UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

UMA HIERARQUIA DE MEMÓRIA PARA UM MODELO RTL DO PROCESSADOR RISC-V SINTETISÁVEL EM FPGA

PROPOSTA DE TRABALHO DE GRADUAÇÃO

Aluno: Ary Guedes Lins (agl2@cin.ufpe.br)
Orientador: Edna Natividade da Silva Barros (ensb@cin.ufpe.br)
Área: Arquitetura de Computadores

17/04/2017

Resumo

O mercado crescente da internet das coisas tem demandado por sistemas embarcados com baixo consumo de energia e de tempo real. Como solução, projetistas tem usado *hardware* personalizado para aplicações específicas, melhorando o desempenho e reduzindo o consumo de energia, porém aumentando a complexidade do projeto de *hardware*. O RISC-V, um ISA livre e aberto, independente de implementação e que possui já um *compiler toolchain* e binários de sistemas operacionais, é ideal para criar um *hardware* específico para uma aplicação, reduzindo o tempo e custo de projeto. Este trabalho propõe o projeto e implementação de uma hierarquia de memória para um modelo RTL existente do ISA RISC-V de 32 *bits* sintetizável em FPGA.

Introdução

Com o advento da internet das coisas o mercado de sistemas embarcados continuará com o crescimento exponencial das últimas décadas. Um sistema embarcado pode ser definido como um computador de propósito específico. Seus elementos são microprocessador, memória, dispositivos de entrada e saída, possivelmente sensores e atuadores, e um software aplicação específica **[8]**. As diversas aplicações têm demandado cada vez mais por *Systems on a Chip (SoC)* com desempenho em tempo real e baixo consumo de energia.

Tendo isso em vista os *SoCs* estão incorporando *hardwares* personalizados para aplicações específicas, com diferentes frequências de *clocks* no mesmo *chip,* com objetivo de reduzir o consumo de energia e melhorar o desempenho **[10]**. Isso aumenta a complexidade no desenvolvimento de *hardware*, demandando por metodologias e ferramentas que possibilitem o desenvolvimento de *chips* de forma mais ágil.

Surgiu então o RISC-V **[1]**, uma especificação de um conjunto de instruções *(ISA)* aberto e livre. Foi desenvolvido por um grupo de pesquisa da Universidade da Califórnia – Berkeley, e está disponível nas versões de 32, 64 e 128 *bits*. O RISC-V permite diferentes implementações de *hardware*, sendo ideal para customizar criando aceleradores de *hardware* para aplicações específicas, que podem não só aproveitar a estrutura base da *CPU*, mas também a *toolchain* dos compiladores e binários de sistemas operacionais desenvolvidos para o ISA RISC-V.

Neste contexto, o trabalho **[6]** propôs uma implementação do *ISA* RISC-V de 32 bits capaz de executar operações de inteiros usando um *pipeline* de 3 estágios. Esta *CPU* foi projetada para ser usada como base para a criação de aceleradores de *hardware* para aplicações específicas. O projeto foi implementado no nível de abstração *RTL (Register Transfer Level)* usando SystemVerilog **[9]** uma linguagem de descrição de hardware *(HDL)*.

Uma limitação atual da *CPU* desenvolvida em **[6]** é que essa possui apenas duas memórias de acesso direto, uma de instrução e outra de dados. Existem, no entanto, aplicações que demandam mais espaço de armazenamento (e.g. aplicações de processamento de imagem), gerando uma necessidade por um sistema mais robusto, com mais espaço de armazenamento e perca de desempenho mínima. Levando em conta que o simples aumento na capacidade dessas memórias diretamente acoplada na *CPU* implica num aumento no tempo de acesso, gerando então um gargalo de desempenho, este trabalho propões o projeto e implementação de uma hierarquia de memória **[4]** para esta *CPU*, conceito que será explicado a seguir.

Nos dispositivos de memória de um computador há sempre um *trade off* entre velocidade, armazenamento e custo. Sendo uma tarefa muito complexa otimizar um ou dois desses fatores sem piorar outro. Para amenizar essa diferença de velocidades entre a *CPU* e as unidades de armazenamento, usualmente utilizamos um sistema de

hierarquia de memórias **[4],** em que as memórias mais rápidas e menores estão mais próximas da *CPU* (em um nível mais alto), enquanto memórias mais lentas e maiores estão mais distantes (em um nível mais baixo). Nesse sistema cada nível memória guarda uma cópia de parte do conteúdo da memória de nível logo abaixo se baseando nos princípios da localidade espacial e temporal **[5]**.

