SiliconGraphics
Computer Systems

An Illustration of the Benefits of the MIPS® R12000® Microprocessor and OCTANE™ System Architecture

Ian Williams

White Paper

An Illustration of the Benefits of the MIPS® R12000® Microprocessor and OCTANE™ System Architecture

Ian Williams

Overview

In comparison with other contemporary microprocessors, many running at significantly higher clock rates, the MIPS R10000® demonstrates competitive performance, particularly when coupled with the OCTANE system architecture, which fully exploits the microprocessor's capabilities.

As part of Silicon Graphics' commitment to deliver industry-leading application performance through advanced technology, the OCTANE platform now incorporates both system architectural improvements and a new-generation MIPS microprocessor, R12000. This paper discusses the developments in the MIPS R12000 microprocessor design and describes the application performance improvements available from the combination of the microprocessor itself and OCTANE system architecture updates.

Table of Contents

1. Introduction—OCTANE in the Current Competitive Landscape

When introduced in 1996, the MIPS R10000 microprocessor incorporated many advanced features. Operating in the Indigo2™ platform at clock speeds of 175 and 195 MHz, the systems demonstrated industry-leading performance in the desktop workstation marketplace across the spectrum of applications, from entertainment to geophysical, science, medical, and engineering.

Subsequently, the OCTANE system was launched; it incorporated the same MIPS R10000 processor, at 195 MHz, along with Indigo2 IMPACT™ graphics, called SI and MXI graphics. Due to the system design, which included advanced components such as the crossbar switch, OCTANE realized application performance improvements in both CPU and graphics-oriented activities ranging from 10% to 50%. In some cases the improvements were even higher.

Next, MIPS R10000 clock speed was increased 28% to 250 MHz, yielding an improvement in application CPU performance of typically 25%, and the introduction of E-series graphics yielded a further 20% graphics performance boost for many applications.

Figure 1 shows the system architecture of OCTANE and the key components of the MIPS R10000 microprocessor, heart memory controller, crossbar system interconnect switch, and SI graphics. For a full description of the OCTANE system architecture see the *OCTANE Technical Reference guide* [1].

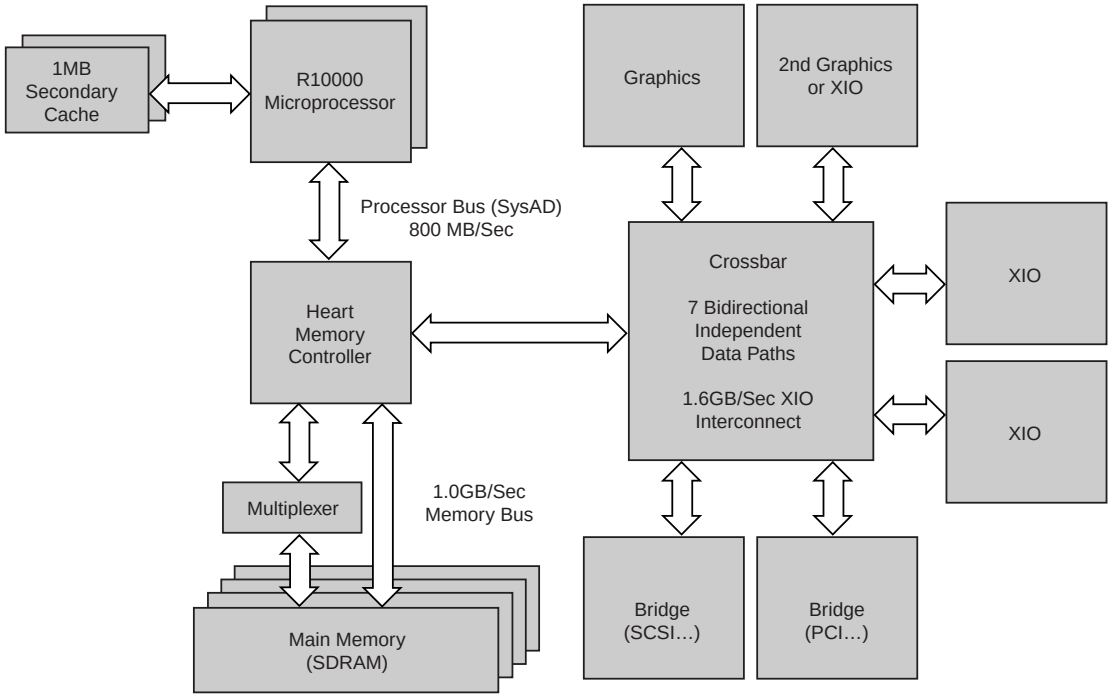

Figure 1. OCTANE System Architecture

Both the SPECint95 and SPECfp95 suite of benchmarks are developed from a range of real-world application examples, specifically chosen to exhibit integer and floating point dominated behavior, respectively, and are useful to position OCTANE against current competitive systems. The published baseline and peak results for both the SPECint95 and SPECfp95 test suites are shown in Figures 2 and 3, respectively, for a range of current desktop systems, including OCTANE. The full range of SPEC results is available from their Web site [2].

Although typically the peak SPECint95 and the SPECfp95 results are quoted when comparing system CPU performance, the base results have been included since they represent a microprocessor's performance without the benefit of aggressive software optimization. Because it can be difficult to obtain a high level of optimization for all sections in a large application, looking at both the base and peak results is important to provide a more accurate prediction of a microprocessor's performance.

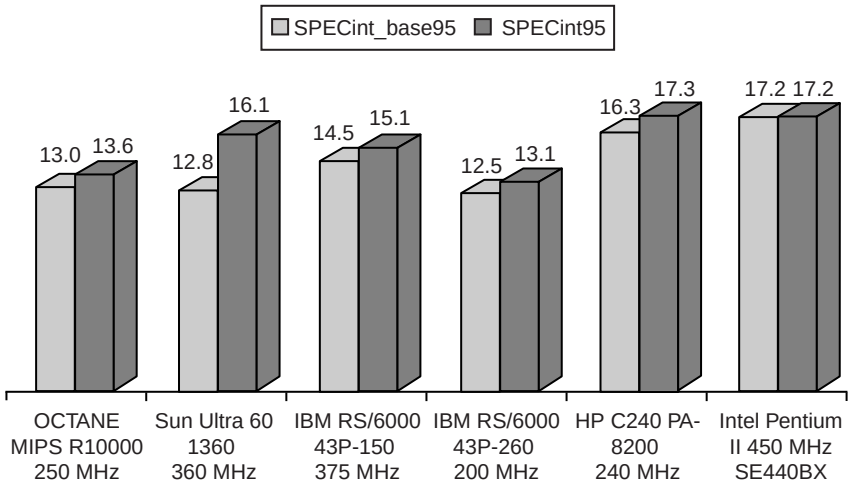

Figure 2. Published SPECint95 Results for OCTANE and Competitive Systems

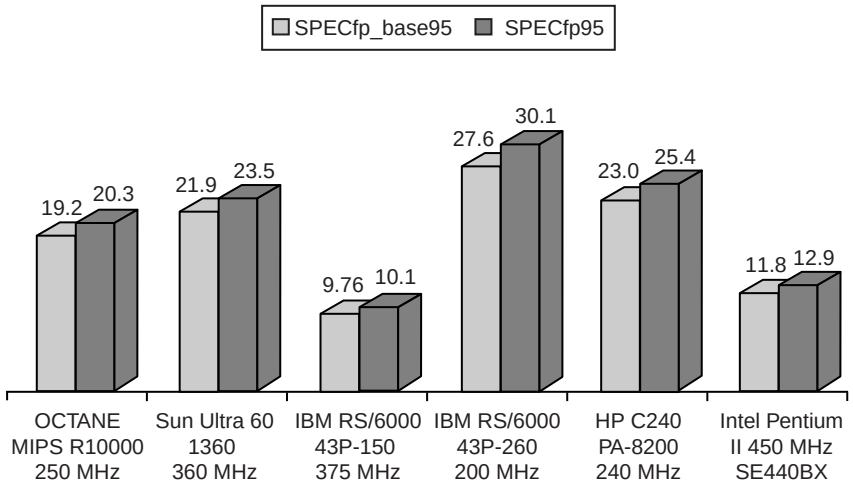

Figure 3. Published SPECfp95 Results for OCTANE and Competitive Systems

Clearly the SPECint95 results for Sun UltraSPARC II 360 MHz demonstrate a significant advantage from aggressive compiler optimization since without compiler optimizations, MIPS R10000 exhibits a slight performance advantage. The IBM PowerPC 604e running at 375 MHz in the RS/6000 43P-150 results also show relatively strong integer performance; however, its floating point performance is poor and the effects of this are likely to be evident in application performance. The same applies to the Intel Pentium II 450 MHz, which, although again demonstrating very good integer performance, has poor floating point performance.

The SPECfp95 results for both IBM Power3 and HP PA-8200 show an advantage through aggressive compiler optimization. To illustrate this, Figures 4 and 5 show the ratio of SPECfp95 to SPECfp_base95 for both HP PA-8200 and IBM Power3 relative to MIPS R10000 across all SPECfp95 tests.

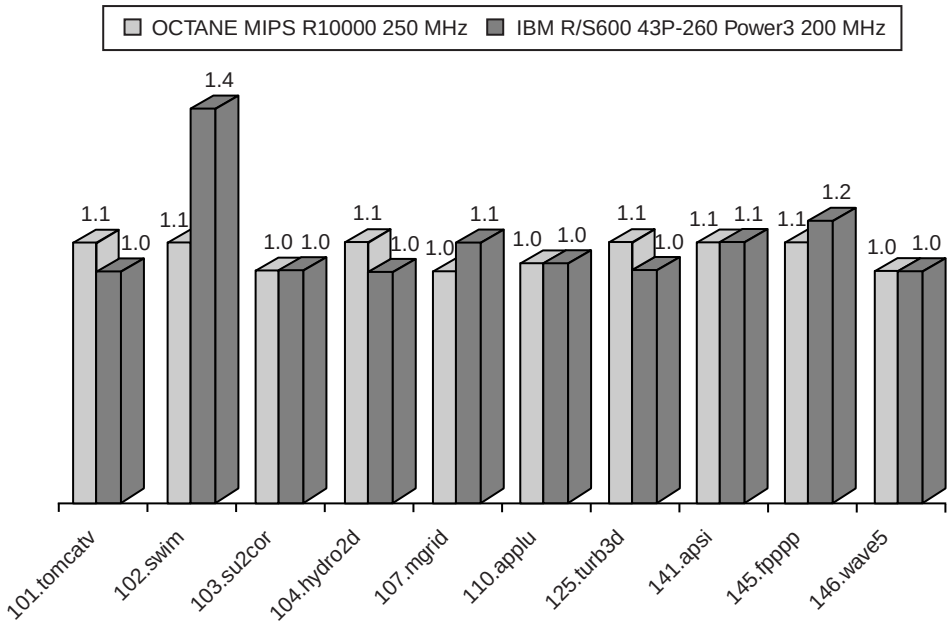

Figure 4. Comparison between PA-8200 and MIPS R10000 of SPECfp95 to SPECfp_base95 Results for All Tests

