High Performance SQL Server

SQL Server Hardware Choices Made Easy

Glenn Berry

ISBN: 978-1-906434-72-4

SQL Server Hardware Choices Made Easy

Glenn Berry

First published by Simple Talk Publishing, 2011

ISBN: 978-1-906434-72-4

Edited by Tony Davis
Technical Review by Denny Cherry
Cover Image by Andy Martin
Typeset & Designed by Matthew Tye & Gower Associates

Table of Contents

About the Author

Glenn Berry is a Database Architect at Avalara in Denver, Colorado.
He is a SQL Server MVP, and he has a whole collection of Microsoft certifications, including MCITP, MCDBA, MCSE, MCSD, MCAD, and MCTS, which proves that he likes to take tests. His expertise includes DMVs, high availability, hardware selection, full text search, and SQL Azure. He is also an Adjunct Faculty member at University College – University of Denver, where has been teaching since 2000. He has completed the Master Teacher Program at Denver University – University College. He is the author of two chapters in the book, *SQL Server MVP Deep Dives*, and blogs regularly at HTTP://SQLSERVERPERFORMANCE.WORDPRESS.COM. Glenn is active on Twitter, where his handle is **@GlennAlanBerry**.

Introduction

Relational databases place heavy demands on their underlying hardware. Many databases are mission-critical resources for multiple applications, where performance bottlenecks are immediately noticeable and often very costly to the business. Despite this, many database administrators are not very knowledgeable about server hardware.

The idea behind this eBook is to provide a quick "heads up" to the issues that I have found to be most important when choosing and configuring hardware for use with SQL Server, by providing a series of specific tips or pieces of advice. In 30 short tips, I provide advice on how to:

- choose a processor, and related hardware, for database servers, and the factors that will influence that choice
- choose a processor for test and development systems
- provision and configure your disk subsystem.

Of course, in 30 tips, spanning about 60 pages, I can't cover *everything* that may impact your choices, and I also can't provide in-depth "tutorial-style" material. For fuller and finer details on all the topics covered here, plus coverage of how to install and configure the operating system and SQL Server on your chosen hardware, for maximum performance and reliability, please check out my book, SQL Server Hardware.

I would like to stress that hardware is a fast- and ever-changing field. In several places I specifically quote the date at which a piece of advice was valid (time of writing, July 2011). My plan is to release updated versions of this eBook as new hardware technology emerges and my "best practice" advice and recommendations adapt accordingly.

Server processors and related hardware

This section will offer twenty tips for evaluating and selecting a processor and associated hardware for a database server. It will explain the options available, offer advice on how to choose the most appropriate choice of processor and chipset for SQL Server given the nature of the workload, and discuss other factors that will influence your choices.

1 Know your workload type, Part 1

The type of workload that must be supported will have a huge impact on your choice of hardware, including processors, memory, and the disk I/O subsystem, as well as on sizing and configuration of that hardware. It will also affect your database design and indexing decisions, as well as your maintenance strategy.

There are two primary relational workload types that SQL Server commonly has to deal with: Online Transaction Processing (OLTP) and Decision Support System (DSS)/Data Warehouse (DW).

OLTP workloads are characterized by numerous short transactions, and a relatively high level of data volatility. There is usually much higher write activity in an OLTP workload than in a DSS workload and most OLTP systems generate more input/output operations per second (IOPS) than an equivalent-sized DSS system.

When choosing a processor for an OLTP-style workload, it's important to consider that the short, fast transactions that will (or at least should!) comprise such a workload won't benefit from parallelization (where processing of a given transaction is split across multiple threads and cores), and so the overriding factor in your choice of CPU (central processing unit) will be its single-threaded performance.

A DSS or DW system usually has longer-running queries than a similar size OLTP system, and the data is usually more static, with much higher read activity than write activity. In such a system, it is much more important to be able to be able to process a large of amount of data quickly, than it is to support a high number of I/O operations per second.

When choosing a processor for a DSS-style workload, it's important to consider that the longer, more complex transactions that characterize such a workload will benefit greatly from parallelization, so the overriding factor in your choice of CPU becomes the **multi-threaded performance**, and the availability of a high number of real, physical cores.

You really should try to determine what type of workload your server will be supporting as soon as possible in the system design process. You should also strive to segregate OLTP and DSS/DW workloads onto different servers and I/O subsystems whenever you can. Of course, this is not always possible, as some workloads are mixed in nature, with characteristics of both types of workloads.

One way of determining the size and nature of your workload is to retrieve the read/write ratio for your database files. The higher the proportion of writes, the more "OLTP-like" is your workload. The query shown in Listing 1, which uses the `sys.dm_io_virtual_file_stats` Dynamic management View (DMV), can be run on an existing system to help characterize the I/O workload for the current database. This query will show the read/write percentage, by file, for the current database, both in the number of reads and writes and in the number of bytes read and written.

```
-- I/O Statistics by file for the current database
SELECT  DB_NAME(DB_ID()) AS [Database Name] ,
        [file_id] ,
        num_of_reads ,
        num_of_writes ,
        num_of_bytes_read ,
        num_of_bytes_written ,
        CAST(100. * num_of_reads / ( num_of_reads + num_of_writes )
                         AS DECIMAL(10,1)) AS [# Reads Pct] ,
        CAST(100. * num_of_writes / ( num_of_reads + num_of_writes )
                         AS DECIMAL(10,1)) AS [# Write Pct] ,
        CAST(100. * num_of_bytes_read / ( num_of_bytes_read
                                   + num_of_bytes_written )
                         AS DECIMAL(10,1)) AS [Read Bytes Pct] ,
        CAST(100. * num_of_bytes_written / ( num_of_bytes_read
                                   + num_of_bytes_written )
                         AS DECIMAL(10,1)) AS [Written Bytes Pct]
FROM    sys.dm_io_virtual_file_stats(DB_ID(), NULL) ;
```

Listing 1-1: Finding the read/write ratio, by file, for a given database.

A more thorough examination of the read/write ratio, using the DMVs, is given in Chapter 6 of the book Performance Tuning with SQL Server Dynamic Management Views, by Louis Davidson and Tim Ford.

2 Don't economize on the CPU

The heart of any database server is the CPU, which executes instructions and temporarily stores small amounts of data in its internal data caches. My basic premise is that, for a database server, you want the very best processor available for each physical socket in the server (SQL Server processor licenses are priced per socket and are relatively expensive – see Tip 3).

Unlike a web server, you don't want to pick a processor for a database server that is one or two models down from the most expensive, top-of-the-line model. You will most likely be stuck with whatever processor you choose for the life of the server, since it rarely makes economic sense to upgrade the processors in an existing server.

Remember, when provisioning CPU, that your server not only has to cope smoothly with the normal workload, but also deal with inevitable spikes in CPU usage, for example during:

- database backups
- index rebuilds and reorganization (the actual process of rebuilding or reorganizing an index is CPU-intensive)
- periods of concurrent, CPU-intensive queries, especially Full Text Search queries.

Get the very best processor that your budget will allow, over-provisioning if possible, and using spare CPU capacity (and memory) to remove load from the disk I/O subsystem.

In particular, the use of excess processor capacity to compress and decompress data as it is written to and read from the disk subsystem has become much more prevalent. SQL Server 2008 and 2008 R2 provide both **data compression** (Page or Row) as well as SQL Server native **backup compression**. Both of these features can be very effective in reducing stress on your I/O subsystem, since data is compressed and decompressed by the CPU before it is written to, or read from, the disk subsystem. The pay-off is extra CPU pressure but, in my experience, modern, multi-core processors can shrug off this extra work.

Trading CPU utilization for I/O utilization is usually a net win; high-performance CPUs are much more affordable than additional I/O capacity, so it makes financial sense to get the best CPU available for a given server model.

3 More cores, fewer sockets…

In the days of single-core processors, if you wanted multiple threads of execution, your only option was to add another physical socket – this is the "slot" on a motherboard where a physical processor is installed.

However, with the advent of multi-core processors and hyper-threading (see Tip 4 for further advice on hyper-threading), we now have available:

- **multiple physical cores** – in a multi-core processor, each physical processor unit contains multiple individual processing cores, enabling parallel processing of tasks
- **multiple logical cores** – with hyper-threading technology, a physical core can be split into two logical cores ("logical processors"), again facilitating parallel execution of tasks.

The critical point to bear in mind here is that, unlike for other database products, such as Oracle or DB2, SQL Server licensing (for "bare metal" servers) is concerned only with **physical processor sockets**, not physical cores, or logical cores. This means that the industry trend toward multiple core processors with increasing numbers of cores will continue to benefit SQL Server installations, since you will get more processing power without having to pay for additional processor licenses. Having additional processor cores helps increase your server workload capacity for OLTP workloads, while it increases both your workload capacity *and* performance for DSS/DW workloads.

Knowing this, you should always buy processors with as many cores as possible (regardless of your workload type) in order to maximize your CPU performance per processor license. For example, it would make sense, in most cases, to have one quad-core processor instead of two dual-core processors.

Likewise, if you're running an old 4-socket machine, you should consider upgrading to a modern 2-socket server. The additional multi-core processing power (as well as memory density and I/O capacity, represented by additional PCI-E expansion slots) of modern 2-socket servers means that you may well be able to support your workload more than adequately. Furthermore, even with SQL Server 2008 R2 Standard Edition licenses, the cost-saving of two processor licenses would pay for the new machine, exclusive of the I/O subsystem! I was recently able to successfully consolidate the workload from four older, (2007 vintage) four-socket database servers into a single, new 2-socket server, saving about $90K in hardware costs and $350K in SQL Server license costs.

Given the rapid advancement in processor architecture, the sky is more or less the limit in terms of available processing power. In mid-2011, AMD will release the "Bulldozer" processor, which will have 16 physical cores per physical processor, while Intel has recently released the Xeon E7 "Westmere-EX" family, which has ten physical cores, plus second-generation hyper-threading, which means that you will have a total of 20 logical cores visible to Windows per physical processor.

4 Don't dismiss hyper-threading (Intel processors)

Early implementations of Intel's hyper-threading, on the Pentium 4 Netburst architecture, did not work well with SQL Server workloads and it was pretty common for database administrators to disable it. However, the newer (second-generation) implementation of hyper-threading (as used in the Intel Nehalem or Westmere-based Xeon 5500, 5600, 6500, and 7500 families) seems to work much better for many server workloads, and especially with OLTP workloads. It is also interesting that every TPC-E benchmark submission that I have seen on Nehalem-EP, Nehalem-EX, and Westmere-EP based platforms has had hyper-threading enabled (see Tip 16 for more on the TPC-E benchmark).

My "rule-of-thumb" advice for use of hyper-threading is as follows:

- enable it for workloads with a high OLTP character
- disable it for OLAP/DW workloads.

This may initially seem counter-intuitive, so let me explain in a little more detail. For an OLAP workload, query execution performance *will* generally benefit from allowing the query optimizer to "parallelize" individual queries. This is the process where it breaks down a query, spreads its execution across multiple CPUs and threads, and then reassembles the result. For an ideal OLAP workload, it would be typical to leave the Max

Degree of Parallelism (`MAXDOP`) setting at its default value of zero (unbounded), allowing the optimizer to spread the query execution across as many cores as are available.

Unfortunately, it turns out that this only works well when you have a high number of true **physical cores** (such as offered by some of the newer AMD Opteron 6100 series "Magny-Cours" processors). Complex, long-running queries simply *do not* run as well on logical cores, and so enabling hyper-threading tends to have a detrimental impact on performance.

In contrast, an OLTP workload is, or should be, characterized by a large number of short transactions, which the optimizer will not parallelize as there would be no performance benefit; in fact, for an ideal OLTP workload, it would be common to restrict any single query to using one processor core, by setting `MAXDOP` to 1. However, this doesn't mean that OLTP workloads won't benefit from hyper-threading! The performance of a typical, short OLTP query is not significantly affected by running on a logical core, as opposed to physical core, so by enabling hyper-threading for an OLTP workload we can benefit from "parallelization" in the sense that more cores are available to process more separate queries in a given time. This improves capacity and scalability, without negatively impacting the performance of individual queries.

Of course, no real workload is perfectly OLTP, or perfectly OLAP, and the only way to know the impact of hyper-threading is to run some tests with your workload. Even if you are still using the older, first-generation implementation of hyper-threading, I think it is a mistake to always disable it, without first investigating its impact in your test environment, under your workload.

Note that Windows and SQL Server cannot tell the difference between hyper-threaded logical processors and true dual-core or multi-core processors. In the DMV query shown in Listing 4-1 (requires `VIEW SERVER STATE` permission) the `hyperthread_ratio` column treats both multi-core and hyper-threading the same (which they are, as far as the logical processor count goes), so it cannot tell the difference between a quad-core processor and a dual-core processor with hyper-threading enabled. In each case, these queries would report a `hyperthread_ratio` of 4.

```
-- Hardware information from SQL Server 2005/2008/2008R2
-- (Cannot distinguish between hyper-threading and multi-core)
SELECT  cpu_count AS [Logical CPU Count] ,
        hyperthread_ratio AS [Hyperthread Ratio] ,
        cpu_count / hyperthread_ratio AS [Physical CPU Count] ,
        physical_memory_in_bytes / 1048576 AS [Physical Memory (MB)]
FROM    sys.dm_os_sys_info ;
```

Listing 4-1: CPU configuration.

5 Don't focus just on speed: cache size is very important

Most people, when evaluating CPUs, focus first on the **rated clock speed**, measured in cycles/second (Gigahertz, GHz). While this is a very important factor, it's equally important to consider the **cache size**, in megabytes (MB).

Evaluating processors

When comparing the expected performance of processors from different manufacturers, or different processor families, you must also consider the overall architecture and technology used in the processors under comparison (see Tip 6, Tip 7, Tip 8, Tip 9, and Tip 10 for further advice).

All Intel-compatible CPUs (which includes AMD processors) have multiple levels of cache. The Level 1 (L1) cache has the lowest latency (i.e. the shortest delays associated with accessing the data) but the least amount of storage space, and the available storage space (and associated latency) increases sequentially with the L2 and L3 caches. In many cases, the L3 cache is shared among multiple processor cores. In older processors, the L3 cache was sometimes external to the processor itself, located on the motherboard.

Whenever a processor has to execute instructions or process data, it searches for the data that it needs to complete the request in the following order:

1. internal registers on the CPU
2. L1 cache (which could contain instructions or data)
3. L2 cache
4. L3 cache
5. main memory (RAM) on the server
6. any cache that may exist in the disk subsystem
7. actual disk subsystem.

The further the processor has to follow this data retrieval hierarchy, the longer it takes to satisfy the request, which is one reason why cache sizes on processors have gotten much larger in recent years. Table 5-1 shows the typical sizes and latency ranges for these main levels in the hierarchy.

L1 Cache	L2 Cache	L3 Cache	Main Memory	Disk
32 KB	256 KB	12 MB	72 GB	Terabyte
2 ns	4 ns	6 ns	50 ns	20 ms

Table 5-1: Data retrieval hierarchy for a modern system.

For example, on a newer server using a 45 nm Intel Nehalem-EP processor, you might see an L1 cache latency of around 2 nanoseconds (ns), L2 cache latency of 4 ns, L3 cache latency of 6 ns, and main memory latency of 50 ns. When using traditional magnetic hard drives, going out to the disk subsystem will have an average latency measured in milliseconds. A flash-based storage product (like a Fusion-io card) would have an average latency of around 25 microseconds. A nanosecond is a billionth of a second; a microsecond is a millionth of a second, while a millisecond is a thousandth of a second. Hopefully, this makes it obvious why it is so important for system performance that the data is located as short a distance down the chain as possible.

