Conjunto de Instruções Multimídia

Celso Suzuki (RA 004859)
Instituto de Computação
UNICAMP
celso.suzuki@ic.unicamp.br

ABSTRACT

Este texto descreve as extensões multimídia incorporadas ao conjunto de instruções de processadores de uso geral para acelerar a execução de aplicações multimídia. São descritas as operações SIMD paralelas mais comuns entre as extensões, como operações aritméticas, deslocamentos de bits, comparação de subpalavras e empacotamento de subpalavras. As extensões multimídia para a arquitetura x86 são descritas em detalhes, e um breve experimento é realizado para demonstrar o aumento de desempenho com o uso de instruções multimídia.

1. INTRODUÇÃO

Extensões multimídia são novas instruções e recursos adicionados aos processadores de uso geral para melhorar o desempenho de programas que processam mídia, como imagem, vídeo, gráficos 2D e 3D, animações e áudio.

A arquitetura PA-RISC 1.1 (HP) foi a primeira a introduzir extensões multimídia, denominada MAX-1 (Multimedia Acceleration eXtensions) [15], em Janeiro de 1994. Para demonstrar a eficácia do MAX-1, juntamente foi apresentado um programa decodificador de vídeo MPEG-1 que podia decodificar um vídeo de alta-fidelidade, com 30 frames por segundo, em uma estação PA-RISC de 80 MHz. Até então, a decodificação de MPEG em tempo real só podia ser realizada com o uso de hardware dedicado. Em seguida, a Sun adicionou a extensão VIS (Visual Instruction Set) [19] à arquitetura de processadores Sparc. A HP então introduziu o MAX-2 [14], a segunda geração de extensões multimídia para a arquitetura de processadores 64 bits PA-RISC 2.0 [13]. Em Outubro de 1996, a SGI (Silicon Graphics) anunciou o MDMX (Mips Digital Media eXtensions) [7] para a arquitetura MIPS V, e a Intel introduziu processadores x86 com instruções MMX [11]. Em Março de 1997, a DEC (agora Compaq) anunciou um pequeno conjunto de instruções MVI (Motion Video Instructions) [18] para processadores Alpha, específicas para acelerar a codificação de vídeo MPEG-2. Em 1999, a Motorola anunciou a extensão AltiVec [6] para a arquitetura de processadores PowerPC. Em 2001, a ARM anunciou extensões multimídia incluídas na arquitetura ARMv6 [4]. Todas essas extensões são implementações de instruções SIMD (Single Instruction, Multiple Data), que aplicam operações em múltiplos dados em paralelo, com uma única instrução. As extensões diferem no conjunto de instruções, nas operações disponíveis, no tamanho e tipo dos dados e no número de novos registradores.

A característica comum e fundamental dessas extensões é o paralelismo de subpalavras, e será descrito na seção 2. A seção 3 trata das operações multimídia mais comuns, a seção 4 descreve as extensões multimídia para a arquitetura x86, a mais usada em computadores *desktop*, e a seção 5 traz uma avaliação do aumento de desempenho com o uso de instruções multimídia em uma aplicação de processamento de imagem.

2. PARALELISMO DE SUBPALAVRAS

Aplicações multimídia orientadas a pixel, como processamento de imagens e processamento de vídeo, apresentam um alto grau de paralelismo de dados de baixa precisão (menor que 16 bits). Os pixels de entrada e de saída normalmente possuem componentes de 8 bits ou 12 bits (aplicações médicas), mas o processamento intermediário geralmente requer uma precisão um pouco maior, como 16 bits. Nas aplicações de áudio, o tamanho típico de uma amostra (*sample*) é 16 bits [11]. Com base nestas observações, o conceito de paralelismo de subpalavras foi introduzido na arquitetura de processadores. Neste conceito, uma palavra é particionada em unidades menores denominadas subpalavras. Uma mesma operação pode ser aplicada nas subpalavras em paralelo, provendo uma forma de processamento SIMD. As instruções podem trabalhar em palavras inteiras, ou em subpalavras em paralelo. As subpalavras podem ter tamanhos diferentes, geralmente 8, 16 ou 32 bits, dependendo da arquitetura do processador. A figura 1 ilustra uma operação sendo aplicada em quatro subpalavras em paralelo.

