VOLUME 12, NUMBER 6
MAY 11, 1998

MICROPROCESSOR REPORT

THE INSIDERS' GUIDE TO MICROPROCESSOR HARDWARE

AltiVec Vectorizes PowerPC

Forthcoming Multimedia Extensions Improve on MMX

by Linley Gwennap

Better late than never. Apple, IBM, and Motorola have defined a set of multimedia extensions to the basic PowerPC instruction set, making it the last of the six general-purpose architectures to incorporate such a feature. The first processor to use the new AltiVec extensions will be the G4, due to ship in systems in 1H99. If the G4 ships on schedule, it will be more than two years behind Intel's first MMX chips and will appear at roughly the same time as Katmai, which will include Intel's second-generation MMX extensions.

Wider is better. AltiVec (formerly known as VMX) takes a step beyond all other multimedia extensions by using 128-bit registers and ALUs, twice the width of competing designs. The wider ALUs support a high data bandwidth for applications that can take advantage of the greater degree of parallelism. Alternatively, the wider registers can provide more precision for the same number of operands.

AltiVec provides other advantages over MMX and similar extensions, such as Sun's VIS and MDMX for MIPS. Whereas these extensions operate only on integer data, AltiVec supports both integer and floating-point data types. (We expect Intel's Katmai to support parallel FP data as well.) The AltiVec registers are separate from the integer and FP registers, so there is no switching overhead. One downside: the wide registers and ALUs, which are separate from the existing function units, increase the die area needed for multimedia support in a PowerPC chip.

Apple expects to use AltiVec to improve the performance of its Macintosh systems. Even without AltiVec, the current G3 processors do well on multimedia tasks when compared against Intel's MMX processors. With AltiVec, the G4 should significantly exceed Intel's best performance on some multimedia tasks.

The new extensions will also appeal to designers of high-end embedded systems, particularly in the networking area. Manipulating data 128 bits at a time will speed these performance-hungry applications, but the added die area will exclude AltiVec from low-cost designs, at least initially.

Buffed Register File Pumps Out Data

As Figure 1 shows, the AltiVec register file is bigger than the integer register file and the FP register file combined. (In a 64-bit PowerPC chip, the integer file would be twice as wide as shown in Figure 1, bringing the comparison with AltiVec to parity. The G4 processor, however, implements the standard 32-bit PowerPC instruction set, and there are no 64-bit desktop PowerPC processors planned.)

The AltiVec register file holds eight times as much data as Intel's MMX register file (see MPR 3/5/96, p. 1), reducing the number of time-wasting cache accesses on some applications. Intel appears to be readying a larger register file for Katmai (see MPR 05/11/98, p. 4), which could close this gap.

Separating the AltiVec registers from the other registers allows them to be wider. The ALUs can thus operate on twice as much data per cycle, increasing throughput. In addition, there is no penalty for mixing AltiVec, integer, and FP instructions. In Intel's design, by contrast, the MMX registers are mapped onto the FP registers, creating a performance penalty when switching from MMX mode

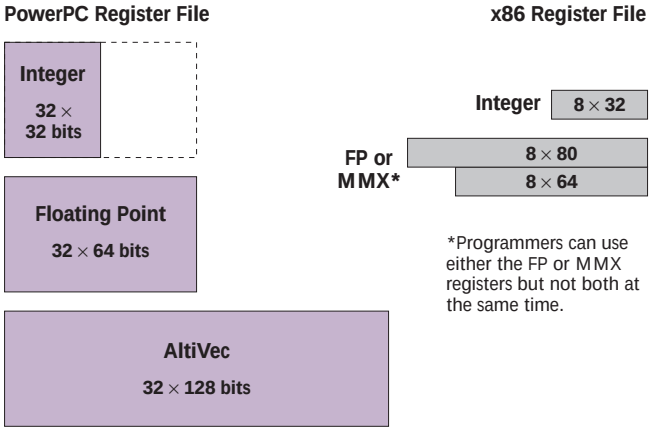

Figure 1. Adding AltiVec more than doubles the size of the existing PowerPC register file due to its 128-bit registers. The AltiVec register file can store eight times as much data as Intel's MMX register file and is not overlapped with the FP registers.

to FP mode. Sun's VIS (see MPR 12/5/94, p. 16) has a similar penalty.

