Extensão do simulador de dependências em nível de instrução AD3W para agregar maior interação com o usuário e tratamento de instruções MIPS

Tiago Siqueira Brum[1], Guilherme Schievelbein[1], Alexandre dos Santos Roque[1]

[1]Departamento de Engenharias e Ciência da Computação – DECC, Universidade Regional Integrada do Alto Uruguai e das Missões – URI – Campus Santo Ângelo
CEP – 91.501-970 – Santo Ângelo, RS – Brasil

tiagobrum@hotmail.com.br, {guilhermschievelbein, ale.roque}@gmail.com

Resumo. *Este artigo apresenta a extensão do simulador educacional AD3W. Nesta versão é agregada a análise de instruções MIPS e também a possibilidade de parametrizar a quantidade de unidades funcionais, com o objetivo de ilustrar de forma educacional as possibilidades de paralelismo. O algoritmo de controle trata desvios condicionais juntamente com as técnicas de escrita em registradores para detectar dependências em nível de instrução.*

1. Introdução

No estudo de arquitetura de computadores, diversas ferramentas e técnicas são usadas para auxílio no processo de ensino aprendizagem [Ribeiro e Rimsa, 2008] [Gonçalves *et al*, 2007] [Coutinho, 2006]. No contexto de ferramentas de auxílio os simuladores se destacam por serem ótimas opções para ilustrar conceitos de maneira simples e com um bom nível de interação, promovendo uma prática que muitas vezes fica somente em leituras avançadas de determinado conteúdo [Brorsson, 2002] [Grunbacher, 1998] [Macêdo, 2009]. Por conseguinte, em arquiteturas paralelas também é importante ilustrar de forma educativa as possibilidades de paralelismo em nível de instrução, como um escalonador pode distribuir instruções em arquiteturas e ambientes multiprocessados, com o objetivo de prover um ganho de *speedup* (tempo total / n processadores) e economia de recursos [Dongarra, 2003].

Desta forma, o uso de ferramentas e *softwares* de apoio ao ensino encoraja e estimula o estudo nas áreas de processamento paralelo e computação de alto desempenho (PAD/HPC). Assim, é possível efetuar um estudo detalhado das propriedades de um sistema paralelo e observar seu funcionamento interno [Gonçalves *et al*, 2007]. Dentro deste contexto, o objetivo deste artigo é apresentar a extensão do Simulador AD3W, proposto por Schievelbein *et al* (2013), que é uma ferramenta desenvolvida para auxiliar o processo de ensino-aprendizagem com relação à verificação de dependências de dados (*data hazards*) em instruções pseudo-*assembly*, com o intuído de demonstrar as possibilidades de paralelismo em determinados conjuntos de instruções. Sendo assim o presente artigo apresenta a inclusão da verificação de dependências de dados em instruções MIPS além da viabilidade de parametrização do número de unidades funcionais.

Este artigo esta organizado da seguinte maneira: A Seção II faz uma breve contextualização do tema; a Seção III apresenta o desenvolvimento do simulador; a Seção IV apresenta as conclusões relativas ao desenvolvimento do trabalho.

2. Contextualização

Para obtenção do paralelismo em nível de instrução a utilização de *pipelines* paralelos nas arquiteturas é fundamental. Três políticas de iniciação de instruções são tratadas para o aprendizado dos *pipelines* [Hennessy e Patterson, 2008]: I. Iniciação em Ordem com terminação em Ordem; II. Iniciação em Ordem com terminação fora de Ordem; III. Iniciação fora de Ordem com Terminação fora de Ordem. A dependência de dados ocorre quando uma instrução depende do resultado de outra instrução que ainda não foi executada (ainda está no *pipeline*). Este tipo de dependência é originado na natureza sequencial do código em execução, cuja ordem normal é alterada dentro do *pipeline*.

De acordo com [Hennessy e Patterson, 2008] os tipos de dependências *(hazard classification*) são organizados em: *data hazards*, *structural hazard* e *control hazard*/*branching hazards*.

A análise de dependências nestes níveis é importante para tratamento de diversos problemas comuns que são abordados na literatura de ensino de arquitetura de computadores, possibilitam diversos algoritmos e técnicas para tratamentos específicos, como por exemplo, um algoritmo para renomeação de registradores, o algoritmo de Tomasulo (que é mais abrangente), algoritmos de predição de desvios (*branch prediction*) [Hennessy e Patterson, 2008]. No simulador desenvolvido são tratadas as dependências de dados (*data hazards WAR, RAW e WAW)* e os desvios de controle (*branching hazards),* com o intuito de abordar os referidos algoritmos. O simulador AD3W bem como as melhorias inseridas nesta extensão, foram desenvolvidos na linguagem de programação JAVA.

Para o estudo de um conjunto de instruções, simuladores são frequentemente utilizados no ensino de características intrínsecas a arquitetura de computadores, e muitos deles baseiam-se no processador MIPS [Brorsson, 2002] [Avelar *et al*, 2008]. Para o desenvolvimento deste trabalho foram analisados alguns trabalhos, como por exemplo, o R10k, um simulador de arquitetura superescalar [Gonçalves *et al*, 2007] e o PS – CAS MIPS: um simulador de *pipeline* do processador MIPS [Maia *et al,* 2009].

3. Desenvolvimento do Simulador

Para o desenvolvimento do simulador foram considerados aspectos relativos à programação orientada a objetos, estabelecendo relação entre uma classe e seus atributos com a composição de uma instrução na linguagem *assembly* genérica ou do MIPS. Tendo em vista que há a possibilidade de inserção e simulação de instruções em ambas as formas.

Como cada instrução possui vários atributos que divergem entre si e estes são comparados uns com os outros para detectar as dependências de dados, esta comparação acaba se tornando menos complexa quando cada aspecto destes é definido como um atributo de uma classe, neste caso como a definição das classes: Linha, Dependência e Label, implementadas para o desenvolvimento do simulador e visualizado na Figura 1.

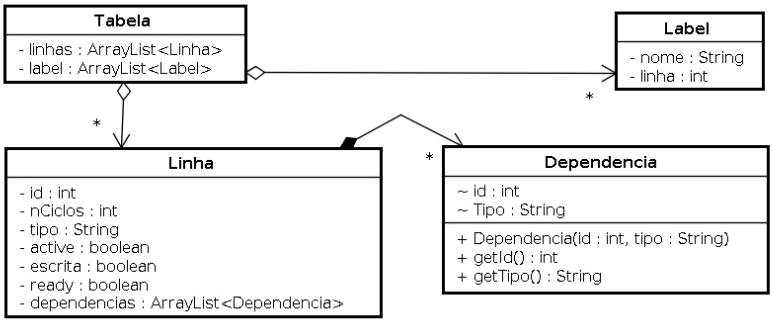

Figura 1. Classes que compõe uma instrução

Além destes valores cada instrução possui uma lista contendo quais instruções possuem dependências da mesma. As instruções são carregadas em uma lista, onde cada uma ocupa uma posição. A entrada das instruções no simulador é feita por meio de um texto com instruções na própria interface ou a inserção de um arquivo do tipo texto, contento uma das duas linguagens citadas anteriormente, sendo que o simulador detecta automaticamente qual é a linguagem que deverá ser analisada, a Figura 2 apresenta a tela dos resultados ao usuário.

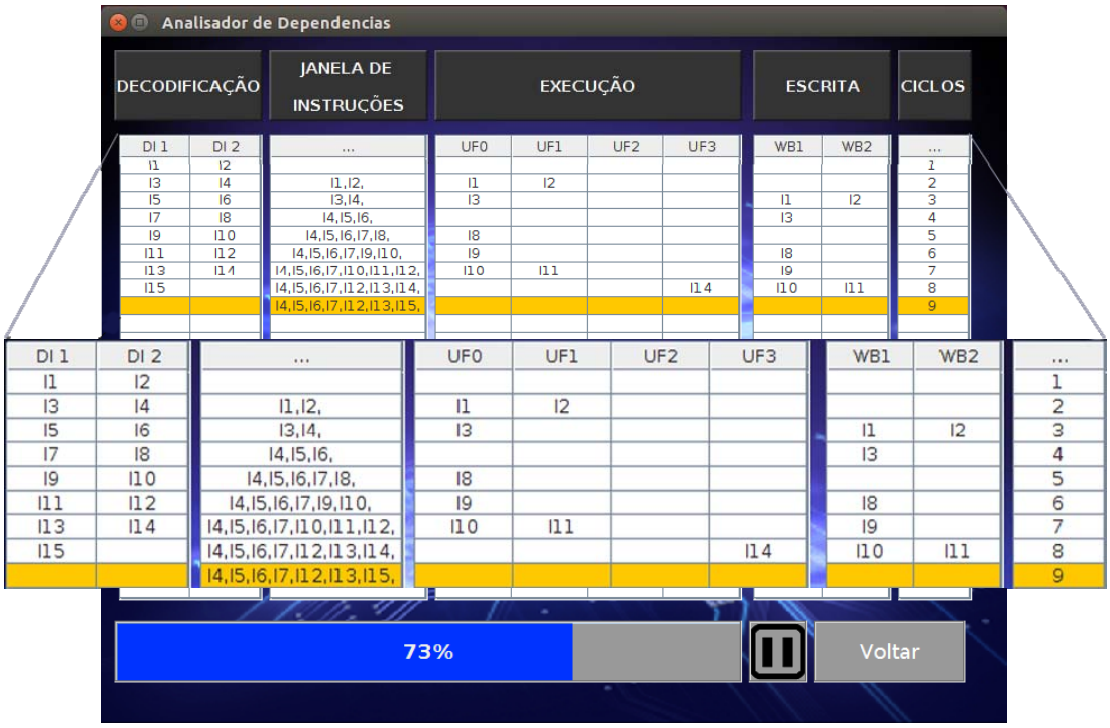

Figura 2. Tela de Apresentação dos Resultados ao Usuário.

A análise da entrada do usuário resulta em três informações de saída: a forma de execução das instruções, as colunas de decodificação, janela (implementada de forma centralizada), a execução, a escrita e qual o tipo de dependência que cada instrução possui com as demais instruções, desta forma, é possível verificar o conflito que gera uma dependência de dados.

4. Conclusões

Com a apresentação da extensão do simulador AD3W é proposta uma forma alternativa, de maneira simples e rápida, para ensino dos conceitos de paralelismo em nível de instrução com identificação de diferentes tipos de dependências. Pode-se destacar a nova funcionalidade que permite a análise de duas linguagens sendo elas: Pseudo-*Assembly* e MIPS, bem como, a opção inicial de parametrização da quantidade e tipo de unidades funcionais existentes para a simulação da execução das instruções.

A utilização de simuladores em sala de aula promove uma evolução no processo de ensino-aprendizagem de diversos conceitos de arquitetura de computadores, tendo em vista da possibilidade de simulação passo a passo do processo de análise das instruções.

Referências

Ribeiro, A. I. J. T., & Rimsa, A. (2008). Técnica Motivacional Para o Ensino de Arquitetura de Computadores com Ênfase nos Grandes Desafios da Computação. Em *Workshop sobre Educação em Arquitetura de Computadores-WEAC*.

Gonçalves, A. N., Gonçalves, R. L., Martini, J. A. (2007). R10k: Um Simulador de Arquitetura Superescalar. Em *Workshop Sobre Educação em Arquitetura de Computadores–WEAC*.

Maia, D. W., Vieira, M. M., & Pessoa, R. F. (2009). PS-CAS MIPS: Um simulador de pipeline do processador MIPS 32 bits para estudo de Arquitetura de Computadores. Em *Workshop sobre Educação em Arquitetura de Computadores*.

Brorsson, M. (2002, May). MipsIt: a simulation and development environment using animation for computer architecture education. Em *Proceedings of the 2002 workshop on Computer architecture education: Held in conjunction with the 29th International Symposium on Computer Architecture* (p. 12). ACM.

Avelar, C. P., Jonathan Filho, J. C., Carvalho, V., & Martins, C. A. (2008). MPDSim: Simulador didático do Pipeline do MIPS de 32 bits. In *Workshop sobre Educação em Arquitetura de Computadores WEAC* (Vol. 2008, pp. 9-12).

Hennessy, J. L., & Patterson, D. A. (2008). *Arquitetura de computadores: uma abordagem quantitativa*. Elsevier.

Grunbacher, H. (1998). Teaching computer architecture/organisation using simulators. In *Frontiers in Education Conference, 1998. FIE'98. 28th Annual*. IEEE.

Macêdo, J. A. D., & Dickman, A. G. (2009). Simulações computacionais como ferramentas auxiliares ao ensino de conceitos básicos de eletricidade. *XVIII Simpósio Nacional de Ensino de Física, Vitória-ES*, 1-12.

Dongarra, J. Foster, I. Fox, G. Gropp, W. Kennedy, K. Torczon L. and White A. (2003) *Sourcebook of Parallel Computing. Morgan Kaufmann Publishers*, San Francisco, USA.

Coutinho, L. M., Mendes, J. L., & Martins, C. A. (2006). Web-MHE: Ambiente web de auxílio ao aprendizado de hierarquia de memória. In *Workshop sobre Educação em Arquitetura de Computadores-WEAC*.

Schievelbein, G; Losekan. M; Roque, Alexandre; Silva, Denílson R. AD3W: Um simulador educacional para análise de dependências de dados em nível e instrução. *International Journal of Computer Architecture Education, v. 2, p. 41-44*, 2013.