UNIVERSIDADE FEDERAL DE PELOTAS

Programa de Pós-Graduação em Computação

Dissertação

Estudo de Arquiteturas Reconfiguráveis para Aumento de Desempenho de Aplicações DSP

Júlio César Mesquita Ruzicki

Pelotas, 2014

JÚLIO CÉSAR MESQUITA RUZICKI

ESTUDO DE ARQUITETURAS RECONFIGURÁVEIS PARA AUMENTO DE DESEMPENHO DE APLICAÇÕES DSP

Dissertação apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Júlio Carlos Balzano de Mattos

Pelotas, 2014

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação na Publicação

R986e Ruzicki, Júlio César Mesquita

Estudo de arquiteturas reconfiguráveis para aumento de desempenho de aplicações DSP / Júlio César Mesquita Ruzicki ; Júlio Carlos Balzano de Mattos, orientador. — Pelotas, 2014.

75 f. : il.

Dissertação (Mestrado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2014.

1. Processamento digital de sinais. 2. Reconfiguração. 3. Reconfiguração dinâmica. 4. Aumento de desempenho. 5. Redução de consumo de energia. I. Mattos, Júlio Carlos Balzano de, orient. II. Título.

CDD : 005

Elaborada por Aline Herbstrith Batista CRB: 10/1737

Banca examinadora:

.....

Prof. Dr. Júlio Carlos Balzano de Mattos (Orientador)

.....

Prof. Dr. Mateus Beck Rutzig (UFSM)

.....

Prof. Dr. Leomar Soares da Rosa Junior (UFPel)

.....

Prof. Dr. Rafael Iankowski Soares (UFPel)

Agradecimentos

Quero agradecer, em primeiro lugar a Deus que me sustenta e dá forças para travar todas as batalhas diárias de minha vida.

À minha esposa Carmen Lúcia, pelo carinho, apoio e paciência nos diversos momentos de nossas vidas. À minha sogra, Neli, por toda ajuda que me tem dado.

Aos meus pais, por terem me oportunizado e incentivado a estudar, eu não teria chegado até sem estas pessoas na minha vida.

Agradeço aos meus colegas de GACI, Eduardo Nicola e Luis Martins, que me ajudaram nas implementações de aplicações e revisões de artigos, tarefas necessárias para realizar este trabalho.

À Martha, secretaria do PPGC, pela gentileza e agilidade no atendimento das demandas.

Aos professores do PPGC, aos quais tive a honra de ser aluno. Vocês me ensinaram muito e espero passar o mesmo aos meus alunos.

Aos meus colegas do Mestrado, da primeira turma, que me ajudaram de alguma forma a chegar até aqui, pelas trocas de experiência e ajuda nos trabalhos das diversas disciplinas e pelos momentos de descontração.

À Mateus Beck, por me auxiliar no entendimento da ferramenta e validação dos dados gerados.

Por fim, agradeço ao meu orientador Julio Carlos Balzano de Mattos, pela dedicação e paciência incansáveis, seja de dia ou de noite, na UFPEL ou em sua casa. Por sempre me receber bem, por ter me aceitado como orientado, mesmo sabendo que eu não teria dedicação total ao programa, por causa de meu estágio probatório no IFSul, por me orientar sempre que precisei.

Resumo

RUZICKI, Julio C. **Estudo de Arquiteturas Reconfiguráveis para Aumento de Desempenho de Aplicações DSP. 2014. 62f.** Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Pelotas, Pelotas.

As aplicações que utilizam técnicas de processamento digital de sinais estão nas mais variadas aplicações do cotidiano das pessoas. Usualmente, para execução destas tarefas eficientemente são utilizados processadores específicos para este fim, os processadores digitais de sinais que possuem unidades que aceleram a execução de tarefas as quais os processadores de uso geral não conseguem fazer de maneira eficiente. Mesmo os PDSPs sendo tão eficientes, os atuais métodos e técnicas utilizados pela indústria para fabricação de circuitos integrados não conseguem garantir a evolução de tais dispositivos segundo o previsto pela Lei de Moore. Para garantir a evolução, novas técnicas e metodologias são necessárias, uma das áreas de pesquisa que tem aumentado e contribuído para isto é a de arquiteturas reconfiguráveis. Este trabalho apresenta a aplicação desta técnica através da acoplagem de um sistema reconfigurável em um processador MIPS32, analisando os impactos em desempenho, área e consumo energético. O trabalho também realiza a comparação entre PDSP comerciais, GPP e Sistema Reconfigurável selecionados. As comparações foram realizadas através de levantamento e análise dos dados da execução de aplicações com características DSP, executadas nas ferramentas de simulação selecionadas. Os resultados mostraram que foi obtida aceleração na execução das aplicações na ordem de 1,79 vezes com economia de energia da ordem de 45%.

Palavras-chave: Processamento Digital de Sinais, reconfiguração, reconfiguração dinâmica, aumento de desempenho, redução de consumo de energia.

Abstract

RUZICKI, Julio C. **Estudo de Arquiteturas Reconfiguráveis para Aumento de Desempenho de Aplicações DSP. 2014. 62f.** Dissertação (Mestrado em Ciência da Computação). Universidade Federal de Pelotas, Pelotas.

Applications which use techniques of Digital Signal Processing (DSP) are in the most varied applications of everyday life. Normally, to perform these tasks efficiently specialized for this purpose processors are used, the Programmable Digital Signal Processors (PDSP) that have units that speed up the execution of tasks which the general purpose processors can't do efficiently. Even the PDSP being so efficient, current methods and techniques used by the industry for the manufacture of integrated circuits can't guarantee the progress of such devices in accordance with Moore's Law. To ensure progress, new techniques and methodologies are needed, one area of research that has contributed to increased and this is reconfigurable computing. In this paper, we present the application of this technique by coupling a reconfigurable system on a MIPS32 processor, analyzing the impacts on performance, area and energetic consumption. In this paper also is realized a comparison between commercial PDSPs, GPP and Reconfigurable System selected. The comparisons were performed by collecting and analyzing the data of the execution of applications with DSP features, implemented in the selected simulation tools. The results obtained show speed up in time execution of applications on the order of 1.79 times and energy saving on order of 45%.

Keywords: Digital Signal Processing, reconfiguration, Dynamic reconfiguration, performance increase, reduction of energy consumption.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

A/D	Analógico Digital
ASIC	*Application Specific Integrated Circuit*
CAD	*Computer-Aided Design*
CLB	*Configurable Logic Block*
CPI	Ciclos por instrução
CPU	*Central Processing Unit*
D/A	Digital Analógico
DIL	*Dataflow Intermediate Language*
DMA	*Direct Memory Access*
DSP	*Digital Signal Processing*
FFT	*Fast Fourier Transform*
FIR	*Finite Impulse Response*
FPGA	*Field Programmable Gate Array*
FPL	*Field-Programmable Logic*
FPU	*Functional Processing Unit*
FU	*Functional Unit*
GPP	*General Purpose Processor*
JPEG	*Joint Photographic Experts Group*
MAC	*Multiply-accumulate*
MEMS	*Microelectromechanical System*
PC	*Personal Computer*
PDSP	*Programmable Digital Signal Processors*
PE	*Processing Elements*
RCP	*Reconfigurable Communications Processor*
RFU	*Reconfigurable Fabric Unit*
RISC	*Reduced Instruction Set Computer*
RPF	*Reconfigurable Processing Fabric*
RTR	*Run-Time Reconfiguration*
RU	*Reconfigurable Unit*
SIMD	*Single-Instruction, Multiple-Data*
SISD	*Single-instruction, single-data*

ULA Unidade Lógica Aritmética

Sumário

1. INTRODUÇÃO

O processamento digital de sinais (DSP) está presente em diversas aplicações: multimídia, telefonia celular, instrumentação industrial e biomédica, eletrodomésticos, entre outras. Em algumas dessas aplicações, o processamento digital de sinais tem papel fundamental para o sucesso, em outras aplicações assume um papel secundário, servindo como etapa intermediária entre o mundo analógico e a etapa computacional digital.

O uso de processadores digitais de sinais programáveis (PDSPs) é preferido nas aplicações devido ao ganho em desempenho de execução das tarefas (algoritmos), além de reduzir a dificuldade de realização destas, caso fossem feitas de forma analógica(LAPSLEY, BIER, *et al.*, 1997). Além disso, em sistemas que possuem requisitos temporais ou sistemas de tempo real, tais dispositivos possuem desempenho superior aos processadores de uso geral (LAPSLEY, BIER, *et al.*, 1997). Em aplicações multimídia, por exemplo, o PDSP tem papel fundamental na amostragem e filtragem dos sinais, devido a sua arquitetura especializada. Além destas características, os PDSPs também são preferidos pela sua facilidade de interfacear com outros dispositivos, como os sensores, memórias e conversores A/D e D/A.

Os sistemas embarcados são sistemas dedicados que possuem uma funcionalidade restrita para atender uma tarefa específica em sistemas maiores nos quais estão inseridos (MARWEDEL, 2006). Neste cenário, várias aplicações embarcadas necessitam de processamento digital de sinais, tendo em vista que esses sistemas estão naturalmente envolvidos com o interfaceamento com o meio externo. Os fabricantes de sistemas embarcados esperam o crescimento do mercado conforme apresentado na Figura 1, desta forma o número de aplicações que envolvem DSP também continuará crescendo.

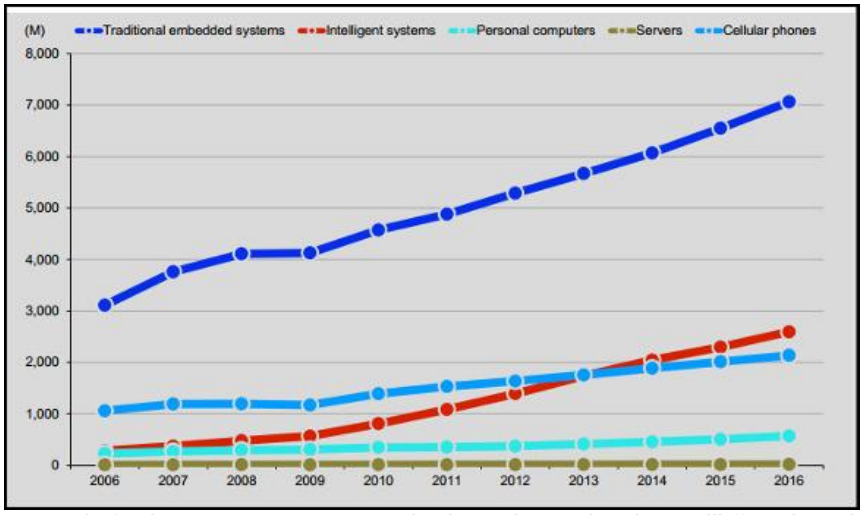


Figura 1. Previsão da AMD para o mercado de embarcados (em milhões de unidades).
Fonte: (CLARK, 2013).

A área de aplicações móveis é um outro exemplo de uso de aplicações DSP. Os aparelhos celulares há alguns anos tinham grandes dimensões, consumiam grande quantidade de energia e tinham funcionalidades básicas, como receber e efetuar chamadas e enviar mensagens de texto. Com a evolução, novas funcionalidades foram sendo agregadas e hoje, os celulares são verdadeiros sistemas computacionais com grande capacidade de processamento utilizando sistemas operacionais com diversos dispositivos integrados, como GPS, bússola, acelerômetros, magnetômetros, telas sensíveis a toque, etc. O mercado de Smartphones forneceu, em 2012, 722 milhões de unidades e aproximadamente 987 milhões em 2013, segundo (CLARKE, 2013)

Como dito anteriormente, o processamento digital de sinais (DSP) está cada vez mais presente como em áreas de instrumentação biomédica. A Figura 2 aponta o crescimento esperado para os próximos anos na aplicação de MEMS em biomédica.

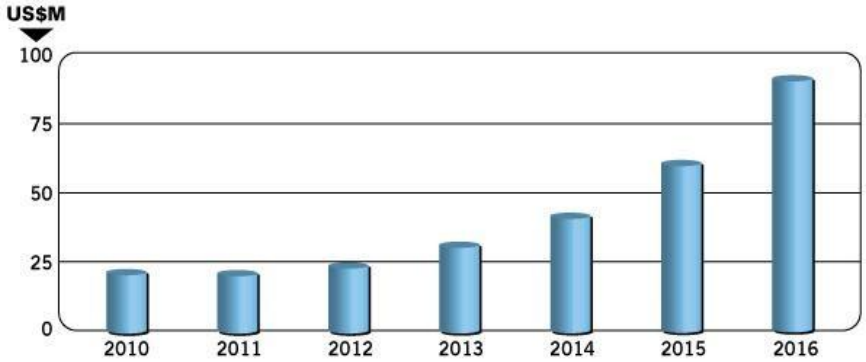


Figura 2. Previsão para o mercado de MEMS em biomédica.

Fonte:(Solid State Technology, 2013)

A presença dos PDSPs no mercado foi garantida pelo aumento na densidade dos processadores, fornecendo suporte para o uso de DSP na execução de tarefas envolvendo multimídia, instrumentação e comunicação de dados, através da adição de unidades especializadas. Este aumento, também garantiu a evolução dos processadores como previsto na Lei de Moore. Contudo, a evolução fornecida por esta abordagem encontrou o seu limite devido aos métodos e materiais utilizados.

Mesmo assim, a demanda por aplicações mais complexas continua aumentando e os fabricantes procuram novos materiais e métodos para garantir a evolução dos dispositivos que realizam DSP. Aplicações como, Android em Smartphones e tablets utilizam as características de PDSPs para aumentar a autonomia de baterias, velocidade de execução e diminuição do tamanho dos dispositivos. Porém, estes não são a única possibilidade para executarmos DSP.

Para executar aplicações DSP, três tipos de dispositivos são normalmente utilizados: GPP (*General Purpose Processor)*, ASIC *(Application Specific Integrated Circuit*) e PDSP (YOONJIN KIM, 2009). Os GPPs oferecem a flexibilidade da programação e permitem atender a uma grande gama de aplicações. Porém, oferecem um bom desempenho na execução das aplicações com um alto consumo energético, pois sua arquitetura não explora adequadamente a complexidade, a computação intensiva e o paralelismo de dados exigidos pelas das aplicações DSP (YOONJIN KIM, 2009). Os ASICs fornecem alto desempenho em consumo energético e tempo de execução, uma vez que são específicos para cada tarefa e por este motivo não apresentam a flexibilidade dos GPPs e o custo do projeto é elevado, sendo viável apenas em altos volumes.

Por outro lado, os PDSPs oferecem a flexibilidade da programação oferecida pelos GPPs, garantindo a cobertura de uma grande gama de aplicações. Os PDSPs também fornecem uma ótima relação custo/benefício em termos de consumo energético e desempenho sem os custos de projeto de um ASIC. Além destes motivos, os PDSPs são preferidos para executar aplicações DSP, pois suas arquiteturas especializadas permitem explorar o paralelismo e a computação intensiva, características presentes em tais aplicações. Em sua arquitetura, os PDSPs possuem unidades de execução adicionais, especializadas, que permitem executar tarefas como: FIR, FFT, MAC, DFT e acesso à memória (LAPSLEY, BIER, *et al.*, 1997), de forma a acelerar a execução das aplicações.

Além da demanda crescente, há a necessidade da evolução de tais dispositivos que esbarram nas limitações citadas anteriormente. Enquanto a indústria não adota novos materiais, em substituição aos atuais, o caminho é a pesquisa por novas metodologias para garantir a evolução. Uma das áreas que vem crescendo em pesquisa é a área de Computação Reconfigurável. Esta área propõe aumentar o desempenho dos processadores sem aumentar a taxa do relógio, acoplando unidades funcionais capazes de executar porções de código de programa diretamente em hardware, evitando o fluxo de dados do processador, acelerando a execução das aplicações. Além disto, oferece uma arquitetura adaptável, dado o aumento de aplicações com características distintas que podem degradar o desempenho dos processadores comerciais, como ocorre nos Smartphones, por exemplo (RANA, SANTAMBROGIO e SCIUTO, 2007).

1.1 Objetivos e resultados esperados

Como apresentado na Seção anterior, a evolução dos PDSPs é necessária para atender a demanda por aplicações mais complexas em um mercado em crescimento. Porém, a evolução está limitada pelos métodos e materiais usados.

O objetivo deste trabalho é demonstrar que através do uso da técnica conhecida como Computação Reconfigurável incorporada a um processador de propósito geral pode-se acelerar a execução de algoritmos de processamento digitais de sinais. Também é objetivo deste trabalho, realizar a comparação entre processadores DSP comerciais, GPP e Sistema Reconfigurável selecionados. As comparações foram realizadas através de levantamento e análise dos dados da

execução de aplicações com características DSP, executadas nas ferramentas de simulação selecionadas. De posse dos resultados foi possível verificar a aplicação da técnica de Computação Reconfigurável em um GPP.

1.2 Organização do trabalho

Este trabalho está organizado em seis capítulos. O capítulo 2 realiza uma revisão dos conceitos de processamento digital de sinais apresentando suas características e também apresenta as principais características dos processadores DSP, detalhando as principais características que os tornam tão eficientes. No terceiro capítulo é feita uma introdução à Computação Reconfigurável, onde são introduzidos ao leitor diversos conceitos relacionados ao assunto, com exemplos/revisão bibliográfica de arquiteturas propostas nesta área de pesquisa.

O capítulo 4 e 5 apresentam, respectivamente, a metodologia e ferramentas utilizadas no trabalho, resultados obtidos da realização deste trabalho. Por fim, o capítulo 6 apresenta as conclusões do trabalho e a perspectiva de trabalhos futuros.

2. Processamento digital de sinais

Segundo LAPSLEY (LAPSLEY, BIER, *et al.*, 1997), um sistema de processamento digital de sinais é qualquer sistema eletrônico que faz uso de processamento digital de sinais. Dentro disto também está definido como aplicação de processamento de sinais, a aplicação de operações matemáticas para representar sinais digitalmente. Sinais são digitalmente representados como sequências de amostras, frequentemente obtidas de sinais físicos (sinais de áudio, por exemplo) através do uso de transdutores (tal como microfones) e conversores analógico para digital. Após o processamento matemático, sinais digitais podem ser convertidos de volta para sinais físicos através de um conversor digital para analógico.

Em alguns sistemas, o uso de DSP é central para o funcionamento destes. Por exemplo, modems e telefones celulares dependem muito fortemente da tecnologia DSP. Em outros produtos, no entanto, esta dependência é menor, mas oferecem vantagens competitivas em termos de características, desempenho e custo. Por exemplo, fabricantes de dispositivos eletrônicos de consumo, como amplificadores de áudio, empregam a tecnologia DSP para fornecer novas características a seus produtos.

O processamento digital de sinais apresenta diversas vantagens sobre sistemas de processamento analógico. A mais significativa destas é de que sistemas DSP são capazes de realizar tarefas de forma mais econômica do que sistemas analógicos, onde poderiam ser feitas de maneira muito difícil, exigindo muitos recursos. Exemplos de tais aplicações incluem sintetizadores de fala, reconhecimento de fala, e modems de alta velocidade que utilizam codificação de correção de erros (*error-correction coding*). Todas estas tarefas envolvem uma combinação de processamento de sinais e controle os quais são extremamente difíceis de serem implementados usando técnicas analógicas.

Sistemas DSP também oferecem imunidade aos agentes do ambiente como, ruídos elétricos, variações de temperatura e choques mecânicos, estes sistemas também não apresentam problemas com desvio no valor dos componentes, duas características que afetam os circuitos analógicos. Além destas, tais sistemas oferecem outra vantagem em relação aos circuitos analógicos: previsibilidade e o

comportamento repetitivo. Como a saída de um sistema DSP não varia por causa dos fatores ambientais ou da variação dos componentes, é possível projetar sistemas com resposta exata e que não variam. Alguns sistemas DSP podem também ter outras duas vantagens sobre sistemas analógicos (LAPSLEY, BIER, *et al.*, 1997):

1. Programabilidade: se um sistema DSP é baseado em um processador programável, este pode ser reprogramado em campo para realizar outras tarefas. Em contraste, sistemas analógicos requerem componentes diferentes para realizar tarefas diferentes;
2. Tamanho: o tamanho de componentes analógicos varia de acordo com seus valores: por exemplo, um capacitor de 100 µF usado em um filtro analógico é fisicamente maior do que um capacitor de 10 nF, usado em outro filtro. Em contraste, implementações DSP de ambos os filtros tem o mesmo tamanho, pois usam o mesmo hardware, diferindo apenas nos coeficientes dos filtros, podendo, inclusive, ter um tamanho menor do que as duas implementações analógicas.

