Utilizando algoritmo genético para exploração do espaço de projeto em sistemas multiprocessados embarcados

Alexandre Specian Cardoso[1]
Marcio Seiji Oyamada[1]

Resumo: A crescente complexidade dos sistemas embarcados requer o desenvolvimento de novas metodologias e ferramentas para apoiar o projeto. Um exemplo de ferramenta para apoio do projeto de sistemas embarcados são as plataformas virtuais, que são modelos de simulação visando facilitar a avaliação de arquiteturas quanto ao desempenho e ao consumo de potência, entre outros requisitos de um sistema embarcado. Para diminuir o tempo de exploração do espaço de projeto, métodos heurísticos podem ser utilizados para diminuir o número de simulações necessárias. Esse trabalho apresenta o desenvolvimento de um ambiente para modelagem visual e exploração do espaço de projeto de arquiteturas multiprocessadas. O ambiente utiliza um algoritmo genético para permitir uma rápida exploração do espaço de projeto e a otimização da configuração base definida pelo usuário. Um estudo de caso de um decodificador JPEG paralelo foi utilizado e os resultados demonstraram que o algoritmo genético permite minimizar o número de arquiteturas simuladas quando comparado com a simulação exaustiva, além de obter arquiteturas com melhor compromisso entre desempenho e consumo de potência.

Palavras-chave: Exploração do espaço de projeto. Multiprocessador. Sistemas embarcados.

Abstract: *The increasing complexity in the design of embedded systems calls for new tools and methodologies. As instance, virtual platforms are proposed to provide an early simulation model of the system, easing the architecture evaluation in terms of performance, power consumption and other requirements. In order to minimize the design exploration time, heuristics can be used to optimize the architecture and decrease the number of simulation runs. In this work, an environment for visual modeling and design exploration for multiprocessor embedded systems is proposed. The environment uses a genetic algorithm to provide a fast design exploration and architecture optimization. A case study of a parallel JPEG decoder was developed and the results showed that the use of genetic algorithm allowed the reduction of simulation runs compared to exhaustive simulation, and also provided architectures with best tradeoff between performance and power consumption.*

Keywords: Design space exploration. Embedded systems. Multiprocessor.

1 Introdução

Atualmente, é comum se deparar com aparelhos eletrônicos com algum processador embarcado, denominados Sistemas Embarcados (SE). Segundo Marwedel [1], um SE é um sistema de processamento de informações que é incorporado em um produto maior e, normalmente, não é diretamente visível pelo usuário. Como exemplos de sistemas embarcados podem ser citados, dentre outros, os sistemas aviônicos e os eletrônicos.

O desenvolvimento de hardware e software para SE possui algumas diferenças em relação ao desenvolvimento de sistemas computacionais de propósito geral. Ao se desenvolver um SE, deve-se pensar na

[1] Curso de Ciência da Computação, UNIOESTE, Campus Cascavel – Rua: Universitária, 2069-Cascavel (PR) - Brasil
{alexandre.cardoso, marcio.oyamada @unioeste.br}

http://dx.doi.org/10.5335/rbca.2012.2080

arquitetura que será utilizada, pois esta apresenta requisitos restritos de desempenho, potência, custo de produção, design, tolerância a falhas e tempo de projeto.

O desempenho está diretamente relacionado ao requisito de potência, uma vez que processadores com maior desempenho tendem a consumir mais potência. Esse é um atributo chave em um SE, pois a grande maioria desses dispositivos é alimentada por baterias. Assim, um grande consumo de potência resultará em um grande consumo de energia, necessitando a recarga constante, fato incômodo em alguns aparelhos como celulares e *players*. Entre os elementos de um processador, a cache tem um impacto importante no consumo de potência, sendo responsável, em alguns casos, por mais de 50% do consumo de potência do processador [2]. Além da cache, outros elementos da arquitetura são influenciados pelo tipo da aplicação e têm impacto direto no desempenho e consumo de potência, tais como número de processadores e unidades funcionais. Dessa forma, no desenvolvimento de um sistema embarcado deve-se considerar a combinação otimizada dos componentes de hardware e software.

Diante das inúmeras possibilidades é necessário explorar o espaço de projeto visando obter a melhor configuração da arquitetura de hardware para uma determinada aplicação, principalmente para projetos com processadores de alto desempenho em SE. Tal fase auxilia os projetistas na detecção e resolução de problemas decorrentes do projeto da arquitetura. Segundo Carro e Wagner [3], a fase de exploração do espaço de projeto deve encontrar a solução para três questões: 1) Quantos e quais processadores e blocos dedicados serão necessários? 2) Qual o mapeamento ideal entre funções e componentes de hardware? e 3) Qual a estrutura de comunicação ideal para conectar os componentes entre si? Considerando os requisitos necessários para um SE, pode-se fazer uso de estimadores que tenham condições de informar, com certo grau de precisão, o comportamento do sistema, estimando dados como desempenho e consumo de potência. No intuito de facilitar a exploração do espaço de projeto, plataformas virtuais são utilizadas para simular as arquiteturas em um estado inicial de projeto e, consequentemente, obter uma estimativa de desempenho antes que o hardware seja construído. No entanto, mesmo com o uso de estimadores o tempo de exploração de espaço de projeto pode ser elevado, visto o grande número de soluções possíveis. Uma das formas de minimizar esse problema é utilizar métodos heurísticos para auxiliar a escolha de soluções candidatas. Neste trabalho é proposto o uso de algoritmos genéticos para esse fim.

Este trabalho tem como primeiro objetivo facilitar a exploração do espaço de projeto de um SE através de um ambiente visual para a modelagem de arquiteturas multiprocessadas, que possibilitará a simulação, utilizando a ferramenta VIPRO-MP[4]. O segundo objetivo visa a utilização de métodos heurísticos, neste caso um Algoritmo Genético (AG), que será acoplado à plataforma visual desenvolvida, possibilitando a otimização da arquitetura proposta pelo projetista com redução do tempo de exploração do espaço de projeto. O ambiente proposto neste trabalho é denominado VIPEX-VMP (Virtual Platform Exploration - VIPRO-MP). Apesar de outros trabalhos explorarem o uso de algoritmos genéticos para exploração do espaço de projeto, como será apresentado na Seção 2, o ambiente proposto integra modelagem visual, facilitando a construção da arquitetura base, e a exploração do espaço de projeto com múltiplos processadores, inclusive possibilitando a configuração de cada processador com parâmetros diferentes.

