UNIVERSIDADE ESTADUAL DE FERA DE SANTANA
DEPARTAMENTO DE TECNOLOGIA/ÁREA DE ELETRÔNICA E SISTEMAS
TEC499 – MI – SISTEMAS DIGITAIS

Problema #1 – Linguagem de Máquina e Microarquitetura

1 Tema

Implementação e teste de um processador *softcore*.

2 Objetivos de Aprendizagem

Ao final da realização deste problema o discente deve ser capaz de:

- Programar em linguagem de baixo nível (Assembly);
- Identificar conjunto de instruções em processadores;
- Compreender o fluxo de execução interna de um mono-processador RISC;
- Implementar circuito utilizando Linguagem de Descrição de Hardware (Verilog);
- Identificar características de operação e restrições em projetos de sistemas digitais;
- Analisar a implementação de circuitos sob o ponto de vista de dispositivos FPGA;
- Compreender a integração a nível de microarquitetura.

3 Contexto

Na matemática, computação e telecomunicação, a detecção e correção de erros é um assunto de grande relevância na manutenção da integridade dos dados. Em um sistema de comunicação pode-se dizer que é comum a ocorrência de erros, pois diversos fatores interferem na troca de mensagens a todo instante de um local para outro. Os erros podem ser causados por interferências eletromagnéticas, alta densidade de integração dos circuitos integrados, envelhecimento de componentes, curto-circuitos, ou quaisquer outros fatores que afetem as mensagens, fazendo com que, por exemplo, um "0" seja enviado, e alongo do processo de comunicação, acabe sendo transformado em "1", ou seja, receptor recebe informação diferente daquela que foi enviada (adaptado de https://pt.wikipedia.org/wiki/Detecção_e_correção_de_erros).

Esses erros podem ser detectados e corrigidos empregando-se códigos de detecção e correção de erros. O CRC (*Cyclic Redundancy Check*) é uma das técnicas mais usadas para detecção de erros na transmissão de dados digitais. No método CRC são adicionados *check bits*, também chamados de *checksum*, os quais são anexados à mensagem que irá ser transmitida. O receptor pode assim avaliar se os *check bits* estão de acordo com os dados transmitidos, e determinar, com um certo grau de certeza, se ocorreram erros na transmissão ou não (retirado de http://logos.cs.uic.edu/366/notes/ErrorCorrectionAndDetectionSupplement.pdf).

Sob o ponto de vista da implementação de sistemas computacionais, destacam-se duas abordagens que diferem no que diz respeito a desempenho, potência e flexibilidade: **circuitos dedicados** (ASIC - *Application Specific Integrated Circuit*) e **dispositivos lógicos programáveis** (PLD - *Programmable Logic Devices*). Atualmente o projeto de sistemas baseados em PLD vem sendo bastante difundido, principalmente em função da evolução do processo de fabricação dos *chips* FPGA (*Field Programmabla Gate Array*). Dispositivos FPGA modernos são capazes de implementar circuitos complexos, equivalentes a mais de 100.000 portas lógicas e têm, cada vez mais, substituído projetos baseados em *standard cells*.

O NIOS II é um processador *softcore* RISC de 32 bits com arquitetura Harvard, desenvolvido pela Altera (atual Intel FPGA). Um processador *softcore* é uma implementação de um processador descrito em linguagem de hardware, que pode ser customizado e sintetizado em um FPGA ou ASIC. Uma vantagem evidente no uso de *softcores* está na flexibilidade, pois eles possibilitam a fácil conexão com outros periféricos, alteração do conjunto de instruções e, o mais evidente, alteração das estruturas internas do processador, tais como tamanho da memória cache, priorização de interrupções, dentre outros (retirado de https://www.embarcados.com.br/altera-nios-ii).

4 Problema

A empresa **IP-SoC** atua no desenvolvimento de IP-*cores* digitais licenciáveis para projeto de sistemas computacionais há mais de dez anos. Durante sua trajetória de mercado, vários produtos foram desenvolvidos nas áreas de processamento gráfico, microprocessadores de propósito geral e processamento digital de sinais. Nos últimos anos, a empresa tem se destacado no mercado de semicondutores com a venda dos seus *cores* licenciáveis.

Atenta às mudanças de paradigma no projeto de sistemas, a IP-SoC está de olho no mercado de plataformas embarcadas projetadas a partir de dispositivos FPGA. Neste sentido, a empresa começou uma força tarefa para apropriação do conhecimento acerca das soluções tecnológicas que podem ser exploradas com o uso desse tipo de plataforma, a começar pelas opções de núcleos de processamento geral.

Sua equipe de projeto foi destacada para desenvolver um sistema digital que calcule o CRC-32 de uma sequencia de dados com o tamanho de 1KB. O sistema deverá ser implementado em FPGAs ALTERA, usando o processador NIOS. Além de conhecer o funcionamento básico do processador, você deve avaliar o nível de complexidade de programação e as limitações do dispositivo, a partir do desenvolvimento de uma série de códigos em linguagem assembly deste processador.

5 Produto

No prazo indicado no cronograma a seguir, cada equipe deverá apresentar:

1. A descrição do processador em Verilog e demais elementos utilizados para teste e validação do funcionamento do *core*;
2. Código Assembly para calculo do CRC;
3. Um relatório técnico contendo informações acerca das etapas de síntese lógica e física do processador, incluindo, mas não limitando-se a: (i) indicação do caminho crítico do circuito; (ii) área total ocupada pelo circuito em função dos elementos internos do dispositivo FPGA (LEs, LABs, FFs, Memória, DSPs, etc.); e (iii) uma análise da taxa de transferência (*throughput*) de dados que o seu circuito é capaz de atingir.