The performance of SQL Server, like most other relational database engines, has a huge dependency on the size of the L2 and L3 caches. Most processor families will offer processor models with a range of different L2 and L3 cache sizes, with the cheaper processors having smaller caches; where possible, I advise you to favor processors with larger L2 and L3 caches. Given the business importance of many SQL Server workloads, economizing on the L2 and L3 cache sizes is not usually a good choice.

6 Use the High Performance power option

By default, the power consumption characteristics of a modern server can be controlled and moderated, to some extent, by the processor (BIOS settings) or the operating system (Windows Power Plan settings). The idea is that the clock speed of your processors is reduced, in order to use less electrical power when the processor is not under a heavy load.

Monitoring clock speed with CPU-Z

You can watch the effect of power management settings in near real-time with a tool like CPU-Z, which displays the current clock speed of Core 0. See Tip 14 *for further advice.*

On the surface, this seems like a good idea, since electrical power costs can be pretty significant in a datacenter. Throttling back a processor can save some electricity and reduce your heat output, which can reduce your cooling costs in a datacenter. Unfortunately, as reported by Brent Ozar and Paul Randal, with some processors and with some types of SQL Server workloads (particularly OLTP workloads), you will pay a pretty significant performance price for those electrical power savings. My Geekbench tests, initially reported on my BLOG and with more detailed investigation in my BOOK, suggest that operating in power-saving mode can reduce processor performance for OLTP workloads by up to 20–25%.

The performance problem stems from the fact that some processors don't seem to react fast enough to an increase in load by quickly increasing the clock speed, especially in response to very short OLTP queries that often execute in a few milliseconds.

This problem is particularly prevalent with Intel Xeon 5500, 5600 and 7500 series processors (which are the Nehalem and Westmere families) and with AMD Opteron 6100 series (Magny-Cours family). Many older processors don't have any power management features, and some slightly older processors (such as the Intel Xeon 5300 and 5400 series) seem to handle power management a bit better.

As hinted earlier, there are two types of power management of which you should be aware.

- **Software based** – the operating system (Windows Server 2008 and above) is in charge of power management using one of the standard **Windows Power Plans**, or a customized version of one of those plans. Options are:
  - *balanced* – the default, allowing the OS to moderate power consumption according to workload; turbo boost technology (see Tip 7) will not be used
  - *power saver* – minimizing power consumption is a priority
  - *high performance* – maximizing performance at all times.
- **Hardware based** – the main system BIOS (firmware) of a server is set to allow the processors to manage their own power states, based on workload. Options are:
  - *none* – your processors will run at full rated speed at all times, regardless of the processor load or temperature, and regardless of the Windows Power Plan setting

- *OS control* – Windows will dictate and control the power management for the server, using one of the Windows Power Plans
- *hardware* – the processor itself controls its power usage based on its load and temperature; I think that is the least desirable setting, since it takes control out of the hands of both the DBA and of Windows.

My current advice is to first check your Windows Power Plan setting, and make sure you are using the **High Performance** power plan. This can be changed dynamically without a restart. Next, run CPU-Z, and make sure your processor is running at, or above, its rated speed. If you are using the High Performance Power Plan and the CPU is still running at less than its rated speed, it means that your hardware power management settings are overriding the Windows setting. During your next maintenance window, go into your BIOS settings and either disable power management or set it to **OS Control** (which I prefer). This will require a server restart.

I give this advice reluctantly, since power conversation is a very laudable goal, and it is well worth keeping track of advances in power-saving technology as new processors emerge. Promisingly, I have noticed that the Intel's Sandy Bridge processors seem to handle power management very well, and don't show as much of a noticeable performance decrease when power management is enabled (at least with the desktop Core i7 2500K and 2600K that I have tested – see Tip 22).

7 Use "turbo boost" technology

Both Intel and, more recently, AMD have started to make technology available in their modern processors to enable "intelligent overclocking" of individual cores of a physical processor.

We'll start with Intel's second-generation Turbo Boost Technology 2.0 which, in response to the operating system requesting the highest processor performance state (P0), allows individual cores of the processor to have their core speed increased temporarily by varying amounts, based on how many cores are busy, the total power usage of the processor, the ambient temperature of the system, and so on.

This core speed increase is very helpful for OLTP workloads, especially if they are processor-dependent (CPU-bound). The potential speed increase varies according to the model of the processor but, for the newest Sandy Bridge processors, Turbo Boost 2.0 can typically result in a 400 MHz speed boost over the base clock speed.

I really don't see any possible downside to this. The base clock speed on Intel processors is rated quite conservatively, with lots of thermal headroom. Having various algorithms in place to temporarily increase the clock speed of individual cores poses very little risk of causing any problems, such as overheating or shortening the life of the processor.

In a similar vein, AMD has introduced **"Turbo CORE"** technology into their Phenom desktop processor, and plans to include a greatly enhanced version in the upcoming Bulldozer family of processors.

According to AMD:

> *"AMD Turbo CORE is deterministic, governed by power draw, not temperature as other competing products are. This means that even in warmer climates you'll be able to take advantage of that extra headroom if you choose. This helps ensure a max frequency is workload dependent, making it more consistent and repeatable."*

AMD Turbo CORE allows individual cores in the processor to speed up from the base clock speed up to the Thermal Design Power (TDP) level, automatically adding extra single-threaded performance for the processor. Conceptually, it is the opposite of AMD PowerNow! Technology; instead of trying to watch for usage patterns and lowering the processor core speed to try to reduce power consumption, Turbo CORE is watching the power consumption to see how high it can increase clock speed.

Should the processor get too close to the TDP power limit, it will automatically throttle back somewhat to ensure that it is continuing to operate within the specified TDP guidelines. This allows for significantly higher maximum clock speeds for the individual cores.

AMD has stated that Bulldozer will boost the clock speed of all 16 cores by 500 MHz, even when all cores are active with server workloads. Even higher boost states will be available with half of the cores active. AMD has not disclosed how much the clock speed boost will be when only half of the cores are active. When the Bulldozer processor is finally launched you will see processors marketed with a base and a maximum frequency; base will reflect the actual clock speed on the processor and max will reflect the highest AMD Turbo CORE state.

Just like with Intel Turbo Boost Technology, I think this is a very beneficial feature and one that you should take advantage for database server usage. I don't see any controversy here (such as with hyper-threading).

One free tool you can use to monitor the effect of Turbo Boost or Turbo CORE in real-time is called TMonitor, which is available from CPUID. Figure 7-1 shows the tool in action. What you see is the speed changes in each core, with all of the cores displayed simultaneously (unlike CPU-Z, which only shows the speed of one core at a time).

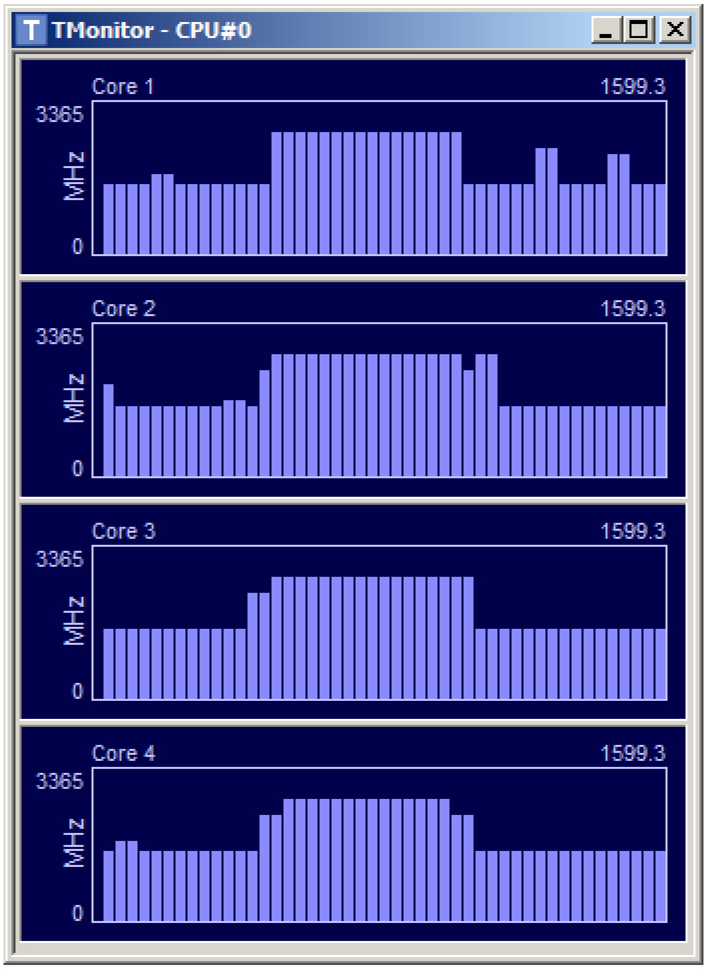
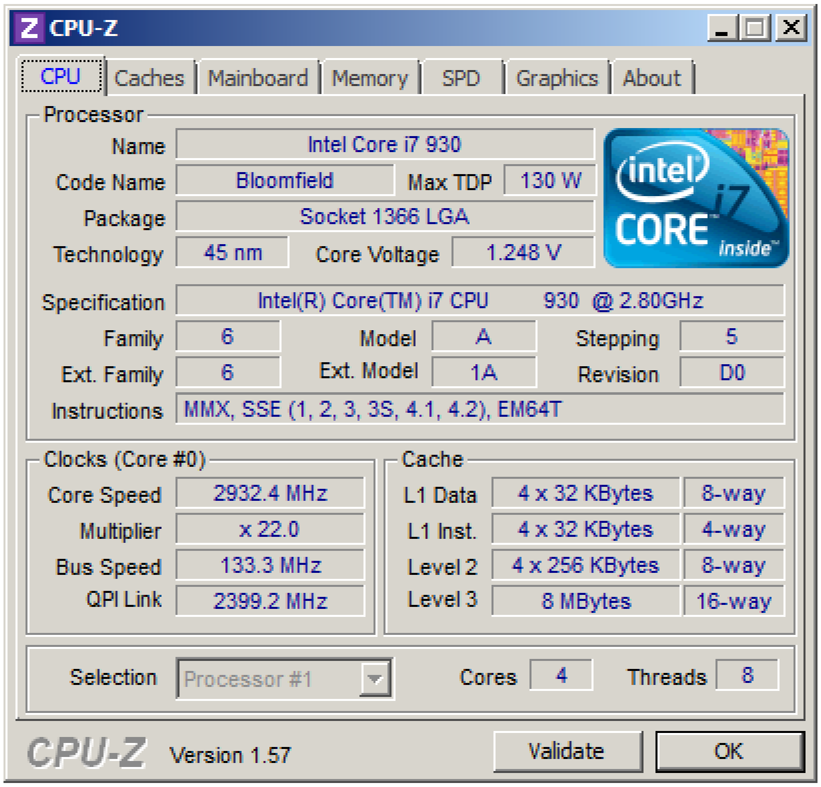


Figure 7-1: TMonitor running while the system is under a load.

Figure 7-2 was captured on my desktop, rather than a database server, using the CPU-Z tool (see TIP 14). It shows the core speed of Core #0 running at 2,932 MHz (i.e. 2.93 GHz), even though the rated base clock of my Core i7 930 is only 2.8 GHz. The Core i7 930 processor only has the first generation Turbo Boost Technology, which is much less aggressive than the newer Turbo Boost Technology 2.0 in the Sandy Bridge processors.



Processor

Name: Intel Core i7 930
Code Name: Bloomfield
Max TDP: 130 W
Package: Socket 1366 LGA
Technology: 45 nm
Core Voltage: 1.248 V
Specification: Intel(R) Core(TM) i7 CPU 930 @ 2.80GHz
Family: 6
Model: A
Stepping: 5
Ext. Family: 6
Ext. Model: 1A
Revision: D0
Instructions: MMX, SSE (1, 2, 3, 3S, 4.1, 4.2), EM64T

Clocks (Core #0)

Core Speed: 2932.4 MHz
Multiplier: x 22.0
Bus Speed: 133.3 MHz
QPI Link: 2399.2 MHz

Cache

L1 Data	4 x 32 KBytes	8-way
L1 Inst.	4 x 32 KBytes	4-way
Level 2	4 x 256 KBytes	8-way
Level 3	8 MBytes	16-way

Selection: Processor #1
Cores: 4
Threads: 8

CPU-Z Version 1.57

Figure 7-2: CPU-Z showing Core #0 running above rated speed.

8 Carefully evaluate the motherboard and associated chipsets

When you are evaluating server models (such as a Dell PowerEdge R710) from a particular server vendor, you should always find out which chipset is being used in that server. Intel processors are designed to use Intel chipsets. For each processor sequence (3000, 5000, 6000, 7000, or 9000), several different chipsets will be compatible with a given processor. Intel usually has at least two different chipsets available for any processor, with each chipset offering different levels of functionality and performance.

What you want to focus on, as a SQL Server DBA, are:

- number of sockets
- number of memory slots and supported memory architecture
- number and type of expansion slots and reliability, availability, and serviceability (RAS) features
- type and number of other integrated components, such as Network Interface Cards (NICs).

The number of sockets will determine how many physical processors you can install (see Tip 9 for further discussion).

The number of memory slots, along with the limitations of the memory controller in the chipset or the CPU itself, ultimately determines how much RAM you can install in a database server (See Tip 10 for further discussion).

The number and types of expansion slots affect how many RAID (Redundant Array of Independent Disk) controllers or Host Bus Adapters you can install in a server, which ultimately limits your total available I/O capacity and performance (see Tip 28). Also, with the widespread acceptance of 2.5" SAS (Serial Attached SCSI) drives, it is possible to have many more internal drives in a 1U or 2U system than you could have with 3.5" drives.

Also important are reliability, availability, and serviceability (RAS) features. The goal is to have a system that is always available, never corrupts data, and delivers consistent performance, while allowing maintenance during normal operation. Examples of RAS features include, for example, hot swappable components for memory, processors, fans, and power supplies. You also want to see redundant components that eliminate common single points of failure, such as a power supply, RAID controller, or an NIC.

9 Two x 2-sockets is usually better than one x 4-socket

Traditionally, it has been very common to use a 4-socket machine for most database server work, while 2-socket servers were often used for web servers or application servers. Historically, 2-socket database servers simply did not have enough processor capacity, memory capacity, or I/O capacity to handle many "heavy" database workloads.

For example, back in 2006, you could buy a 2-socket Dell PowerEdge 1850 server, with two Intel Xeon "Irwindale" 3.2 GHz processors and 16 GB of RAM (with a total of four logical cores, with hyper-threading enabled). It had a Geekbench score of about 2,200. It was fine for an application or web server, but it did not have the CPU horsepower or memory capacity for a heavy-duty database workload.

Around the same time, you could buy a 4-socket Dell PowerEdge 6800, with four dual-core, Intel Xeon 7040 "Paxville" 3.0 GHz processors and 64 GB of RAM (with a total of 16 logical cores, with hyper-threading enabled). This was a much better choice for a database server because of the additional processor, memory, and I/O capacity compared to a PowerEdge 1850. Even so, its Geekbench score was only about 4,400, which is pretty pathetic by today's standards. Back in 2006–2007, it still made sense to buy a 4-socket database server for most database server workloads.

