLCMD
1> -- Make the two filegroups
2> ALTER DATABASE test ADD FILEGROUP data_2010
3> GO

1> ALTER DATABASE test ADD FILEGROUP data_2011
2> GO

1> -- Historic FG lives on the SAN...
2> ALTER DATABASE test ADD FILE (NAME = 'data_2010', FILENAME = 'S:\data\data_2010.mdf',
    SIZE = 1000000MB, FILEGROWTH = 1000MB) TO FILEGROUP data_2010
3> GO

1> -- Active FG lives on Virident...
2> ALTER DATABASE test ADD FILE (NAME = 'data_2011', FILENAME = 'V:\data\data_2011.mdf',
    SIZE = 300000MB, FILEGROWTH = 1000MB) TO FILEGROUP data_2011
3> GO

1> -- Define the partition function, only have 1 range point as anything past
2> CREATE PARTITION FUNCTION part_yearly(DATETIME) AS RANGE LEFT FOR VALUES
    ( '20111231 23:59:59.999' );
3> GO

1> -- If it's prior to first cutoff go to 2010, otherwise 2011
2> CREATE PARTITION SCHEME partsch_yearly AS PARTITION part_yearly TO ( data_2010, data_2011 );
3> GO

1> -- Make a new partitioned table and populate it...
2> CREATE TABLE orders_part ( ... ) on partsch_yearly;
3> GO