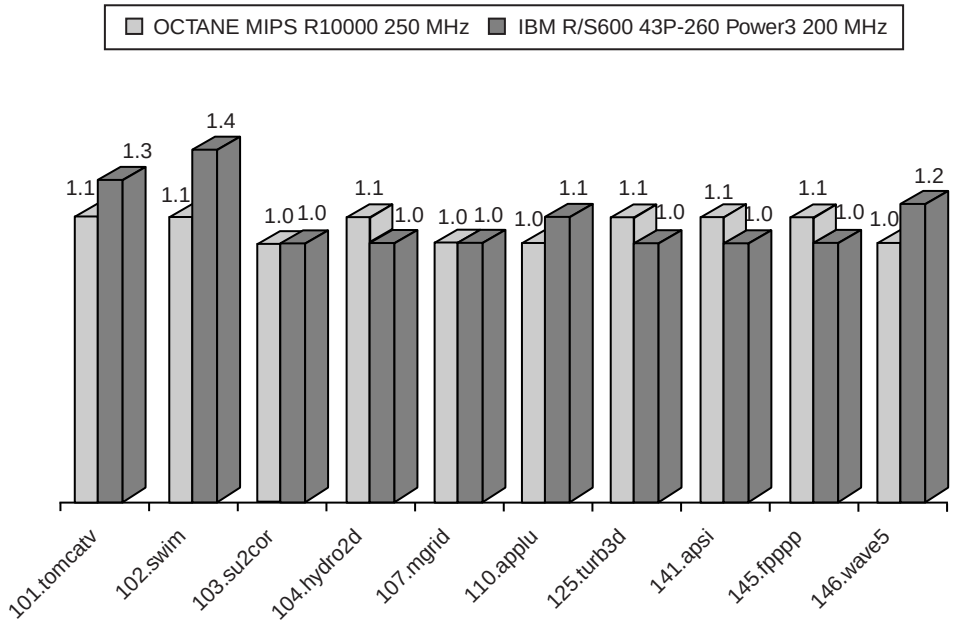

Figure 5. Comparison between Power3 and MIPS R10000 of SPECfp95 to SPECfp_base95 Results across All Tests

Although the architecture of both the PA-8200 and Power3 microprocessors is designed to deliver high floating point performance, clearly the SPECfp95 results for both also show the benefit of aggressive compiler optimization, yielding large performance increases on one or two tests within the overall suite.

An interesting comparison between OCTANE and HP C240 can also be made from a system perspective: due to the size and complexity of the PA-8200 design there is insufficient space on the chip to locate a primary cache. The HP C240, therefore, employs a single-level 4MB off-chip cache. In order to provide sufficient bandwidth between the cache and the processor, thereby avoiding significant performance implications, fast memory is required. The inclusion of this amount of expensive memory has a significant impact on overall system cost. Through the on-chip primary cache, coupled with a smaller secondary cache, MIPS R10000 in OCTANE is able to deliver competitive application performance at a lower list price.

As shown in the above results, the Power3 microprocessor clearly has strong floating point performance, however, its integer performance is lacking in comparison. When comparing application performance, the RS/6000 43P-260 with the Power3 microprocessor running at 200 MHz presents strong competition to MIPS R10000 250 MHz in OCTANE. There are some situations, however, where the IBM R/S6000 43P-260 isn't quite so formidable: Figures 6 and 7 show the published baseline results for the SPECint95 129.compress test and the SPECfp95 146.wave5 tests, respectively. The 129.compress test represents the compression of large text files (about 16MB) and the 146.wave5 test represents the solving of Maxwell's equations on a Cartesian mesh in an electromagnetic example. Clearly MIPS R10000 running at 250 MHz in OCTANE is very competitive with IBM R/S6000 43P-260 and all other systems.

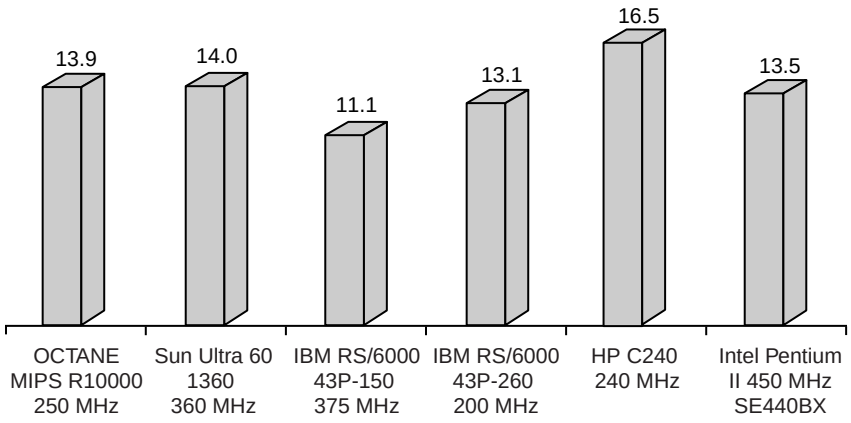

Figure 6. Published SPECint95 129.compress Base Test Results.

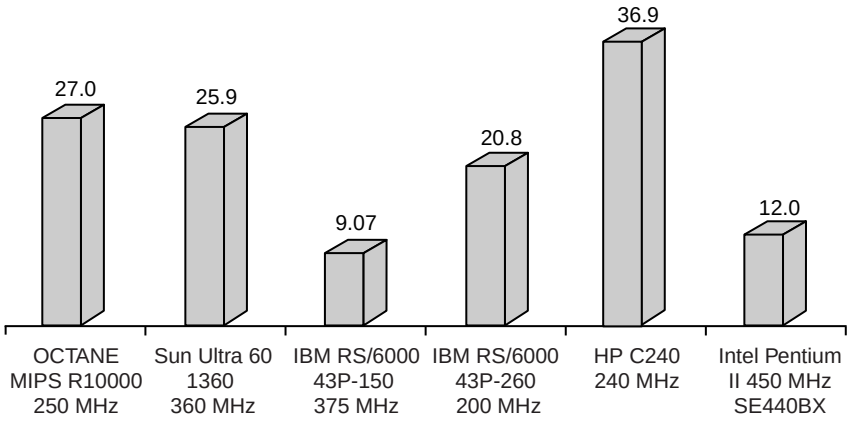

Figure 7. SPECfp95 146.wave5 Base Test Results

So why is MIPS R10000 in OCTANE able to deliver competitive application performance at slower clock speeds, offering balanced integer and floating point performance? The answer lies in the architectural features of its design, and these will be covered in the next section. When these features are combined with the OCTANE system architecture, both are synergistic to overall system performance and throughput.

2. Advantages of MIPS R10000 and MIPS R12000 Microprocessors

This section is divided into two subsections: the first describes the advanced architectural features in MIPS R10000 that allow it to deliver very competitive application performance, and the second extends this further, describing in detail the improvements in MIPS R12000.

2.1 Architectural Features of the MIPS R10000 Microprocessor

MIPS R10000 is a four-way superscalar RISC microprocessor. It fetches and decodes four instructions per cycle and speculatively executes beyond branches with a four-entry branch stack. It uses dynamic out-of-order execution, implements register renaming logic using map tables and achieves in-order graduation for precise exception handling.

This section explains these features and their relevance to application performance. Inevitably, however, some aspects probably don't receive as much attention as they should, so the reader is invited to refer to [2] and [3] for further details. It's also assumed that most readers will be familiar with many of the computing terms and concepts discussed. Some readers, however, may not be as familiar, so for their benefit the first part of this section provides a brief description of some of the key technologies in MIPS R10000 before discussing the architecture in further detail. References [4] and [5] provide further information on the more general aspects of microprocessor and computing technology.

Most system users are probably familiar with the concept of cache memory: a phenomenon called locality of reference shows that executing code has a tendency to access memory in close proximity to other recently accessed locations. When accessing a memory location, by loading it and the data immediately surrounding it into a small amount of high-speed cache memory, subsequent accesses may be fulfilled from cache, therefore significantly reducing the overall time spent accessing main memory. MIPS R10000 uses two levels of set-associative cache memory: the primary level is on-chip and has 32KB for data and 32KB for instructions; the off-chip second level is unified for both data and instructions and can range in size from 512KB to 16MB. In OCTANE its size was previously set at 1MB, although recently a 2MB option has been introduced.

Pipelining is a technique that is applied to many situations to speed up the overall execution of a process. In the simplest case, an overall task is divided into a series of operations, or stages, that when applied sequentially perform the overall task; however, each operation can also be performed in parallel with the rest. As one item progresses through the pipeline, other items can be initiated before the first has completed all stages, and over time, with many items progressing through the pipeline, this yields a significant increase in throughput. When applied to a microprocessor, the simple pipeline is extended so that some stages may be performed individually rather than all sequentially—contributing to the completion of a overall task. Executing an integer instruction for example, is independent of executing a floating point instruction. Since both instructions need to be fetched from main memory, or cache, along with associated data, which may take a varying amount of time, pipelining the whole process yields significant throughput advantages. When a microprocessor can perform several operations simultaneously in one stage of its pipeline it is called superscalar. The MIPS R10000 microprocessor is superscalar.

Microprocessors using pipelining also use a clock to synchronize their overall operation and the individual stages. Operations, therefore, are typically defined as taking a certain number of clock ticks; however, all operations in the pipeline may not take the same length of time. Because division, for example, is usually implemented as a series of additions, it's easy to see why this might arise.

Pipelines work most efficiently when there is a continuous flow of instructions through them. If for whatever reason a pipeline needs to be stopped, then depending on how many stages it contains, or depth as it is also called, there will be a long delay while the instructions currently being executed are flushed. Such a situation can occur with a branch instruction as would be generated from an IF-THEN loop within code. One method to significantly reduce the potential performance penalties that may arise between encountering a branch instruction and until the outcome of the branch is known is to use branch prediction. This is where the processor takes a guess at which path to follow after the branch and continues to issue instructions into the pipeline after following this path. Because it's not known whether the branch decision is correct when these instructions are issued, they are executed speculatively until the result of the branch is known. If the branch decision turns out to be correct, no interruption occurs. If the branch decision turns out to be incorrect, then all the speculative instructions are discarded. Clearly the correct branch choice won't be made all the time; however, if it is on balance, then the overall effect can be a significant improvement in throughput and performance. The MIPS R10000 microprocessor employs both branch prediction and speculative execution.

Because of the effects of caching, some instructions may have to wait longer for their data to be retrieved from memory. As a result, when many instructions can be scheduled and executed in a pipeline, being able to issue them for execution only in the order in which they arrive could yield a significant reduction in pipeline throughput and efficiency. A technique used on MIPS R10000 to overcome this is called out-of-order execution, which allows instructions to be issued independent of the order in which they are decoded. After execution the instructions are regrouped back into their original order in the active list. Completed instructions can be removed from the active list in order. This process is called graduation. After being retrieved from main memory or cache, an instruction's data, or operands, are stored in registers during execution. To allow an instruction's operands to correctly follow it as it progresses through the execution pipeline, a technique called register renaming is used. This ensures that each instruction can unambiguously reference its operands regardless of the order in which it is being executed.

All current high-performance workstations use a virtual memory model that allows efficient use of a system's main memory. It also makes application programming easier since no specific knowledge is needed about architectural features of a system such as how memory is allocated, etc. Applications reference virtual memory addresses that are subsequently translated into physical locations by the operating system kernel. Since every memory access for both instructions and data goes through this mechanism, most microprocessors employ dedicated hardware to speed up the translation using a translation lookaside buffer (TLB). To ease organization, speed, and access, main memory is divided into a series of pages and an address corresponds to a page number and offset. The TLB caches recently accessed pages, and if a desired address happens to be in one of those pages, the translation will occur very quickly. If an address causes a TLB miss, then there is a significant delay while the page mapping for the particular process is accessed and the address subsequently

translated. MIPS R10000 employs a 64-entry fully associative TLB. A smaller eight-entry TLB, which contains a subset of the main one, is dedicated to instructions. Full associativity allows TLB entries to vary in size; hence MIPS R10000 can accommodate different memory page sizes, between 4B and 16MB. This significantly enhances the ability to tune a system to meet the demands of a wide range of applications.