Estas vantagens, aliadas a alta densidade dos circuitos integrados digitais, fazem do DSP a melhor opção de escolha para processamento de sinais.

2.1 Características de sistemas DSP

Além das características citadas na seção anterior, LAPSLEY (LAPSLEY, BIER, *et al.*, 1997) define que existem características comuns para todos os sistemas DSP: algoritmos, taxa de amostragem, taxa de relógio (*clock rate*) e tipos de aritméticas.

Sistemas DSP são frequentemente caracterizados pelos algoritmos utilizados. Dos vários tipos de aplicações DSP, podemos citar: codificação e decodificação de dados, encriptação e desencriptação de voz, compressão e descompressão de imagem, algoritmos para modems, cancelamento de ruído, equalização de áudio, emulação de ambiente acústico, mixagem e edição de áudio, síntese de som, entre outros.

O algoritmo especifica as operações aritméticas a serem realizadas, mas não especifica como tal aritmética será implementada. Esta pode ser implementada em programa em um GPP, ou com PDSP, ou ainda em um ASIC. A escolha de uma

tecnologia de implementação é determinada em parte pelos requisitos de desempenho e precisão aritmética.

A taxa de amostragem é a característica chave de um sistema DSP, pois determina a taxa na qual as amostras são consumidas, processadas, ou produzidas. Combinada com a complexidade dos algoritmos, esta taxa determina o requisito de desempenho da tecnologia de implementação. Um exemplo familiar é o *CD player,* o qual produz amostras a uma taxa de 44,1 kHz em dois canais. Sistemas DSP podem usar mais de uma taxa de amostragem; tais sistemas são chamados de sistemas DSP de múltiplas taxas. Um exemplo é um conversor de CD de taxa de amostragem de 44,1 kHz para uma fita de áudio digital (DAT) com taxa de 48 kHz.

Como pode ser observado na Figura 3, a faixa de taxas de amostragem é enorme. Além disto, a posição de algumas aplicações no que diz respeito à complexidade dos algoritmos e a taxa de amostragem é uma informação importante. Como pode ser visto, algoritmos DSP usados com taxas de amostragem maiores tendem a serem mais simples do que os usados a taxas de amostragem menores.

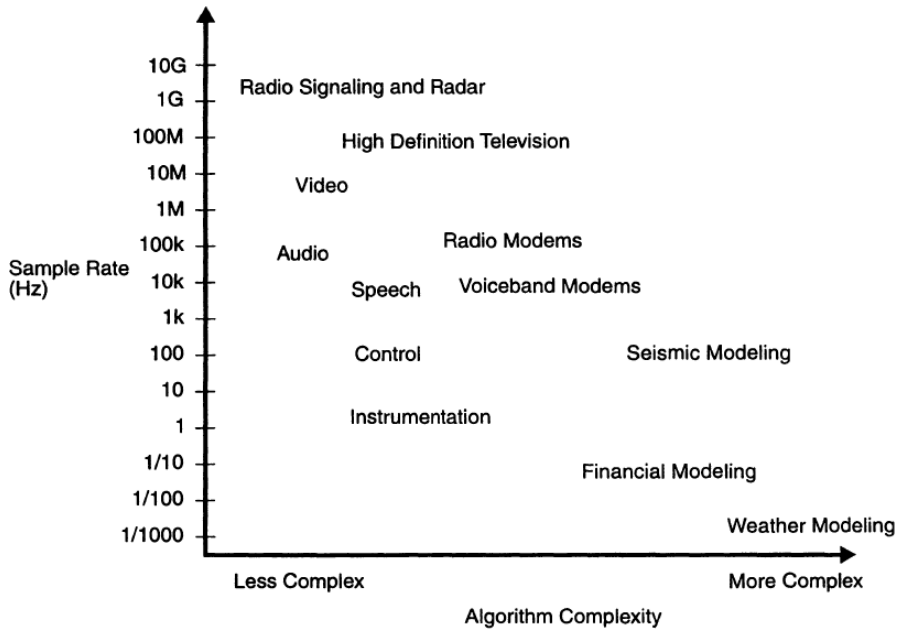


Figura 3. Taxas de amostragem e complexidades de algoritmos para uma variedade de classes de aplicações de DSP. Fonte: (Lapsley,1997, p.5).

Sistemas digitais eletrônicos são frequentemente caracterizados pela taxa do relógio. A taxa de relógio usualmente refere à taxa na qual o sistema realiza o seu ciclo básico de execução. Na produção em massa, produtos comerciais, com taxas

de até 100 MHz são comuns, taxas maiores são destinadas a produtos de alto desempenho.

Para sistemas DSP, a relação entre a taxa do relógio e a taxa de amostragem é uma das características mais importantes para determinar como o sistema será implementado. A relação entre a taxa de relógio e a taxa de amostragem parcialmente determina o hardware necessário para implementar um algoritmo com uma dada complexidade em tempo real. Se a proporção entre a taxa de amostragem com a taxa do relógio aumenta, o mesmo acontece com a quantidade e complexidade do hardware exigido para implementar o algoritmo.

As operações numéricas, como adição e multiplicação, são o coração de sistemas e algoritmos DSP. Como resultado, as representações numéricas e tipos de aritméticas usadas podem ter uma profunda influência no comportamento e desempenho de um sistema DSP. A mais importante escolha para o projetista é entre aritmética de ponto fixo ou ponto flutuante. Aritmética de ponto fixo representa números dentro de uma faixa fixa (por exemplo, -1,0 à +1,0) com um número finito de bits de precisão, chamada de *largura de palavra*. Por exemplo, um número de ponto fixo, de 8 bits, fornece uma resolução de 1/256 sobre a faixa na qual o número pode variar. Números que se encontram fora desta faixa, não podem ser representados; operações aritméticas que poderiam resultar em um número fora desta faixa ou saturam, isto é, são limitados no maior valor positivo ou negativo representável, ou os bits gerados em excesso são descartados.

Aritmética de ponto flutuante expande enormemente a faixa de representação de valores. A aritmética de ponto flutuante representa cada número em duas partes: a mantissa e um expoente. A mantissa é, em efeito, forçado para situar-se entre -1,0 e +1,0, enquanto o expoente mantém o controle do valor pelo qual a mantissa deve ser dimensionada (em termos de potências de dois), a fim de criar o valor real representado. Ou seja: $valor = manstissa \times 2^{\text{expo}ente}$.

A aritmética de ponto flutuante fornece uma faixa dinâmica muito maior (ou seja, a proporção entre o maior e o menor valor que podem ser representados) do que a aritmética de ponto fixo. Isto é possível porque ela reduz a probabilidade de estouro e a necessidade de dimensionamento, simplificando consideravelmente o projeto de algoritmos e programas. Porém, aritmética de ponto flutuante é geralmente mais lenta e mais cara em termos de recursos de hardware, do que a

aritmética de ponto fixo, além de ser mais complicada de ser implementada em hardware do que a aritmética de ponto fixo.

Ao longo dos anos diversos trabalhos foram propostos com a finalidade de melhorar o desempenho das características anteriormente citadas. Jagadeesh (JAGADEESH, RAVI e MALLIKARJUN, 2013) propõe uma unidade MAC de 64 bits de alto desempenho para aplicações DSP. O circuito alcançou uma frequência de operação de 217 Mhz e 177,732 mW de potência dissipada na tecnologia de 0,18 um. Ferdous (FERDOUS, 2012) propõe um PDSP de 32 bits contendo uma unidade MAC com tempo de execução de um ciclo, baseado em uma arquitetura Hazard otimizada com pipeline de 2 estágios. Os resultados obtidos mostram que processador proposto alcança uma vazão de dados (throughput) de 12,06 MB/s contra 8 MB/s se comparado com o MUN DSP-2000, que é um processador similar com um pipeline de 5 estágios.

Nas seções seguintes serão mostrados alguns exemplos de PDSPs comerciais e suas características.

2.2 Caracterísiticas dos processadores DSP

Como afirmado na Seção anterior, normalmente utilizamos três tipos de dispositivos para realizar o processamento digital de sinais: GPPs, ASICs ou PDSPs. A Tabela 1 abaixo resume as vantagens e desvantagens de cada abordagem.

Tabela 1. Resumo das abordagens GPP, ASIC e PDSP.

Dispositivo	Vantagem	Desvantagem
GPP	Flexíveis. Existem várias ferramentas de programação no mercado.	Baixo desempenho em consumo. Não conseguem explorar o paralelismo das aplicações DSP.
ASIC	Alto desempenho em consumo e tempo de execução.	Não são flexíveis. Tempo e custo de projeto elevados. Economicamente viáveis em altas quantidades.
PDSP	Oferecem a flexibilidade dos GPP. Apresentam melhor desempenho do que	Limitações em desempenho em função das

	os GPP por causa de sua arquitetura especializada (MAC, FPU, etc...). Não apresentam os custos proibitivos dos ASIC. Exploram o paralelismo em DSP.	características distintas das aplicações.

Os GPPs são baratos, flexíveis e existem diversas ferramentas comercias para programá-los. Por outro lado, oferecem uma baixa relação custo/benefício, com um bom desempenho, porém um alto consumo energético na execução de aplicações, principalmente para DSP, pois sua arquitetura não explora adequadamente a computação intensiva e o paralelismo de dados exigidos pelas das aplicações DSP (YOONJIN KIM, 2009). Já o ASICs, oferecem baixo consumo energético e um alto desempenho em termos de processamento, porém, só são economicamente viáveis para grandes volumes e os processos de projeto e fabricação são extremamente caros. Como alternativa para estas opções, temos os PDSPs, que oferecem a flexibilidade dos GPPS e o alto desempenho e baixo consumo dos ASICs, sem os custos proibitivos. Os PDSPs são preferidos para executar aplicações DSP, porque ao contrário dos GPP, sua arquitetura explora adequadamente a complexidade, a computação intensiva e o paralelismo de dados exigidos pelas das aplicações DSP.

Uma das características mais citadas dos PDSPs está na habilidade de realizar operações de multiplica e acumula de forma rápida, chamada de MAC, normalmente em um único ciclo de relógio. Tal habilidade é extremamente útil em algoritmos que envolvem computação de produtos vetoriais, tais como, filtros digitais, correlação, FFT, FIR, IDFT, DCT, etc. Para alcançar este objetivo, PDSPs possuem unidades MAC presentes dentro de suas unidades de processamento aritméticas.

Além desta característica, tais processadores fornecem bits extras em seus registradores acumuladores, para evitar o estouro aritmético que pode ocorrer da realização repetitiva de operações do tipo MAC.

A segunda característica que os PDSPs compartilham é a habilidade de completar diversos acessos à memória, em um único ciclo de instrução. Isto permite ao processador buscar uma instrução enquanto busca os operandos ou armazenar o resultado da instrução anterior na memória. Uma grande largura de banda entre o

processador e a memória é essencial para um bom desempenho se repetitivas operações de dados são exigidos em um algoritmo, o que é comum em aplicações DSP.

Em diversos processadores, acessos múltiplos à memória em um único ciclo estão sujeitos a restrições. Tipicamente, algumas destas posições de memória podem residir no processador, como a última posição acessada, por exemplo, e também instruções que fazem múltiplos acessos à memória podem tomar prioridade sobre instruções menos importantes. Para suportar acessos de múltiplas posições de memória simultaneamente, PDSPs fornecem múltiplos barramentos, memórias de múltiplas portas, e em alguns casos, múltiplos bancos de memória independentes. As estruturas de memória de PDSPs são a grande diferença para os GPPs.

Para permitir processamento aritmético à um desempenho máximo e permitir a especificação de múltiplos operandos em uma pequena palavra de instrução, PDSPs incorporam unidades de geração de endereços dedicadas. Uma vez que o endereço do registrador apropriado tenha sido configurado, a unidade de geração do endereço opera em segundo plano, formando o endereço exigido para acessar o registrador em paralelo com a execução das instruções aritméticas.

Unidades de geração de endereço tipicamente suportam uma seleção do modo de endereçamento adaptado para aplicações DSP. O mais comum destes, é o endereçamento indireto de registradores com pós-incremento, o qual é usado em situações onde uma computação repetitiva é realizada em uma série de dados armazenados sequencialmente na memória.

Devido ao fato de que algoritmos DSP envolvem a realização de computações repetitivas, muitos PDSPs fornecem suporte especial para a realização eficiente de laços. Frequentemente, uma instrução especial para realizar um *for-next*, por exemplo, é fornecida para evitar a perda de ciclos de instrução nas tarefas de atualização e teste de contadores de laço, ou para saltar novamente para o topo do laço. Alguns PDSPs fornecem outras características de controle de execução para melhorar o desempenho, tais como, rápido chaveamento de contexto com baixa latência, interrupções com baixa sobrecarga para realizar rápidas operações de entrada e saída.

Para permitir baixo custo, alto desempenho de entradas e saídas, muitos PDSPs incorporam uma ou mais interfaces paralelas ou seriais, e mecanismos

especializados de gerenciamento tais como, DMA. As interfaces de periféricos de processadores DSP são frequentemente projetadas para interfacear diretamente dispositivos periféricos comuns tais como, conversores A/D e conversores D/A. Como as técnicas de fabricação de dispositivos tem melhorado, em termos de densidade e flexibilidade, PDSPs não só incorporam interface para os periféricos como também incorporam dispositivos periféricos completos dentro de seus chips. Exemplos destes são os PDSPs projetados para aplicações em telefones celulares, os quais incorporam conversores A/D e D/A dentro de seus chips.

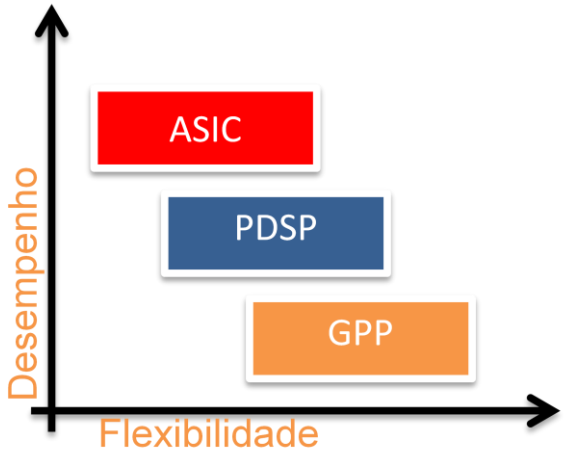


Figura 4. Relação entre desempenho e flexibilidade entre GPP, ASIC e PDSP.

Portanto, a Figura 4 mostra a relação entre desempenho e flexibilidade para as três abordagens utilizadas na realização de aplicações DSP.

2.3 Exemplos de processadores DSP

Para exemplificar as características citadas na seção anterior, são apresentados alguns exemplos de processadores que possuem tais características. Os processadores DSP modernos atingem um alto desempenho através de aspectos tecnológicos e também pelos avanços/características arquiteturais apresentadas na seção anterior.

O mercado de fabricantes de processadores DSP é um tanto restrito atualmente e é composto pelas seguintes empresas: Analog Devices, Freescale (antiga Motorola), Texas Instruments, Renesas e Microchip. Contudo, a maior parte do mercado é dominado por três empresas: Analog Devices, Freescale e Texas Instruments. A Tabela 2 apresenta as principais famílias dos principais fabricantes e

suas características (ANALOG, 2013a) (INSTRUMENTS, 2014c) (FREESCALE, 2014).

Tabela 2. Principais famílias dos fabricantes Analog Devices e Texas Instruments.

Fabricante	Família	Aplicação
Texas Instruments	TMS320C2x	Desempenho médio
Texas Instruments	TMS320C5x	Eficiente em potência
Texas Instruments	TMS320C6x	Alto desempenho
Analog Devices	SHARC	Alto desempenho, eficiente em potência
Analog Devices	Tiger SHARC	Alto desempenho (multiprocessadores)
Analog Devices	Blackfin	Alto desempenho e baixa potência
Freescale	DSP56K	PDSP de uso geral de baixo consumo energético

Na sequência são apresentados alguns processadores. Estes processadores são: o ADSP-BF504, da família Blackfin, o ADSP-21477, da família SHARC e o DSP56311, da família DSP56K.

Na Figura 5, vemos o diagrama de blocos do processador ADSP-BF504 (ANALOG, 2013c), onde podemos observar a existência de memórias cache L1, para dados e instruções, separadas e com barramentos independentes para acesso do processador e controlador DMA. Diversos periféricos: *timers*, contadores, portas de comunicação (SPI, UART, CAN, etc), portas de entrada e saída (PORT F, G e H), conversor A/D, etc. A família de processadores Blackfin (ANALOG, 2013c) foi desenvolvida para aplicações de baixo custo que exigem altas taxas de processamento digital de sinais, como vídeo e instrumentação. Dentre muitas de suas características, destacam-se as seguintes: instrução MAC (*multiply-accumulate*) de 16 bits (possibilidade de executar até 800 milhões de MACs por segundo) e duas ULAs (Unidade Lógica e Aritmética) de 40 bits.

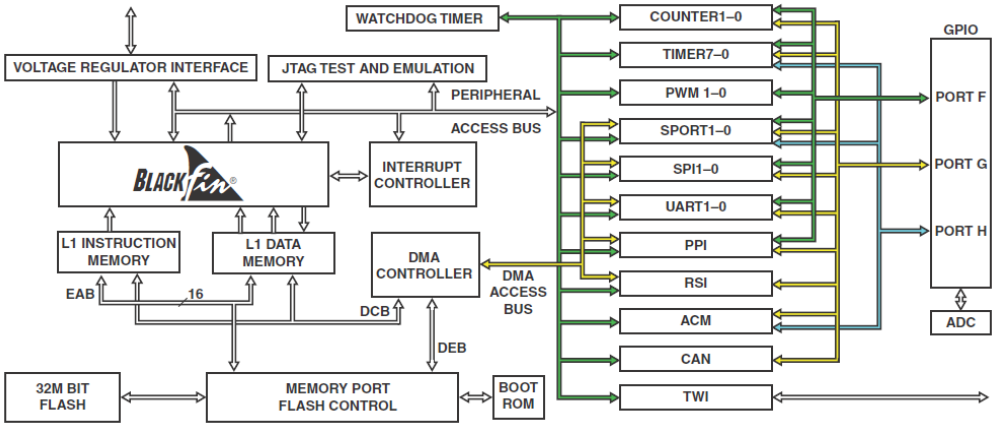


Figura 5. Diagrama de blocos do processador ADSP-BF504. Fonte: (ANALOG, 2013c).

Na Figura 6, temos o diagrama de blocos do processador ADSP-21477 (ANALOG, 2013b), de 32 bits, baseado na arquitetura Super Harvard que é uma otimização da arquitetura Harvard tradicional, está otimização é obtida com a adição de um processador para controlar o acesso aos barramentos especializados. Além destas, podemos citar como características comuns de sua arquitetura: unidade aritmética de 32/40 bits de ponto flutuante; multiplicadores de 32 bits, ponto fixo; todas as instruções são executadas em um ciclo. Como no caso do processador Blackfin, podemos observar a diversidade de barramentos para acelerar o acesso do processador aos diversos recursos disponíveis, sem atrasar a execução do programa. Podemos observar ainda, a existência de blocos para execução de FIR, FFT e IIR. Este processador também permite dois modos de execução de instruções: SISD e SIMD.

