به نام خدا

عنوان:

تکلیف شماره دوم درس طراحی سیستم‌های دیجیتالی

نام درس:

طراحی سیستم‌های دیجیتالی

استاد درس:

دکتر مروستی

اعضا:

محمدعلی مجتهدسلیمانی-۹۹۲۰۲۳۰۳۹

تاریخ:

۱۴۰۳/۰۲/۲۰

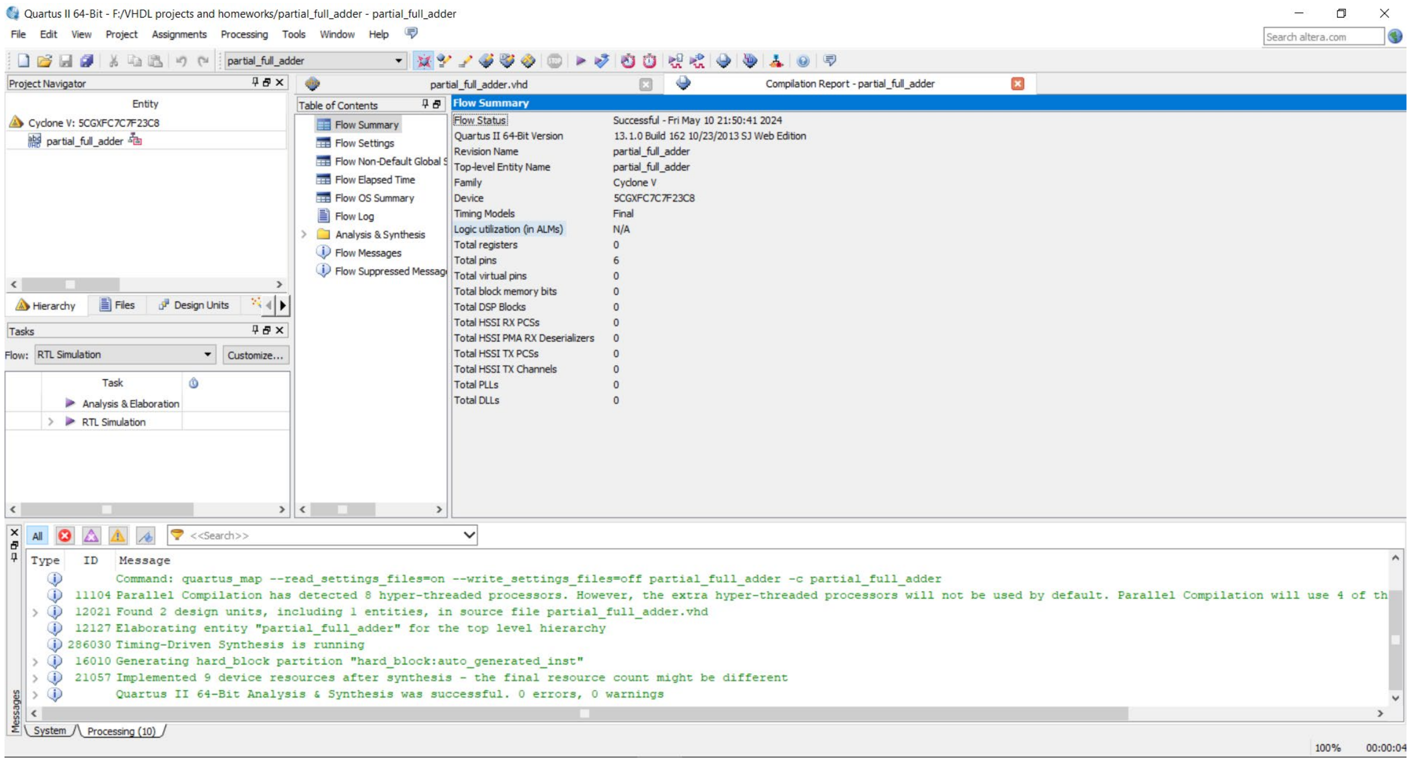
Question 1

Q1. For verify a 4-bit carry look-ahead adder we should first make a test bench to simulate all possible inputs and check the outputs against expected values which test bench going to do that. For the following code I assumed that you already have a Carry-Look-Ahead Adder module, for my code I already done it in first homework question 4.