6 Cronograma

Semana	Data	Descrição
01	19/03/2018 (seg)	Apresentação do Problema #1
	22/03/2018 (qui)	Sessão Tutorial #1
02	26/03/2018 (seg)	Sessão Tutorial #2
	29/03/2018 (qui)	Feriado — Semana santa
03	02/04/2018 (seg)	Sessão Tutorial #3
	05/04/2018 (qui)	Lab0 – Introdução à plataforma de desenvolvimento
04	09/04/2018 (seg)	Sessão Tutorial #4
	12/04/2018 (qui)	Lab1 – Introdução à plataforma de desenvolvimento
05	16/04/2018 (seg)	Sessão Tutorial #5
	19/04/2018 (qui)	Lab2 – Layout físico de circuitos FPGA (1)
06	23/04/2018 (seg)	Feriado — Micareta
	26/04/2018 (qui)	—
07	30/04/2018 (seg)	Ponto facultativo
	03/05/2018 (qui)	Entrega do Problema #1

7 Avaliação

Para avaliar o envolvimento do grupo nas discussões e na apresentação, o tutor poderá fazer perguntas sobre o funcionamento de qualquer componente, a qualquer membro, tanto nas sessões tutoriais quanto na apresentação.

Formato da Avaliação

A nota final será a composição de 3 (três) notas parciais:

Desempenho Individual	nota de participação individual nas sessões tutoriais, de acordo com o interesse e entendimento demonstrados pelo aluno, assim como sua assiduidade, pontualidade e contribuição nas discussões; **Peso: 4,0 pontos**.
Documentação	nota atribuída à cada grupo, referente ao relatório técnico; **Peso: 3,0 pontos**
Processador	nota atribuída à cada grupo, oriunda da análise da implementação em Verilog do processador, incluindo sua descrição funcional e estruturas de teste/validação; **Peso: 3,0 pontos**.

8 Orientações

Geral

O atendimento ao que está sendo solicitado somente será possível com a organização do grupo, visitas aos laboratórios e trabalho de pesquisa, em fontes confiáveis, fora do horário das reuniões tutoriais. As reuniões tutoriais deverão ser usadas para análise, explanações sobre o que foi estudado, levantamento de hipóteses e para tomadas de decisão. É recomendado ainda que todos os membros do grupo tutorial mantenham-se atualizados quanto à possíveis alterações no cronograma, ou nos requisitos do problema por meio de acesso ao sítio do módulo, acessível em: http://sites.ecomp.uefs.br/tec499.

Nós encorajamos fortemente que os grupos trabalhem juntos, no sentido da troca ideias acerca das suas propostas de solução. A melhor forma de desenvolver novas habilidades é comparar hipóteses e discutir aspectos de projeto com seus colegas e professores (inclusive com o seu tutor). Todavia, sob nenhuma circunstância, compartilhe seu código-fonte.

Versionamento

Um dos critérios de avaliação utilizados em **TEC499** é a análise do projeto final por meio de uma ferramenta de controle de versionamento de código-fonte. Toda avaliação será conduzida a partir do conteúdo armazenado neste repositório, sendo de inteira responsabilidade do grupo a manutenção da sua integridade.

Cada grupo deve indicar em um arquivo `README` as instruções de compilação síntese e execução dos testes. Além disso, recomenda-se que os testes sejam especificados em uma pasta exclusiva, contendo um novo arquivo `README` especificando a estrutura e descrevendo, de forma, objetiva, cada um dos testes.

Documentação Técnica

A documentação deve seguir o modelo adotado na disciplina, baseado no ipPROCESS e apresentar (não exclusivamente) os seguintes elementos:

- Uma introdução contendo uma descrição geral do propósito do documento e como ele está organizado, além de uma lista de possíveis acrônimos e abreviações utilizadas ao longo do mesmo;
- Uma visão geral da arquitetura do processador, de forma textual, usando elementos gráficos (ou melhor, ambos), incluindo ainda as principais características e requisitos funcionais e não funcionais;
- Descrição completa e detalhada da arquitetura do conjunto de instruções do processador;

9 Recursos

Visite a página da disciplina para ter acesso ao acervo de documentos e modelos de documentação indicados para uso no decorrer do semestre. Para teste do código Assembly sugere-se o uso do *software open source* JNIOSEmu, disponível online.

Referências

BROWN, S. D.; VRANESIC, Z. G. *Fundamentals of Digital Logic with Verilog Design*. 2nd. ed. [S.l.]: McGraw-Hill Higher Education, 2008.

HARRIS, D. M.; HARRIS, S. L. *Digital Design and Computer Architecture*. 2. ed. USA: Elsevier, 2013. ISBN 9780123944245.

KILTS, S. *Advanced FPGA Design: Architecture, Implementation, and Optimization*. [S.l.]: John Wiley & Sons, 2007.

PATTERSON, D. A.; HENNESSY, J. L. *Arquitetura de Computadores*. 3. ed. Brasil: Editora Elsevier, 2014. 709 p. Impresso. ISBN 9788535235852.

SPEAR, C. *System Verilog for Verification : A Guide to Learning the Testbench Language Features*. 2nd. ed. [S.l.]: Springer, 2008.