10306 W. Emerald Street
Boise, Idaho 83704
V: 208.343.8521
F: 208.343.8520

www.digitar.com
williamsjj@digitar.com

Cruisin' with a T2K

Thomas Rampelberg, Director of Development
&
Jason J. W. Williams, COO/CTO
DigiTar
2006-05-09

Everything that I heard about the Niagara processor and the T2000 specifically made us very excited. Because a majority of DigiTar's applications do not use much floating point, the T2000 was an opportunity for us to cheaply ameliorate potential scaling issues as well as markedly decrease power and heat problems in the data center.

The first time that I ever really paid much attention to the T2000 is when the local Sun sales representative pointed to a V440 and said that a T2000 could approximate the same performance. He suggested that I go to the website and simply sign up for the Try & Buy program going on. After ordering the 8 core (1.0 ghz) model with 8 GB of ram, I anxiously waited for it to arrive. Within a couple weeks, a shiny Sun box containing my trial model appeared. We were very surprised to get one so quickly, as the blogosphere had long been complaining about the wait times. Needless to say we were very pleased.

Unpacking of the T2000 was a very enjoyable experience. The chassis (identical in many aspects to a X4200) is well thought out. Everything has been labeled with pictures showing what to do for each part and the whole box can be taken apart rather quickly. After experiencing up close and personal the 10K rpm SAS drives that come with this box, I'd like to see more of my servers come with similar drives. They are wicked fast! Seemingly more so than their 3.5" counterparts. Perhaps the most striking part of all is when you open the top cover. It's extremely Spartan inside! Everything is clean and you have to wonder for a couple moments if any parts are missing.

I'm sure everyone that deals with rack mountable servers has some experience with different vendors' slide rails and the pain and suffering that you experience from them. Some vendors make the rails too flimsy…others give you one big piece of steel but make it impossible for the server to actually slide. I'd like to say that the T2000 rails are the best I've ever worked with, and definitely better than the HP rails that are the bane of the DigiTar data center. Not only do they feel solid and are easy to work with, but they fit very well with the general look of the box and make it a really finely finished product.

When I went to install the T2000, the first thing I noticed is that it is a headless server. That's a bit of change for a group running a largely Opteron data center. After

quickly looking through the installation instructions, I hunted down a console cable and had everything up and running in no time.

I'd like to point out a little observation that I had after turning the power on. First, that startup sound is really nifty…for a few seconds it gives a beautiful throaty exhaust note just like gunning a 6 liter Corvette a couple of times. Whoever thought that up was brilliant! Second, I've got the T2000 racked right above one of our HP DL145 G2s. Putting your hand behind the T2000, the exhaust air is pretty cool, definitely not much more than ambient. Conversely, if you put your hand behind the DL145 exhaust, it feels like a blast furnace. The difference is night and day.

Something that I'd like to acknowledge upfront is the previous experience I've had with Solaris 10 coming into this trial. It can really be summed up in one word – none. We're a Linux shop and therefore learning all the little differences between Solaris and Linux was a bit frustrating. Everything was just similar enough to get completely in the way.

Solaris 10 comes pre-installed with the T2000s and after answering a couple configuration questions, I was able to play around some. Being someone that likes to test the bleeding edge and seeing an opportunity to not only test out the T2000 but some of Solaris 10's newer features, I loaded Solaris Express 3/06 onto the T2000 and lit the engines.

Excited about what the T2000 could do, we first did some basic benchmarks of the default applications included with Solaris (Apache, MySQL) as well as with some internal applications. Unfortunately, the numbers made the T2000 look extremely slow. Using the defaults for the benchmarks (primarily ab, sysbench and some home grown ones at this stage) the DL145 G2 that I was testing against performed faster than the T2000 no matter what I did. **Oddly enough, however, some of the built in performance monitoring tools with Solaris showed that the CPU was almost entirely idle**. This prompted me to look online for some ways to optimize these applications.

Shifting to MySQL (its the application we have the most experience with, and which could most benefit us), I discovered a great write up on BigAdmin about optimizing MySQL InnoDB specifically for Solaris (http://developers.sun.com/solaris/articles/mysql_perf_tune.html). Taking one of the suggestions from this article, I downloaded Sun Studio 11 and compiled MySQL with the options suggested. Another problem with the T2000 cropped up from this – it compiles *slow*. Because Sun's normal configure/make is single threaded, it takes an eternity to get an application the size of MySQL compiled.

In addition to some of Sun's performance suggestions, I discovered that by modifying some of the benchmark options I could get more complimentary numbers to occur.