The wider AltiVec registers double the time needed to save and restore CPU state on a context switch. To reduce this overhead, AltiVec includes a flag that software can manipulate to mark whether the new multimedia registers have been used; if the flag is not set, the registers need not be saved.

All PowerPC operating systems must be modified to save and restore the new registers on a context switch. As long as only one program uses the AltiVec registers, it may work properly on an unmodified OS, but this combination is not recommended.

By overlapping the MMX registers with the FP registers, Intel did not add any processor state to its processors, avoiding the need for operating-system changes. PowerPC supports far fewer operating systems than x86, however, simplifying the changeover. Motorola is already working with the pertinent OS vendors (mainly Apple and the key RTOS companies), so OS support for AltiVec should be widespread by the time the first chips arrive.

Up to 16 Operations at Once

Like other multimedia extensions, AltiVec performs parallel operations on a number of small operands in a SIMD (single instruction, multiple data) format. As Figure 2 shows, each register can hold 16 operands of 8 bits each, or 8 operands of 16 bits, or 4 operands of 32 bits. While the first two formats support only integer data types, the third format allows either integers or single-precision floating-point data. Thus, a single AltiVec function unit can perform up to 16 integer operations or 4 floating-point operations in parallel.

Except for the new data types, most AltiVec instructions are similar to the standard PowerPC arithmetic operations. They use three-operand (nondestructive) addressing and operate only on registers, not memory.

AltiVec has the usual integer arithmetic operations, such as add, subtract, and multiply. It also includes an average function ((A+B)÷2) that simply shifts the sum right by one bit. The integer operands can be signed or unsigned. Overflows can be handled by saturation (clamping to the maximum or minimum value) or by modulo arithmetic (wrapping around). AltiVec supplies the standard logical and shift operations as well.

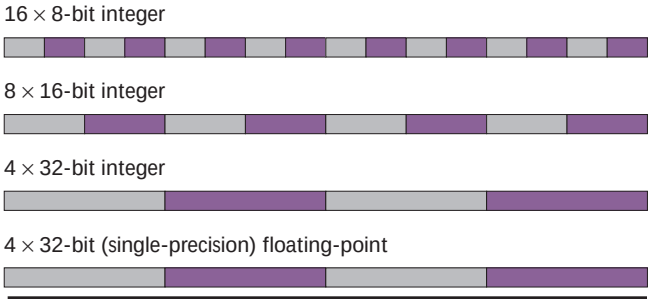

Figure 2. AltiVec supports four data formats, including one for floating-point data.

AltiVec also supports a variety of floating-point instructions, all of which operate on four single-precision values in parallel. The add, subtract, and multiply-add instructions are similar to the ones in the standard PowerPC instruction set. Unlike the standard PowerPC, AltiVec has no multiply instruction; to multiply, one must use multiply-add with an addend register that has been initialized to 0.0.

The floating-point handling in AltiVec is IEEE compliant but implements only the IEEE default exception handling and only the "nearest" rounding mode. For better performance, an even more restricted non-IEEE mode, in which denorms are essentially ignored, is provided. Some scientific code will be forced to use the standard FPU, for either double-precision arithmetic or full IEEE support. But many popular algorithms, particularly those for 3D graphics, can live within the AltiVec definitions and will gain up to four times better performance by using AltiVec.

The AltiVec units do not implement divide or square root, as these functions require too much hardware to replicate four times. Instead, AltiVec provides reciprocal estimate and reciprocal square-root estimate. These simpler instructions can be easily implemented and fully pipelined for high performance. The 12-bit "estimate" produced by these instructions can be quickly refined to higher levels of precision using the Newton-Raphson method.

For example, the quotient Q=A÷B is calculated as:

```
y0 = VREFP B              ;Estimate 1÷B
t = VNMSUBFP y0, B, 1     ;First refinement
y1 = VMADDFP y0, t, y0    ;  calculated in y1
Q0 = VMADDFP A, y1, 0     ;Q=A × y1
R = VNMSUBFP B, Q0, A     ;Calculate remainder
Q1 = VMADDFP R, y1, Q0    ;Refine quotient
```

According to Motorola, this sequence produces a quotient (Q1) that is accurate to "almost" 24 bits of precision (unless B is so small that VREFP generates an infinity, a case that can be explicitly guarded against). To guarantee a full 24 bits of precision, as required by the IEEE specification, a second refinement must be made, adding two instructions.

