CORSAIR™

APPLICATIONS NOTE

AN501: Latency Settings and their Impact on Memory Performance

John Beekley, VP Applications Engineering, Corsair Memory, Inc.

Introduction

Memory modules are currently available which support a wide variety of different speeds and latency settings. Speed is easy to understand - in general, faster is better. But what do the latency settings mean? And, what impact do they have on memory performance? This paper will provide a brief background on latency settings and what they mean. Then, we will move to the lab, where we will run a suite of benchmarks over a wide variety of latency settings, and we will measure the impact of these settings on benchmark scores.

Understanding RAMs

In order to understand even the basics of memory latency, we need to have a general understanding of how computer memory works. Let's look a little closer at how the memory within a memory DIMM is organized.

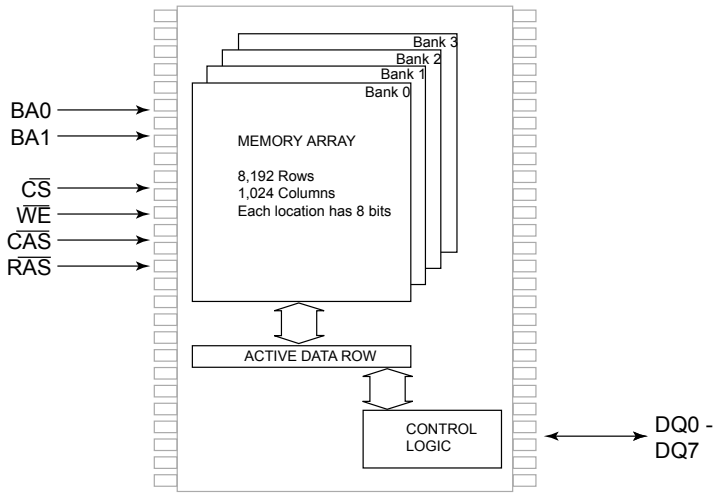

Figure 1. RAM IC Organization

Chip Basics

In Figure 1 we have a greatly simplified block diagram of a RAM IC. These devices are extremely complicated, and have far more structures than illustrated in the diagram. However, the core of the RAM is illustrated here – the memory array, where all the data is stored. To understand memory latency, we must first learn a bit about this device.

The memory array is a lot like a huge spreadsheet. It has rows and columns, and contains important information at each location defined by a row address and a column address. It has multiple banks, each of which is like another sheet of data.

The memory array in the RAM shown in Figure 2 has 8,192 rows and 1,024 columns. Each location defined by a given row address and column address has eight bits of data. So, each array

has 8,192 rows times 1,024 columns times one eight-bit byte, which equals eight megabytes, or 64 megabits. This memory IC has four banks, each of which has a memory array like the one we have just discussed. So, all put together, the memory chip would have four banks times eight megabytes per bank, equivalent to thirty-two megabytes (or 256 megabits).

Signal Names and Meanings

The most important control inputs for the RAM are shown on the left side of Figure 1. These signals perform the following basic functions:

- **CS#** is the chip select signal. When this signal is active, the RAM is selected. When this signal is not active, the RAM is standing by. The RAM must be active to read from or be written to.
- **BA0** and **BA1** are the bank address signals. They are used to define which bank in the memory array is to be read from or written to. There are four possible combinations of BA0 and BA1, one for each bank in the array.
- **RAS#**, **CAS#**, and **WE#** together comprise the Command Inputs. Based on the values of these signals, activities such as activating a bank, reading from the memory, writing to the memory, or configuring options are specified.

RAM Commands

To discuss specific memory latency settings, we first need to understand the basic commands that are used to control the RAM. Table 1 shows these commands, and gives a very general description of what they do.

	Signal State		
Command Name	**RAS#**	**CAS#**	**WE#**
No Operation	High	High	High
ACTIVE	Low	High	High
READ	High	Low	High
WRITE	High	Low	Low
PRECHARGE	Low	High	Low

Table 1. RAM Command Summary

It takes a sequence of instructions to read the RAM. First, the **ACTIVE** command must be issued to the row containing the desired data. Then, **READ** commands can be issued to read data from the active row. Back-to-back **READ** commands can be performed, so that a continuous, uninterrupted flow of data from the active row can be supplied. Writes have similar operating characteristics to reads – they are initiated with the **WRITE** command, and continuous, uninterrupted writes can be performed on the selected row.