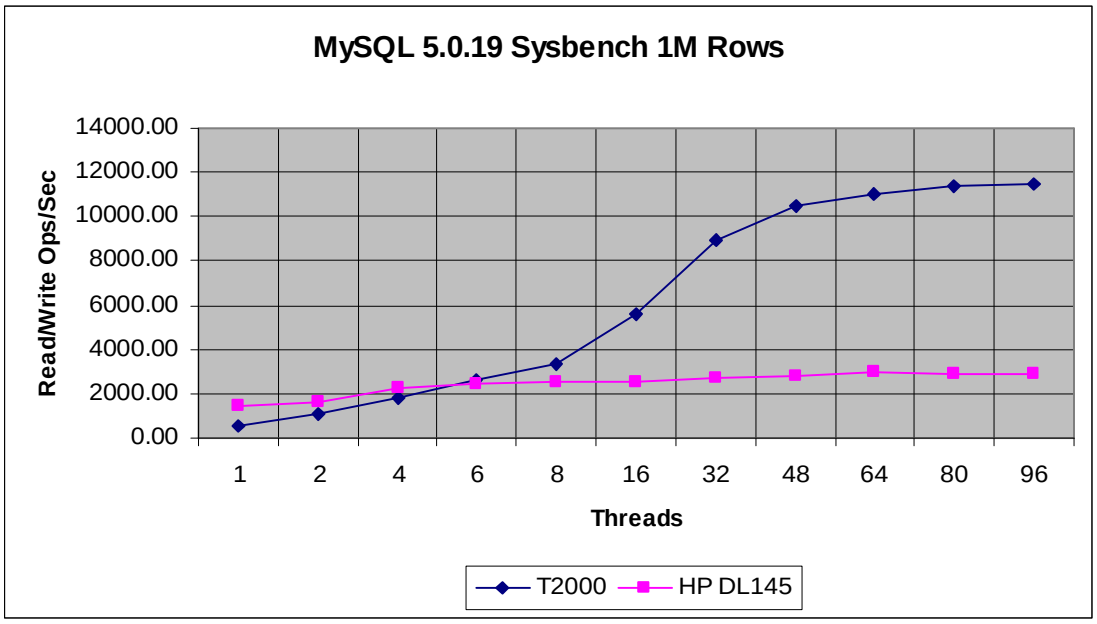
For the comparison, I'm using the T2000 and a HP DL145 G2, the stats for the HP are as follows:

CPU – 2x Opteron 244s
Memory – 4GB
HDDs – 1x U320-SCSI 73GB
OS – Gentoo 2006.0 (x64 optimized)

The stats for the T2000 used are:

CPU – 1x UltraSparc T1 8 core @ 1.0 ghz
Memory – 8 GB
HDDs – 2x 10k RPM 72GB SAS
OS – Solaris Express 3/06

Here are some preliminary numbers comparing the beginning optimizations of the T2000 to the base build of MySQL that comes with Gentoo. In both of these cases, I am using the "my-innodb-heavy-4G.cnf" configuration file.



My patience paid off. Comparing the max ops/sec of the DL145 to the T2000 showed an **approximately 4x performance increase** for the T2000. Something to note here is that the DL145 G2 has faster response times until the thread counts begin to get higher and higher. At this point, the T2000 takes over but it takes a significant amount of threads to start observing this behavior.

