Mips vs. Lexra: Definitely Not Aligned

Patent Lawsuit Hinges on Unusual Instructions in MIPS Architecture

by Tom R. Halfhill and Rich Belgard

From its start in 1997, Lexra has pursued a simple business plan: design embedded processor cores that are largely compatible with the popular MIPS architecture, license the cores as synthesizable models that are easily portable to different IC processes, undercut the cost of a MIPS license, and avoid legal entanglements with Mips Technologies. Lexra has done a good job of executing the first parts of that strategy, but Mips keeps foiling the last objective.

For the second time in two years, Mips has filed a lawsuit against Lexra (see MPR 11/15/99, p. 16). In 1998, Mips sued Lexra over trademark issues and product claims. That led to an out-of-court settlement in which Lexra agreed to drop the letter "R" from its product names (for example, the LXR4080 became the LX4080) and to stop promoting its cores as "MIPS compatible." After that skirmish, Lexra may have thought the worst was over. But in October Mips filed a patent-infringement lawsuit that's a new threat to Lexra's future. The patent lawsuit strikes at the heart of Lexra's technology, alleging that the cores violate some key Mips patents.

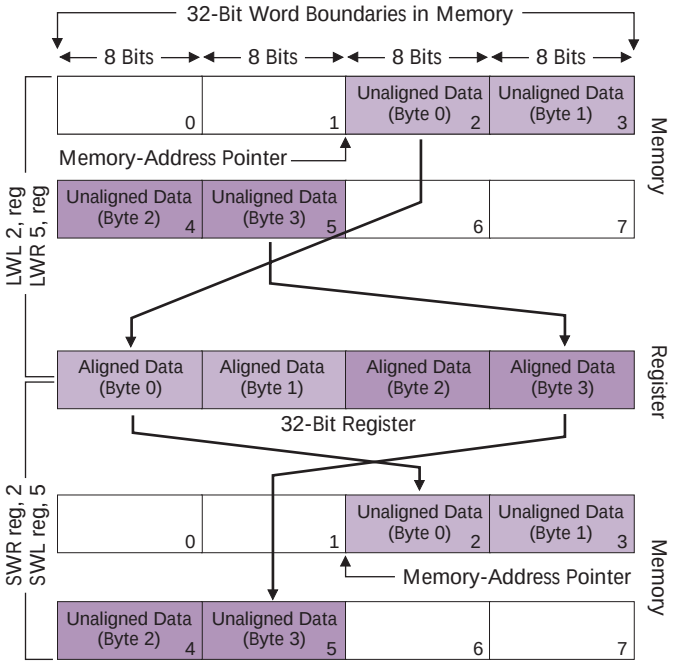

Figure 1. This example shows how the LWL and LWR instructions load two fragments of unaligned data from memory, shift them into word alignment, and merge them in a register. The SWR and SWL instructions reverse the process, storing the data to memory across a word boundary. For this example, memory addressing is big-endian, as are all of Lexra's cores.

The lawsuit's timing seems calculated. Mips and Lexra have been negotiating terms for a MIPS license for several months, and Mips filed the complaint after the talks reached an impasse, so it could be interpreted as a negotiating tactic. Also, Lexra is proving to be a feisty competitor. Lexra has signed up 17 licensees, has beaten Mips to the punch with a synthesizable core by a whole year, and has introduced new DSP extensions and other innovative features. Nevertheless, the lawsuit raises important questions about Lexra's technology, the Mips patent portfolio, some unusual features of the MIPS architecture, and the general feasibility of building independent CPUs that execute a popular instruction set.

In some ways, the controversy is reminiscent of the interminable battles that AMD fought with Intel over x86 microcode and other issues. We think the outcome will be similar. Eventually, an out-of-court settlement will save face and legal costs for both parties without resolving the larger questions about technology patents, instruction-set compatibility, and the intellectual property embodied in microprocessors. Still, it's worth examining the case in more depth, if only because the popularity of successful architectures is bound to attract similar competition in the future.

Unaligned Loads and Stores

The Mips complaint alleges that Lexra is infringing on at least two and possibly as many as eleven Mips U.S. patents. The two patents at the center of the complaint are 4,814,976 and 5,864,703. Mips applied for the '976 patent in 1986, during the Bronze Age of microprocessors; it was granted in 1989. Mips applied for the '703 patent in 1997; it was granted last January. The '976 patent describes load and store instructions that handle data not aligned on word boundaries in memory. The '703 patent describes technology for reducing the loss of precision while performing arithmetic operations in single-instruction, multiple-data (SIMD) format.