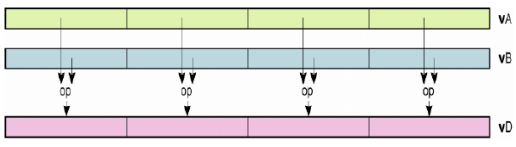

Figura 1: Exemplo de paralelismo de subpalavras. Uma operação é aplicada em quatro subpalavras em paralelo.

Uma unidade funcional baseada em palavras pode ser particionada em unidades de subpalavras em paralelo, com pouca adição de hardware. Por exemplo, um somador inteiro de complemento de dois, de 64 bits, pode ser particionado em quatro somadores inteiros com subpalavras de 16 bits, simplesmente bloqueando o bit *carry* nos três limites de 16 bits. Este somador particionável permite quatro somas de 16 bits,

ou uma soma de 64 bits, executadas em um único ciclo de *clock*. Nas extensões MAX-1, MAX-2, MVI, VIS e MMX, as subpalavras são números inteiros, mas o conceito de paralelismo de subpalavras pode ser aplicado em números de ponto flutuante, como nas extensões 3DNow! da AMD, SSE, SSE2 e SSE3 da Intel e AltiVec.

3. INSTRUÇÕES MULTIMÍDIA

Esta seção descreve as instruções mais comuns disponibilizadas nas extensões multimídia.

3.1 Operações aritméticas

A soma e subtração paralela são as operações presentes em praticamente todas as extensões multimídia, e geralmente trabalham com subpalavras de 8, 16 ou 32 bits. As subpalavras podem ser com ou sem sinal. A multiplicação paralela possui grande diferença entre as extensões, principalmente porque o número de bits do produto resultante pode ser maior que o número de bits dos operandos, ultrapassando o tamanho de uma subpalavra. A extensão MVI não suporta multiplicação de subpalavras. A extensão VIS permite apenas a multiplicação entre um operando de 16 bits e um operando de 8 bits, resultando nos 16 bits mais significativos do produto de 24 bits. A extensão MMX permite a multiplicação paralela entre operandos de 16 bits, resultando na seleção dos 16 bits mais ou menos significativos, do produto de 32 bits.

3.2 Aritmética de Módulo e de Saturação

A diferença entre a aritmética de módulo e de saturação é como o *overflow* é tratado. Na aritmética de módulo, o *overflow* é ignorado, e o resultado é dado pelo módulo 2^n, para subpalavras de n bits. Por exemplo, para uma palavra de 8 bits com valor 255, $(255 + 1)$ mod 256 é 0. Na aritmética de saturação, um *overflow* positivo, isto é, um resultado além do maior número representável em n bits, faz com que o resultado seja convertido para o maior número representável. Um *overflow* negativo ou *underflow*, isto é, um resultado abaixo do menor número representável, faz com que o resultado seja convertido para o menor número representável. Um exemplo de uso da aritmética de saturação é a mudança de brilho de uma imagem. Para pixels de 8 bits sem sinal, o aumento de brilho em um pixel de valor 255 deve manter o valor do pixel em 255, e a diminuição de brilho em um pixel de valor 0 deve manter o valor do pixel em 0. Se fosse usada a aritmética de módulo, o aumento de brilho em um pixel de valor 255 alteraria o valor do pixel, resultando em uma imagem incorreta. As extensões MVI e VIS não possuem operações aritméticas de saturação. A figura 2 ilustra o resultado de uma soma de subpalavras de 8 bits sem sinal com aritmética de módulo e de saturação.

3.3 Deslocamento

As instruções de deslocamento lógico ou aritmético deslocam bits de subpalavras, e os bits deslocados para fora de uma subpalavra não interferem na subpalavra adjacente. O deslocamento paralelo pode ser usado como uma forma eficiente de se multiplicar ou dividir as subpalavras por potências de dois. O deslocamento aritmético mantém o sinal das subpalavras deslocadas. Algumas extensões, como MDMX e instruções multimídia ARMv6, podem opcionalmente saturar o resultado do deslocamento caso ocorra um *overflow*

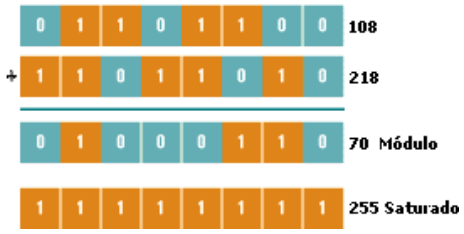

Figura 2: Exemplo de soma de subpalavras de 8 bits sem sinal com aritmética de módulo e de saturação.

com sinal no deslocamento à esquerda. As extensões MVI e VIS não suportam deslocamentos de subpalavras.