As mentioned above, both primary and secondary caches on MIPS R10000 are set associative. Set associativity refers to how the cache memory is organized. It is searched when the processor requests the contents of a memory location. Instead of searching through the entire contents of cache memory for a match to the given address, only a small set of locations are searched in parallel. Conceptually a set-associative cache can be thought of as having several ways that comprise a series of sets that are all a cache-line in length (see Figure 8). In a two-way set-associative cache, therefore, there are two ways. The number of sets and the cache-line length, however, depend on both the processor and the cache level.

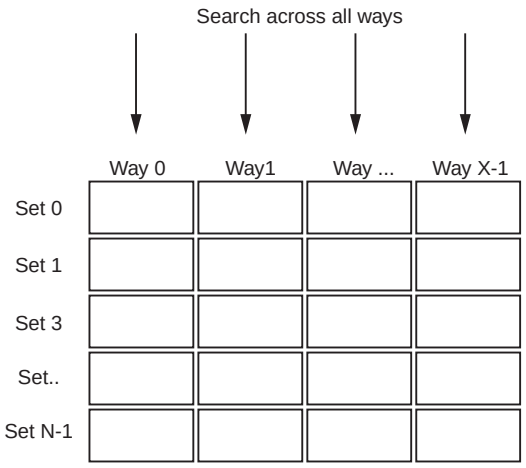

Figure 8. Organization of an X-Way Set-Associative Cache

Searching the set-associative cache on MIPS R10000 begins with the memory address, which comprises several components: the tag, the index, a bit that determines the cache bank, and two bits to determine the specific double word in a set. When a memory address is given to cache, the index selects the appropriate set to search. All ways within the cache typically have a dedicated tag array, sometimes called a directory, with the number of entries corresponding to the number of sets. Once the set is known, the tag array in each way is searched and a hit occurs if the tag matches the entry. The double word in a cache line is then determined by the corresponding two bits in the address and the contents of the address are returned to the CPU. If a match is not found in any way, a miss occurs and is reported back to the CPU. Figure 9 shows this process for the primary data cache on MIPS R10000.

Both primary data and instruction caches on MIPS R10000 are implemented such that the tags are checked in both ways simultaneously, allowing searches to be made across both ways in parallel. Since both these caches are on-chip, this is a more efficient way to implement this compared with performing the searches sequentially. The secondary cache, however, is off-chip, and implementing it in the same fashion would physically increase the number of connectors necessary to connect the CPU to the external world, as well as potentially cause inefficiencies when accessing memory that is multiplexed. To overcome this MIPS R10000 allows both ways to share the same path to the processor and incorporates a way prediction, or most recently used (MRU), table. The MRU table significantly improves the chances of making the choice the correct way to search first.

If a primary cache miss occurs during a read, the address is passed to secondary cache. If the secondary cache hits, the least recently used primary cache line is replaced with data from secondary cache. If the secondary cache misses, a request is sent to retrieve the relevant data from main memory and the least

recently used secondary cache line is replaced. This replacement algorithm is called a least recently used (LRU) mechanism.

MIPS R10000 employs a write-back protocol in the cache hierarchy. This means that stored data is always written from the CPU into primary cache. When lines in primary cache are replaced with lines from secondary cache the replaced line is first checked to see if it has been modified. If it has, it is written to secondary cache before being replaced. In a similar way, when a line in secondary cache is replaced it is first checked to see if it has also been modified and, if so, is written back to main memory.

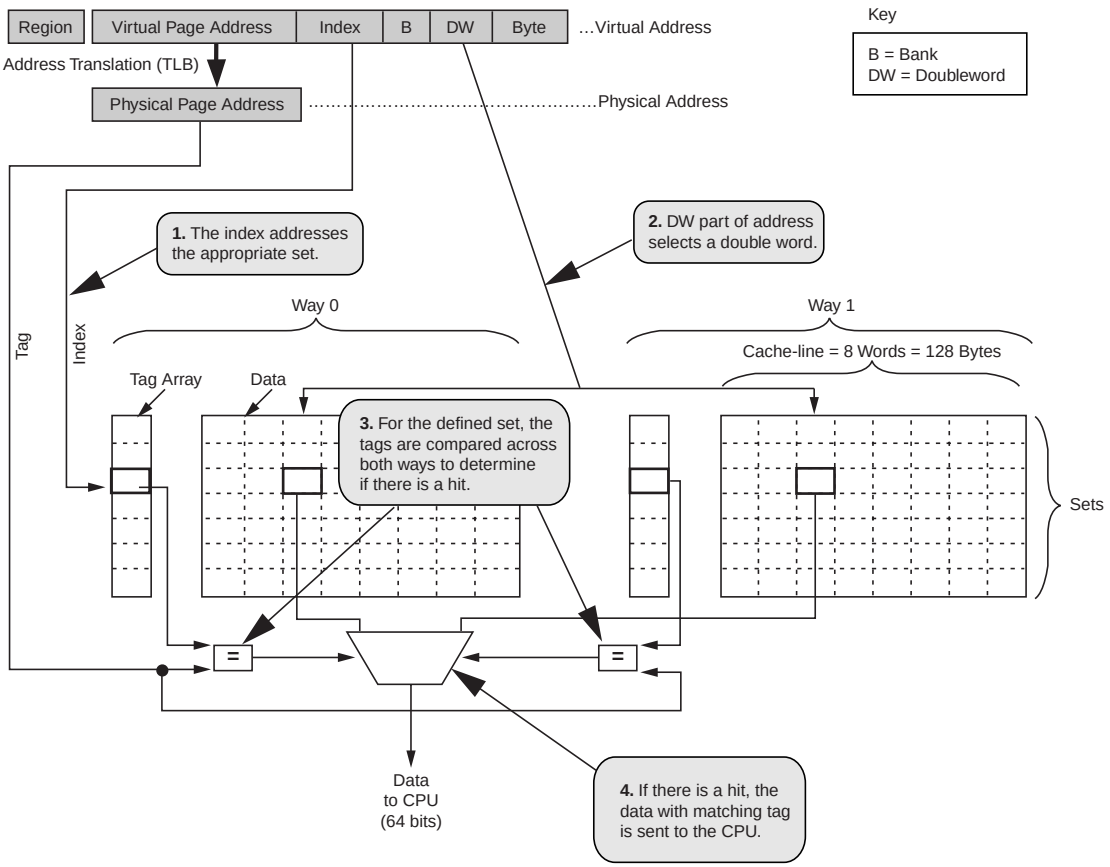

Figure 9. MIPS R10000 Primary Data Cache

In a write-through cache, every store instruction updates both the cache and main memory. This can generate a lot of traffic on the system bus. A write-back protocol can yield a performance advantage with a significant reduction in bus traffic when cache sizes are large to ensure hits in all levels within the hierarchy.

The MIPS R10000 primary data cache implements one parity bit per byte. The primary instruction cache has parity for each instruction. The secondary cache stores a 9-bit Error Checking and Correction (ECC) code for each quadword. This provides single-bit error correction and double-bit error detection. Neither the parity nor ECC checking incurs an overhead on cache access.

Figures 10 and 11 detail the architecture and pipeline stages of the MIPS R10000 microprocessor and outline the steps when executing instructions. The following paragraphs refer to these diagrams and explain the process of retrieving and executing instructions using many of the terms just described.