The load/store instructions that Mips claims are covered by the '976 patent were also a sticking point in the 1998 lawsuit over trademark issues. There are four relevant instructions: LWL (load word left), SWL (store word left), LWR (load word right), and SWR (store word right).

Figure 1 shows how these instructions work. LWL and LWR allow a program to load a 32-bit word of data that straddles a 32-bit boundary in memory, shift the unaligned fragments into proper alignment, and merge them in a register. Likewise, SWL and SWR allow a program to store a 32-bit word across a memory boundary. These load and store operations accomplish in two instructions (and only two bus cycles) what would otherwise require about 20 instructions and many more cycles.

Lexra's documentation states that its cores don't support unaligned loads and stores in hardware. (Indeed, this was another condition for settling the 1998 lawsuit.) The LX4080 data sheet states: "...the LX4080 executes MIPS-I instructions with the following exclusions: the unaligned loads and stores (LWL, SWL, LWR, SWR) are not supported because they add significant silicon area for little benefit in most applications. Occasional unaligned data handling is more cost-effectively handled without these specialized ops."

Although this makes it clear that the LX4080 doesn't support unaligned loads and stores in hardware, it's likely that the reason is more legal than technical. The instructions would have a greater impact on critical speed paths than on silicon area, because they must perform a load or store, a shift, and a merge in a single cycle. But what's even more likely than either technical explanation is that Lexra was aware of the 1989 patent when designing its cores and sought to avoid a confrontation with Mips.

Instead of supporting unaligned load/store instructions in hardware, Lexra provides licensees with a special trap handler and a C library to perform equivalent operations in software. The unsupported instructions trigger a reserved-instruction trap that executes a routine of alternate instructions, as Figure 2 shows. Although Lexra supplies the code in C for easier maintenance, the figure shows the assembly output of a GNU compiler for illustrative purposes.

The Compatibility Controversy

How common are unaligned load/store operations? It depends greatly on the application. Programs that mix data types of different lengths, or that compress data by omitting unneeded bits, might be riddled with 32-bit words that straddle 32-bit boundaries. That kind of code, especially if it occurs in a critical inner loop, could impair the performance of a Lexra core, because the routines shown in Figure 2 are only a small part of the total code executed during a reserved-instruction trap. The operating system's exception handler adds hundreds of instructions to the detour.

To avoid that penalty, a compiler can align data on word boundaries by padding unaligned words with zeroes. Of course, this inflates the size of the data, which can be significant for some memory-starved embedded applications.

Lexra says unaligned loads and stores are rare in MIPS embedded software, and authoritative sources tend to agree. A discussion of the MIPS architecture in John L. Hennessy and David A. Patterson's *Computer Architecture: A Quantitative Approach* states: "A rare event in most programs, [the instructions are] included for COBOL programs where the programmer can force misalignment by declarations."

Not many embedded programmers are using COBOL these days. William Dally, a professor of electrical engineering and computer science at Stanford University, says unaligned data is rare in embedded programs for RISC and modern CISC architectures: "You always align the data for performance reasons, if nothing else."

There are some cases in which software tuned for the MIPS architecture uses unaligned loads and stores. One example is a special BSD version of the C library function memcpy, which makes liberal use of the instructions. The source code for this function is listed in Dominic Sweetman's *See MIPS Run*, a detailed textbook on the MIPS architecture.

In any case, the need for a special trap handler and emulation routine does mean that Lexra's cores are not 100% MIPS compatible. That's why Lexra agreed to stop referring to its cores as "MIPS compatible" when settling the 1998 lawsuit. When we applied that term to Lexra's new LX4280 in a recent article (see MPR 8/2/99, p. 13), we received a polite but firm letter expressing the concern of the Mips legal department. The letter said our description was inaccurate "because (at least) the LX4280 does not support MIPS unaligned load and store instructions in hardware."

Point well taken. But it's difficult to reconcile that point with Mips's patent-infringement lawsuit, which appears to argue that Lexra supports unaligned loads and stores to a degree that violates the '976 patent. Are Lexra's cores MIPS compatible after all? Mips says no, but won't elaborate. Evidently, Mips believes that achieving the same result in software isn't sufficient to make Lexra's cores MIPS compatible, but is sufficient to infringe on the patent.

The '976 patent contains 14 claims. Claims 1–3 are independent apparatus claims: claim 1 covers loading hardware; claim 2 covers storing hardware; and claim 3 covers loading and storing hardware. These claims use "means plus function" language, so Lexra's implementations would likely have to be very similar to the patented implementations—which describe hardware.