1> -- Go get some coffee and a good book...
2> INSERT orders_part SELECT * FROM orders;
3> GO
```

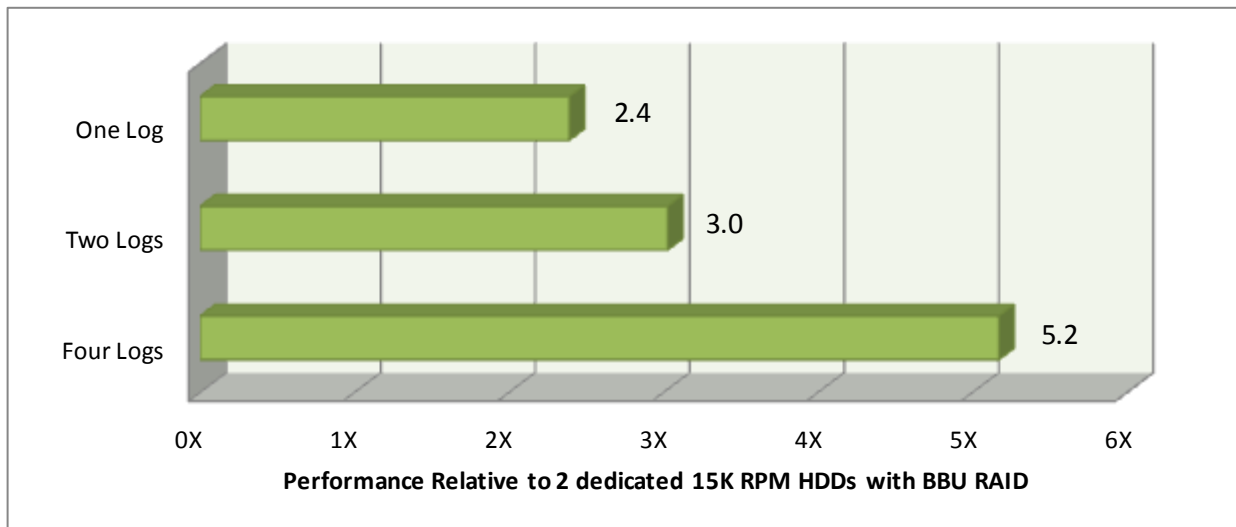
## Transaction logs on FlashMAX

Transaction logs are written in small blocks, sequentially. The size of the transaction log varies widely with the database size, as does the activity logged therein. For OLTP workloads, where many updates and inserts are performed, the transaction log can often become a limiting factor. Conversely, for DSS workloads which have much less frequent updates, the transaction log may not be particularly performance sensitive.

On rotating media, transaction logs are often stored on their own RAID mirrored drive set. This is because if they were stored on the main table volume set, the random access nature of table and index access would

destroy their sequential write access pattern. Since HDDs perform sequential write accesses significantly faster than random, the cost (in terms of money, power, and space) of the additional log drive pair is outweighed by the performance improvement received.

No such design decision needs to be made with the Virident FlashMAX. Log and database can successfully coexist on the same drive without any performance impact. Multiple database logs can also be stored on a single FlashMAX as well, without any of the negative performance implications such a configuration would have on rotating media. The FlashMAX Drive provides enough raw IO horsepower to handle sequential transaction logging without impacting the main database random read/write access, and vice versa.



**Optimization Note**

Only a single log file per database is written to by SQL Server. Specifying multiple log files per database on the Virident FlashMAX, or any other storage, will not improve performance.

**TempDB on FlashMAX**

SQL Server uses tempdb to store intermediate query results as well as during index creation and updates. Writes to it are often large block sequential in nature, while reads are commonly random in nature. It also has a high data turnover, since the data inside each temp table is only valid for a single query or index generation step. This is an ideal use for the very high write bandwidth and random IOPS.

## Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> MKDIR V:\tempdb
PS C:\Users\Administrator> SQLCMD
1> -- Move TEMPDB file to Virident, and add additional 7 for 8 total files
2> ALTER DATABASE tempdb MODIFY FILE (NAME=tempdev , FILENAME='V:\tempdb\tdb0.mdf' , SIZE=500MB);
3> ALTER DATABASE tempdb ADD FILE (NAME=tempdev1 , FILENAME='V:\tempdb\tdb1.mdf' , SIZE=500MB);
4> ALTER DATABASE tempdb ADD FILE (NAME=tempdev2 , FILENAME='V:\tempdb\tdb2.mdf' , SIZE=500MB);
5> ALTER DATABASE tempdb ADD FILE (NAME=tempdev3 , FILENAME='V:\tempdb\tdb3.mdf' , SIZE=500MB);
6> ALTER DATABASE tempdb ADD FILE (NAME=tempdev4 , FILENAME='V:\tempdb\tdb4.mdf' , SIZE=500MB);
7> ALTER DATABASE tempdb ADD FILE (NAME=tempdev5 , FILENAME='V:\tempdb\tdb5.mdf' , SIZE=500MB);
8> ALTER DATABASE tempdb ADD FILE (NAME=tempdev6 , FILENAME='V:\tempdb\tdb6.mdf' , SIZE=500MB);
9> ALTER DATABASE tempdb ADD FILE (NAME=tempdev7 , FILENAME='V:\tempdb\tdb7.mdf' , SIZE=500MB);
10>
11> -- Don't forget to move the log, too
12> ALTER DATABASE tempdb MODIFY FILE (NAME=templog , FILENAME='V:\tempdb\tdb.ldf' , SIZE=100MB);
13>
14> GO
The file "tempdev" has been modified in the system catalog. The new path will be used the next
time the database is started.
The file "templog" has been modified in the system catalog. The new path will be used the next
time the database is started.
1> EXIT
PS C:\Users\Administrator> NET STOP MSSQLSERVER
The SQL Server (MSSQLSERVER) service is stopping.
The SQL Server (MSSQLSERVER) service was stopped successfully.

PS C:\Users\Administrator> NET START MSSQLSERVER
The SQL Server (MSSQLSERVER) service is starting.
The SQL Server (MSSQLSERVER) service was started successfully.
```

## Optimization Note

Standard best practices for tempdb should be followed. Ensure that tempdb is specified with as many separate files as there are cores in your server to minimize any tempdb allocation bottlenecks. This is especially important for highly utilized SQL Server installations to allow multiple queries to utilize the tempdb in parallel. Also, be sure to create each tempdb file with the same size to avoid known issues with SQL Server's tempdb autogrowth.