Note the extensive use of the negative multiply-subtract (VNMSUBFP) instruction, which calculates C–A×B. This instruction was added to help speed the Newton-Raphson

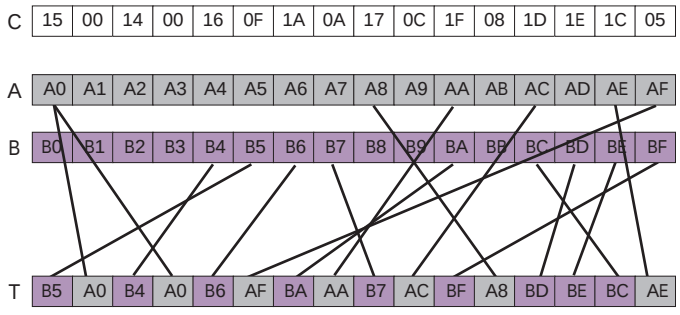

Figure 3. The permute instruction creates a new data word containing any arbitrary set of bytes selected from either of two source operands (A and B) by a control operand (C).

Byte B	Halfword H	Word W	Unsigned modulo UM	Unsigned w/sat US	Signed modulo SM	Signed w/sat SS	Mnemonic	Description	Mnemonic	Description
▲	▲	▲	■	■		■	VADD	Add	VAND	Logical AND
		▲	■				VADDC	Add, write carry outs	VOR	Logical OR
▲	▲	▲	■	■		■	VSUB	Subtract	VXOR	Logical XOR
▲	▲	▲	■	■		■	VSUBC	Subtract, write carry outs	VANDC	Logical AND complement
▲	▲		■		■		VMULO	Multiply odd	VNOT	Logical NOT
▲	▲		■		■		VMULE	Multiply even	VPERM	Permute bytes from A, B
	▲					■	VMHADD	Multiply high and add	VSEL	Select bits from A or B
	▲					■	VMHRADD	Multiply high, round, add	VSL/VSR	Shift register left/right by bits
	▲		■				VMLADD	Multiply low and add	VSLO/VSRO	Shift register left/right by bytes
▲	▲		■		■		VMSUM	Multiply and sum	VSLDOI	Shift left immediate and OR
	▲			■					VADDFP	Add floating point
		▲				■	VSUM	Sum across to one sum	VSUBFP	Subtract floating point
		▲				■	VSUM2	Sum across to two sums	VMAXFP	Select maximum FP
▲				■			VSUM4	Sum across to four sums	VMINFP	Select minimum FP
▲	▲					■			VMADDFP	Fused multiply-add
▲	▲	▲	■		■		VAVG	Average	VNMSUBFP	Fused negative multiply-subtract
▲	▲	▲	■		■		VMIN	Select minimum	VREFP	Reciprocal estimate
▲	▲	▲	■		■		VMAX	Select maximum	VRSQRTEFP	Reciprocal square-root estimate
▲	▲	▲	■		■		VCMPGT[.]	Compare greater than [record]	VLOGEFP	Base 2 logarithm estimate
▲	▲	▲	■				VCMPEQ[.]	Compare equal to [record]	VEXPTEFP	2 to the exponent estimate
▲	▲	▲	------n/a------				VRL	Rotate elements left	VCMPGTFP[.]	Compare greater than [record]
▲	▲	▲	------n/a------				VSL	Shift elements left	VCMPEQFP[.]	Compare equal to [record]
▲	▲	▲	------n/a------				VSR	Shift elements right	VCMPGEQFP[.]	Compare >= [record]
▲	▲	▲	------n/a------				VSRA	Shift right arithmetic	VCMPBFP[.]	Bounds check [record]
	▲	▲	■	■			VPKU	Pack unsigned integer	VRFIN	Round to nearest
	▲	▲	■	■			VPKS	Pack signed integer	VRFIZ	Round toward zero (truncate)
		▲	■				VPKPX	Pack into 1/5/5/5 pixels	VRFIP	Round toward positive infinity
▲	▲		■				VUPK[H/L]	Unpack integer high/low	VRFIM	Round toward minus infinity
	▲		■				VUPKPX[H/L]	Unpack 1/5/5/5 pixels	VCTUXS	Convert to unsigned integer w/sat
▲	▲	▲	------n/a------				VMRG[H/L]	Merge high/low	VCTSXS	Convert to signed integer w/sat
▲	▲	▲	------n/a------				VSPLT	Splat (replicate data)	VCFUX	Convert from unsigned integer
▲	▲	▲	------n/a------				VSPLTIS	Splat signed immediate	VCFSX	Convert from signed integer
---n/a---			------n/a------				LVEX, LVEXL	Load 128 bits into register	DST	Data stream touch
---n/a---			------n/a------				STVEX, STVEXL	Store 128 bits to memory	DSTST	Data stream touch for store
▲	▲	▲	------n/a------				LVExX	Load element of width x	DSS	Data stream stop
▲	▲	▲	------n/a------				STVExX	Store element of width x	MTVSCR	Move to vector control register
---n/a---			------n/a------				LVSL, LVSR	Calculate alignment control value	MFVSCR	Move from vector control register

