Conjunto de instruções Multimídia em Processadores de Propósito Geral

Fábio Augusto Menocci Cappabianco
Universidade Estadual de Campinas
Caixa Postal 6176, 13084-971 Campinas, SP - BRASIL
Campinas-SP, Brasil
fabioamc@ic.unicamp.br

ABSTRACT

Nos últimos anos, os microprocessadores receberam um novo impulso de aplicações multimídia, como áudio, vídeo e gráficos 3D. Tornou-se necessário obter maior desempenho e um tratamento mais específico da parte de processadores de propósito geral. Este trabalho é um survey a respeito de instruções multimídia em microprocessadores de propósito geral. São apresentados um histórico da computação multimídia, as principais arquiteturas desenvolvidas, uma descrição das instruções mais comuns e uma comparação de desempenho entre arquiteturas. Também são abrangidos os principais problemas e dificuldades encontrados na programação com instruções multimídia e algumas possíveis soluções e melhoras.

Categories and Subject Descriptors

H.5.1 [**Information Systems**]: Multimedia information systems; C.1.2 [**Processor Architectures**]: Multiple data Stream Architectures, SIMD; C.4 [**Performance of Systems**]

General Terms

Design Studies Performance

Keywords

MMX, SSE, 3DNow!, AltiVec, MDMX, VIS, VMX, multimedia, architecture, parallel, SIMD

1. INTRODUÇÃO

Desde a introdução dos primeiros conjuntos de instruções específicos para multimídia em processadores de propósito geral, aplicações de áudio, vídeo, gráficos 3D e outras têm tomado uma parcela cada vez maior de tempo e importância para os usuários [10].

Esta adaptação não foi trivial e ainda permite diversas melhorias, devido ao comportamento atípico dos programas multimídia. Na década passada, por exemplo, enquanto programas comuns utilizavam dados de 32 e 64 bits, aplicações de áudio e vídeo processavam dados de 8 e 16 bits [22, 21]. Além disso, algumas destas aplicações são preponderantemente seqüenciais. Portanto, foi fundamental tirar maior proveito de paralelismo com estruturas especializadas para decodificação de instruções e para a memória.

As soluções mais utilizadas atualmente são de sistemas embarcados especializados, para uma determinada aplicação multimídia, com baixo custo e baixo consumo de energia, e processadores de propósito geral como PC's para soluções genéricas que serão enfocados neste artigo.

A seção 2 contém uma breve revisão histórica do processamento multimídia. A seção 3 descreve os principais conjuntos de instrução para processadores de propósito geral, na seção 4, encontra-se uma descrição das principais instruções fornecidas pelos fabricantes e, por fim, a seção **??** conclui o trabalho.

2. HISTÓRICO DO PROCESSAMENTO MULTIMÍDIA

O processamento multimídia por computadores tem atualmente aplicações em diversas arquiteturas.

A arquitetura DSP, que ainda é bastante utilizada, remonta de 1980. Ela baseava-se em um pipeline formado por um multiplicador acoplado a uma ALU convencional. Assim, podia realizar operações do tipo MAC (multiply-and-accumulate) em paralelo. Por apresentar esta propriedade, as DSP's eram utilizadas como chips para modems e para codificação de voz, extremamente velozes para a época em comparação aos processadores de propósito geral. [12]

Em 1991, os processadores de arquitetura DSP ganharam campo em processamento de vídeo devido ao aumento de sua freqüência de execução e a utilização em modo de operação single instruction stream, multiple data stream(SIMD). Em 1993, os processadores DSP ganharam campo em processamento multimídia de aplicações no formato MPEG-2, devido ao uso de very long instruction words(VLIW) no modo de operação SIMD, denominado SIMD+ [15]. Estes processadores passaram a ser muito utilizados em receptores e tratadores de sinal de TV a cabo e de vídeo sob demanda, DVD's e outros dispositivos.

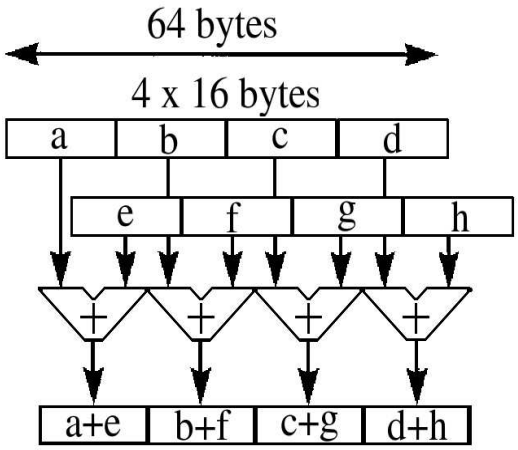

Figure 1: Exemplo de adição de palavras pequenas.

Alguns chips de aplicação específica também ganharam bastante espaço. Os chips de aceleração gráfica 3D em PC's, por exemplo, são um grande avanço no tratamento de funções multimídia, desde o início dos anos 90. Eles tratam de maneira extremamente rápida a renderização de polígonos em cenas tridimensionais [9]. Aplicações em FPGA's também ganharam seu espaço para processamento de imagens e sons se mostrando uma alternativa barata e de fácil implementação, com um desempenho muito acima dos processadores de propósito geral [19, 2].