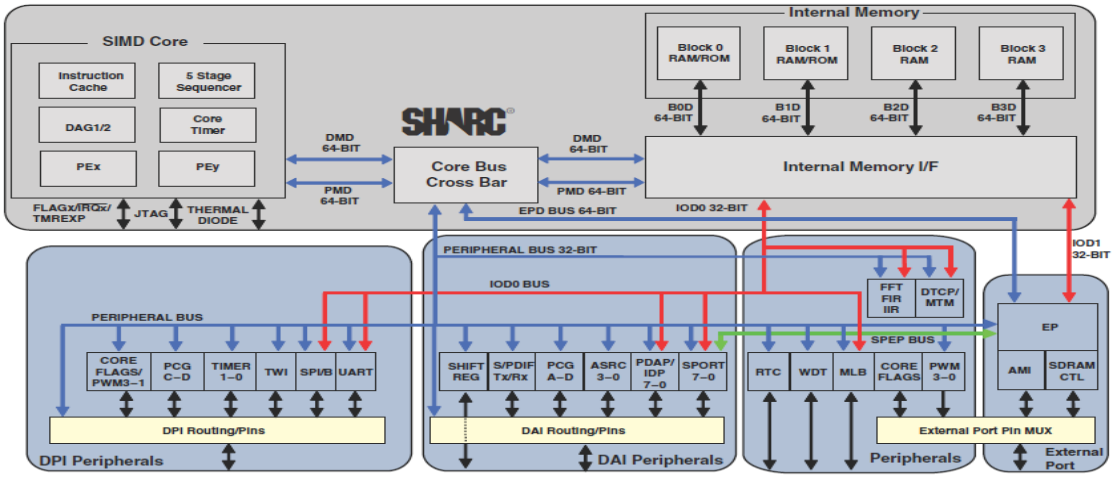


Figura 6. Diagrama de blocos do processador ADSP-21477. Fonte: (ANALOG, 2013b).

Além dos processadores acima, citamos o DSP56311 da família DSP56K (FREESCALE, 2014), cujo diagrama de blocos pode ser visto na Figura 7. Este processador de 24 bits foi desenvolvido para aplicações que requerem grandes quantidades de memória, como comunicações em rede sem fio. Em sua arquitetura há um coprocessador de filtragem otimizado chamado de EFCOP (Enhanced Filter Coprocessor) que executa operações de filtragem em paralelo com o núcleo podendo acelerar aplicações, como correlação e algoritmos baseados em convoluções.

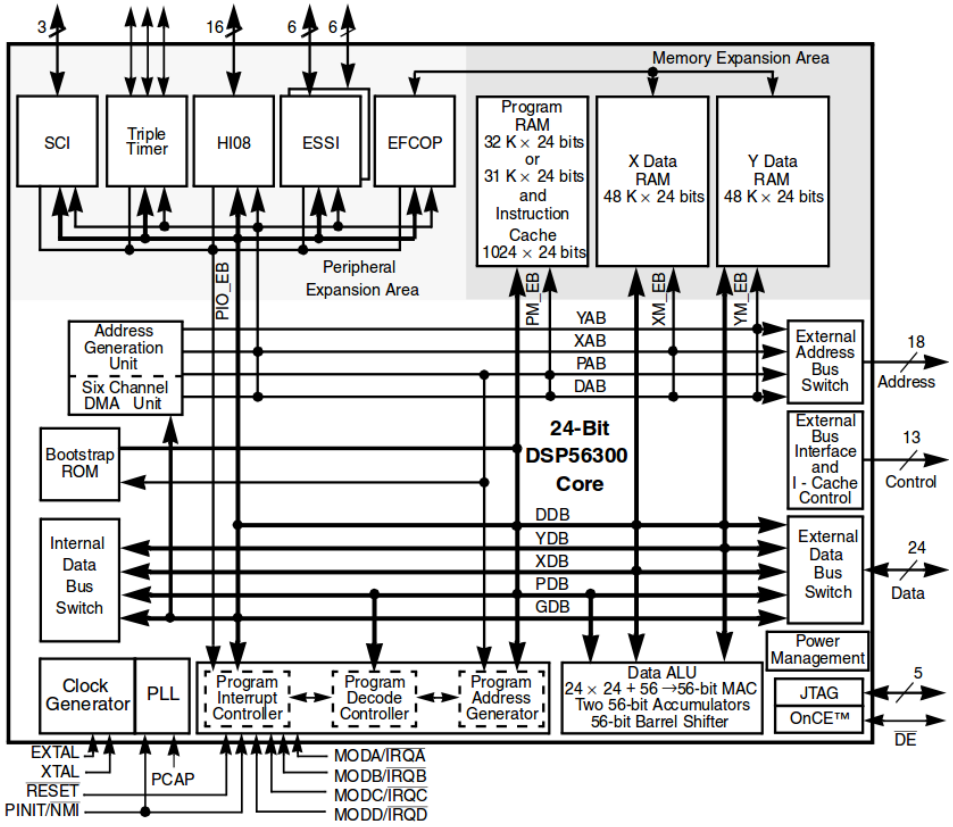


Figura 7. Diagrama de blocos do processador DSP56311. Fonte: (FREESCALE, 2014).

2.4 Síntese do capítulo

Neste capítulo, foram apresentadas as características mais relevantes de DSP, destacando principalmente as arquiteturas especializadas necessárias para o uso desta técnica. Foi apresentada também que tal arquitetura está presente nos PDSPs, que de maneira diferente dos GPPs e ASICs, oferecem o melhor custo-benefício na execução de aplicações que utilizam DSP.

O capítulo seguinte descreve uma introdução a computação reconfigurável acompanhado de uma revisão bibliográfica e sua aplicação em DSP.

3. Arquiteturas reconfiguráveis

Por vários anos o campo de pesquisa conhecido como Computação Reconfigurável ou Arquiteturas Reconfiguráveis vem contribuindo para acelerar a execução das aplicações mantendo as sobrecargas de área e consumo tão baixas quanto possíveis. Os trabalhos citados a seguir refletem o esforço da comunidade científica em contribuir com esta área de pesquisa, explorando os diferentes aspectos envolvidos na execução e aceleração das aplicações.

Nesta seção são apresentados os diversos conceitos acerca de Arquiteturas Reconfiguráveis, necessários para melhor compreensão deste trabalho. A organização dos conceitos está baseada conforme (DEHON, 2007), como apresentando na Figura 8. Ao longo do texto são apresentadas as principais características e os trabalhos enquadrados nestas.

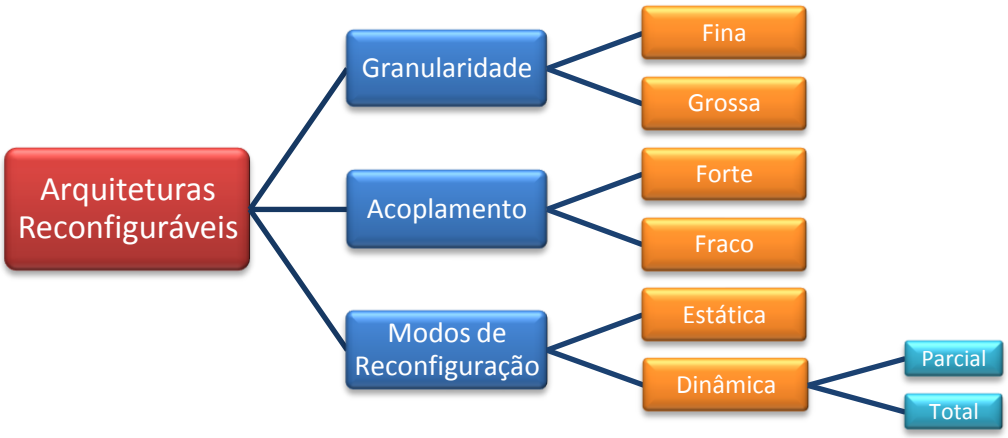


Figura 8. Classificação das Arquiteturas Reconfiguráveis.

3.1 Granularidade

Uma das definições sobre arquiteturas reconfiguráveis é quanto ao tipo de RPF (*Reconfigurable Processing Fabric*) que será utilizada. Diferentes sistemas têm diferentes tipos de granularidade. Estas variam de estruturas de granularidade fina (*fine-grained*), que manipulam dados em nível de *bit,* similar aos FPGA's comerciais,

até estruturas de granularidade grossa (*coarse-grained*), que manipulam grupos de *bits* através de unidades funcionais complexas tal como ULA’s e multiplicadores.

3.1.1 Granularidade fina

Tais estruturas oferecem ao projetista o benefício de implementar tarefas de manipulação de bits sem desperdiçar recursos reconfiguráveis, ou seja, altamente otimizadas. Contudo, para quantidades grandes e complexas de cálculo, numerosas PE's (*Processing Elements*) com granulação fina são necessárias para realizar uma computação básica. Isso resulta em taxas de relógio mais lentas do que as possíveis se os cálculos fossem mapeados para estruturas de granularidade grossa(DEHON, 2007, p. 35).

Além disso, a granularidade fina também limita a quantidade de estruturas que podem ser armazenadas na unidade reconfigurável por causa dos limites de capacidade de armazenamento.

3.1.2 Granularidade grossa

Estruturas de granularidade grossa visam explorar paralelismo em nível de operação, através da conexão de PE's adequadamente. Esta abordagem pode ser melhor para aplicações de computação intensiva (CALLAHAN;HAUSER;WAWRZYNEK, 2000). Arquiteturas deste tipo se utilizam de unidades especialmente projetadas para determinado fim, como por exemplo, unidades de ponto flutuante para aceleração de DSP. Kim (YOONJIN KIM, 2009) classifica as estruturas de granularidade grossa como, matrizes reconfiguráveis lineares ou baseadas em malhas (*mesh-based*). São consideradas matrizes reconfiguráveis baseadas em malhas as que apresentam distribuições regulares de PEs, organizados em matrizes retangulares 2-D, oferecendo grandes quantidades de recursos de comunicação para tornar o paralelismo eficiente.

No caso de matrizes reconfiguráveis lineares temos apenas uma linha de PEs, normalmente utilizados em sistemas com *pipeline* e reconfiguração dinâmica.

A Figura 9 ilustra um conjunto de unidades reconfiguráveis conectadas através de uma barreira de passagem ou *Crossbar*. Na parte inferior da figura temos apenas uma linha de PEs, classificando tal arquitetura como linear PipeRench (Y.

CHOU, 2000) e RaPID (C. EBELING, 1997) são exemplos de estruturas lineares. RaPID oferece diversos recursos computacionais como, ULAs, RAM e multiplicadores, porém irregularmente distribuídos em uma dimensão e configurados estaticamente. Por outro lado, PipeRench conta com reconfiguração dinâmica, permitindo a reconfiguração de PEs a cada ciclo de execução. A arquitetura do PipeRench, vista na Figura 10, permite a reconfiguração de um estágio enquanto outros estágios estão realizando computações concorrentemente.

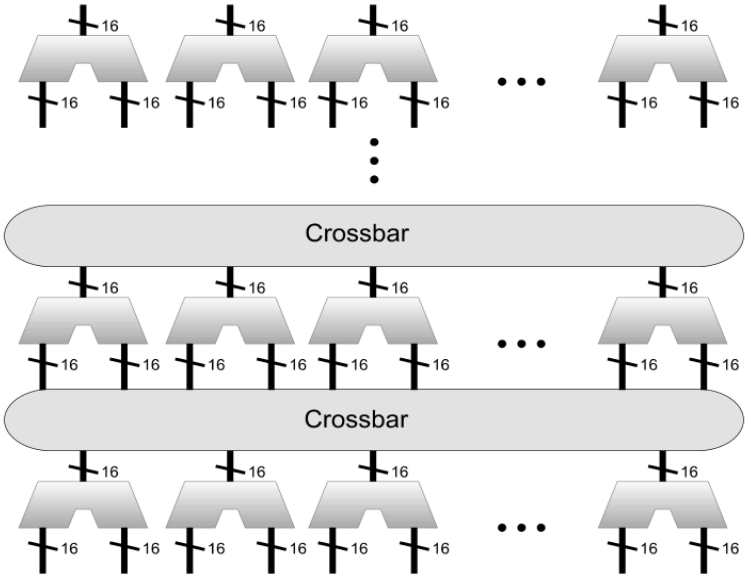


Figura 9. Um exemplo de unidades de reconfiguração grossa.
Fonte:(BECK e CARRO, 2010, p. 28).

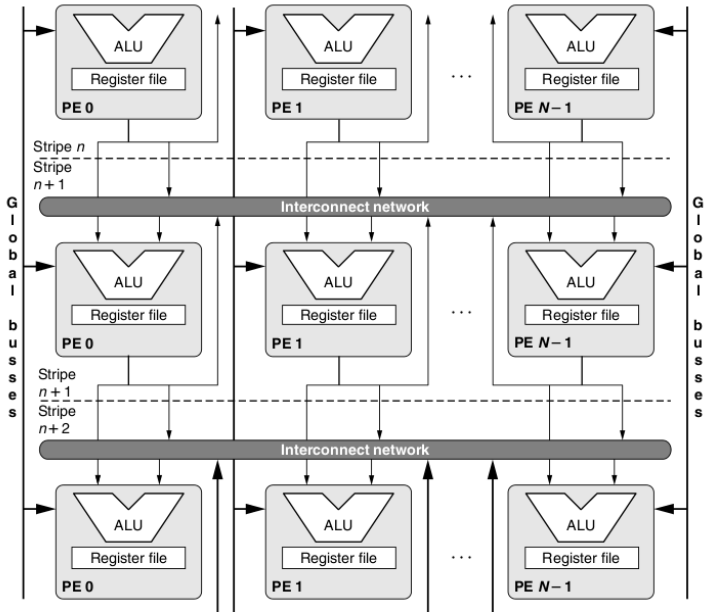
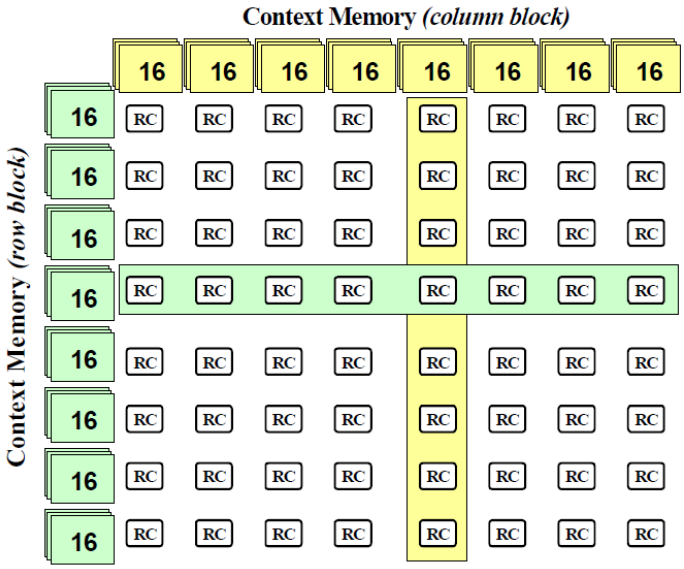
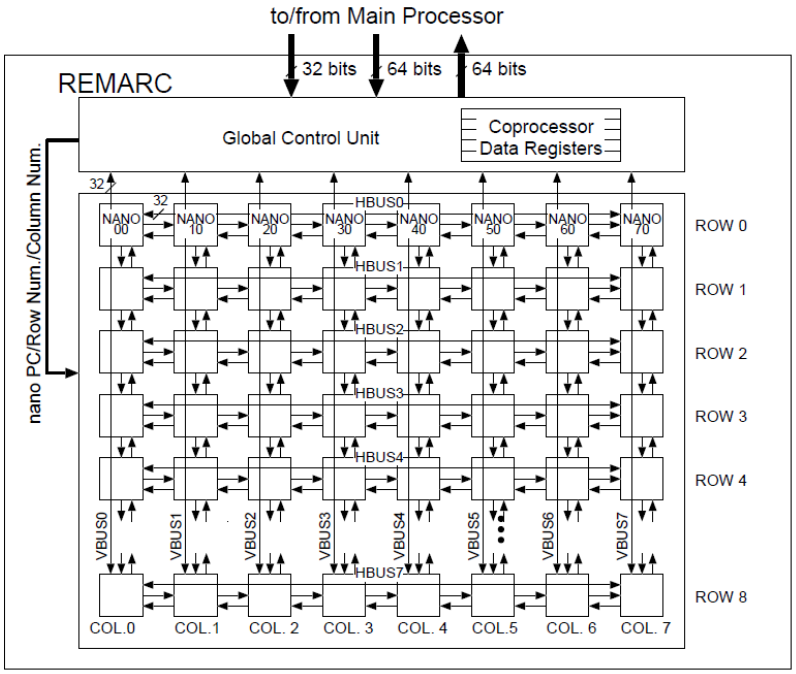


Figura 10. RPF do PipeRench.
Fonte:(DEHON, 2007, p. 33)

Caso a arquitetura permita termos mais de uma linha de PEs, então esta é chamada de baseada em malha. Isto ocorre nas arquiteturas do MorphoSys (H. SINGH, 2000), vista na Figura 11(a) e no REMARC (OLUKOTUN, 1998), vista na Figura 11(b). Como visto na Figura 11, as duas propostas apresentam RPFs retangulares 2-D.



(a)



(b)

Figura 11. Arquitetura do MorphoSys em (a) e do REMARC em (b).
Fonte:(H. SINGH, 2000, p. 2),(OLUKOTUN, 1998, p. 3)

Como apresentado no início desta seção, a granularidade afeta no tamanho da memória de contexto e no tempo de reconfiguração. Na granulação fina, é necessário muito mais informação para descrever a instrução de reconfiguração. Já na granulação grossa a descrição é mais compacta, mas por outro lado podemos limitar o desempenho da operação por causa do nível do recurso disponível.

3.2 Acoplamento

Enquanto a RPF em um sistema computacional reconfigurável dita os recursos lógicos programáveis, um sistema computacional reconfigurável tipicamente tem também um microprocessador, memória, e possivelmente outras estruturas. Outra definição das características das arquiteturas reconfiguráveis é quanto à integração da unidade reconfigurável.

Como pode ser visto na Figura 12, existem diversas maneiras de integrar uma RPF dentro da hierarquia de memória de um sistema computacional. Os diferentes componentes de memória do sistema são desenhados como retângulos preenchidos, onde os de cor escurecida indicam um acoplamento mais forte do componente de memória com o processador.

Os tipos de integração da RPF para estes sistemas computacionais são ilustrados como retângulos arredondados, onde a cor escurecida indica um acoplamento mais forte da RPF com o processador.

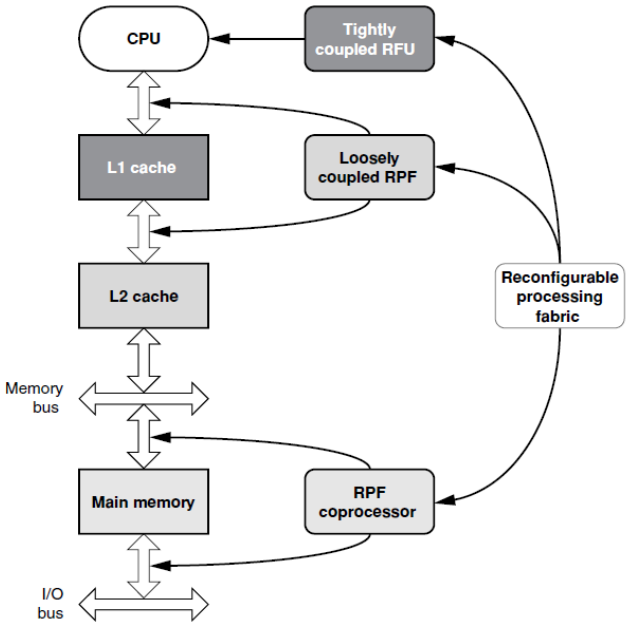


Figura 12. Estrutura do PipeRench: interconexão e PE's. Fonte:(DEHON, 2007, p. 35)

Alguns sistemas têm a RPF como um processador separado no sistema, porém, muitas aplicações requerem um microprocessador em algum lugar para gerenciar o complexo controle. De fato, algumas plataformas reconfiguráveis separadas são atualmente definidas para incluir um processador gerenciador que realiza a interface com a RPF (ARNOLD, 1996).

Infelizmente, quando a RPF está integrada dentro do sistema computacional como um co-processador independente, a largura de banda limitada entre a CPU e a lógica reconfigurável pode ser um gargalo significante no desempenho. Outros sistemas incluem uma RPF como uma unidade funcional extra acoplada com um núcleo processador mais tradicional em um chip. Quão bem acoplada é a RPF com o plano de controle do processador, é variável.