However, with the advances in processor technology and the improvements in memory density over the past three or four years, it is time to turn that conventional wisdom on its head. In general, the latest 2-socket systems, from Intel particularly, but also from AMD, are extremely capable, and should have more than enough capacity to handle all but the very largest database server workloads. As discussed in Tip 3, this can lead to substantial savings in the cost of SQL Server licenses.

Also, there are two reasons why it makes sense to use two 2-socket servers in preference to one 4-socket server, assuming you can split your database workload between two database servers (by moving databases or using vertical or horizontal partitioning of an existing large database). Firstly, the technology of 4-socket Intel systems (in terms of processors and chipsets) usually lags about one year behind 2-socket server systems. For example, the processor technology available in the 2-socket 45 nm Intel Xeon 5500 systems, launched in early 2009, was not available in 4-socket systems until early 2010.

AMD processors

Note that, historically, AMD processors have used the same processor technology in both 2- and 4-socket servers, though they have struggled to match the speed and power of Intel processors, especially in the OLTP space – but see TIP 19 *for further discussion.*

By mid-2010, the latest 2-socket Intel systems were based on 32 nm Intel Westmere-EP architecture processors. For example, you could buy a 2-socket Dell PowerEdge R710, with powerful six-core INTEL XEON X5680 processors (with a total of 24 logical cores), and a Geekbench score of about 22,500.At this time, the latest 4-socket systems were still based on the older 45 nm Intel Nehalem-EX architecture processors (e.g. the 45 nm 2.26 GHz Xeon X7560 for systems with four or more sockets. The Xeon X5690 is much faster than the Xeon X7560 for single-threaded OLTP performance. The 4-socket-capable 32 nm Intel Xeon E7-4800 Westmere-EX was not introduced until April 5, 2011.

The second factor to consider is **hardware scaling**. You might assume, incorrectly, that a 4-socket system will have twice as much overall load capacity or scalability as the equivalent 2-socket system. In fact, depending on the exact processor and chipsets involved, you typically see a hardware scaling factor of anywhere from 1.5 to 1.8 (rather than the theoretical maximum of 2.0) as you double the number of sockets.

Overall, therefore, the latest 2-socket Intel processors will use newer technology and have faster clock speeds and better single-threaded performance than the latest 4-socket Intel processors. As noted in TIP 1, single-threaded performance is especially relevant to OLTP workloads, where queries are usually relatively simple and low in average elapsed time, normally running on a single processor core. Finally, going through the architectural and engineering work required to partition or shard your database is a good long-term strategy, since it will allow you to scale out at the database level, whether it is with on-premises SQL Server or with SQL Azure.

The only big exception to the rule of preferring 2-socket to 4-socket servers would be a case where you absolutely need to have far more memory in a single server than you can get in a 2-socket machine (a Dell PowerEdge R710 can now go up to 288 GB if you are willing to pay for 16 GB DIMMs (Dual in-line Memory Modules), and you are unable to do any re-engineering to split up your load.

10 Optimize memory capacity

For any given processor, a server vendor may make available a number of different motherboards and associated chipsets that support it. When you are evaluating and selecting a motherboard and chipset, a very important consideration (along with number of sockets, number of expansion slots) is the number of memory slots and supported memory architecture.

As discussed in the previous tip, it has been standard practice to choose a 4-socket system for most database server applications, simply because 2-socket systems usually did not have enough processor cores, memory sockets or expansion slots to support a normal database server workload. This prescription has changed dramatically since late 2008. Prior to that time, a typical 2-socket system was limited to 4–8 processor cores and 4–8 memory slots. Now, we see 2-socket systems with up to 24 processor cores, and 18–24 memory slots.

The number of memory slots, along with the limitations of the memory controller in the chipset or the CPU itself, ultimately determines how much RAM you can install in a database server. The basic rule of thumb for SQL Server is that you can never have too much RAM. Server RAM is relatively inexpensive, especially compared to SQL Server licenses, and having as much RAM as possible is extremely beneficial for SQL Server performance in many ways.

- It will help reduce read I/O pressure, since it increases the likelihood that SQL Server will find in the buffer pool any data requested by queries.
- It can reduce write I/O pressure, since it will enable SQL Server to reduce the frequency of checkpoint operations.
- It is cheap insurance against I/O sizing miscalculations or spikes in your I/O workload.

So, on this basis, the more memory slots, the better. For example, many server vendors currently offer 1U, 2-socket servers that have a total of twelve memory slots, while they also offer nearly identical 2U servers that have a total of eighteen memory slots. In most cases, I would prefer the server that had eighteen memory slots, since that would allow me to ultimately have more memory in the database server. This assumes that the memory controller (whether in the chipset or the integrated memory controller in the processor) will support enough memory to fill all of those available memory slots.

A related consideration is **memory density**, by which I mean the memory capacity of each memory stick. Any relatively new database servers should be using DIMMs with one of the modern DRAM (Dynamic Random Access Memory) implementations, specifically either DDR2 SDRAM or DDR3 SDRAM memory.

At the time of writing (July 2011), 8 GB SDRAM DIMMs represented the "sweet spot" in the price-performance curve, because of the prohibitive cost of 16 GB variety. Violating this sweet spot rule might cause you to spend far more on memory for the server than the rest of the server combined. However, once 32 GB sticks of RAM are available in 2011, the price-performance sweet spot will shift pretty quickly towards 16 GB sticks of RAM.

Many older processors and chipsets will not be able to use that capacity offered by the forthcoming 32 GB sticks. One known exception will be the 32 nm Intel E7 (Westmere-EX) processor series that is meant for use in 4-socket, or larger, systems. This opens up the possibility, sometime in 2011, of a 4-socket system with 4 processors x 16 DIMMS per processor x 32 GB per DIMM = 2 TB of RAM. You'll have to have very deep pockets though; 32 GB SDRAM DIMMs will be very, very expensive when they are first available.

11 Use 64-bit hardware, operating system and SQL Server

I firmly believe that you should *only* be using 64-bit SQL Server for new installations. The only reason you should even consider using a 32-bit version of SQL Server is if you have a legacy application or data source that only works with a 32-bit version of SQL Server. The whole "32-bit versus 64-bit" question will be moot fairly soon for new installs, since I strongly suspect that SQL Server Denali will be the last version of SQL Server that will even have 32-bit support.

Most recent releases of Windows Server are available in three different versions: x86 (32-bit), x64 (64-bit), and ia64 (64-bit Itanium). However, Windows Server 2008 R2 is 64-bit only (and will be the last version of Windows Server that will support ia64). Other server products from Microsoft, such as Exchange 2010 and SharePoint 2010, are also 64-bit only. In short, the 32-bit era is close to being over, and I will not miss it one bit. I strongly recommend you move to 64-bit Windows and SQL Server versions as quickly as possible.

Historically, the biggest barrier to its adoption of 64-bit SQL Server has been lack of hardware support, but this obstacle has largely been removed. All server-grade processors released in the last six to seven years have native 64-bit support. The other major obstacle, lack of 64-bit drivers for hardware components such as NICs, HBAs (Host Bus Adapters), and RAID controllers, is not really an issue any more, as the advent of 64-bit Windows Server 2008 R2, in 2009, has meant that 64-bit drivers are much more readily available for nearly all devices.

You can confirm if your processor has x64 support by running CPU-Z and looking at the Instructions section on the CPU tab. If you see EM64T (for Intel processors) that means it does have x64 support. Unless your processor is extremely old, it will support x64.

One sticking point, and the bane of many a DBA's life when it comes to upgrades, is third-party databases. Some common reasons for still using an x86 version of Windows Server include:

- third-party ISV databases that require x86
- third-party ISV databases that have not been "certified" on x64
- third-party applications using older data access technology, such as 32-bit ODBC drivers or 32-bit OLE-DB providers that do not work with an x64 database server.

I am one of the lucky ones; at NewsGator, we have no third-party databases, and we are a 100% Microsoft shop, so all of the applications that use my databases use the ADO.NET Provider. As a result, NewsGator's production SQL Server environment has been 100% 64-bit since April 2006.

12 Understand SQL Server licensing restrictions

As a slight counterbalance to the CPU and memory free-for-all, there are certain licensing restrictions, depending on the version and edition of SQL Server that you plan to use; it's very important that you understand these restrictions, and their implications, when changing SQL Server versions and editions.

The Enterprise editions of both SQL Server 2005 and SQL Server 2008 were unrestricted in terms of physical cores, but were limited to 64 logical processors. However, SQL Server 2008 R2 Standard Edition and SQL Server 2008 R2 Enterprise Edition come with more restrictive hardware license limits compared to the SQL Server 2008 versions of both of those editions.

SQL Server 2008 R2 Enterprise Edition imposes a new limit of eight physical processor sockets, but will theoretically let you use up to 256 logical processors (as long as you are running on Windows Server 2008 R2). However, this really is currently a theoretical limit, since it would require a processor with 32 logical cores. As of April 2011, the highest logical core count you can get in a single processor socket is 20 (the new Intel Xeon E7 series).

Also, the RAM limit for R2 EE has changed from "operating system limit," as it was in the 2008 release, to a hard limit of 2 TB. If you need more than eight physical processors or more than 2 TB of RAM, you will need to run SQL Server 2008 R2 Data Center Edition.

In **SQL Server 2008 R2 Standard Edition**, the physical socket limit is unchanged from SQL Server 2008, and is still four processor sockets. More significant is a new RAM limit for SE of 64 GB. This lowered limit may catch many people by surprise, since it is very easy to have much more than 64 GB of RAM, even in a 2-socket server. You should keep this RAM limit in mind if you are buying a new server and you know that you will be using Standard Edition. One possible workaround for this limit would be to have a second or third instance of SQL Server 2008 R2 Standard Edition installed on the same machine, so you could use more than the 64 GB limit for a single instance.

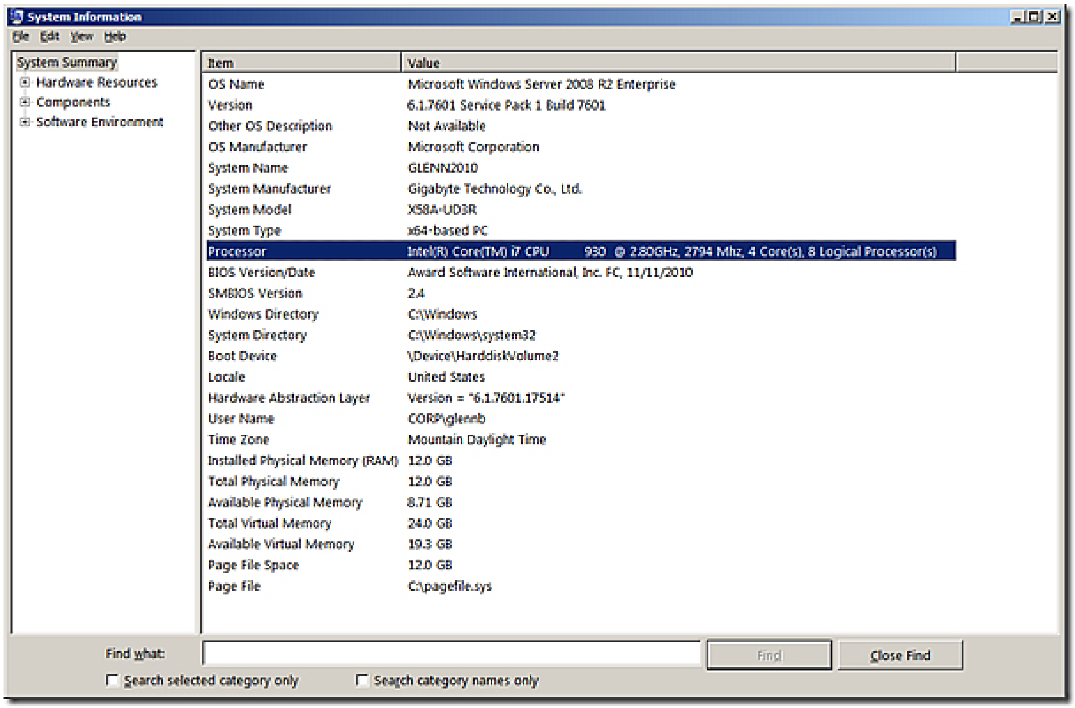
13 Use the built-in hardware discovery tools

I am always disappointed when I ask a DBA what kind of hardware is in one of their database servers, and I get some vague answer like "I don't know for sure. I think it is a Xeon?" As a database professional, you really do need to know the gory details of the hardware that you are using to run SQL Server. Old, under-powered hardware can make even the best designed and optimized database run poorly, making you, the DBA look bad.

There are several useful tools, built right into Windows, which can give you some level of detail regarding the installed hardware, operating system and software: **msinfo32**, **Windows Task Manager** and **Computer Properties** dialog. Each of these tools requires that you have access to log in to that server, which might be a problem for some people, depending on the policies of their organization. If you are barred from accessing your server directly, take a look at TIP 15 to find out how to get some information from T-SQL and the Dynamic Management Views.

The MSINFO32.EXE tool is built into all recent versions of Windows. You can simply type `msinfo32` in a **Run** window, and you will see the System Information dialog shown in Figure 13-1, providing details of operating system and build number, manufacturer and model number of the server, processor information, physical and virtual memory details, and more.

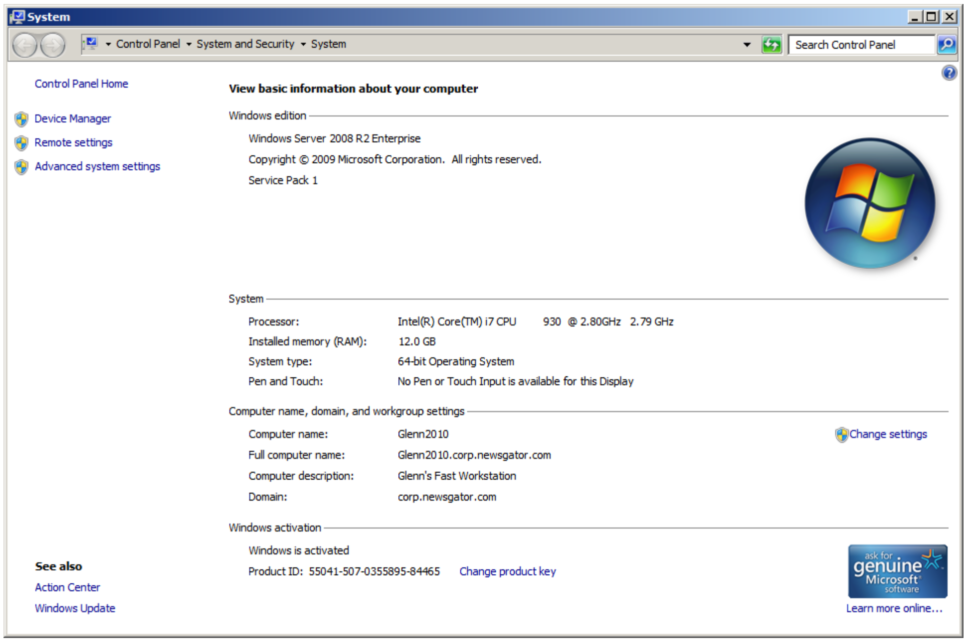
In this case, the tool shows that I have an Intel Core i7 930 CPU, with 12 GB of RAM, running Windows Server 2008 R2 Enterprise Edition, with Service Pack 1.