Os microprocessadores, por fim, começaram a ficar interessantes para o tratamento de aplicações multimídia a partir do momento em que apresentaram um conjunto de instruções específico para elas. Como os dados multimídia são representados por 8 ou 16 bits e as arquiteturas possuíam registradores e ALU's que tratavam menos 32 bits, algum paralelismo pôde ser obtido por dividir uma ALU ou registrador de palavras longas em várias ALU's ou registradores de palavras pequenas, denominadas palavras ou dados particionados. Desta forma cada registrador e ALU particionados ficam responsáveis por processar um dado multimídia.A Figura 1 ilustra uma adição de palavras particionadas de 16 bits provenientes da divisão de uma palavra de 64 bits utilizando também várias ALU's pequenas que juntas formam uma ALU grande.

O paralelismo em microprocessadores de aplicações multimídia iniciou-se com instruções gráficas [8]. A seguir, aplicou-se também à decodificação de imagens [13]. Por fim, abrangeu o processamento de vídeos. A Tabela 1 contém alguns dos conjuntos de instrução mais conhecidos neste contexto.

Estes conjuntos de instrução foram tentativas de gerar um suporte mais apropriado para aplicações variadas de multimídia. Um novo conjunto de instruções era idealizado na tentativa das empresas de atender melhor à demanda do mercado e assim conseguir estabelecer o seu padrão como o principal. Atualmente, o padrão mais utilizado é o SSE em suas três versões, adotado tela Intel e pela AMD. Segue uma breve descrição destes conjuntos.

3. CONJUNTOS DE INSTRUÇÃO

Como mostra a Tabela 1 existe uma grande variedade no tipo e tamanho das extensões multimídia. Inicialmente pode-se classificar as extensões em três classes:

1. as que compartilham o banco de registradores inteiro e, conseqüentemente suas unidades de processamento, com os registradores e instruções multimídia;
2. as que compartilham o banco de registradores e unidades de processamento de ponto flutuante com os registradores e instruções multimídia;
3. as que possuem um banco de registradores exclusivo para registradores e instruções multimídia.

A primeira classe caiu em desuso por questões arquiteturais que tornam mais complexa a implementação e pode acarretar em perda de desempenho. Também por tomar espaço de operações com ponteiros, constantes e variáveis de desvios que são inteiros com dados multimídia, piorando ainda mais o desempenho dos programas.

A segunda classe ainda é utilizada em algumas arquiteturas, pois dificilmente instruções de ponto flutuante são utilizadas juntamente com instruções multimídia.

Já a terceira classe é a mais comum atualmente, principalmente por causa da demanda por registradores multimídia de 128 bits, para ganhar mais paralelismo, enquanto os registradores de ponto flutuante variavam entre 64 e 80 bits.

A seguir, há uma breve descrição de cada uma destes padrões quanto às suas instruções, metas de projeto e outras observações.

3.1 MAX-1 da HP

MAX-1 da Hewlett Packard foi a primeira extensão multimídia para microprocessadores que operava apenas sobre números inteiros. Ele permitia a decodificação de software MPEG-1 de resolução 352x240 pixels em um CPU PA-7100LC de 80MHz. Antes de possuir instruções multimídia, a estação HP720 de 50MHz processava um vídeo sem áudio em 4-5fps. O PA-7100LC foi idealizado para realizar operações multimídia, sem impactar no agendamento, complexidade, período de clock e área de chip, sendo portanto, bastante simples [14]. Como se tratava de uma arquitetura RISC e seguiu-se as mesmas diretivas de projeto da mesma, observou-se as operações multimídia mais freqüentes, quebrando-as em primitivas mais rápidas que foram implementadas em hardware [13]. Só com esta melhoria um vídeo com áudio podia ser executado a 30fps.

3.2 VIS da Sun

As operações do Visual Instruction Set (VIS) também atuam apenas sobre inteiros, utilizando, porém o banco de registradores de ponto flutuante. O VIS foi implementado inicialmente nos processadores UltraSPARC I, II e III e sua

Table 1: Principais conjuntos de instrução multimídia

Conjunto de Instuções	Empresa	Ano	instruções	banco de registradores
MAX-1	HP	1994	9	Integer(31x64b)
VIS	SUN	1995	121	FP(32x64b)
MAX-2	HP	1995	8	Integer(32x64b)
MDMX	MIPS	1996	29	FP(32x64b)+Ac(1x192b)
MIPS-64	MIPS	1996	74	FP(32x64b)
MMX	Intel	1997	57	FP(8x64b)
MVI	DEC	1997	13	Integer(31x64b)
Extended MMX	Cyrix	1997	12	FP(8x64b)
3DNow!	AMD	1998	21	FP(8x64b)
SSE	Intel	1999	70	8x128b
AltiVec	Motorola	1999	162	32x128b
MIPS-3D	MIPS	1999	23	FP(32x64b)
Enhanced 3DNow!	AMD	1999	24	FP(8x64b)
SSE2	intel	2000	144	8x128b
3DNow! Professional	AMD	2000	70	8x128b
SSE3	intel	2004	157	8x128b

principal motivação foi de criar uma plataforma padrão para aplicações multimídia como visualização 3D e codificação MPEG-1, MPEG-2. Antes do VIS, aplicações gráficas necessitavam de hardware especializado, e a partir de então o custo de sistemas diminuiu e slots de extensão foram economizados. O conjunto de instruções do VIS foi gerado, baseando-se em aplicações gráficas e multimídia. Todas as instruções são caracterizadas por: serem executadas em apenas um ciclo ou serem particionada em pipeline facilmente; serem aplicáveis a diversos algoritmos; e não afetarem o período de clock [24].