Este trabalho é organizado da seguinte forma: a Seção 2 apresenta os trabalhos relacionados, a Seção 3 apresenta o modelo VIPEX-VMP e o processo de exploração de espaço de projeto. Na Seção 4 apresenta o estudo de caso desenvolvido, e a Seção 5 apresenta as conclusões e trabalhos futuros.

2 Trabalhos relacionados

O uso de protótipos virtuais possibilita a exploração do espaço de projeto, otimizando as arquiteturas e consequentemente diminuindo o tempo de projeto. Essas plataformas têm o poder de simular as arquiteturas e mostrar estimativas de desempenho, potência consumida entre outras características que o hardware irá possuir.

O ConvergenSC[5] , da CoWare, é uma ferramenta para projeto de SE baseada em plataformas de hardware. Sua biblioteca possui os modelos dos processadores ARC e MIPS e também modelos de barramento como o AMBA. Caso o projetista necessite de um componente que não consta na biblioteca é possível criá-lo, utilizando o LISATek [6], e adicioná-lo à biblioteca na forma de um componente.

A base do ConvergenSC é o SystemC[7], que possui recursos de modelagem de componentes de hardware e software, controle da simulacão e análise de sinais. Seus recursos permitem a realização de tarefas

como otimização de arquiteturas SoC [8]. O ConvergenSC possui interface gráfica para modelagem de plataformas, facilitando a tarefa de montar, simular e avaliar arquiteturas. Entretanto, a exploração é realizada manualmente, dificultando e elevando o tempo necessário para realizar essa atividade.

O Open Virtual Platform (OVP) [9] é uma plataforma virtual de código aberto, flexível e gratuito, que permite a criação do modelo de simulação de arquiteturas, inclusive com suporte para multiprocessadores. Outra vantagem do OVP é a possibilidade de conectar o simulador ao depurador gdb, facilitando o desenvolvimento de software embarcado. O OVP é apoiado por um conjunto de empresas fabricantes de processadores embarcados que disponibilizam modelos de simulação para o ambiente. Dessa forma é possível criar plataformas virtuais com modelos de processadores validados pelo fabricante. No entanto, o ambiente não provê interface gráfica e muito menos métodos heurísticos para exploração do espaço de projeto.

Este trabalho utiliza o Virtual Prototype for Multiprocessor Architectures (VIPRO-MP) [4], uma plataforma virtual para plataformas multiprocessadas. O ambiente utiliza a linguagem SystemC para modelagem de componentes de hardware. Cada núcleo é um simulador SimpleScalar encapsulado em SystemC. A configuração de cada núcleo SimpleScalar mantém a sintaxe de configuração original, facilitando o reuso de configurações já testadas. Cada núcleo pode ser configurado em termos de tamanho de caches, número de unidades funcionais etc. Essa característica possibilita um grande grau de liberdade, podendo o projetista definir elementos microarquiteturais de cada núcleo e avaliar o impacto global no desempenho do sistema.

Visando minimizar o tempo de exploração do espaço de projeto para avaliação de arquiteturas, métodos heurísticos são frequentemente utilizados. Métodos como os Algoritmos Genéticos (AG) são capazes de otimizar o número de arquiteturas simuladas pela plataforma. Como exemplo, pode-se citar o Platune [10], que é uma plataforma de exploração de espaço de projeto que permite a variação de 26 parâmetros de uma plataforma composta por um processador e um bloco de hardware dedicado para cálculo DCT. O Platune utiliza AG para otimizar o número de arquiteturas simuladas. Outro exemplo da utilização AG na otimização de arquitetura embarcada pode-se citar o algoritmo NSGAII [11], que visa otimizar a memória cache de um processador executando uma dada aplicação.

Este trabalho também utiliza um AG para auxiliar na exploração do espaço de projeto, no entanto, o ambiente possibilita a configuração de múltiplos processadores, além de permitir a variação também no número de unidades funcionais de cada processador. Assim, é possivel determinar se a utilização de multiplos núcleos com diferentes configurações, mas com um mesmo conjunto de instruções, possibilita otimizar a execução de uma dada aplicação.

3 VIPEX-VMP (Virtual Platform Exploration - VIPRO-MP)

O VIPEX-VMP (Virtual Platform Exploration – VIPRO-MP) desenvolvido neste trabalho tem como objetivo prover uma plataforma visual para exploração de espaço de projeto visando otimizar arquiteturas em termos de desempenho (ciclos para execução) e consumo de potência. O ambiente é baseado no simulador para arquiteturas multiprocessadas denominado VIPRO-MP. Nas subseções seguintes são descritos a plataforma virtual VIPRO-MP, o ambiente de modelagem visual de arquiteturas e o módulo de exploração de espaço de projeto.

3.1 VIPRO-MP

O VIPRO-MP (Virtual Prototype for Multiprocessor Architectures) [4] é um ambiente desenvolvido para a simulação de plataformas multiprocessadas, que utiliza o SimpleScalar como simulador do processador, o framework para cálculo de potência Wattch (Sim-Wattch) e a linguagem SystemC [7] para modelar os demais componentes de hardware .

O modelo arquitetural implementado no VIPRO-MP permite a execução de aplicações descritas em linguagem C e compiladas para conjunto de instrução PISA, semelhante à arquitetura MIPS. Cada processador possui uma memória de instruções e de dados privada, além da memória compartilhada, sendo essa mapeada em um conjunto de endereços predefinido. A comunicação entre os processadores é realizada exclusivamente através da memória compartilhada.

Os parâmetros que configuram cada processador mantém a sintaxe do SimpleScalar, fato que facilita a utilização do VIPRO-MP. A Figura 1 demonstra como o VIPRO-MP está organizado, sendo possível a inclusão de módulos para estender a plataforma virtual.