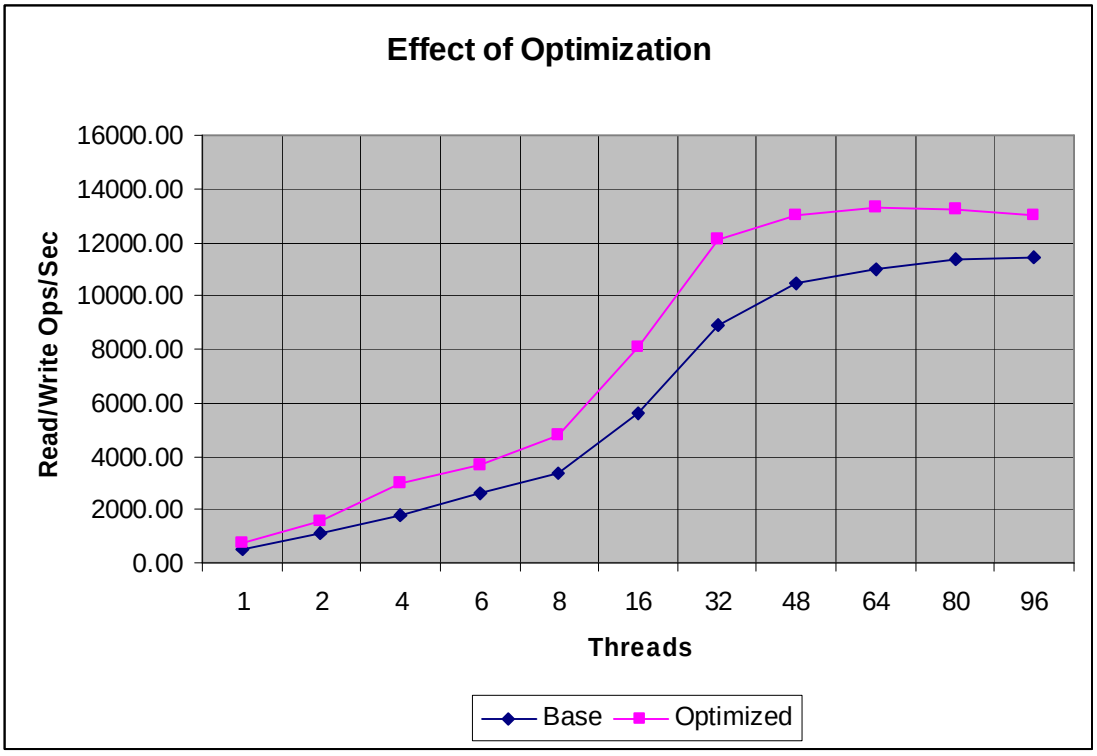
Not being comfortable, I decided to send an email to the author of the BigAdmin article (Jenny Chen who is a Software Engineer in Market Development Engineering at Sun) and ask if there was anything else I could do to get even more performance out of this box.

At this point, I was introduced to the holy grail of T2000 performance optimization: http://www.sun.com/servers/coolthreads/tnb/applications.jsp. In addition to the link to this page, Jenny shared some other tips that I could use for optimization including using the mtmalloc library when compiling, as well as, switching from the default mutexes to Solaris mutexes. (It would help the T2000 immensely in our opinion if it were easier to find the CoolThreads application optimization page. Without Jenny it was impossible to locate.)

Here is a quick summary of everything that I did to get a fully optimized MySQL build for the T2000 on Solaris 3/06:

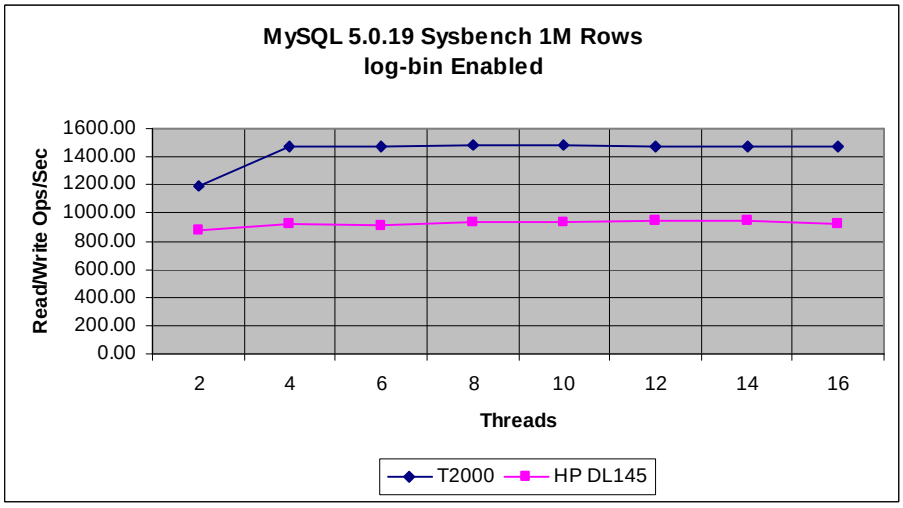
1. Changes to the /etc/system file (these require a reboot).
    a. set ip:ip_squeue_bind = 0
       set ip:ip_squeue_fanout = 1
       set ipge:ipge_tx_syncq=1
       set ipge:ipge_taskq_disable = 0
       set ipge:ipge_tx_ring_size = 2048
       set ipge:ipge_srv_fifo_depth = 2048
       set ipge:ipge_bcopy_thresh = 512
       set ipge:ipge_dvma_thresh = 1
       set segkmem_lpsize=0x400000
       set consistent_coloring=2
       set pcie:pcie_aer_ce_mask=0x1
    b. set maxphys = 1048574
    c. set segmap_percent=50
2. Modifications to MySQL.
    a. Added LIBS="-lmtmalloc" to the top of mysql-src/configure
    b. Modified the InnoDB mutex source to support Solaris mutexes (If anyone would like the modifications that were used, please send an email to thomas@digitar.com).
    c. Once the process was started, I switched MySQL from the default time-shared scheduling mode to fixed priority. (priocntl –s –c FX <mysqlpid>)
3. Moved from UFS to a mirrored ZFS solution across the two SAS drives that come with the T2000.