3.3 MAX-2 da HP

O MAX-2 fez parte de uma segunda abordagem multimídia utilizada pela HP em uma processadores da arquitetura PA-RISC 2.0. Novamente o MAX-2 oferece um conjunto reduzido de primitivas para as instruções multimídia, no entanto a arquitetura PA-RISC possui recursos adicionais. Por exemplo, duas abordagens diferentes foram utilizadas para multiplicação, dependendo do fluxo de mídia a ser processado. Para aplicações de áudio e gráficos 3D, uma unidade multiplicadora e acumuladora para ponto flutuante é utilizada. Para aplicações de baixa precisão, como multiplicação por constantes, estas são realizadas por séries de shifts e adições provenientes de instruções MAX-2 [**?**].

3.4 MDMX da MIPS

O padrão MCMX especifica que um banco de processadores de ponto flutuante contendo 32 registradores de 64 bits seja compartilhado para o uso de instruções multimídia sobre números inteiros. Além destes, foi especificado um registrador de 192 bits com o propósito de ser acumulador de operações. Este acumulador é especialmente útil pois operações de multiplicação ou uma sequência de somas pode ocupar mais bits que os registradores fontes suportam, obtendo-se assim resultados mais precisos [20].

3.5 MIPS-64 da MIPS

O conjunto de instruções MIPS-64 foi o primeiro a implementar instruções multimídia para registradores de ponto flutuante a serem executados dentro do microprocessador. Seu principal objetivo foi complementar o MDMX para aplicações de OpenGL como geometria 3D e Virtual Reality Modeling Language(VRML) dentre outras [11].

3.6 MMX da Intel

O padrão MMX se limitou a instruções multimídia de números inteiros, utilizando o conjunto de registradores de ponto flutuante a partir do processador Pentium P55. Isto impossibilitava que operações multimídia e de ponto flutuante fossem executadas simultaneamente. Foi com certeza o mais popular de todos os conjuntos de instrução até a própria Intel lançar o conjunto de instruções SSE [18].

A prioridade do projeto do MMX foi de aprimorar o processamento de aplicações multimídia, de comunicações de internet, incluindo vídeos nos formatos MPEG-1 e MPEG-2, síntese de músicas, compressão e reconhecimento de fala, processamento de imagens, gráficos 3D, vídeo conferências, áudio e modems. Na avaliação da Intel, desejou-se executar com maior eficiência as seqüências de código mais freqüentes e desejou-se também manter a compatibilidade com a arquitetura dos processadores anteriores.

3.7 MVI da DEC

As instruções Motion Vídeo Instructions (MVI) foram incorporadas primeiramente pelo Alpha 21164PC. A DEC, idealizadora do MVI, foi comprada pela HP. O principal foco da DEC, para o padrão MVI eram a reprodução de teleconferências e DVS a 30 fps com áudio stereo. Desejava-se a reprodução comparável aos hardwares dedicados. [3] A DEC implementou apenas 13 instruções de multimídia alegando que a largura de banda de memória era insuficiente para alimentar o processador com tanto paralelismo. A DEC alegou também que o conjunto MMX era complexo devido às deficiências das arquiteturas anteriores que necessitavam ser cobertas [17].

3.8 Extended MMX da Cyrix

Assim como a AMD a Cyrix licenciou o uso da extensão MMX da Intel. A Cyrix não produz mais processadores desde 1997 quando juntou-se à National Semiconductors.

A Cyrix expandiu as instruções de MMX para evitar destruição de dados durante operações. Enquanto o padrão MMX utilizava como registrador de destino um dos registradores de origem, o padrão Extended MMX implicitamente escolhia um registrador de destino diferente, dependendo dos registradores de origem [25].

3.9 3DNow! da AMD

Depois de licenciar o padrão MMX da Intel a AMD expandiu este conjunto de instruções, criando o padrão 3DNow! para o processador K6. Esta extensão utiliza as mesmas instruções multimídia para números inteiros e o mesmo banco de registradores. A inovação foi incrementar instruções para números de ponto flutuante. Duas operações de ponto flutuante de precisão simples eram executadas ao mesmo tempo. O motivo da AMD utilizar a expansão MMX da Intel foi de não querer gastar recursos com uma arquitetura que requeresse maior área ou de complexidade, sem muitos benefícios de desempenho [16].

3.10 SSE da Intel

Streaming SIMD Extension (SSE) foi a primeira expansão da Intel ao conjunto de instruções MMX. Quando proposta era denominada Internet Streaming SIMD Extension (ISSE), mas a aplicação foi além da Internet. Ela foi desenvolvida para cálculos de geometria 3D, renderização, codificação e decodificação de vídeos e reconhecimento de voz. A grande diferença desta arquitetura é de suportar cálculos com números de ponto flutuante. Além disso, incorporou instruções sugeridas por vendedores de software multimídia.