3.4 Empacotamento e Desempacotamento de dados

As instruções de empacotamento de dados convertem palavras em subpalavras, para serem usadas pelas operações paralelas. As instruções de desempacotamento realizam o processo inverso, juntam subpalavras para formarem uma palavra. As extensões MMX e AltiVec permitem o empacotamento com saturação. Por exemplo, no empacotamento com saturação de duas palavras de 64 bits, com quatro subpalavras de 16 bits cada palavra, em oito subpalavras de 8 bits, os valores são saturados caso os valores das subpalavras de 16 bits sejam maiores ou menores que os valores permitidos nas subpalavras de 8 bits. A figura 3 ilustra este empacotamento na extensão MMX.

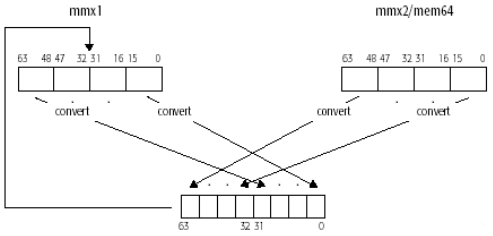

Figura 3: Exemplo de empacotamento de duas palavras de 64 bits, com quatro subpalavras de 16 bits cada palavra, em oito subpalavras de 8 bits.

3.5 Permutação

As instruções de permutação permitem o rearranjo das subpalavras, com ou sem repetição de qualquer subpalavra. As extensões SSE, VIS 2.0, AltiVec, MDMX e MAX-2 permitem a permutação de subpalavras. A figura 4 ilustra alguns exemplos de permutação de subpalavras.

3.6 Comparação

A comparação paralela de subpalavras gera, para cada par de subpalavras, um bit ou flag indicador de verdadeiro ou falso, ou uma máscara de bits todos um ou todos zero. A comparação pode testar a igualdade de subpalavras, ou se uma subpalavra é maior ou menor que outra. As extensões VIS e MDMX usam a estratégia de um bit por comparação, ilustrada na figura 5. A estratégia da máscara de bits, usada pelas extensões MMX e AltiVec, é ilustrada na figura 6. As

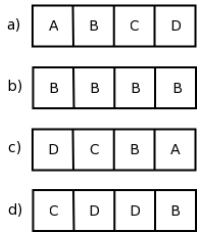

Figura 4: Exemplos de permutações de subpalavras. (a) Palavra original. (b) Subpalavra B repetida em todas as posições. (c) Subpalavras com posições invertidas. (d) Permutação com repetição de subpalavras.

extensões MAX e MVI, e instruções multimídia ARMv6 não suportam a comparação paralela de subpalavras.

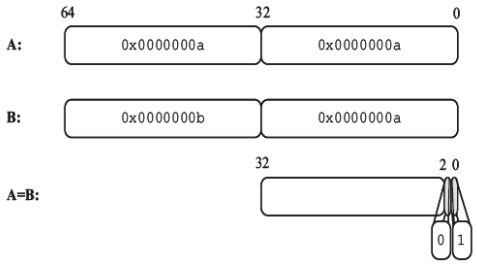

Figura 5: Comparação paralela que testa a igualdade de subpalavras de 32 bits, gerando um bit verdadeiro ou falso para cada par testado.

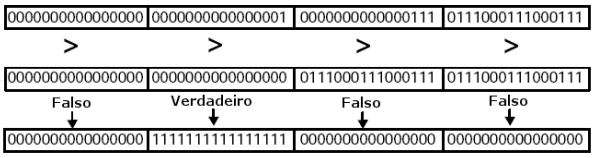

Figura 6: Comparação paralela que testa se subpalavras de 16 bits do primeiro operando são maiores que as subpalavras do segundo operando, gerando uma máscara de bits um se verdadeiro e uma máscara de bits zero se falso.

4. EXTENSÕES MULTIMÍDIA X86

Esta seção descreve as extensões multimídia presentes na arquitetura x86.