After all these optimizations, here is a before and after graph of the changes that they made:

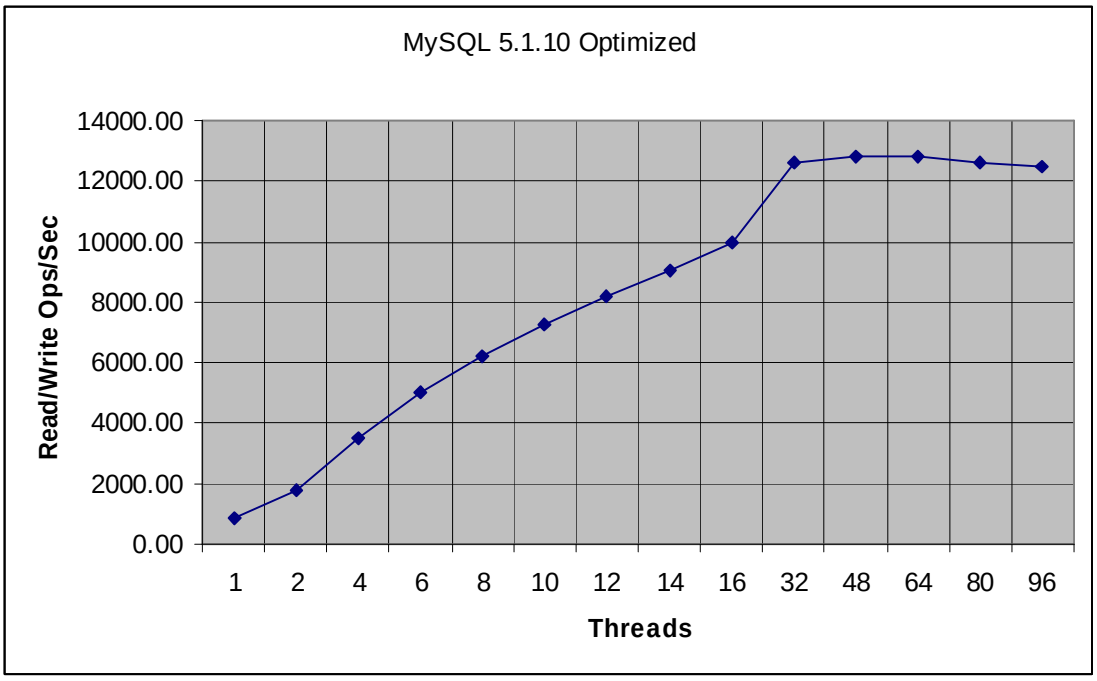


As you can see, the differences end up being pretty decent. At 32 threads, there is a 18.31% improvement from the above listed optimizations. **This increases the T2000's lead to 4.5x over the DL145 G2.**

So far, I have been trying to get a theoretical grasp on the max performance of the T2000 and decided to model the configuration options more closely to what was happening in the real world. For production servers, we use replication between all MySQL servers in a master/slave topology for high-availability purposes. To enable replication, you must enable the "log-bin" option in MySQL. For all tests, I have put the log-bin in the data directory along with the databases being used. Here is what happened:



To say that I was shocked is an understatement. Replication is something that is a requirement in my environment and it became very apparent that performance is significantly reduced when log-bin is enabled. Hoping that this wasn't the whole picture, I looked around online for some solutions. Not coming up with much, I decided to try out the latest MySQL release to see whether this was a known issue and if it had been fixed or not. So, I downloaded the 5.1.10-beta-nightly-20060412 package and got it setup with all the previously mentioned optimizations.



When using 5.1.10 there was no longer a performance penalty for having the log-bin turned on! What a huge difference!

Watching iostat, the difference in disk IO between 5.1.10 and 5.0.19 is pretty large. **I believe that the tests I'm running with 5.0.19 are IO bound and the T2000 could actually go faster.** (We'll post some more tests once we get her hooked up to our new Sun StorageTek FLX210 array). This is based on the fact that "iostat –cD" shows my data device fully utilized and that by moving to a mirrored ZFS device, the read/write ops/sec increased. With 5.1.10, disk IO does not appear to be as big of an issue. The ZFS device seems to be not fully utilized with utilization percentages in the 20-40% range.