## Data Reliability

As with all direct attach storage, proper care should be taken of databases stored on the Virident FlashMAX drive. As an easy start, any existing data protection and availability plans can be implemented identically on the FlashMAX drive.

## Flash Management

FlashMAX implements the highest level of onboard flash wear leveling and error detection and correction available. It actively monitors flash performance and can move data behind the scenes, transparently to SQL Server, to ensure continual data availability.

## Flash-Aware RAID-5 per drive

On top of the standard flash management and data protection, additional protection against hardware failure on a single drive can be implemented using FlashMAX 's built-in hardware RAID-5 like mode. There is no penalty in performance on reads in this mode. Write performance is reduced slightly, as is raw drive capacity. This mode provides hardware assisted reliability and can survive the complete loss of multiple modules without data loss.

### Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> cd 'C:\Program Files\Virident\FashMAX\Utils'
PS C:\Program Files\Virident\FashMAX\Utils> vgc-config -d vgca -5 enabled
Run vgc-commit.exe to apply the new configuration changes. [cmd: vgc-commit.exe]
Warning: Applying new configuration changes may erase existing data on 'vgca'. Please backup the
data on 'vgca'
PS C:\Program Files\Virident\FashMAX\Utils> vgc-commit
Pending Configuration:
      vgca0          mode=balanced      sector-size=512      raid=enabled

Warning: Applying new configuration changes may erase existing data.
Do you want to continue? (y/n) : y
Configuring FlashMAX card a...
Preparing device vgca0
Device details GW ID 0 part ID 0
      vgca0 : prepared successfully
Device configuration completed successfully
```

### Optimization Note

Configure RAID-5 on the Virident FlashMAX before loading your database as the conversion between non-RAID and RAID-5 is destructive to data on the drive.

## RAID-1 across multiple drives

Multiple drives can be striped in a mirrored (RAID-1) volume using the standard Disk Manager. No special

setup or configuration is required.

### Command Example

```
Windows PowerShell
Copyright (C) 2009 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> DISKPART

Microsoft DiskPart version 6.1.7601
Copyright (C) 1999-2008 Microsoft Corporation.
On computer: KANHA03