O conjunto de instruções SSE trouxe outra grande alteração na arquitetura Intel, o que não ocorria desde a migração de 16 para 32 bits do 80286 para o 80386. Esta mudança foi a adição de um novo conjunto de registradores exclusivo para operações multimídia, tirando os dados multimídia dos registradores de ponto flutuante. Isto simplificou a implementação de novos processadores e possibilitou a utilização de instruções de ponto flutuante ao mesmo tempo que instruções multimídia eram processadas.

O Pentium III foi o primeiro processador a adotar o padrão SSE. Ele foi desenvolvido para executar duas operações de ponto flutuante de 64 bits por ciclo de clock, operando assim com até quatro dados de 32 bits, somando 128 bits ao mesmo tempo [23].

3.11 AltiVec da Motorola

O conjunto de instruções AltiVec foi utilizado pelo processador MPC 7400 da estação Apple G4, adicionando ao PowerPC unidades de execução de 128 bits. Ao contrário de muitas arquiteturas, que investiram limitadamente em instruções multimídia, o AltiVec ocupa grande parte da área do processador com inovações para o processamento multimídia, pois está mais focado em funcionalidades que em custos. Além disso, o paralelismo oferecido pelo AltiVec se aplica não apenas às instruções multimídia, mas a qualquer uma que possa atuar paralelamente.

O AltiVec, assim como o SSE utiliza um banco de registradores exclusivo para o processamento multimídia. Este banco de registradores, sendo de 32 bits facilita o trabalho de compiladores para paralelizar códigos de programas em pipeline e loop unrolling. As instruções desta extensão também são bem peculiares, pois permitem a especificação de quatro operandos (três fontes e um destino). Isto proporciona um grande largura de banda, permite instruções extras como multipply-add, permute, etc e evita trocas de valores entre registradores que perdem dados como no caso do SSE, acelerando bastante o processamento [4].

3.12 MIPS-3D da MIPS

O conjunto de instruções MIPS-3D consiste em uma extensão especifica para aplicações (ASEs), que são opcionais para se adicionar ao MIPS-64. Também foi projetado de maneira simples para sistemas embarcados de baixo consumo, a fim de realizarem processamento 3D [6].

3.13 Enhanced 3DNow! da AMD

Esta extensão da AMD aos seus conjuntos de instrução MMX e 3DNow! foi utilizada no processador Athlon. Ela adicionou operações particionadas de inteiros e de pontos flutuantes para que ficasse funcionalmente equivalente ao SSE.

3.14 SSE2 da Intel

Este conjunto de instruções foi inserido no processador Pentium 4. Ele adicionou basicamente instruções inteiras de multimídia realizadas nos registradores extras de 128 bits, que eram realizadas nos registradores de ponto flutuante e adicionou o tipo de dado de dupla precisão para aplicações científicas, de engenharia e raytracing [1].

3.15 3DNow! Professional da AMD

A extensão 3DNow! Professional foi utilizada no processador Athlon MP da AMD. Ela contém um conjunto extra de instruções para fazer total compatibilidade com o conjunto de instruções SSE e adicionalmente ao seu próprio conjunto de instruções Enhanced 3DNow! [5].

3.16 SSE3 da Intel

O SSE3 foi introduzido no mercado na versão Prescott do Pentium 4. As instruções novas deste conjunto ajudam a sincronizar threads e auxiliam em aplicações específicas como mídia e jogos. [7]

4. INSTRUÇÕES MULTIMÍDIA

Esta seção descreve as principais instruções multimídia dos conjuntos apresentados anteriormente. Primeiramente serão apresentadas as instruções sobre números inteiros, depois sobre as de ponto flutuante, seguidas das polimorfas, seguidas das de comparação e de fluxo e por último as sobre memória.

4.1 Instruções Aplicadas a Inteiros

Todas as extensões apresentadas tratam de números inteiros, apesar de haver grandes variações de tipo e largura dos dados utilizados em cada uma delas. A Tabela 2 lista as instruções inteiras nas arquiteturas que foram examinadas na seção anterior.

4.1.1 Conversão de Tipo e Largura de Dados

Mudança na largura do dado é fundamental para instruções multimídia. A operação chamada empacotamento reduz a largura de um dado. Por exemplo, um registrador de 64

Table 2: Instruções multimídia de números inteiros das arquiteturas. Os números representam a quantidade de bits dos operandos de cada instrução. Quando precedidos de 'U','S' e 'FP' operam sobre inteiros sem sinal inteiros sinalizados e números de ponto flutuante, respectivamente. Quando seguidos de '*', significa que a instrução é possível, mas através de outras instruções.