Claims 9 and 12 appear more applicable to software because they are independent method claims. Claim 9 appears

Emulating an Unaligned Load

```
/* LWL */
SUBU    $v0,$a0,$a2
LW      $a1,0($v0)
LI      $v0,-1
SLLV    $v0,$v0,$v1
NOR     $v0,$zero,$v0
AND     $v0,$a3,$v0
SLLV    $a1,$a1,$v1
OR      $a3,$v0,$a1
JR      $ra
SW      $a3,0($t0)

/* LWR */
SUBU    $v0,$a0,$a2
LI      $v1,3
SUBU    $v1,$v1,$a2
SLL     $v1,$v1,0x3
LW      $a1,0($v0)
LI      $v0,-1
SRLV    $v0,$v0,$v1
NOR     $v0,$zero,$v0
AND     $v0,$a3,$v0
J       bfc118a4
SRLV    $a1,$a1,$v1
```

Emulating an Unaligned Store

```
/* SWL */
SUBU    $a0,$a0,$a2
SLL     $v1,$a2,0x3
LI      $v0,-1
SRLV    $v0,$v0,$v1
NOR     $v0,$zero,$v0
LW      $a1,0($a0)
J       bfc1191c
SRLV    $v1,$a3,$v1

/* SWR */
LI      $v1,3
SUBU    $v1,$v1,$a2
SLL     $v1,$v1,0x3
LI      $v0,-1
SLLV    $v0,$v0,$v1
NOR     $v0,$zero,$v0
LW      $a1,0($a0)
SLLV    $v1,$a3,$v1
AND     $a1,$a1,$v0
OR      $a3,$v1,$a1
JR      $ra
SW      $a3,0($a0)
```

Figure 2. This emulation code executes when a Lexra core traps the unaligned load/store instructions (LWL, LWR, SWL, SWR).

to cover any series of operations that loads an unaligned piece of data from memory, shifts the data to a word boundary, and merges it with another piece of data so they're properly aligned. Claim 12 appears to cover the corresponding store situations. If a court rules that these claims apply to Lexra's software emulation, Lexra may have a problem.

It all depends on how broadly a court interprets the claims. Microprocessors were loading and storing unaligned data long before Mips applied for the '976 patent in 1986. Early programmers paid little attention to word boundaries, because memory conservation was more important and CISC instruction sets didn't encourage rigid data alignment. As Sweetman points out in *See MIPS Run*, "Because CISC architectures such as the MC680x0 and Intel x86 do handle unaligned loads and stores, you may come across this as a problem when porting software."

Dally, the Stanford professor, recalls working with a 68000 exception handler in the early 1980s that performed unaligned loads and stores almost exactly the way Lexra does. Later 68K (and all x86) processors handle unaligned loads and stores in hardware, often with microcode.

The 68K and x86 architectures preceded the MIPS architecture (and Mips's filing of the '976 patent) by at least seven years. For Mips to prevail on this count, a court will have to interpret the patent broadly enough to include Lexra's method for loading and storing unaligned data but narrowly enough to exclude the methods used by other architectures that have existed since the 1970s.

Extended-Precision SIMD Instructions

Another count in the Mips complaint refers to the '703 patent, which covers SIMD operations with intermediate extended precision. This patent describes the MDMX (MIPS digital-media extensions, also known as Mad Max) technology that Mips announced in 1996 (see MPR 11/18/96, p. 24).

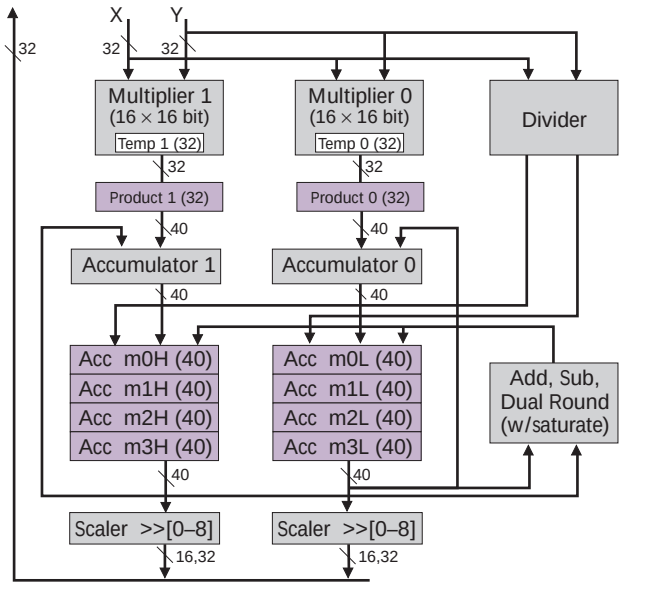

Figure 3. The LX5280's dual MAC units support SIMD instructions and use 40-bit-wide accumulators for intermediate results.