Once all the desired data is obtained from and/or written to the active row, a **PRECHARGE** command is issued, essentially closing the row, and allowing another row to be activated.

Latency Basics

The speed at which the memory can provide data to the processor (also known as the "memory bandwidth") is determined by both the speed at which the memory is running (which is

obvious) and the latency settings of the memory (which is not so obvious). A more detailed look at latency follows.

What is "Latency"?

Just what is latency, anyhow? The dictionary defines it as "the period between stimulation and response". A real world example might include the delay between dialing a phone number and getting an answer on the other end of the line. Or, the time spent waiting for the coffee to brew after you have started the coffee maker. I think we can agree that in most cases, less latency is better. And in no case is this more true than in the computing world.

Primary RAM Latencies

Figure 2 shows the instruction sequence for reading a RAM, with the appropriate latencies inserted. As you can see, there are lots of latencies that contribute to the amount of time it takes to read the RAM.

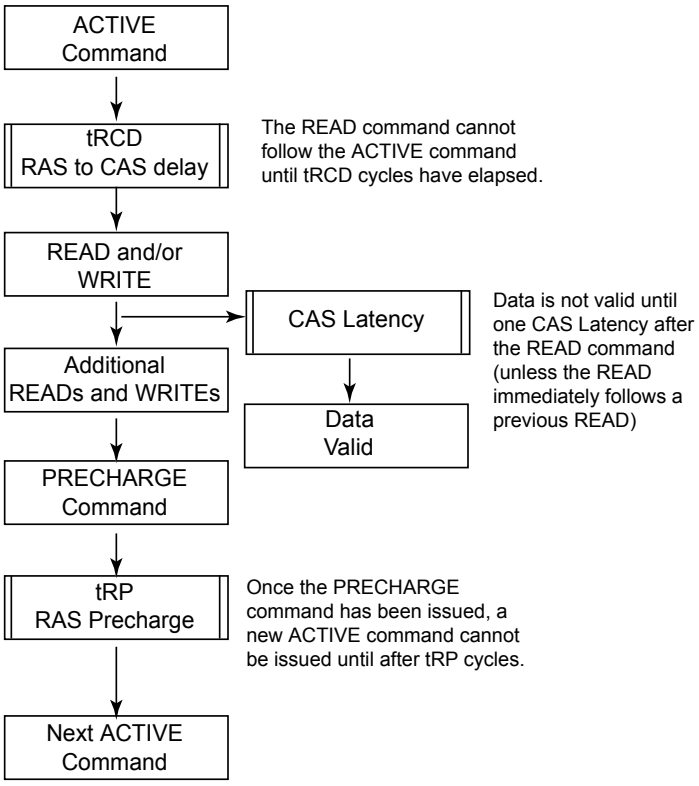

Figure 2. RAM Operation, including latencies

The most important latencies to be aware of during standard RAM operation are the following:

- **CAS Latency** is the delay, in clock cycles, between when the READ command is issued and when the data on the DQ pins is valid. Standard values for DDR memory are 2 and 2.5 clock cycles. Values of 3 and/or 1.5 clock cycles are available in some systems and are supported by some (but not all) RAMs. Note that a CAS latency of 2 cycles means that the data is valid on the rising edge of the second clock after READ is issued; a latency of 2.5 cycles means data will be available on the falling edge of the second clock following READ.

- **RAS-to-CAS Delay** is known as tRCD. It is the delay, in clock cycles, from when the ACTIVE command is issued to when a READ or a WRITE command can be issued. Is generally set to either 2, 3, or 4 clock cycles.

- **RAS Precharge** is also known as tRP. It is the delay, in clock cycles, from when the PRECHARGE command is issued to when the ACTIVE command can be issued for another row. Common settings for RAS Precharge are 2, 3, or 4 clock cycles.

- tRAS equals minimum **ACTIVE to PRECHARGE delay**. Once an ACTIVE command is issued for a given row, a PRECHARGE command cannot be issued for the row until tRAS has elapsed. tRAS is measured in clock cycles, and typical values are generally somewhere in the neighborhood of 5 to 10 clock cycles.

There is one more latency that we must be aware of, commonly known as **Command Rate**. Command Rate is the delay, in clock cycles, between when the CS# signal is activated and when any command (ACTIVE, for example) can be issued to the RAM. Common values for command rate are either 1 or 2 clock cycles.