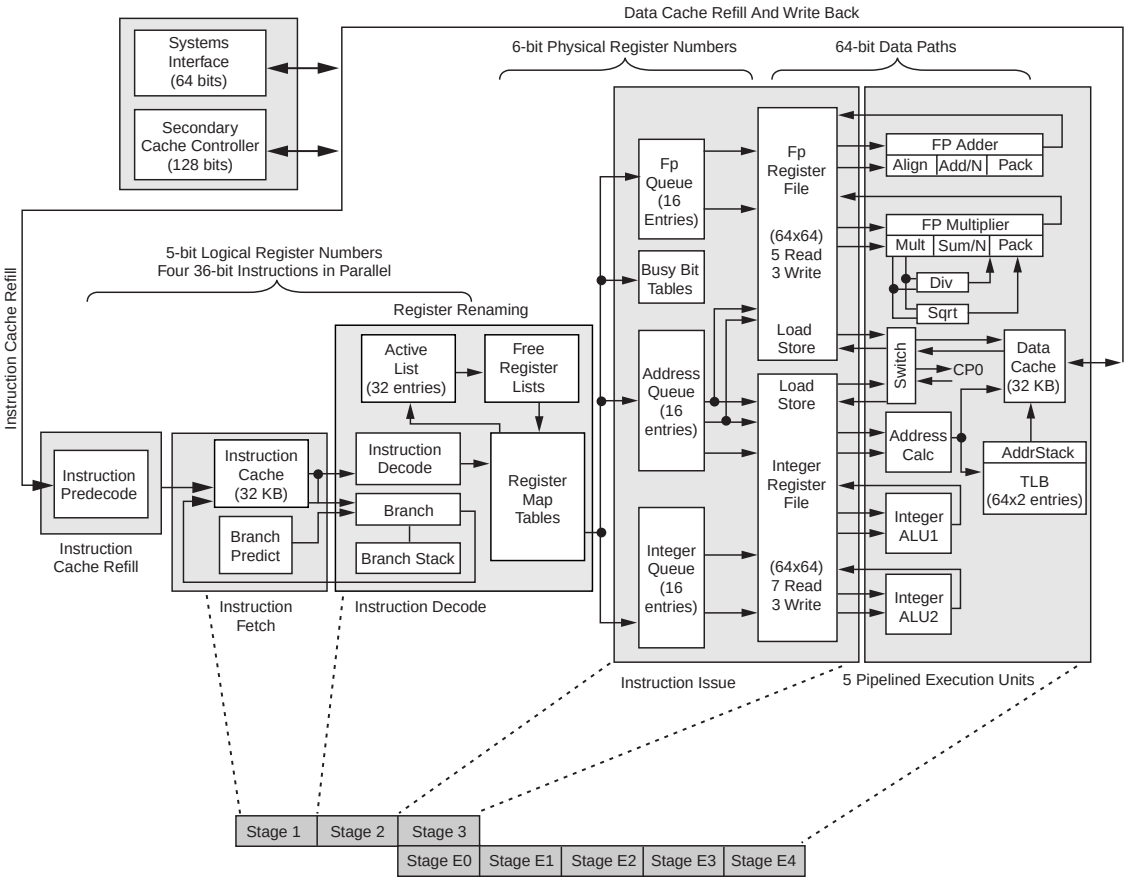

Figure 10a. MIPS R10000 Microprocessor Architecture

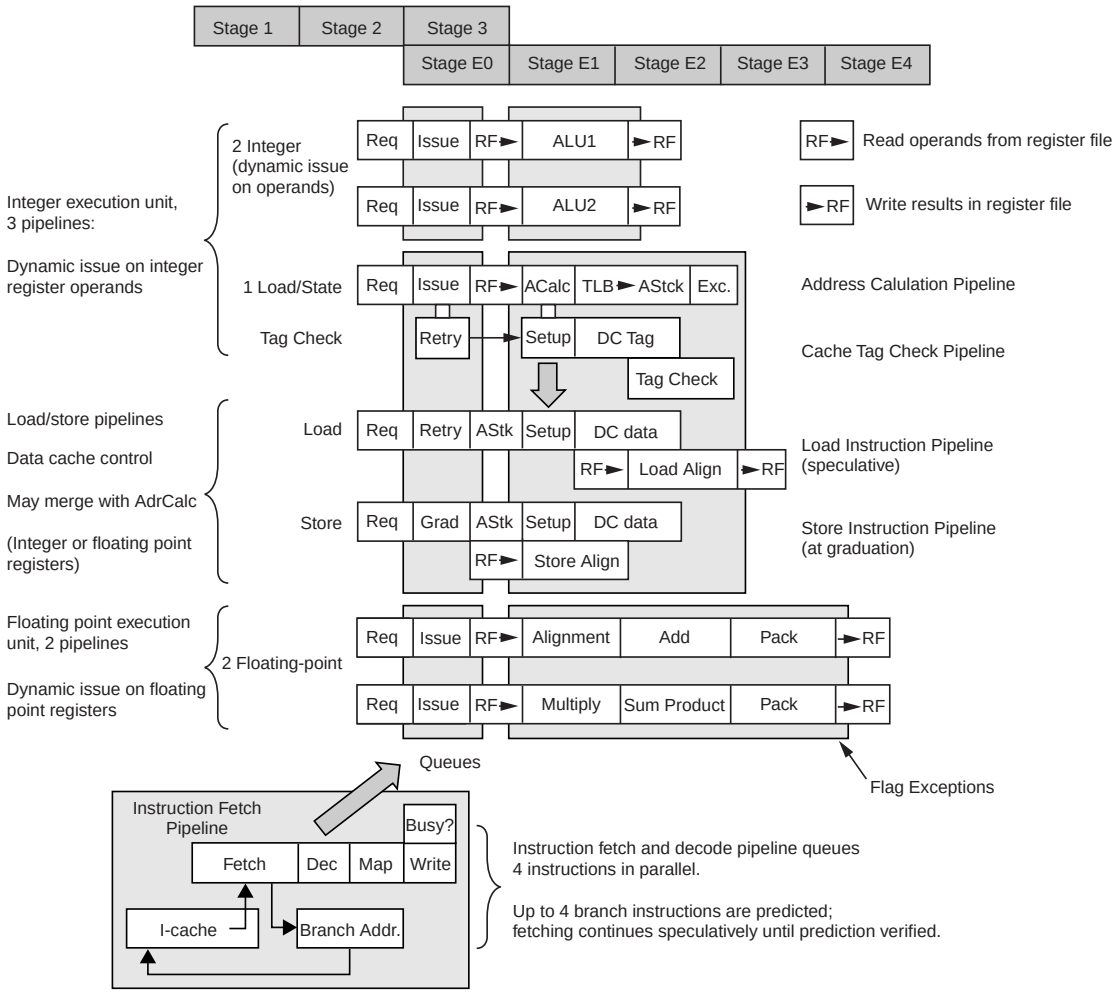

Figure 10b. MIPS R10000 Pipeline Timing

The instruction fetch pipeline fetches and decodes four instructions simultaneously from the instruction cache. In order to ensure this process occurs rapidly, instructions are predecoded, wherein fields are rearranged and a code is appended indicating the required execution unit before they are placed in the instruction cache. Each decoded instruction is also recorded in the active list, which has 32 entries. The active list also retains the original program order.

During stage 1, shown in Figures 10a and 10b, instructions are fetched sequentially. During stage 2 target addresses are calculated. When conditional branch or jump instructions occur, a new program address that introduces a one-cycle delay will be selected. The condition determining a branch may not be known for many more cycles, so if the processor were to wait until the result was known it would introduce a significant delay, yielding a noticeable impact on application performance. Studies have shown that branch instructions typically comprise between 15-20% of all instructions for many application codes. Accurately predicting correct branch outcomes and continuing to fetch and execute instructions will remove a major potential bottleneck, significantly improving application performance. To this end, MIPS R10000 uses branch prediction based upon a 2-bit, 512-entry table.

When a branch instruction is fetched and decoded, the tag is used to reference a counter value in the prediction table. When the condition determining the branch is subsequently known, if the branch was taken, the counter value increments by one, and it decrements by one if the branch was not taken. When the same instruction is encountered again, the high-bit of the 2-bit counter value determines if the branch is taken or not. Again, if the branch was subsequently taken the value increments by one and it decrements by one if the branch is not taken. Using counters with more bits typically hasn't shown a significant increase in overall prediction accuracy. To facilitate quick recovery should a branch decision turn out to be incorrect, MIPS R10000 stores its state at the time the branch occurs in a 4-entry branch stack. When instructions are speculatively executed after a branch, if the decision turns out to be incorrect all these instructions are discarded and register values etc. are reloaded from the branch stack.

During stage 2, four fetched instructions are decoded and loaded into one of three queues depending on what functional unit is required: integer, floating point, or load store. As part of the decoding process operands and destination registers are renamed. Register renaming, as this is called, is an elegant way to handle dependencies during out-of-order execution. Logical registers are mapped into physical registers in a process similar to mapping virtual address into physical ones.

Register renaming occurs in a similar fashion for both integer and floating point registers. It is explained as follows: logical registers are the registers defined in the instruction set architecture. They are selected by 5-bit fields within each instruction. Physical registers are the actual memory elements that store register values within the microprocessor. During instruction decode the logical registers are mapped to physical registers. After mapping dependencies can be determined by comparing physical registers without concern for the original instruction order. There are 64 physical integer and 64 physical floating point registers. The 32-entry mapping table sortes the 6-bit physical register number currently associated with each logical register. There is also a 32-entry free list that stores the physical registers numbers not currently used. As a new mapping is created, its physical register is taken from the free list. Later as it graduates, its physical register is returned to the free list. When a branch is predicted, shadow copies are made of both integer and floating point mapping tables, the active list write pointer, and both free list read pointers.

During stage 3, once their operands become available and are read from the register file, instructions are issued to the appropriate pipeline for execution during stages E0 to E4. As shown in Figures 10a and 10b, the five execution units comprise two integer arithmetic logic units (ALUs), a floating point multiplier, a floating point adder, and a load/store unit. Each execution unit is pipelined with a single cycle repeat rate, which means that another instruction and operand(s) can be issued in the next cycle after the last one was issued.

On MIPS R10000, instructions in the integer and floating point queues are issued to the two ALUs after their operands become available, but in no particular order. Likewise, instructions in the floating point queue are issued to the floating point adder or multiplier. Once instructions in the integer or floating point queues are issued, they are deleted. Instructions in the address queue are issued to the address calculation unit and data cache. The data cache has two independent banks that can operate concurrently when loading, storing, and refilling. Loading refers to transferring data from cache to the processor register files. Storing refers to transferring data from the processor register files back to cache. Refilling refers to transferring a cache line from secondary cache or main memory to the data cache.

Transferring data from main memory to cache takes a relatively long time. For performance reasons the MIPS R10000 load and store operations can overlap with up to four cache refills and can be performed out of order. During a cache refill the particular word of interest is transferred first, and can also be passed directly to the execution units, thus further improving overall performance.

2.2 Architectural Improvements of the MIPS R12000 Microprocessor

The main architectural improvements of MIPS R12000 compared with MIPS R10000 are summarized in Figure 11a, which shows the microprocessor architecture and timing pipeline. The timing pipeline is shown in Figure 11b.

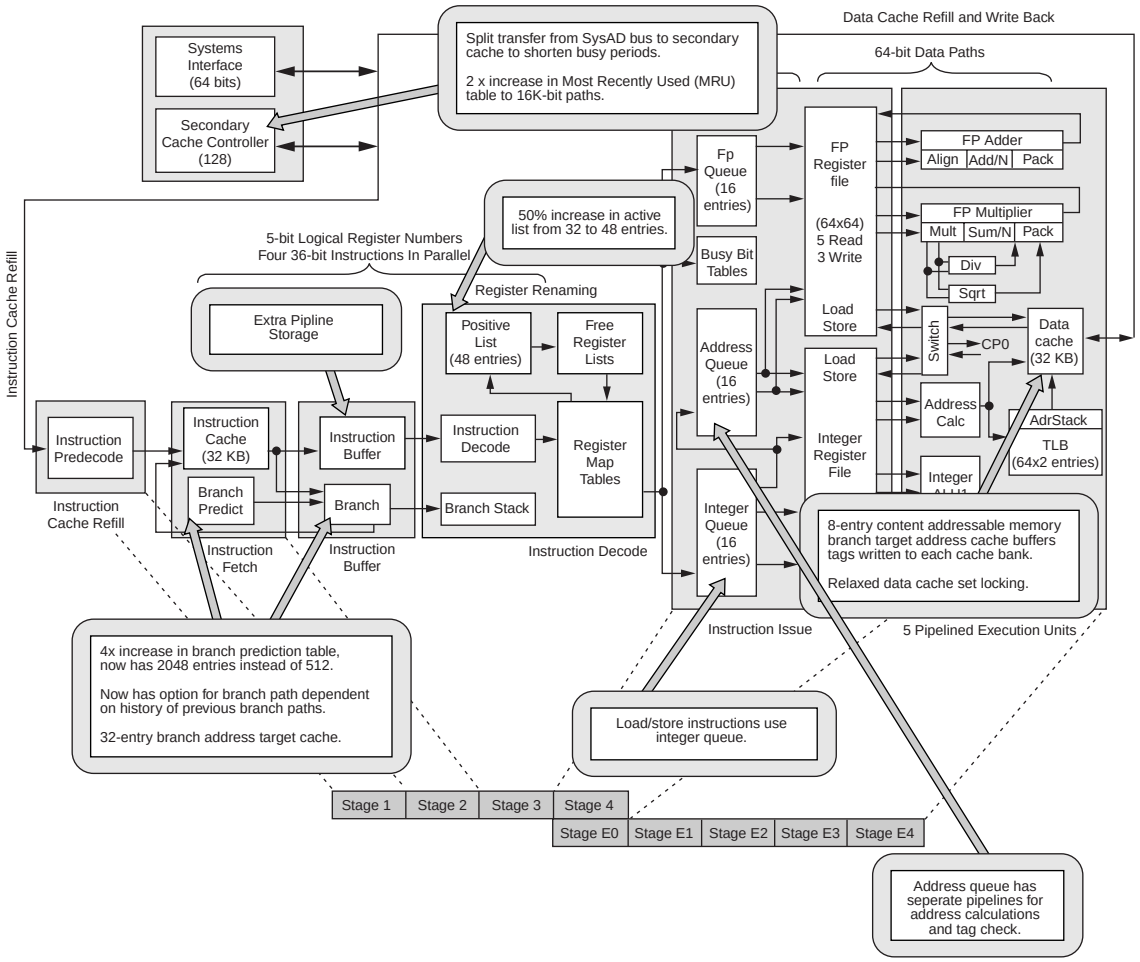

Figure 11a. Summary of Changes Between MIPS R10000 and MIPS R12000 Microprocessor Architectures