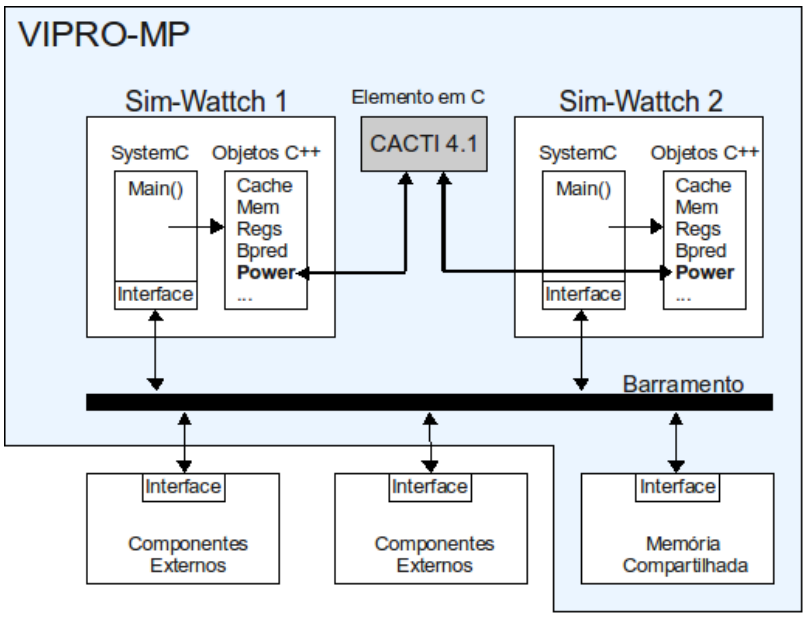


Figura 1: Exemplo de arquitetura modelada no VIPRO-MP

Um dos problemas no VIPRO-MP é que todo o processo de montagem da arquitetura é feito a partir de códigos C++, dificultando a exploração do espaço de projeto. Um dos objetivos deste trabalho visa sanar esse problema por meio de uma interface visual para modelagem de arquiteturas.

3.2 Ambiente visual para modelagem

Para o desenvolvimento do ambiente visual, foi utilizada a linguagem Java. O ambiente desenvolvido visa tornar a tarefa de construção e simulação de arquiteturas mais simples e interativa. Para montar uma arquitetura, basta que o projetista selecione os componentes e os coloque na tela em suas respectivas posições, podendo também conectá-los com facilidade, como apresentado na Figura 2(a).

O VIPEX-VMP permite também que o projetista altere as propriedades dos componentes utilizados, como demonstra a Figura 2(b). A definição do software embarcado que irá executar em cada processador é feita na tela de propriedades do próprio processador, na guia “Arquivo”.

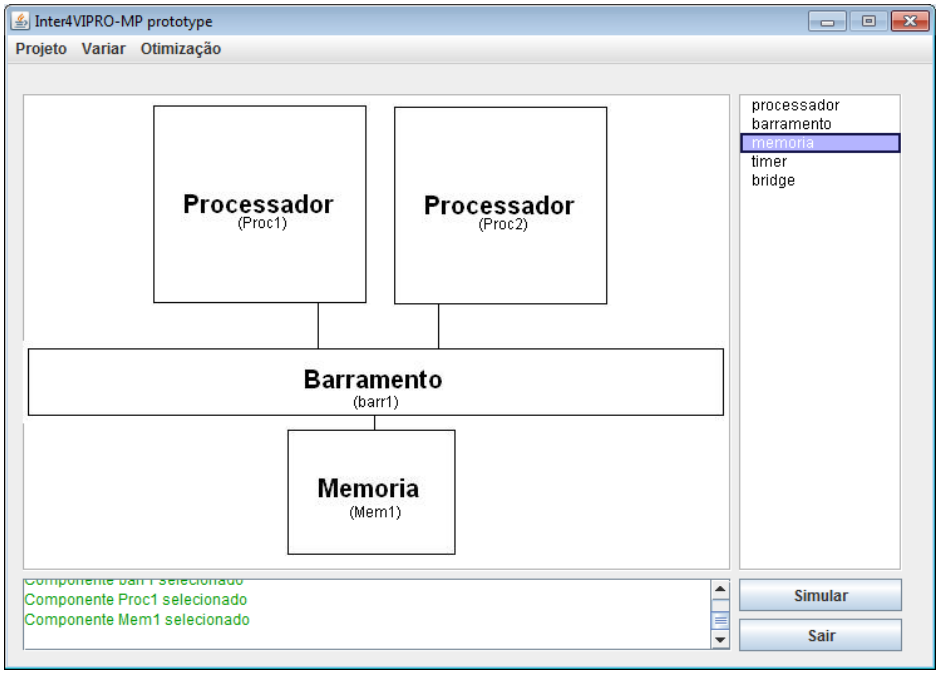
Como a simulação das arquiteturas é realizada pela plataforma VIPRO-MP, o VIPEX-VMP gera automaticamente, a partir da arquitetura modelada pelo projetista, os seguintes arquivos:

a) arquivo em linguagem C++ contendo a declaração, instanciação e conexão dos componentes da arquitetura;

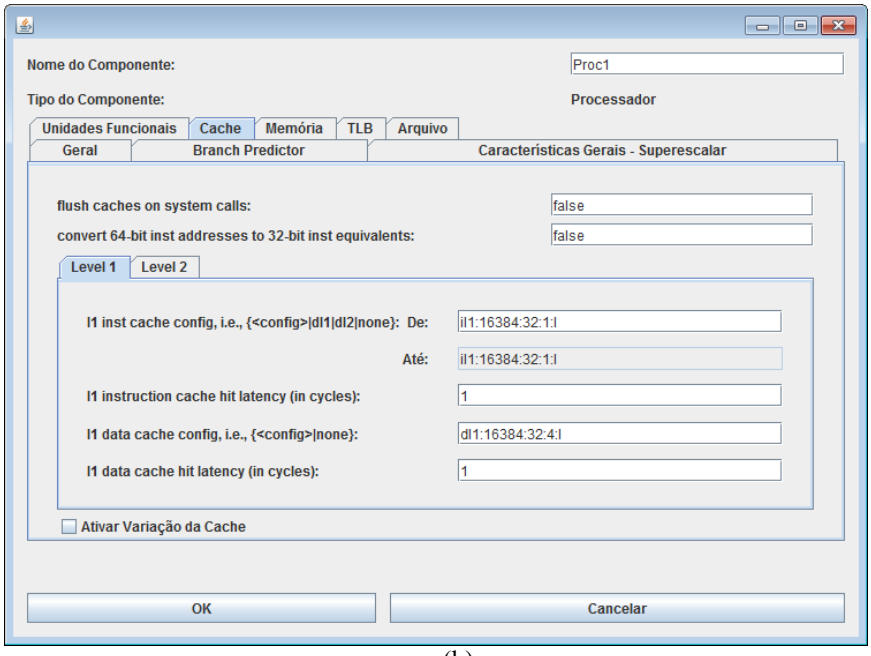
b) arquivos de configuração dos processadores contendo os parâmetros configurados pelo usuário;