Module latencies are often expressed as a combination of these values. The sequence we will use in this paper is CL-tRCD-tRP-tRAS-Command. So, a module with the following designation:

PC3200 2-3-4-5-1T

would have a clock rate of 200 MHz, CAS latency of 2 cycles, RAS-to-CAS delay of three cycles, RAS Precharge of four cycles, ACTIVE to PRECHARGE of five cycles, and a Command Rate of one cycle.

Test Description

We would like to determine the effect of memory latency settings on system performance. In order to do this, we will build a high performance computing platform. We will then run a suite of benchmarks to get an accurate measure of system performance. We will repeat this exercise under two conditions:

- Tightest possible latency settings at a fixed, 200MHz (PC3200) memory bus frequency. These tight latencies will consist of a CAS latency of 2 cycles, tRCD of 2, tRP of 2, tRAS of 5, and command rate of 1T. This will be denoted as “2-2-2-5-1T”.
- More relaxed, “nominal” latency settings, at the same 200MHz frequency. These nominal, relaxed latencies will consist of a CAS latency of 3 cycles, tRCD of 3, tRP of 3, tRAS of 8, and command rate of 2T. This will be denoted as “3-3-3-5-2T”.

The benchmarks used are a mix of synthetic benchmark and real-world benchmarks. Synthetic benchmarks are programs that are specifically designed to measure system performance. Real-world benchmarks are benchmarks based on commercial programs and/or real-world applications.

Benchmark Descriptions

The following benchmarks will be used to measure system performance:

- **3DMark 2001SE**. Out of all the 3DMark benchmarks, 2001SE was chosen because it displays greater dependency on CPU and memory and less dependency on video card performance than it successors, 3DMark 2003 and 3DMark2005. The 3DMark benchmarks are gaming-oriented, and are designed to estimate the relative gaming performance of the system under test.
- **PCMark 2004** - Memory test suite. PCMark is designed to measure relative performance

in general computing functions. The PCMark memory test suite focuses on system memory, so it makes a good measure of memory subsystem performance.

- **SiSoft Sandra 2005** - This system diagnostic has a memory benchmarking tool that is designed to measure memory bandwidth. It provides two output values; one for integer processing, and one for floating point processing.
- **Lavalys Everest** - This program is very similar to SiSoft Sandra, and provides a memory bandwidth measurement benchmark. Everest provides two output values; memory READ bandwidth and memory WRITE bandwidth.
- **Doom 3 timedemo, demo1** - This demo is included with the retail version of Doom 3, and provides a measurement of frames per second. By setting display resolution to 640x480 pixels, the benchmark score focuses on CPU/memory performance, rather than video card performance. This is a real-world benchmark, completely based on a retail game that is available to the public.
- **Super Pi** - Super Pi is a simple application which calculates pi (π) to a specified number of digits. Two million digits were chosen for this benchmark rather than the one million more commonly used, as the one second resolution of the measurement did not provide adequate granularity for a system of this performance. We will measure the time in seconds it takes to complete this calculation.
- **ScienceMark 2 Membench** - This is another synthetic memory performance benchmark, which tests a series of different memory bandwidth algorithms. It provides a single memory bandwidth measurement score.
- **DVD Encode using TMPGenc** - TMPG is a very popular MPEG encoding program. In this benchmark we measure the time it takes, in seconds, to encode a reference file into MPEG format for a DVD. This is another benchmark based on performance of a real-world task using a commercially available program.

Test Platform - Hardware

In order to test the performance impact of various latency settings, the first step was to build a high-performance test platform. For this experiment, we selected a test setup that is representative of the type of platform being built by hardware enthusiasts at the time of this study. A photograph of the test setup is shown in Figure 3. The following primary hardware components were used:

- Athlon 64 3500+ Socket 939 processor
- MSI K8N Neo2 motherboard
- 1 GByte Corsair TwinX1024-3200XL RAM
- 36GB Western Digital “Raptor” hard drive
- BFG GeFORCE 6800 GT video card

The Athlon64 platform was selected for this test because its on-board memory controller is very sensitive to latency settings, as there is no external memory controller on the Northbridge

to buffer the impact of the latency-related delay. High performance components were selected in all areas; the idea was to remove as many potential performance bottlenecks as possible.

The memory used in the tests is rated for performance at 2-2-2-5-1T at PC3200. It is known to overclock substantially beyond this specification, and is able to run all speed and latency combinations the tests require without a problem.