Item	Value
OS Name	Microsoft Windows Server 2008 R2 Enterprise
Version	6.1.7601 Service Pack 1 Build 7601
Other OS Description	Not Available
OS Manufacturer	Microsoft Corporation
System Name	GLENN2010
System Manufacturer	Gigabyte Technology Co., Ltd.
System Model	X58A-UD3R
System Type	x64-based PC
Processor	Intel(R) Core(TM) i7 CPU 930 @ 2.80GHz, 2794 Mhz, 4 Core(s), 8 Logical Processor(s)
BIOS Version/Date	Award Software International, Inc. FC, 11/11/2010
SMBIOS Version	2.4
Windows Directory	C:\Windows
System Directory	C:\Windows\system32
Boot Device	\Device\HarddiskVolume2
Locale	United States
Hardware Abstraction Layer	Version = "6.1.7601.17514"
User Name	CORP\glennb
Time Zone	Mountain Daylight Time
Installed Physical Memory (RAM)	12.0 GB
Total Physical Memory	12.0 GB
Available Physical Memory	8.71 GB
Total Virtual Memory	24.0 GB
Available Virtual Memory	19.3 GB
Page File Space	12.0 GB
Page File	C:\pagefile.sys

Figure 13-1: System Information dialog.

The Computer Properties dialog, shown in Figure 13-2, is accessed by going to the Start Button, right-clicking on **Computer**, and selecting **Properties**. It reveals the Windows Version and Edition, the type and number of physical processors, the amount of installed memory, and whether the operating system is 32-bit or 64-bit.



View basic information about your computer

Windows edition

Windows Server 2008 R2 Enterprise
Copyright © 2009 Microsoft Corporation. All rights reserved.
Service Pack 1

System

Processor:	Intel(R) Core(TM) i7 CPU 930 @ 2.80GHz 2.79 GHz
Installed memory (RAM):	12.0 GB
System type:	64-bit Operating System
Pen and Touch:	No Pen or Touch Input is available for this Display

Computer name, domain, and workgroup settings

Computer name:	Glenn2010
Full computer name:	Glenn2010.corp.newsgator.com
Computer description:	Glenn's Fast Workstation
Domain:	corp.newsgator.com

Change settings

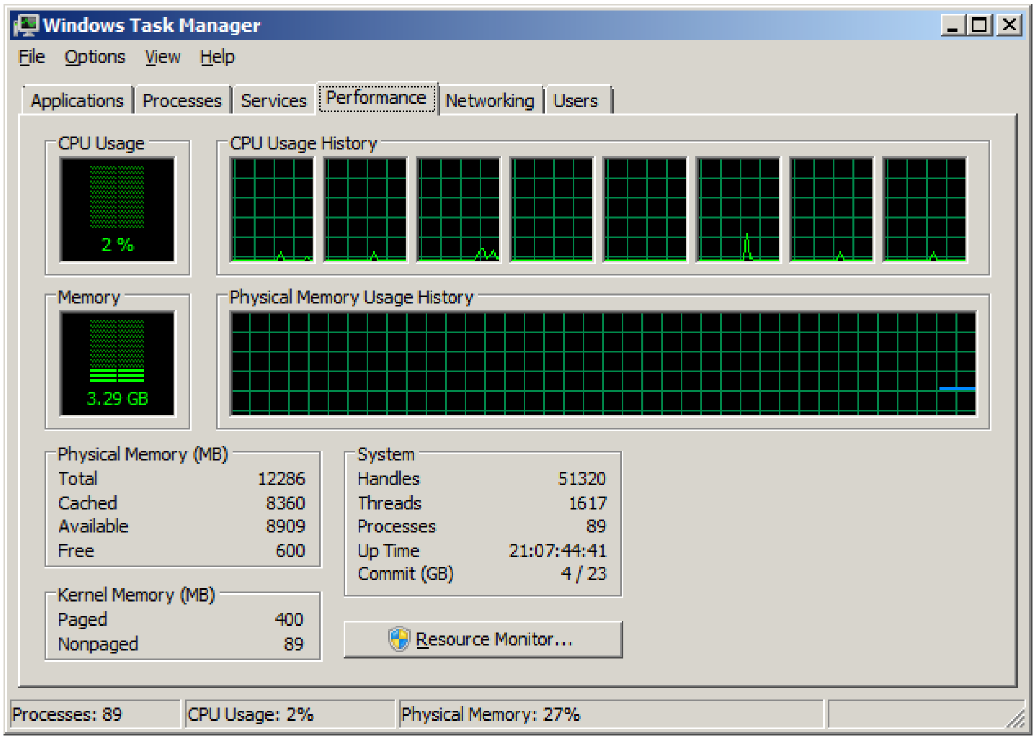
Windows activation

Windows is activated
Product ID: 55041-507-0355895-84465 Change product key

ask for genuine Microsoft software
Learn more online...

Figure 13-2: Computer Properties dialog.

Windows Task Manager, shown in Figure 13-3, is a useful tool for gathering and confirming general hardware information. You can get there by right-clicking on the Task Bar, and choosing Start Task Manager. The Performance tab tells you how many logical processors are visible to Windows (the number of sections you see in CPU Usage History), along with the amount of RAM that you have installed.



Physical Memory (MB)	
Total	12286
Cached	8360
Available	8909
Free	600

Kernel Memory (MB)	
Paged	400
Nonpaged	89

System	
Handles	51320
Threads	1617
Processes	89
Up Time	21:07:44:41
Commit (GB)	4 / 23

Processes: 89 | CPU Usage: 2% | Physical Memory: 27%

Figure 13-3: Windows Task Manager.

All of these tools are built into the operating system, so there should be absolutely no issues with running them on a server.

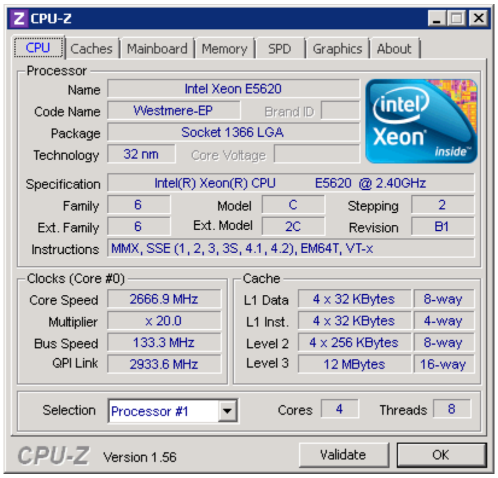
14 Get to know your hardware intimately with CPU-Z

While the built-in Windows tools are useful, if you want a much deeper level of detail regarding your processors and hardware, I recommend that you install the wonderful CPU-Z utility, available for free from cpuid.com. The latest release of the tool is version 1.58, which came out on June 24, 2011. I always download and use the 64-bit, English, no install zip version of the tool. You simply copy the small executable out of the zip file. By running this executable, you can analyze the system in a few seconds, answering several important questions that you may have about the machine including those below.

- Is the hardware 64-bit capable?
- What is the size of the installed memory sticks – e.g. is it four 1 GB sticks or two 2 GB sticks?
- How many cores? How many threads (which equates to logical CPUs visible to the operating system)?
- What processor(s) is/are in use? CPU-Z provides processor model number, codename, manufacturing technology, clock speed (rated and actual), supported instruction sets, and cache types and sizes.

Figure 14-1 shows the **CPU** tab of the tool, for a quad-core Intel Xeon E5620 (Westmere-EP) processor, and gives you a good feel for the very high level of detail you can get about the CPU(s) in a machine. This screen also confirms whether the processor is 64-bit capable, which it is in this case, since EM64T is listed in the **Instructions** list, and whether the processor supports hardware virtualization (which it does, since VT-x is also in the list). When you see the number of threads listed at double the number of cores, that tells you that the processor has hyper-threading enabled. The cores and threads listing is per physical processor. If you want to know the number of physical processors, you need to look at the **Selection** drop-down to list each physical processor.

With AMD processors, you will see "x86-64" in the **Instructions** list to indicate that the processor is 64-bit capable, and you would see "AMD-V" (which means AMD Virtualization) to indicate hardware virtualization support.



Processor
Name: Intel Xeon E5620
Code Name: Westmere-EP   Brand ID
Package: Socket 1366 LGA
Technology: 32 nm   Core Voltage
Specification: Intel(R) Xeon(R) CPU E5620 @ 2.40GHz
Family: 6   Model: C   Stepping: 2
Ext. Family: 6   Ext. Model: 2C   Revision: B1
Instructions: MMX, SSE (1, 2, 3, 3S, 4.1, 4.2), EM64T, VT-x

Clocks (Core #0)
Core Speed: 2666.9 MHz
Multiplier: x 20.0
Bus Speed: 133.3 MHz
QPI Link: 2933.6 MHz

Cache
L1 Data: 4 x 32 KBytes, 8-way
L1 Inst.: 4 x 32 KBytes, 4-way
Level 2: 4 x 256 KBytes, 8-way
Level 3: 12 MBytes, 16-way

Selection: Processor #1   Cores: 4   Threads: 8

Version 1.56

Figure 14-1: Example of the CPU tab in CPU-Z.

Rather than show screens of **Caches**, **Mainboard**, **Memory**, and **SPD** tabs, I'll simply summarize the sort of information available on each, and leave you to investigate further.

- **Caches** – more information about the various CPU caches, including the L1 Data Cache, L1 Instruction Cache, L2 Cache, and L3 Cache.
- **Mainboard** – information about the motherboard, chipset, and main BIOS version.
- **Memory** – type and total amount of RAM installed in the system. You can compare the values you find here with the specifications for your server model to see if you can add any additional RAM.
- **SPD** (Serial Presence Detect) – tells you how many memory slots are available in the system, and the size, type, speed, and manufacturer of the memory stick that is installed in each slot. May be blank, depending on the type and age of your machine, and the version of CPU-Z.

One question I get asked often is WHETHER IT IS SAFE to run CPU-Z on a production SQL Server. All I can say is that I have been using it for years, with absolutely no problems. Many other well-known people in the SQL Server community have been doing the same thing. I think it is an extremely valuable tool.

15 No RDP access? Investigate hardware using sys.dm_os_sys_info

Occasionally, a DBA will not have direct access to their database servers (i.e. they cannot log in to their database server via Remote Desktop Protocol (RDP). Fortunately, there is quite a bit you can learn about your hardware using SSMS, T-SQL and a DMV.

You can query SYS.DM_OS_SYS_INFO and find out the physical CPU Socket count, hyper-thread ratio, logical CPU count, and the amount of physical memory in the machine. Each new major version of SQL Server has added some additional columns to `sys.dm_os_sys_info`, which makes this query a little more useful. That is why I have three different versions of the query shown in Listing 15-1.

```
-- Hardware Information for SQL Server 2005
SELECT  cpu_count AS [Logical CPU Count] ,
        hyperthread_ratio AS [Hyperthread Ratio] ,
        cpu_count / hyperthread_ratio AS [Physical CPU Count] ,
        physical_memory_in_bytes / 1048576 AS [Physical Memory (MB)]
FROM    sys.dm_os_sys_info
OPTION  ( RECOMPILE ) ;

-- Hardware information from SQL Server 2008 and 2008 R2
SELECT  cpu_count AS [Logical CPU Count] ,
        hyperthread_ratio AS [Hyperthread Ratio] ,
        cpu_count / hyperthread_ratio AS [Physical CPU Count] ,
        physical_memory_in_bytes / 1048576 AS [Physical Memory (MB)] ,
        sqlserver_start_time
FROM    sys.dm_os_sys_info
OPTION  ( RECOMPILE ) ;
```

```
-- Hardware information from SQL Server Denali
SELECT  cpu_count AS [Logical CPU Count] ,
        hyperthread_ratio AS [Hyperthread Ratio] ,
        cpu_count / hyperthread_ratio AS [Physical CPU Count] ,
        physical_memory_kb / 1024 AS [Physical Memory (MB)] ,
        affinity_type_desc ,
        virtual_machine_type_desc ,
        sqlserver_start_time
FROM    sys.dm_os_sys_info
OPTION  ( RECOMPILE ) ;
```

Listing 15-1: Getting hardware information from SQL Server with T-SQL.

Note that the `virtual_machine_type_desc` column, providing details of the hardware virtualization environment is brand new in the CTP1 build of SQL Server Denali and does not, at time of writing, show up in the BOL entry.

Unfortunately, we still have no idea what type of processor we are dealing with. Is it a top-of-the-line, fire-breathing, 3.46 GHz Intel Xeon X5690, or is it a much more humble, older processor? This is information a DBA needs at his or her fingertips when, for example:

- investigating the possibility of implementing data compression for some indexes in a database
- attempting to allocate and balance workload across all available servers
- looking for opportunities to upgrade or consolidate hardware.

The DBA needs to be able to access this processor information even if they can't get direct access to the server (and so to tools like CPU-Z), and this could be solved easily, with the addition of a "processor_description" column to the `sys.dm_os_sys_info` DMV.

If you agree with me about this issue, you can help convince Microsoft to take care of it in the SQL Server Denali release cycle, by going to this Microsoft Connect Item and voting and/or leaving a comment.

16 Analyze the TPC-E benchmarks

The Transaction Processing Performance Council (TPC) is a non-profit organization, founded in 1988, which aims to define transaction processing and database benchmarks, and to disseminate to the industry objective, verifiable performance data. Their benchmarks are used widely in evaluating the performance of database systems.

The benchmark of particular interest is the TPC-E, an OLTP performance benchmark that simulates the OLTP workload of a brokerage firm that interacts with customers using synchronous transactions and with a financial market using asynchronous transactions.

Compared to the older TPC-C benchmark, the TPC-E:

- uses a more realistic data model (with referential integrity, realistic data skews, and so on)
- is more CPU-intensive
- reduces the I/O workload – making it both less expensive and more realistic, since the sponsoring vendors will not feel as much pressure to equip their test systems with disproportionately large disk subsystems in order to get the best test results
- requires the storage media to be fault tolerant (which means no RAID 0 arrays).

The TPC-E implementation is broken down into a **driver** and a **System Under Test** (SUT), separated by a mandatory network. The driver represents the various client devices that would use an N-tier client-server system, abstracted into a load generation system. The SUT has multiple application servers (Tier A) that communicate with the database server and its associated storage subsystem (Tier B). TPC provides a transaction harness component that runs in Tier A, while the test sponsor provides the other components in the SUT.

The performance metric for TPC-E is transactions per second (**tpsE**). The actual tpsE score represents the average number of Trade Result transactions executed within one second. To be fully compliant with the TPC-E standard, all references to tpsE results must include the tpsE rate, the associated price per tpsE, and the availability date of the priced configuration.

Some people tend to treat with skepticism, or even completely disregard, the value of these benchmark results, regarding them as little more than marketing exercises by various vendors, using unrealistic workloads and incredibly expensive hardware. I think this attitude is a mistake; as long as you know how to interpret the results, and realize their purpose and limitations, I believe there is real value to be had in comparing the TPC benchmarks for various systems.

Earlier this year, for example, I imported the current official TPC-E results spreadsheet into a SQL Server 2008 R2 database, did a little data cleansing, added a few useful columns that are not in the original official results spreadsheet (such as `MemorySize`, and `SpindleCount`) and then wrote a few queries to see if anything interesting might reveal itself from the actual raw data. My results, ranked by TpsE per Socket, can be found on my BLOG and I won't repeat them here, but they provided a useful insight into the current best-performing processors for OLTP (confirming the Intel Xeon X5680 as the current cream of the crop – see TIP 18), and correlated well with my own Geekbench results (see TIP 17).

