Digital System Design Lab Report, Lab4

Author: Ying Yiwen
Number: 12210159

Abstract

Contents

1 Introduction

2 Sequential Signal Generator

2.1 VHDL Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity sequence_generator is
    Port (
        clk : in STD_LOGIC;
        reset : in STD_LOGIC;
        dataout : out STD_LOGIC
    );
end sequence_generator;

architecture Behavioral of sequence_generator is
    signal current_state : std_logic_vector  (2 downto 0) := (others => '0');
    signal next_state :  std_logic_vector  (2 downto 0) := (others => '0');
begin
    -- accident
process (clk, reset) is
begin
    if reset = '1' then
        current_state <= "000";
    elsif clk'event and clk = '1' then
        current_state <= next_state;
    end if;
end process;

-- state transfer
process (current_state) is
begin
    case current_state is
        when "000" =>
            dataout <= '1';
            next_state <= "001";
        when "001" =>
            dataout <= '1';
            next_state <= "010";
        when "010" =>
            dataout <= '0';
            next_state <= "011";
        when "011" =>
            dataout <= '1';
            next_state <= "100";
        when "100" =>
            dataout <= '1';
            next_state <= "101";
        when "101" =>
            dataout <= '1';
            next_state <= "110";
        when "110" =>
```

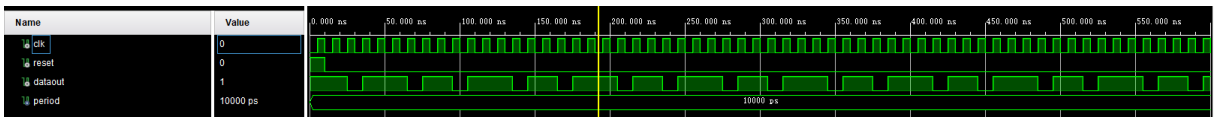
```
            dataout <= '0';
            next_state <= "000";
        when others =>
            dataout <= '0';
            next_state <= "000";
    end case;
end process;
end Behavioral;
```

2.2 Testbench Code

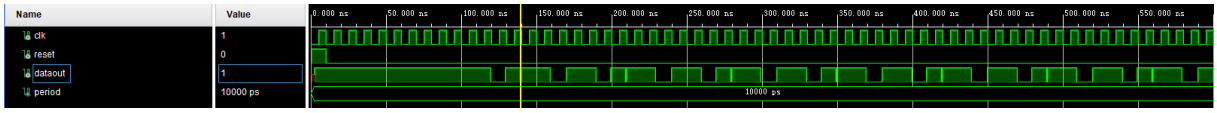
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity testbench is
end testbench;

architecture tb of testbench is
component sequence_generator is
  Port (
    clk : in STD_LOGIC;
    reset : in STD_LOGIC;
    dataout : out STD_LOGIC
  );
end component;

signal clk : STD_LOGIC := '0';
signal reset : STD_LOGIC := '0';
signal dataout : STD_LOGIC;
constant period : time := 10 ns;

begin
  uut: sequence_generator port map (
    clk => clk,
    reset => reset,
    dataout => dataout
  );

  clk <= not clk after period / 2;

  tb: process is
  begin
    reset <= '1';
    wait for period;
    reset <= '0';
    wait for period;
    wait;
  end process;

end tb;
```