Figure 3. Benchmark Testing Setup

Test Platform - BIOS Settings

Prior to running any tests the BIOS was reset to default settings using jumpers on the motherboard. CPU FSB frequency was set to 200MHz, or PC3200. Hypertransport ("HT") multiplier was set to 5x, yielding an HT frequency of 1 GHz. CPU ratio was set to 11x, which results in a CPU clock rate of approximately 2.2 GHz (actual listed value is 2211 MHz). During testing, memory latency settings were modified as appropriate, but no other BIOS settings were modified in any way.

Test Results - Tight Latency vs. Relaxed Latency

Each of the benchmarks described above was run using two different memory settings - tight latency settings of 2-2-5-1T and relatively relaxed memory settings of 3-3-3-8-2T. The relaxed settings are typical motherboard default settings.

The results are shown in Figures 4. The test results are normalized to 100% for relaxed latency performance. This allows the results from all of the tests to be easily compared with each other. The blue bars in the graph show, for each benchmark, the additional performance that can be realized by utilizing tight latency settings. The actual benchmark result is included in the graph as well, to allow them to be compared with results from other sources.

A quick look at the results makes it clear that tightening up latency settings provides a substantial increase in performance. As one would expect, the most dramatic gains are seen in the synthetic memory bandwidth benchmarks. However the 5% to 8% gains seen in the real-world benchmarks are also quite significant. This is represents a substantial gain in performance which may be available to your system with just the simple change of a few BIOS settings.

A Closer Look: Impact of Latency on Doom3 Performance

After seeing the performance gains that could be achieved by moving from tight latencies to relaxed latencies, we decided to take a more detailed look at the impact of each latency setting. We wanted to do this under real-world conditions. We evaluated each of the benchmarks,

