UNIVERSIDADE FEDERAL DE VIÇOSA

CAMPUS FLORESTAL

PABLO FERREIRA - 3480
SAMUEL SENA - 3494

CAMINHO DE DADOS MIPS PIPELINE

PARTE II - FPGA

FLORESTAL
2019

INTRODUÇÃO:

O trabalho a seguir tem como objetivo a implementação do trabalho Caminho de dados MIPS com Pipeline em FPGA.

Todos os arquivos de desenvolvimento, códigos fonte e de síntese encontram-se na pasta “MIPS_PIPELINE_FPGA”. Os arquivos de cada módulo auxiliar encontram-se ainda separados de maneira organizada nas pastas “Modulos” e “Pipeline”.

As configurações de entrada e saída na placa são: botão “*KEY3*” para reset (com led “*LEDG6*” para a indicação de pressionamento do mesmo), botão “*KEY2*” para visualizar próximo resultado (com led “*LEDG4*” para indicação de pressionamento do mesmo),displays de sete segmentos para exibição de resultados, “*LEDG0*” indicando pulsos de *Clock* .

INSTRUÇÕES DE UTILIZAÇÃO:

Inicialmente abra o arquivo de projeto “MIPS_PIPELINE_FPGA.*qpf*” no programa “Quartus Prime”, caso ache necessário, abra em seguida o arquivo “MIPS_PIPELINE_FPGA.*v*”. O projeto já se encontra inicialmente sintetizado, no entanto, caso deseje não há problemas em refazer a síntese. Em seguida a programação do FPGA já pode ser efetuada.

Logo de inicio, é necessário apertar o botão “*KEY3*” da placa para que o “Reset” inicial no caminho de dados seja dado. Após cerca de 2 segundos, todas as instruções já se encontram processadas, então para dar inicio a visualização dos resultados da ALU, o botão “*KEY2*” deverá ser pressionado e assim resultado por resultado será exibido nos 8 displays sempre que o botão for pressionado novamente. Ao final da contagem dos resultados obtidos, o numero “99999999” será exibido (indicando o fim dos resultados, como ilustrado na figura 1), caso deseje recomeçar a visualizar do inicio, basta apertar novamente o botão “*KEY2*”.

Figura 1

Fonte: FPGA DE2-115

Dentre as instruções arbitrariamente configuradas para rodar no caminho de dados, seus resultados obtidos na ALU serão exibidos no display de 7 segmentos. Lembrando que apesar de alguns resultados serem iguais, a instrução de cada um deles é diferente, vale lembrar também que todas as instruções e valores de registradores configuradas originalmente podem ser mudadas para os devidos testes desejados. Alguns resultados encontram-se ilustrados nas figuras 2, 3 e 4.

Figura 2

Fonte: FPGA DE2-115

Figura 3

Fonte: FPGA DE2-115

Figura 4

Fonte: FPGA DE2-115

DESENVOLVIMENTO:

O desenvolvimento da implementação Caminho de dados MIPS com Pipeline em FPGA consistiu basicamente na adaptação da primeira parte do trabalho em módulos sintetizáveis. Partindo do conceito que as entradas e saídas se tornam físicas e o *Clock* automático, algumas modificações necessárias foram efetuadas.

Para uma melhor experiência e visualização dos resultados no display de sete segmentos, três novos módulos auxiliares foram criados, o primeiro grava todos os resultados obtidos da *ALU*, dessa forma a visualização de cada resultado é possibilitada de forma mais fácil, o segundo novo módulo recebe cada um desses resultados na forma de binário de 32 bits e em seguida o converte para a visualização em sete segmentos. E por último, o terceiro novo módulo tem como intuito dividir o *Clock* nativo da placa, tornando-o assim mais apropriado para se trabalhar. Além destas adições para uma melhor implementação em FPGA, modificações como adicionar entradas de "*Reset*" em módulos já presentes também foram realizadas.

Todo o caminho de dados apresentado tem como base a versão sem pipeline do caminho de dados MIPS apresentado na disciplina de Organizações de Computadores I (CCF252). Logo, para transformar o caminho de dados anterior em uma nova versão com pipeline, novos módulos responsáveis por funções especificas do pipeline foram criados, dentre eles estão: registradores de pipeline (Presentes na pasta Pipeline/Regs_Pipeline) e unidades de detecção de Harzard e Forwarding (Presentes na pasta Pipeline/Hazard_Forwarding). Além disso novos tipos de multiplexadores foram criados, tendo como única diferença o tamanho em bits de suas entradas e saídas. Com a adesão de novos módulos que estão envolvidos em todo o controle do caminho de dados, todas as ligações entre as interfaces de cada módulo necessitaram ser refeitas, porem ao final, todo o caminho de dados possuía as mesmas funcionalidades de sua base sem pipeline.

CONSIDERAÇÕES FINAIS:

O respectivo trabalho foi de excelente ajuda para entender melhor de forma prática o funcionamento do caminho de dados, além de propiciar uma oportunidade/necessidade de se dominar melhor a linguagem de descrição de hardware Verilog.

Agradecimentos ao Professor José Augusto Nacif, por todas as dúvidas tiradas e exemplos dados em sala de aula.