<u>Relatório para o trabalho VI</u>

Projeto do Banco de Registradores do RISC-V

Aluno: Cristiano Tolentino Santos

Matrícula: 21102850

Professor: Ricardo Pezzuol Jacobi

Turma: T02 (2024.2 - 35T23)

Objetivo:

Projeto e a implementação de um banco de registradores conforme a arquitetura RISC-V, com foco na leitura e escrita de registradores,

Simulando a constante zero em XREGS[0]

A simulação da constante zero em XREGS[0] foi realizada impedindo a modificação de registradores para rd igual a zero.

Código

testbench.vhd:

```
Unset
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity testbench is
end testbench;

architecture test of testbench is
    constant WSIZE : natural := 32;
    signal clk, wren : std_logic := '0';
    signal rs1, rs2, rd : std_logic_vector(4 downto 0) := (others => '0');
    signal data : std_logic_vector(WSIZE-1 downto 0) := (others => '0');
```

```vhdl
    signal ro1, ro2 : std_logic_vector(WSIZE-1 downto 0);
    constant clock_duration : time := 2 ns;

    component XREGS
        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;

begin

    base: XREGS
        generic map (WSIZE => 32)
        port map (
            clk => clk,
            wren => wren,
            rs1 => rs1,
            rs2 => rs2,
            rd => rd,
            data => data,
            ro1 => ro1,
            ro2 => ro2
        );

    process
    begin
        for i in 0 to 130 loop
            clk <= '0';
            wait for clock_duration/2;
            clk <= '1';
            wait for clock_duration/2;
        end loop;
        wait;
    end process;

    process
    begin
        wait for (clock_duration*2);

        -- T0
        wren <= '1';
        rd <= ("00000");
        data <= std_logic_vector(x"FFFFFFFF");
        wait for clock_duration;
        wren <= '0';
        rs1 <= "00000";
        wait for clock_duration;
        assert ro1 = x"00000000"
                                                   report     "T0     FAILURE:
"&"ro1("&integer'image(to_integer(unsigned(rd)))&"):   "&to_string(ro1)&"   /
expected: "&to_string(std_logic_vector(x"00000000")) severity Note;
        if ro1 = x"00000000" then
             report                       "T0                       SUCESS:
"&"ro1("&integer'image(to_integer(unsigned(rd)))&"):   "&to_string(ro1)&"   /
expected: "&to_string(std_logic_vector(x"00000000")) severity Note;
        end if;

        --T1&2
        for i in 1 to 31 loop

        --T1
        wren <= '1';
        rd <= std_logic_vector(to_unsigned(i, 5));
        data <= std_logic_vector(to_unsigned(i, 32));
        wait for clock_duration;
        wren <= '0';
        rs1 <= std_logic_vector(to_unsigned(i, 5));
        wait for clock_duration;
        assert ro1 = std_logic_vector(to_unsigned(i, 32))
             report                       "T1                      FAILURE:
"&"ro1("&integer'image(to_integer(unsigned(rd)))&"):   "&to_string(ro1)&"   /
expected: "&to_string(data) severity Note;
        if ro1 = std_logic_vector(to_unsigned(i, 32)) then
             report                       "T1                       SUCESS:
"&"ro1("&integer'image(to_integer(unsigned(rd)))&"):   "&to_string(ro1)&"   /
expected: "&to_string(data) severity Note;
        end if;

        --T2
        wren <= '1';
        rd <= std_logic_vector(to_unsigned(i, 5));
        data <= std_logic_vector(to_unsigned(i, 32));
        wait for clock_duration;
        wren <= '0';
        rs2 <= std_logic_vector(to_unsigned(i, 5));
        wait for clock_duration;
        assert ro2 = std_logic_vector(to_unsigned(i, 32))
             report                       "T2                      FAILURE:
"&"ro2("&integer'image(to_integer(unsigned(rd)))&"):   "&to_string(ro2)&"   /
expected: "&to_string(data) severity Note;
        if ro2 = std_logic_vector(to_unsigned(i, 32)) then
```

```
             report                        "T2                        SUCESS:
"&"ro2("&integer'image(to_integer(unsigned(rd)))&"):   "&to_string(ro2)&"   /
expected: "&to_string(data) severity Note;
        end if;

        end loop;
        report "End tests" severity Note;
        wait;
    end process;
end test;
```

design.vhd:

```
Unset
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

entity 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 XREGS;


architecture behavior of XREGS is
    type registradores is array (0 to 31) of std_logic_vector(WSIZE-1 downto
0);
    signal regs : registradores := (others => (others => '0'));
begin
    process (clk, wren, rs1, rs2, rd, data)
    begin
    if rising_edge(clk) then
        if wren = '1' and unsigned(rd) /= 0 then
          regs(to_integer(unsigned(rd))) <= data;
        end if;
        ro1 <= regs(to_integer(unsigned(rs1)));
        ro2 <= regs(to_integer(unsigned(rs2)));
    end if;
    end process;
end behavior;
```