Intel® Wireless MMX™ Technology: A 64-Bit SIMD Architecture for Mobile Multimedia

N.C. Paver, B.C. Aldrich, M.H. Khan
Intel Corporation, 1501 S. Mopac, Suite 400, Austin, Texas, USA, 78746

nigel.paver@intel.com

Abstract - The growing demand for multi-media rich applications in the wireless mobile domain challenges the capabilities of current wireless handheld devices. Optimizing instruction set architecture is a logical approach towards attaining higher performance in multimedia applications. Intel® Wireless MMX™ technology is a 64-bit Single Instruction Multiple Data, (SIMD), coprocessor for the Intel® XScale™ microarchitecture. It accelerates multi-media applications in handheld and wireless devices by taking advantage of the inherent parallelism and data types of targeted applications. This paper provides an overview of the Wireless MMX architecture, its instruction set, pipeline organization, and functional units. Initial benchmark results measured on silicon are also presented.

I. INTRODUCTION

As the mobile-device market segment matures (cell phones, smartphones, PDAs, etc.), the end-users are demanding richer multi-media experiences, including: video and audio playback, 3D graphics, digital photography and interactive gaming. This trend is challenging the computational capabilities of current wireless platforms.

Many multimedia algorithms typically execute the same set of operations on a large number of data elements. For example, the same image processing operation is usually applied to all pixels in the image. This creates an opportunity for parallel processing, referred to as data parallelism. In addition, the dynamic range of the multi-media content can be represented with lower precision (fewer bits) than the processing-width of a typical processor, thus, giving rise to sub-word level parallelism.

Wireless MMX technology is a 64-bit SIMD coprocessor targeted towards hand-held wireless devices. This exploits the sub-word (multiple of 8 bit and 16 bit) and word level (multiple 32 bit) data parallelism present in a large number of multimedia algorithms, by executing the same operation on different data elements in parallel. This is accomplished by packing data elements into a single register and introducing new types of instruction to operate on packed data. This paper, presents an architectural overview of Wireless MMX and initial benchmark results.

II. TECHNOLOGY OVERVIEW

Significant research effort and desktop processor development has been under-taken related to SIMD processing for media applications [1][2][3][4]. Wireless MMX technology integrates equivalent functionality to all of Intel® MMX™ technology [5] and the integer functions from SSE [2] to the Intel® XScale™ microarchitecture [6]. Like MMX technology and SSE, Wireless MMX technology utilizes 64-bit wide SIMD instructions, which allow it to concurrently process up to eight data elements in a single cycle. This style of programming is well known to existing software developers.

Wireless MMX technology defines three packed data types (8-bit byte, 16-bit half word and 32-bit word) and the 64-bit double word. The elements in these packed data types may be represented as signed or unsigned fixed point integers. Using special SIMD instructions it is possible to operate on data elements in the packed format, where each data element is treated as an independent item.

III. ARCHITECTURE

The Wireless MMX unit is a tightly coupled coprocessor of the XScale microarchitecture. The programmer's model is an extension of the XScale microarchitecture programming model. Thus, a multimedia application can maintain a single thread of control while taking advantage of the SIMD acceleration only on the critical section of the algorithm. As Wireless MMX is an extension of the XScale microarchitecture it takes advantage of the existing memory subsystem for the XScale microarchitecture without the need for extra dedicated memories and the power comsumption associated with them. The power efficiency is further improved by using advanced power management, where the Wireless MMX unit is only activated, when required, on an instruction by instruction basis.

Wireless MMX technology comprises five key functional units to implement the programmers model. Figure 1 shows the organization of the functional units within the coprocessor.

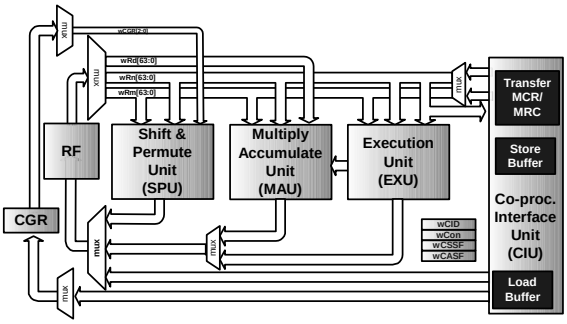

Fig. 1. Micro-Architecture

The Shift and Permute Unit is responsible for performing shift and permute operations. These operations include the alignment, shift, rotation, packing and shuffling.

The Execute Unit is responsible for performing arithmetic and logic operations and it also provides a saturation capability. Operands may be received from the SPU when saturation is required and the CIU when the transfer instructions are issued.