c) arquivo makefile para compilação e geração do executável;

d) *script* para execução da simulação.



(a)



(b)

Figura 2: Montagem de uma arquitetura (a) e configuração dos parâmetros de um processador (b)

3.3 Exploração do espaço de projeto

O VIPEX-VMP possibilita dois tipos de exploração da arquitetura: exaustiva e utilizando algoritmo genético. Atualmente, os parâmetros que podem ser avaliados são: configuração da latência da memória, número de unidades funcionais e configuração da memória cache de instruções e cache de dados.

No caso da simulação exaustiva, o usuário define quais parâmetros serão variados e o ambiente se encarrega de gerar todos os modelos de simulação e de executá-los.

A Figura 3 representa o cenário do explorador do espaço de projeto utilizando algoritmo genético. O ambiente tem como entrada a configuração base da arquitetura, definida pelo projetista, e a aplicação que será, por esta, executada. Ao receber os dados de entrada, o algoritmo genético cria a primeira população a partir da arquitetura base e em seguida realiza a simulação de todos os indivíduos que foram criados. Ao finalizar a simulação de toda a população, ocorre a transição três, na qual os dados obtidos pela simulação são armazenados no banco de dados e alimentam o algoritmo genético.

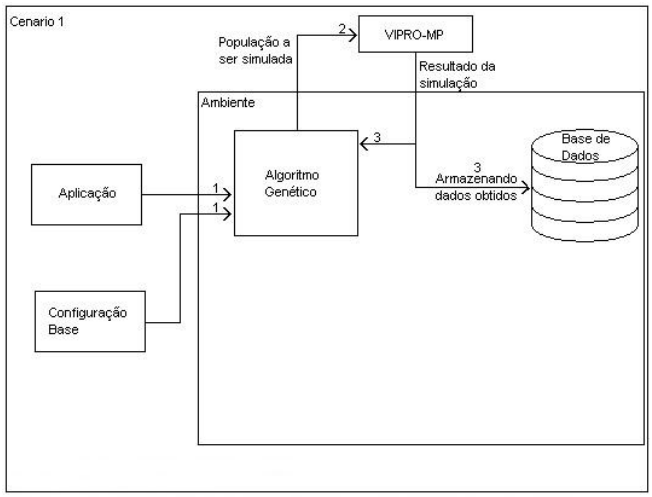


Figura 3: Fluxo de execução da ferramenta

O método de seleção utilizado foi o método do torneio, que seleciona *n* indivíduos aleatoriamente e define o melhor, que entra para um novo torneio, e assim sucessivamente até que um indivíduo seja escolhido como o melhor. Esse método foi escolhido por não favorecer os melhores indivíduos da população. No entanto, a única vantagem que os melhores indivíduos têm é que, sendo selecionados, vencerão o torneio [13]. O método de mutação implementado foi o método da mutação aleatória, no qual são definidas as chances que um cromossomo tem de sofrer a mutação. Caso a mutação ocorra, o cromossomo é substituído por um valor randômico entre o mínimo e o máximo permitido. A mutação aleatória, além de possuir uma fácil implementação, é muito eficiente para retirar o AG de um máximo ou mínimo local. O método de cruzamento escolhido foi o do corte mediano. A ideia base do corte mediano é de que cada "casal" (dois indivíduos resultantes do método de seleção) irá gerar dois "filhos", um constituido da primeira metade de cromossomos do "pai" e da segunda metade de cromossomos da "mãe", e o outro "filho" é composto pela segunda metade do pai e a primeira metade da mãe. Esse método permite que o número de cromossomos dos indivíduos permaneça o mesmo em todas as gerações, o que, nesse caso, é necessário, pois o cromossomo determinará qual a configuração de um determinado componente da arquitetura.

Na interface visual o usuário pode configurar os seguintes parâmetros do algoritmo genético: tamanho da população, número de gerações, critério de convergência e taxa de mutação. Para o critério de convergência o usuário informa o número máximo de repetições (em gerações) do melhor indivíduo.

4 Estudo de caso

Para avaliação do VIPEX-VMP, foi utilizado como estudo de caso de um codificador JPEG paralelo. Uma arquitetura similar a apresentada na Figura 1 com dois processadores e uma memória compartilhada, conectados entre si através de um barramento, foi utilizada como configuração base.

O algoritmo de compressão JPEG, adaptado do benchmark MiBench [12], foi utilizado nessa simulação. Como o algoritmo é sequencial, foi necessário adaptá-lo para suportar a execução paralela. O algoritmo é implementado em duas fases conforme apresentado na Figura 4. Na primeira fase, após o carregamento da imagem a ser comprimida na memória compartilhada, feito pelo processador mestre, cada processador do ambiente fica incumbido de realizar a transformada DCT (Discrete Cosine Transform) de N/P blocos JPEG e reescrever na memória compartilhada (N é o número total de blocos JPEG e P o número de processadores).

Quando todos os processadores terminam a primeira fase, o mestre executa a fase final e sequencial do algoritmo, a compressão (entropy encoder) e gera o arquivo de saída. Para a sincronização entre os processadores variáveis de espera são utilizadas na memória compartilhada, sendo que os processadores utilizam a espera ativa.