I also find it interesting, more generally, to dig around in any newly-published benchmarks, to see what configurations and settings were used to achieve the results, and so on. Recently, TPC-E results were posted for an HP PROLIANT DL580 G7 SERVER with a 2,454.51 tpsE score for a 4-socket system. This system has four, 10-core INTEL XEON E7-4870 processors that have a total of 80 logical cores for the system. It also has 1 TB of RAM and 1,100 spindles in its I/O subsystem, using an 11 TB initial database size for the test.

If you take a look in the **Executive Summary** section, you can see that it is running SQL Server 2008 R2 Enterprise Edition on top of Windows Server 2008 R2 Enterprise Edition SP1. It is using RAID 10 for both the data files and log file, with (950) 72 GB, 6 Gbps, 15K SAS drives, (150) 146 GB, 6 Gbps, 15K SAS drives, and four 400 GB SSDs (Solid State Drives).

Digging deeper into the **Supporting Files** for the submission, you can find how HP decided to configure their SQL Server 2008 R2 instance for the benchmark. There are some settings that I do not agree with (even for a benchmark), and several settings that I would never want to use in a production environment. The complete set is shown in Listing 16-1.

```
-- HP SQL Configuration Options for TPC-E Benchmark
-- Don't use these settings on a production server!
EXEC sp_configure 'show advanced options', '1'
RECONFIGURE WITH OVERRIDE
go

EXEC sp_configure 'max server memory', 1038000  -- Value depends on RAM
EXEC sp_configure 'recovery interval', '32767'  -- Don't use
EXEC sp_configure 'awe enabled', '0'
EXEC sp_configure 'max degree of parallelism', '1'
EXEC sp_configure 'lightweight pooling', '1'   -- Don't use
EXEC sp_configure 'priority boost', '1'        -- Don't use
EXEC sp_configure 'max worker threads', 3000   -- Don't use
EXEC sp_configure 'default trace enabled', 0   -- Don't use
go
RECONFIGURE WITH OVERRIDE
Go
```

Listing 16-1: Configuration for HP Proliant DL580 G7 server in TPC-E benchmark.

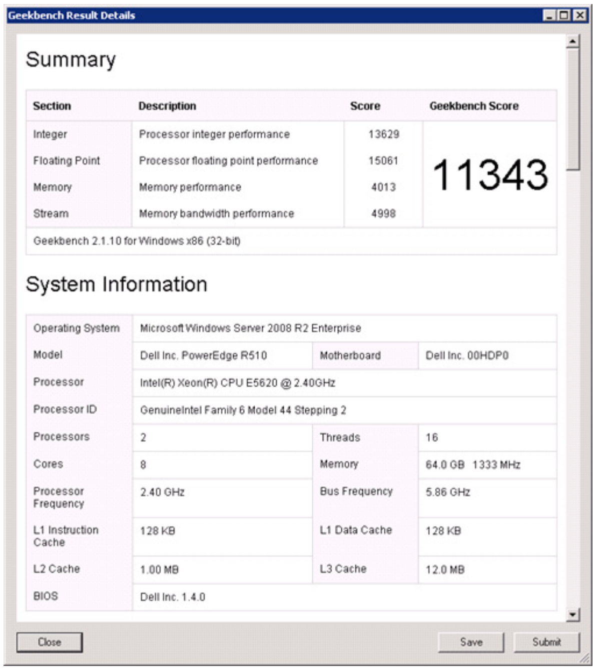
17 Perform your own benchmarks with Geekbench

Geekbench is a cross-platform, synthetic benchmark tool from Primate Labs. It provides a comprehensive set of benchmarks designed to quickly and accurately measure processor and memory performance.

There are 32-bit and 64-bit versions of Geekbench, but in trial mode you can only use the 32-bit version. A license for the Windows version is only $12.99. The latest released version is 2.1.13, which became available on March 12, 2011.

One nice thing about Geekbench is that there are no configuration options whatsoever. All you have to do is just install it and run it, and within two to three minutes you will have an overall benchmark score for your system, which is further broken down into four sections, two measuring processor performance, **Integer** (12 scores) and **Floating Point** (14 scores), and two measuring memory bandwidth performance, **Memory** (5 scores), and **Stream** (8 scores).

I tend to focus first on the overall Geekbench score, and then look at the top level scores for each section, as shown in Figure 17-1. These scores can be used to measure and compare the absolute processor and memory performance between multiple systems, or between different configurations on the same system.



Summary

Section	Description	Score	Geekbench Score
Integer	Processor integer performance	13629	11343
Floating Point	Processor floating point performance	15061	
Memory	Memory performance	4013	
Stream	Memory bandwidth performance	4998	

Geekbench 2.1.10 for Windows x86 (32-bit)

System Information

Operating System	Microsoft Windows Server 2008 R2 Enterprise		
Model	Dell Inc. PowerEdge R510	Motherboard	Dell Inc. 00HDP0
Processor	Intel(R) Xeon(R) CPU E5620 @ 2.40GHz		
Processor ID	GenuineIntel Family 6 Model 44 Stepping 2		
Processors	2	Threads	16
Cores	8	Memory	64.0 GB 1333 MHz
Processor Frequency	2.40 GHz	Bus Frequency	5.86 GHz
L1 Instruction Cache	128 KB	L1 Data Cache	128 KB
L2 Cache	1.00 MB	L3 Cache	12.0 MB
BIOS	Dell Inc. 1.4.0		

Figure 17-1: Geekbench summary and system information.

I always run each test at least three times in succession, and take the average overall Geekbench score. This, in just a few minutes, gives me a pretty good idea of the overall processor and memory performance of the system.

I like to run Geekbench on every available non-production system, so that I can save the various system configurations and Geekbench score results in a spreadsheet. Then, I can use this information to roughly compare the overall CPU/memory "horsepower" of different systems. This is very useful if you are doing capacity or consolidation planning.

For example, let's say that you have an existing database server with four dual-core 3.4 GHz Xeon 7140M processors and 64 GB of RAM, and this system has an averaged Geekbench score of 5,282. You are assessing a new system that has two six-core 3.33 GHz Xeon X5680 processors and 72 GB of RAM, and the new system has an averaged Geekbench score of 22,484. In this situation, you could feel extremely confident from a CPU and RAM perspective that the new, 2-socket system could handle the workload of the old 4-socket system, with plenty of room to spare. You could use the extra CPU capacity of the new system to handle additional workload, or you could use it to reduce your I/O requirements by being more aggressive with SQL Server data compression and backup compression.

In the absence of a large number of different systems on which to run Geekbench, you can still browse online the published Geekbench results for various systems. Simply look up the results for the system closest in spec to the one being evaluated. You can use the search function on that page to find systems with a particular processor, and then drill into the results to get a better idea of its relevance.

18 For OLTP server workloads, consider Intel Xeon X5690 (July 2011)

Until recently, Intel provided several families of processors, depending on the number of sockets in your server. For example, the Xeon 3xxx family was for single-socket servers, the Xeon 5xxx family was for 2-socket servers, and the Xeon 7xxx family was for 4-socket (or more) servers. In addition, they have now introduced the Intel Xeon E7 processor series, providing, for example, a Xeon E7-2800 series processor for a 2-socket server, a Xeon E7-4800 series processor for a 4-socket server, or a Xeon E7-8800 series processor for an 8-socket (or more) server.

Intel Processor numbering

It is well worth taking the time to study and understand Intel's classic numbering system for Xeon processors, as well as the new system introduced for the E3 and E7 (and soon E5) Xeon processor families, as it will help you understand the capabilities, relative age, and relative performance of a particular processor. An overview of how to decode the processor model numbers can be found on Intel's website, and I also explain them in more detail in my SQL Server Hardware book.

For an OLTP workload on a 2-socket server, the Xeon X5690 processor is currently "best in class" and is the one I recommend in the 2010 to late-2011 timeframe (at which point, look out for the Sandy Bridge-EP Tock release (see Tip 20).

The Xeon X5690 is part of the Intel Xeon 5600 series (Westmere-EP) of 4- or 6-core processors. This processor series is based on the Nehalem microarchitecture, and it has both hyper-threading technology and turbo boost capability. It is built on 32 nm process technology, and has clock speeds ranging from 1.87 GHz to 3.46 GHz with an L3 cache size of 12 MB. QPI (Quick Path Interconnect) bandwidth ranges from 5.86 GT/s to 6.4 GT/s. It uses Socket LGA 1366, and the Intel 5500 or 5520 chipset. It has an improved memory controller and a larger L3 cache compared to the Xeon 5500 series.

The X5690 is the top-of-the-range processor in this series and has a base clock speed of 3.46 GHz, with a max turbo frequency of 3.73 GHz. This processor has six physical cores, plus hyper-threading, so that you get 12 logical cores (which are visible to the operating system) for each physical processor. My analysis of published TPC-E benchmark scores confirms that the X5690 is Intel's highest performance processor for single-threaded workloads, which means that it is especially suited for OLTP workloads.

X = Performance

The alpha prefix in the classic Xeon number system indicates electrical power usage and performance, as follows: **Xnnnn** *= Performance,* **Ennnn** *= Mainstream,* **Lnnnn** *= Power-Optimized. Always choose X models for database work. The additional cost, over that of an E series, is minimal compared to the overall hardware and SQL Server license cost of a database server system. Also, avoid the power-optimized L series,* since they can reduce processor performance *by 20–30% while only saving 20–30 watts of power per processor, which is pretty insignificant compared to the overall electrical power usage of a typical database server.*

If you are running a smaller OLTP workload, on a single-socket server, I'd choose the Xeon E3-1280 (32 nm Sandy Bridge). If you are using a 4-socket (or more) server, I'd move to the E7 series, choosing the Xeon E7-4870 (32 nm Westmere-EX) for 4-socket servers and the Xeon E7-8870 (32 nm Westmere-EX) for 8-socket. These E7 processors are also suitable for DSS/DW workloads – see Tip 19 for further details.

Lest you accuse me of bias, I will also mention the best AMD offering, which is the **Opteron 6180 SE**. In all honesty, however, I believe you are better off with an Intel processor for SQL Server OLTP workloads. The Intel Xeon X5690 will simply smoke the Opteron 6180 SE in single-threaded performance. However, the Opteron is much more competitive for multi-threaded performance in the DSS/DW space (see Tip 19).

Another factor to consider is the number of Intel-based systems that have been submitted and approved for the TPC-E OLTP benchmark, compared to how few AMD based systems have been submitted and accepted. As of January 2011, there were 37 Intel-based TPC-E results compared to four AMD-based TPC-E results. I don't think this is an accident.

In Q3 2011, AMD will release their new "Bulldozer" range of processors (see Tip 20). If they live up to their advanced billing, it will make AMD much more competitive for OLTP workloads, which can only be a good thing. Watch this space.

19 For DSS/DW server workloads, consider AMD Opteron 6180 SE or Intel E7 Series (July 2011)

In the x86/x64 space, AMD has various versions of the Opteron family. Recent AMD Opteron processors are identified by a four-digit model number in the format ZYXX, where Z indicates the product series (e.g. 8YXX is up to 8 sockets), Y denotes number of cores (e.g. 82XX = 2 cores, 83XX = 4 cores, 84XX = 6 cores) and XX denotes the various models in the series. There is also a two-digit product suffix after the model number, that denotes performance characteristics. For database work, you should always opt for the "SE" prefix, meaning performance-optimized. For a full explanation of AMD processor numbering, please refer to the AMD website.

The most interesting series, from the perspective of supporting DW/DSS workloads, is the AMD Opteron 6100 Magny-Cours and, in particular, the 12-core **Opteron 6180 SE**, which is their new, top-of-the-line model for 2- or 4-socket servers.

The Opteron 6160 SE is built on 45 nm process technology and has a clock speed of 2.5 GHz. Each core has a 512 K L2 cache; while there is a 12 MB shared L3 cache. It is basically two six-core "Istanbul" processor dies combined in a single package, with improved, dual DDR3 memory controllers and improved hyper-threading connections between the processor dies. Each processor die has its own memory controller, so there are four DDR3 SDRAM memory channels. It requires a new socket called Socket G34.

The Magny-Cours Opteron 6160 SE is AMD's best-performing processor in the 2010 to mid-2011 timeframe. Its twelve physical cores per processor, without use of hyper-threading to produce logical cores, make it well-suited to multi-threaded workloads (such as reporting, data warehouses, and so on), where you often have long-running, complex queries that are likely to be parallelized by the query optimizer.

For 64-bit platforms, Intel has recently launched a new range of Westmere-EX processors, called the **Intel Xeon E7 series**, with separate model numbers for 2-, 4-, and 8-socket servers. With their high core counts (up to ten cores, plus hyper-threading in most models) and large L3 caches sizes (18–30 MB), they are an interesting, if considerably more expensive option when supporting DSS/DW workloads (in mid-2011, the Intel E7 processors were roughly three times more expensive than the AMD Opteron 61xx processors).

The Intel Xeon E7 processors have four QPI links and two memory controllers, which each have two dual-channel interfaces per memory controller. The new memory controllers support 32 GB DIMMs and low-power memory modules. This means that a 4-socket system could support up to 2 TB of RAM, which is the current operating system limit for Windows Server 2008 R2. Of course, you will need pretty deep pockets to reach this limit, because 32 GB DIMMs will be very expensive when they are first available (my guess is around $4,000 each, at least initially).

Using the on-board memory buffer, the E7 processors can run DDR3-1333 memory at data rates of 800, 978 and 1066 MHz. The E7 processors will support AES instructions, Trusted Execution Technology, and Virtualization VT-x, VT-d and VT-c.

Intel claims up to 40% better database performance for the top E7-4870 4-socket model, in comparison to the previous generation Xeon X7560 model. Performance of the E7-4870 CPU in integer and floating-point applications is better than the X7560 by up to 22% and 19% respectively. The E7 processors will be socket compatible with the current Xeon 7500 processors, which means that existing systems from your favorite vendor should be able to use them as soon as they are available.

The E7-4850 and above (E7-2850 and above in the 2-socket range) have 10 cores, plus hyper-threading, so a 4-socket server could have 80 logical cores. Don't forget, however, that you need to have SQL Server 2008 R2 running on top of Windows Server 2008 R2 in order to use more than 64 logical cores.

20 Keep track of the major processor vendors

We've already mentioned several of the best available processor models in previous tips, but it's important to keep a close eye on new releases and innovations by the major server processor vendors, namely Intel and AMD, as hardware evolves at a reasonably rapid pace.

Since 2006, **Intel** has adopted a very structured **Tick-Tock** strategy for developing and releasing new processor models. Every two years, they introduce a new processor family, incorporating a new microarchitecture; this is the **Tock** release. One year after the Tock release, they introduce a new processor family that uses the same microarchitecture as the previous year's Tock release, but using a smaller manufacturing process technology and usually incorporating other improvements, such as larger cache sizes or improved memory controllers. This is the **Tick** release.

This Tick-Tock release strategy benefits the DBA in a number of ways. It offers better predictability regarding when major (Tock) and minor (Tick) releases will be available. This helps the DBA plan upgrades. Tick releases are usually socket-compatible with the previous year's Tock release, which makes it easier for the system manufacturer to make the latest Tick release processor available in existing server models quickly, without completely redesigning the system. In most cases, only a BIOS update is required to allow an existing system to use a newer Tick release processor. This makes it easier for the DBA to maintain servers that are using the same model number (such as a Dell PowerEdge R710 server), since the server model will have a longer manufacturing life span.

As a DBA, you need to know where a particular processor falls in Intel's processor family tree if you want to be able to meaningfully compare the relative performance of two different processors. Historically, processor performance has nearly doubled with each new Tock release, while performance usually goes up by 20–25% with a Tick release.