4.1 MMX

A primeira extensão multimídia para a arquitetura x86 foi introduzida pela Intel, na linha de processadores Pentium MMX [11]. Para simplificar o projeto e evitar a modificação do sistema operacional para guardar estados adicionais na troca de contexto, a extensão MMX faz uso dos oito registrados de ponto flutuante já existentes na arquitetura x86, nomeados MM0 a MM7. Esta abordagem dificulta o uso de operações de ponto flutuante e operações SIMD ao mesmo tempo. Com isso, é necessária a execução de uma instrução *(EMMS - Empty MMX State)* para a troca do estado dos registradores do modo MMX para o modo de ponto flutuante, que pode demorar até 50 ciclos de *clock* para terminar [10], em contraste às operações SIMD que podem ser executadas em um ciclo de *clock*. A tecnologia MMX introduz 57 novas instruções e adiciona três tipos de dados empacotados de 64 bits: Oito Bytes Empacotados (Packed Bytes), com sinal e sem sinal, Quatro Palavras Empacotadas (Packed Words), com sinal e sem sinal, e Duas Palavras Duplas Empacotadas (Packed Doublewords), com sinal e sem sinal. Os novos tipos de dados são ilustrados na figura 7.

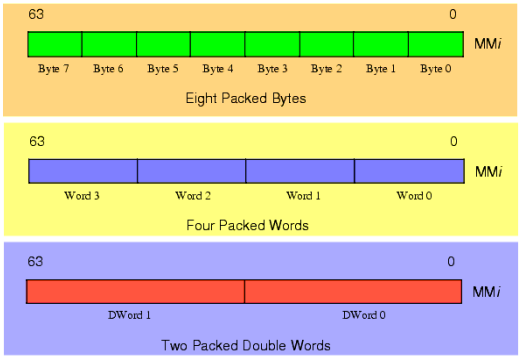

Figura 7: Novos tipos de dados suportados pela extensão MMX.

Estes tipos de dados permitem operações com oito bytes em paralelo, quatro palavras em paralelo ou duas palavras duplas em paralelo. Uma limitação do MMX é o suporte somente de operações inteiras. Segundo o Intel Media Benchmark, o uso de instruções MMX em processadores Pentium aumenta o depempenho multimídia em aproximadamente 60% [8, 9]. A figura 8 mostra a comparação do desempenho multimídia entre processadores Pentium. Segundo testes realizados por Bhargava et al. (1998), o uso de instruções MMX em algoritmos de processamento de sinais aumenta o desempenho em até 6,61 vezes [5].

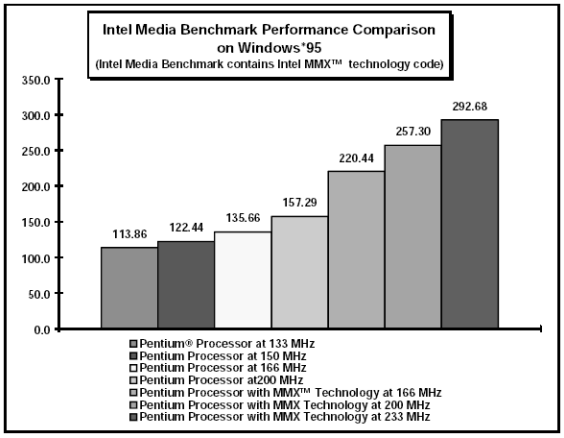

Figura 8: Comparação de desempenho multimídia entre processadores Pentium e Pentium MMX.

4.2 3DNow!

A AMD introduziu a extensão multimídia 3DNow! [1] na linha de processadores K6-2, em Maio de 1998. A tecnologia 3DNow! complementa a extensão MMX com operações

SIMD de ponto flutuante de 32 bits, e adiciona 21 novas instruções. Como essas operações usam os mesmos registradores de 64 bits da tecnologia MMX, é possível executar operações em paralelo em dois dados de ponto flutuante de 32 bits por instrução. Foi também incluída uma instrução de *prefetch*, que indica ao sistema de memória que o programa acessará o endereço de memória especificado em breve. A instrução carrega uma linha da memória cache L1 com os dados do endereço. É implementado basicamente como um *load* sem registrador de destino [16]. Como nenhum registrador de destino é modificado, a instrução pode ser retirada do fluxo de instruções logo que o endereço é calculado, minimizando o impacto na execução do programa.