Table 1. AltiVec adds 162 new instructions that operate on the 128-bit vector registers in a SIMD fashion. Instructions on left side of the table take up to three options for operand width (▲) and up to four options for overflow handling (■); instructions on right side use a single operand format and do not overflow. Floating-point instructions shown in purple. x=byte, halfword, or word n/a=not applicable

process. This process can also be used to refine the estimates from the floating-point reciprocal square root, log, and exponent instructions.

To simplify working in AltiVec floating-point math, the extensions include instructions to convert from packed FP data to packed fixed-point data, and vice versa. These include the unfortunately named VCFUX and VCTUXS instructions.

Highly Flexible Bit Manipulation

A highlight of AltiVec is its completely arbitrary permute instruction. As Figure 3 shows, a data mask in one operand (C) controls the creation of a new 128-bit value in which each byte can be taken from any arbitrary byte in either of two source operands (A and B). Each byte in C controls the corresponding byte in the target register. The upper nibble selects the source register, either A or B, and the lower nibble selects any one of the 16 bytes in that register, which is then copied to the target register.

While we can admire this flexibility, the permute unit is designed to perform specific tasks. For example, it can pack and unpack data as well as merge data from two registers, as Table 1 shows. It can fill a register from a single byte ("splat"), easily creating constants or clearing a register. It can perform long data shifts, repair unaligned data, and perform table lookups. Special pack and unpack instructions support 1/5/5/5 pixel format, which permits 1 bit of α and 5 bits each of R, G, and B.

The AltiVec compare instructions perform parallel data comparisons such as "equal to" or "greater than," storing the results in the target register as a series of boolean values.

	PowerPC AltiVec	Intel MMX	Sun VIS	MIPS V/ MDMX	HP MAX2	Alpha MVI
Register File	32 × 128	8 × 64	32 × 64	32 × 64	32 × 64	32 × 64
Mapped Onto	Separate	FP	FP	FP	Integer	Integer
Integer Support	8/16/32	8/16/32	8/16/32	8/16 bit	16/32 bit	8 bit
FP Support	Yes	MMX2	No	MIPS V	No	No
The Usual Stuff†	Lots	Lots	Lots	Lots	Some	None
Multiply / MAC	Lots	Mult	Mult	Lots	Some	None
Min / Max /Avg	Yes	No	No	Min/Max	Avg	Min/Max
Pack / Unpack	Yes	Yes	Yes	Yes	Yes	Yes
Byte Reordering	All	Some	Some	Many	All	None
Unaligned Data	3 instr	No	2 instr	Yes	No	No
Announced	2Q98	2Q96	4Q94	4Q96	4Q95	4Q96
First Shipped	Mid-99	1Q97	4Q95	None	2Q96	4Q97

Table 2. Although PowerPC is the last of the major desktop architectures to introduce multimedia extensions, AltiVec offers the most complete feature set. †saturating and nonsaturating adds, subtracts, compares. (Source: vendors)

If the comparison is true, the target word is set to all ones; if the comparison is false, it is set to all zeroes. The resulting string of boolean values can be used as a bit mask by the logical operations. It can also be used by the select (VSEL) instruction, which transfers bits from one source register or the other, depending on the contents of the bit mask.