The Multiply and Accumulator Unit is responsible for performing all multiply and accumulate operations. Operands are also received from the EXU when executing the sum of absolute difference instruction. The MAU unit is a three stage pipeline with internal accumulator forwarding.

The Coprocessor Interface Unit transfers data between the Wireless MMX^{TM} unit registers and the Intel$^{®}$ XScale^{TM} microarchitecture. In addition to supporting coprocessor data transfer, it is also responsible for storing and loading data to and from the memory.

The main Register File (RF) is organized as sixteen 64-bit registers, located in the coprocessor 0 space (CP0). The large register file allows the Wireless MMX instructions to support an increased number of intermediate values in complex calculations. For example, multiple output samples of a filter may be calculated in parallel or a single pass, half pixel motion search can be implemented (all eight intermediate block comparison results are calculated concurrently). The increased storage allows the programmer to take advantage of the spatial and temporal data locality as found in many multi-media applications. This reduces the required load store bandwidth and improves processing efficiency. Alignment support instructions also increase the effectiveness of this data re-use. These combined techniques are referred to as *Multi-Sample Technology.*

The control and status registers are mapped into coprocessor 1 space (CP1). There are four 32-bit general-purpose control registers used for alignment and shift control. As the shift and alignmet offset is usually invarient across the inner loops the registers are designed to hold constant values. This saves the use of a packed data registers.

There are also a number of control and status registers also mapped onto coprocessor 1 space:

wCASF—SIMD Arithmetic Status Flags- A set of four flags for each byte/half-word/word/double-word operation

N when the result is negative C when there is a carry out
Z if the result is zero V if the result over-flowed

wCSSF—SIMD Saturation Flags which set the respective flag if an operation on a particular element saturates

wCON coprocessor control register—support for reducing memory traffic on a context switch.

A. Pipeline Structure

To achieve the same clock speed as the Intel XScale core, the Wireless MMX unit employs the same super-pipelined structure, as shown below in figure 2. The coprocessor is a five-stage pipeline with two extra stages of instruction fetch provided by the main core. Wireless MMX pipeline operates in lock step with the main core pipeline, providing a single thread of control and no complex synchronizations required between the two pipelines.

The XScale microarchitecture with Wireless MMX technology is a single issue machine. An instruction can be issued to the main core pipeline or coprocessor pipeline. The architecture allows instructions to be retired out of order. The pipeline organization also supports multiple out-standing loads, which improves memory throughput. This feature combined with the out-of-order completion, allows non-dependent instructions to execute, reducing the impact of memory latency in system on a chip applications.

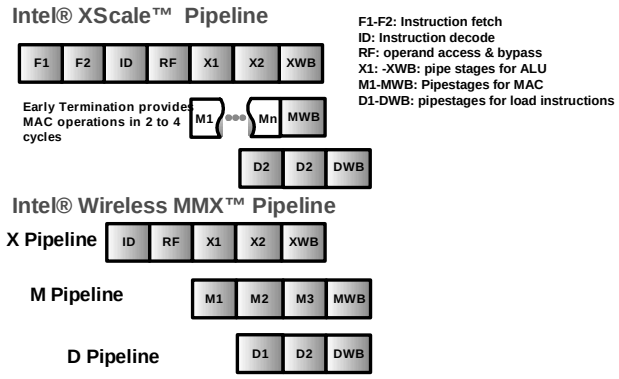

Fig. 2. Pipeline Structure

B. Instruction set Overview

Wireless MMX technology provides a rich set of instructions that perform parallel operations on multiple data elements packed into 64-bit words. In addition, backwards compatibility is provided with existing XScale microarchitecture coprocessor 0 instructions, which operate on XScale core registers (TMIA, TMIAxy, TMIAPH).

In total there are 43 new instructions in the architecture. Table 1 provides a overview of the instruction set.