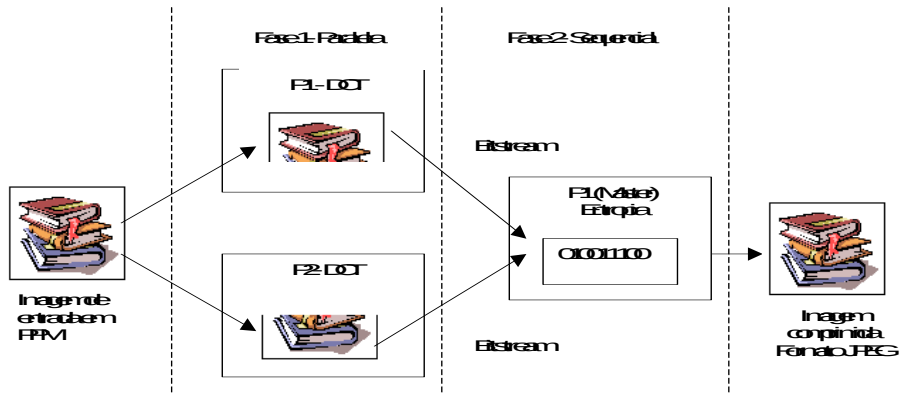


Figura 4: Algoritmo de compressão JPEG paralelizado

4.1 Simulação exaustiva

Na simulação exaustiva, o VIPEX-VMP varia os parâmetros do processador criando novas arquiteturas e simulando uma a uma. Após definido as configurações da arquitetura, para realizar a simulação exaustiva, basta selecionar a opção "variar" no menu do ambiente visual. O VIPEX-VMP simulará cada uma das arquiteturas e ao final gerará um arquivo na pasta de saída com o nome "dump", contendo em cada linha a configuração, o desempenho e a potência da arquitetura simulada. No estudo de caso foram variados quatro parâmetros, conforme apresentado na Tabela 1. Os valores mínimos e máximos foram fixados considerando as configurações de processadores embarcados e de propósito geral disponíveis atualmente.

Tabela 1: Parâmetros utilizados pela simulação exaustiva

Componente	Mínimo	Máximo	Incremento
Latência da memória compartilhada	4	16	2^n
Cache de dados nível 1 (DL1)	64 KB	65536 KB	2^n
Cache de instruções nível 1 (IL1)	256 KB	2621444 KB	2^n
Número de unidades funcionais	1	8	1

A memória compartilhada, a memória cache IL1 e cache DL1 são variadas em intervalos de 2^n e o número de unidades funcionais varia em incrementos de um. Cada configuração é aplicada em ambos os processadores, sendo assim, o número total de combinações é de 2904 arquiteturas. Cada simulação executada pelo VIPRO-MP leva cerca de 90s para decodificar uma imagem de 16 x 16 pixels, resultando em um tempo total de simulação de cerca de 72h.

A Figura 5 apresenta os resultados obtidos com a simulação das 2904 arquiteturas, através da relação entre o número de ciclos e a potência consumida pela arquitetura, portanto, quanto mais à direita a arquitetura estiver, maior o número ciclos utilizados, ao passo que quanto mais acima estiver, maior a potência consumida. Consideramos a melhor arquitetura a que apresenta o melhor compromisso entre desempenho e potência. Nesse caso é a arquitetura que estiver mais próxima da origem do gráfico, pois apresentou o menor consumo de potência e o melhor desempenho (menor número de ciclos) para finalizar a tarefa. A arquitetura mais balanceada consome 461046 ciclos de execução e tem consumo de potência de 201,74 Watts. Na Figura 5, estão assinalados também alguns exemplos de arquiteturas interessantes do ponto de vista do projeto obtidas com a exploração do espaço de projeto, como a arquitetura com o menor tempo de execução que necessita de 458560 ciclos de execução, porém consome 629,59 Watts. Da mesma forma, se o requisito for baixa potência, temos o outro extremo, onde a arquitetura consome 721430 ciclos, porém, consome apenas 153,11 Watts.

Em cada uma das simulações, ambos os processadores receberam a mesma configuração, porém, para uma melhor otimização da arquitetura, os processadores deveriam ser testados com todas as combinações possíveis, o que, para nosso caso de teste, resultaria em 8433216 arquiteturas diferentes, elevando o tempo de otimização para cerca de 210830 horas (aproximadamente 24 anos). Essas estimativas demonstram a inviabilidade da realização de testes exaustivos caso apenas uma máquina seja utilizada para execução da simulação.