Dentro desta definição de acoplamento, podemos classificar as RPF's como: fracamente acopladas ou fortemente acopladas.

3.2.1 Fracamente acoplada

O processador comercial Reconfigurable Communication Processor (RCP) (HAUCK; DEHON, 2008, p.41) foi criado pela Chameleon Systems Inc. Este possui um subsistema com processador embarcado com uma RPF através de um barramento compartilhado proprietário conhecido como barramento RoadRunner (Figura 13).

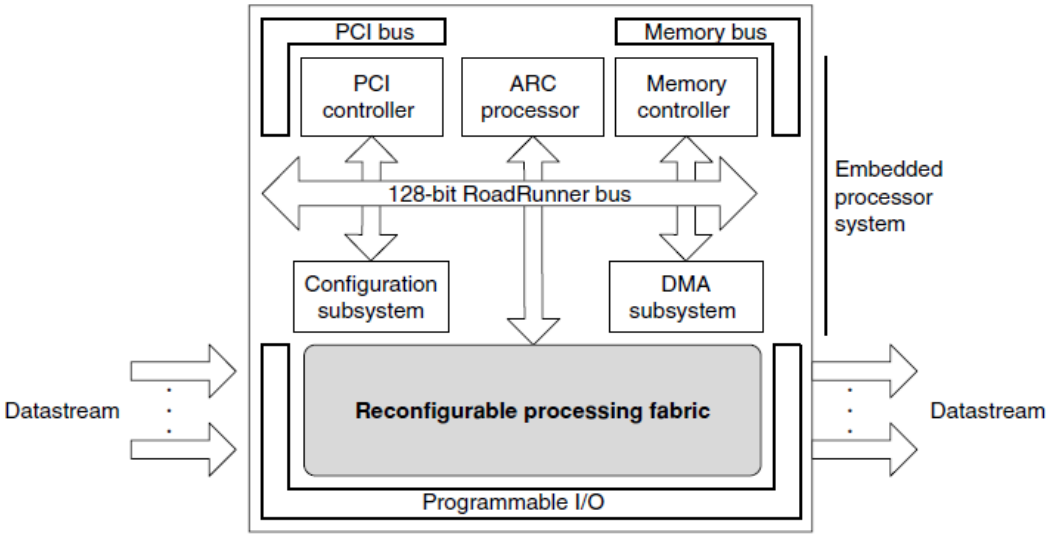


Figura 13. Arquitetura RCP do Chamaleon. Fonte: (HAUCK; DEHON, 2008, p.41)

A RPF tem acesso direto ao processador como também acesso direto à memória (DMA). A estrutura reconfigurável também tem uma interface de entradas e saídas programável, sendo assim, os usuários podem processar fora do chip independente do resto do sistema embarcado *on-chip*. Isso permite mais flexibilidade para a RPF do que em arquiteturas de computação reconfigurável típicas, as quais a RPF normalmente possui acesso somente ao processador e memória. A arquitetura Chameleon foi capaz de fornecer melhor custo/benefício em relação aos DSP's de mais alto desempenho de sua época, mas o RCP consumia muito mais energia por causa da RPF. Após 2002, existiam poucas menções do Chameleon ou sobre o RCP. Conceitualmente, o produto foi uma idéia interessante, mas ele não conseguiu capturar um nicho de mercado durante a desaceleração do mercado de eletrônicos.

3.2.2 Fortemente acoplada

A Figura 14, ilustra a arquitetura de fluxo de dados de um processador tradicional com a RPF integrada como uma RFU. Tais sistemas acoplam fortemente a RFU ao fluxo de dados de uma unidade de processamento central de maneira similar à tecnologia das FU's de CPU's tradicionais, tal como uma ULA, um multiplicador e o FPU. Em alguns casos estas arquiteturas fornecem somente acesso de dados para a RFU através de um registrador da mesma maneira como o tradicional CPU FU's, como por exemplo Chimaera (S. HAUCK, 1997) e PRISC (R. RAZDAN, 1994). Outras arquiteturas permitem a RFU acessar dados armazenados em uma memória/*cache* local diretamente (ex. OneChip (J. E. CARRILLO, 2001)).

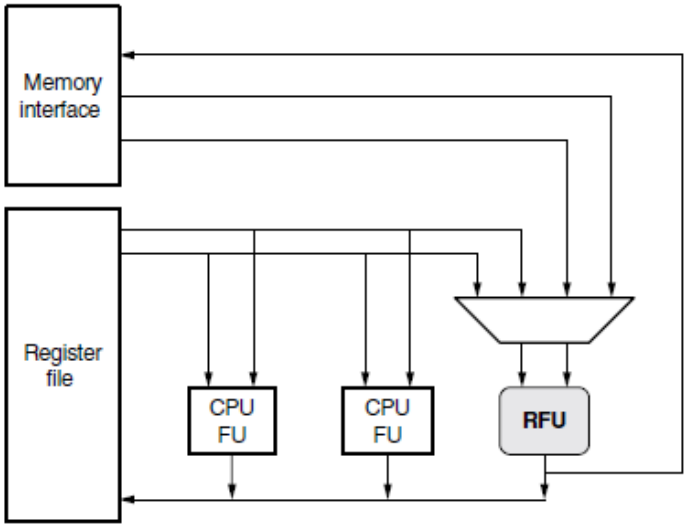


Figura 14. O fluxo de dados do processador + RFU arquitetura.
Fonte: (HAUCK; DEHON, 2008, P.42)

Para arquiteturas computacionais reconfiguráveis nas quais a RFU é fortemente acoplada com o núcleo do processador, o *pipeline* deve ser feito como mostrado na Figura 15.

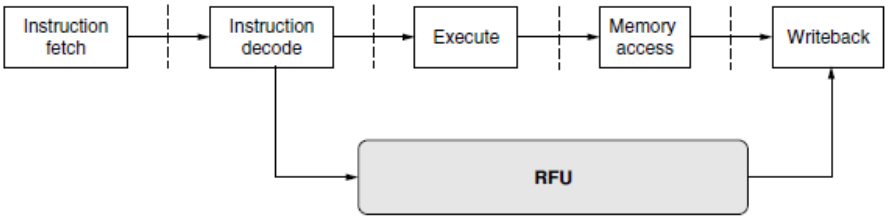


Figura 15. Exemplo de um pipeline de um processador.
Fonte: (HAUCK; DEHON, 2008, P.42)

Na seção seguinte serão apresentados alguns modos de gerenciamento das unidades de reconfiguração.

3.3 Modos de reconfiguração

Uma RU (*Reconfigurable Unit*) pode ser programada em diferentes momentos. Se feita somente no início, antes da execução começar, então esta RU não é reconfigurável e sim configurável (BECK FL.; CARRO). Se a RU pode ser programada após a inicialização então ela é chamada de reconfigurável. Uma aplicação pode ser dividida em blocos segundo suas funcionalidades, então uma RU pode ser reconfigurada de acordo com as necessidades de cada bloco

individualmente. Dessa maneira, a adaptação do programa é feita em um bloco básico.

A lógica reconfigurável é mais simples de ser implementada se a estrutura está bloqueada durante a reconfiguração. Contudo, se a unidade reconfigurável puder ser usada enquanto está sendo reconfigurada, é possível aumentar o desempenho. Isto pode ser feito, por exemplo, dividindo-se a unidade reconfigurável em segmentos os quais podem ser configurados independentemente uns dos outros. O processo de reconfigurar apenas partes da lógica é chamada de reconfiguração parcial (COMPTON; 2002).

Os tempos de reconfiguração dependem do tamanho dos dados de configuração. Os dados de configuração são usualmente compostos de bits de configuração para a reconfiguração da unidade, bem como de informações sobre o contexto da entrada. Estes tempos dependem do método de configuração usado. Por exemplo, no processador PRISC (ATHANAS;1993), a unidade reconfigurável é configurada copiando os dados de configuração diretamente para dentro da memória de configuração através de instruções *load/store*.

Se a tarefa é realizada por uma unidade de configuração na qual é capaz de buscar os dados de configuração, enquanto o processador está executando código, um ganho de desempenho pode ser obtido. Assim, pré-buscar os dados de configuração pode reduzir o tempo em que o processador está parado esperando pela reconfiguração. O emprego desta abordagem pode ser feito automaticamente pelas ferramentas de software (PANAINTE;2007).

A seguir são apresentados dois modos de gerenciamento de reconfiguração: a estática e a dinâmica.

3.3.1 Reconfiguração estática

Como apresentado no início da seção anterior uma unidade de reconfiguração pode ter diversos modos de gerenciamento. É dito que uma RU é uma unidade estática quando é configurada apenas durante a inicialização do sistema, não podendo assim, ser alterada durante a execução da aplicação. Tais unidades são muito empregadas em FPGA’s. Nestes, suas unidades são configuradas no momento em que a configuração é carregada para o FPGA, normalmente com o uso de uma memória externa. Claramente há uma perda na

flexibilidade, pois a reconfiguração (ou simplesmente configuração) é feita apenas durante a inicialização.

Segundo (SANCHEZ; et. al, p.2 1999) um sistema estático tem dois objetivos principais: 1) melhorar o desempenho (por exemplo, tempo de execução) para uma certa tarefa, na qual resulta em um rápido co-processador (um co-processador de MPEG) estendendo assim, o conceito de co-processador; e 2) otimizar a utilização dos recursos (número de portas e consumo) usando a maior quantidade possível da superfície da RU. Por exemplo, uma tarefa pode ser dividida em várias sub-tarefas, cada qual implementada com sua configuração e então carregada para dentro da RU no momento da inicialização.

É certo que neste tipo de sistema há a perda da flexibilidade dada pela reconfiguração, porém podem ser citados como vantagens: a eliminação do tempo de reconfiguração durante a execução e a economia de energia devida a esta tarefa.

3.3.2 Reconfiguração dinâmica

Em oposição aos sistemas estaticamente reconfiguráveis temos os sistemas dinamicamente reconfiguráveis, que por sua vez podem ser divididos em: sistemas parcialmente ou totalmente reconfiguráveis. Lysecky (R. LYSECKY, 2006) foi um dos primeiros a propor técnicas de reconfiguração dinâmica para detectar partes do código de programa que podem ser aceleradas por uma UR, a partir do código binário ao invés de usar um compilador especialmente feito para tal finalidade.

Tais sistemas visam melhorar pelo menos dois pontos de um sistema computacional: 1) adaptação a mudanças nas especificações da tarefa (por exemplo, um sistema robótico autônomo colocado em um novo ambiente); e 2) eliminar a variável humana no re-projeto do sistema (SANCHEZ; et. al, p.2 1999).

3.3.2.1 Reconfiguração parcial

O tempo gasto na reconfiguração da RU é uma característica a qual interfere diretamente no desempenho do sistema. Como nem sempre as reconfigurações necessitam alterar o hardware inteiro, podemos reduzir o tempo gasto nesta etapa se reconfigurarmos apenas os blocos necessários, obtendo melhor desempenho e economia de energia.

Reconfiguração parcial também pode permitir múltiplas configurações independentes a serem trocadas no hardware, ou seja, uma configuração pode ser re-fixada seletivamente no chip enquanto outra configuração é deixada intacta. Além disso, podemos aproveitar o endereçamento para modificar apenas uma parte da configuração já localizada no chip se alguma de suas estruturas corresponde a uma nova configuração que necessita ser carregada. Por exemplo, num circuito de criptografia a maior parte da configuração pode permanecer a mesma, quando a chave é alterada, e apenas alguns poucos recursos podem necessitar serem alterados com base no valor da nova chave. Reconfiguração parcial pode permitir o sistema reconfigurar apenas os recursos alterados, ao invés do circuito todo.

O FPGA Xilinx 6200 (Xilinx; 1997) foi um dos primeiros dispositivos parcialmente reconfiguráveis onde cada bloco lógico podia ser programado individualmente. Sendo assim, transformou-se em uma plataforma para uma grande quantidade de estudos de arquiteturas de configuração e RTR (*Run-time reconfiguration*). Atualmente FPGAs parcialmente reconfiguráveis comerciais incluem a AT40K Atmel (Atmel;1999) e a Xilinx Virtex família FPGA.

A série Virtex é mais grosseiramente reconfigurável do que 6200. Nesta série ao invés de endereçar cada bloco lógico independente, os blocos lógicos são agrupados em quadros. No Virtex II, um quadro corresponde a uma parte de uma coluna inteira de recursos e o tamanho do quadro aumenta com o número de linhas de blocos lógicos no dispositivo. Já no Virtex 5, os quadros têm um tamanho fixo de 41 palavras de 32 bits (independentemente do tamanho do dispositivo) o qual representa uma coluna parcial de recursos.

Embora projetos parcialmente reconfiguráveis forneçam mais flexibilidade para sistemas RTR, eles podem ainda sofrer com problemas potenciais. Primeiro, se as configurações ocupam grandes áreas do dispositivo, o tempo economizado transmitindo dados de configuração pode ser superado pelo tempo gasto na transmissão dos endereços de configuração.

Neste caso, um FPGA programado serialmente pode ser mais apropriado. Segundo, e mais crítico para sistemas RTR, configurações parciais são geralmente fixadas em localizações específicas no dispositivo. Se duas configurações independentes são implementadas nas mesmas localizações de hardware, elas não poderão operar simultaneamente.

3.3.2.2 Total

FPGA de memória de contexto simples são programados usando um canal serial para receber a informação de configuração. Por causa disto, somente acessos seqüenciais são permitidos, quaisquer mudanças na configuração deste tipo de FPGA exige uma reprogramação completa da RU do chip. Através desta técnica o circuito de reconfiguração fica mais simples, porém cria uma penalidade muito alta em tempo e energia quando se é preciso reconfigurar apenas uma pequena parte da RU. Muitos FPGAs comerciais usam este estilo de reconfiguração, incluindo os da série Xilinx 4000 (Xilinx;1994), a série Altera Flex 10k (Altera;1998), e a série da Lucent Orca (Lucent;1998). Este tipo de FPGA é, portanto, mais adequado para aplicações que podem se beneficiar da computação reconfigurável sem reconfiguração em tempo de execução.

Na Figura 16 são apresentados dois tipos diferentes de reconfiguração. Na parte superior temos um exemplo de reconfiguração utilizando um FPGA com memória de contexto simples onde temos novos dados de configuração chegando, representados pelo quadrado à esquerda. Neste, o retângulo em cinza representando os dados novos e o restante, em preto, os dados que não serão alterados. No centro da Figura 16 temos a situação atual da RU e no canto mais a direita o estado da RU após da reconfiguração.

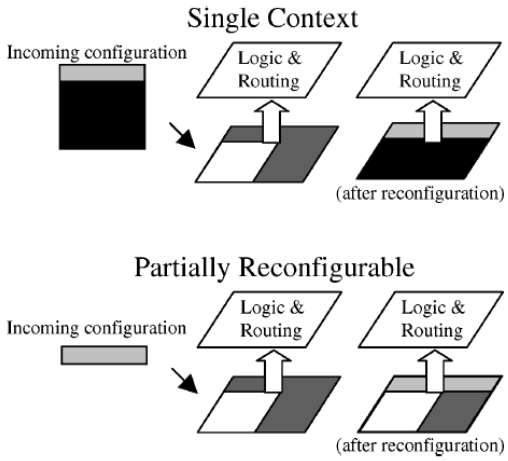


Figura 16. Reconfiguração Total (contexto simples) e reconfiguração Parcial. Cada um realizando uma reconfiguração. (Fonte: COMPTON, K., HAUCK S., 2002, p.196)

Para se utilizar este tipo de FPGA em aplicações de reconfiguração em tempo de execução, deve-se tomar o cuidado de manter as configurações dentro de

contextos, onde serão trocados inteiramente na memória do FPGA quando necessários. Por causa destas operações de troca de conteúdo da memória de configuração do FPGA, um bom particionamento entre os contextos é essencial para diminuir o atraso com o tempo de reconfiguração. Se todas as configurações, usadas dentro de certo período de tempo, estão presentes dentro do mesmo contexto, isso eliminará a necessidade da reconfiguração. Por outro lado, se um número de configurações sucessivas estão presentes em particionamentos diferentes, diversas reconfigurações serão necessárias tornando lenta a reconfiguração do sistema.

Também na Figura 16, na parte inferior temos um exemplo de FPGA que utiliza a reconfiguração parcial em sua RU. Como pode ser visto apenas a área com novos dados de configuração será afetada mantendo o restante da RU. Este tipo de FPGA é mais adequado para aplicações com reconfiguração em tempo de execução, pois uma vez que parte de sua RU pode ser reconfigurada o tempo gasto na tarefa pode ser minimizado.

3.4 Estruturas heterogêneas

Para fornecer maior desempenho ou flexibilidade na computação, alguns sistemas reconfiguráveis utilizam uma estrutura heterogênea, onde as capacidades das células lógicas são diferentes ao longo do sistema. Um dos usos para tal arquitetura é a de, por exemplo, fornecer blocos multiplicadores embarcados dentro de estruturas reconfiguráveis. Isto porque, a multiplicação é uma das mais difíceis computações de serem implementadas eficientemente na estrutura de um FPGA tradicional, um bloco multiplicador personalizado embarcado dentro de uma matriz reconfigurável possibilita que tal sistema realize a operação de forma mais eficiente.

Outro exemplo de uso de estruturas heterogêneas é a de fornecer blocos de memória espalhados pela estrutura reconfigurável. Isto permite armazenar dados e variáveis frequentemente usados, fazendo com tais dados sejam acessados mais rapidamente pela proximidade de tal memória com os blocos lógicos que precisem consumir tais dados. Estruturas de memória embarcada dentro de RU’s podem ser encontradas sendo usadas simplesmente como LUT’s trabalhando como estruturas

de RAM, como na série Xilinx 4000 (Xilinx; 1994) e na série de FPGA's Virtex (Xilinx; 1999).

Algumas arquiteturas incluem blocos de memória dedicados dentro de suas matrizes, tal como a série Xilinx Virtex (Xilinx; 1999) e no FPGA's Altera (Altera; 1998), como também no dispositivo CS2000 RCP, da Chameleon Systems, Inc. (Chameleon;2000). Estes blocos de memória tem maior desempenho em tamanho maiores do que estruturas similares feitas de diversas pequenas LUT's, enquanto estas estruturas são um pouco menos flexíveis do que memórias baseadas em LUT's, elas podem também fornecer alguma personalização. Por exemplo, o FPGA Altera FLEX 10k (Altera; 1998) fornece memórias embarcadas as quais tem um número total de fios limitado, mas apresentam um bom custo benefício entre o número de linhas de endereço e na largura de bits dos dados.

3.5 Arquitetura reconfigurável e DSP

Processadores de sinais digitais de uso geral (PDSP's), têm se aproveitado do sucesso das últimas duas décadas. Eles são baseados em um paradigma de um conjunto reduzido de instruções (RISC) com uma arquitetura que consiste de pelo menos um multiplicador matricial rápido (por exemplo, 16 x 16 bits à 24 x 24 bits de ponto fixo, ou 32 bits em ponto flutuante), com um acumulador com largura de palavra estendida.

A vantagem do PDSP vem do fato de que muitos algoritmos de processamento de sinais usam a multiplicação e acumulação (MAC) intensivamente. Pelo uso de uma arquitetura com pipeline de múltiplos estágios, PDSP's podem alcançar taxas de MAC limitadas somente pelo desempenho do multiplicador matricial. Pode-se afirmar que um FPGA pode ser usado também para implementar células MAC (MEYER-BAESE, 2007, p.30), mas questões de custo dão vantagem aos PDSP's se estes encontram a taxa MAC desejada. Por outro lado, podem-se encontrar diversas aplicações de processamento de sinais com altas larguras de banda tais como: wireless, multimídia, ou transmissão via satélite, e a tecnologia FPGA pode fornece mais largura de banda através do uso de múltiplas células MAC em um único chip.

Meyer-Baese (2007) afirma que FPL’s (field-programmable logic) são mais eficientes do PDSP’s para a execução de diversos algoritmos, tais como: CORDIC, NTT ou algoritmos de correção de erros. Meyer-Baese (2007) afirma também que os PDSP’s irão dominar as aplicações que exigem algoritmos complicados, enquanto que FPGA’s irão dominar aplicações de monitoramento de sensores que se utilizam de filtros FIR, algoritmos CORDIC, ou FFT’s.