	AMD	Cyrix	DEC	HP	Intel	MIPS	Motorola	Sun
Add/Sub Resto	8,16,32	8,16,32	-	16	8,16,32,64	8,16	8,16,32	16,32
Add/Sub Sat.	8,16	8,16	-	16	8,16	S16	8,16,32	-
Média	U8,U16	U8	-	S16	U8,U16	-	8,16,32	-
Min/Max	U8,U16	S16	U8,S16	-	U8,S16	U8,S16	8,16,32	-
Mul Trucada	US16	S16	-	-	US16	-	S16	S16,S32
Mul Arredond	S16	S16	-	-	-	-	S16	-
Mul Completa	US16	S16	-	-	S16,U32	U8,S16	US8,US16	-
Right Shift Lgc	16,32,64	16,32,64	-	16	16,32,64	8,16	8,16,32	-
Right Shift Art	16,32	16,32	-	16	16,32	8,16	8,16,32	-
Left Shift Lgc	16,32,64	16,32,64	-	16	16,32,64	8,16	8,16,32	-
Merge	8,16,32	8,16,32	-	-	8,16,32	32	8,16,32	8
Align/Rotate	-	-	-	-	-	8	8	8
Shuffle	$16(4^4)$	-	-	$16(4^4)$	$16(4^4),32(4^4)$	8(8),16(8)	$8(32^16)$	-
Align/Rotate	16	-	-	-	16	-	8,16,32	-
Pack FP Satur	FP32→S32,16	-	-	-	FP32→S32	-	FP32→US32	-
Pack FP Trunc	-	-	-	-	FP64→FPS32	-	-	-
Pack 32b Satur	S32→S16	S32→S16	-	-	S32→S16	-	S32→S16	S32→8,16
Pack 16b Satur	S16→US8	S16→US8	-	-	S16→US8	-	S16→US8	S16→U8
Pack 32b Trunc	32→16*,8*	32→16*8*	32→8	32→16	32→16*,8*	32→16	32→16	-
Pack 16b Trunc	16→8*	16→8*	16→8	-	16→8*	16→8	16→8	-
Unpack FP	-	-	-	-	FP32→FP64	-	-	-
Unpack 32b	S32→FP32	-	-	-	S32→FP32,64	-	US32→FP32	-
Unpack 16b	US16→U*FP32	U16→U32*	-	16→32*	U16→U32*	U16→U32	U16→U32*	U16→U32
Unpack 8b	U8→U16*	U8→U16*	8→16,32	-	U8→U16*	US8→US16	U8→U16*	U8→U16*

bits contendo 4 palavras de 16 bits pode ser convertido por colocar os 8 bits menos ou mais significativos de cada palavra de 16 bits em 32 bits de outro registrador. A operação de desempacotamento, por sua vez aumenta a largura de um dado adicionando zeros à esquerda ou expandindo conforme o sinal do dado representado.

A expansão MVI possui empacotamento apenas dos bits mais significativos e desempacotamento apenas com expansão de zeros. A MAX, por sua vez, não possui operações diretas de empacotamento desempacotamento, utilizando outros recursos para executar tais operações.

Apesar dos dados em PC's atuais sejam de precisão de 32 e 64 bits, dados multimídia muitas vezes operam sobre 8 e 16 bits, por isso, as operações diversas de empacotamento e desempacotamento são extremamente importantes.

4.1.2 Saturação e Overflow

Tradicionalmente, nas instruções com número inteiros, se um ocorrer um overflow ou um underflow, a operação retorna o resto da divisão do valor resultante pelo número máximo ou mínimo que a arquitetura pode representar, respectivamente. Este comportamento é indesejável em muitas aplicações multimídia. Em processamento de imagens por exemplo, em uma soma de brilhos de pixels, caso o valor total exceda o máximo representável é desejável que o total retornado pela operação seja o brilho máximo e não o resto da divisão do total pelo número pelo máximo. A Figura 2 demonstra este tipo de operação que é denominada 'saturada'.

A implementação de aritmética saturada é um pouco diferente da aritmética comum, e exige um tratamento diferente para números com e sem sinal, o que aumenta um pouco a complexidade do hardware. Isso pesa na escolha do banco de registradores a ser utilizado pelas instruções multimídia. Caso sejam implementadas sobre os registradores inteiros como no caso das extensões MAX e MVI, pode haver aumento no número de ciclos para as operações básicas aritméticas, ou um aumento no período do clock. Isto com certeza seria uma característica indesejável.

4.1.3 Adição e Subtração

A Tabela **??** lista os tipos de adição e subtração nas arquiteturas que foram examinadas na seção anterior. A extensão AltiVec destaca-se por incluir instruções de 32 bits para adição e subtração saturada. No entanto, operações multimídia sobre som e imagem utilizam dados de 8 ou 16 bits, o que reduz a abrangência desta. Há um contraste com a extensão VIS da Sun que não possui adições ou subtrações com saturação, já que foi projetada para atuar preferencialmente sobre imagens. Apesar das extensões MIPS só possuírem operações de adição saturada para inteiros sinalizados de 16 bits, o acumulador da arquitetura consegue lidar eficientemente com dados enormes devido ao seu tamanho de 192 bits.