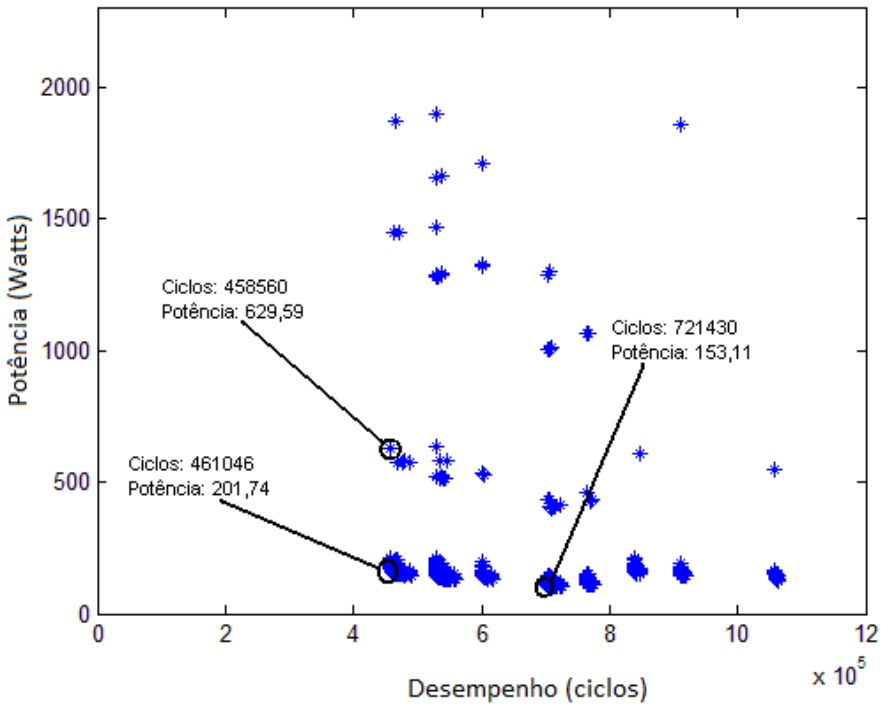


Figura 5: Relação desempenho e consumo de potência obtida na simulação exaustiva

4.2 Exploração do espaço de projeto utilizando algoritmo genético

No VIPEX-VMP, um algoritmo genético (AG) foi utilizado para gerar as arquiteturas que serão simuladas. O AG foi modelado de maneira que cada um dos indivíduos seja uma arquitetura, sendo que seus cromossomos carregam parâmetros da configuração dos processadores. A primeira população é gerada a partir de mutações da arquitetura base fornecida pelo projetista e cada uma das arquiteturas é simulada pelo VIPRO-MP, que por sua vez gera um arquivo contendo o número de ciclos e a potência consumida pela arquitetura simulada.

Após todas as arquiteturas da geração serem simuladas, são aplicados os métodos de seleção, de cruzamento e de mutação sobre a população atual, criando assim a população da próxima geração. O estudo de caso foi configurado para um número máximo de 300 gerações, com dez indivíduos em cada geração. O critério de convergência adotado foi a repetição da melhor arquitetura por 15 gerações.

A função *fitness* utilizada para a avaliação dos indivíduos foi a soma da potência média dos processadores com o desempenho da arquitetura. Como a diferença entre o número de ciclos e a potência consumida é muito grande, com o intuito de melhorar a avaliação, ambos os parâmetros foram normalizados, para isso o desempenho foi dividido por 10^8 e a potência por 10^3.

A configuração base da arquitetura utilizada no algoritmo genético foi a mesma utilizada nos testes exaustivos. O AG foi implementado para variar 46 parâmetros, sendo 23 em cada processador. Em testes iniciais, foi identificado um problema ao perceber que cerca de 47% das arquiteturas geradas pelo AG já tinham sido simuladas em alguma geração anterior. Para essas arquiteturas era então atribuído o resultado já obtido, evitando que fossem novamente simuladas. Isso reduziu o número de arquiteturas simuladas e avaliadas, levando

à convergência precoce para um mínimo local. Para contornar esse problema, a taxa de mutação foi aumentada de 30% para 75%, reduzindo o percentual de arquiteturas repetidas para 16%. Com essas configurações, a convergência foi obtida entre 165 a 270 gerações, resultando em um decréscimo entre 43% a 25% do número de simulações realizadas, quando comparados aos testes exaustivos. É importante, no entanto, notar que na exploração com AG, as configurações dos dois processadores são variadas independentemente, gerando um maior número de soluções possíveis. O tempo de execução desses testes variou de 34 horas para o caso onde a convergência foi obtida após 165 gerações e aproximadamente 56 horas para o caso onde foram necessárias 270 gerações. Os resultados obtidos são apresentados na Figura 6, sendo a arquitetura que está mais próxima da origem a que possui a potência e o desempenho mais balanceado.

Podemos analisar pelo gráfico apresentado na Figura 6 que a exploração do espaço de projeto possibilita ao projetista escolher a melhor arquitetura conforme os requisitos do projeto. Além da arquitetura com o melhor compromisso entre desempenho e potência (com 485589 ciclos e consumo de potência de 154,24 Watts), temos também a arquitetura que apresentou o menor número de ciclos consumidos para execução 406635, porém obtendo uma potência de 1015,54 Watts. Em outro extremo temos a arquitetura com o menor consumo de potência (140,05 Watts), que necessita de 1349390 ciclos para execução. Esse é o caso típico em um sistema embarcado, onde o projetista terá que escolher a arquitetura baseado nos requisitos impostos pela aplicação, como tempo de execução e limite máximo de potência e energia consumida.

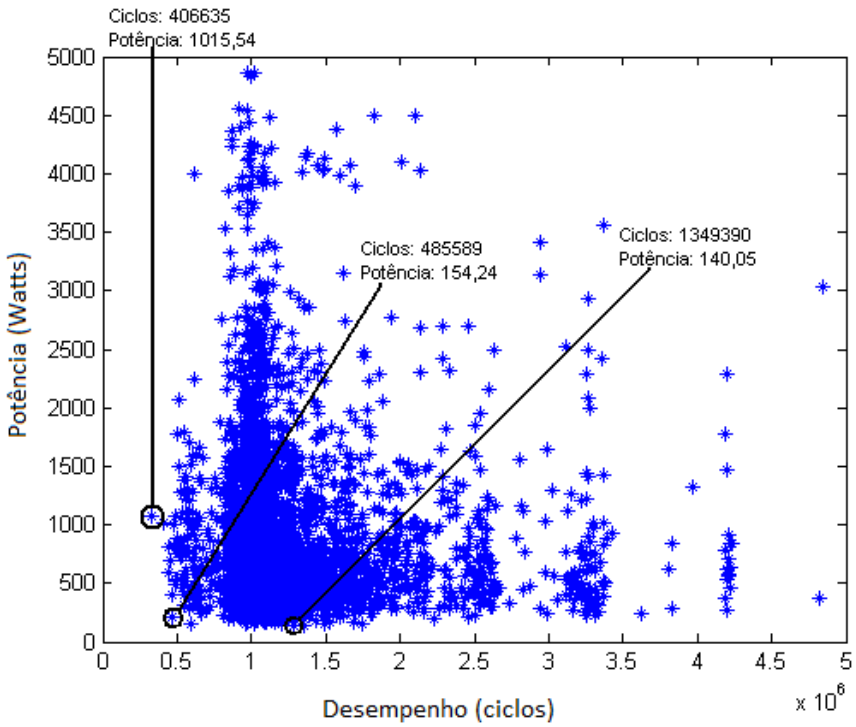
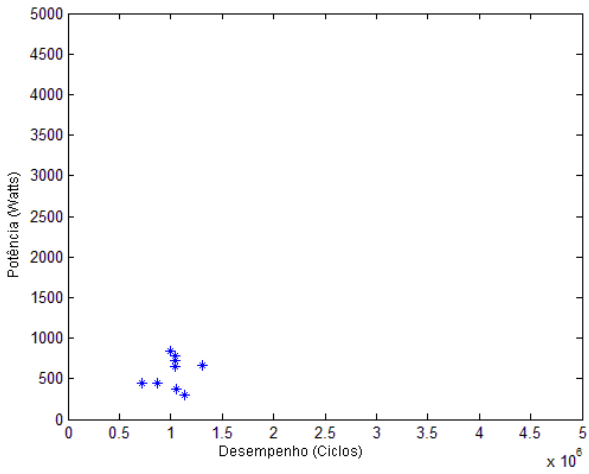
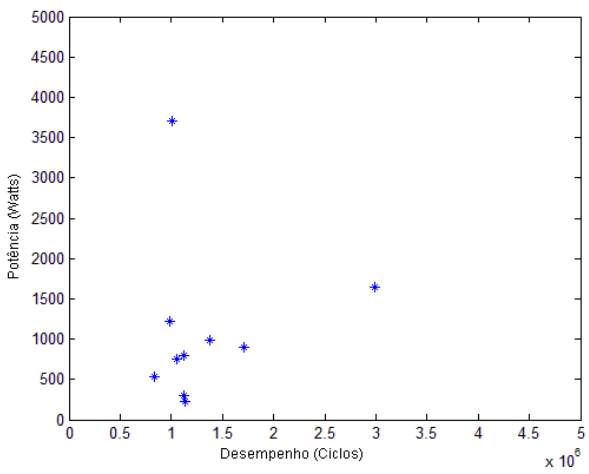