2.3 Therotical Analysis

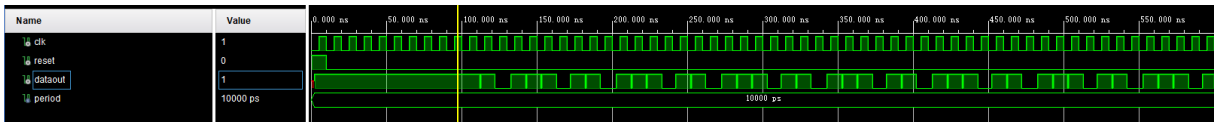
2.4 Simulation Result



(a) Behavioral Simulation Result

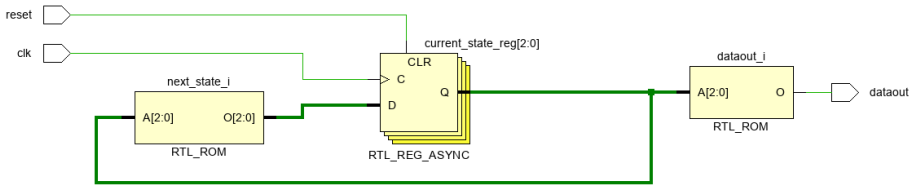


(b) Post-Synthesis Simulation Result

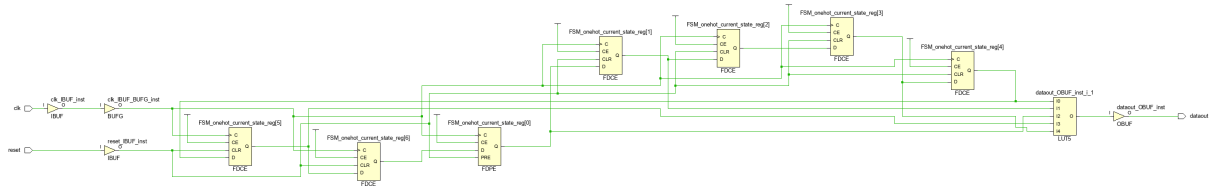


(c) Post-Implementation Simulation Result

Figure 1: Simulation Result



(a) Behavior Schemetic



(b) Behavior Schemetic

Figure 2: Schemetic Result

3 Sequential Signal Detector

3.1 VHDL Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity sequence_generator is
    Port (
        clk : in STD_LOGIC;
```

```
        reset : in STD_LOGIC;
        input : in STD_LOGIC;
        dataout : out STD_LOGIC
    );
end sequence_generator;

architecture Behavioral of sequence_generator is
    signal current_state : std_logic_vector(1 downto 0) := "00";
    signal next_state : std_logic_vector(1 downto 0);
begin
    process(clk, reset)
    begin
        if reset = '1' then
            current_state <= "00";
        elsif CLK'event and CLK = '1' then
            current_state <= next_state;
        end if;
    end process;

    -- next state logic
    with current_state & input select
        next_state <=
            "00" when "000",
            "01" when "001",
            "10" when "010",
            "01" when "011",
            "00" when "100",
            "11" when "101",
            "11" when "110",
            "11" when "111",
            "00" when others;

    with current_state & input select
        dataout <=
            '0' when "000" | "001" | "010" | "011" | "100",
            '1' when "101" | "110" | "111",
            '0' when others;

end Behavioral;
```

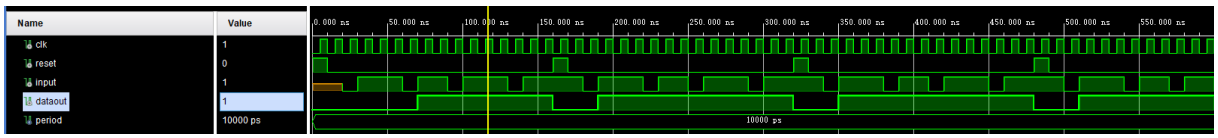
3.2 Testbench Code

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity testbench is
end testbench;

architecture tb of testbench is
component sequence_generator is
  Port (
    clk : in STD_LOGIC;
    reset : in STD_LOGIC;
```