4. METODOLOGIA DO TRABALHO

O objetivo deste trabalho é demonstrar que através do uso da técnica conhecida como Computação Reconfigurável incorporada a um GPP pode-se acelerar a execução de algoritmos DSP. Também é objetivo deste trabalho, realizar a comparação entre PDSP comerciais, GPP e Sistema Reconfigurável selecionados. As comparações foram realizadas através de levantamento e análise dos dados da execução de aplicações com características DSP, executadas nas ferramentas de simulação selecionadas. De posse dos resultados foi possível verificar a aplicação da técnica de Computação Reconfigurável em um GPP.

Neste capítulo está descrita a metodologia e ferramentas empregadas na realização deste trabalho para atingir os seus objetivos. A Figura 17, mostra o fluxo de desenvolvimento da primeira parte do trabalho, que consistiu de executar aplicações selecionadas em um simulador para o MIPS32, gerando um arquivo de saída contendo o traço de execução das aplicações. Este arquivo serve como parâmetro de entrada para o simulador do Sistema Reconfigurável que também emula o MIPS32 acoplado ao sistema, gerando assim os resultados para comparações futuras. Na etapa final, as aplicações foram novamente executas no Sistema reconfigurável modificado para suportar instruções MAC afim de extrair os resultados das execuções das aplicações.

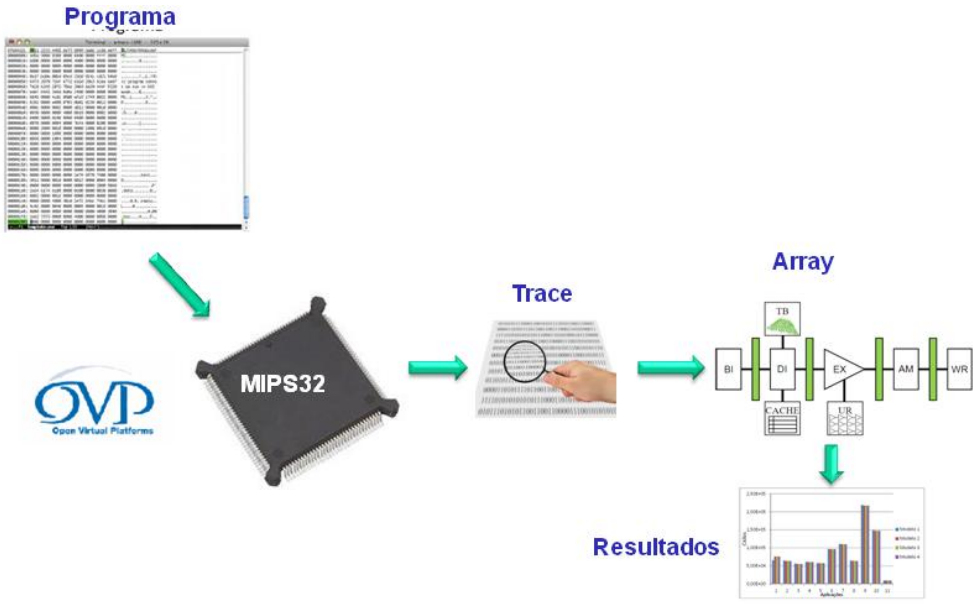


Figura 17. Fluxo de desenvolvimento do trabalho para o MIPS32.

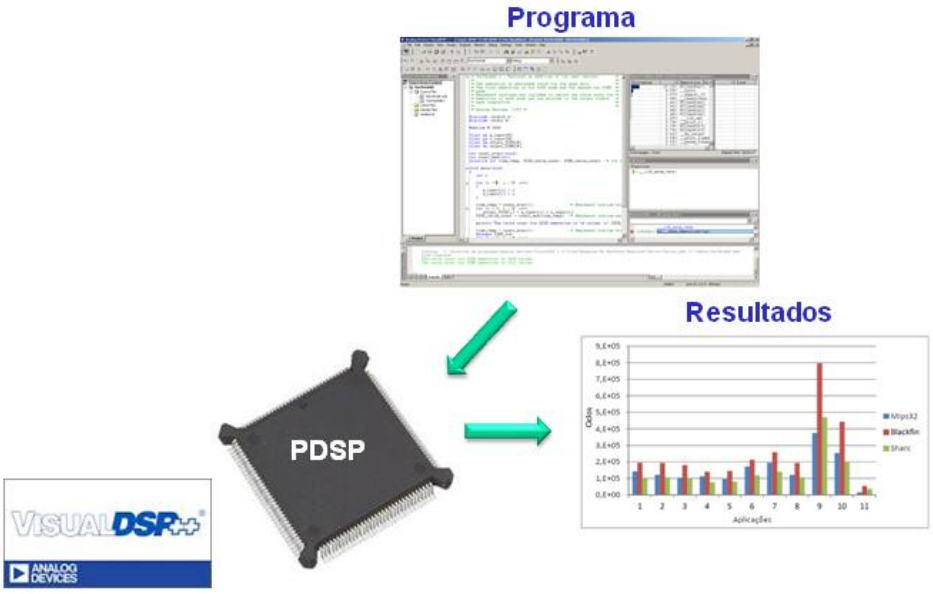


Figura 18. Fluxo de desenvolvimento do trabalho para os PDSP.

Na segunda etapa do trabalho, como visto na Figura 18, as mesmas aplicações selecionadas, foram executadas em uma ferramenta de simulação para PDSP de onde foi possível extrair o número de ciclos de execução. De posse de todos os resultados gerados nas várias etapas de desenvolvimento do trabalho, foi possível verificar o desempenho da proposta segundo cenários criados para as comparações, que serão explicados a seguir.

O capítulo 5 apresenta os resultados obtidos e estes são discutidos.

4.1 Ferramentas utilizadas

Para a seleção das ferramentas, foi realizada uma busca entre diversos fabricantes de PDSPs. Dos diversos fabricantes pesquisados, foram selecionadas três ferramentas para avaliação: Code Composer(TEXAS, 2013), Symphony Studio(FREESCALE, 2013) e VisualDSP++(ANALOG, 2013d).

A ferramenta VisualDSP++, da Analog Devices, possui inúmeros recursos, dentre eles, compilador C/C++, medida de desempenho estatística (*statistical profiling*), simulador, montador, ligador e suporte a emulação.

Após a seleção, as ferramentas foram avaliadas a fim de verificar se as mesmas poderiam gerar os dados necessários para a comparação, tais como: trace de instruções, desempenho e consumo energético. Embora as três ferramentas

forneçam dados necessários a partir das aplicações, somente a ferramenta VisualDSP++ não necessita de uma placa acoplada ao computador, eliminando a necessidade de placas de aquisição. Pelas razões expostas acima, a ferramenta VisualDSP++ foi a escolhida. A Figura 19 apresenta a tela da ferramenta VisualDSP++. Esta ferramenta fornece o tempo de execução (em ciclos) e também um histograma que consegue fornecer dados sobre o percentual de tempo de execução em termos de funções derivadas do código C. Porém não disponibiliza um histograma relativo às instruções em assembly mais executadas.

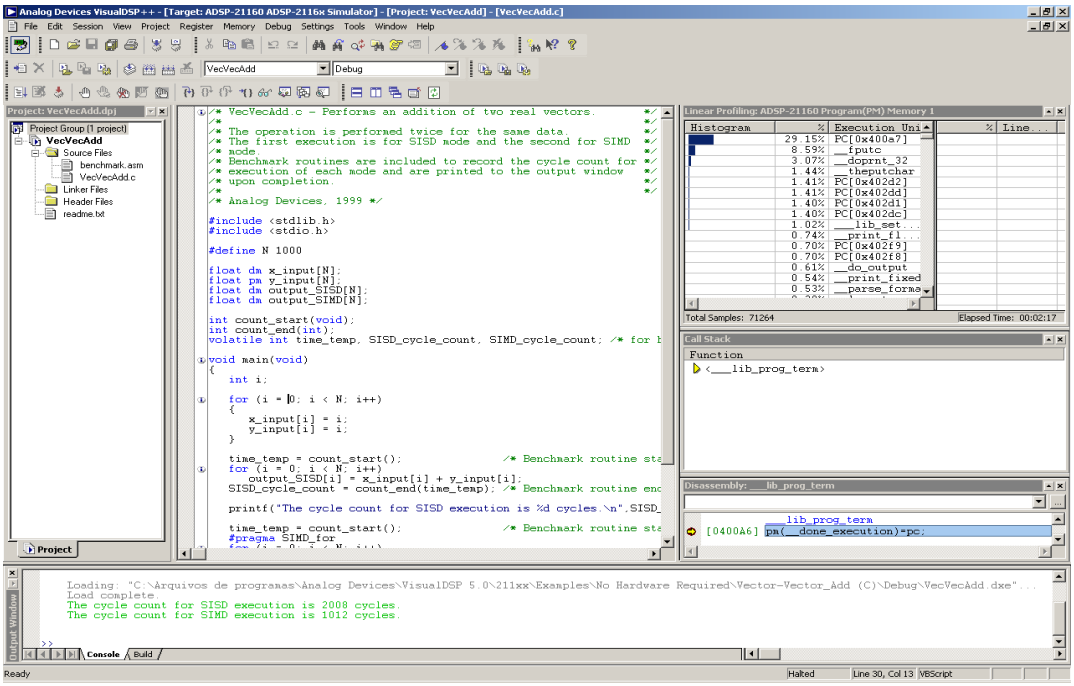


Figura 19. Ferramenta VisualDSP++ da Analog Devices.

Para a realização deste trabalho foi selecionado um *array* reconfigurável, proposto por (BECK, 2008). Este array utiliza uma unidade reconfigurável que funciona em conjunto com um hardware de tradução binária (*Binary Translation*). Esta unidade reconfigurável é fortemente acoplada ao processador, funcionando como mais uma unidade funcional no estágio de execução. Ela é composta por unidades funcionais simples (ULA, deslocadores, multiplicadores etc.). O array usa um simulador descrito em linguagem C para analisar o traço de uma aplicação e executá-la no modelo de array previamente configurado e compilado. Neste trabalho são propostos alguns modelos e em seguida são verificados os impactos nas variáveis de interesse como, tempo de execução, potência e energia.

Para simular o MIPS32 foi utilizado o OVPSIM (OVP, 2013), que é um simulador multiplataforma com precisão de instrução e código fonte aberto. Este fato foi importante para adaptar a saída deste simulador para que o ambiente do array pudesse interpretar os traços das aplicações.

4.2 Dos processadores selecionados

Com a ferramenta VisualDSP++ selecionada, é possível trabalhar com as seguintes famílias de processadores DSP (ANALOG, 2013a): ADSP-21XX, Blackfin, SHARC e TigerSHARC. Destas, foram selecionados dois processadores: o ADSP-BF504, da família Blackfin e o ADSP-21477, da família SHARC. A família de processadores Blackfin (ANALOG, 2013c) foi desenvolvida para aplicações de baixo custo que exigem altas taxas de processamento digital de sinais, tais como: vídeo e instrumentação. Dentre muitas de suas características, destacam-se as seguintes: instrução MAC(*multiply-accumulate*) de 16 bits (possibilidade de executar até 800 milhões de MACs por segundo) e duas ULAs (Unidade Lógica e Aritmética) de 40 bits.

Os processadores SHARC (ANALOG, 2013b), de 32 bits, são baseados na arquitetura super-harvard. Além destas, podemos citar como características comuns de sua arquitetura: unidade aritmética de 32/40 bits de ponto flutuante; multiplicadores de 32 bits, ponto fixo; todas as instruções são executadas em um ciclo.

De posse da ferramenta e da escolha dos processadores, passou-se a seleção de aplicações com características DSP.

4.3 Das aplicações selecionadas

Selecionados a ferramenta e os processadores de interesse, passou-se à seleção de aplicações com características DSP. A ferramenta VisualDSP++ contém um conjunto de aplicações exemplo que servem para verificar se os dados gerados atenderiam as necessidades deste trabalho e foram mantidas por apresentarem características DSP. Além destas aplicações, também foram incluídas aplicações largamente conhecidas nos meios acadêmico e comercial, como: *FFT, DFT, IFFT, IDFT* e *FIR*. Na Tabela 3, encontra-se uma lista de aplicações selecionadas e uma breve descrição sobre as mesmas.

Tabela 3. Lista de aplicações selcionadas.

Num.	Aplicação
1	Multiplicação de dois vetores reais
2	Adiciona duas matrizes bidimensionais
3	Subtrai duas matrizes bidimensionais
4	Multiplica os elementos de uma matriz por um escalar
5	Calcula a média de uma matriz
6	Calcula o valor RMS de uma matriz de dados
7	Acumula o produto de duas matrizes
8	Adiciona duas matrizes reais
9	Algoritmo FIR
10	Algoritmo FFT
11	Algoritmo IFFT
12	Algoritmo DFT
13	Algoritmo IDFT

As aplicações selecionadas apresentam uma série de características que viabilizam o uso de DSP, tais como: acesso a memória, operações com matrizes, operações com ponto flutuante, filtragem digital e cálculo de transformadas.

4.4 O sistema reconfigurável

O Sistema Reconfigurável selecionado foi proposto por Beck em (BECK, 2008) e (RUTZIG, 2008). Este sistema é composto por um hardware de tradução binária acoplado à unidade de decodificação, por um barramento, e por isto é dita como fortemente acoplado ao processador. A Unidade Reconfigurável do sistema é dita de granularidade grossa, porque processa informações na mesma largura de dados de uma unidade funcional básica do processador e encontra-se acoplada à unidade de execução. O sistema é dito dinâmico, pois realiza a reconfiguração em tempo de execução do programa (BECK, 2008). A arquitetura da Unidade Reconfigurável é composta por elementos simples, como ULAs, multiplicadores, registradores, multiplexadores, dentre outros, organizados de forma combinacional.

A unidade foi projetada originalmente para interpretar as instruções do processador MIPS R3000 e, como primeira contribuição deste trabalho, a unidade foi acoplada ao processador MIPS32 (MIPS32, 2012). Como segunda contribuição, foram adicionadas as instruções *madd* e *maddu* à Unidade Reconfigurável, que executam as operações de multiplica e acumula para valores inteiros positivos e negativos, respectivamente. Com a adição destas instruções, a Unidade

Reconfigurável passou a suportar algumas das características semelhantes às unidades MAC, presente nos PDSPs.

A Unidade Reconfigurável foi projetada para explorar ao máximo o paralelismo das instruções. Como podemos observar na Figura 20, a unidade pode executar as instruções de forma paralela ou sequencial. Instruções são executadas paralelamente quando não há dependência de dados entre elas, assim, são alocadas dentro da mesma linha. Havendo algum tipo de dependências entre as instruções, então, estas são alocadas em linhas diferentes, verticalmente.

A Figura 20 também mostra como são organizadas as unidades disponíveis. Para esta configuração, podemos ter até 4 instruções sendo executadas nas ULAs disponíveis de cada linha, duas multiplicações de números inteiros e dois Loads/stores (acessos à memória), todas estas executadas simultaneamente. Cada nível representa a quantidade de operações executadas dentro do tempo correspondente a um ciclo do processador. Sendo o circuito totalmente combinacional, não há barreiras temporais entre os níveis. Na Figura 20, também podemos notar a presença de uma tabela ao lado das unidades de multiplicação. Esta tabela armazena as informações referentes aos registradores que estão sendo escritos em cada linha da execução e é usada, também, pelo tradutor binário, para verificar a dependências das instruções no momento da alocação das unidades disponíveis.

Também podemos observar na Figura 20, que a arquitetura da Unidade Reconfigurável é regular, possibilitando ao projetista uma fácil configuração desta quanto ao número de unidades funcionais disponíveis. Esta facilidade é de fundamental importância para projetos de sistemas embarcados, visto que em tempo de projeto é necessário verificar o desempenho em área, velocidade e consumo destes sistemas antes da fabricação (BECK, 2008).

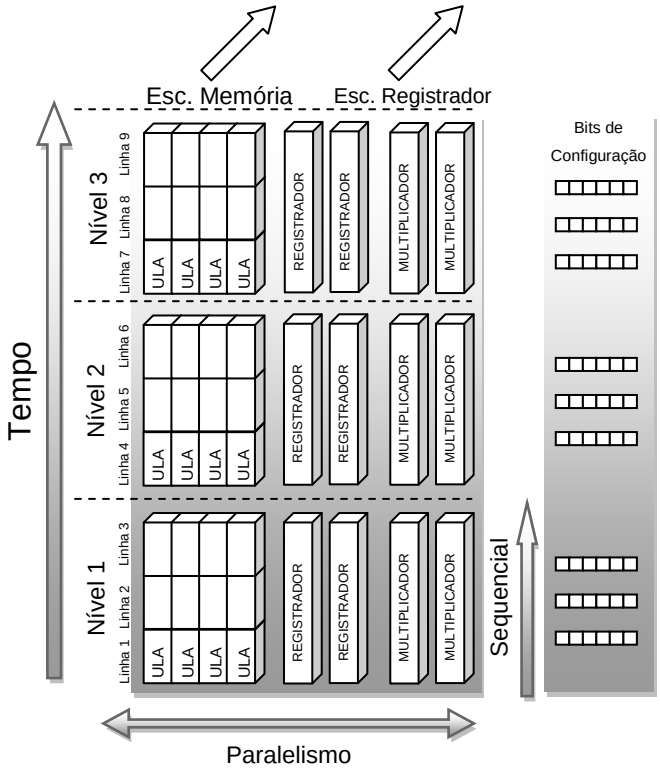


Figura 20. Estrutura da Unidade Reconfigurável.

Fonte: (BECK, 2008).

4.5 Da interconexão das unidades

As unidades funcionais da Unidade Reconfigurável são conectadas por multiplexadores, ao contexto de entrada e por demultiplexadores ao contexto de saída, como pode ser observado na Figura 21. Estas conexões estão presentes em todos os níveis da estrutura. Seu objetivo é o de fornecer os valores do contexto de entrada para as unidades funcionais e os valores de saída, após computados, para então serem gravados no contexto de saída.

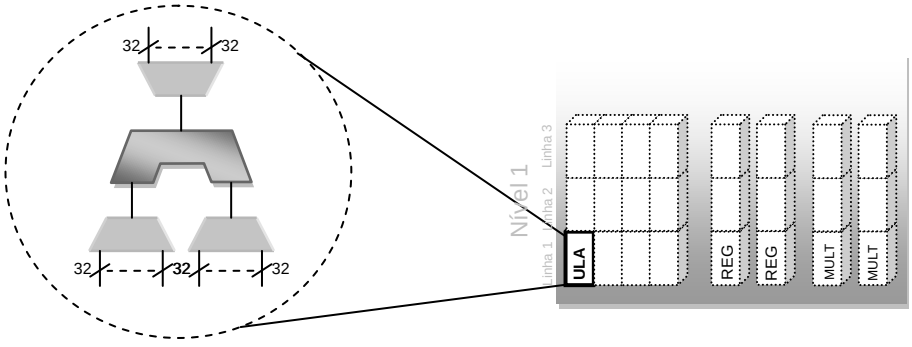


Figura 21. Conexão dos dados.

Fonte: (BECK, 2008).

Segundo Rutzig(RUTZIG, 2008, p. 35), a interconexão entre as unidades funcionais da estrutura aumenta a flexibilidade na alocação das instruções. Porém, traz como consequência o aumento na área e no consumo de energia, consequentes da intensa comunicação firmada entre as unidades funcionais.

4.6 Sistema completo

Na Figura 22, é mostrado o diagrama de blocos do sistema reconfigurável completo proposto por Beck (BECK, 2008). O sistema acoplado ao processador é composto pela Unidade Reconfigurável (UR), memória *cache* de reconfigurações (Cache) e pelo circuito de tradução binária (TB). Estes blocos são os responsáveis pela execução das aplicações de forma reconfigurável. Os blocos: busca da instrução (BI), decodificação da instrução (DI), execução (EX), leitura e escrita na memória (AM) e escrita no registrador (ER), representam os estágios do *pipeline* do processador mips32.