Figura 6: Resultados da exploração do espaço de projeto utilizando algoritmo genético

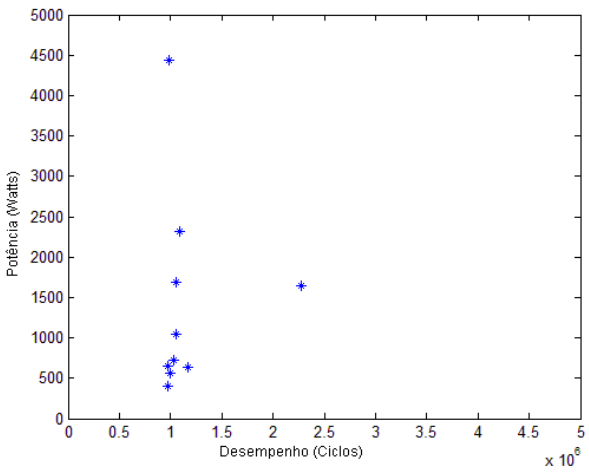
Analisando os resultados obtidos, pode-se observar que a cada geração o AG busca tornar os indivíduos o mais balanceado possível, tentando reduzir tanto a potência quanto o desempenho. Para melhor avaliação dos AGs, foram gerados gráficos das gerações 1, 75, 150, 225 e 300, representados na Figura 7. Os gráficos apresentados na Figura 7 foram resumidos utilizando faixas de desempenho com intervalos de 5000 ciclos, facilitando assim a visualização. Dessa forma, a cada 5000 ciclos, é representada apenas a arquitetura com melhor compromisso entre desempenho e consumo de potência. Devido à alta taxa de mutação utilizada, é comum a ocorrência de arquiteturas atípicas, que obtém resultados muito diferentes dos outros indivíduos da geração. Para avaliar o comportamento dos AGs perante a otimização de arquiteturas, o critério de convergência foi removido, para que o AG pudesse trabalhar até a geração de número 300. O tempo de execução desse teste foi de aproximadamente 62h, com 517 arquiteturas repetidas, representando, portanto, 17,23% do total de arquiteturas.



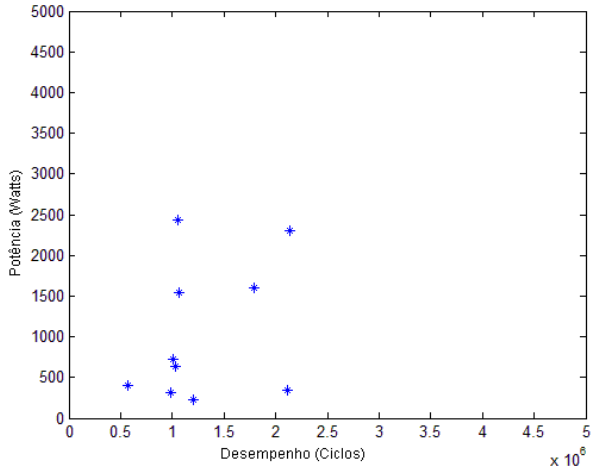
(a) Geração 1



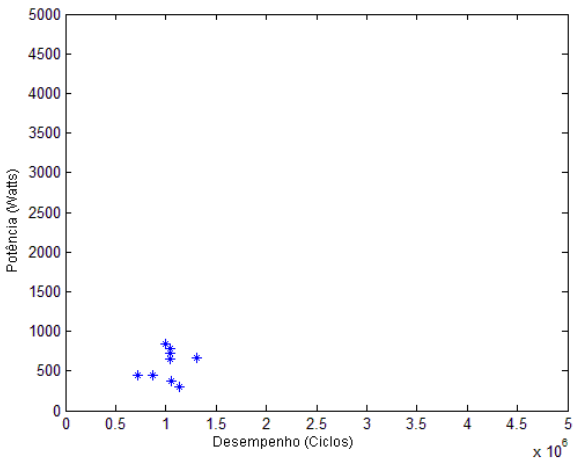
(b) Geração 75



(c) Geração 150



(d) Geração 225



(e) Geração 300

Figura 7: Resultados obtidos na exploração do espaço de projeto dividido em gerações do AG

Utilizando o AG, a arquitetura mais balanceada obteve um desempenho de 485589 ciclos e um consumo de apenas 154,247 watts. A melhor arquitetura obtida pelos testes exaustivos descrito na seção 4.1 obteve um desempenho de 461046 ciclos e um consumo de 201,74 watts. A arquitetura obtida pelo AG tem desempenho 5% inferior, porém apresenta a redução de 23% no consumo de potência, sendo que essa diferença foi obtida devido à possibilidade de utilização de configurações diferentes em cada processador da arquitetura. A Tabela 2 resume os resultados obtidos com a simulação exaustiva e com algoritmo genético. O número e o tempo de simulação utilizando-se algoritmo genético foram o melhor e o pior caso obtidos durante os testes, dependendo do número de gerações necessárias para obter a convergência da solução. É importante notar que mesmo com o tempo de simulação próximo o algoritmo genético obtém uma arquitetura mais balanceada. Como o algoritmo genético configura os dois processadores com diferentes parâmetros, para que fosse possível otimizar a arquitetura utilizando a simulação exaustiva seria necessário a simulação de 8433216 arquiteturas diferentes, o que demandaria aproximadamente 24 anos considerando-se a utilização de apenas um computador para realizar a tarefa.

