Curso Superior de Bacharelado em Engenharia Elétrica

Disciplina: Sistemas Digitais
Professor: Lincoln Machado de Araújo
Aluno1: Sheyla Cantalupo. Matrícula: 20192610029.
Aluno2: Yasmin Alves. Matrícula: 20192610004.

Mini projeto para oitava semana de atividades remotas

Desenvolva uma descrição em VHDL para um CRC Generator/Checker baseado no CRC-16-USB seguindo as seguintes especificações:

```
entity CRC_16_USB is
port(
CLK: in std_logic;-- Sinal de Clock. Para esse projeto, trabalhando em borda de subida.
DATA_IN: in std_logic;-- Serial Input iniciando a transmissão com o MSB.
CLR: in std_logic;-- Sinal assíncrono para limpar todos os FFs em Active High.
CRC_OUT: out std_logic_vector(15 downto 0);-- Saída paralela dos FFs.
CRC_STATUS: out std_logic -- ‘1’ quando CRC_OUT for igual à 0, ‘0’ caso contrário.
);
end CRC_16_USB;
```

Requisitos:

- Feito em dupla ou individual.
- Todos os membros do mesmo grupo devem enviar o mesmo arquivo, finalizando a atividade no Google Classroom.
- O arquivo de Testbench deve mostrar o bom funcionamento do dispositivo gerando 16 bits de CRC para os seguintes pacotes:
    a) 0x632B
    b) 0x56AF
    c) 0x4332
    d) 0x356C
    e) 0x865A
- Em seguida teste os dados+CRC com o uso de Asserts no testbench para mostrar o status (íntegro ou corrompido) no console.
- Cole em local e tamanho visível ao fim deste arquivo o código e o link para o código no EDA Playground.

```
-- testbench
library IEEE;
use IEEE.std_logic_1164.all;

entity testbench is
end testbench;

architecture test of testbench is

component CRC_16_USB is
port(
  CLK: in std_logic;
  DATA_IN: in std_logic;
  CLR: in std_logic;
  CRC_OUT: out std_logic_vector(15 downto 0);
  CRC_STATUS: out std_logic
  );
end CRC_16_USB;


signal clk: std_logic;
signal data_in: std_logic;
signal clr: std_logic;
signal crc_out: std_logic_vector(15 downto 0);
signal crc_status: std_logic;

signal data : std_logic_vector(31 downto 0):=    "00000000000000000110001100101011";

signal data1 : std_logic_vector(31 downto 0):=    "00000000000000000101011010101111";

signal data2 : std_logic_vector(31 downto 0):=    "00000000000000000100001100110010";

signal data3 : std_logic_vector(31 downto 0):=    "00000000000000000011010101101100";

signal data4 : std_logic_vector(31 downto 0):=    "00000000000000001000011001011010";

begin

dut: CRC_16_USB port map(clk, data_in, clr, crc_out, crc_status);

  clr <= '1',
        '0' after 20 ns;

process
begin
```

```
--exemplo a
for i in 31 downto 0 loop
    data_in <= data(i);
    clk <='1';
    wait for 20 ns;
    clk <= '0';
    wait for 20 ns;
end loop;

--exemplo b
for i in 31 downto 0 loop
    data_in <= data1(i);
    clk <= '1';
    wait for 20 ns;
    clk <= '0';
    wait for 20 ns;
end loop;

--exemplo c
for i in 31 downto 0 loop
    data_in <= data2(i);
    clk <='1';
    wait for 20 ns;
    clk <= '0';
    wait for 20 ns;
end loop;

--exemplo d
for i in 31 downto 0 loop
    data_in <= data3(i);
    clk <='1';
    wait for 20 ns;
    clk <= '0';
    wait for 20 ns;
end loop;

--exemplo e
for i in 31 downto 0 loop
    data_in <= data4(i);
    clk <='1';
    wait for 20 ns;
    clk <= '0';
    wait for 20 ns;
end loop;
wait for 50 ns;
wait;
```

```
end process;

end test;
```

```
-- design
library IEEE;
use IEEE.std_logic_1164.all;

entity CRC_16_USB is
port(
CLK: in std_logic;
DATA_IN: in std_logic;
CLR: in std_logic;
CRC_OUT: out std_logic_vector(15 downto 0);
CRC_STATUS: out std_logic
);
end CRC_16_USB;


architecture crc of CRC_16_USB is

signal data: std_logic_vector(15 downto 0) := (others => '0');

begin

        CRC_OUT <= data;
   CRC_STATUS <= '1' when data = "0000000000000000" else '0';

        process(CLK)
        begin

   if(CLR) then
        data <= (others =>'0');

        elsif(CLK'event and CLK = '1') then

        data(0)  <= DATA_IN xor data(15);
      data(1)  <= data(0);
      data(2)  <= data(1) xor data(15);
      data(3)  <= data(2);
      data(4)  <= data(3);
      data(5)  <= data(4);
```

```
        data(6)  <= data(5);
        data(7)  <= data(6);
        data(8)  <= data(7);
        data(9)  <= data(8);
        data(10) <= data(9);
        data(11) <= data(10);
        data(12) <= data(11);
        data(13) <= data(12);
        data(14) <= data(13);
        data(15) <= data(14) xor data(15);

           end if;
           end process;


end crc;
```

https://www.edaplayground.com/x/GNwd