What I don't understand is why 5.1.10 with log-bin enabled on the DL145 doesn't show the same drastic improvement over 5.0.19 that the T2000 shows. I've looked for ways to fix this problem but have been unable to uncover any way to do it. **If this is a problem that cannot be fixed, the T2000 comes out 13.5 times faster than my DL145.**

Conclusion

Something that these numbers don't show that I'd like to bring up is CPU utilization. During all these tests, I've been unable to get the CPU to have higher than 80% utilization. This opens up a whole new world for having an extremely fast multipurpose server in the T2000. By utilizing Solaris containers, you could consolidate your MySQL and Apache servers into the same box (or a pair for HA). There are no concrete numbers about the performance you could expect but from my preliminary findings, **you can run MySQL with the performance noted above, as well as Apache, and get Apache marks similar to a DL145 just running Apache**. Not only do you see the performance benefit from MySQL but you can also get a wickedly fast Apache server.

Here is a table with a quick recap of some of the performance numbers that I've discovered along with the "SWaP" metric:

System	Pre-Optimization	Optimized	Log-bin enabled (5.0.19)	Log-bin enabled (5.1.10)
Sun T2000	18.30	**22.24**	2.5	**21.38**
HP DL145		5.12	1.64	1.64

Unit of Measurement: Performance/(Rack Units Occupied * Wattage Consumed)

Not only does the T2000 run "cool and quiet" but it blows the doors off the HP DL145 G2 that I used for comparison. Even if we ignore the "cool and quiet" difference that your pocket book feels after running in a data center for some time (not just power and AC, but system density), there is a good case to be made with pure performance savings.

The list cost of the T2000 that was used in this test comes out to be $13,395. For the HP DL145 G2 we paid $3446. When coupled with Solaris 10, just looking at the 13.5x performance improvement between these two boxes, you'll see quite a price benefit. This completely ignores the very real but harder to quantify factors like scaling 14 DL145s instead of 1 T2000, as well as having to manage and wait for some percentage of 14 DL145s to fail. In addition, if you don't need some of the features (i.e. RAM, SAS drives) that come with the T2000, Sun has a T1000 that appears to perform similarly for only $6995. **As it is, DigiTar will save between 50-75% of our MySQL operations costs by moving our mission-critical MySQL operations to a pair of T2000s. Overall, it will help us eliminate the need for 8 HP DL145 G2s, not too mention drastically simplify our HA environment and increase our possible capacity by a factor of 2. Being in a highly competitive market, the T2000 will help us deliver the quality our customers expect of us at markedly lower costs than our competitors.**

From the time we received the T2000 everything was very exciting. The hardware is beautiful, easily setup, and you can really tell the difference in power and heat between it and current systems. *However, I then tried to actually use it and quickly became disenchanted.* Luckily, some of my initial user experience was based purely on seat of the pants "slowness" in applications that the T2000 is not very fast at (i.e. compiling). After spending a great deal of time searching for suggestions on

optimization I was able to get a couple real world applications running very quickly for me. It would be nicer if the needed optimizations were more readily accessible.

The T2000 is a killer product that really proves itself in the right situation. In my opinion, here are some things that if changed with the initial user experience would make the T2000 sell itself.

1. Have the default install of Solaris come with some basic optimizations – the changes I made to /etc/system don't seem to have any negative effects and increase the performance across the board.
2. Menu-driven suggested optimizations built into Solaris setup – when you first power the T2000 on, it would be immensely helpful for it to come up with some applications and their suggested optimizations.
3. Default workload setup – for some very specific workloads (Solaris, Apache, MySQL, PHP/Perl/Python aka SAMP), allow a user to select auto-configuration and optimization so that as soon as the T2000 boots from its initial install, a user can start using it immediately.
4. Increase the visibility of the T2000 tuning recommendations – have a link included in the setup documentation and during Solaris setup to a central information repository.
5. Include optimized builds of popular applications (i.e. MySQL) along with the tuning recommendations – **a major part of my negative user experience was related to how long it takes to compile things.** If I had been able to just download a pre-compiled package that was optimized, not only would I have skipped all the slower numbers in benchmarking but the whole negative experience associated with compiling.
6. Give sample benchmarks – another part of the tuning page could be benchmarks to use in testing comparisons between the T2000 and existing servers. Part of this would be binaries of the benchmarks for multiple platforms to ease this process as well as some common options that you can use.

Despite the effort required, we very much enjoyed our T2000 experience, and intend to evangelize its use to all of our colleagues whose businesses and livelihoods depend on mission-critical MySQL. The killer combination of Solaris 10, ZFS and the T2000 has earned the T2K a solid place in our data center and all of our future deployments.