MC542 - 2S 2011

Laboratório 01

Cristiano J. Miranda RA 083382

Descrição

Implementação de um banco de registradores(register file), utilizando a estrutura:

```
Entity RF is
	Generic(W : natural := 32);
	port(A1 : in std_logic_vector(4 downto 0);
		A2 : in std_logic_vector(4 downto 0);
		A3 : in std_logic_vector(4 downto 0);
		WD3 : in std_logic_vector(W-1 downto 0);
		clk : in std_logic;
		We3 : in std_logic;
		RD1 : out std_logic_vector(W-1 downto 0);
		RD2 : out std_logic_vector(W-1 downto 0));
End RF;
```

Que realize leitura assincrona e escrita sincrona.

Implementação

Implementei no arquivo rf.vgh, o banco de registrador que realiza leitura assincrona e escrita sincrona delimitado pela borda de subida do clock.

A leitura sincrona funciona da seguinte maneira: coloca-se o endereço que se queira ler em A1, e o valor do registrador endereçado por A1 é colocado no registrador de saída RD1. Da mesma forma, colocando o endereço do registrador que se queira ler em A2 e seu conteúdo é colocado em RD2. Desta forma, como o tamanho de A1 e A2 é 5 bits então o banco de registradores suporta 64 registradores(2^5), cada registrador com tamanho de 32 bits, limitados pelo tamanho dos registradores de saída RD1 e RD2.
A escrita ocorre colocando se o endereço do registrador que se queira armazenar a informação em A3(4bits), e o conteúdo em WD3(32bits), para habilitar a escrita deve setar o flag de escrita We3(1 bit). Como a escrita ocorre sincronamente, ou seja, o conteúdo de WD3 só é armazenado no registrador endereçado por A3, na borda de subida clk = 1.
O registrador 0 sempre deve conter o valor 00000000000000000000000000000000.

Para representar os registradores (64 no total) não realizei a implementação de flip-flops, oque poderia ter sido feito, utilizei um array que o vhdl prove:

```
type reg_type is array (0 to 31) of std_logic_vector(31 downto 0);
```

Test Bench

Os teste implementados para validar a implementação do banco de registradores esta disponivel em tb_rf.vhd, e pode ser executado executando script ***executa_tb.sh***. Foram

implementados 10 teste:

1. **Teste 1**: Escreve o valor 00000000000000000000000000001010 no registrador indexado por 00100, e verifica se de fato foi escrito, colocando se o endereço do registrador em A1 e verificando se tal valor ocorre em RD1.
2. **Teste 2**: Assim como o teste anterior seta um valor no registrador indexado por 00101, e verifica se foi escrito de fato.
3. **Teste 3**: Apos a exeução do teste 2 verifica se o conteudo do registrador 00100 setado no teste1, ainda esta consistente.
4. **Teste 4**: Teste de carga e consulta de valor no registador 00111.
5. **Teste 5**: Assim como o teste 3, verifica novamente a consistencia do registrador 00100.
6. **Teste 6**: Limpa o conteudo do registrador 00100, setando 00000000000000000000000000000000, e valida se esta consistente.
7. **Teste 7**: Verifica se o valor do registrador 0 A1 = 00000, contem o valor 00000000000000000000000000000000.
8. **Teste 8**: Realiza carga no registrar 00001 e verifica consistencia.
9. **Teste 9**: Tenta setar um valor no registrador 0 e verifica se ele ainda contem o valor 00000000000000000000000000000000.
10. **Teste 10**: Teste de carga no registrador 11111.