Somente o conjunto da DEC não possui adições sobre números particionados, requerendo operações intermediárias para evi-

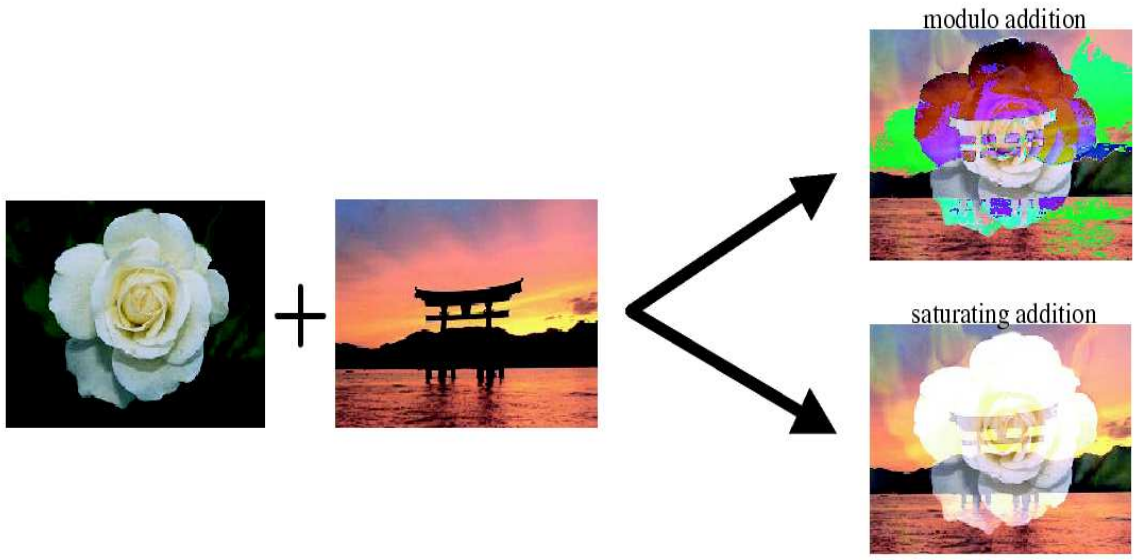

Figure 2: Adição de brilhos de pixels em imagem saturada e imagem de resto.

tar overflow e underflow.

4.1.4 Soma das Diferenças Absolutos

Uma das poucas instruções que a DEC colocou na extensão MVI é a de calcular a soma de da diferença absoluta de bytes desempacotados. O conjunto MMX, ao contrário, apesar de ser muito mais rica não possui tal instrução. Ela foi incluída nas arquiteturas da Intel, Cyrix e AMD em suas respectivas expansões do conjunto MMX, assim como a VIS. Esta instrução produz um grandioso benefício para instruções multimídia com um paralelismo de 8x para arquiteturas de 64 bits e substitui muitas operações escalares. Na decodificação de vídeos MPEG-2, por exemplo, a porção central do código pode ser substituída por duas operações de soma das diferenças absolutas.

4.1.5 Multiplicações

A multiplicação de elementos particionados, envolve a multiplicação dos elementos empacotados respectivos. As multiplicações particionadas são as instruções com a maior diferença entre as implementações. Elas são mais demoradas, levando de três a cinco ciclos e ocupam três vezes mais espaço no chip. Além disso, são difíceis de manejar, devido ao resultado ocupar até o dobro do espaço dos operandos fontes.

A complexidade da multiplicação particionada impediu que as extensões que possuem seu banco de registradores multimídia compartilhado com o banco de registradores inteiros implementássem-nas, que são o caso da MVI, MAX-1 e MAX-2. Por esta e outras razões, o compartilhamento do banco de registradores inteiros mostrou ser uma má idéia.

Por ser maior que os operandos de origem, o resultado da multiplicação teve diversos tratamentos.

1. **Redução: retorna a soma de elementos particionados do resultado da multiplicação, atuando como um multiply-add. A extensão MMX implementou esta instrução.**
2. **Par/impar: selecionar os elementos pares ou ímpares do resultado da multiplicação. AltiVec implementou este tratamento.**
3. **Truncado: São pré-determinados alguns bits que são perdidos após a multiplicação. Normalmente os bits menos significativos. A extensão MMX implementou esta instrução.**
4. **Registrador de destino com maior largura: Este é o caso do acumulador de 192 bits implementado na arquitetura MIPS com conjunto de instruções MDMX. Os acumuladores, no entanto, mostraram ser ao mesmo tempo um obstáculo para o aumento da velocidade de processamento, uma vez que todas as operações que causam overflow ou underflow passam por ele. Além disso, o acumulador impede o processamento superscalar como demonstra a história da arquitetura [11].**
5. **Primitivas de multiplicação: Esta solução adotada pela VIS é problemática. Ao invés de gerar o resultado completo de uma multiplicação, esta extensão possui primitivas ou sub-instruções para gerar o resultado. Isto aumenta as dependências de nomes, o que afeta o paralelismo e gera a necessidade por mais registradores.**

Uma opção muito sugerida por pesquisadores para minimizar os conflitos de instruções que geram números mais

largos que os operandos de origem é de utilizar uma representação dos dados diferente para armazenamento e para execução de instruções multimídia. Esta diferença de tamanhos reduz o gasto com instruções de empacotamento/desempacotamento, pois os dados são lidos da memória para o formato de execução e são rearranjados com saturação ou resto, se necessário para a armazenagem [21].

4.1.6 shifts ou deslocamentos

Os shifts servem para realizar multiplicações e divisões por potências de dois facilmente, alinhar dados, entre outras aplicações. São especialmente importantes em arquiteturas que não fornecem instruções de multiplicação.