MDMX is similar to Intel's MMX—it adds new multimedia instructions and defines new registers that are mapped onto the existing floating-point registers. Both architectural extensions allow SIMD instructions to operate on multiple 8- or 16-bit integers packed into a 64-bit register. Unlike MMX, however, MDMX holds intermediate results in an extended-precision format to prevent lost precision due to overflows and underflows.

To hold those intermediate results, MDMX defines a 192-bit accumulator, which can hold eight 24-bit values or four 48-bit values. When an instruction finishes its arithmetic operations, it converts the extended-precision intermediate result back to the original degree of precision by scaling, rounding, or clamping, which the '703 patent jointly refers to as "transforming."

All of the patent's claims are method claims, not apparatus claims, so they don't cover the hardware that executes MDMX instructions. Instead, they cover using the hardware to execute the instructions.

Ironically, in the three years since Mips announced MDMX, neither Mips nor any of its licensees has introduced a processor that uses the technology. NEC's MIPS-compatible VR5400 comes the closest by implementing similar multimedia extensions (see MPR 3/9/98, p. 1), but they aren't technically or legally the same as MDMX, and they were jointly designed with SandCraft, which at the time wasn't a Mips licensee. (SandCraft finally joined the fold on October 28, the same day Mips sued Lexra.)

The '703 patent still stands, of course, whether or not Mips ever implements MDMX. But this is another way Lexra may have annoyed Mips. Although Lexra doesn't support MDMX, Lexra has announced a core that will have SIMD extensions, and Lexra is licensing the extensions to Mips licensees at no charge. Furthermore, the new core and extensions will probably reach the market before MDMX does.

That core is the LX5280 (see MPR 5/10/99, p. 5) with the so-called Radiax extensions. Radiax consists of 36 new DSP-type instructions that seem to be the focus of the Mips complaint. Curiously, the Mips lawsuit accuses two Lexra cores of violating the '703 patent, but only the LX5280 supports Radiax. The other core Mips identifies is the LX4280 (see MPR 8/2/99, p. 13). Although the LX4280 does have a multiply-accumulate (MAC) instruction, it lacks the more complete Radiax operations found in the LX5280.

Comparing Radiax With MDMX

Radiax differs in some important ways from MDMX. It's designed primarily for real-time DSP applications, not multimedia. For that reason, it supports fractional arithmetic, modulo addressing, post-modified pointers, and other DSP-type operations.

But there are similarities between Radiax and MDMX too. Both have instructions that perform the essential

operations claimed by Mips in the '703 patent: SIMD arithmetic, intermediate extended precision, and transformation to the original precision (see MPR 8/23/99, p. 19).

For example, the LX5280's MADDA2[.S] is a MAC-type SIMD instruction that performs dual 16 × 16-bit multiplies, stores the results in a pair of 40-bit accumulators that have eight guard bits for overflow protection, and adds the products to the contents of two more 40-bit accumulators. In a separate operation, scalers transform the 40-bit results back into 16-bit values. Figure 3 shows the dual multipliers, accumulators, and scalers that carry out these operations in parallel.

Even in this example, there are differences. One potentially important difference is that a single MDMX instruction can carry out all those operations, while the LX5280 requires multiple instructions to do the same thing, and some of the operations are optional. Saturation is optional, depending on whether the programmer appends the .S suffix to MADDA2. An optional mode bit specifies whether MADDA2.S performs saturation on 40 or 32 bits. Rounding the 40-bit intermediate result down to 16 bits requires a separate instruction (RNDA2). Another instruction (MFA2) moves the dual 16-bit results from the accumulators to general-purpose registers, with an optional right shift.

If the '703 patent applies only when a single instruction performs all those operations, Lexra may be off the hook. But it's not clear to us whether the patent applies only to single instructions or to multiple instructions.

If '703 covers multiple instructions, Lexra could ask the court to invalidate the patent because of the existence of prior art. To do this, Lexra could show that somebody else disclosed similar instructions at least one year before Mips filed the '703 patent on October 9, 1997.

Almost all fixed-point DSPs use extended precision in the form of guard bits to protect intermediate results against overflows and underflows, and some also have SIMD instructions. Some other kinds of processors also perform these operations. One early example we found is Motorola's 88110, introduced in 1991 (see MPR 12/4/91, p. 1).