❖Main code for implement partial full adder:

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

-- Define the entity for the Partial Full Adder
entity partial_full_adder is
    Port (
        A    : in  STD_LOGIC;  -- Input bit A
        B    : in  STD_LOGIC;  -- Input bit B
        Cin  : in  STD_LOGIC;  -- Carry input
        S    : out STD_LOGIC;  -- Sum output
        P    : out STD_LOGIC;  -- Propagate output
        G    : out STD_LOGIC   -- Generate output
    );
end partial_full_adder;

-- Architecture declaration using a behavioral model
architecture Behavioral of partial_full_adder is
begin
    -- Generate the Sum (S) output
    S <= A xor B xor Cin;


    -- Generate the Propagate (P) output

    P <= A or B;


    -- Generate the Generate (G) output

    G <= A and B;


end Behavioral;
```

❖Complication Report:



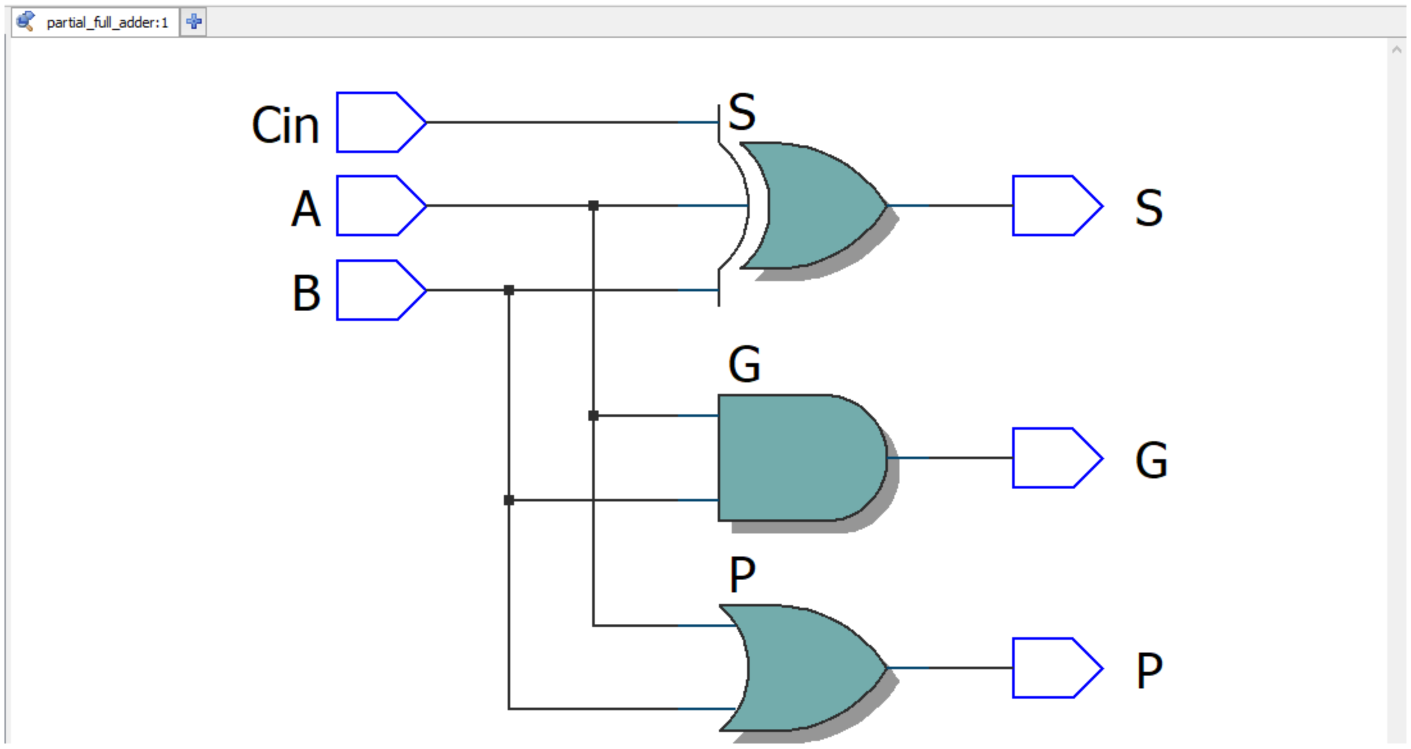
Quartus II 64-Bit - F:/VHDL projects and homeworks/partial_full_adder - partial_full_adder

Flow Summary

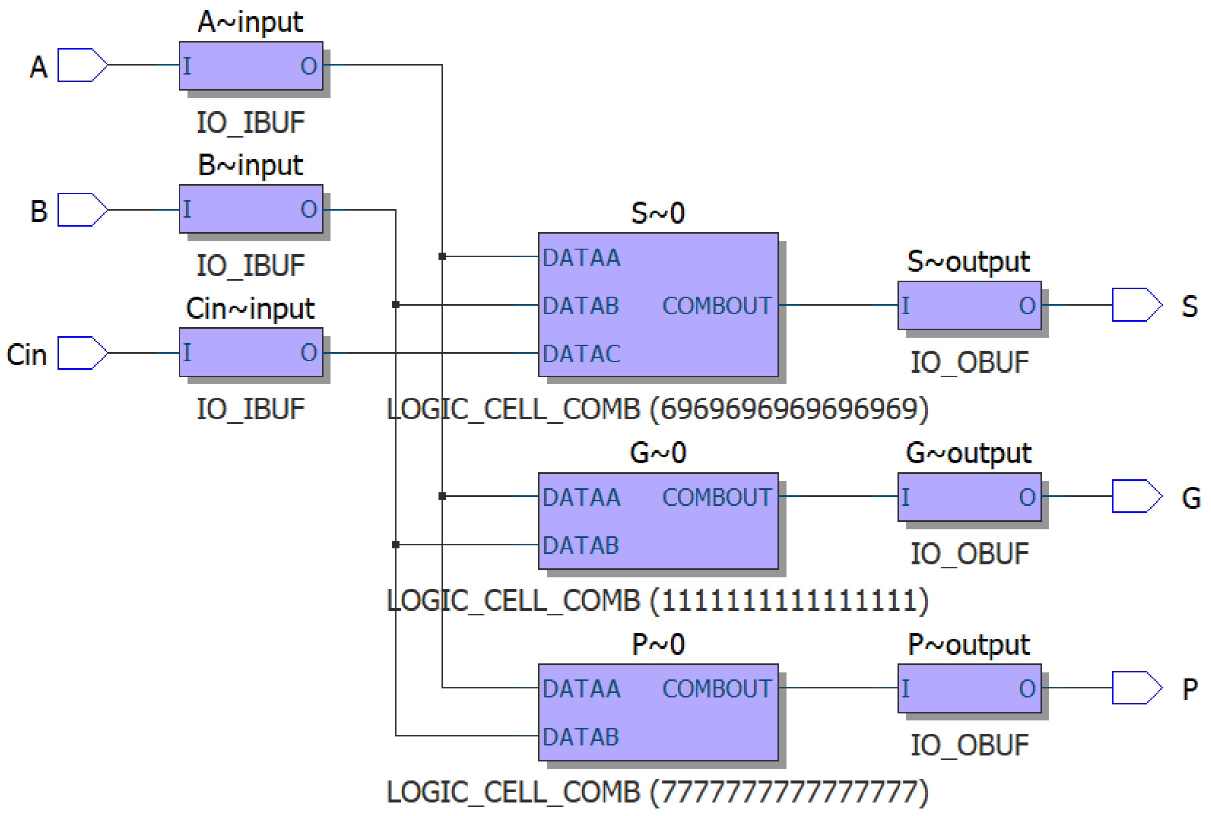
Flow Status	Successful - Fri May 10 21:50:41 2024
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	partial_full_adder
Top-level Entity Name	partial_full_adder
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	6
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI TX Channels	0
Total PLLs	0
Total DLLs	0