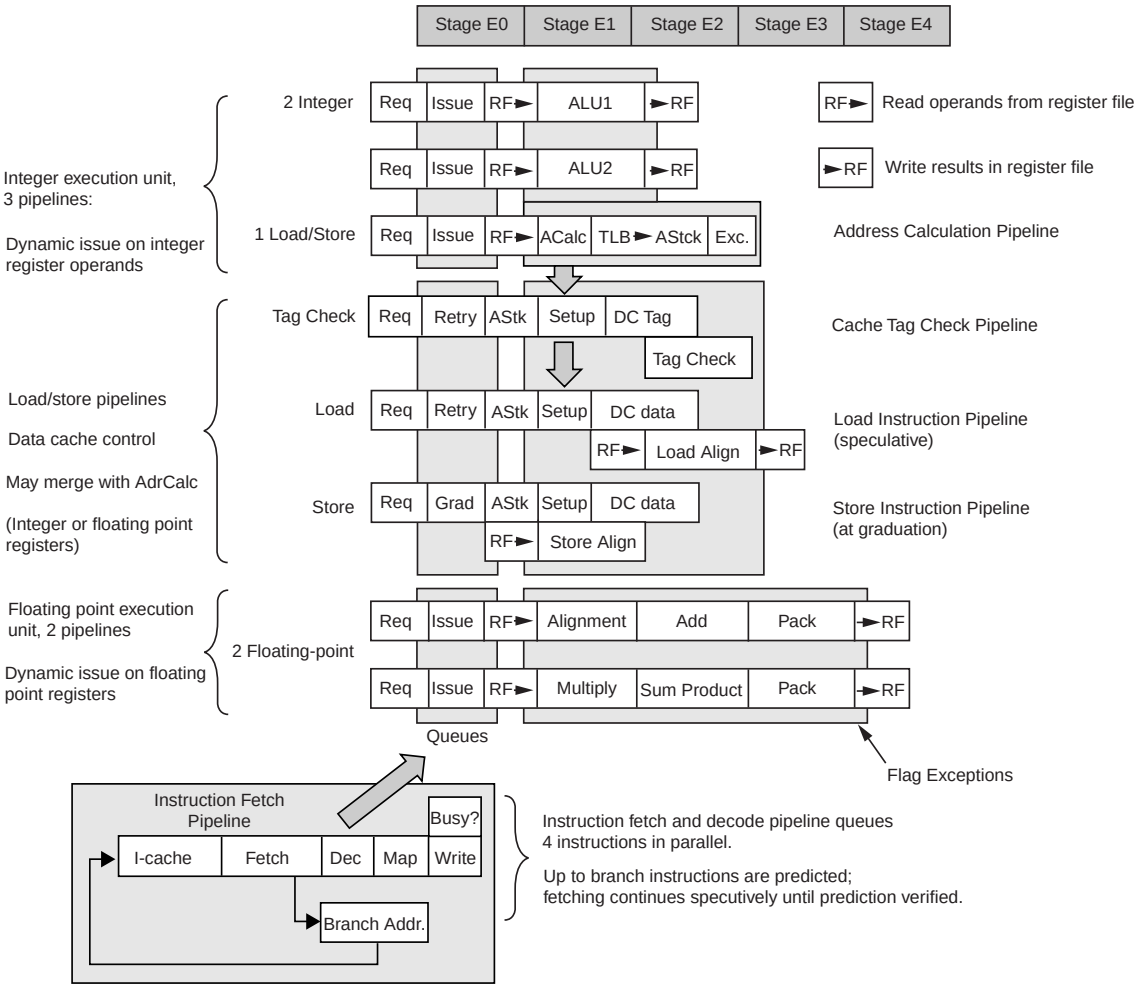

Figure 11b. MIPS R12000 Timing Pipeline

On MIPS R12000 the active list has been increased 4x, from 32 to 48 entries. Increasing the size of the active list allows more commands to be either waiting in queues for execution or be executing in the instruction units. When encountering a branch, the increase also allows the processor to speculatively execute more commands, which is also called deeper speculation. In most situations, both these effects will yield an increase in application performance.

The address queue now has separate pipelines for address calculation and tag checking. This improves overall efficiency and can improve application performance. For example, if the cache tag memory is busy, the address calculation can independently determine the cache bank, which can subsequently save a tag check cycle.

On MIPS R10000, transferring data from the system bus to the secondary cache locked up the cache controller for a significant period while a complete cache line was read in from main memory. Since the system bus clock rate is significantly slower than the processor clock rate, a full cache line transfer can cause the secondary cache controller to wait dozens of cycles before completion. During this time no other requests can be submitted. On MIPS R12000, transferring a cache line from the system bus to the secondary cache is divided into sub-blocks, four for the data cache and three for the instruction cache, with idle cycles between each sub-block.

During these idle cycles, other operations can be initiated by the secondary cache controller. If data arrives on the system bus during this time it will be buffered until the transfer resumes. Overall, this change substantially reduces the time spent by other operations waiting to be initiated. This improvement significantly helps performance for applications that frequently miss both primary and secondary cache.

Another large improvement in MIPS R12000 is a 2x increase in the size of the MRU, or way prediction table compared with MIPS R10000. This now allows a bit to be assigned to the way predict in each set of 4Mb of secondary cache. This larger table better predicts which way to search first, thus saving cycles when checking secondary cache. This improves application performance when secondary cache misses are a significant factor.

Two modifications to improve MIPS R12000 performance for branch instructions. First, a 4x increase in the length of the branch prediction table improves its accuracy. Second, a 32-entry Branch Target Address Cache products the target addresses for branch instructions. In MIPS R100000 every taken branch causes a 1-cycle bubble in the pipeline while the target address is calculated. This bubble is eliminated whenever the branch hits in the target cache. Branch instructions can comprise between 15% and 20% of the total number of instructions in many applications. Thus, the branch prediction table length increase and the branch target address cache can significantly contribute to improve overall application performance.

On MIPS R10000, instructions couldn't be decoded if the address queue became full. On MIPS R12000, all load, store, cacheop, and prefetch instructions are sent to the integer queue and issued to the address calculation unit. They are then removed from the integer queue and placed in the load/store queue. Although this puts additional instructions into the integer queue, they are usually issued and removed quickly. This change considerably simplifies the design of this component of the processor without affecting application performance.

Another improvement to MIPS R12000 is relaxed set locking of the data cache. On MIPS R10000 whenever a load or store instruction accesses a cache-line, it locks the line until the instruction graduates. If another instruction requires access to the same line before the first instruction is completed, both can share the same lock. If, however, another instruction needs a different line within the same cache set, it is stalled until the other instruction graduates. Allowing another instruction to lock the corresponding line in the other way of the same set could give rise to a deadlock situation because of out-of-order execution. To prevent such a deadlock situation, MIPS R10000 only allowed the oldest load or store instruction to obtain a lock on the other cache way. In certain situations this may degrade performance since the oldest load or store instruction may already own the lock on the first way, preventing further instructions from accessing the second way of the cache for that set. MIPS R12000 allows instructions to obtain a lock on the second way provided it is the oldest entry that does not already own a lock. Thus stop instructions that already own a lock do not prevent other younger instructions from accessing the second way. The net effect in certain situations is less contention in the data cache, which can yield a corresponding improvement in application performance.

3. OCTANE System Architecture Improvements

Figure 12 shows the OCTANE system architecture including MIPS R12000, with the architectural changes highlighted.

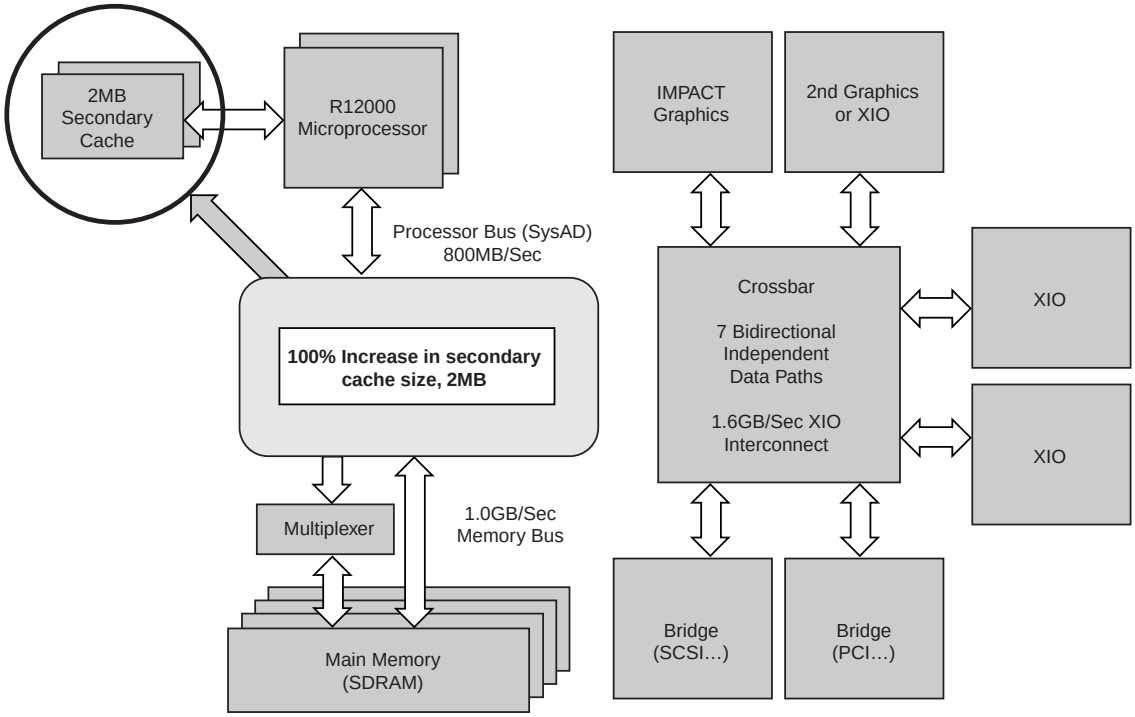

Figure 12. OCTANE System Architectural Changes with MIPS R12000

As can be seen from Figure 12 the key change is that the secondary cache size is doubled to 2MB. Increasing the secondary cache size increases the chances of the contents of a desired memory location being found in cache, reducing the number of times the CPU has to access main memory. This yields a noticeable improvement in application performance.

4. The Benefits of MIPS R12000 and OCTANE Architectural Changes on Application Performance

This section shows how the combined effect of the MIPS R12000 microprocessor and OCTANE system architectural improvements influence application performance on a real-world example. The application used was CATIA Version 4, supplied by Dassault Systemes, and the test was a SolidE update operation performed on a crankshaft. The model is shown in Figure 13.

The SolidE update operation automatically updates necessary geometry after a change has been made to some aspect, or aspects, of a model. These changes may include modifications such as changing the radius of a fillet or the diameter of a hole. From a system perspective the SolidE update operation is a very CPU-intensive operation and will typically utilize the CPU very close to 100%, provided there is sufficient physical memory to prevent paging.