4.3 SSE

Em Fevereiro de 1999, a Intel lançou a linha de processadores Pentium III com a extensão multimídia SSE (Streaming SIMD Extensions) [17]. Nesta extensão, a Intel corrigiu os dois problemas principais da tecnologia MMX: o reuso de registradores de ponto flutuante, que dificulta o uso de operações de ponto flutuante e SIMD ao mesmo tempo, e o suporte somente de operações inteiras. A extensão SSE adiciona oito novos registradores de 128 bits, nomeados XMM0 a XMM7, ilustrados na figura 9, e 70 novas instruções que operam em dados empacotados de ponto flutuante e inteiros. Cada registrador de 128 bits empacota quatro números de ponto flutuante de 32 bits. A extensão SSE não possui instruções para operações inteiras nos novos registradores, e por isso todas as operações com dados inteiros ainda devem ser realizadas nos registradores MMX de 64 bits MM0 a MM7.

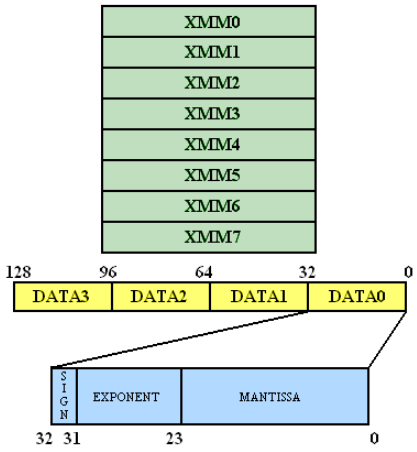

Figura 9: Novos registradores de 128 bits da extensão SSE. Os registradores armazenam números de ponto flutuante de 32 bits empacotados, seguindo o padrão IEEE (1 bit de sinal, 9 bits de expoente e 22 bits de mantissa).

Devido aos novos registradores, é necessário suporte do sistema operacional para guardar os estados adicionais nas trocas de contexto. A extensão também introduz instruções para *prefetch* de dados para a memória cache, e instruções para a gravação de dados não temporais, que não serão usados em breve. Estas instruções de gravação tentam minimizar a poluição da memória cache na gravação dos dados de registradores MMX ou SSE para a memória. O processador não grava os dados na hierarquia de cache, nem carrega os dados do endereço de escrita na linha de cache correspondente [12].

4.4 Enhanced 3DNow!

A extensão Enhanced 3DNow!, também conhecida como Extended 3DNow!, foi introduzida na linha de processadores Athlon da AMD, em Junho de 1999, e adiciona 24 instruções aos conjuntos de instruções 3DNow! e MMX já existentes[2]. As novas instruções incluem gravação de dados não temporais, conversão entre dados inteiros e de ponto flutuante, e operações matemáticas.

4.5 SSE2

A extensão SSE2 [11] foi introduzida na linha de processadores Pentium 4 em 2001. A extensão inclui 144 novas instruções e adiciona cinco novos tipos de dados: Números de ponto flutuante de 64 bits empacotados em 128 bits, Bytes Empacotados em 128 bits, Palavras Empacotadas em 128 bits, Palavras Duplas Empacotadas em 128 bits e Palavras Quádruplas Empacotadas *(Packed Quadword)* em 128 bits, ilustradas na figura 10.

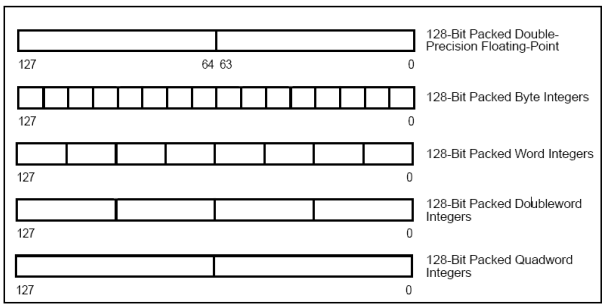

Figura 10: Tipos de dados introduzidos na extensão SSE2.

As instruções incluem operações com dados empacotados inteiros de 128 bits, que dobram o paralelismo quando comparados com as instruções MMX que operam em dados empacotados de 64 bits, e instruções de gravação de dados não temporais para os novos tipos de dados.

