UNIVERSIDADE DE BRASÍLIA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

116394 ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

Projeto da Disciplina: MIPS Pipeline em VHDL

OBJETIVO

O projeto da disciplina consiste no desenvolvimento de uma versão do processador MIPS Pipeline em FPGA, utilizando a linguagem VHDL.

A plataforma de desenvolvimento é Altera. Podem ser utilizados os kits de desenvolvimento DE2-70, disponíveis no laboratório de hardware do CIC.

As ferramentas utilizadas para o desenvolvimento serão o Quartus II e ModelSim-Altera.

DESCRIÇÃO

O diagrama esquemático da arquitetura do MIPS Pipeline é apresentado na figura 1.

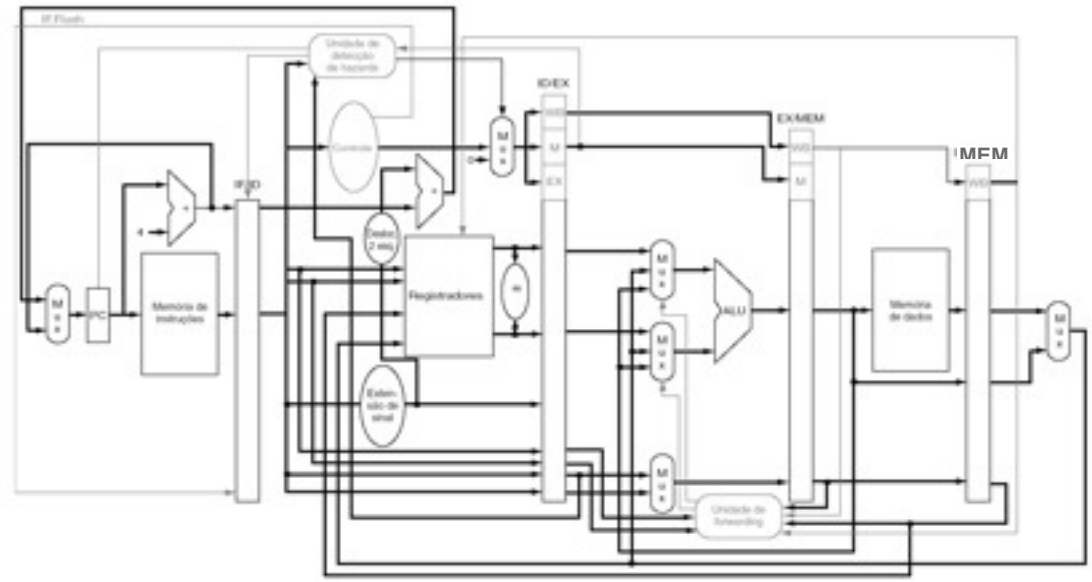

Figura 1. MIPS Multiciclo.

A implementação VHDL consiste na descrição de cada módulo e sua interligação através de sinais.

A parte operativa do MIPS é 32 bits, ou seja, os dados armazenados em memória, os registradores, a ULA, os somadores, as instruções e as conexões utilizam 32 bits.

Devido às restrições de memória das placas FPGA, as memórias de instruções e dados devem ser dimensionadas para suportas pequenos programas de teste. Sugere-se utilizar apenas 8 bits de endereço, de forma a prover um espaço de endereçamento de 256 palavras tanto para o programa quanto para os dados.

A versão do MIPS pipeline a ser implementada inclui a deteção de conflitos de dados no *pipeline* e sua solução por hardware através do adiantamento de dados. Além do adiantamento de dados, a deteção de conflitos deve introduzir uma bolha no caso da instrução LOAD WORD. Não é necessário tratar conflitos de controle.

Módulos principais:

- **PC**: contador de programa. É um registrador de 32 bits. Entretanto, pelas restrições de memória adotadas, apenas o número de bits necessário deve ser enviado à memória de instruções, sendo o restante ignorado;
- **Memória de Instruções (MI)**: memória ROM que armazena o código a ser executado pelo processador. As instruções são de 32 bits. O espaço de endereçamento é reduzido (8 bits). Deve ser iniciada com um programa gerado através do MARS, e carregado em arquivo ".mif". Cada endereço da memória armazena uma instrução de 32 bits. Idealmente, a memória deve funcionar como um bloco combinacional para leitura, ou seja, necessita-se apenas do endereço para ler a instrução/dado, sem sinais adicionais de controle. Essa memória não permite o endereçamento a byte. Desta forma, se forem utilizados 8 bits de endereço, deve-se utilizar os bits 2 a 9 do PC como endereço de instrução.
- **Memória de Instruções (MD)**: armazena dados que podem ser lidos e/ou escritos na memória. Endereços de 8 bits e dados de 32 bits. A memória é escrita na subida do relógio, quando o sinal de controle EscreveMem estiver acionado. O sinal de leitura LeMem não é estritamente necessário, faz com que o conteúdo da posição de memória endereçada seja colocado na saída de dados. Se não houver sinal LeMem, basta fornecer o endereço que o dado é lido, de forma similar à ROM.
- **Banco de Registradores (BREG)**: é constituído por 32 registradores de 32 bits. O registrador índice zero, BREG[0], é uma constante. Sua leitura retorna sempre zero, e não pode ser escrito. O BREG tem duas entradas de endereços, permitindo a leitura de 2 registradores de forma simultânea. Uma terceira entrada de endereço é utilizada para selecionar um registrador para escrita de dados. A escrita de um dado em registrador ocorre na transição de subida do relógio.
- **Unidade Lógico-Aritmética (ULA)**: opera sobre dados de 32 bits. Provê o resultado em 32 bits, juntamente com o sinal **ZERO**, que indica que o resultado da operação realizada é zero.
    - Operações implementadas na ULA:
        - ADD, SUB, AND, OR, XOR, SLT, NOR, SLL, SRL, SRA
- **Multiplexadores 2 para 1**: são utilizados 4 multiplexadores com 2 entradas de 32 bits e uma saída de 32 bits;
- **Somadores**: são utilizados 2 somadores de 32 bits para operar com endereços;

INSTRUÇÕES A SEREM IMPLEMENTADAS

- LW, SW, ADD, SUB, AND, OR, NOR, XOR, SLT, ADDI, SLL, SRL, SRA, J, BEQ, BNE

VERIFICAÇÃO DO MIPS PIPELINE

O processador implementado deve ser capaz de executar um código gerado pelo MARS, utilizando o modelo de memória compacto.

ENTREGA E APRESENTAÇÃO

A implementação em FPGA deverá ser apresentada em data a ser combinada com o professor.

Entregas:
- Código VHDL do projeto
- Relatório descrevendo a implementação

Entregar no Moodle em um arquivo compactado.

Prazo de entrega: 10/07/15