Table 1: Key Instruction Overview

Instruction	Description
WACC	Addition of all 8xbytes, 4xhalf words, or 2xwords in 1 reg
WADD/WSUB	Add/Subtract 8xbytes, 4xhalf words, or 2xwords
WALIGN	Extracts 64-bit value on byte boundaries from 2x64 bit.
WAND/WOR/ WXOR	64-bit logical operations
WAVG2	Unsigned average on vectors of 8- or 16-bit data
WCMPEQ WCMPGT	Compare 8x bytes, 4xhalf words, or 2x word elements in parallel. Result is mask of 1's if true or 0's if false
WMAC	Multiply four signed or unsigned 16-bit half words in parallel and accumulate with a 64-bit register.
WMAX/WMIN	Vector maximum/minimum selection
WMADD	Multiply four 16-bit words in parallel and add
WMUL	Multiply four signed 16-bit words in parallel. Low- or high-order 16 bits of 32-bit result are produced
WROR/WSRA/ WSLL/WSRL	Rotate right, shift arithmetic/logical right, left shift of 4 half words, 2 words, or 64-bit double word, in parallel
WPACK	Pack double word to words or words to bytes
WSAD	Sum of absolute differences on 8xByte or 4x16-bit data.
WSHUFH	Shuffles 16-bit data specified by an 8-bit immediate
WUNPCK	Unpack 8xbytes, 4x16-bit half words, or 2x32-bit words
WLDR/WSTR	Load/Store Byte, half word, word or double word
TANDC TORC	Logical operations across the fields of the SIMD PSR (wCASF) and sends the result to the XScale core CPSR
TBCST	Broadcasts a value from the XScale core source register to every element in the packed destination register
TMIA	32x32 signed multiply-accumulate using operands from the two source XScale core registers
TMIAxy	16x16-bit multiply-accumulate selecting high/low 16-bits from two source XScale core registers
TMIAPH	Dual 16x16 multiply accumulate into 64-bit using signed 16-bit operands from the two source XScale core registers

The WSADB instruction shown in figure 3 is an example of a SIMD instruction. This sums the absolute difference of the eight corresponding bytes in two 64-bit words in a single operation. It is used to accelerate motion search in video encode [7].

C. Mapping to ARM Architecture

The Wireless MMX™ unit maps onto two ARM* [8,9] coprocessors: Coprocessor 0 and coprocessor 1. The two coprocessors are used to encode the instruction set shown above, with the data registers being mapped into coprocessor 0 and the control register being mapped into coprocessor 1.

In common with the Intel® XScale™ microarchitecture

*. Other names and brands may be claimed as the property of others

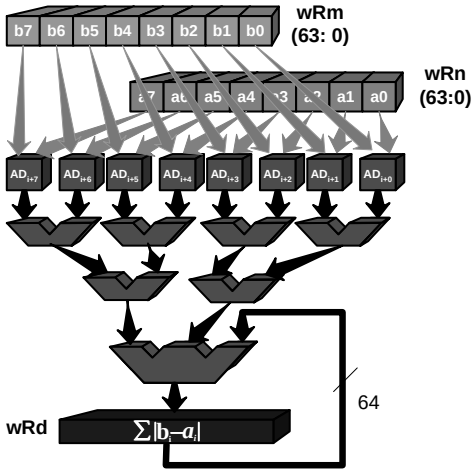

Fig. 3. WSADB Instruction

programing model, Wireless MMX technology supports the conditional execution primitives of the ARM architecture. However, to better integrate the SIMD programing model into the XScale microarchitecture, a unique feature is provided to logically combine the SIMD arithmetic flags and use these in conjunction with the ARM conditional execution mechanism, to provide new execution predicates. This technique is known as *Group Conditional Execution* and is useful in search algorthims.

The SIMD arithmetic flags can be transferred to the ARM status register by performing an AND/OR operation across the SIMD PSR word, and sending the flags to the ARM CPSR. Here, subsequent conditional instructions can use them (both XScale microarchitecture and coprocessor conditional instructions). Use of TANDC and TORC instructions support conditional instructions that operate on conditions such as "*if any* SIMD field is zero" (using TORC) and "*if all* SIMD fields are zero" (using TANDC).

IV. PERFORMANCE BENCHMARKS

A. Video Performance

Support for video decoding and encoding is one of the key wireless media applications. Wireless MMX optimization can be applied to the computational intensive inner loops such as motion compensation, DCT and color conversion, since they yield maximum sub-word parallelism[3]. Figure 4 shows how these inner loops from a MPEG video decoder can be accelerated with Wireless MMX technology. In the figure, MC_R/NR, INT_MC_R/NR represent non-integer and integer motion compensation with/without rounding respectively. 8x8_Inv_DCT is an inverse DCT for an 8x8 block with 16 bit / sample precision and CC_YUV_RGB is color conversion between YUV420 to BGR565 format [10].

Figure 5 shows the frame rate of a laboratory test of a video decoder running on the XScale microarchitecture with

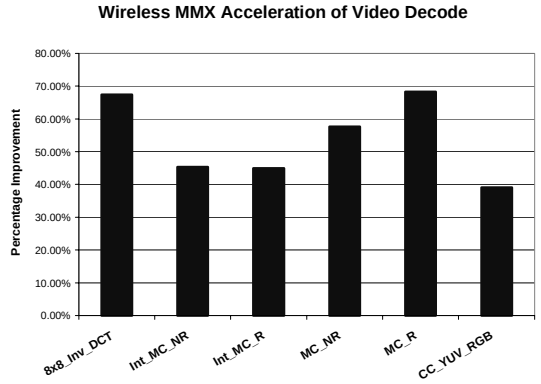

Fig. 4. Acceleration on key MPEG-4 Kernels

Wireless MMX^{TM} unit enabled and disabled. The video decoder is an MPEG4 decoder, simple visual profile [10,11] running on a bare metal system (without OS) decoding the standard coast guard sequence[12] at CIF (352x288) resolution. At the same frequency, Wireless MMX technology gives a 61% improvement in decode frames per second.