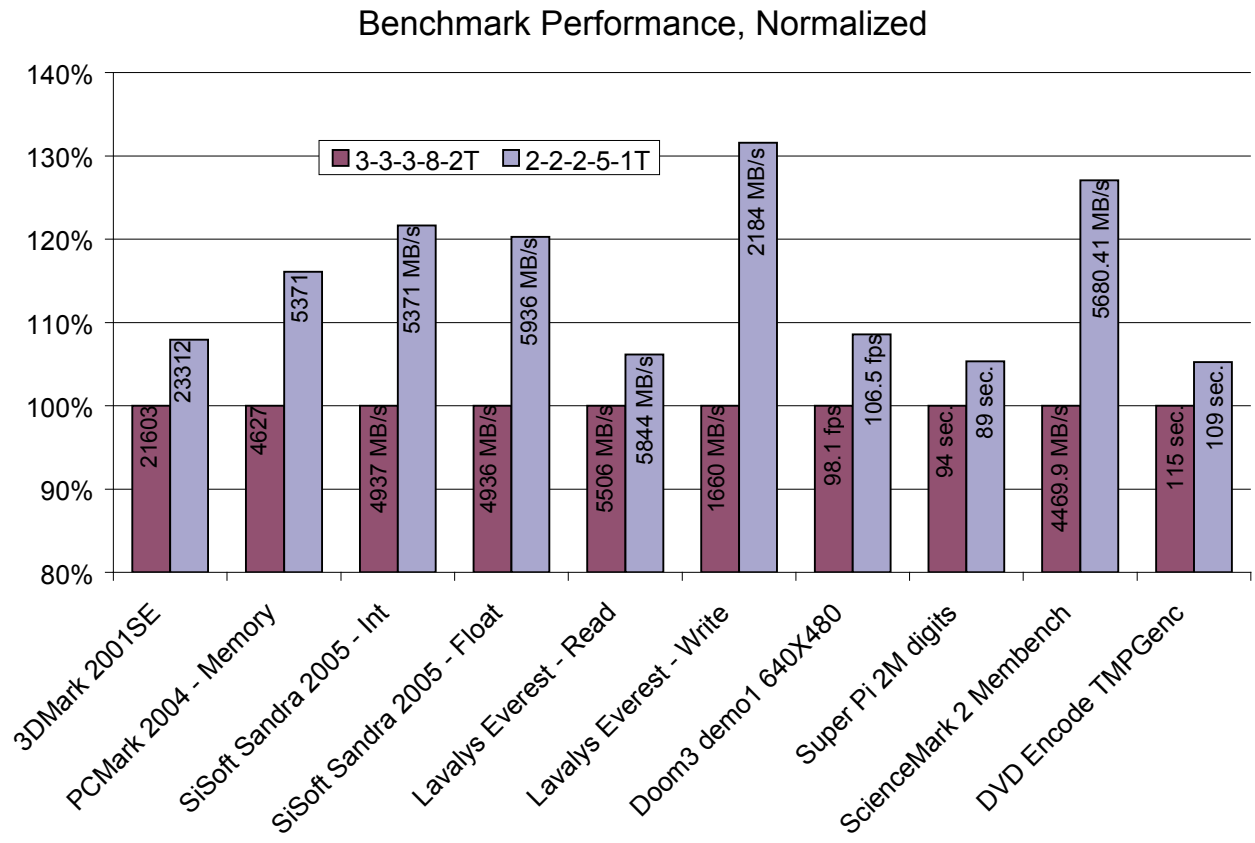

Figure 4. Benchmark Results, by application, for tight and relaxed latencies.

and found that Doom 3 appeared to provide the most consistent and measurable performance dependency on latency settings.

For this test, we started by re-running the Doom 3 demo1 640x480 benchmark at latency settings of 2-2-2-5-1T to confirm the reference value measured earlier. Then, we re-ran the test many times, each time only changing one latency parameter from the reference value of 2-2-2-5-1T.

For each setting, we ran the benchmark four times, and recorded the average of the top three results. This methodology was used to (1) discard the first run, which is always substantially lower than subsequent runs, and (2) ensure that no "rogue" measurements were recorded, either on the high side or the low side.

Results of this testing is shown in Table 2. As you can see, some of the results are dramatic and somewhat surprising. A brief discussion of the effect of the parameter changes follows:

- CAS Latency: This parameter is perhaps the most well known of the latency parameters. And, as expected, the result of relaxing the CAS latency by one cycle was significant. The performance decrease of 2.54% was the second highest measured in this suite of tests.
- RAS to CAS Delay: Recently the introduction of Corsair's 3200XL family has allowed users to set this value at two cycles, where previously only three cycles was possible. Again, the impact on performance is substantial, the 2.25% value representing the third

Latency Settings	Setting Modification	Doom3 demo1 benchmark score	Performance Decrease
2-2-2-5-1T	nominal	106.5 fps	0.00%
3-2-2-5-1T	CL +1	103.8 fps	-2.54%
2-3-2-5-1T	tRCD +1	104.1 fps	-2.25%
2-2-3-5-1T	tRP +1	105.8 fps	-0.66%
2-2-2-8-1T	tRAS +3	106.5 fps	0.00%
2-2-2-15-1T	tRAS +10	105.1 fps	-1.31%
2-2-2-5-2T	Command rate +1	103.0 fps	-3.29%

Table 2. Setting-by-setting latency impact

most significant decrease.

- RAS Precharge: This latency appears to have a minimal impact of overall performance. Similarly, our experience in the lab has been that the value of this setting has a minimal effect on RAM overclockability. So the conclusion that can be drawn is that tRP is a non-critical parameter.
- Active to Precharge: When moving tRAS from 5 cycles to 8 cycles, we saw no impact on system performance. This result was puzzling, so we attempted to confirm this on other benchmarks. On Sandra and Everest, we actually fount moving tRAS from 5 cycles to 8 cycles actually improved scores anywhere from 0.2% to 0.8%. To explore further, we relaxed tRAS to 15 cycles, the loosest supported by our BIOS. This extremely loose value only resulted in a performance decrease from 2-2-2-5-1T of 1.66%. So, as you can see, tRAS is non-critical, bordering on irrelevant.
- Command Rate: The results achieved by relaxing the command rate were dramatic. It was found to have the most significant impact on system performance, showing a 3.29% performance decrease when changing from 1T to 2T. To confirm these results, we performed the same analysis using Sandra. To our surprise, we found that of the roughly 17% performance decrease seen in Sandra when changing settings from 2-2-2-5-1T to 3-3-3-8-2T, *over 90%* of this increase was due to command rate alone!

Summary

Memory latency has always been know to have a substantial impact on performance of the memory subsystem. These tests help quantify that impact, giving a setting-by-setting measure of the performance improvements that result when these settings are optimized.

These tests were run on the Athlon 64 platform, which has an on-board memory controller. Results are likely to be different on other system architectures; we will explore these in future studies. On the tested platform, the results make it clear that the memory latency settings are very important to system performance. Surprisingly, out of all the memory settings measured in this test, a Command Rate of 1T appears to be by far the most critical.