Na linha de processadores de 64 bits da AMD, o número de registradores SSE de 128 bits foi aumentado para 16 (XMM0 a XMM15), e só são acessíveis em programas executados no modo de 64 bits [3]. A Intel adotou os oito registradores adicionais em sua extensão de 64 bits (EM64T) da arquitetura x86.

4.6 3DNow! Professional

A extensão 3DNow! Professional foi lançada com a linha de processadores AthlonXP da AMD. Apenas integra as extensões SSE da Intel e Enhanced 3DNow! da própria AMD, sem adicionar novas intruções.

4.7 SSE3

A extensão SSE3 foi introduzida com a linha de processadores Pentium 4 com suporte a Hyper-Threading [11]. Adiciona 13 novas instruções, onde 10 são operações SIMD, uma

acelera a conversão de um valor de ponto flutuante para inteiro e as duas restantes aceleram a sincronização de *threads*. São introduzidas instruções que operam em subpalavras de um mesmo registrador, chamadas operações horizontais. A figura 11 ilustra a soma horizontal de *Quadwords*.

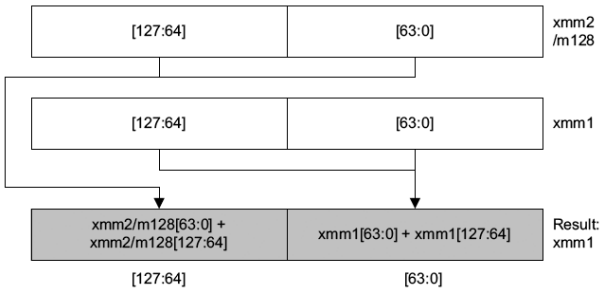

Figura 11: Soma horizontal de *Quadwords* introduzida na extensão SSE3.

5. AVALIAÇÃO

As extensões multimídia SSE e SSE2 foram testadas em uma operação de processamento de imagem. Um *threshold* é aplicado em uma imagem em níveis de cinza, onde os pixels acima do valor de *threshold* devem ficar com o valor 255 e os pixels com valor igual ou menor devem ficar com o valor 0. Os testes foram realizados no sistema operacional GNU/Linux, com os compiladores GCC 3.4.4 e ICC 9.0.021. O processador Athlon XP testado não suporta a extensão SSE2. Os resultados do teste são apresentados na tabela 1. Nota-se que todos os processadores tiveram um melhor desempenho com o uso de instruções multimídia. A implementação com instruções SSE usa os registradores MMX de 64 bits, e a implementação com instruções SSE2 usa os registradores de 128 bits para as operações inteiras. Como a implementação com SSE2 processa o dobro de pixels em paralelo, deveria obter um ganho de desempenho correspondente, mas não é o resultado observado. Apenas o processador Pentium 4 obteve um desempenho diferenciado com o uso de instruções SSE2, e os processadores da AMD mantiveram a mesma faixa de desempenho da implementação com instruções SSE. Isto demonstra que há um gargalo que impede um melhor desempenho mesmo com o dobro de pixels processados em paralelo, e pesquisas demonstram que a deficiência está na largura de banda da memória [20, 21]. O código fonte está disponível sob solicitação.

CPU	Pentium 4 3GHz HT	Athlon 64 3200+	Athlon XP 3200+	Sempron M 2800+
ICC				
flags	-O3 -march= pentium4	-O3 -march= pentiumiii	-O3 -march= pentiumiii	-O3 -march= pentiumiii
Mpixs/s				
C	464	465	470	368
SSE	1940 (4,2x)	1655 (3,6x)	1592 (3,4x)	725 (2,0x)
SSE2	2069 (4,5x)	1650 (3,6x)	sem suporte	723 (2,0x)
GCC				
flags	-O3 -march= pentium4	-O3 -march= k8	-O3 -march= athlon-xp	-O3 -march= k8
Mpixs/s				
C	297	442	396	336
SSE	1924 (6,5x)	1654 (3,7x)	1592 (4,0x)	723 (2,2x)
SSE2	2069 (7,0x)	1654 (3,7x)	sem suporte	722 (2,2x)

Tabela 1: Avaliação de desempenho das extensões SSE e SSE2, em MPixels/s, em uma aplicação de *threshold* em imagem em níveis de cinza. Os números entre parênteses indicam quantas vezes a extensão é mais rápida em relação à versão original em C.

