Multiprocessor Performance Estimation Using Hybrid Simulation

L. Gao, K. Karuri, et. al. Em ACM IEEE Design Automation Conference, 2008, pp 325-330.
Resumo de artigo realizado por Carlos Alberto Petry – RA 095345

Introdução

O crescente uso de sistemas MPSoCs, contendo cada vez mais elementos de processamento, tem exigido maior sinergia dos projetistas. Não raramente, mudanças na arquitetura do hardware podem requerer novos esquemas de particionamento e paralelização do software. Técnicas para se realizarem estimações de desempenho e simulações rápidas e precisas, vem se tronando mandatórias em fases iniciais do desenvolvimento de forma a facilitar a exploração do espaço de projeto. Este artigo enfrenta estes aspectos pelo uso combinado de simulação híbrida, simulação de cache e técnicas de repetição *trace-driven* como forma de obter informações de desempenho para sistemas MPSoCs, objetivando verificação funcional e estimação de desempenho.

Diversas aplicações embarcadas permitem paralelização em nível de tarefa, eficientemente exploradas por arquiteturas multiprocessadas (MPSoC) que envolvem o desenvolvimento simultâneo de hardware e software (co-design). A exploração abrangente do espaço de projeto é determinante para obter êxito no desenvolvimento, envolvendo diferentes simulações frente a alternativas de implementação.

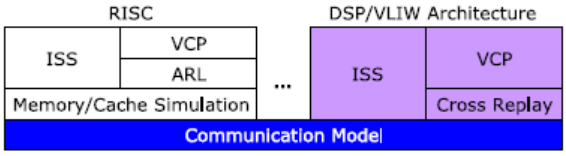

Figura 1 - Modelo de simulação em camadas

A figura 1 mostra um modelo abstrato de uma arquitetura MPSoC contendo diferentes elementos de processamento (PE) e uma arquitetura de comunicação. Neste trabalho foram implementados PEs RISC-DSP/VLIW e barramento compartilhado, considerado-se apenas PEs nas discussões.

PEs podem ser simulados usando-se ISSs ou podem ser executados diretamente na arquitetura alvo. Existem diversas técnicas para aceleração de simulações, entretanto não atingem a velocidade desejada.

As duas principais contribuições deste trabalhos são: (i) fornecer estimativas de desempenho precisas sem fazer uso de preprocessamento ou fases de treinamento; (ii) dispor de uma estrutura aplicável a uma grande quantidade de PEs. Também estende as técnicas de simulação propostas em trabalhos anteriores como o *Hybrid Simulation Framework* (*HySim*) permitindo estimações tanto para PEs RISC como DSP / VLIW.

Estrutura de simulação HySim

HySim recebe uma combinação de código escrito em C e Assembly executados num *mix* de ISS e execução nativa, como forma de aumentar o desempenho. Seções de código independente são chamadas *virtualizáveis* e elementos de programa dependentes são denominados *não-vitalizáveis.* Durante a execução os códigos *não-vitalizáveis* são executados no ISS e os *vitalizáveis* executados no *host* de simulação através do *Virtual Co-Processor* (VCP).

Estimação de desempenho de PEs RISC

O PE RISC é utilizado como elemento de controle da arquitetura de processamento. A estimativa de contagem de ciclo para este PE pode ser obtida a partir da execução de dois tipos de estatísticas: (i) freqüência da operação de execução; (ii) comportamento da memória cache. Operações C são implementadas por conjuntos de instruções RISC. Como o número de ciclos destas instruções é conhecido, torna-se fácil inferir o custo requirido para as executar as operações. Para acessos à memória, o número de ciclos depende do comportamento da cache. Assim para o cálculo global do número de ciclos para uma operação C usa-se a seguinte fórmula:

$$Cycles = \Sigma_{i=1}^{n} N_i \times C_i + N_{hit} \times C_{hit} + N_{miss} \times C_{miss}$$

onde N_i e C_i são a contagem de execução e custo, N_{hit} e N_{miss} são as estimativas de *hits* e *misses* da cache e C_{hit} C_{miss} os seus custos. A estimação de N_i para a operação *i* não é realizada em nível de código C, ao invés disto a aplicação é convertida para um formato intermediário de endereçamento permitindo que um *Instrumentador* possa calcular o custo de cada operação.

Estimação de desempenho do PEs DSP/VLIW

Arquiteturas de domínio específico (DSP NPU VLIW) possuem alguns pressupostos que facilitam estimar seu desempenho, como: (i) a maioria destes PEs não possuem cache levando a uma execução de tempo constante e (ii) seu código frequentemente possui grande volume de dados e um número limitado de caminhos de controle, facilitando inferir seu desempenho. Isto é implementado usando uma técnica de perfil dinâmico denominada *Croos Replay* (CR). A técnica CR executa a função *virtualizada* sobre o VCP gerando e armazenado *traces* únicos relativos ao caminho de controle. Concluída a execução *virtualizada* um cenário é pesquisado, caso não exista a função é re-executada no ISS obtendo e registrando dados de desempenho, existindo o registro é recuperado. Esta abordagem habilita *trace-replay* em nível de função sendo *cross-ISA* e suportando binários otimizados.

Resultados experimentais

Dois grupos de experimentos foram feitos: (i) estimação de desempenho individual para RISC e DSP/VLIW; (ii) estimação dos PEs inseridos no MPSoC. Para PE MIPS, aplicações criptográficas tiveram boas acelerações (9x), porém para JPEG foi modesta (1,4x). Investigações apontaram um particionamento ineficiente. As estimativas foram comparadas com ISSs precisos em ciclo. Estimações para cache foram satisfatórias (erro 0,6% a 8%) exceto MD5 devido a *spills* excessivos. Para DSP mAgic o aumento da velocidade com *trace* foi de 10x a 50x. A sobrecarga com cross-replay foi satisfatória: 0,6% a 9%. Para o MPSoC os resultados foram razoáveis:.comparado a ISSs precisos em ciclo o erro foi de no máximo 3% e a velocidade 3x a 5x maior. A aceleração comparada com a simulação individual é pequena deviso à sincronização global.

Conclusões

A estrutura de simulação híbrida permite ser aplicada para estimação de desempenho de MPSoCs e usada com diferentes PEs, permitindo melhor ajuste do software executado. Os resultados apontam para uma redução significativa no tempo de simulação, mantendo razoável precisão. Segmentos de sincronização *não-vitalizáveis* são o maior gargalo na simulação multiprocessada, sendo alvo para futuras pesquisas bem como atingir precisão nas ocorrências de *spills*.