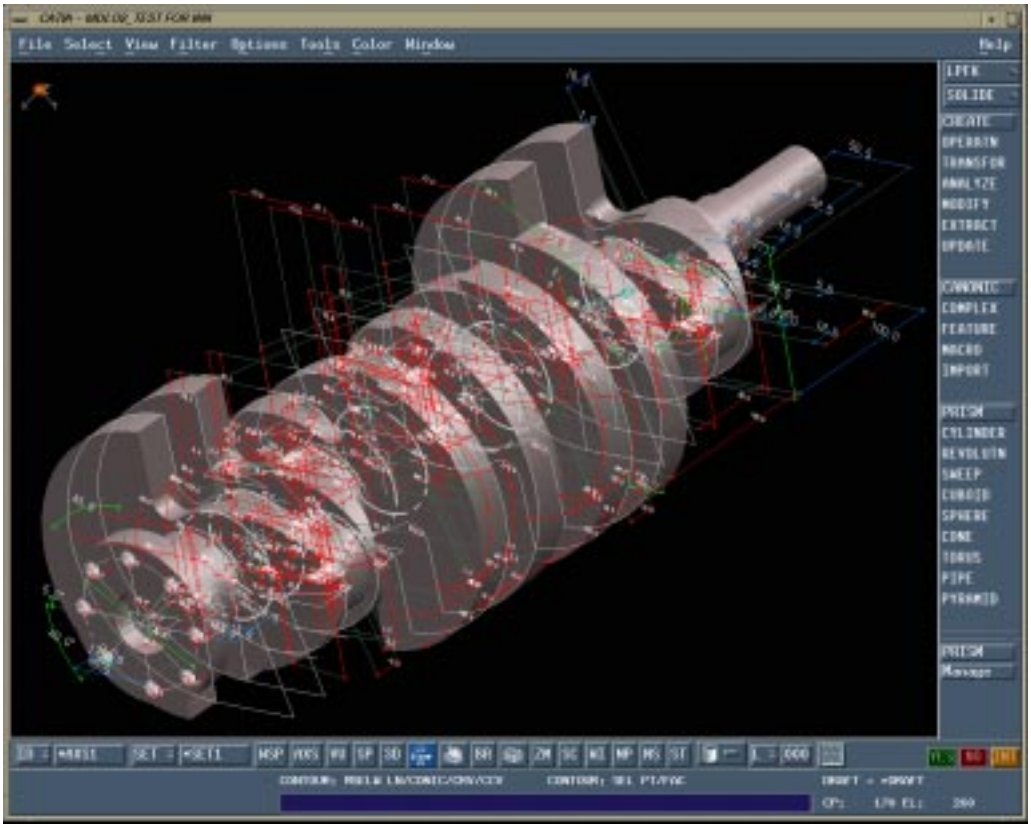

Figure 13. Crankshaft Model Used for SolidE Update Test, Supplied Courtesy of DaimlerChrysler Corporation

The MIPS R10000 and MIPS R12000 microprocessors include performance counters which can be used to measure internal performance. These counters measure metrics associated with cache performance, instructions executed, etc., and were used to generate results for the CATIA SolidE update test. Some counters changed slightly between MIPS R10000 and MIPS R12000 and where this affects comparisions the differences are explained.

The results generated on both MIPS R10000 and MIPS R12000 used the same OCTANE chassis, with the same operating system, installed on the same physical disks and using the same CATIA installation. In fact, the only change between the tests for both microprocessors was to physically swap the processor modules.

To faithfully represent how a user would use the system, the SolidE update test was run in multiuser mode and hence additional background processes, such as the CATIA license daemon, were running concurrently with the main CATIA SolidE update task. The effect of these background processes was found to be negligible, since over several consecutive runs the variation in elapsed time was significantly less than 0.5%. Likewise, it's also true that such background processes may also indirectly influence the test by causing subtle changes in cache contents, etc. Such influences were again found to negligible since the method by which the processor metrics presented below are gathered only records results for the process of interest. Furthermore, several consecutive runs of the CATIA SolidE update test showed variations to be minimal.

Table 1 summarizes the CPU and overall system elapsed times seen during the SolidE update test for OCTANE R10000 250 MHz 1MB secondary cache and OCTANE R12000 300 MHz 2MB secondary cache.

	OCTANE R10000 200 MHz, 1MB Secondary Cache	OCTANE R12000 300 MHz, 2MB Secondary Cache	Speedup Relative to MIPS R10000
Measured CPU time (seconds)	21.220	16.950	1.25x
Measured overall system elapsed time (seconds)	22.880	17.990	1.27x

Table 1. Summary of Measured CPU and Overall System Elapsed Times for CATIA SolidE Update Test

Clearly the combined effect of MIPS R12000 processor and OCTANE system architectural improvements demonstrates an overall performance improvement better than the 1.2x clock rate increase. This rest of this section takes a closer look at how the processor and system architectural differences yield this result.

Increased Active List Length

The number of processor cycles, the issued/decoded instructions, and the graduated instructions and floating point instructions observed during the CATIA SolidE update test for both MIPS R10000 and MIPS R12000 are shown in Table 2.

	OCTANE R10000 250 MHz, 1MB secondary cache	OCTANE R12000 300 MHz, 2MB secondary cache
Issued/decoded instructions	4.344E9	5.723E9
Graduated instructions	4.387E9	4.363E9
Graduated floating point instructions	0.521E9	0.485E9
Cycles	5.241E9	4.900E9

Table 2. Summary of Processor Cycles During Execution, Issued/Decoded Instructions, and Graduated Instructions During CATIA SolidE Update Test for Both MIPS R10000 and MIPS R12000

As can be seen from the results, the issued/decoded instructions for MIPS R12000 are significantly greater than for MIPS R10000. The 50% increase in the active list from 32 to 48 entries contributes to this increase by allowing more instructions to be speculatively executed. Several other features of MIPS R12000 may indirectly contribute as well:

- Splitting transfers from main memory to secondary cache allows the secondary cache controller to schedule other requests, which in turn allows more load and store instructions to be scheduled.
- The branch target address cache can potentially allow target addresses to be found more quickly, allowing more instructions to be speculatively executed.
- Relaxing the data cache set locking will reduce contention in the cache controller across both ways, thus also potentially allow more load and store instructions to be issued.

The effect of these changes, though, is likely to be more subtle compared with the direct effect of increasing the active list.

These comparisons are influenced by minor changes in the performance counters between MIPS R10000 and MIPS R12000. On MIPS R12000 decoded instructions represent a slightly different metric compared with issued instructions on MIPS R10000. The difference is that issued instructions counts instructions issued to the execution units. This doesn't include certain instructions, such as no-ops. This may in turn yield a slight increase in the MIPS R10000 count. Decoded instructions, however, are considered a more accurate measure and was the reason why the metric was changed.

Even though both microprocessors ran exactly the same code on exactly the same operating system, the slight reduction in graduated instructions, graduated floating point instructions, and cycles is likely to arise due to indirect effects of some of the other changes in MIPS R12000. The direct effects of these changes are covered later. Reduced cache contention combined with fewer misses may lead to fewer graduated load and store instructions and better branch prediction will likely yield fewer instructions and floating point instructions graduating. Both factors are likely to contribute to a reduction in cycles.

Separate Address Queue Pipelines and Use of Integer Queue for Load/Store Address Calculation

The issued/decoded loads and stores sent to the address calculation unit, along with the graduated loads and stores from the address calculation unit, are shown in Table 3.

	OCTANE R10000 250 MHz, 1MB secondary cache	OCTANE R12000 300 MHz, 2MB secondary cache
Issued/decoded loads	1.200E9	1.272E9
Graduated loads	1.110E9	1.116E9
Issued/decoded stores	424.9E6	424.2E6
Graduated stores	395.6E6	396.4E6

Table 3. Summary of Issued/Decoded Loads, Graduated Loads, Issued/Decoded Stores, and Graduated Stores to/from the Address Calculation Unit Seen During CATIA SolidE Update Test for Both MIPS R10000 and MIPS R12000

The expected result of separating the address calculation and tag checking pipelines on MIPS R12000 would be a slight increase in the issued load and store operations. The increased speculation depth on MIPS R12000 may also yield additional load and store instructions being issued. In line with these predictions, the results in Table 3 show a slight increase in issued/decoded load instructions between MIPS R10000 and MIPS R12000. The increase is clearly small. It should be remembered that the specific section of executing code and data will exhibit a deterministic effect on the number of load and store instructions relative to other instructions, and on the dynamics of cache-line refills, etc. Other background processes, however, were run concurrent to the test, and these may cause nondeterministic effects on the cache yielding slight variations in these results. Also, as mentioned in the subsection above, comparing issued load and store instructions on MIPS R10000 with decoded loads and stores on MIPS R12000 may also yield slightly higher values for MIPS R12000.

Doubling Secondary Cache MRU Table

The instruction and data mispredictions arising in the secondary cache way prediction are summarized in Table 4.

	OCTANE R10000 250 MHz, 1MB secondary cache	OCTANE R12000 300 MHz, 2MB secondary cache
Instruction misprediction from scache way table	3.463E6	2.414E6
Data misprediction from scache way prediction table	3.491E6	2.939E6

Table 4. Summary of Instruction and Data Misprediction from Secondary Cache Way Prediction Table During CATIA SolidE Update Test for Both MIPS R10000 and R12000

As described above, doubling the size of the MRU table should yield significantly more accurate predictions on which way to search first in the secondary cache. As the results in Table 4 clearly demonstrate, there is a significant reduction in misprediction arising from the way prediction table for both secondary data and

instruction caches on MIPS R12000 compared with MIPS R10000. Since each misprediction introduces a delay of several cycles, this improvement contributes to an overall improvement in application performance when a large number of primary cache misses occur.

Relaxed Data Cache Set Locking

Primary and secondary data cache misses, along with quadwords written back from both primary data and secondary cache, are summarized in Table 5.

	OCTANE R10000 250 MHz, 1MB secondary cache	OCTANE R12000 300 MHz, 2MB secondary cache
Primary data cache misses	36.681E6	35.663E6
Quadwords written back from primary data cache	20.848E3	20.246E3
Store/prefetch exclusive clean block in scache	660.2E3	282.3E3
Secondary data cache misses	8.772E6	7.381E6
Quadwords written back from scache	15.107E3	8.822E3

Table 5. Summary of Primary and Secondary Data Cache Misses along with Quadwords Written Back from Primary Data Cache and Quadwords Written Back from Secondary Cache During CATIA SolidE Update Test for Both MIPS R10000 and R12000

As mentioned above, relaxing the data cache set locking on MIPS R12000 could yield a reduction in contention when accessing data cache, which may in turn lead to fewer cache misses. The results in Table 5 show reduced misses for both the primary and the secondary data caches. It should also be remembered, however, that as is described below, doubling the secondary cache size is also likely to significantly reduce the secondary cache misses.

Reduced contention when accessing the data cache may also mean that load and store instructions experience fewer delays when accessing cache. This may cause fewer requests to replace cache lines and may also yield a reduction in the number of cache-lines marked dirty and hence written back to either secondary cache or main memory. Clearly the results for the quadwords written back from primary data cache and secondary cache shown in Table 5 reflect this result. The influence of increasing secondary cache size, however, should again be noted.

4x Increase in Branch Prediction Table and Branch Prediction Based on History

The mispredicted branches and decoded/resolved conditional branches occurring during the CATIA SolidE update test are summarized in Table 6.

	OCTANE R10000 250 MHz, 1MB secondary cache	OCTANE R12000 300 MHz, 2MB secondary cache
Mispredicted branches	118.5E6	103.6E6
Decoded/resolved conditional branches	484.6E6	495.0E6

Table 6. Summary of Mispredicted Branches and Decode/Resolved Conditional Branches During CATIA SolidE Update Test for Both MIPS R10000 and MIPS R12000.

By increasing the branch prediction table size, the number of stored branch instructions, along with their predicted target addresses, can be increased. As mentioned above, large applications can show branch instructions comprising between 15% and 20% of all instructions, so being able to store a greater number of predicted branch paths is likely to yield a noticeable increase in correctly predicted branch choices. The mispredicted branch results shown in Table 6 clearly confirm these expectations.