The 88110 has SIMD instructions for manipulating pixel values. The PUNPK instruction can unpack four 8-bit pixels into an equal number of 16-bit values by padding them with zeroes—in effect, adding 8 guard bits per pixel to protect against overflows. Various arithmetic instructions, such as PMUL (pixel multiply), can manipulate those values in parallel, returning 16-bit results. Finally, the PPACK instruction can repack the 16-bit values as 8-bit pixels by truncating the least-significant bits, effectively restoring their original precision.

Whether this example or any other is sufficient evidence of prior art is, of course, a matter for the court to decide. Technology patents are often subject to widely different interpretations.

Seeds for Future Lawsuits

An unusual feature of the Mips lawsuit is that it lists nine additional patents Mips believes Lexra may have infringed, pending "further investigation and discovery." We've never seen a patent complaint that leaves the door open to further counts based on information that may be uncovered during the discovery process.

Mips doesn't provide any further explanation, so we'll summarize the additional nine patents. U.S. 4,959,779 (issued 1990), 5,398,328 (1995), 5,408,664 (1995), 5,524,245 (1996), and 5,572,713 (1996) all describe ways of coping with different byte orderings. Lexra's cores are big-endian, not little-endian or bi-endian, so it's not clear how these patents apply to Lexra.

U.S. 4,805,098 (1989) describes a buffer that gathers sequential write requests to the same memory address and combines them into a single write request. This speeds up processing, because the CPU doesn't waste time repeatedly storing data at the same address. Lexra says its cores don't have this feature—they always execute all write requests, even if it means redundantly storing data at the same address.

U.S. 5,027,270 (1991) describes a technique for beginning to process the instructions in a cache block while continuing to fill the block with instructions from main memory. This is another feature that Lexra says its cores don't implement—the CPU must stall until a cache line is completely filled before processing the instructions in that line.

U.S. 4,953,073 (1990) describes a CPU cache in which the address-generating unit and the tag comparator are "packaged together and separately" from a primary cache. The patent appears to describe a CPU with an off-chip primary cache. But all of Lexra's cores have on-chip primary caches, so it's not clear how the patent applies in this case.

U.S. 5,590,294 (1996) describes both a method and an apparatus for restarting a pipeline after servicing one or more interlock processing faults. Lexra says its cores stall for one cycle instead of taking such a fault.

Burying the Hatchet

Building a solid patent portfolio is the best defense against costly lawsuits of this type. It's like the Cold War concept of MAD (mutually assured destruction): nobody will attempt to destroy you if you have enough weapons of your own to strike back.

Unfortunately for Lexra, Mips has most of the missiles. Although Lexra has some patent applications pending, it

For More Information

Mips Technologies' lawsuit, Lexra's counterclaim, and other documentation are available on the companies' Web sites: *www.mips.com* and *www.lexra.com*. To read the patents, go to IBM's Intellectual Property Network at *www.patents.ibm.com/patquery* and enter the patent numbers, omitting the commas.

currently has no portfolio. Mips also has more financial and legal resources than Lexra.

Lexra isn't defenseless, however. It has more licensees for its cores—17 companies that chose Lexra over Mips. And Lexra has a longer track record of licensing portable cores—Mips didn't announce its first soft cores until six months ago (see MPR 5/31/99, p. 18), a year after Lexra's LX4080 was already in silicon. Neither factor would matter in court, but both should matter to Mips. If Lexra and Mips eventually settle their differences out of court, Lexra's licensees could become Mips licensees by proxy, and it's not good business to alienate potential customers. Both companies need to focus on competing against Arm, which is the real competition for the MIPS architecture in the market for licensed cores. In the long run, a Mips-Lexra alliance would strengthen the MIPS architecture.

In addition to the considerable legal costs of pursuing the case, both companies risk other losses. A protracted legal battle could irreparably harm Lexra's growing business, even though Lexra indemnifies licensees against lawsuits of this type. If the case makes it to trial, Mips risks having a court invalidate some of its patents, even if Mips wins. Some of the patents Mips is asserting appear very broad and subject to different interpretations. It might be wiser to leave them unchallenged than to press them in court and remove all doubt.

We expect Mips and Lexra to reach a settlement that results in Lexra's holding a MIPS license. If so, Lexra will finally obtain the privilege of calling its cores "MIPS compatible." This would also save Mips's lawyers the trouble of sending letters of admonishment to errant writers who use that term. Then everyone will live happily ever after. M