This technique can be used for video masking (e.g., blue screening) and 3D clipping functions. Compilers can also use it to eliminate some branch instructions by computing both paths of the branch in parallel. The correct results can then be selected using the conditional instructions.

In addition to the basic integer and FP comparisons, Altivec includes a special bounds-check (VCMPBFP) instruction. This instruction actually performs two comparisons on each value, determining if $-B \le A \le B$. In other words, this instruction checks if the absolute value of A is within the limit specified by B (i.e., $|A| \le B$).

Some algorithms need to examine the result of a series of comparisons. For example, software may want to check if any word has exceeded a saturation value. Using the "record" option (indicated by [.] in PowerPC notation), the compare instructions can be set to modify the PowerPC condition-code register (CR) if all parallel comparisons are true (all ones) or if all comparisons are false (all zeroes). The former case sets CR bit 24; the latter sets CR bit 26. These bits can be checked by subsequent conditional-branch instructions.

Wide Loads and Stores Boost Bandwidth

AltiVec has obligatory load and store instructions to move data into and out of the new registers. Because these loads and stores handle 128 bits of data at a time, they may be used for block memory accesses, as they are four times as efficient as the usual integer loads and stores. The address is calculated from the integer registers using the normal register-indirect-with-index addressing mode.

Unlike the standard PowerPC load and store instructions, the AltiVec instructions do not support unaligned addresses. If software must manipulate data structures that are not aligned, they can be loaded into the vector registers "as is" and aligned using the permute (VPERM) instruction. To assist in such alignment, the LVSL and LVSR instructions do not load data but instead take an unaligned address and compute the necessary control word that allows the VPERM instruction to properly align the data. Thus, unaligned data can be loaded using a three-instruction sequence (LVEX, LVSL, VPERM).

AltiVec also includes a set of "data stream touch" instructions that allow software to attempt to manage the cache/memory hierarchy. These instructions assume the existence of a software- controlled prefetch engine, which apparently will be included in the G4 processor. The DST instructions pass an address to the prefetch engine, which presumably begins fetching a data stream (which may have an arbitrary stride) starting at that address. Variations of the instruction inform the prefetcher whether the data will be used once or frequently, and whether it might be written. The DSS instruction terminates prefetching of one or all data streams.

Twice As Wide As Competition

AltiVec offers a more complete feature set than any of the other multimedia extensions, as Table 2 shows. The biggest difference is the width of the registers. With 128-bit operands, AltiVec instructions will, each cycle, operate on twice as much data as competing implementations. In theory, this should result in twice the peak performance, although clock speed and other architectural considerations come into play.

The other key advantage of AltiVec is its floating-point support. MIPS V (see MPR 11/18/96, p. 24) is the only other announced architecture with this capability, but Silicon Graphics' recent termination of its future MIPS projects (see MPR 4/20/98, p. 1) leaves no planned processors to implement MIPS V. Intel's MMX2, also known as the Katmai New Instructions (KNI), is expected to include similar parallel floating-point instructions, although the company has not revealed whether they will handle two or four operands at once.

AltiVec is designed as a general-purpose architecture instead of being optimized for a single application. For example, it does not include a SAD (sum of absolute differences) instruction, which is found in both VIS and Alpha's MVI (see MPR 11/18/96, p. 24). This instruction is the core of most motion-estimation algorithms, such as MPEG-2 video encoding, and greatly speeds these applications.

AltiVec offers partial compensation with its "sum across" instructions, but a complete SAD operation requires four instructions. Motorola points out that the more accurate SDS (sum of differences squared) method, after some mathematical transformation, can be performed using an inner loop consisting only of a single VMSUM instruction.

Thus, the SDS loop can compute 16 results per cycle, the same speed as if a hard-wired SAD instruction were available. This eliminates the need for special-purpose SAD hardware that would be used only in a single application. The company claims the G4 processor will be able to handle MPEG-2 encoding using the SDS method.