Some of the recent Tick-Tock releases are shown in Figure 20-1.

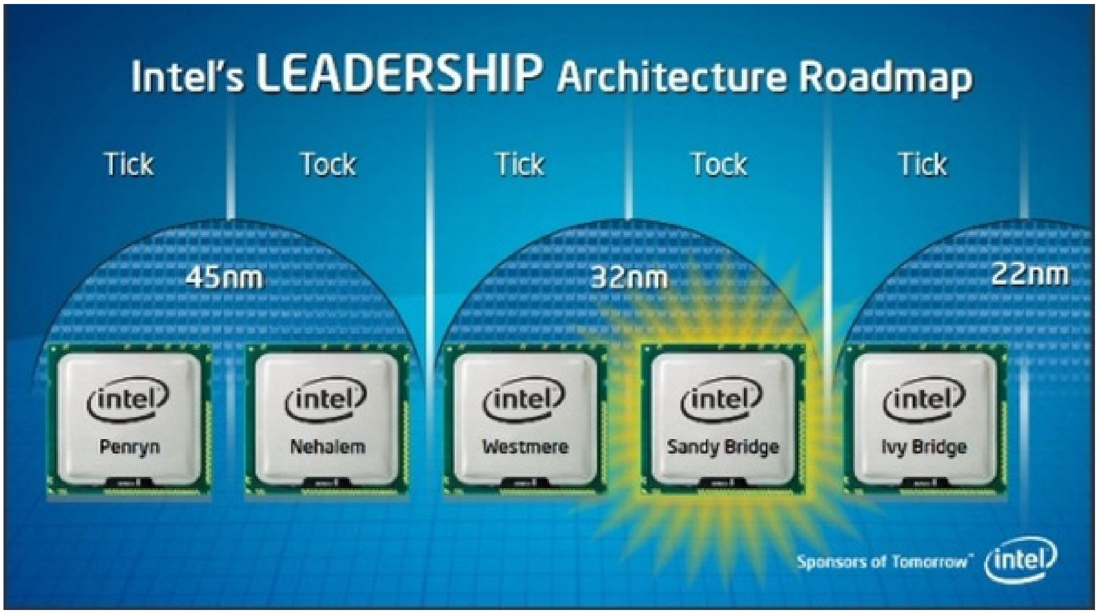


Figure 20-1: Intel's Tick-Tock release strategy.

In 2010, in the server space, Intel released a Tick release, code-named Westmere (XEON 5600 SERIES) that was an improved version of the Nehalem architecture with a die shrink to 32 nm process technology. In 2011, the **Sandy Bridge** Tock release debuted with the E3-1200 SERIES for single-socket servers and workstations. All of these other examples are for 2-socket servers, but Intel uses Tick Tock for all of their processors.

AMD's release cycle is a little less structured and, as discussed in TIP 18, over recent years they have become a little less competitive in regard to single-threaded performance, though their Opteron 6100 Magny-Cours series is still a good choice for DSS/DW workloads.

However, this may all change with the upcoming AMD BULLDOZER family of processors, for release in Q3 2011. This will be a 32 nm processor that will be used in both desktop systems and server systems. It will initially have up to 16 cores in the server version, using a design that is a hybrid between conventional, separate physical cores (like the AMD Opteron 6100 Magny-Cours) and Intel hyper-threading. A block diagram of a Bulldozer module is shown in Figure 20-2.

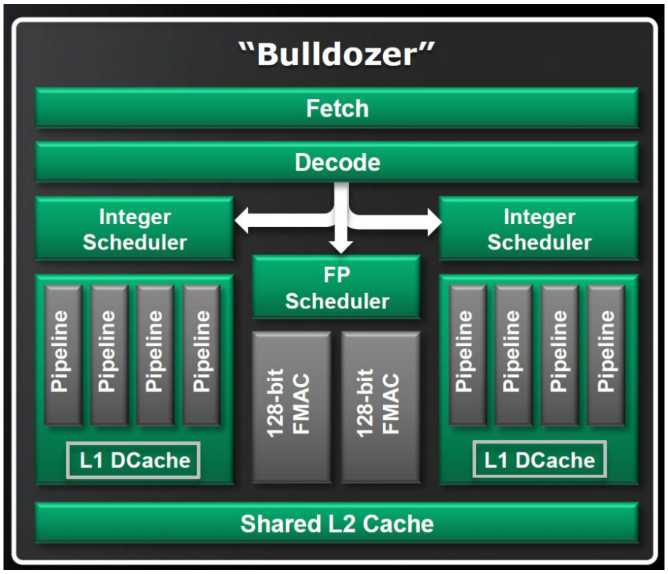


Figure 20-2: AMD Bulldozer module.

A single BULLDOZER processor would have eight of these modules. Each module has 2 MB of shared L2 cache, with up to 16 MB of L3 cache that is shared between all of the modules in the physical CPU. This processor will also support the TURBO CORE feature (see TIP 7). They will be socket compatible with the current Opteron 6100 Magny-Cours, using the same G34 socket, which will make it easier for system vendors to start using it.

I have high hopes for this processor, since I really want AMD to be a viable competitor to Intel from a performance perspective. If AMD cannot compete with Intel, we will all lose, since Intel will have little incentive to continue to innovate at a rapid pace.

Desktop and laptop processors and related hardware

Up to this point, we've focused solely on processors and hardware for server environments. However, it's also very important to make careful choices of hardware for your development and test systems, and for your work laptops, given that many DBAs, developers, and consultants use laptop computers as their primary workstations for working with SQL Server. The following tips offer my current advice for provisioning these systems.

21 $2,000 buys a very capable dev/test system

In many organizations, retired rack-mounted servers or workstations are repurposed as development and test servers. Quite often, these old machines are 3–5 years old (or even older) and their performance and scalability is quite miserable by today's standards.

For example, in the recent past I had a small testing lab populated with a number of 4–6 year-old servers, including:

- Dell PowerEdge 1850, with two Intel Xeon Irwindale 3.0 GHz processors and 8 GB of RAM (Geekbench score 2,250)
- Dell PowerEdge 6800 with four Xeon 7140M 3.4 GHz processors and 64 GB of RAM (Geekbench score 5,023).

By comparison, my current main workstation has an Intel Core i7 930 processor with 12 GB of RAM and a Crucial C300 128 GB SSD. This relatively humble system has a Geekbench score of around 7,300.

In many cases, I think it makes more sense to build or buy a new desktop system, based on a modern platform, rather than using relatively ancient "real" server hardware. You should be able to build or buy a good test system for less than $2,000, perhaps far less, depending on how you configure it, and it would be capable of supporting several virtual machines, all running SQL Server.

The main limiting factors, when buying a new desktop system, will be I/O capacity (throughput and IOPS) and memory capacity. However, by careful choice of processor and motherboard, with a decent number of memory and expansion slots for SATA (Serial Advanced Technology Attachment) drives, plus use of consumer-grade Solid State Drives, it is possible to neutralize these issues.

My current choice for a desktop system for testing or development would be either an Intel Socket 1366 or Socket 1155 platform.

Intel Socket 1366

This platform is a little older, but it does have six DDR3 memory slots, so it can support 24 GB of RAM, using 4 GB DDR3 RAM sticks, which should be sufficient for a number of concurrent virtual machines. You want to get a motherboard that has as many SATA ports as possible (preferably newer 6 Gbps SATA III ports), with hardware RAID support if possible. The newer 1366 motherboards usually have two 6 Gbps SATA III ports (that unfortunately use a slower Marvell controller), and a couple of USB 3.0 ports.

There are two likely processors that I would choose for a 1366 motherboard: the Intel Core i7-970 or the Core i7-960. Either one of these processors would work well in one of the newer Socket 1366 motherboards, such as this ASUS Sabertooth.

The 3.2 GHz 32 nm Intel Core i7-970 "Gulftown" ($579.99 at NewEgg) is the more expensive of the two options. This CPU has six cores, plus hyper-threading, so it has 12 logical cores. It is relatively affordable for a Gulftown CPU (compared to around $1,000 for a Core i7-990X), but it does not offer twice the capacity of the Core i7-960, even though it costs twice as much. It is the desktop equivalent of an Intel Xeon W3670.

The more affordable one is the 3.2 GHz 45 nm Intel Core i7-960 "Bloomfield" ($284.99 at NewEgg). This CPU has four cores, plus hyper-threading, so it has eight logical cores. I would argue that it will give you much more bang for the buck compared to the Core i7-970. It is the desktop equivalent of an Intel Xeon W3570.

Intel Socket 1155 platform

This is the newer, desktop Sandy Bridge platform. It only has four memory slots, so it can currently support 16 GB of RAM (with 4 GB DDR3 RAM sticks), or a maximum of 32 GB of RAM (if you can find 8 GB sticks and are prepared to pay for them). In any event, 16 GB should be plenty of RAM to run for three or four concurrent virtual machines.

The Socket 1155 motherboards have two native 6 Gbps SATA III ports, plus two more Marvell 6 Gbps SATA III ports. They also typically have six 3 Gbps SATA II ports, usually with hardware RAID support. A good example is this ASUS P8Z68-V motherboard.

The best processor for this platform right now is the 3.4 GHz 32 nm Intel Core i7-2600K, Sandy Bridge ($314.99 at NewEgg). It has four cores, plus hyper-threading for a total of eight logical cores. It uses the newer Turbo Boost 2.0 to go up to 3.8 GHz on individual cores. It is the desktop equivalent of the Intel Xeon E3-1275. A Core i7-2600K processor will have roughly equivalent CPU performance to a Core i7-970, for about half the cost. It will also use less electrical power, and run cooler. The Sandy Bridge processors have pretty decent integrated graphics that are more than sufficient for a desktop "server" machine. Depending on which motherboard chipset you select (either H67 or Z68 based), you can choose to use the integrated graphics instead of a discrete graphics card. This saves electrical consumption and reduces your hardware cost, but the integrated graphics will use a little bit of your available RAM.

The new Z68 chipset is somewhat of a hybrid between the P67 and the H67 platform controller hubs (PCHs) and was designed to allow overclocking Sandy Bridge CPUs while using the integrated graphics. In addition, the Z68 chipset supports Intel Smart Response Technology (SRT), allowing you to use a small (64 GB or less), inexpensive SSD as a caching layer in front of your conventional hard drive. The idea is that you install Windows 7 or Windows Server 2008 R2 on the conventional hard drive, and then use Intel's Rapid Storage Technology application to ENABLE CACHING FOR THE SSD. The Rapid Storage driver will write data to the SSD the first time it is read from the conventional hard drive. Then, the next time the system needs that data, it will look first in the SSD, and hopefully find it there, for a nice speed boost.

Possible Sandy Bridge driver issues

Depending on your motherboard vendor, you might run into Sandy Bridge driver issues with Windows Server 2008 R2. The problem is not that there are no drivers, but the fact that the motherboard vendors sometimes wrap the actual driver installation programs in their own installation programs which do OS version checking that fails with Windows Server 2008 R2 (since they assume you will be using Windows 7).

Storage choices

If you are going to build a system from parts (which is really not very hard), then you will need a case, a power supply, and an optical drive. First, buy a large, full tower case, with lots of internal 3.5" drive bays. Then, depending on your space requirements, performance needs and budget, you can fill the bays with a mixture of traditional magnetic and SSD storage.

For example, you might start with a relatively small and affordable SSD boot drive combined with several large traditional hard drives. I really like the larger Western Digital Black SATA drives, such as the 2 TB WD2002FAEX. However, make sure to get the 6 Gbps models with the 64 MB cache instead of the older 3 Gbps models with a 32 MB cache. This will let you have a pretty decent amount of I/O capacity for a relatively low cost.

SSDs are becoming more affordable, but they are still pretty expensive compared to conventional rotating disk drives. On the other hand, they offer much better I/O performance, which is very important for virtual machines. The new generation 6 Gbps SATA III SSDs offer much better throughput performance than the older 3 Gbps SATA II SSDs, especially when they are plugged into a 6 Gbps SATA III port, but any SSD is going to offer excellent performance compared to a magnetic drive. The larger-capacity, (and more expensive) SSDs perform better than the lower-capacity models in the same line because they use more flash memory chips with more internal I/O channels. If you have money to burn, you can get a 480 GB OCZ Vertex 3 SSD ($1,199.00 at NewEgg). For about half the price, you can get a 240 GB OCZ Vertex 3 Max IOPS SSD. For less than half that price again, you can pick up a 128 GB Crucial RealSSD C300 SSD.

22 For a laptop, choose a Sandy Bridge processor (July 2011)

Many DBAs, developers, and consultants use laptop computers as their primary workstations when working with SQL Server. My earlier advice regarding database servers, that you tend to be stuck with your choice of processor for the life of the machine, is doubly true of laptops, unless you are pretty brave and willing to do some major surgery.

Making the wrong choice could mean that you have a lot less processing power, or a lot less battery life, than you might have expected. Once again, don't economize on the processor. It is usually much better to economize, if necessary, by ordering a laptop with a relatively modest amount of RAM, and perhaps a smaller, slower hard drive, and then buying more RAM and a faster hard drive, or SSD, at a later date (newegg.com is a good place to try). This also gives you a chance to do a fresh install of the operating system on a new hard drive or SSD, leaving the original drive intact.

Unfortunately, you cannot usually rely on the sales clerk at Best Buy to give you good advice about which processor to pick for your new laptop. Right now, I strongly recommend that you get a **Sandy Bridge** processor in your new laptop. Now that the storm has settled with regard to the reported issue with Sandy Bridge processors and certain chipsets (which had absolutely nothing to do with the processor itself), lots of Sandy Bridge-based laptops are finally available for purchase.

Sandy Bridge-equipped machines will have significantly better performance and battery life than their 2010 vintage predecessors. Also, the processors have native 6 Gbps SATA III support, which means that you can take advantage of the fastest 6 Gbps SSDs. You will also get USB 3.0 ports, which are a huge improvement over USB 2.0 ports (which are usually limited to about 25–30 MB/sec throughput).

Figure 22-1 shows the new Sandy Bridge logo, which will help you spot Sandy Bridge machines in a store.

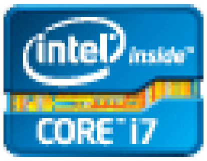


Figure 22-1: New Sandy Bridge logo.

The various ranges of the second-generation Intel Core processors are listed below and come with various clock speeds and cache sizes:

- Core i3 – dual-core plus hyper-threading, no turbo boost; worth avoiding
- Core i5 – quad-core, no hyper-threading, with turbo boost
- Core i7 – quad-core, hyper-threading, turbo boost.

Table 22-1 shows five processors from the i5 and i7 ranges, with their price, my *estimate* of their relative CPU performance, and a simple calculation of the "price per performance."

Model	Price	Relative Performance	$/Perf
Core i5-2520M	$225	100	2.25
Core i5-2540M	$266	104	2.56
Core i7-2720QM	$378	176	2.14
Core i7-2820QM	$568	202	2.81
Core i7-2920XM	$1,096	220	4.98

Table 22-1: Price-per-performance-point calculation for new Sandy Bridge processors.

For example, the Core i5-2520M has a base clock speed of 2.5GHz, while the Core i5-2540M has a base clock speed of 2.6 GHz. Both processors have the same size L3 cache, and their relative Turbo Boost speeds only differ by 100 MHz. Finally, both processors have the same DDR3 memory speed of 1333 MHz. These two processors are basically identical, except for a 100 MHz clock speed difference. Giving the Core i5-2520M an arbitrary baseline performance score of 100, my paper analysis would give the Core i5-2540M a 4% increase due to the 100 MHz increase in clock speed, so the Core i5-2540M gets a performance score of 104.