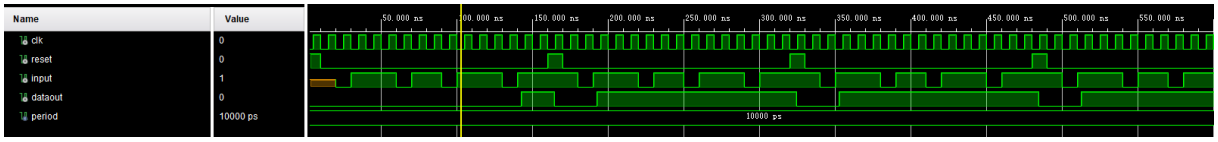
```
    input : in STD_LOGIC;
    dataout : out STD_LOGIC
  );
end component;

signal clk : STD_LOGIC := '0';
signal reset : STD_LOGIC := '0';
signal input : STD_LOGIC;
signal dataout : STD_LOGIC;
constant period : time := 10 ns;

begin
  uut: sequence_generator port map (
    clk => clk,
    reset => reset,
    input => input,
    dataout => dataout
  );

  clk <= not clk after period / 2;

  tb: process is
  begin
    for i in 1 to 10 loop
      reset <= '1';
      wait for period;
      reset <= '0';
      wait for period;

      for j in 1 to 2 loop
        input <= '0';
        wait for period;
        input <= '1';
        wait for period;
        input <= '1';
        wait for period;
        input <= '1';
        wait for period;
        input <= '0';
        wait for period;
        input <= '1';
        wait for period;
        input <= '1';
        wait for period;
      end loop;
    end loop;
    wait;
  end process;

end tb;
```

3.3 Therotical Analysis

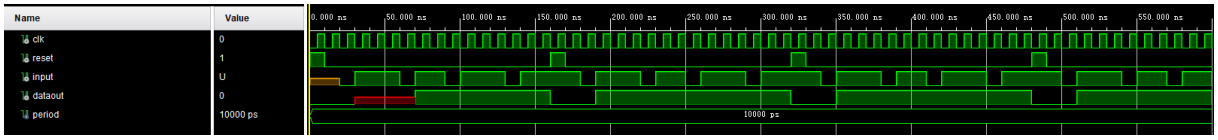
3.4 Simulation Result



(a) Behavioral Simulation Result

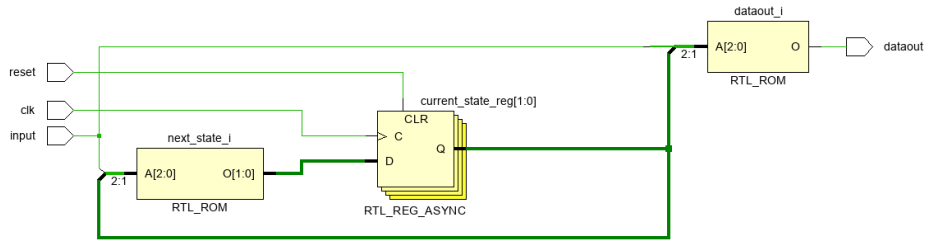


(b) Post-Synthesis Simulation Result

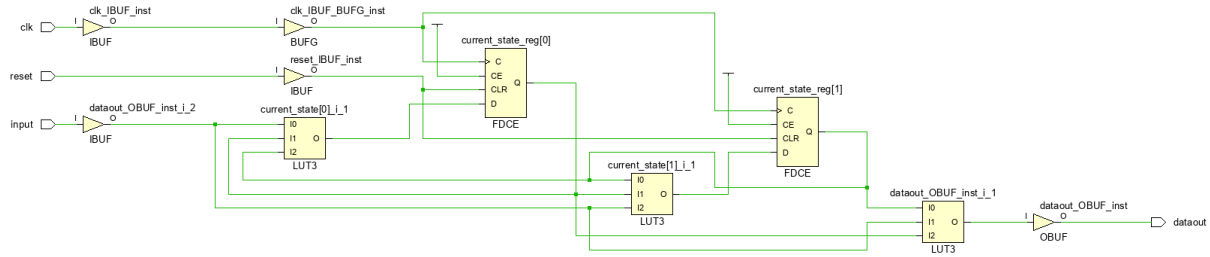


(c) Post-Implementation Simulation Result

Figure 3: Simulation Result



(a) Behavior Schemetic



(b) Behavior Schemetic

Figure 4: Schemetic Result

4 Conclusion