Objetivos

Este trabalho então propõe a implementação de um modelo *RTL* em SystemVerilog de uma hierarquia de memória contendo uma memória principal e uma memória *cache* de nível 1 e sua integração a *CPU* desenvolvida por **[6]**.

Nesta hierarquia a memória cache irá trabalhar com endereços reais de 32 bits, portanto não sendo necessário a implementação de um *Translation look-a-side Buffer (TLB)*. A *cache* usará a técnica de atualização *Write Back,* mais indicada para sistemas embarcados, pois consome menos energia **[5]**, e técnica *Write-Allocate* para lidar com *Write-Misses*. A *cache* deverá ser parametrizável na pré-síntese podendo variar os seguintes parâmetros:

Parâmetro	Valores
Tamanho do bloco	4, 8, 16, 32, 64, ou 128 *bytes*
Associatividade	*Direct Mapped*, *2-Way, 4-Way* ou *8-Way*
Tamanho da *cache*	2kb até 16kb
Algoritmo de Substituição	LRU, Randômico e FIFO

Tabela 1 - Parâmetros variáveis na pré-síntese

Metodologia

Para implementação da hierarquia de memória proposta neste projeto primeiramente será definido um modelo comportamental em linguagem C que será usado para comparação com o modelo no nível de abstração RTL a ser desenvolvido usando a *HDL* SystemVerilog. Para a simulação deste modelo em SystemVerilog será utilizado o software Questasim **[2]**.

Assim que o modelo for validado será realizada a integração na pipeline da *CPU RISCV* desenvolvida por **[6]** então será realizada uma nova validação funcional baseada na simulação das instruções de *load* e *store* do processador.

Após estes passos o modelo estará pronto para ser sintetizado para uma *Field Programmable Gate Array (FPGA).* Um circuito integrado configurável pós fabricação, capaz de implementar sistemas digitais complexos, inclusive os chamados *Soft-Cores* **[8]**, que são *CPUs* implementados em *FPGAs*.

Para essa tarefa utilizaremos o software Quartus **[3]** que faz a síntese automática do modelo estrutural em RTL para o modelo físico usado para programar a FPGA. Então usaremos esse mesmo software para programar a FPGA do modelo Cyclone III **[7]**, então será feita uma nova validação em funcional e em *timing,* e por último a prototipação na plataforma, finalizando o escopo deste trabalho.

Cronograma

	Período																	
Atividade	Março				Abril				Maio				Junho				Julho	
Revisão bibliográfica	X	X	X	X	X	X												
Estudo do ISA RISCV	X	X	X	X	X	X												
Escrita da proposta			X	X	X	X												
Implementação do modelo de referência					X	X	X	X										
Implementação do testbench					X	X	X	X										
Implementação RTL							X	X	X	X								
Validação Funcional									X	X								
Integração na Plataforma											X	X	X	X				
Síntese													X	X				
Validação Funcional, em Timing e Prototipação													X	X				
Escrita do TG													X	X	X	X		
Preparação da apresentação																X		

Referências

[1] Waterman, A. et al. The RISC-V instruction set. In: IEEE HOT CHIPS 25 SYMPOSIUM (HCS), 2013. Anais. . . [S.l.: s.n.], 2013. p.1–1.

[2] Mentor. Questa Simulator. 2013.

[3] Altera. Quartus II. 2013.

[4] David A. Patterson, John L. Henessy. “Computer organization and design: the hardware/software interface”. 2009

[5] David A. Patterson, John L. Henessy. “Computer Architecture: a quantitative approach”. 2009

[6] Ogg, V. O. Modelo RTL do processador RISCV sintetizável em FPGA, 2016. Trabalho de graduação – Universidade Federal de Pernambuco, Centro de Informática. Programa de graduação em Engenharia da Computação.

[7] Altera Corporation, “Cyclone III Device Handbook, Volume 1”, San Jose, Altera Corporation, agosto 2012.

[8] Tong, J. G.; Anderson, I. D. L.; Khalid, M. A. S. Soft-Core Processors for Embedded Systems. In: International Conference On Microeletronics, 2006. Anais. [S.l.: s.n.], 2006. p.170–173.

[9] Acellera Organization. SystemVerilog 3.1a Language Reference Manual. 2004.

[10] Lee, Y. et al. An Agile Approach to Building RISC-V Microprocessors. IEEE Micro, [S.l.], v.36, n.2, p.8–20, Mar 2016.

Possíveis Avaliadores

1. Adriano Sarmento
2. Manoel Eusébio

Assinaturas

Recife, ___ de _____ de _____

_____

Nome

(Aluno)

_____

Nome

(Orientador)