This is achieved with only a 7% increase in the core power consumed, compared to the Intel® XScale^{TM} microarchitecture without Wireless MMX unit enabled. Alternatively, Wireless MMX technology can achieve the same frame rate at 44% less clock frequency. This gives a subsequent power reduction of 27% at the same voltage. Dynamic voltage management can be utilized to further reduce the power consumption by reducing the voltage required to meet the lower frequency target.

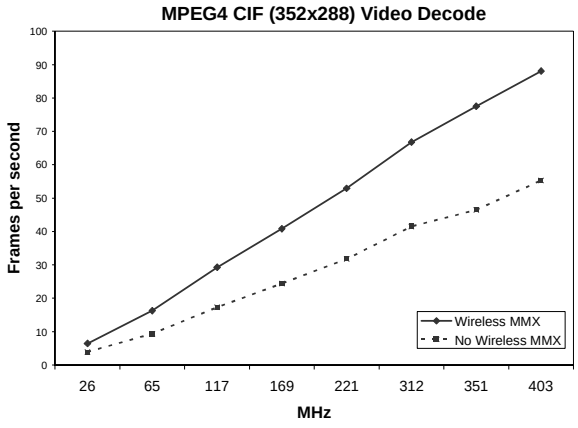

Fig. 5. Video Decode Performance

B. Typical DSP Kernels

Apart from video processing acceleration, Wireless MMX technology offers improvement in many other Digital Signal Processing applications. Table 2 shows the performance across a collection of critical DSP inner loop algorithms. Use of the large register file as a level 0 cache and applying multi-sample techniques enables competitive DSP performance.

Table 2: Performance on DSP Algorithms

Algorithm	Cycle Count
N sample T tap real FIR Filter	5NT/8
N sample T tap complex FIR Filter	5NT/2
N sample T tap real LMS Adaptive Filter	3NT/2
N sample Maximum or Minimum	N/2
N dimensional Vector Dot Product	3N/4
N dimensional Vector Sum	N

V. SUMMARY

In this paper we have described the architecture and instruction set of Wireless MMX technology. We have shown how this can be applied to a video decoder application and we have disclosed some preliminary measurements and benchmarks which display the performance delivered by this technology in the execution of critical inner loops and multimedia applications.

VI. ACKNOWLEDGEMENTS

We acknowledge the significant contribution of the entire Wireless MMX technology development team in Austin, TX and Chandler, AZ and the PCG systems engineering in Hudson, MA.

VII. REFERENCES

[1] Alex Peleg and Uri Weiser, “MMX Technology Extension to Intel Architecture”, IEEE Micro, 16(4):42-50, Aug. 1996.

[2] Keith Diendroff, “Pentium III = Pentium II+ SSE”, Micro Processors Report, 13(3):6-11, March, 1999.

[3] Lee, Ruby, “Subword Parallelism in MAX-2”, IEEE Micro, 16(4), 51-59, Aug. 1996

[4] Tremblay, Marc, et. al. “VIS Speeds Media Processing”, IEEE Micro, 16(4): 10-20, Aug. 1996.

[5] Weiser, Uri, et al, “The Complete Guide to MMX^{TM} Technology”, Mcgraw-Hill, 1997, ISBN 0-07-006192-0.

[6] http://www.intel.com/design/intelxscale/

[7] Kuhn, Peter, “Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation”, Kluwer Academic Press, ISBN 0-7923-8516-0

[8] Seal, David “Advanced RISC Machines Architecture Reference Manual”, Prentice Hall, 1996. ISBN 0-201-73719-1

[9] Furber, S.B., “ARM System-on-Chip Architecture”, Addison Wesley, 2000. ISBN 0-201-67519-6.

[10] “Integrated Performance Primitive”, http://intel.com/software/products/ipp/

[11] “MPEG4 Overview (V.21)”, Edited by Rob Konen, ISO/IEC JTC1/SC29/WG11 N4668.

[12] International Organization for Standardization. "ISO/IEC JTC1/SC29/WG11N1902 14496-2 Committee Draft (MPEG-4)." November 1997.