Projeto e Simulação do Banco de Registradores RISC-V em VHDL

Dérick Daniel Silva de Andrade - 23/1003522

CIC0099 - Organização e Arquitetura de Computadores

Constante zero

A simulação da constante zero no XREGS[0] foi feita a partir da atribuição inicial do valor zero neste registrador e, ao tentar atribuir um valor, só é possível se satisfizer as três condições de ser a subida de clock, o sinal de escrita estar ativado e o registrador de destino for diferente de zero, senão, não é escrito, assim mantendo o valor original zero.

Código do XREG

```
1   LIBRARY ieee;
2   USE ieee.std_logic_1164.all;
3   use ieee.std_logic_unsigned.all;
4   use ieee.numeric_std.all;
5
6   entity XREGS is
7   generic (WSIZE : natural := 32);
8   port (
9   clk, wren    : in std_logic;
10  rs1, rs2, rd : in std_logic_vector(4 downto 0);
11  data         : in std_logic_vector(WSIZE-1 downto 0);
12  ro1, ro2     : out std_logic_vector(WSIZE-1 downto 0));
13  end XREGS;
14
15  ARCHITECTURE arch OF XREGS IS
16  TYPE reg is array (31 downto 0) of std_logic_vector(31 downto 0);
17
18  signal registrador : reg := (others => (others => '0'));
19  signal address1    : integer range 0 to 31;
20  signal address2    : integer range 0 to 31;
21  signal addressd    : integer range 0 to 31;
22
23  begin
24    process(clk, wren, rs1, rs2, rd, data)
25
26    begin
27      addressd <= to_integer(unsigned(rd));
28      if (rising_edge(clk) and addressd /= 0 and wren = '1') then
29          registrador(addressd) <= data;
30      end if;
31
32    end process;
33          address1 <= to_integer(unsigned(rs1));
34          address2 <= to_integer(unsigned(rs2));
35          ro1      <= registrador(address1);
36          ro2      <= registrador(address2);
37  end arch;
```

Código do Testbench

```vhdl
library IEEE;
use IEEE.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.numeric_std.all;

entity testbench is
end testbench;

architecture tb of testbench is

component XREGS is
        generic (WSIZE : natural := 32);
        port(clk, wren: in std_logic;
        rs1, rs2, rd: in std_logic_vector(4 downto 0);
      data: in std_logic_vector(WSIZE-1 downto 0);
      ro1, ro2: out std_logic_vector(WSIZE-1 downto 0));
end component;

signal clk_tb : std_logic                    := '0';
signal wren_tb : std_logic                   := '0';
signal rs1_tb : std_logic_vector(4 downto 0)   := (others  => '0');
signal rs2_tb : std_logic_vector(4 downto 0)   := (others  => '0');
signal rd_tb : std_logic_vector(4 downto 0)    := (others  => '0');
signal ro1_tb : std_logic_vector(31 downto 0)  := (others  => '0');
signal ro2_tb : std_logic_vector(31 downto 0)  := (others  => '0');
signal data_tb : std_logic_vector(31 downto 0) := (others  => '0');

begin

    uut: XREGS port map(clk => clk_tb,
    wren => wren_tb,
    rs1  => rs1_tb,
    rs2  => rs2_tb,
    rd   => rd_tb,
    data => data_tb,
    ro1  => ro1_tb,
    ro2  => ro2_tb );

    process
      begin
      wren_tb <= '1';
      for I in 0 to 31 loop
        clk_tb <= '0';
        data_tb <= std_logic_vector(to_unsigned((I + 1), 32)); -- I + 1 para colocar algo diferente de 0 no reg zero
        rd_tb <= std_logic_vector(to_unsigned(I, 5));
        wait for 1 ns;
        clk_tb <= '1';
        wait for 1 ns;
      end loop;

      wren_tb <= '0';
      for I in 0 to 31 loop
        clk_tb <= '0';
        rs1_tb <= std_logic_vector(to_unsigned(I, 5));
        rs2_tb <= std_logic_vector(to_unsigned(I, 5));
        wait for 1 ns;
        clk_tb <= '1';
        if I /= 0 then
                assert ro1_tb = std_logic_vector(to_unsigned((I + 1), 32)) report "falhou" severity error; -- se o I for diferente de 0, testa se o valor no reg é I + 1
                assert ro2_tb = std_logic_vector(to_unsigned((I + 1), 32)) report "falhou" severity error;
        else
                assert ro1_tb = X"00000000" report "falhou" severity error; -- senão, testa se o valor do reg zero é 0
                assert ro2_tb = X"00000000" report "falhou" severity error;
        end if;
        wait for 1 ns;
      end loop;

      wait;
    end process;
end tb;
```