According to this analysis, the sweet spot for price/performance is definitely the entry level quad-core, Core i7-2720QM with a $2.14/performance point figure. The top-of-the-line Core i7-2920XM, which is an Extreme Edition part, is not a good deal at all. You would be far better off to spend that extra money on a fast 6 Gbps SSD.

Recently, a reader posted a GEEKBENCH SCORE of 11,052 for a CLEVO W150HNM with an INTEL CORE I7 2820QM processor, which is pretty impressive. According to this score, this laptop has more CPU horsepower than a relatively recent vintage (early 2008) 4-socket database server equipped with four INTEL XEON X7350 processors!

For more comparison results, the Geekbench blog has a POST with a number of results from different models of the MacBook Pro, going back to the early 2010 models. You can use this to get a rough idea of how much better a Sandy Bridge based machine (Mac or PC) will perform compared to various older processors.

Configuring the disk subsystem

RAM capacity has increased unremittingly over the years and its cost has decreased enough to allow us to be lavish in its use for SQL Server, to help minimize disk I/O. Also, CPU speed has increased to the point where many systems have substantial spare capacity that can often be used to implement data compression and backup compression, again, to help reduce I/O pressure. The common factor here is "helping to reduce disk I/O." While disk capacity has improved greatly, disk speed has not, and this poses a great problem; most large, busy OLTP systems end up running into I/O bottlenecks.

Just as there are many factors to consider when choosing an appropriate processor and associated chipsets, there are equally as many considerations when sizing and configuring the storage subsystem. It is very easy to hamstring an otherwise powerful system through poor storage choices. Important factors, discussed in this chapter, include:

- disk seek time and rotational latency limitations
- type of disk drive used:
  - traditional magnetic drive – SATA, SCSI, SAS and so on
  - Solid State Drives (SSDs)
- storage array type: Storage Area Network (SAN) vs. Direct Attached Storage (DAS)
- RAID configuration of your disks.

The following tips offer some advice on how to provision and configure your disk subsystem, and how the size and nature of the workload will influence your choices. All of these issues, and more, are discussed in greater depth in my SQL Server Hardware book.

23 Know your workload type, Part 2

As discussed in Tip 1, the nature of your workload will have a huge impact on the hardware choices you make. This is just as true for the disk subsystem as it was for processors.

The number, size, speed, and configuration of the disks that comprise your storage array will be heavily dependent on the size and nature of the workload. Every time that data required by an application or query is not found in the buffer cache, it will need to be read from the data files on disk, causing read I/O. Every time data is modified by an application, the transaction details are written to the transaction log file, and then the data itself is written to the data files on disk, causing write I/O in each case.

In addition to the general read and write I/O generated by applications that access SQL Server, additional I/O load will be created by other system and maintenance activities such as transaction log backups, database checkpoints, index maintenance, full text search, log shipping, database mirroring, and so on.

The number of disks that make up your storage array, their specifications in terms of size, speed and so on, and the physical configuration of these drives in the storage array, will be determined by the size of the I/O load that your system needs to support, both in terms of IOPS and I/O throughput, as well as in the nature of that load, in terms of the read I/O and write I/O activity that it generates.

A primarily OLTP workload will generate a high number of I/O operations (IOPS) and a high percentage of write activity; it is not that unusual to actually have more writes than reads in a heavy OLTP system. This will cause heavy write (and read) IO pressure on the logical drive(s) that house your data files and, particularly, heavy write pressure on the logical drive where your transaction log is located, since every write must go to the transaction log first. The drives that house these files must be sized, spec'd and configured appropriately, to handle this pressure.

In most OLTP databases, the read/write activity is largely random, meaning that each transaction will likely require data from a different part of the disk. All of this means that in most OLTP applications, the hard disks will spend most of their time seeking data, and so the **seek time** of the disk is a crucial bottleneck for an OLTP workload. The seek time for any given disk is determined by how far away from the required data the disk heads are at the time of the read/write request. Generally speaking, OLTP systems are characterized by lots of fast disks, to maximize IOPS to overcome disk latency issues with high numbers of random reads and writes.

Finally, almost all of the other factors that cause additional I/O pressure, mentioned previously, are more prominent for OLTP systems than for DSS/DW systems. High write activity, caused by frequent data modifications, leads to more regular transaction log backups, index maintenance, more frequent database checkpoints, and so on.

A DSS or DW system is usually characterized by longer-running queries than a similar-sized OLTP system. The data in a DSS system is usually more static, with much higher read activity than write activity. The disk activity with a DSS workload also tends to be more sequential and less random than with an OLTP workload. Therefore, for a DSS type of workload, sequential I/O throughput is usually more important than IOPS performance.

Adding more disks will increase your sequential throughput until you run into the throughput limitations of the I/O channels in your system (i.e. the RAID controllers in a DAS array, or HBAs in fiber channel SAN or Ethernet cards in an iSCSI SAN, which provide dedicated, separate paths to the storage). This is especially true when a DSS/DW system is being loaded with data, and when certain types of complex, long-running queries are executed.

In general, DW/DSS systems require lots of I/O channels, in order to handle peak sequential throughput demands. The more I/O channels you have, the better.

In summary, it is very important that you understand the nature of your workload, using, for example, the query shown in Listing 1-1, to determine the read/write ratio, when provisioning your disk subsystem.

24 Attached storage: for magnetic drives choose SAS or SATA

All modern blade and rack-mounted database servers will come with a number of internal drive bays where you can mount 2.5" drives (usually SAS drives). The first important point to make here is that, for all but very lightest database workloads, you simply **won't have enough internal drive bays to completely support your I/O requirements**.

Ignoring this reality is a very common mistake that I see made by many DBAs and companies. They buy a new, high-performance database server with fast, multi-core processors and lots of RAM, and then try to run an OLTP workload on six internal drives. This is like a body-builder who only works his upper body, but completely neglects his legs, ending up completely out of balance, and ultimately not very strong. You will need some external, attached storage, and database servers have traditionally used magnetic hard drive storage, though I expect SSDs to start becoming more prominent.

Seek times for traditional magnetic hard disks have not improved appreciably in recent years, and are unlikely to improve much in the future, since they are electro-mechanical in nature. Typical seek times for modern hard drives are in the 5–10 ms range. The rotational latency for magnetic hard disks is directly related to the rotation speed of the drive. The current upper limit for rotational speed is 15,000 rpm, and this limit has not changed in many years. Typical rotational latency times for 15,000 rpm drives are in the 3–4 ms range.

Disks are categorized according to the type of interface they use and, in a modern server, you'll want to be using either SATA drives or SAS drives. Older, parallel SCSI drives are still common, but you should **not** be using PATA or IDE drives for any serious database server use. They are limited to two drives per controller, one of which is the master and the other the slave, and offer very low levels of throughput.

Server-class magnetic hard drives have rotation speeds ranging from 7,200 rpm (for SATA) to either 10,000 rpm or 15,000 rpm (for SCSI and SAS). Higher rotation speeds reduce data access time by reducing the rotational latency. Drives with higher rotation speed are more expensive, and often have lower capacity sizes compared to slower rotation speed drives. Over the last several years disk buffer cache sizes have grown from 2 MB all the way to 64 MB. Larger disk buffers usually improve the performance of individual magnetic hard drives, but often are not as important when the drive is used by a RAID array or is part of a SAN, since the RAID controller or SAN will have its own, much larger cache that is used to cache data from multiple drives in the array.

25 Start planning to incorporate SSDs

Solid State Drives (SSDs), and Enterprise Flash Disks (EFDs), are different from traditional magnetic drives in that they have no spinning platter, drive actuator, or any other moving parts. Instead, they use flash memory, along with a storage processor, controller and some cache memory, to store information.

The lack of moving parts eliminates the rotational latency and seek-time delay that is inherent in a traditional magnetic hard drive. Depending on the type of flash memory, and the technology and implementation of the controller, SSDs can offer dramatically better performance compared to even the fastest Enterprise-class magnetic hard drives. This performance does come at a much higher cost per gigabyte, and it is still somewhat unusual for database servers, DAS or SANs, to exclusively use SSD storage, but this will change as SSD costs continue to decrease.

SSDs perform particularly well for random access reads and writes (typical of OLTP workloads), and for sequential access reads. Some earlier SSDs do not perform as well for sequential access writes, and they also have had issues where write performance declines over time, particularly as the drive fills up. Newer SSD drives, with better controllers and improved firmware, have mitigated these earlier problems.

Depending on your database size and your budget, it may make sense to move the entire database to solid state storage, especially with a heavy OLTP workload. For example, if your database(s) are relatively small, and your budget is relatively large, it may be feasible to have your data files, your log files, and your `TempDB` files all running on SSD storage.

If your database is very large, and your hardware budget is relatively small, you may have to be more selective about which components can be moved to SSD storage. For example, it may make sense to move your `TempDB` files to SSD storage if your `TempDB` is experiencing I/O bottlenecks. Another possibility would be to move some of your most heavily accessed tables and indexes to a new data file, in a separate file group that would be located on your SSD storage.

26 Use the correct RAID level, ideally RAID 10

Redundant Array of Independent Disks (RAID) is a technology that allows use of multiple hard drives, combined in various ways, to improve redundancy, availability and performance, depending on the RAID level used. When a RAID array is presented to a host in Windows, it is called a logical drive.

Using RAID, the data is distributed across multiple disks in order to:

- overcome the I/O bottleneck of a single disk, as described previously
- get protection from data loss through the redundant storage of data on multiple disks
- avoid any one hard drive being a single point of failure
- manage multiple drives more effectively.

Regardless of whether you are using traditional magnetic hard-drive storage or newer solid-state storage technology, most database servers will employ RAID technology. RAID improves redundancy, improves performance, and makes it possible to have larger logical drives. RAID is used for both OLTP and DW workloads. Having more spindles in a RAID array helps both IOPS and throughput, although ultimately throughput can be limited by a RAID controller or HBA (see Tip 28).

Wherever your SQL Server data files, log files, TempDB files, and SQL Server backup files are located, they should be protected by an appropriate RAID level, depending on your budget and performance needs. We want to keep our databases from going down due to the loss of a single drive.

RAID is no substitute for SQL Server backups

Never, never, never let anyone, whether it is a SAN vendor, a server admin from your Operations team, or your boss, talk you into not doing SQL Server backups as appropriate for your Recovery Point Objective (RPO) and Recovery Time Objective (RTO) metrics. I cannot emphasize this point enough! Regardless of any pressure to which you are subjected, you must stand your ground. As the old saying goes, "If you don't have backups, you don't have a database."

There are a number of commercially-available RAID configurations, which we can't review in depth here; they are covered in my SQL Server Hardware book, as well as on Wikipedia, and elsewhere. RAID 0 should be avoided for database work, as it offers no redundancy at all (if any disk in a RAID 0 array fails, the array is offline and all of the data in the array is lost). RAID 1 (disk mirroring) is used in a limited capacity, for example, for storing the SQL Server binaries, but the most common levels used to protect data and log files, etc., are RAID 5 (striping with parity data) and RAID 10.

When considering which level of RAID to use for different SQL Server components, you have to carefully consider your workload characteristics, keeping in mind your hardware budget.

In RAID 10 (striped set of mirrors), the data is first mirrored and then striped. This configuration offers high read/write performance, plus high fault tolerance; it is possible to survive the loss of multiple drives in the array (one from each side of the mirror), while still leaving the system operational. However, it incurs a roughly 100% storage cost penalty.

If cost is no object, I prefer to use RAID 10 for everything, i.e. data files, log file, and `TempDB`.

In RAID 5, the data and calculated parity information is striped across the physical disks by the RAID controller. This provides redundancy because, if one of the disks goes down, then the missing data from that disk can be reconstructed from the parity information on the other disks. Also, rather than losing 50% of your storage, in order to achieve redundancy, as for disk mirroring, you only lose 1/N of your disk space (where N equals the number of disks in the RAID 5 array) for storing the parity information. For example, if you had six disks in a RAID 5 array, you would lose ⅙ of your space for the parity information. However, you will notice a very significant decrease in performance while you are missing a disk in a RAID 5 array, since the RAID controller has to work pretty hard to reconstruct the missing data.

Also, there is a write performance penalty with RAID 5, since there is overhead to write the data, and then to calculate and write the parity information. As such, RAID 5 is usually not a good choice for transaction log drives, where we need very high write performance. I would also not want to use RAID 5 for data files where I am changing more than 10% of the data each day. One good candidate for RAID 5 is your SQL Server backup files. You can still get pretty good backup performance with RAID 5 volumes, especially if you use backup compression.

27 High-performance DAS is a reality

This disk latency limitation discussed in Tip 23, and the need to optimize storage capacity, has led to the proliferation of vast Storage Area Networks (SANs), allowing data to be striped across numerous magnetic disks, and leading to greatly enhanced IO throughput.

A SAN is a dedicated network that has multiple hard drives (anywhere from dozens to hundreds of drives) with multiple storage processors, caches, and other redundant components. Multiple database servers can share a single, large SAN (as long as you don't exceed the overall capacity of the SAN), and most SANs offer features that are not available with Direct Attached Storage (DAS), such as SAN snapshots.

Many larger SANs have a huge number of drive spindles, so this architecture is able to support a very high rate of random input/output operations per second (IOPS). Use of a SAN for the I/O subsystem in OLTP workloads makes it relatively easy (but expensive) to support dozens to hundreds of disk spindles for a single database server. The general guideline is that you will get roughly 100 IOPS from a single 10,000 rpm magnetic drive and about 150 IOPS from a single 15,000 rpm drive. For example, if you had a SAN with two hundred 15,000 rpm drives, that would give the entire SAN a raw IOPS capacity of 30,000 IOPS. However, note that if the HBAs in your database server were the older, 4 Gbps models, then your sequential throughput would still be limited to roughly 400 MB/second for each HBA channel (see Tip 28).

With the additional expense of the SAN, you do get a lot more flexibility. The real problem with SANs, however, apart from their cost, is that they tend to be vast, complex, and often fault-prone. The fact that these SANs are generally shared by many databases adds even more complexity and often results in a disappointing performance, for the cost.

I predict that, slowly, the relevance of SANs will decrease over time, since it is now possible to build simpler, very high throughput DAS arrays that are especially applicable to DSS/DW workloads, but will also satisfy most OLTP workload requirements.

DAS drives are directly attached to a server with an eSATA, SAS, or SCSI cable. Typically, you have an external enclosure, containing anywhere from 8–24 drives, attached to a RAID controller in a single database server. It is possible to get very good IOPS and I/O throughput performance by using multiple DAS enclosures with multiple RAID controllers (one per DAS enclosure).

When using DAS, it is very important to spread the I/O workload among the multiple logical drives that each represent a dedicated DAS enclosure. This means having one or more dedicated logical drives for SQL Server data files, a dedicated logical drive for the log file (for each user database, if possible), a dedicated logical drive for your `TempDB` data and log files, and one or more dedicated logical drives for your SQL Server backup files.

Of course your choices and flexibility are ultimately limited by the number of drives that you have available, which is limited by the number of DAS enclosures you have, and the number of drives in each enclosure.

Especially for DW/DSS systems, where I/O throughput is the most important factor, and the throughput of a SAN array can be limited to the throughput capacity of a switch or individual HBA, DAS is a good choice. With the use of multiple DAS devices, each on a dedicated RAID controller, we can get high levels of throughput at a relatively low cost.