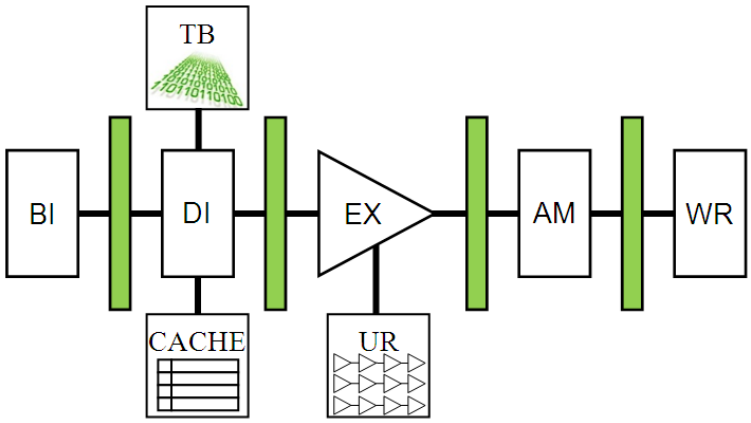


Figura 22. Diagrama de blocos do Sistema Reconfigurável.

Fonte: (BECK, 2008).

O bloco de Tradução Binária é responsável pela correta configuração das instruções na Unidade Reconfigurável, mantendo-se a compatibilidade binária, entre o processador e UR. O mesmo bloco pode ser aproveitado em trabalhos futuros que utilizem a Unidade Reconfigurável em outro processador, por exemplo.

A memória *cache* de reconfigurações armazena as configurações geradas na Unidade Reconfigurável para que as mesmas possam ser reutilizadas sem a

necessidade de uma nova avaliação por parte do bloco de detecção. Isto aumenta o desempenho do sistema, economizando tempo e energia na etapa de reconfiguração da unidade, diferentemente dos processadores supercalares que verificam a cada instrução a dependência entre as mesmas para então alocar as instruções nas diversas unidades. Esta técnica é chamada de reuso de rastro (do inglês *trace reuse*). Nesta unidade ficam armazenados os dados necessários para a configuração dos multiplexadores e demultiplexadores, dos contextos de entrada e de saída da unidade reconfigurável, e as informações de configuração de cada unidade de processamento (ULAs), sendo utilizado o contador de programa como indexador para a primeira instrução encontrada possível de ser usada na Unidade de Reconfiguração.

Como terceira contribuição deste trabalho, foram propostos cenários de comparação envolvendo PDSPs comerciais e a Unidade Reconfigurável acoplada ao processador MIPS32. Na seção seguinte são detalhados os cenários propostos.

5. RESULTADOS OBTIDOS

Nesta seção são mostrados os resultados obtidos da execução das aplicações selecionadas nos diferentes cenários, além da análise do impacto nas variáveis de interesse: desempenho (tempo de execução), área ocupada, tempo de reconfiguração, energia e potência. Para facilitar as análises os resultados são apresentados na forma de gráficos, considerando que os processadores executam as aplicações nas frequências máximas permitidas que são: 190 MHz para o MIPS32 (considerando o processador 4k), 200 MHz para o processador SHARC e 400 MHz para o processador Blackfin.

5.1 Cenários de comparação

Como mencionado na seção anterior, a terceira contribuição deste trabalho foi na comparação do processador MIPS32, com o sistema reconfigurável acoplado, frente a dois PDSPs comerciais selecionados.

Em seguida, estes dados são usados para comparar os desempenhos, a fim de se verificar se foi obtido aceleração com o uso da técnica de reconfiguração. As comparações foram feitas segundo os seguintes cenários propostos:

(1) PDSPs vs. MIPS32;

(2) PDSPs vs. MIPS32+U.R.;

(3) PDSPs vs. MIPS32+U.R. DSP.

O cenário 1, foi utilizado para verificar qual a diferença de tempo de execução das aplicações nos processadores selecionados. No cenário 2 a Unidade Reconfigurável foi acoplada ao processador MIPS32 e as aplicações selecionadas foram executadas novamente para realizar a segunda comparação de desempenho. No cenário 3 temos o processador MISP32 com a UR acoplada e modificada para aceitar as instruções *madd* e *maddu* (MAC) e a comparação de seu desempenho contra os PDSPs.

As diferentes aplicações foram executadas nos processadores comerciais, e os dados de execução foram comparados com os dados de execução gerados pelo processador MIPS32(MIPS32, 2012) com e sem o Sistema Reconfigurável acoplada.

5.2 Avaliação das aplicações em diferentes modelos da UR

Com objetivo de avaliar as diferentes aplicações de DSP no Sistema Reconfigurável, foram utilizados quatro modelos de configuração para o sistema reconfigurável, propostos por (RUTZIG, 2008), apresentados na Tabela 4. Os diferentes modelos diferem na quantidade de PEs que impactam na área ocupada pela unidade, além de energia e potência, e também podem impactar em termos de tempo de execução.

Tabela 4. Modelos da Unidade Reconfigurável.

	Modelo 1	Modelo 2	Modelo 3	Modelo 4
Total de linhas	24	48	150	189
Total de colunas	11	16	20	203
ULA/Linha	8	8	12	55
Mul/Linha	1	2	2	14
Reg/linha	2	6	6	134

5.2.1 Área ocupada

A inserção da UR acarreta em um aumento significativo em área ocupada. A Tabela 5 mostra a área ocupada para cada modelo proposto, em termos de componentes lógicos, em número de portas.

Tabela 5. Área ocupada para cada modelo (RUTZIG, 2008, p. 53).

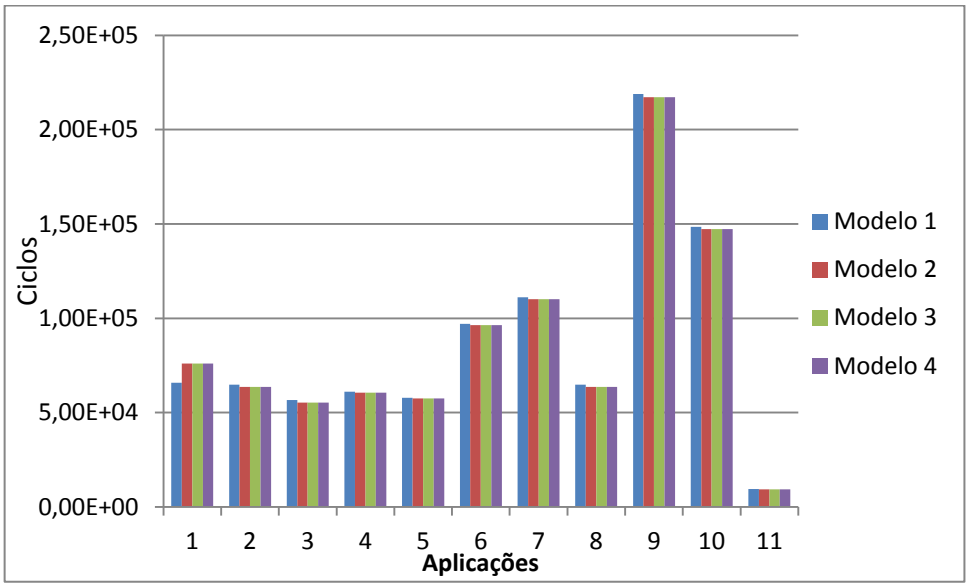
	Modelo 1	Modelo 2	Modelo 3	Modelo 4
ULA	294.032	600.576	2.815.200	11.870.760
Load	34.112	31.488	98.400	2.021.792
Multiplicador	20.067	214.048	668.900	4.307.716
Multiplexador	378.780	657.408	2.824.800	18.487.032
Demultiplexador	58.752	117.504	367.200	337.824
Total (em portas)	785.743	1.621.024	6.774.500	37.025.124

Em termos de elementos lógicos o processador MIPS32 4kc (MIPS32, 2012), ocupa em torno de 8 mil elementos lógicos (ALTERA, 2014). Sendo assim, o modelo 1 ocupa em torno de 98 vezes mais área que o processador MIPS32 4kc. A área

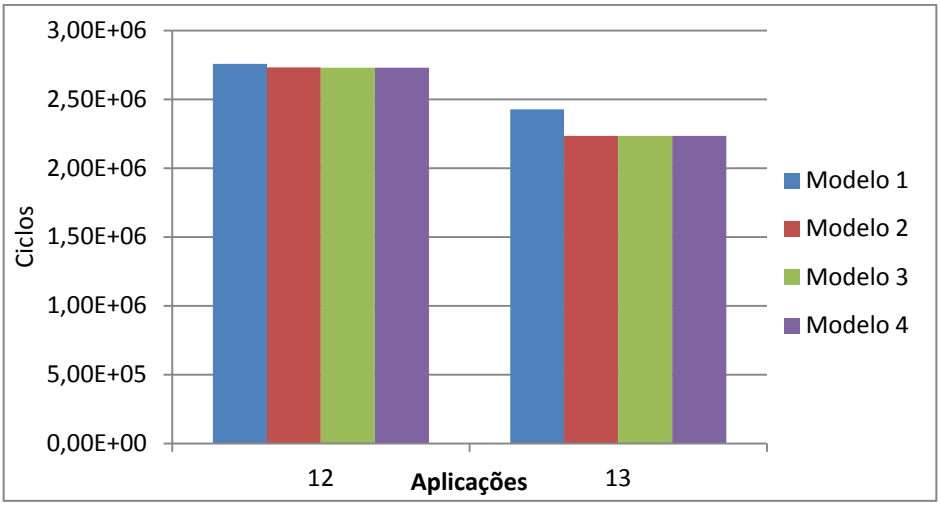
ocupada pelo modelo 2 aumenta para 202 vezes e para os modelos 3 e 4, respectivamente, aumentam para 847 e 4628 vezes mais.

5.2.2 Avaliação em termos de tempo de execução

Na Figura 23 apresenta o tempo de execução considerando os diferentes modelos propostos. Podemos observar que as aplicações executadas no modelo 1, apresentam um número de ciclos ligeiramente maior do que nos outros modelos, porém, este é o que menos ocupa área. Já os modelos 2, 3 e 4 não apresentam diferenças significativas em número de ciclos, porém, ocupam enormes áreas. Portanto, podemos concluir que o modelo 1 apresenta o melhor custo benefício e isto ficará mais evidente nas seções seguintes. As tabelas com os dados das simulações encontram-se no Apêndice A e B.



(a)



(b)

Figura 23. Tempo de execução das aplicações nos modelos propostos.

Os resultados de aceleração das aplicações nos modelos propostos para a arquitetura da UR são mostrados na Figura 24. Podemos observar que houve uma aceleração na execução das aplicações que foi, na média, de 1,79 em comparação com o processador MIPS32 sem a UR acoplada. Também é possível observar que o modelo 1 já extrai o máximo paralelismo presente nas aplicações. Isto mostra que a possibilidade de configurar a quantidade de elementos disponíveis na unidade reconfigurável permite ao projetista verificar em tempo de projeto os impactos mencionados anteriormente.

Para completar as comparações propostas, foi adicionado suporte às instruções *madd* e *maddu*, mencionadas na seção 4.4. Estas instruções foram implementadas somente no modelo 1, pois, este modelo apresenta o melhor custo benefício em termos de aceleração, área e consumo. Por estes motivos este modelo foi selecionado para o restante das comparações. Os outros modelos foram descartados porque apresentam aumentos significativos de área e consumo muito maiores do que no modelo 1, não justificando o seu uso.

Podemos observar, na Figura 25, que a adição do suporte às instruções mencionadas forneceu um aumento na aceleração obtida apenas para as aplicações que utilizaram tais instruções. As aplicações 3, 5, 8 e 13 não se utilizaram destas instruções e, portanto, não foi observado aceleração. Na média a aceleração com suporte as instruções DSP ficou em 1,92, contra 1,79 no modelo sem suporte.

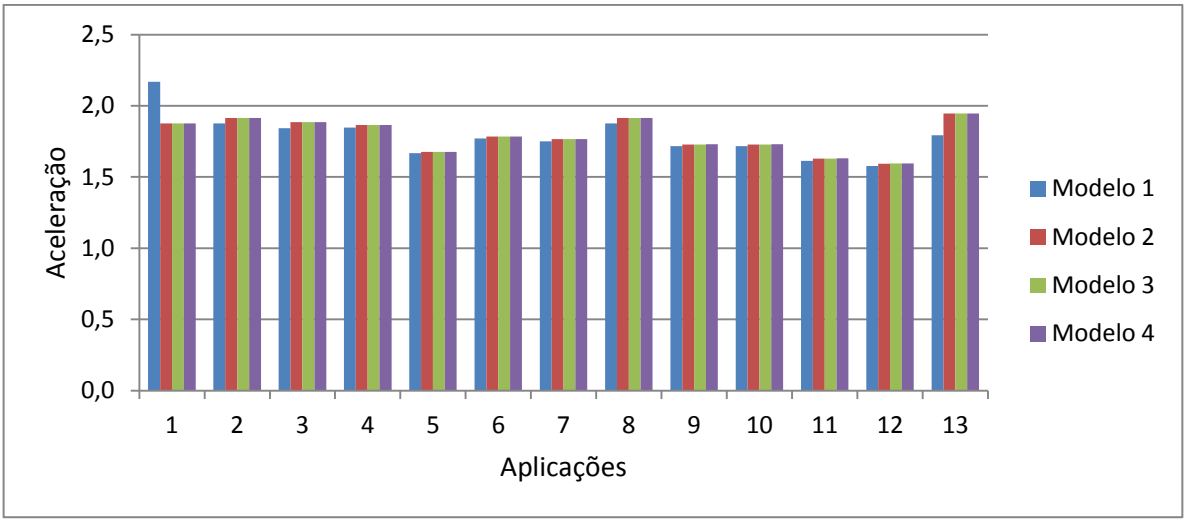


Figura 24. Aceleração das aplicações selecionadas nos diferentes modelos.

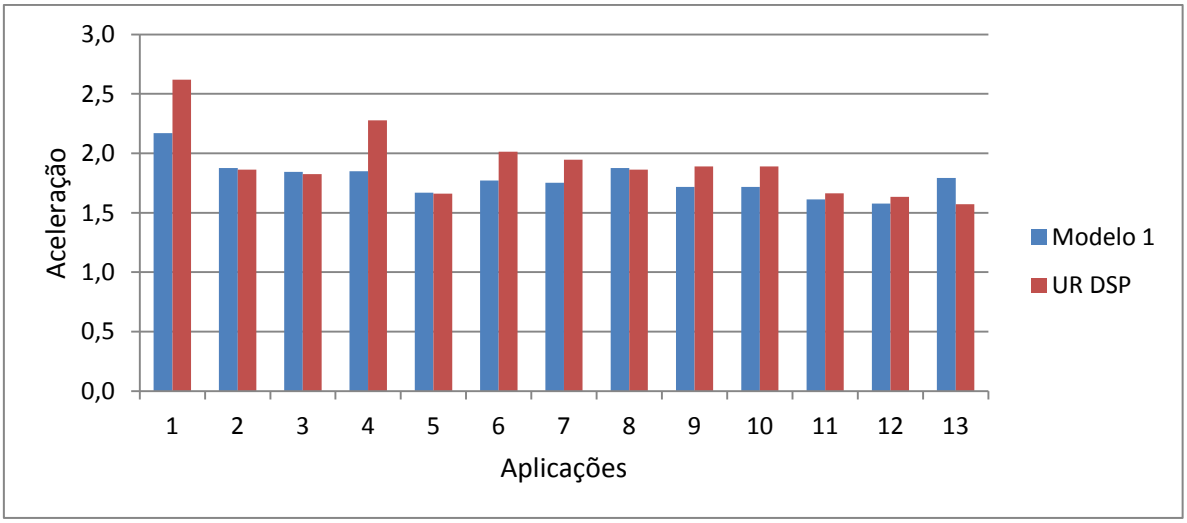


Figura 25. Aceleração das aplicações selecionadas no Modelo 1 e UR DSP.

5.2.3 Tempo de reconfiguração

O tempo gasto pela arquitetura para efetuar a reconfiguração é um parâmetro de especial importância porque ela indica se houve ou não aceleração na execução. Na Figura 26, temos a quantidade de ciclos de relógio gastos para realizar as reconfigurações do sistema para cada aplicação e modelos propostos.

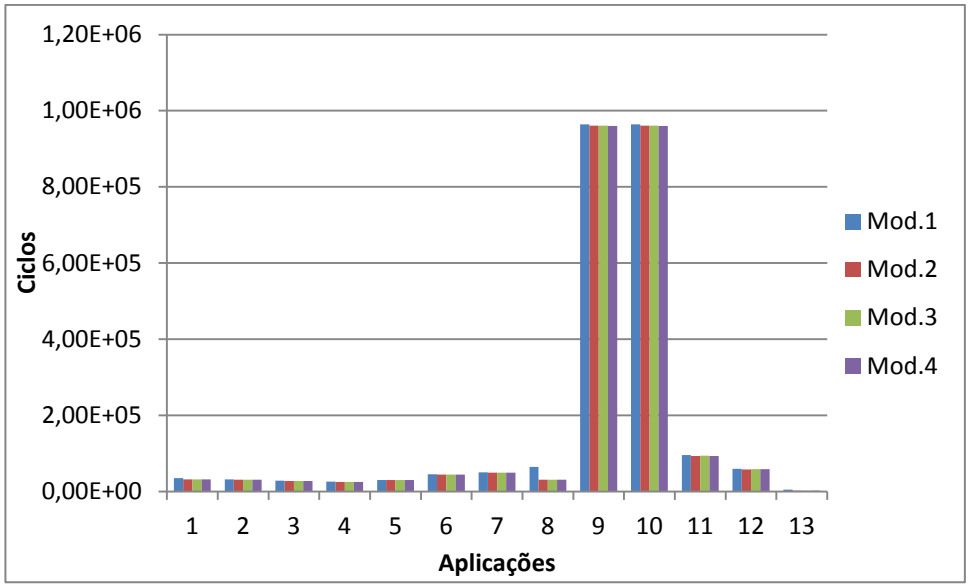


Figura 26. Número de ciclos gastos na reconfiguração.

Podemos observar, da Figura 26, que o número de ciclos gastos nas reconfigurações não é afetado pelo tamanho do sistema reconfigurável e sim pela natureza das próprias aplicações. Isto fica evidente nas aplicações 9 e 10, estas aplicações se utilizam de matrizes alocadas no banco de registradores.

5.2.4 Consumo Energético

É de extrema importância o conhecimento do consumo energético em aplicações de sistemas embarcados, portáteis, principalmente em tempo de projeto. Com o aumento dos dispositivos móveis, como celulares, por exemplo, a autonomia da bateria é de fundamental importância para o projetista.

Na Figura 27, observamos o consumo energético do sistema reconfigurável acoplado ao processador MIPS32, para cada modelo proposto e do processador sem a acoplagem do sistema. Pode-se observar que o consumo do sistema aumenta juntamente com o tamanho dos modelos, o que era esperado, devido ao aumento do número de elementos de processamento de cada modelo.

A Figura 27, também apresenta que a acoplagem do sistema contribuiu para uma diminuição da energia total consumida durante a execução das aplicações nos modelos propostos, para quase todas as aplicações, mostrando que é vantajoso o uso desta técnica em sistemas embarcados. A redução de energia ficou na ordem de 45%, 36%, 11% e 2%, na média, respectivamente, para os modelos propostos, na tecnologia de 90 nm. Já na Figura 28 demonstra o consumo energético para a

tecnologia de 180 nm, onde podemos observar uma redução de consumo da ordem de 42%, 34%, 14% e 4%, na média, respectivamente para os modelos propostos. Onde se justifica o uso do modelo 1, por este ter apresentado a maior redução no consumo.

