Projeto Final - OAC
Relatório de Implementação

Alex Nascimento Souza - 15/0115474
Guilherme da Silva Fontes Lopes - 15/0128215

Objetivo

A implementação desse trabalho tem como objetivo representar o processador MIPS em linguagem de Hardware VHDL. A implementação do processador é baseada no método pipeline, que é uma forma de processar instruções de maneira paralela com visando aumentar o desemepenho do processador

Implementação.

MIPS

Inicialmente, o projeto foi dividido em vários módulos, os quais representam os componentes do circuito que compõe o processador. Cada componente foi interligado por fios, que são caracterizados por sinais na linguagem utilizada.

O método de pipeline do MIPS possui uma ideia semelhante ao processador com método multiciclo, apresentando técnicas para aumento de desempenho de uma única tarefa em um único processador. Diferenciando-se pelo fato de executar as etapas paralelamente, o pipeline faz o uso de registradores, chamados de registradores de pipeline, que armazenam sinais, em um ciclo de clock, de cada etapa. Vale a pena ressaltar que o pipeline é dividido em cinco etapas: IF, ID, EX, MEM e WB e que os sinais armazenados pelos registradores de pipeline só são repassados após a batida de clock.

Componentes

Breg

Representa os 32 registradores do MIPS. As operações que podem ser realizadas sobre o banco de registradores são leitura e escrita dos mesmos. A escrita ocorre no flanco de descida do clock. Além disso, ler e escrever na mesma batida do clock não funciona, uma vez que a leitura seguida de uma escrita retorna o valor antigo (isto é, o último valor a ser escrito) do registrador. O registrador 0 ($zero) não pode ser escrito.

PC

O contador do programa é um registrador que armazena o valor de PC e só muda no flanco de subida do clock.

Somador

O Somador é um componente que soma duas entradas de 32 bits. Esse componente foi implementado sem incluir os sinais de overflow e de carry, visto que os mesmos não foram utilizados nas instruções implementadas.

ULA

A Unidade Lógica Aritmética (ULA) realiza as operações de ADD,SUB, Shifts e Lógicas (Or, Nor, Xor, And). Os sinais de overflow e de carry não são utilizados pela ULA.

Comparador

O módulo do comparador é utilizado para gerar sinais para detecção das instruções BNE e BEQ. Ele recebe dois vetores de 32 bits e retorna o sinal em high caso eles sejam iguais.

Conversor 7 Segmentos

O Conversor 7 Segmentos é responsável por converter o dado que queremos expor nos displays da FPGA para o display de 7 segmentos. A FPGA possui 8 displays.

Mux 4 Entradas

O Multiplexador de 4 entradas é utilizados nas etapas:

IF, para escolher qual entrada será carregada para o PC, seja ela para saltos, desvios ou execução sequencial padrão do programa.

ID, para decidir qual será registrador enscrito (Entrada WR do Breg) e WB para decidir qual dado será escrito no registrador explicitado em WR.

Mux 2 Entradas

O único Multiplexador 2-para-1 utilizado serve para decidir qual caso da instrução JALR estaremos utilizando, seja a instrução com apenas um argumento ou a instrução com dois argumentos.

Memória de Instrução

A memória de instruções possui todas as instruções do programa a ser executado, as quais devem ser escritas manualmente em um arquivo separado(mem_inst.mif). O endereço das instruções possui 8 bits, devido a arquitetura da FPGA utilizada.

Memory Data

A memória de dados possui todos os dados armazenados ao longo da execução do programa. As únicas instruções capazes de modelar esses dados são LW e SW, que acessam os mesmos por meio dos endereços gerados pela instrução. Vale a pena ressaltar que a memória de dados possui apenas 8 bits para endereçamento e saída de 32 bits.

Mipspipeline

Módulo principal do processador, é o componente que interliga todos os outros e implementa a lógica do ***pipeline***. Tal módulo é utilizado para conectar todos os fios (interpretados como sinais) entre os registradores de pipeline, multiplexadores e os dispositivos presentes no circuito do processador.