Tabela 2: Comparação algoritmo genético x simulação exaustiva

	Exaustiva		Genéticos	
	Desempenho (ciclos)	Potência (watts)	Desempenho (ciclos)	Potência (watts)
Arquitetura com melhor compromisso consumo x potência	461046	201,74	485589	154,24
Arquitetura com melhor desempenho	458560	629,59	406635	1015,54
Arquitetura com menor consumo de potência	721430	153,11	1349390	140,05
Número de arquiteturas simuladas	2904		1650 – 2700	
Tempo de simulação	72 horas		34 - 56 horas	

5 Conclusões

Cada vez mais os projetistas se preocupam em desenvolver um SE que possua o máximo desempenho juntamente com o mínimo de consumo de potência. Para isso, plataformas virtuais são utilizadas, permitindo uma rápida configuração e coleta de estimativas já no início do projeto. Entretanto, mesmo facilitando a simulação de uma arquitetura, faz-se necessário adicionar a essas plataformas métodos que possam otimizar a arquitetura base de forma eficiente.

Visando facilitar a simulação do protótipo virtual VIPRO-MP, o VIPEX-VMP foi desenvolvido. Esse sistema possui um ambiente visual que permite ao projetista criar e configurar arquiteturas com rapidez e facilidade, a partir da qual são gerados os simuladores VIPRO-MP com base nos parâmetros informados. Porém, o tempo gasto para realizar a otimização de uma arquitetura tende a ser grande, por ser necessária a simulação exaustiva de todas as possíveis arquiteturas. Um algoritmo genético (AG) foi utilizado para gerar arquiteturas e diminuir o tempo de exploração do espaço de projeto, apresentando melhores resultados quando comparados aos testes exaustivos, comprovando sua eficiência. Em termos de tempo de exploração do espaço de projeto, o uso de um algoritmo genético reduziu o tempo de simulação e o número de arquiteturas simuladas. Adicionalmente, pela possibilidade de testar configurações diferentes em cada núcleo o algoritmo possibilitou a obtenção de uma arquitetura com melhor balanceamento entre consumo de potência e desempenho, reduzindo em até 23% o consumo de potência, com a penalidade de apenas 5% no desempenho.

Como trabalhos futuros, estuda-se a paralelização da execução da simulação em clusters de computadores. Adicionalmente, para o algoritmo genético outros métodos de seleção, cruzamento e mutação podem ser testados, além da inclusão da exploração do número de processadores utilizados na arquitetura.

Agradecimentos

Os autores agradecem o apoio financeiro concedido pela Fundação Araucária para o desenvolvimento do trabalho.

Referências

[1] MARWEDEL, P. *Embedded System Design*. Editora Springer 2006.

[2] ZHANG, C., VAHID, F., LYSECKY, R.L. A Self-Tuning Cache Architecture for Embedded Systems. In: Design Automation and Test in Europe, 2004 Proceedings... IEEE Computer Society Press, p. 142-147, Paris, 2004.

[3] CARRO, L.; WAGNER, F.R. *Sistemas Computacionais Embarcados*. In: JAI 2003. Campinas: Sociedade Brasileira de Computação, 2003, v.1.

[4] GARCIA, M. S.; SCHUCK, M ; OYAMADA, M. S. VIPRO-MP: a virtual prototype for multiprocessor architectures based on the SimpleScalar. In: 17th Annual IFIP International Conference on Very Large Scale Integration VLSI-SoC, 2009, Florianopolis. *Proceedings* of VLSI-SoC 2009, 2009. v. 1.

[5] CONVERGENSC, Disponível em <http://www.coware.com>. Acesso em: jul. 2010.

[6] HOFFMAN, A. A Novel Methodology for Design of Aplication Specific Instruction Set Processor (ASIP) Using a Machine Description Language. *IEEE Transactions on Computer-Aided-Design*, p.1338-1354, novembro, 2001.

[7] SYSTEMC. Disponivel em <http://www.systemc.org/home>. Acesso em: jul. 2010.

[8] LEI, T., YANHUI, Y., SHAOJUN, W. Optimizing SoC Platform Architecture for Multimedia Applications. ASICON 2005. pp.94-97, 2005.

[9] OVP- Open virtual platform. Disponível em: <www.openworld.org>. Acesso em: mar. 2011.

[10] GIRVARGIS, T; VARID, F.P. A Tuning Framework for Systems-on-a-Chip Platforms. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, v. 21, n. 11, 2002.

[11] SILVA-FILHO, A. G.; BASTOS-FILHO, C. J. A; FALCÃO D. M. A; CORDEIRO F. R.; CASTRO M. S. C. An Optimization Mechanism Intended for Two-Level Cache Hierarchy to Improve Energy and Performance using the NSGAII Algorithm. SBAC-PAD '08. p.19-26, 2008.

[12] GUTHAUS, M.; RINGENBERG, J.; ERNST,D.; AUSTIN, T.; MUDGE, T.; BROWN, R. MiBench: A free, commercially representative embedded benchmark suite. IEEE 4th Annual Workshop on Workload Characterization, Austin, 2001.

[13] LINDEN, R. *Algoritmos genéticos*: uma importante ferramenta da inteligência computacional. 2 ed. Rio de Janeiro: Brasport, 2008.