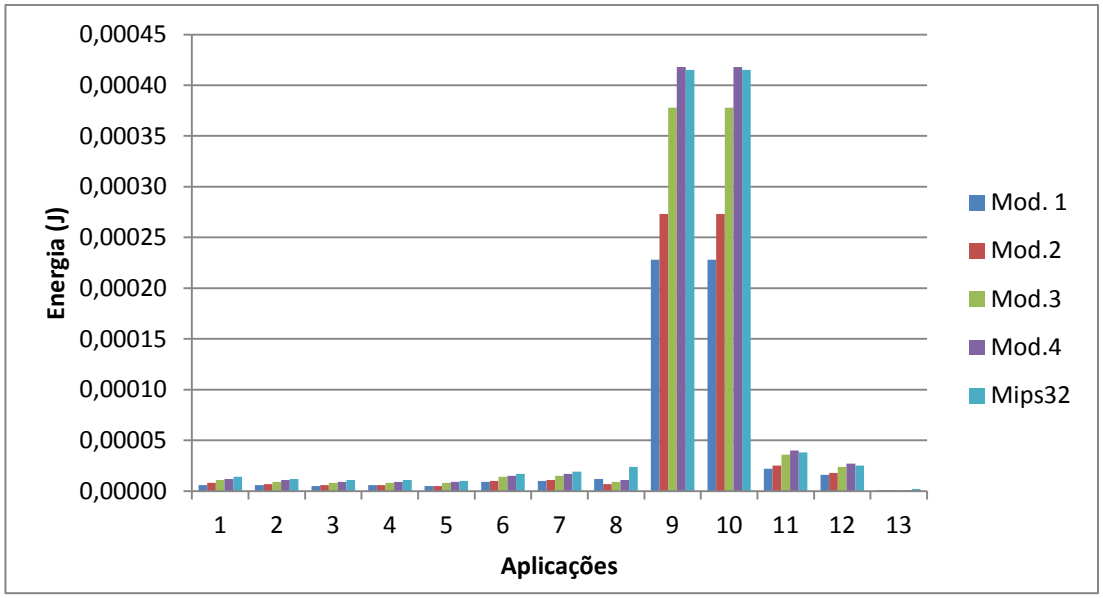


Figura 27. Consumo energético para 90 nm.

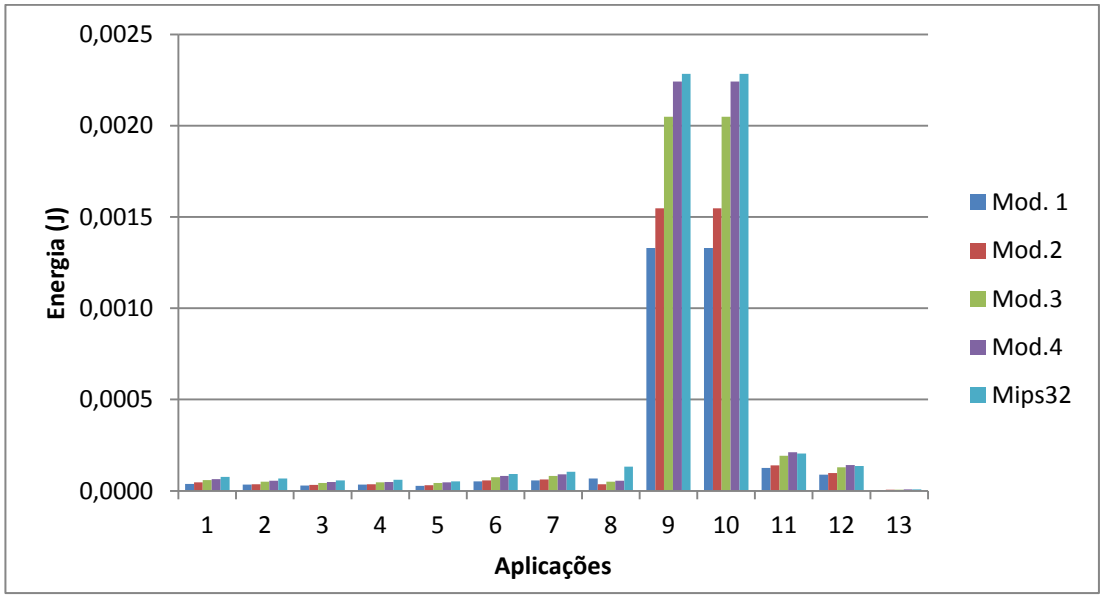


Figura 28. Consumo energético para 180 nm.

5.2.5 Potência dissipada

Embora o sistema tenha apresentado uma redução no consumo de energia, para os modelos 2, 3 e 4, sem o suporte à instruções MAC, a potência dissipada por todo o sistema aumentou. Como podemos ver na Figura 29, a potência aumentou consideravelmente, em ordem de 39% e 55% nos modelos 3 e 4. Também foi verificado um aumento de 1% no modelo 2. Porém, verificou-se que no modelo 1 houve uma redução de 14% em relação ao processador MIPS32 sem a UR acoplada. Já na Figura 30, podemos observar que os modelos 1 e 2 ficaram abaixo da potência consumida pelo processador, na média, na ordem de 13% e 2%, respectivamente e acima deste para os modelos 3 e 4, na ordem de 34% e 47%, respectivamente.

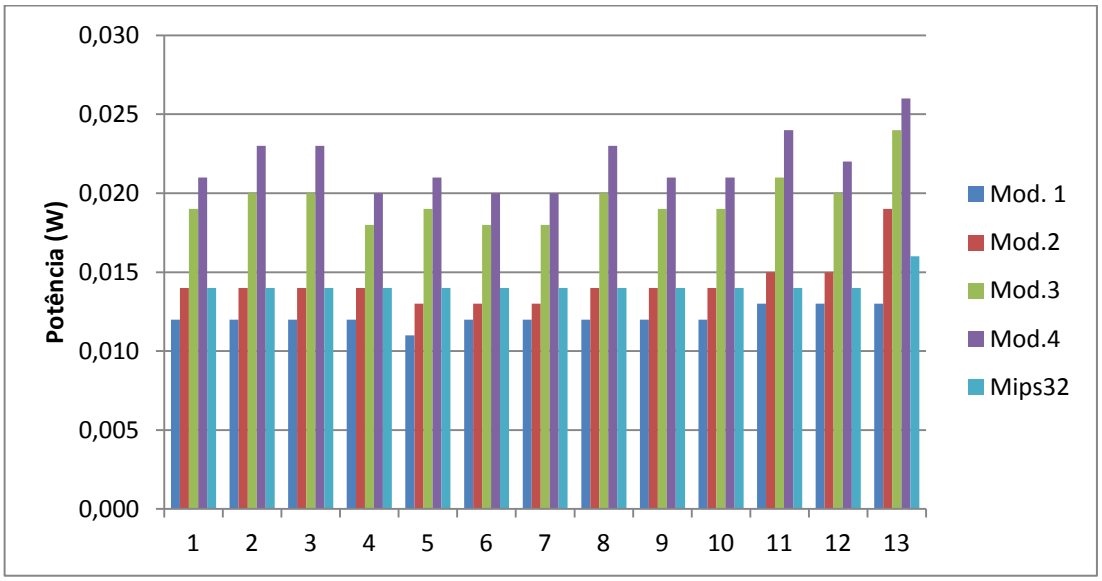


Figura 29. Potência consumida para 90 nm.

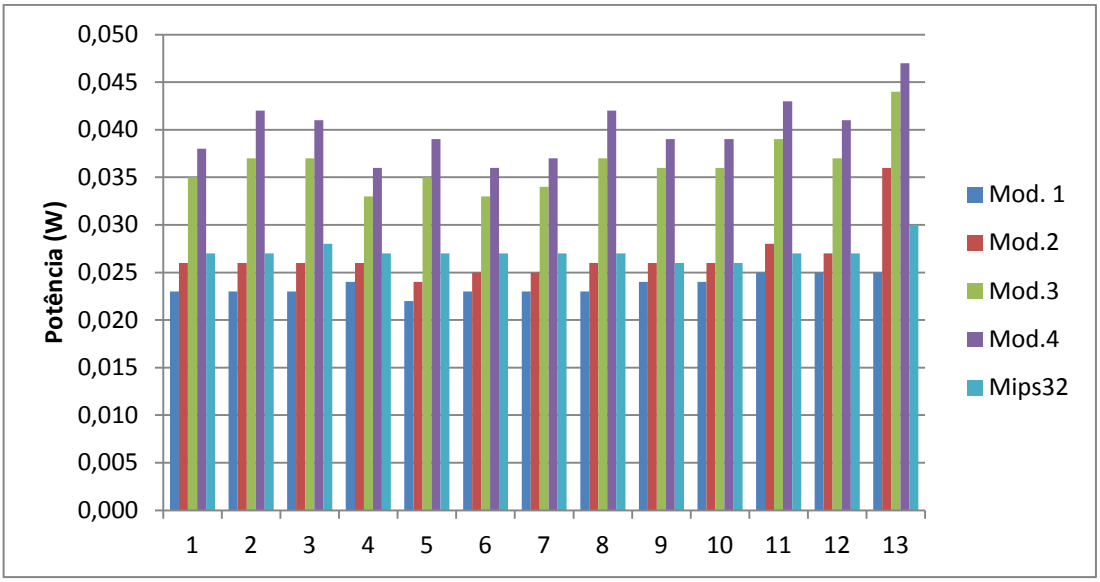


Figura 30. Potência consumida para 180 nm.

5.3 Comparação entre os diferentes cenários

Como mencionado no capítulo 4, os resultados da execução das aplicações selecionadas, foram obtidos através da simulação destas nos ambientes VisualDSP++, para os processadores Blackfin e SHARC, e OVPSim, para o processador MIPS32 e o simulador do Array Reconfigurável para os cenários que envolvem o sistema reconfigurável.

5.3.1 Resultados do cenário 1

O cenário 1 apresenta a comparação dos dois processadores comerciais DSP selecionados (das famílias Blackfin e SHARC) com um processador de propósito geral (GPP) que o caso é o MIPS32.

Os resultados da execução das aplicações em tempo de execução (segundos), para o cenário 1, podem ser vistos na Figura 31. Os resultados demonstram que, considerando um CPI=1 (segundo os manuais técnicos dos mesmos) e frequências de operação de 190 MHz, 400 MHz e 200 MHz para os processadores MPIS32 (0,18 um), Blackfin e SHARC, o processador MIPS32 executa as aplicações em um tempo médio de 0,004217 s, já o processador Blackfin em 0,00338 s e o processador SHARC em 0,00298 s.

As aplicações selecionadas foram executadas usando-se precisão dupla, como forma de se tentar manter uma comparação mais justa entre os

processadores, uma vez que os processadores MISP32 e Blackfin usam emulação para realizar operações com ponto flutuante. Já o processador SHARC possui uma FPU, o que fornece um alto desempenho de processamento. Porém, quando se usa precisão dupla, as operações são emuladas e a FPU não é utilizada. Isto conta como desvantagem na estratégia adotada pela maioria dos PDSPs em disponibilizar unidades especializadas para certos tipos de operações, pois uma vez que uma unidade está presente e não está sendo utilizada, o preço pago pela área e talvez e energia, estejam sendo desperdiçados.

Mesmo usando a emulação, o processador SHARC obteve melhor desempenho do que os outros processadores, neste primeiro cenário. O processador SHARC permite executar as instruções em dois modos diferentes: SISD e SIMD. Era esperado que as instruções executadas em modo SIMD gerassem um aumento ainda maior no desempenho em tempo de execução, porém isto não ocorreu. Verificou-se que para ativar o modo SIMD era necessário habilitar a função de otimização de software da ferramenta. Para realizar uma comparação justa, todas as otimizações de compilação das ferramentas utilizadas foram desabilitadas.

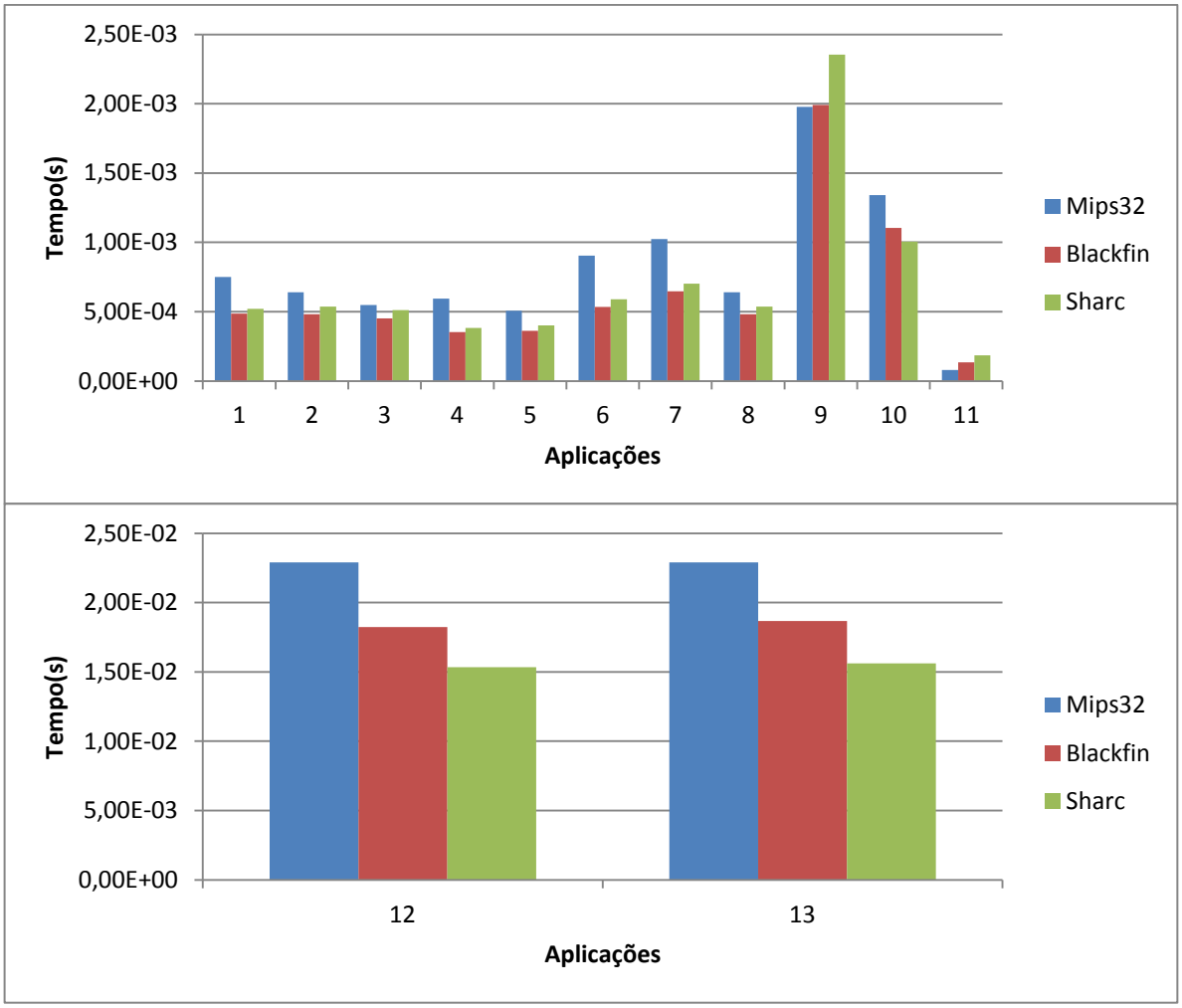


Figura 31. Desempenho em tempo de execução das aplicações selecionadas no cenário 1.

5.3.2 Resultados do cenário 2

No cenário 2 a unidade reconfigurável, descrita no capítulo 4, foi acoplada ao processador MIPS32 e as aplicações selecionadas foram executadas novamente para realizar a segunda comparação. A Figura 32 apresenta os resultados da execução das aplicações nos PDSPs comerciais e do processador MIPS32 com a UR acoplada. Podemos observar que o acoplamento da UR resultou em uma aceleração no tempo de execução, tornando o sistema mais rápido do que o processador SHARC. Esta aceleração foi na média de 1,2.

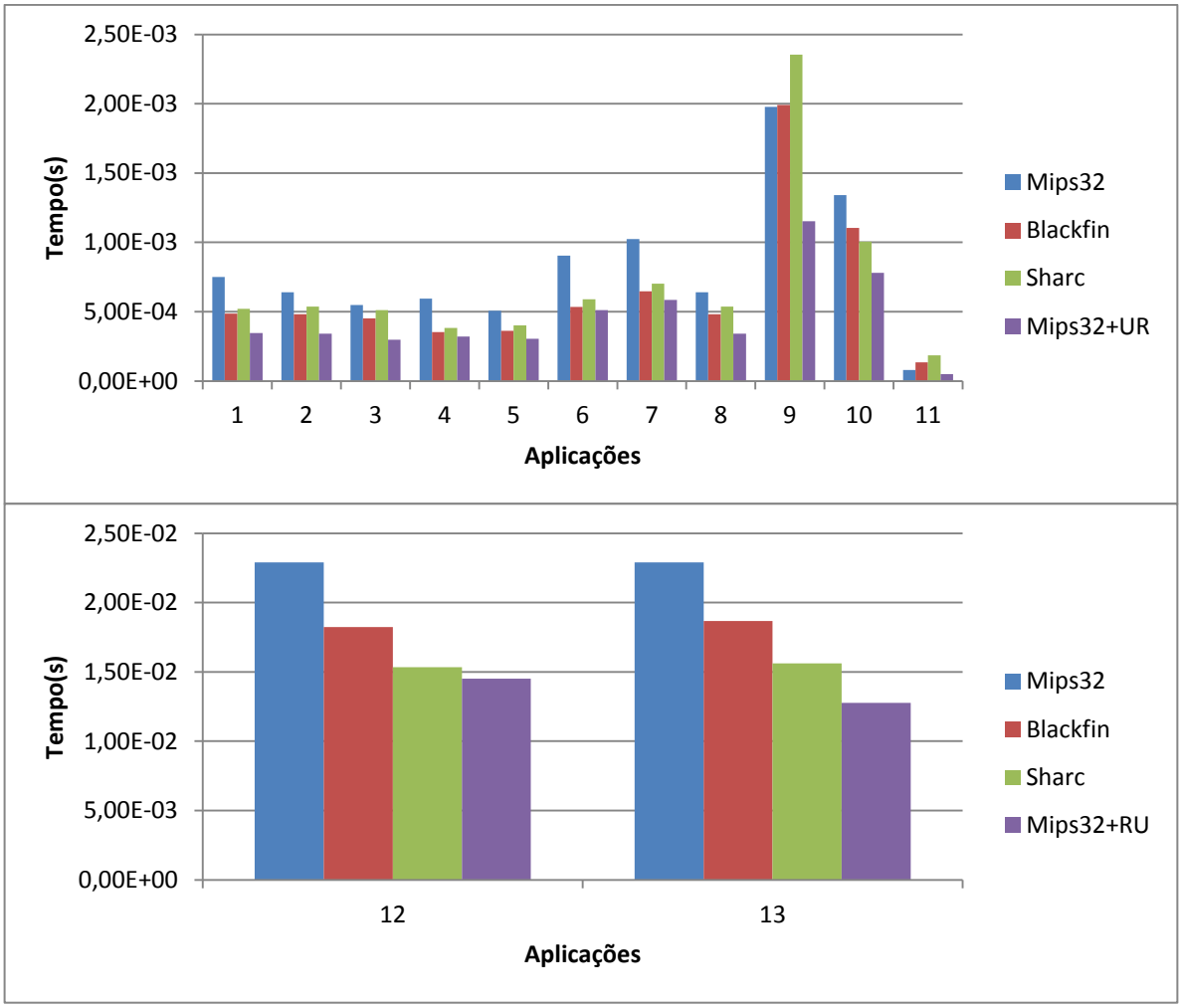


Figura 32. Desempenho em tempo de execução das aplicações selecionadas no cenário 2.

5.3.3 Resultados do cenário 3

Neste cenário foi adicionado ao sistema de reconfiguração o suporte à operações do tipo MAC (instruções *madd* e *maddu*), como mencionado na seção 4.4. Os resultados da execução das aplicações podem ser vistos na Figura 33, onde observamos que a aceleração média geral subiu para 1,9. Esta aceleração não foi superior, porque nem todas as aplicações se utilizaram das instruções MAC.