AltiVec's performance on other applications will vary. Considering only the SIMD instructions, peak performance on inner loops will be four times greater than with standard PowerPC instructions. Compared with MMX or VIS, peak performance will be doubled for integer operands (assuming similar clock speeds and implementations) and quadrupled for single-precision floating-point operands.

Peak performance could be even better for code that takes advantage of permute and other special instructions. Because overhead (nonarithmetic) instructions will not be sped up by AltiVec, however, actual throughput will lag peak performance, often by a large amount. More performance data will be published once G4 prototypes are available.

AltiVec Eats Silicon

The biggest drawback of AltiVec is its size. The 128-bit registers and operations require a separate register file and separate ALUs (both integer and floating-point), both twice as large as the existing units. This overhead requires a physically large implementation. In contrast, most other multimedia designs share an existing register file and existing ALUs, adding only decode logic and a bit of special-purpose logic.

In Sun's UltraSparc design, for example, the VIS logic adds only 3% to the total die area, about 4 mm^2 in a 0.29-micron process. We estimate the P55C, which has physically separate registers and ALUs for MMX, devotes to its multimedia functions about 15 mm^2 in a similar process. In contrast, the AltiVec logic is likely to be twice that size if implemented in a similar process, according to Motorola.

Because the G4 will be built in a more advanced 0.25-micron process, however, the actual area impact will be only about 17 mm^2. While this will represent a sizable fraction of the G4's die area, Motorola says the G4 will still weigh in well below 100 mm^2, hardly a monster-sized chip. If necessary, Motorola could trim the die size further by getting rid of AltiVec's FP support, which is unneeded in many embedded applications.

Taking Aim At Embedded

Motorola hopes to drive the G4 and other AltiVec processors into embedded applications. The target applications are distinguished by a willingness to pay a premium for processors that can process data by the bucketful. These applications include network routers, voice over IP (VoIP), encryption and decryption, multichannel modems, speech processing, and video processing. Potential customers include Cisco and other networking companies, as well as telecom vendors.

For More Information

The VMX extensions are not available in a shipping PowerPC processor. For more information about the extensions, contact your local Motorola representative or check the Web at *motorola.com/AltiVec*.

Motorola's competition in these areas comes from a variety of sources. Cisco, for example, uses standard MIPS processors in many of its systems. MIPS has developed a set of integer multimedia instructions called MDMX (see MPR 11/18/96, p. 24) that are similar to AltiVec but provide half the processing power. Like AltiVec, MDMX has yet to ship.

Many of the telecom vendors use DSP chips instead of general-purpose CPUs. Therefore, Motorola has compared the G4's AltiVec-enhanced performance against that of Texas Instruments' C62xx (see MPR 2/17/97, p. 14), a VLIW-based high-performance DSP. The Philips TM-1 media processor may even compete with the G4 in some situations. Motorola believes its chip will offer superior price/performance, but it is impossible to evaluate this claim until the company reveals the price and performance of the G4.

While the AltiVec extensions might be useful in other embedded applications, the initial cost of the chip is likely to be too high. For example, in a set-top box the G4 could perform audio and video decoding and even support video conferencing or a cable-modem interface. Perhaps a future 0.18-micron version could be cost-effective enough for this type of consumer device.

A New Processor Paradigm

The new extensions will have a greater impact on the desktop, at least initially. AltiVec will help Apple's Mac systems compete against Katmai-powered PCs on 3D and multimedia software. We expect IBM will use AltiVec in its workstations to improve performance on many graphics and scientific applications, although the new instructions will not help applications that require double-precision math or obscure IEEE modes.

The initial set of multimedia extensions (MAX, VIS, MMX) took a step forward by establishing a new data type for audio and video. These first attempts, however, devoted the minimum possible amount of silicon to the new features, treating multimedia as a second-class citizen.

With AltiVec, the PowerPC vendors have taken a different approach, moving multimedia to the front of the bus. By devoting a significant portion of the die to AltiVec, the G4 processor emphasizes the growing role of multimedia. Most of the performance-hungry applications today can be vectorized to take advantage of AltiVec. If multimedia performance becomes the driving factor in the PC market, AltiVec's 128-bit architecture could tip the performance scales in PowerPC's favor. M