DISKPART> SELECT DISK=3
Disk 3 is now the selected disk.
DISKPART> CONVERT DYNAMIC
DiskPart successfully converted the selected disk to dynamic format.
DISKPART> SELECT DISK=4
Disk 4 is now the selected disk.
DISKPART> CONVERT DYNAMIC
DiskPart successfully converted the selected disk to dynamic format.
DISKPART> CREATE VOLUME MIRROR DISK=3,4
DiskPart successfully created the volume.
DISKPART> FORMAT FS=NTFS LABEL=Virident-R1 QUICK
    100 percent completed
DiskPart successfully formatted the volume.
DISKPART> ASSIGN LETTER=V
DiskPart successfully assigned the drive letter or mount point.
DISKPART> EXIT
Leaving DiskPart...
```

### Optimization Note

RAID is not a backup strategy. A software error or unexpected “DROP DATABASE” can destroy all user data protected by any of these methods.

## Data Availability

Microsoft SQL Server provides both mirroring and log shipping modes to ensure high data availability in cases of server failure. Individual application availability requirements will dictate which mode is appropriate, but both modes are supported transparently on the Virident FlashMAX drive.

### SQL Server Mirroring

Microsoft SQL Server provides mirroring capabilities to allow a primary and secondary database server to stay synchronized through a network connection. This mode supports asynchronous and synchronous replication. The data consistency guarantees for these two modes vary somewhat and the proper mode depends on your specific availability requirements. Asynchronous mode gives the highest performance replication for

frequently updated databases.

Whichever mode of replication is utilized, the lowest latency network interconnect possible should be used. While the 10x improvement in bandwidth available when moving from 1G Ethernet to 10G Ethernet may not be useful, the reduction in latency of smaller messages will impact replication traffic and responsiveness.

#### **Optimization Note**

Servers at both ends of a mirroring operation should be similarly designed. If the database at the primary is placed on a Virident FlashMAX, it should be on a FlashMAX on the secondary mirror as well. Otherwise the performance of the system can become limited to that of the lowest performing server (not only in the case of failover but in steady state due to the replication).

### **Transaction Log Shipping**

Transaction log shipping allows for a master database to periodically send updated transaction log copies to one or many secondary servers over a shared storage. Logs are backed-up at the primary server, then copied to secondary servers and finally restored there to present a delayed view of the primary database at the secondary servers.

There is little that changes when implementing this replication on servers utilizing the Virident FlashMAX. The additional bandwidth available for backing up or restoring the transaction logs, when stored on the FlashMAX drive, can slightly reduce the load on any server during this operation.

#### **Optimization Note**

Transaction log shipping to secondary servers is often used for reporting or analysis work, so as to avoid impacting the main database server. By implementing a Virident FlashMAX on that main server, however, the need for these secondary servers can be greatly diminished or reduced entirely. The limiting performance factor in most database servers is the storage subsystem. The FlashMAX has enough IO performance and flexibility to support on-line analysis on live databases without the need for secondary copies.

### **Backup Strategies**

A tested and regularly executed database backup strategy can provide data availability in the face of catastrophic failures. One concern about running backups under traditional storage, however, is the amount of IO traffic that such an operation can cause. The additional IO load required to read the entire database out using either SQL Server's internal BACKUP command, or a third party tool, can cause average IO latency to skyrocket on traditional storage solutions. This increase in latency can cause significant application SLA violations and diminished user responsiveness. In some cases, applications can even result in a "dogpile" effect

where applications will try to increase the number of connections to the slowed-down server to compensate, resulting in an even worse response time.

To avoid these issues, backups are often scheduled, and often infrequently, at “low-usage times.” The trouble with this is that such low usage times may not really exist, or they may not always occur at the same time, especially with a globally connected workforce.

The Virident FlashMAX can effectively alleviate this issue thanks to its uncompromising performance. Even under high SQL Server loads, it has enough additional IO resources to handle the large streaming reads required for backup, without impacting main application performance. Backups can be scheduled more frequently with little concern about their effect on users, giving database administrators more freedom to do them when business needs and convenience dictate.

## Conclusion

Adding the Virident FlashMAX is a simple, cost effective way of architecting flash storage into SQL Server deployments. By working in any standard server, it provides an upgrade path for existing servers while at the same time not limiting server selection for future acquisitions.

Multiple application classes can be supported on a single Virident-accelerated infrastructure. Thanks to FlashMAX’s high sequential bandwidth decision support and business intelligence workloads can perform at their fullest, while OLTP type applications can immediately benefit from the incredibly high, sustained random read and write IOPs of FlashMAX drives.

Most standard SQL Server best practices are appropriate when adding the Virident FlashMAX to a database application, with certain exceptions. Most importantly, with the FlashMAX there is no need to “short stroke” the database store on the adapter thanks to its uncompromising performance under full utilization.

Finally, the combination of the Virident FlashMAX’s raw power and database administrators’ best practices can enable new applications or simplify old ones. SQL jobs that used to be run in batch mode in the evenings due to their effect on the IO subsystem can now be run interactively, allowing the information stored in the database to be converted into real business knowledge.



Virident Systems, Inc., 500 Yosemite Drive, Suite 108 | Milpitas | CA 95035

P 408.503.0100 | F 408.263.7760 | [www.virident.com](http://www.virident.com)

© 2012 Virident Systems, Inc. All rights reserved.

Virident, [virident.com](http://virident.com), the Virident logo, and FlashMAX are trademarks or registered trademarks of Virident Systems, Inc. in the United States, other countries, or both. If these and other Virident trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by Virident at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries.