CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
CAMPUS TIMÓTEO

Elias Luiz da Silva Júnior

ANÁLISE DE DESEMPENHO DE UMA IMPLEMENTAÇÃO DE UNIDADE DE MEMORIZAÇÃO DE TRAÇOS DINÂMICOS EM FPGA

Timóteo

2016

Elias Luiz da Silva Júnior

ANÁLISE DE DESEMPENHO DE UMA IMPLEMENTAÇÃO DE UNIDADE DE MEMORIZAÇÃO DE TRAÇOS DINÂMICOS EM FPGA

Monografia apresentada à Coordenação de Engenharia de Computação do Campus Timóteo do Centro Federal de Educação Tecnológica de Minas Gerais para obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Bruno Rodrigues Silva

Timóteo

2016

.

.

.

Resumo

Palavras-chave:

Abstract

Keywords:

Sumário

1	**INTRODUÇÃO**	**5**
1.1	**Justificativa**	**5**
1.2	**Problema**	**6**
1.3	**Objetivos**	**6**
1.4	**Estrutura da tese**	**7**
2	**PROCEDIMENTOS METODOLÓGICOS**	**8**
2.1	**Escolha da Arquitetura**	**8**
2.2	**Revisão da literatura**	**9**
2.3	**Adaptação e Implementação**	**9**
2.4	**Testes e Análise**	**10**
3	**FUNDAMENTOS HISTÓRICOS, TEÓRICOS E METODOLÓGICOS**	**12**
	REFERÊNCIAS	**13**

1 Introdução

A Lei de Moore possui algumas variações quanto ao seu enunciado, porém todas afirmam que a capacidade computacional dos processadores cresceria exponencialmente devido aos avanços na tecnologia. Por 50 anos essa previsão se manteve consistente com os produtos lançados no mercado, como descrito em. Porém, limitações físicas na criação de circuitos integrados ameaçam o continuidade dessa evolução. (MACK, 2011)

Mas com o crescente aumento da demanda por computação é necessário que os projetistas encontrem maneiras de aperfeiçoar ainda mais o funcionamento das unidades de processamento. Uma solução que vem sendo utilizada é acoplar vários processadores para funcionar em paralelo, porém isso aumenta a complexidade de projetos tanto a nível de hardware como de software, além de amplificar o consumo energético do sistema.

O grande desafio da arquitetura de computadores é buscar soluções eficientes, conciliando fatores como desempenho do sistema, consumo de energia, custo de produção e tamanho e complexidade do produto final. Em muitas situações, esses fatores concorrem entre si, levando o projetista a ter de tomar decisões sobre qual abordagem será escolhida para solucionar determinado problema.

O que ocorre então é a criação de sistemas especialistas para determinadas funções, enquanto outros projetos mais gerais lidam com uma gama mais diversa de aplicações. Em ambos os casos, projetistas consideram qual problema buscam resolver para criar a solução mais adequada dentro das restrições.

Como exemplo podemos comparar as diferentes abordagens assumidas ao projetar um *system-on-chip* para aplicação em um sistema embarcado e na criação de uma unidade de processamento gráfico. Enquanto sistemas embarcados prezam por tamanho reduzido e baixo consumo de energia, unidades gráficas têm como prioridade a velocidade para cálculos de ponto flutuante, sendo otimizadas para executar instruções simples a diversos dados de entrada simultaneamente. (TANENBAUM; ZUCCHI, 2009)

Assim, é importante conhecer e desenvolver técnicas que possam tornar os projetos mais eficientes. Desenvolver para que o custo-benefício do produto seja melhorado independentemente de avanços na tecnologia de produção, mas sim por um design melhor elaborado. Conhecer para que seja possível ponderar como e quais técnicas aplicar para que o objetivo final possa ser atingido de maneira ótima, com um máximo de desempenho e mínimo de recursos despendidos.

1.1 Justificativa

Como demonstrado em (COSTA, 2001), muitos programas acabam por ter instruções redundantes ao longo de seu fluxo de execução. Assim, tempo computacional é perdido para se obter resultados já calculados.

Uma das técnicas propostas para reduzir esse desperdício de poder de processamento é a *DTM: Dynamic Trace Memoization*, ou Memorização Dinâmica de Traces. A DTM armazena o resultado de conjuntos de instruções executados anteriormente e, caso detecte uma execução redundante do mesmo conjunto, é capaz de armazenar os resultados e desviar o fluxo de controle para a instrução a ser executada após esse conjunto, substituindo a execução linear de cada instrução pelo resultado final, como se o bloco inteiro fosse uma instrução somente. A técnica será abordada com mais detalhes na seção **??**.

Em simulações realizadas por (COSTA, 2001), essa técnica foi capaz de aumentar o desempenho de programas do *SpecInt95 Benchmark Suite* de 1% até 21%, variando de acordo com o programa e os parâmetros utilizados na construção das unidades responsáveis por implementar o mecanismo DTM.

É possível então notar que há aplicações para as quais a implementação de uma unidade de DTM poderia melhorar significativamente o desempenho. Sendo assim, é interessante conhecer as os impactos desta para que seja possível melhor avaliar em que situações a utilização da DTM é proveitosa, considerando os *trade-offs* causados por sua presença.

1.2 Problema

A arquitetura de um circuito e a tecnologia utilizada para a sua geração estão intimamente ligados às características do circuito resultante. Considerando isso, algumas métricas utilizadas nesses circuitos resultantes servem para comparar apenas um dos fatores, seja uma mesma arquitetura em diversas tecnologias ou diversas arquiteturas em uma mesma tecnologia.

Segundo (CHU, 2006), as principais métricas que podem ser utilizadas nessas medições são área de chip, velocidade, consumo de potência e custo de produção. Esses quesitos são correlacionados, fazendo que alterações mudem os valores de mais de um ponto, senão todos.

O problema que motiva este trabalho é saber, considerando essas métricas, se a DTM é uma técnica viável para aplicações práticas. Caso seja, quais os *trade-offs* que o projetista deve considerar ao aplicar a DTM no seu projeto.

1.3 Objetivos

O objetivo deste trabalho é avaliar como as métricas citadas na seção 1.2 são alteradas com a implementação do mecanismo de DTM em um processador.

Mais especificamente, os objetivos podem ser descritos nos seguintes tópicos:

1. Implementar a memorização dinâmica de traços em uma arquitetura de processadores;
2. Produzir um circuito físico do processador e comparar os resultados da arquitetura padrão e da arquitetura com DTM nas seguintes métricas:

- área de chip;
- potência consumida;
- latência de ciclo;
- Executar programas de *benchmark* sobre as duas arquiteturas e comparar os resultados de performance de ambas.

1.4 Estrutura da tese

Esta tese está organizada em **??** capítulos. Esses capítulos foram ordenados em uma sequência lógica que facilitasse a compreensão do leitor, já que cronologicamente houve certo paralelismo durante a produção de algumas etapas.

- No capítulo 2 é demonstrado o processo metodológico adotado no desenvolvimento deste trabalho. Nele é descrito o processo de seleção da literatura base, a definição das ferramentas e parâmetros para a implementação da DTM e como foi planejado a análise e comparação dos resultados obtidos.
- Sequencialmente, no capítulo **??** é apresentada a fundamentação que serve como base teórica para o trabalho, incluindo uma conceituação mais apurada sobre a técnica de memorização dinâmica de traces, além de discorrer sobre a arquitetura de processadores escolhida como referência para a implementação e as tecnologias utilizadas para o desenvolvimento.

2 Procedimentos metodológicos

Está pesquisa é descritiva em se tratando dos objetivos, já que tem como objetivo descrever as características observadas do objeto de pesquisa. É aplicada do ponto de vista de sua natureza, considerando que busca contrastar os resultados obtidos em modelos teóricos e simulações com resultados medidos em uma implementação real. É classificada como qualitativa quanto à abordagem ao problema, pois os resultados apresentados são medições de grandezas do objeto de estudo.

Os procedimentos metodológicos adotados podem ser descritas na seguinte sequência:

1. realizar um levantamento sobre arquiteturas de computadores disponíveis em código aberto, considerando sua compatibilidade com os equipamentos disponíveis no laboratório e características da arquitetura e implementação, selecionando a que mais se adequasse a um dos padrões propostos;
2. reunir e estudar trabalhos publicados relacionados ao problema principal abordado, que em suma são os trabalhos onde a DTM é primeiramente apresentada e alguns outros trabalhos relacionados analisando algumas facetas diferentes da mesma;
3. adaptar a técnica DTM para esse conjunto de instruções de máquina (*Instruction Set Architecture - ISA*), implementar a unidade de memorização dinâmica de traces em linguagem de descrição de hardware, testá-la de forma isolada e integrá-la ao processador da maneira menos invasiva, isto é alterando ao mínimo a unidade de processamento;
4. realizar testes em um processador sem DTM, que age como unidade de controle, e no processador com DTM, comparando os resultados de ambos obtidos através de simulação e execução física em placas de FPGA (*Field-programmable gate array*);

Essas etapas serão expostas em mais detalhes nas seções subsequentes deste capítulo.

2.1 Escolha da Arquitetura

O primeiro passo do desenvolvimento deste trabalho foi selecionar uma arquitetura de processadores que atendesse os requisitos necessários, listados abaixo:

- Disponível em código aberto na forma de alguma linguagem de descrição de hardware, preferencialmente em Verilog ou VHDL;
- Sintetizável e gravável no FPGA disponibilizado pela instituição, o chip Altera Cyclone II EP2C35F672C6;

- ISA do tipo RISC (*Reduced Instruction Set Computer*) ou Java;

O motivo para que a escolha da arquitetura se desse antes da revisão de literatura se dá pelo fato desta ser influenciada pelo tipo da arquitetura utilizado. Enquanto grande parte dos trabalhos utilizados seriam os mesmos, a bibliografia básica descrevendo a DTM está sujeita ao tipo da arquitetura.

Caso fosse selecionada uma arquitetura baseada em Java como o JOP (SCHOEBERL, 2005), em que o funcionamento do processador é baseado em uma máquina de pilha, seria utilizado como referência básica (SILVA, 2006), já que este trabalho lida de maneira mais detalhada com a técnica DTM aplicada a uma máquina Java.

Porém, por possuir melhor compatibilidade com o FPGA utilizado, decidiu-se utilizar a arquitetura LEON3, atualmente mantido pela Aeroflex Gaisler, que possui como ISA o SPARC v8 (GAISLER, 2010). Por ser uma arquitetura de modelo RISC tal qual a arquitetura MIPS, utilizada em (COSTA, 2001), este foi tido como a peça de bibliografia central para este trabalho.

2.2 Revisão da literatura

Como dito anteriormente na seção 2.1, a peça central da literatura utilizada neste trabalho é (COSTA, 2001) devido a sua abordagem minuciosa ao descrever todas as características que compõe a técnica de memorização dinâmica de traces. Sendo o trabalho mais completo tratando de DTM e contendo uma explicação detalhada em como implementar essa técnica em um processador RISC, é notável a semelhança com este trabalho, o que o torna uma referência de suma importância.

Cabe notar que, enquanto há outros trabalhos que trazem diferentes abordagens sobre DTM, como reuso especulativo através da predição de valores de entrada, neste trabalho o escopo foi limitado a uma abordagem mais simples, analisando o impacto dos conceitos básicos de DTM em um processador. Outras abordagens adicionando uma maior capacidade e complexidade à unidade serão tratadas na seção **??**, onde serão apresentadas algumas possíveis adições para futuros trabalhos que compartilhem essa temática.

Porém, devido à natureza prática deste trabalho, a revisão de literatura não se ateve ao estudo teórico da técnica. Uma seção considerável da bibliografia utilizada concerne às tecnologias componentes do trabalho, desde manuais confeccionados pela Altera tratando aspectos físicos do chip de FPGA utilizado à documentos da Aeroflex Gaisler referentes à comunicação com o processador para *debug* de software. Essas referências utilizadas na implementação serão melhor demonstradas nos capítulos **??** e **??**, onde ocorrerá um maior detalhamento das questões técnicas envolvidas no projeto.

2.3 Adaptação e Implementação

Em (COSTA, 2001) a técnica DTM é explorada aplicada a um processador cujo conjunto de instruções é o MIPS I. Neste trabalho a aplicação se dá em um processador que

implementa o SPARC v8. Apesar de possuírem muitas semelhanças há também diferenças entre eles, como o fato do SPARC implementar janelas de registradores para uma troca de contexto de execução mais rápida.

Diferenças como a citada se mostram como um empecilho para a aplicação direta de DTM em uma arquitetura diferente seguindo a implementação de (COSTA, 2001). Portanto neste trabalho decidiu-se por estudar o funcionamento da técnica de forma ampla e genérica e projetar uma unidade específica para a arquitetura utilizada, sem fugir dos conceitos básicos que definem o DTM.

A implementação se dá alterando o código-fonte do processador LEON3 na linguagem de descrição de hardware VHDL (*VHSIC Hardware Description Language* onde VHSIC significa *Very High Speed Integrated Circuit*). Por utilizar *generics*, ou seja uma implementação parametrizada, o LEON3 pode ter suas características adaptadas de acordo com a definição do usuário antes do processo de compilação ser iniciado.

Neste trabalho, o processador foi configurado da seguite forma:

- Discutir com o Bruno qual a melhor configuração.

2.4 Testes e Análise

Após completada a implementação do processador com DTM, será iniciada a etapa de testes e análise dos resultados.

Como descrito em 1.3, os testes realizados tem a função de determinar a área de chip, potência dissipada, latência de ciclo de execução e ganho de performance na execução de programas de *benchmark*.

Para que os resultados tenham valor significativo, o mesmo processador porém sem DTM desempenha o papel de controle, servindo de referência para os valores lidos e permitindo uma visão mais completa dos impactos do mecanismo nas características gerais da unidade de processamento.

A área do chip pode ser estimada pela quantidade de células ou *slices* necessários para a gravação do circuito em FPGA. Essa métrica impacta diretamente no tamanho do circuito resultante, custo por unidade produzida usando tecnologias ASIC (*Application Specific Integrated Circuits*) e qual o número mínimo de células que um dispositivo lógico programável deve possuir para suportá-lo.

Além disso, a área do chip impacta indiretamente a potência dissipada e a latência do circuito. Indiretamente pois, apesar de não haver relação de causalidade direta, comumente existe uma correlação entre circuitos com uma área de chip maior e maior quantidade de elementos lógicos, o que pode acarretar em uma perda de potência maior.

A potência dissipada no circuito, ou seja, a energia consumida por este, é outro aspecto importante a ser considerado ao determinar a aplicabilidade da técnica e que pode ser medido diretamente com auxilio de equipamentos para medições elétricas. Além de possuir

uma relação direta com o calor produzido no circuito, explicada pelo efeito Joule, a fonte utilizada para alimentação do circuito deve possuir a capacidade de suprir a potência necessária. Isso é um fator determinante em aplicações de sistemas embarcados, nos quais muitas vezes busca-se o sistema com o menor consumo possível devido a estar embutido em outros sistemas, normalmente em um ambiente não idealmente preparado e estar ativado por longos períodos de tempo.

Também é possível perceber uma correlação entre área de chip e latência, apesar de mais fraca. O ponto que envolve ambos é o chamada caminho crítico, isto é, o maior caminho de dados possível dentro do circuito. Este caminho é o fator determinante para a determinação da latência do circuito e está diretamente associado com a quantidade de elementos pelos quais o sinal deve passar no circuito. Como cada elemento contido no circuito possui um *delay* ou atraso próprio, a quantidade de elementos do caminho crítico é diretamente proporcional ao atraso total do caminho, sendo este o fator que determina a latência do circuito. Portanto, a relação entre área de chip e latência do circuito depende do design deste, já que um projeto que explore características de paralelismo é capaz de possuir uma área maior sem necessariamente estender o caminho crítico.

A latência por sua vez define qual o tempo de ciclo ou *clock* mínimo para o circuito sem que haja perda de dados, portanto determinando o limite máximo para a frequência de trabalho do processador. Apesar de haverem outros fatores envolvidos no desempenho final do circuito, o tempo de ciclo é um fator fundamental. Com a implementação de DTM, observando como foi alterado o caminho crítico determina-se como foi alterado o tempo mínimo de *clock*, o que é possível através de análise do circuito resultante e medindo qual a frequência de *clock* máxima aplicável para o circuito.

Com a execução de programas de *benchmark* é possível determinar ganhos em tempo de execução de diferentes tarefas, cada programa executando um tipo de aplicação e quantificando o desempenho do sistema. Assim, juntamente com a informação sobre alteração na frequência máxima suportada, será possível ter uma visão mais completa sobre os impactos que a implantação de DTM em uma unidade de processamento possui no desempenho final do sistema.

3 Fundamentos históricos, teóricos e metodológicos

Referências

CHU, P. P. *RTL hardware design using VHDL: coding for efficiency, portability, and scalability*. Hoboken: John Wiley & Sons, 2006. Citado na página 6.

COSTA, A. T. da. *Explorando dinamicamente o reuso de traces em nível de arquitetura de processador*. 2001. Tese (Doutorado) — COPPE/UFRJ, Rio de Janeiro, 2001. Citado nas páginas 5, 6, 9 e 10.

GAISLER, A. Leon3 processor. *Nanoscale Integration and Modeling (NIMO) Group*, 2010. Citado na página 9.

MACK, C. A. Fifty years of moore's law. *IEEE Transactions on semiconductor manufacturing*, IEEE, v. 24, n. 2, p. 202–207, 2011. Citado na página 5.

SCHOEBERL, M. *JOP: A Java optimized processor for embedded real-time systems*. 2005. Tese (Doutorado) — Vienna University of Technology, 2005. Citado na página 9.

SILVA, B. R. *Memorização e reuso dinâmico de traços em uma arquitetura de processador Java*. 2006. Dissertação (Mestrado) — COPPE/UFRJ, Rio de Janeiro, 2006. Citado na página 9.

TANENBAUM, A. S.; ZUCCHI, W. L. *Organização estruturada de computadores*. [S.l.]: Pearson Prentice Hall, 2009. Citado na página 5.