The decoded/resolved branches shown in Table 6 reflect the number of branches both correctly and incorrectly predicted and hence, like the number of mispredicted branches, this result would be expected to decrease as a result of the larger branch prediction table on MIPS R12000. The reason the results don't reflect this, however, is because on MIPS R10000 the number of branches decoded is only incremental by one in one cycle. Sometimes there may be multiple floating point conditional branches verified in one cycle, and hence in this situation MIPS R10000 doesn't reflect the correct total number of branches resolved. On MIPS R12000, however, the number of resolved conditional branches is incremented by the total number of conditional branches resolved and hence more accurately reflects the true result. This is why MIPS R12000 does not show the trend expected for the resolved conditional branch results.

2x Increase in Secondary Cache Size

Primary instruction and data cache misses, secondary data and instruction cache misses, and quadwords written back to primary and secondary caches during the CATIA SolidE update test are summarized in Table 7.

	R10000 250 MHz, 1MB secondary cache	R12000 300 MHz, 2MB secondary cache
Primary instruction cache misses	38.337E6	37.337E6
Primary data cache misses	36.681E6	35.663E6
Quadwords written back from primary data cache	20.848E3	20.246E3
Secondary instruction cache misses	2.557E6	1.398E6
Secondary data cache misses	8.772E6	7.381E6
Quadwords written back from scache	15.107E3	8.822E3

Table 7. Summary of Primary and Secondary Data and Instruction Cache Misses, as well as Quadwords Written Back from Primary and Secondary Cache During CATIA SolidE Update Test for Both MIPS R10000 and MIPS R12000

As would be expected, doubling the secondary cache should, in most situations, yield a noticeable reduction in secondary misses since it is more likely that requested data will be located in secondary cache. The results in Table 7 clearly demonstrate this for both secondary instruction and data caches. The reduction in secondary instruction cache misses compared with the reduction in secondary data cache misses, however, will clearly be very dependent on both the specific section of code along with any associated data.

As mentioned above, quadwords written back from scache represent the data transferred from secondary cache to main memory. Clearly given a noticeable reduction in the number of secondary cache misses, a corresponding reduction in the amount of data being written back to main memory is also likely, although it isn't guaranteed. The results in Table 7 show a significant reduction in quadwords written back from secondary cache. Since this reduction is noticeably higher than the reduction in secondary cache misses, it's likely that other improvements in MIPS R12000 over MIPS R10000, such as the relaxed data cache set locking described above, may also be influencing this result.

To place the significance of the OCTANE secondary cache size improvements in perspective, both primary and secondary cache results are included in Table 7. Given that the cost in terms of time spent waiting for data as a result of a secondary cache miss is between five and 10 times greater compared with a primary cache miss, it is easy to see why a reduction in secondary cache misses is significant in terms of application performance.

Figures 14a and 14b show the measured secondary cache misses over the complete duration of the SolidE Update test on both MIPS R10000 and MIPS R12000, respectively.

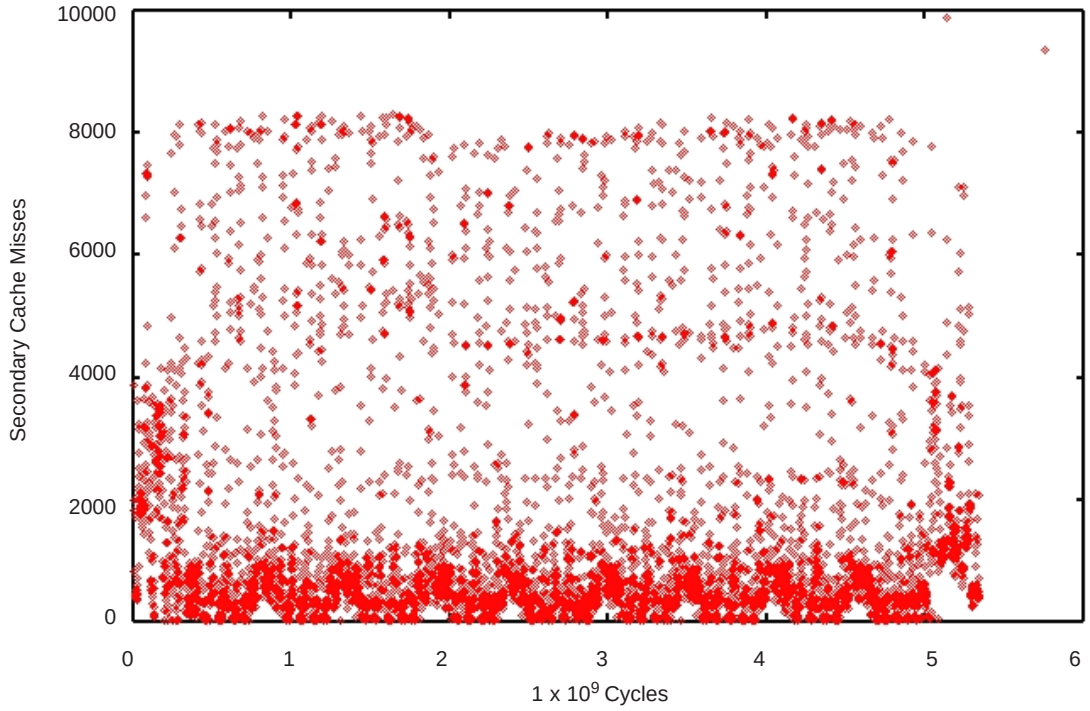

Figure 14a. Observed Secondary Cache Misses Seen on MIPS R10000 During CATIA SolidE Update Test.

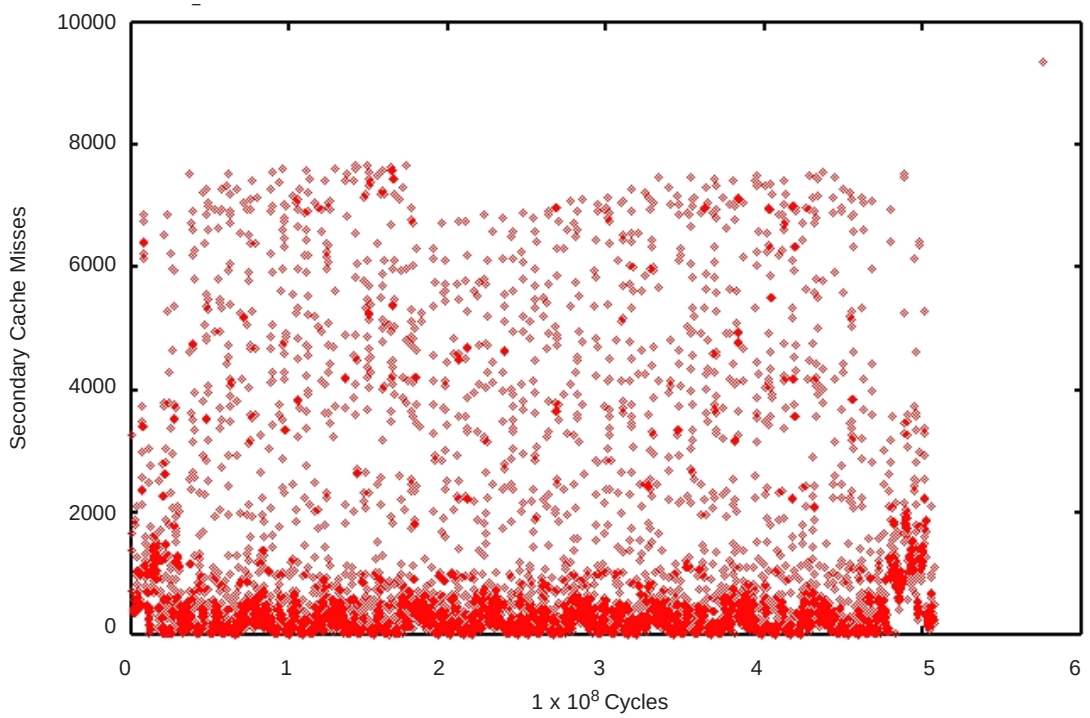

Figure 14b. Observed Secondary Cache Misses Seen on MIPS R12000 During CATIA SolidE Update Test

Although it's perhaps slightly difficult to see from the raw data, Figures 14a and 14b show the average value for secondary cache misses for MIPS R12000 are slightly lower compared with those of MIPS R10000. This concurs with the overall results presented in Table 7 above and the net effect of this is an improvement in application performance.

5. Summary—OCTANE on the Competitive Horizon

The previous section demonstrated the benefit to the end user through application performance of architectural improvements in both MIPS R12000 and OCTANE. In isolation, however, such information doesn't place these improvements into context, since it leaves the question of competitive positioning unanswered. This section aims to address this by comparing the performance of OCTANE with the MIPS R12000 microprocessor to current and recently announced competitive systems.

Figure 15 shows the peak and base SPECint95 results for the OCTANE with MIPS R12000 along with the recently announced competitive processors and platforms from Sun, IBM, and HP, including the 450 MHz UltraSPARC II and the HP PA-8500. Likewise, Figure 16 shows the peak and base SPECfp95 results for the same platforms.

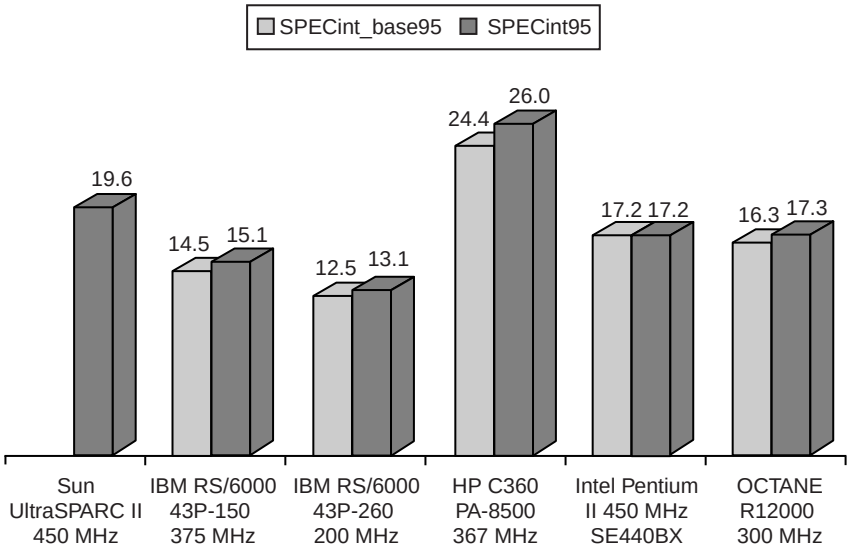

Figure 15. Comparison of MIPS R12000 SPECint95 Results with Competitive Systems

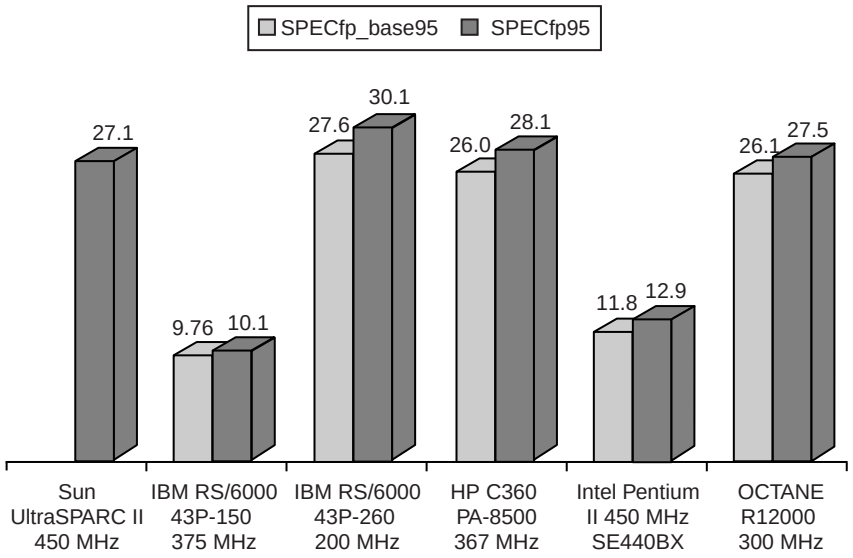

Figure 16. Comparison of R12000 SPECint95 Results with Competitive Systems

Clearly the integer performance demonstrated by both the Sun UltraSPARC II 450 MHz and, even more so, the HP PA-8500 367 MHz at first sight appear impressive. The SPECint95 baseline results for the UltraSPARC II 450 MHz are not known at the time of writing, however [7]; as the baseline SPECint95 results for the UltraSPARC II 360 MHz show (see Figure 2), it's likely that there is a significant gain through aggressive compiler optimizations. As a consequence, although MIPS R12000 appears to be behind the UltraSPARC II 450 MHz in integer performance, in situations where compiler optimizations yield little improvement, as is often the case with many operations in large applications, it is clearly well positioned to provide strong competitive performance.