6. CONCLUSÃO

O uso de extensões multimídia é essencial para aumentar o desempenho das aplicações, e hoje praticamente todos os processadores de uso geral possuem um conjunto de instruções SIMD. Com o uso de instruções multimídia, o gargalo de desempenho que antes era a velocidade de computação, passa a ser a largura de banda de memória. O experimento demonstra que os compiladores atuais não são capazes de otimizar da melhor forma possível os trechos paralelizáveis dos programas, e os trechos críticos ainda devem ser codificados diretamente com instruções multimídia.

7. REFERÊNCIAS

[1] Advanced Micro Devices, Inc. *3DNow! Technology Manual*, Mar. 2000.

[2] Advanced Micro Devices, Inc. *AMD Extensions to the 3DNow! and MMX Instruction Sets Manual*, Mar. 2000.

[3] Advanced Micro Devices, Inc. *AMD64 Architecture Programmer's Manual Volume 1: Application Programming*, Mar. 2005.

[4] ARM Ltd. *The ARM Architecture Version 6 (ARMv6)*, Jan. 2002.

[5] R. Bhargava, L. K. John, B. L. Evans, and R. Radhakrishnan. Evaluating mmx technology using dsp and multimedia applications. In *MICRO 31: Proceedings of the 31st annual ACM/IEEE international symposium on Microarchitecture*, pages 37–46, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press.

[6] K. Diefendorff, P. K. Dubey, R. Hochsprung, and H. Scales. Altivec extension to powerpc accelerates media processing. *IEEE Micro*, 20(2):85–95, 2000.

[7] L. Gwennap. Digital, mips add multimedia extensions. *Microprocessor Report*, 10(15), Nov. 1996.

[8] Intel Corporation. *Intel Architecture MMX Technology in Business Applications*, June 1997.

[9] Intel Corporation. *Pentium Processor with MMX Technology at 233MHz Performance Brief*, Jan. 1998.

[10] Intel Corporation. *Intel Architecture Optimization Reference Manual*, Feb. 1999.

[11] Intel Corporation. *IA-32 Intel Architecture Software Developer's Manual Volume 1: Basic Architecture*, Sept. 2005.

[12] Intel Corporation. *IA-32 Intel Architecture Software Developer's Manual Volume 2A: Instruction Set Reference, A-M*, Sept. 2005.

[13] G. Kane. *PA-RISC 2.0 Architecture*. Prentice Hall PTR, 1996.

[14] R. Lee and J. Huck. 64-bit and multimedia extensions in the pa-risc 2.0 architecture. In *COMPCON '96: Proceedings of the 41st IEEE International Computer Conference*, page 152, Washington, DC, USA, 1996. IEEE Computer Society.

[15] R. B. Lee. Realtime mpeg video via software decompression on a pa-risc processor. In *COMPCON '95: Proceedings of the 40th IEEE Computer Society International Conference*, page 186, Washington, DC, USA, 1995. IEEE Computer Society.

[16] S. Oberman, G. Favor, and F. Weber. Amd 3dnow! technology: Architecture and implementations. *IEEE Micro*, 19(2):37–48, 1999.

[17] S. K. Raman, V. Pentkovski, and J. Keshava. Implementing streaming simd extensions on the pentium iii processor. *IEEE Micro*, 20(4):47–57, 2000.

[18] P. Rubinfeld, B. Rose, and M. McCallig. Motion video instruction extensions for alpha. Technical report, Semiconductor Engineering Group, 77 Reed Road, HLO2-3/D11 Hudson, MA 01749, USA, Oct. 1996.

[19] Sun Microsystems. *The VIS Instruction Set*, June 2002.

[20] J. Sébot and N. Drach-Temam. Memory bandwidth: The true bottleneck of simd multimedia performance on a superscalar processor. In *Euro-Par '01: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*, pages 439–447, London, UK, 2001. Springer-Verlag.

[21] D. Talla, L. K. John, and D. Burger. Bottlenecks in multimedia processing with simd style extensions and architectural enhancements. *IEEE Trans. Comput.*, 52(8):1015–1031, 2003.