Trabalho 5 – Organização e Arquitetura de Computadores
Banco de Registradores no Risc-V

Aluno: Gustavo Pereira Chaves
Matrícula: 19/0014113
Turma: C

1. Objetivos

Este trabalho tem por objetivo a simulação do banco de registradores do RISC-V, com barramentos de escrita e leitura ativados de forma síncrona por um clock.

2. Documentação

Para a implementação do problema, foram desenvolvidos dois arquivos em vhdl, o trabalho5.vhd, que armazena propriamente o módulo desenvolvido que simula o banco de registradores, e o arquivo testbench.vhd, que testa o programa.

2.1. trabalho5.vhd

Nesse arquivo foi criado o componente XREGS, que tem como parâmetros de entrada o clock (clk), o habilitador de escrita (wren), reset (rst), rs1, rs2, rd (endereços do registrador), data (dado a ser escrito), e como saída os barramentos ro1 e ro2. Para representar o banco de registradores foi criado um tipo de dado "registradores", que nada mais é que um array de 32 posições, em que cada posição é representada por um vetor de 32 bits.

Já com relação a implementação da arquitetura foi criado um processo ativado na borda de subida do clock, realizando a operação de acordo com as flags ativadas.

Caso **rst** esteja em ativo alto, todo o banco de registradores é zerado.

Caso **wren** esteja ativado, o valor do barramento **data** é copiado para o endereço do registrador indicado pela porta **rd**. Vale ressaltar que em um RISC-V o registrador zero não pode ser alterado, possuindo sempre o valor 0. Dessa forma, uma alternativa encontrada para obter esse comportamento é realizar uma condicional, em que caso o valor de rd seja igual a zero, a operação de escrita não deve ser realizada.

E por fim, caso nenhuma das duas flags estejam ligadas, é lido o valor indicado pelos endereços **rs1** e **rs2** nos barramentos de saída **ro1** e **ro2** respectivamente.

2.2 testbench.vhd

Já no testbench, é declarado o componente **XREGS**, bem como os sinais auxiliares para a interconexão com o processo de estímulo, que deve testar o módulo desenvolvido.

Inicialmente é necessário um clock para o funcionamento do componente. Dessa forma, cria-se um sinal **clk** com valor inicial 0, que tem o seu valor negado a cada 5 ns, gerando então uma forma de onda quadrada que simula o clock do processador.

Tratando do processo de estímulo em si, em todos os casos é realizada uma escrita em um registrador e em seguida uma leitura, utilizando a função **assert** para comparar o resultado obtido com o esperado.

É criado então um loop que percorre todo o vetor de registradores, escrevendo o valor da variável de iteração **(i+1)** no barramento data e realizada a posterior leitura e teste nos barramentos **ro1** e **ro2**. Para o caso específico da variável **i = 0**, deve-se testar se o valor do registrador continua zero após a tentativa de escrita.

Por fim, deve-se testar a função reset. Assim, após ligada a flag **rst**, é feita a verificação um a um de todos os registradores, que devem apresentar o valor lido "0".