Registradores de Pipeline

IF/ID

O registrador IF/ID armazena o valor de PC+4 e a instrução que está no endereço atual de PC.

ID/EX

O registrador ID/EX armazena o valor de PC+4, os registradores RS, RT, RD, os campos Shamt e KTE estendido de sinal.

EX/MEM

O registrador EX/MEM armazena o valor de PC+4, o resultado da ULA e o índice do registrador que poderá ser escrito ao final da etapa WB.

MEM/WB

O registrador MEM/WB armazena o valor de PC+4, o endereço de memória de dados e o dado guardado no próprio endereço de memória.

Controle

Os dados enviados pelo controle dependem diretamente da instrução que é buscada na memória de instruções. Sendo assim, são gerados sinais que controlam todos os multiplexadores, exceto o multiplexador presente na etapa IF e o multiplexador presente em ID para determinar o endereço utilizado na instrução JAL, a partir dos três conjuntos presentes nos registradores de pipeline: WB, MEM e EX.

A etapa ID gera os bits de controle para WB, MEM e EX no registrador de pipeline ID/EX que passam os bits para os conjuntos WB e MEM presentes no registrador de pipeline EX/MEM, que por sua vez, passa os bits para o conjunto WB presente no registrador MEM/WB.

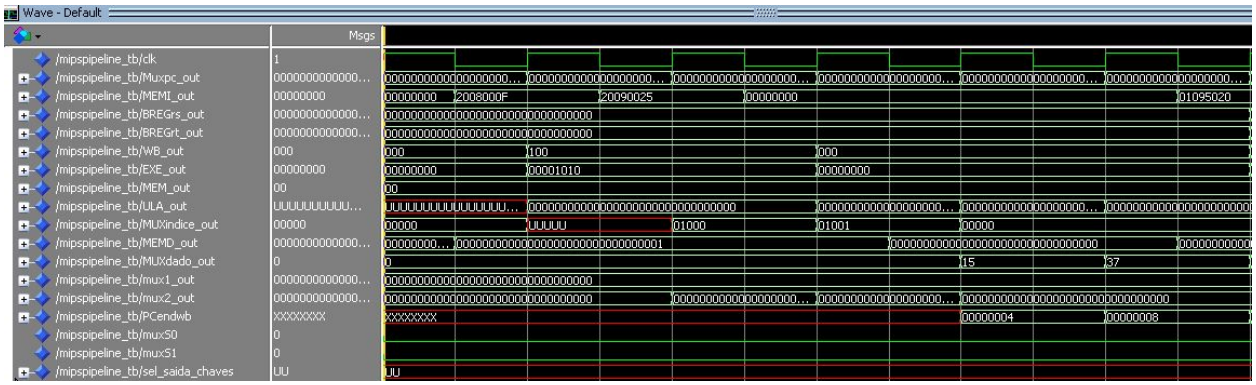
Testes e Resultados

Simulador

Para realizar os testes, foi necessário observar os sinais mais importantes de cada etapa de pipeline colocando-os no testbench do módulo principal denominado **mipspipeline_tb**.

Sendo assim, todas as instruções exigidas (LW, SW, ADD, SUB, AND, OR, NOR, XOR, SLT, SLTi, ADDi, J, BEQ, BNE, JAL, JALR, JR, LUI) foram executadas, de acordo com um programa escrito no ***MARS,*** separadamente observando o resultado dos sinais designados no testbench.

Os resultados obtidos analisando o simulador ***Modelsim*** foram os esperados e estavam de acordo com o esperado em cada etapa específica do ***pipeline***. Alguns resultados são demosntrados a seguir:



FPGA

Para realizar a implementação no FPGA, foi necessário utilizar um componente adicional de conversor de 7 segmentos e implementar um arquivo **.qsf**, responsável pelo mapeamento das saídas do pipeline para a placa. O programa utilizado para teste .asm foi implementado no MARS com todas as instruções detalhadas anteriormente obtendo-se o resultado esperado nos quatro sinais que foram mapeados para as chaves da FPGA: Memória de Instruções, Memória de Dados, Program Counter e saída da ula no estágio de execute.