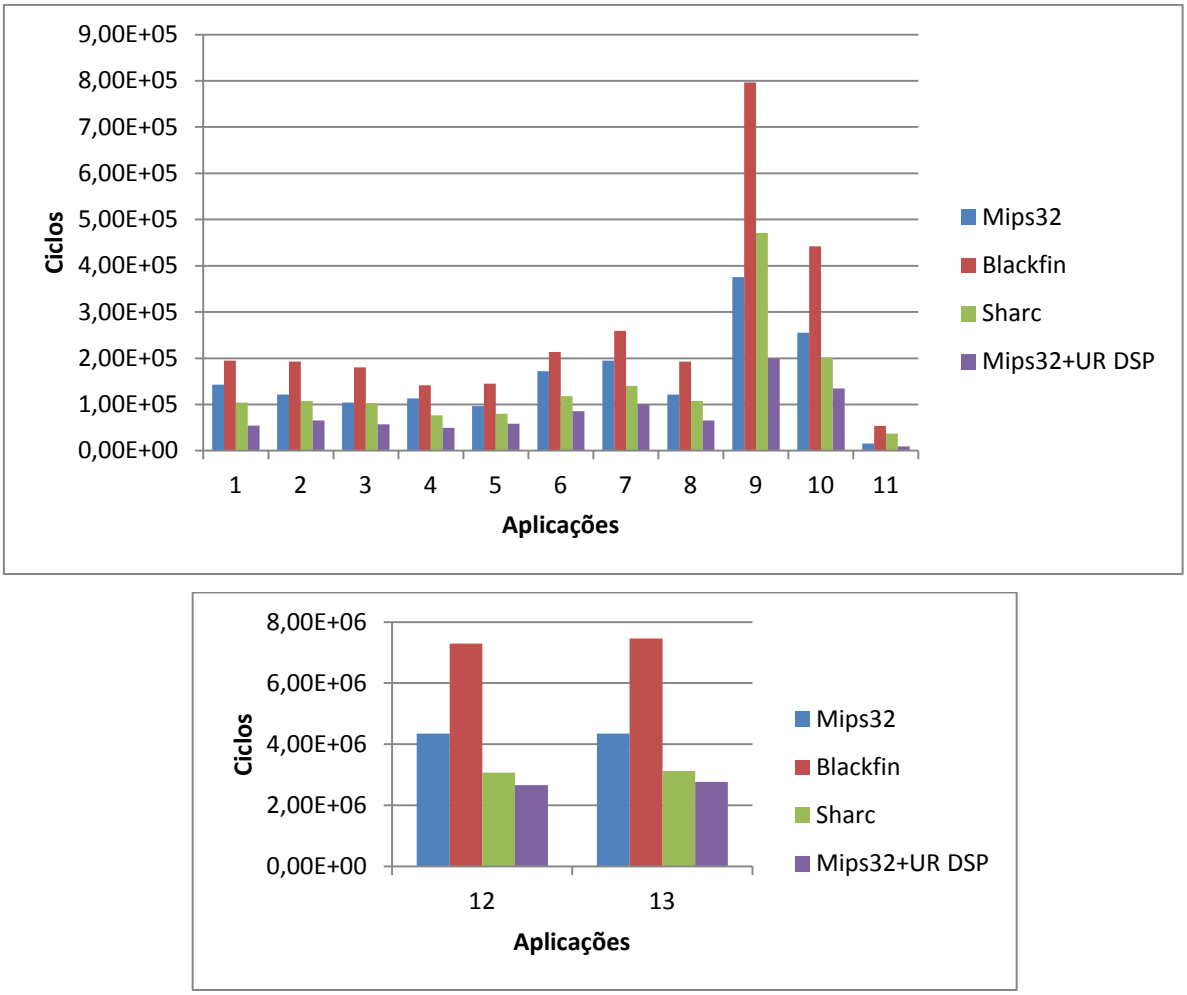


Figura 33. Desempenho em tempo de execução das aplicações selecionadas no cenário 3.

6. CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho apresentou a aplicação desta técnica através da acoplagem de um sistema reconfigurável em um processador MIPS32 e sua aplicabilidade em aplicações de processamento digital de sinais. O trabalhou apresentou as possibilidades de aceleração para aplicações DSP, utilizando um sistema reconfigurável proposto por (BECK, 2008). Como contribuições deste trabalho estão: a adaptação do tradutor binário para suporte ao MIPS32, adição de suporte à instruções MAC à Unidade Reconfigurável, disponíveis no MIPS32 e comparações em diferentes cenários envolvendo PDSPs comerciais e a Unidade Reconfigurável acoplada ao processador MIPS32.

Os resultados dos experimentos demostraram que a execução das aplicações foi acelerada atingindo tempos de execução inferiores aos PDSP comerciais. Adicionando suporte às instruções MAC ao tradutor binário a aceleração obtida aumentou. Além da aceleração obtida, os resultados também mostraram que houve economia em consumo energético. Com tais resultados, concluímos que é vantajoso o uso da técnica de reconfiguração para acelerar aplicações DSP. Como desvantagem do uso da técnica há o aumento da área, mas foi mostrada a vantagem do uso da configuração em tempo de projeto, o que fornece ao projetista a possibilidade de avaliar o sistema nas variáveis de interesse o que ajuda na escolha do tamanho da unidade reconfigurável.

Como trabalhos em andamento estão a análise de outras aplicações mais robustas (que continuam os algoritmos utilizado neste trabalho) como por exemplo, codificação e decodificação de áudio e vídeo, etc.

Como trabalho futuro está prevista a avaliação de um modelo energético para os PDSPs através de placas de prototipação e/ou elaboração de um modelo energético. Também se pretende propor a inserção de novas unidades especializadas (DSP) no sistema reconfigurável, como por exemplo, buffer circular e bit reverse. Poderia ser adicionado o suporte à aritmética de ponto flutuante, porém estas alterações demandariam atualizar o tradutor binário para outro processador MIPS que tenha suporte à tais instruções.

Outra possibilidade é atualizar o tradutor binário para o ISA do ARM. Esta é a família de processadores mais utilizada pelo mercado e seria interessante verificar o potencial de aceleração que poderia ser alcançado com a reconfiguração.

Referências

ALTERA. Altera. **Site da Altera**, 2014. Disponivel em: <http://www.altera.com/devices/processor/mips/mp32/proc-mp32.html>. Acesso em: 24 Março 2014.

ANALOG. **ANALOG**, 2013a. Disponivel em: <http://www.analog.com/en/processors-dsp/products/index.html>. Acesso em: 22 Julho 2012.

ANALOG. **SHARC Processors**, 2013b. Disponivel em: <http://www.analog.com/en/processors-dsp/sharc/products/index.html>. Acesso em: 22 Julho 2012.

ANALOG. **Blackfin Processors**, 2013c. Disponivel em: <http://www.analog.com/en/processors-dsp/blackfin/products/index.html>. Acesso em: 22 Julho 2012.

ANALOG. **Analog Devices**, 2013d. Disponivel em: <http://www.analog.com/en/processors-dsp/blackfin/vdsp-bf-sh-ts/products/product.html>. Acesso em: 22 Julho 2012.

BECK, A. C. S. **Transparent Reconfigurable Architecture for Heterogeneous Applications**. Instituto de Informática, UFRGS. Porto Alegre, p. 188f. 2008.

BECK, F. A.; CARRO, L. **Dynamic Reconfigurable Architectures and Transparent Optimization Techniques**. [S.l.]: Springer, 2010.

C. EBELING, D. C. A. P. F. Configurable computing:The catalyst for high-performance architectures. **Proc. IEEE Int.Conf. Appl.-Specific Syst., Arch., Process**, Julho 1997. 364–372.

CLARK, J. The Register. **The Register**, 2013. Disponivel em: <http://www.theregister.co.uk/2013/04/23/amd_gseries_embedded_chips/>. Acesso em: 8 Janeiro 2014.

CLARKE, P. **EE Times**, 2013. Disponivel em: <http://www.eetimes.com/document.asp?doc_id=1318969>. Acesso em: 8 Janeiro 2014.

DEHON, S. H. A. A. **Reconfigurable Computing Thetheory and practice Of FPGA-Based computation**. 1ª Edição. ed. Burlington: Elsevier, 2007.

FERDOUS, T. Design and FPGA-based implementation of a high performance 32-bit DSP processor. **Computer and Information Technology (ICCIT), 2012 15th International Conference on**, Dezembro 2012. vol., no., pp.484,489, 22-24.

FREESCALE. **freescale**, 2012. Disponivel em: <http://www.freescale.com/>. Acesso em: 22 Julho 2012.

FREESCALE. **freescale**, 2013. Disponivel em: <http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=SYMPH_STUDIO>. Acesso em: 22 Julho 2012.

FREESCALE. DSP56K/Symphony Digital Signal Processors. **Freescale**, 2014. Disponivel em: <http://www.freescale.com/webapp/sps/site/homepage.jsp?code=563XXGPDSP>. Acesso em: 09 Março 2014.

FREESCALE. Freescale. **Freescale**, 2014. Disponivel em: <http://www.freescale.com/webapp/sps/site/homepage.jsp?code=563XXGPDSP>. Acesso em: 11 fevereiro 2014.

G. STITT, F. V. The Energy Advantages of Microprocessor Platforms with On-Chip Configurable Logic. **IEEE Design and Test of Computers**, Los Alamitos, Novembro 2002. v. 19, n. 6, p. 36 – 43.

H. SINGH, M.-H. L. G. L. F. J. K. N. B. E. M. C. F. MorphoSys: An integrated reconfigurable system for data-parallel and computation-intensive applications. **IEEE Trans.Comput.**, v. 49, nº 5, p. 465–481, Maio 2000.

HENKEL, J. A low power hardware/software partitioning approach for core-based embedded systems. **Design Automation Conference, DAC**, New York: ACM Press 2005, 1999. 122-127.

IHS iSuppli's. **IHS iSuppli's**, 2013. Disponivel em: <http://www.isuppli.com/Mobile-and-Wireless-Communications/Pages/Competition-intensifies-among-handset-OEMs.aspx?utm_source=iSi&utm_medium=MW&utm_campaign=070813>. Acesso em: 8 janeiro 2014.

INSTRUMENTS, T. Texas Instruments. **Texas Instruments**, 2014c. Disponivel em: <http://www.ti.com/lsds/ti/dsp/overview.page>. Acesso em: 25 Fevereiro 2014.

J. E. CARRILLO, E. P. C. The effect of reconfigurable units in superscalar processors. **Proceedings of the Ninth ACM International Symposium on Field-Programmable Gate Arrays**, Fevereiro 2001.

JAGADEESH, P.; RAVI, S.; MALLIKARJUN, K. H. Design of high performance 64 bit MAC unit. **Design of high performance 64 bit MAC unit**, 2013. vol., no., pp.782,786, 20-21.

JAMES W. TSCHANZ, S. G. N. Y. Y. B. A. B. S. B. V. D. Dynamic Sleep Transistor and Body Bias for Active Leakage Power Control of Microprocessors. **IEEE JOURNAL OF SOLID-STATE CIRCUITS**, Novembro 2003. VOL. 38, NO. 11, 1838-1845.

KIM, N. S. et al. Leakage current: Moore's law meets static power. **Computer 36(12)**, p. 65-73, 2003.

LAMBERTI, F. et al. Reducing the Computation Time in (Short Bit-Width) Two's Complement Multipliers. **Computers, IEEE Transactions on**, Fevereiro 2011. vol.60, no.2, pp.148,156.

LAPSLEY, P. et al. **DSP Processor Fundamentals-Architectures and Features**. New York: IEEE Press, 1997.

M. WAN, E. A. An Energy Conscious Methodology for Early Design Space Exploration of Heterogeneous DSPs. **Custom Integrated Circuits Conference**, Santa Clara, 1998. 111-117.

MARWEDEL, P. **Embedded Systems Design**. Dordrecht: Springer, 2006.

MICROSOFT. **Microsoft**, 2011. Disponivel em: <http://www.microsoft.com/en-us/news/features/2011/oct11/10-27EmbeddedDYK.aspx>. Acesso em: 15 Maio 2013.

MIPS32. **MIPS32**, 2012. Disponivel em: <http://www.mips.com/products/architectures/mips32/>. Acesso em: 22 Julho 2012.

OLUKOTUN, T. M. A. K. A quantitative analysis of reconfigurable coprocessors for multimedia applications. **Proc. IEEE Symp. FPGAs for Custom Comput.**, Março 1998. 15–17.

OVP. **Open Virtual Platforms**, 2013. Disponivel em: <http://www.ovpworld.org/>. Acesso em: 2 Janeiro 2013.

R. LYSECKY, G. S. F. V. Warp Processors. **ACM Transactions onDesign Automation of Electronic Systems**, New York, Julho 2006. v.11, n. 3, p. 659 – 681.

R. RAZDAN, M. S. A high-performance microarchitecture with hardware programmable functional units. **Proceedings of the 27th Annual IEEE/ACM International Symposium on Microarchitecture**, Novembro 1994.

RANA, V.; SANTAMBROGIO, M.; SCIUTO, D. Dynamic Reconfigurability in Embedded System Design. **ISCAS 2007. IEEE International Symposium on**, New Orleans, LA, 2007.

RUTZIG, M. B. **Gerenciamento Automático de Recursos Reconfiguráveis Visando a Redução de Área e Consumo de Potência em Dispositivos Embarcados**. UFRGS. Porto Alegre, p. 112. 2008.

S. HAUCK, T. W. F. M. H. J. P. K. The Chimaera reconfigurable functional unit. **Proceedings of the IEEE Symposium on Field-Programmable Custom Computing Machines**, Abril 1997.

SOLID State Technology. **Solid State Technology**, 2013. Disponivel em: <http://electroiq.com/blog/2013/04/health-awareness-spurs-quadrupling-in-mems-sensor-market-for-wea/>. Acesso em: 8 janeiro 2014.

TEXAS. **Texas Instruments**, 2013. Disponivel em: <http://www.ti.com/tool/ccstudio>. Acesso em: 22 Julho 2012.

TEXAS. Texas Instruments. **Texas Instruments**, junho 2013. Disponivel em: <http://www.ti.com/lit/ds/sprs586f/sprs586f.pdf>. Acesso em: 11 fevereiro 2014.

TEXAS. **Texas Instruments**, 2013a. Disponivel em: <http://www.ti.com/tool/ccstudio>. Acesso em: 22 Julho 2012.

TEXAS. Texas Instruments. **Texas Instruments**, junho 2013b. Disponivel em: <http://www.ti.com/lit/ds/sprs586f/sprs586f.pdf>. Acesso em: 11 fevereiro 2014.

Y. CHOU, P. P. H. S. A. J. P. S. “PipeRench Implementation of the Instruction Path Coprocessor. **Proc. Ann. IEEE/ACM**, Dezembro 2000. 147-158.

YOONJIN KIM, R. N. M. I. P. A. K. C. Low power reconfiguration technique for coarse-grained reconfigurable architecture. **EEE Trans. Very Large Scale Integr. Syst.**, Maio 2009. 593-603.

YOONJIN KIM, R. N. M. I. P. K. C. Low Power Reconfiguration Technique for Coarse-Grained Reconfigurable Architecture. **IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS**, Maio 2009. VOL. 17, NO. 5, 593-603.

APENDICE A - Resultados em ciclos de execução das aplicações

Tabela 6. Número de ciclos de execução das aplicações.

Aplicação	MIPS32	Blackfin	SHARC	Array DSP
1	142788	194956	104076	54499
2	121763	192892	107600	65394
3	104380	180617	102491	57163
4	112948	141722	76445	49582
5	96563	144978	80103	58135
6	171965	213558	118021	85470
7	194585	259383	140448	99941
8	121763	192892	107600	65394
9	375578	796573	470816	198719
10	254761	442220	201264	134794
11	15356	53711	37317	9234
12	4351925	7297239	3068560	2663357
13	4351925	7464755	3122085	2768400

Tabela 7. Número de ciclos de execução das aplicações nos modelos propostos.

Aplicação	Modelo 1	Modelo 2	Modelo 3	Modelo 4
1	65831	76072	76072	76072
2	64871	63584	63584	63584
3	56636	55345	55345	55345
4	61119	60562	60562	60562
5	57891	57615	57615	57615
6	97155	96339	96339	96339
7	111128	110122	110122	110122
8	64871	63584	63584	63584
9	218868	217223	217223	217097
10	148462	147346	147346	147261
11	9520	9427	9421	9415
12	2757874	2731905	2728480	2728480
13	2427175	2235195	2235195	2235195

APENDICE B - Resultados de consumo energético da execução das aplicações

Tabela 8. Dados de potência (Watts) na tecnologia 90 nm.

Aplicação	Modelo1	Modelo2	Modelo3	Modelo4	Mips32
1	0,012	0,014	0,019	0,021	0,014
2	0,012	0,014	0,020	0,023	0,014
3	0,012	0,014	0,020	0,023	0,014
4	0,012	0,014	0,018	0,020	0,014
5	0,011	0,013	0,019	0,021	0,014
6	0,012	0,013	0,018	0,020	0,014
7	0,012	0,013	0,018	0,020	0,014
8	0,012	0,014	0,020	0,023	0,014
9	0,012	0,014	0,019	0,021	0,014
10	0,012	0,014	0,019	0,021	0,014
11	0,013	0,015	0,021	0,024	0,014
12	0,013	0,015	0,020	0,022	0,014
13	0,013	0,019	0,024	0,026	0,016

Tabela 9. Dados de potência (Watts) na tecnologia 180 nm.

Aplicação	Modelo1	Modelo2	Modelo3	Modelo4	Mips32
1	0,023	0,026	0,035	0,038	0,027
2	0,023	0,026	0,037	0,042	0,027
3	0,023	0,026	0,037	0,041	0,028
4	0,024	0,026	0,033	0,036	0,027
5	0,022	0,024	0,035	0,039	0,027
6	0,023	0,025	0,033	0,036	0,027
7	0,023	0,025	0,034	0,037	0,027
8	0,023	0,026	0,037	0,042	0,027
9	0,024	0,026	0,036	0,039	0,026
10	0,024	0,026	0,036	0,039	0,026
11	0,025	0,028	0,039	0,043	0,027
12	0,025	0,027	0,037	0,041	0,027
13	0,025	0,036	0,044	0,047	0,030

Tabela 10. Dados de energia (Joules) na tecnologia 90 nm.

Aplicação	Modelo1	Modelo2	Modelo3	Modelo4	Mips32
1	0,000006	0,000008	0,000011	0,000012	0,000014
2	0,000006	0,000007	0,000009	0,000011	0,000012
3	0,000005	0,000006	0,000008	0,000009	0,000011
4	0,000006	0,000006	0,000008	0,000009	0,000011
5	0,000005	0,000005	0,000008	0,000009	0,000010
6	0,000009	0,000010	0,000014	0,000015	0,000017
7	0,000010	0,000011	0,000015	0,000017	0,000019
8	0,000012	0,000007	0,000009	0,000011	0,000024
9	0,000228	0,000273	0,000378	0,000418	0,000415
10	0,000228	0,000273	0,000378	0,000418	0,000415
11	0,000022	0,000025	0,000036	0,000040	0,000038
12	0,000016	0,000018	0,000024	0,000027	0,000025
13	0,000001	0,000001	0,000001	0,000001	0,000002

Tabela 11. Dados de energia (Joules) na tecnologia 180 nm.

Aplicação	Modelo1	Modelo2	Modelo3	Modelo4	Mips32
1	0,0000380	0,0000460	0,0000590	0,0000640	0,0000770
2	0,0000340	0,0000370	0,0000510	0,0000560	0,0000670
3	0,0000290	0,0000320	0,0000440	0,0000490	0,0000580
4	0,0000340	0,0000360	0,0000460	0,0000490	0,0000610
5	0,0000280	0,0000310	0,0000430	0,0000470	0,0000520
6	0,0000520	0,0000570	0,0000740	0,0000810	0,0000920
7	0,0000570	0,0000620	0,0000820	0,0000900	0,0001040
8	0,0000670	0,0000370	0,0000510	0,0000560	0,0001330
9	0,0013300	0,0015470	0,0020490	0,0022410	0,0022840
10	0,0013300	0,0015470	0,0020490	0,0022410	0,0022840
11	0,0001250	0,0001400	0,0001920	0,0002120	0,0002050
12	0,0000880	0,0000980	0,0001300	0,0001420	0,0001370
13	0,0000050	0,0000060	0,0000070	0,0000080	0,0000090