4.1.7 Operações de comunicação de dados

Este tipo de instruções serve basicamente para rearranjar palavras particionadas em um registrador e entre registradores. Segue uma lista com as principais instruções de comunicação.

- *Merge:* alterna os elementos da parte superior ou inferior de dois registradores, a instrução
- *Align ou rotate:* permite o re-ajuatamento de palavras particionadas em dois registradores, as instruções
- *Insert e extract:* permitem que um elemento empacotado seja extraído como um escalar ou que um escalar seja inserido como elemento empacotado, a instrução.
- *Shuffle:* Permite que se escolha uma ordem diferente para elementos de um palavra. Os conjuntos de instrução MDMX, MAX-1, MAX-2, MMX e AltiVec. Apesar de se mostrar uma excelente instrução é muito custosa para se implementar, pois exige um registrador auxiliar e muitos barramentos adicionais para a execução.

4.2 Instruções Aplicadas a Números de Ponto Flutuante

A maioria das aplicações de ponto flutuante para multimídia trabalha com dados de precisão simples (32 bits), ou precisão dupla (64 bits). Todas as extensões multimídia iniciaram com suporte apenas para dados de precisão simples. A primeira dar suporte a precisão dupla foi a SSE2 da Intel. A partir de então, as outras extensões passaram a ter este recurso.

Apenas quatro linhagens de conjuntos de instrução aplicam-se a ponto flutuante multimídia: SSE, 3DNow!, AltiVec e MIPS-64/3D.

Além das instruções básicas aritméticas, as instruções de ponto flutuante particionadas incluem algum tipo de aproximação de inversão e raiz quadrada. Estas instruções são muito utilizadas pela geometria 3D. Elas são implementadas através de tabelas pré-estabelecidas, que facilitam seu cálculo. As extensões SSE e AltiVec retornam valores com precisão de 12 bits para estas operações, a 3DNow! retorna 14 e 15 bits e a MISP-3D retorna precisão de 24 bits.

Exceções são um problema para operações de ponto flutuante, pois é muito complexo determinar o elemento que gerou a exceção. Portanto, os conjuntos 3DNow! não geram instruções. O conjunto de instruções AltiVec, por outro lado, ao invés de gerar uma interrupção, faz com que o elemento do resultado da operação, onde ocorre overflow ou underflow, sofra alguma alteração, assim como no caso da saturação.

Os conjuntos SSEs incluem registradores de controle/status para exibir as exceções, no entanto o elemento particionado que sofreu a exceção não é apontado. O MIPS-64 tem uma abordagem semelhante aos SSEs.

4.3 Operações Polimorfas

Este tipo de instrução é aplicado aos dados independentemente de seu particionamento. As extensões MVI e MAX-1 e 2 não possuem nenhuma adição de lógica para suas arquiteturas, pelo fato de pertencerem à primeira classe de conjunto multimídia, que compartilha o banco de registradores e unidades de processamento inteiros. Estas unidades de processamento já são providas da lógica necessária, e portanto, não requer instruções adicionais.

No caso do MIPS, somente a extensão MDMX provê lógica adicional, utilizada sobre as unidades de processamento de ponto flutuante compartilhadas. As demais arquiteturas possuem este tipo de instrução.

4.4 Comparação e Controle de Fluxo

A computação em paralelo fica complicada caso uma das operações deva ser realizada condicionalmente. Como visto anteriormente, não há como gerar um flag ou exceção diferente para cada operação executada paralelamente. Duas soluções para o problema foram idealizadas.

A primeira é utilizar elementos máscara. Estes são conjuntos de bits do tamanho do elemento comparado compostos só de zeros (0x0000 para 16 bits) ou somente de uns (0xffff para 16 bits). Estes elementos são utilizados em conjunto com operações lógicas do tipo AND, NAND ou OR para atingir o resultado desejado. A vantagem de se utilizar este tipo de tratamento é de não criar dependências extras com os elementos máscara, mas ter somente a dependência do resultado da operação lógica.

A segunda consiste em gerar um vetor de bits, onde um único bit representa o resultado de uma comparação lógica para cada elemento particionado. Os bits restantes podem ser armazenados se necessário.

4.5 Memória

As operações de memória para multimídia são apenas Load e Store. Não utiliza-se referência a memória em nenhuma instrução de nenhum dos conjuntos de instrução multimídia apresentados.O tipo mais comum de leitura e escrita são os seqüenciais, indicados apenas pelo endereço do dado a ser lido/escrito e seu tamanho.

No entanto, algumas aplicações são mais difíceis de serem tratadas no seu fluxo de dados da memória. Isso acontece muito em imagens com três canais, RGB, onde cada canal é armazenado seqüencialmente, e deseja-se adquirir as três componentes de um pixel que estão situadas em posições

diferentes da memória, porém com certa regularidade de intervalo. Assim, o load e store poderiam ser realizados a partir de três atributos: largura dos elementos lidos; deslocamento do dado lido a partir do endereço base para iniciar a leitura/escrita; e distância de um endereço efetivo de um elemento para o próximo a ser lido.

Mais detalhes sobre instruções de comparação e controle de fluxo e de memória encontram-se em [22, 21, 12].

5. CONCLUSÕES

O aumento da demanda e do número de aplicações multimídia levou à introdução dos conjuntos de instruções específicos para multimídia em processadores de propósito geral.