28 Maximize spindle count and I/O channel performance

One common mistake that you should avoid in selecting storage components is to only consider space requirements when looking at sizing and capacity requirements. For example, if you had a size requirement of 1 TB for a drive array that was meant to hold a SQL Server data file, you could satisfy the size requirement by using 3 x 500 GB drives in a RAID 5 configuration. Unfortunately, for the reasons discussed relating to disk latency, the performance of that array would be quite low. A better solution from a performance perspective would be to use either 8 x 146 GB drives in RAID 5, or 15 x 73 GB drives in RAID 5 to satisfy the space requirement with many more spindles. You should always try to maximize your spindle count instead of just using the minimum number of drives to meet a size requirement with the level of RAID you are going to use.

A common limiting factor in determining overall I/O performance is the throughput capabilities of your I/O channels – namely RAID controllers for DAS, or Host Bus Adaptors (HBAs) for SANs. The throughput of such controllers, **usually measured in gigabits per second, e.g. 3 Gbps, will dictate the upper limit for how much data can be written or read from a disk per second. This can have** a huge effect on your overall IOPS and disk throughput capacity for each logical drive that is presented to your host server in Windows. As such, it is very important to choose and configure such controllers carefully.

In this discussion, we'll consider only the RAID controller, which is (to quote from Wikipedia):

> *"a device which manages the physical disk drives and presents them to the computer as logical units. It almost always implements hardware RAID, thus it is sometimes referred to as a RAID controller. It also often provides additional disk cache."*

Figure 28-1 shows a typical hardware RAID controller.

Figure 28-1: Typical hardware RAID controller.

For database server use (with recent vintage servers), you usually have an embedded hardware RAID controller on the motherboard, that is used for your internal SAS, SATA, or SCSI drives.

For a DAS setup, you will also have one or more (preferably at least two) hardware RAID controller cards that will look similar to the one in Figure 28-1. Each card goes into an available PCI-e expansion slot in your server, and then is connected by a relatively short cable to an external storage enclosure (such as you see in Figure 28-2).

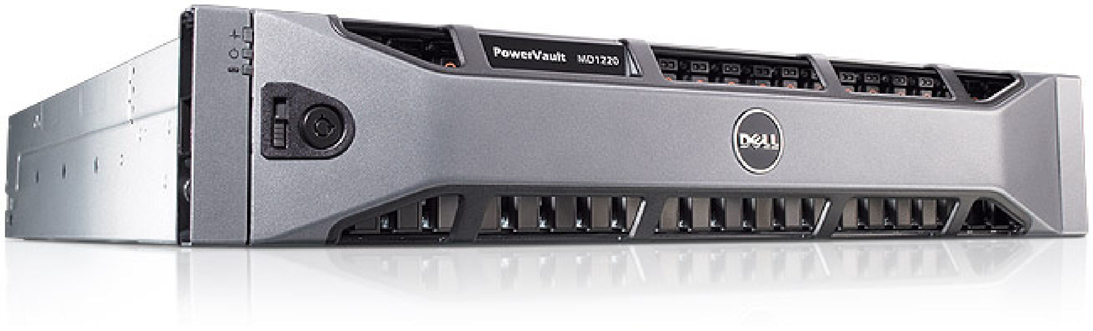


Figure 28-2: Dell PowerVault MD1220 DAS array.

Each DAS array will have anywhere from 14–24 drives. The RAID controller(s) are used to build and manage RAID arrays from these available drives, which are eventually presented to Windows as logical drives, usually with drive letters. For example, you could create a RAID 10 array with 16 drives and another RAID 10 array with 8 drives from a single 24 drive DAS array. These two RAID arrays would be presented to Windows, and show up as, say, the `L:` drive and the `R:` drive.

A server-class hardware RAID controller will have a dedicated, specialized processor that is used to calculate parity information; this will perform much better than using one of your CPUs for that purpose. Besides, your CPUs have more important work to do, so it is much better to offload that work to a dedicated RAID controller.

A server-class hardware RAID controller will also have a dedicated memory cache on the card itself, usually around 512 MB to 1 GB in size. The cache in a RAID controller stores data temporarily, so that whatever wrote that data to the cache can return to another task without having to wait to write the actual physical disk(s). This cache memory can be used to cache reads or to cache writes, or it can be split between both.

For SQL Server OLTP workloads, it is a standard best practice to devote your cache memory entirely to write caching. You can also choose between write-back and write-through cache policies for your controller cache. Write-back caching provides better performance, but there is a slight risk of having data in the cache that has not been written to the disk if the server fails. That is why it is very important to have a battery-backed cache if you decide to use write-back caching.

For both performance and redundancy reasons, you should always try to use multiple RAID controllers whenever possible. While most DAS enclosures will allow you to "daisy-chain" multiple enclosures on a single RAID controller, I would avoid this configuration if possible, since the RAID controller will be a single point of failure, and possibly a performance bottleneck as you approach the throughput limit of the controller.

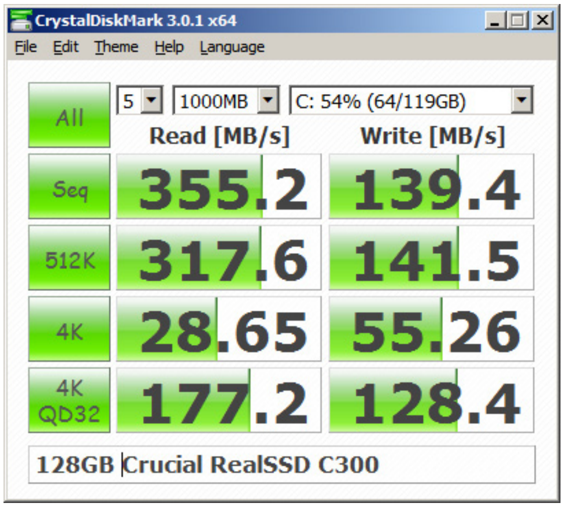
Instead, I would want to have one RAID controller per DAS array (subject to the number of PCI-E slots you have available in your server). This gives you both better redundancy and better performance. Having multiple RAID controllers allows you to take advantage of the dedicated cache in each RAID controller, and helps ensure that you are not limited by the throughput capacity of the single RAID controller or the expansion slot that it is using.

29 Use CrystalDiskMark to measure I/O capacity

CrystalDiskMark, available from Crystal Dew World is a widely used I/O component benchmark. You can select the number of test runs, desired file size, and which logical drive you are testing and it allows you to measure:

- **sequential** read and write performance in megabytes/second
- **random** read and write performance for a 512 K block size, a 4 K block size, and a 4 K block size with a queue depth of 32.

Figure 29-1 shows the results from a pretty decent consumer-grade 6 Gbps MLC SSD (the 128 GB Crucial RealSSD C300). It performs very well for both sequential and random reads and writes. Bear in mind that most consumer SSD drives perform better in their larger capacities (so a 256 GB drive will have better performance than a 128 GB drive of the same model line). You also want to make sure that you use a 6 Gbps SATA III port for a 6 Gbps SATA III SSD, or else performance will be sub-optimal.



CrystalDiskMark 3.0.1 x64

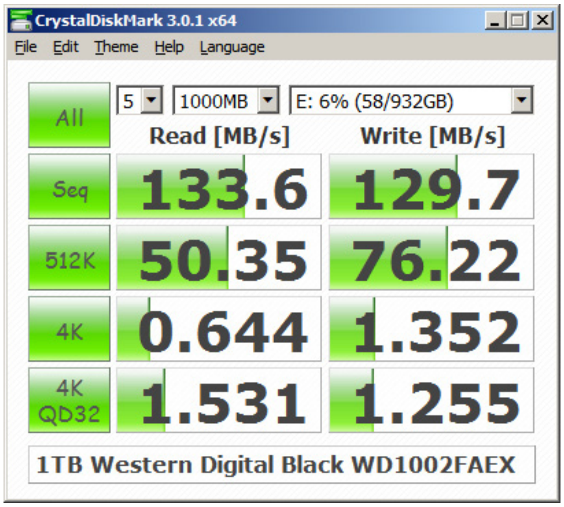
5 | 1000MB | C: 54% (64/119GB)

	Read [MB/s]	Write [MB/s]
Seq	355.2	139.4
512K	317.6	141.5
4K	28.65	55.26
4K QD32	177.2	128.4

128GB Crucial RealSSD C300

Figure 29-1: 128 GB Crucial RealSSD C300 SATA III SSD.

For comparison, Figure 29-2 shows the results for a modern, 7,200 rpm SATA drive, with a 64 MB cache.



5 | 1000MB | E: 6% (58/932GB)

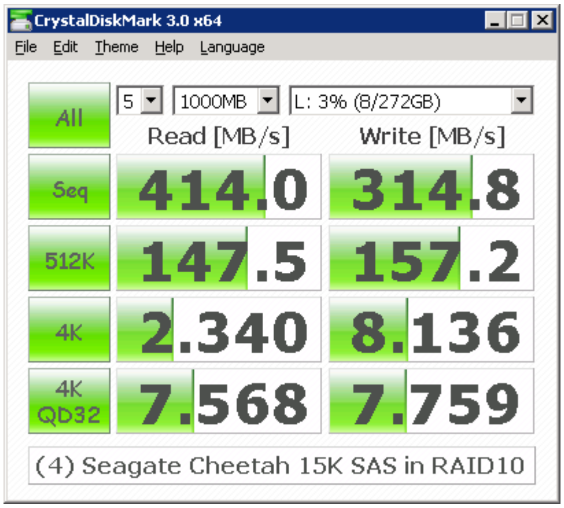
	Read [MB/s]	Write [MB/s]
Seq	133.6	129.7
512K	50.35	76.22
4K	0.644	1.352
4K QD32	1.531	1.255

1TB Western Digital Black WD1002FAEX

Figure 29-2: Western Digital Black WD1002FAEX SATA III hard drive.

It is dramatically out-performed by the SSD in all aspects. In particular, the scores for 4 K random reads and writes fall off pretty dramatically with traditional magnetic hard drives. This is an area where SSDs really do very well.

As another comparison point, Figure 29-3 shows the results for four 146 GB Seagate Cheetah 15 K SAS drives in a RAID10 configuration. Notice the 414.0 MB/s sequential read and 314.8 MB/s sequential write scores are rather good, but that random I/O performance is still low compared to a single consumer-grade Multi-Level Cell (MLC) SSD drive.



5 | 1000MB | L: 3% (8/272GB)

	Read [MB/s]	Write [MB/s]
Seq	414.0	314.8
512K	147.5	157.2
4K	2.340	8.136
4K QD32	7.568	7.759

(4) Seagate Cheetah 15K SAS in RAID10

Figure 29-3: CrystalDiskMark scores for four 15 K SAS drives in RAID10.

There are other tools available that will do a much more thorough job of benchmarking your disk subsystem, but they are generally not as easy to work with as CrystalDiskMark.

However, one in particular that is very valuable is SQLIO, and using CrystalDiskMark should certainly be a supplement to other disk benchmarking.

30 Build resilience into your system

In this final tip, I'll try to summarize some of the first steps you would take as part of designing a high availability solution for your data tier, in order to increase the resiliency and availability of an individual database server. The basic principle here is to try to eliminate as many single points of failure as possible, at the hardware level.

Two internal drives in a RAID 1 configuration for the operating system and SQL Server binaries. These drives should be using the embedded hardware RAID controller that is available on most new rack-mounted servers. I try to get at least 146 GB, 15 K 2.5" drives for this purpose. Using 15 K drives will help Windows Server boot a little faster, and will help SQL Server load a little faster when the service first starts up. Using 146 GB (or larger) drives will give you more room to accommodate things like SQL Server error log files, dump files, etc., without being worried about drive space.

Dual power supplies for the server, each plugged into separate circuits in your server room or datacenter. You should also be plugged in to an Uninterruptable Power Supply (UPS) on each circuit, and ideally have a backup power source, such as a diesel generator for your datacenter. The idea here is to protect against an internal power supply failure, a cord being kicked out of a plug, a circuit breaker tripping, or loss of electrical power from the utility grid.

Multiple network ports in the server, with Ethernet connections into at least two different network switches. These network switches should be plugged in to different electrical circuits in your datacenter. Most new rack-mounted servers have at least four gigabit Ethernet ports embedded on the motherboard.

Multiple RAID controller cards (if you are using DAS), or multiple Host Bus Adapters (HBAs) if you are using a fiber channel SAN, or multiple PCI-e Gigabit (or better) Ethernet cards with an iSCSI SAN. This will give you better redundancy and better throughput, depending on your configuration.

Minimize the boot time and SQL Server startup time for your database servers. Reducing your total reboot time for each server has a direct effect on your high availability numbers. I always go into the BIOS setup for the server, and **disable the memory testing** that normally occurs during the POST sequence. This will shave a significant amount of time off of the POST sequence (often many minutes), so the server will boot faster. I think this is pretty low risk, since this testing only occurs during the POST sequence. It has nothing to do with detecting a memory problem while the server is running later (which is the job of your hardware monitoring software). I am sure some people may disagree with this advice, so I would love to hear your opinions!

Appendix A: Recommended Intel and AMD processors

As of July 2011, the following lists summarize my recommendations for Intel and AMD processors, across the range of single- and multi-socket systems, and for a variety of workload types.

Intel

1-socket server (OLTP)

Xeon E3-1280 (32 nm Sandy Bridge)

- 3.50 GHz, 8 MB L3 Cache, 5.0 GT/s Intel QPI
- 4 cores, Turbo Boost 2.0 (3.90 GHz), hyper-threading
- 2 memory channels

1-socket server (DW/DSS)

Xeon W3690 (32 nm Westmere)

- 3.46 GHz, 12 MB L3 Cache, 6.40 GT/s Intel QPI
- 6 cores, Turbo Boost (3.73 GHz), hyper-threading
- 3 memory channels

2-socket server (OLTP)

Xeon X5690 (32 nm Westmere-EP)

- 3.46 GHz, 12 MB L3 Cache, 6.40 GT/s Intel QPI
- 6 cores, Turbo Boost (3.73 GHz), hyper-threading
- 3 memory channels

2-socket server (DW/DSS)

Xeon E7-2870 (32 nm Westmere-EX)

- 2.40 GHz, 30 MB L3 Cache, 6.40 GT/s Intel QPI
- 10 cores, Turbo Boost 2.0 (2.8 GHz), hyper-threading
- 4 memory channels

4-socket server (any workload type)

Xeon E7-4870 (32 nm Westmere-EX)

- 2.40 GHz, 30 MB L3 Cache, 6.40 GT/s Intel QPI
- 10 cores, Turbo Boost 2.0 (2.8 GHz), hyper-threading
- 4 memory channels

8-socket server (any workload type)

Xeon E7-8870 (32 nm Westmere-EX)

- 2.40 GHz, 30 MB L3 Cache, 6.40 GT/s Intel QPI
- 10 cores, Turbo Boost 2.0 (2.8 GHz), hyper-threading
- 4 memory channels

AMD

In all cases, I believe that AMD processors are currently better suited to DSS/DW work and that Intel processors are a better choice for OLTP.

1-socket or budget 2-socket server

- Opteron 4184 (45 nm Lisbon), 6 cores
- 2.8 GHz, 6 MB L3 cache, 6.4 GT/s

2- or 4-socket server

- Opteron 6180 SE (45 nm Magny-Cours), 12 cores
- 2.5 GHz, 12 MB L3 Cache, 6.4 GT/s