Compared with both the Sun and HP microprocessors, it should also be appreciated that the SPECint95 tests typically sit very comfortably in cache. This is rarely the case with large applications, such as CATIA. Consequently, the SPECfp95 results may present a better indication of how the performance of how all microprocessors will be affected when secondary cache misses become more significant, thus forcing more accesses to main memory. Clearly, the SPECfp95 results show that MIPS R12000 performance is very well positioned against all platforms.

As a further illustration of this point, Tables 8a, b, and c compare the increase in SPECint95 and SPECfp95 among MIPS R12000 300 MHz, UltraSPARC II 450 MHz, and the PA-8500 microprocessors with previous-generation microprocessors in each case.

	Clock Rate	SPECint95	SPECfp95
MIPS R10000	250 MHz	13.6	20.3
MIPS R12000	300 MHz	17.3	27.5
Increase	20%	27%	35%

Table 8a. Summary of SPECint95 and SPECfp95 Results for MIPS R12000 Relative to MIPS R10000

	Clock Rate	SPECint95	SPECfp95
Sun UltraSPARC II	360 MHz	16.1	23.5
Sun UltraSPARC II	450 MHz	19.6	27.1
Increase	25%	22%	15%

Table 8b. Summary of SPECint95 and SPECfp95 Results for UltraSPARC II 450 MHz Relative to UltraSPARC II 360 MHz Microprocessor

	Clock Rate	SPECint95	SPECfp95
HP PA-8200	240 MHz	17.3	25.4
HP PA-8500	367 MHz	26.0	28.1
Increase	53%	50%	11%

Table 8c. Summary of SPECint95 and SPECfp95 Results for PA-8500 Relative PA-8200 Microprocessor

Clearly these results demonstrate that MIPS R12000 is the only microprocessor to demonstrate a speed up higher than clock rate when compared with the previous-generation processors on both SPECint95 and SPECfp95 results.

The floating point increase of the HP C360 compared with the HP C240 can only be described as disappointing. When announced, the PA-8500 quoted SPECfp95 performance on the order of 40-60 [8]; however, the C-class architecture is clearly a major limitation. This probably arises because, as mentioned above, the SPECfp95 tests cause a significantly number of cache misses compared with the SPECint95 tests and the resulting accesses to main memory amplify limitations in the C-class architecture. Secondary cache misses, however, frequently occur when running large applications. The overall improvement in application performance offered by the C360 PA-8500 360 MHz compared with the C240 PA-8200 240 MHz, therefore, is likely to lie somewhere between the improvements seen for the SPECint95 and SPECfp95.

Another factor affecting the improvement between the HP C240 and HP C360 is that the PA-8500 microprocessor includes 1.5Mb of on-chip single-level cache compared with 4MB off-chip single-level cache on the PA-8200 microprocessor. Clearly for many application codes this will yield an increase in the number of cache misses; however, in fairness to the PA-8500 microprocessor, it does incorporate modifications to compensate. The latency of fetching data from cache in the PA-8500 has been significantly reduced compared with the PA-8200 and the cache has also been changed from direct mapping to four-way set associative, which will typically yield a noticeable increase in hit rates.

Like the HP C240 though, the HP C360 again clearly benefits from aggressive compiler optimization. Figure 17 clearly demonstrates this by showing the ratio of peak-to-base SPECfp95 results across all tests and shows large similarities to the data presented in Figure 4.

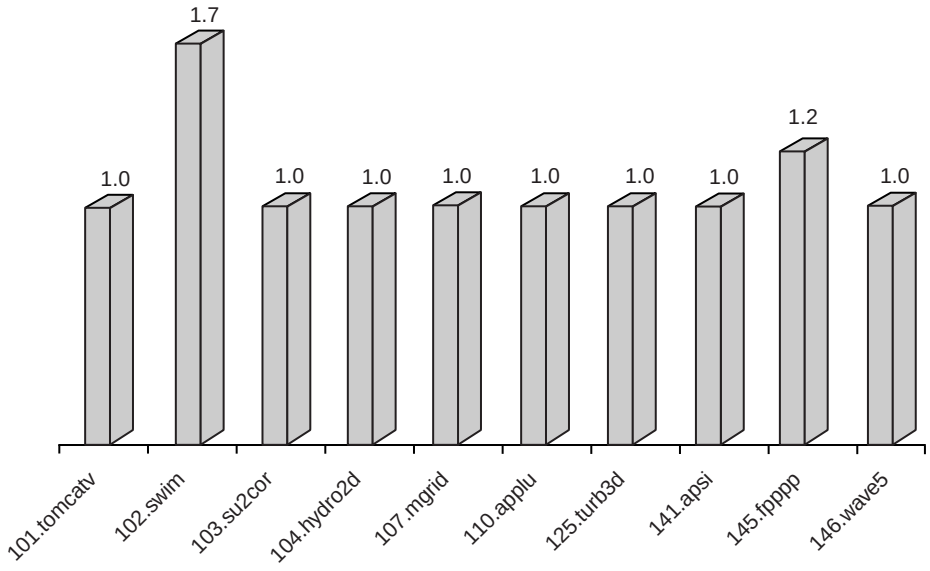

Figure 17. Ratio of Peak/Base SPECfp95 Results Across All Tests for HP PA-8500 Microprocessor

As previously discussed in the introduction, the IBM Power3 200 MHz microprocessor in the R/S6000 43P-260 system presents strong competition for MIPS R10000 250 MHz in OCTANE. With MIPS R12000, however, the SPEC results shown in Figures 13 and 14 clearly demonstrate that the advantage is severely eroded, if not eradicated. What was previously competitive integer performance is now noticeably lagging, and what was once a significant floating point lead is now a marginal advantage.

Some of the features of the Power3 design are very similar to those of MIPS R10000 and MIPS R12000, although MIPS R10000 was first introduced into the Indigo2 and OCTANE systems over two years prior to the 43P-260 being released. An example of such features [9], is that the Power3 can fetch and decode up to four instructions in one cycle. Also, like both MIPS R10000 and MIPS R12000, it uses register renaming and out-of-order execution; however, its branch prediction model is different in that it separates instructions that generate branch conditions from instructions that change program flow. This allows branch conditions to be generated in advance although the benefit will clearly be very dependent on the section of code executing.

Part of the Power3's floating point performance can be explained because it allows up to eight instructions to begin execution in one cycle: two floating point instructions, two load/store instructions, two single-cycle integer instructions, a multiple-cycle integer instruction, and a branch instruction. Compared with the five that can be issued in one cycle on MIPS R12000 (see Section 2.1), this is certainly likely to contribute to the Power3's slight advantage in the SPECfp95 results. Its interesting to note, however, that to the user the advantage is not that significant since MIPS R12000 is able to demonstrate SPECfp95 results within 10% of the Power3's peak result. When the benefits of aggressive optimization are removed, to reveal perhaps a more typical performance of a microprocessor, the Power3's advantage is nearly cut in half to less than 6%. Clearly applications can be compiled and optimized to take advantage of such architectural features; however, typically improvements will be diluted by other factors such as an increase in the integer component of code, where MIPS R12000 actually demonstrates an advantage, along with the effects of memory latency, etc.

In summary, therefore, the SPECint95 and SPECfp95 results demonstrate that MIPS R12000 is very well positioned against all current competitive systems and microprocessors. Even compared with extreme cases such as the integer performance of HP's PA-8500 367 MHz or the floating point performance of IBM's Power3 200 MHz, the balanced architecture of MIPS R12000 300 MHz delivers strong overall performance. When

combined with OCTANE system architecture this yields significant improvements in application performance relative to MIPS R10000 250 MHz and firmly places OCTANE among the performance leaders in current UNIX® systems.

This paper has also demonstrated how the MIPS R12000 microprocessor and OCTANE system architecture improvements yield benefits in terms of application performance. Moving forward, OCTANE will continue to incorporate further microprocessor developments, such as MIPS R14000®, along with other system improvements, such as new graphics hardware. In the same way as has been shown in this paper, these improvements will result in application performance benefits, which are clearly of direct benefit to system users; however, since the improvements will be incorporated into existing hardware, they also protect current investment, yielding noticeable cost savings.

6. Acknowledgments

Any piece of work requires the help of many people. To this end I thank the following people for their contributions: John Schimpf, Kenneth Yeager, Keith Jaslow, Gary Walters, Eric Miller, Alexander Poulos, and Rob Jackson, all of Silicon Graphics, as well as Roberta Waterworth of DaimlerChrysler corporation.

7. References

1. "OCTANE Technical Report," Silicon Graphics, Inc.
2. The published results of the SPEC organization are located at *www.spec.org*.
3. "The MIPS R10000 Superscalar Microprocessor," Kenneth C. Yeager, IEEE Micro, April 1996.
4. "200-MHz Superscalar RISC Microprocessor," Nader Vasseghi, Kenneth Yeager, Egino Sarto, and Mahdi Seddighnezhad, IEEE Journal of Solid-State Circuits, Vol. 31, No. 11, November 1996.
5. "High-Performance Computer Architecture," 3rd Edition. Harold S. Stone, Addison-Wesley, 1993.
6. "Computer Architecture Single and Parallel Systems," Mehdi R. Zargham, Prentice Hall, 1996.
7. "Sun Announces 450 MHz and 400 MHz UltraSPAR-II Microprocessors," Sun Microsystems, Nov. 2, 1998.
8. "PA-8500's 1.5M Cache Aids Performance," Linely Gwennap Microprocessor Report, Nov. 17, 1997.
9. "IBM's Power3 to Replace P2SC," Peter Song, Microprocessor Report, Nov. 17, 1997.

Corporate Office
2011 N. Shoreline Boulevard
Mountain View, CA 94043
(650) 960-1980
www.sgi.com

U.S. 1(800) 800-7441
Europe (44) 118-925.75.00
Asia Pacific (81) 3-54.88.18.11
Latin America 1(650) 933.46.37

Canada 1(905) 625-4747
Australia/New Zealand (61) 2.9879.95.00
SAARC/India (91) 11.621.13.55
Sub-Saharan Africa (27) 11.884.41.47

2232 (2/99)