Como visto, surgiram diversas extensões de grandes empresas, com a finalidade de atender a esta demanda. Diferentemente do que ocorreu em relação às instruções comuns de propósito geral, os projetos mais ousados e com mais recursos foram mais bem sucedidos. Enquanto a MVI e a MAX-1 e 2 tiveram desempenho baixo, arquiteturas mais complexas como SSE e AltiVec obtiveram um resultado melhor.

Outra visão dada pelo artigo foi a divisão das classes de expansões multimídia, quanto ao banco de registradores e seus recursos de processamento. Dentre as classes, a que destacou mais foi a que utiliza os bancos de registradores específicos para multimídia, pois facilitam a implementação e fornecem maior grau de paralelismo.

A última contribuição foi a visão geral dos tipos de instruções multimídia existentes, com uma análise crítica de algumas delas e a correlação com as extensões que as fornecem.

6. REFERENCES

[1] A. Kumar. *SSE2 Optimization - OpenGL Data Stream Case Study*, 1998.

[2] M. G. Alina N. Moga. Parallel image component labeling with watershed transformation. *IEEE Transactions on Pattern Analisys and Machine Inteligence*, 19(5):441–450, May 1997.

[3] R. Carlson, D.A.; Castelino and R. Mueller. Multimedia extensions for a 550-mhz risc microprocessor. *IEEE Journal of Solid-State Circuits*, 32:1618–1624, Nov 1997.

[4] K. Diefendorff, P. Dubey, R. Hochsprung, and H. Scale. Altivec extension to powerpc accelerates media processing. *Micro, IEEE*, 20:85–95, Apr 2000.

[5] J. Huynh. The amd athlontm mp processor with 512kb l2 cache. *AMD White Paper*, pages 1–16, May 2003.

[6] M. T. Inc. A process-portable 64b embedded microprocessor with graphicsextensions and a 3.6 gb/s interface. *IEEE International Solid-State Circuits Conference, 2001. Digest of Technical Papers. ISSCC.*, pages 234–235,451, 2001.

[7] Intel. Processors - define sse2 and sse3. *Intel White Paper*, page 1, Oct 2005.

[8] Intel corporate literature. *i860TM microprocessor family programmers reference manual*, 1992.

[9] Y.-W. Jeon, Y.-S. Kwon, Y.-H. Im, J.-H. Lee, S.-J. Nam, B.-W. Kim, and C.-M. Kyung. 3d graphics system with vliw processor for geometry acceleration. *Proceedings of the Second IEEE Asia Pacific Conference on ASICs - AP-ASIC*, pages 367–370, Aug 2000.

[10] M. V. Jesus Corbal, Roger Espasa. Mom: a matrix simd instruction set architecture for multimedia applications. *ACM/IEEE conference on Supercomputing - CDROM*, pages 1–12, Jan 1999.

[11] D. P. John Hennessy. *Computer Architecture A Quantitative Approach, Fourth Edition*. Elsevier, 2006.

[12] T. Kuroda, I.; Nishitani. Multimedia processors. *Proceedings of the IEEE*, 86:1203–1221, Jun 1998.

[13] R. Lee. Accelerating multimedia with enhanced microprocessors. *Micro, IEEE*, 15:22–32, Apr 1995.

[14] R. Lee. Realtime mpeg video via software decompression on a pa-risc processor. *'Technologies for the Information Superhighway' Compcon.*, pages 186–192, Mar 1995.

[15] T. Nishitani. Trend and perspective on domain specific programmable chips. *Signal Processing Systems, SIPS - IEEE Workshop on Design and Implementation.*, pages 1–8, Nov 1997.

[16] G. W. F. Oberman, S.; Favor. Amd 3dnow! technology: architecture and implementations. *Micro, IEEE*, 19:37–48, Apr 1999.

[17] M. M. P Rubinfeld, B Rose. Motion video instruction extensions for alpha. *Digital Equipment Corporation-enlight.ru*, pages 1–13, Oct 1996.

[18] U. Peleg, A.; Weiser. Mmx technology extension to the intel architecture. *Micro, IEEE*, 16:42–50, Aug 1996.

[19] I. Rambabu, C.; Chakrabarti and A. Mahanta. Flooding-based watershed algorithm and its prototype hardware architecture. *Image and Signal Processing, IEE Proceedings-Vision*, 151:224–234, Jun 2004.

[20] Silicon Graphics. *MIPS Extension for Digital Media with 3D*, 1997.

[21] A. Slingerland, N.; Smith. Measuring the performance of multimedia instruction sets. *IEEE Transactions on Computers*, 51:1317–1332, Nov 2002.

[22] N. T. Slingerland and A. J. Smith. Multimedia instruction sets for general purpose microprocessors: a survey. Technical report, Berkeley, CA, USA, 2000.

[23] T. Thakkur, S.; Huff. Internet streaming simd extensions. *Computer*, 32:26–34, Dec 1999.

[24] J. Tremblay, M.; O'Connor. Ultrasparc i: a four-issue processor supporting multimedia. *Micro, IEEE*, 16:42–50, Apr 1996.

[25] VIA-Cyrix. *Application Note 108: Cyrix Extended MMX Instruction Set*, 1998.