```
Type   ID     Message
              Command: quartus_map --read_settings_files=on --write_settings_files=off partial_full_adder -c partial_full_adder
        11104 Parallel Compilation has detected 8 hyper-threaded processors. However, the extra hyper-threaded processors will not be used by default. Parallel Compilation will use 4 of th
        12021 Found 2 design units, including 1 entities, in source file partial_full_adder.vhd
        12127 Elaborating entity "partial_full_adder" for the top level hierarchy
       286030 Timing-Driven Synthesis is running
        16010 Generating hard_block partition "hard_block:auto_generated_inst"
        21057 Implemented 9 device resources after synthesis - the final resource count might be different
              Quartus II 64-Bit Analysis & Synthesis was successful. 0 errors, 0 warnings
```

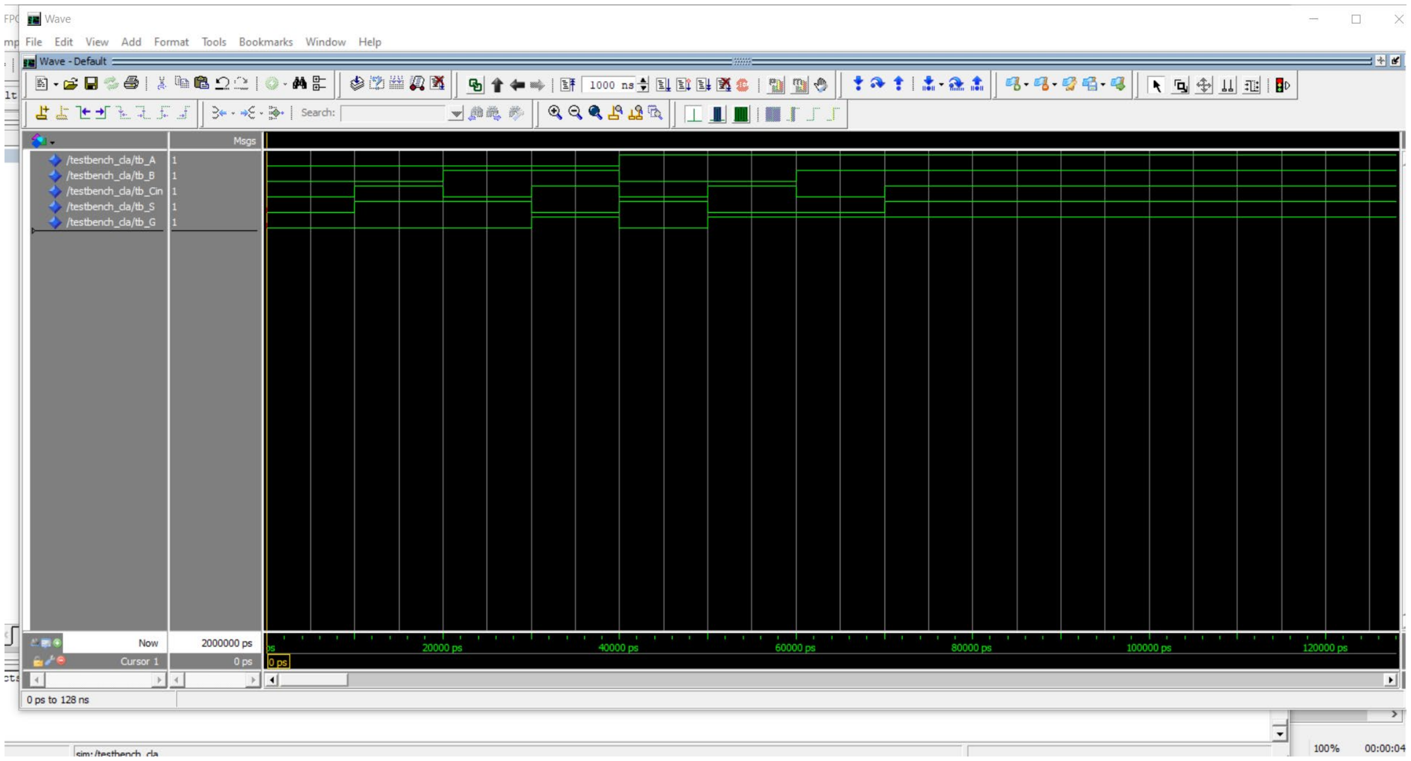
❖RTL view:



❖Post-mapping view:



❖Wave Form:



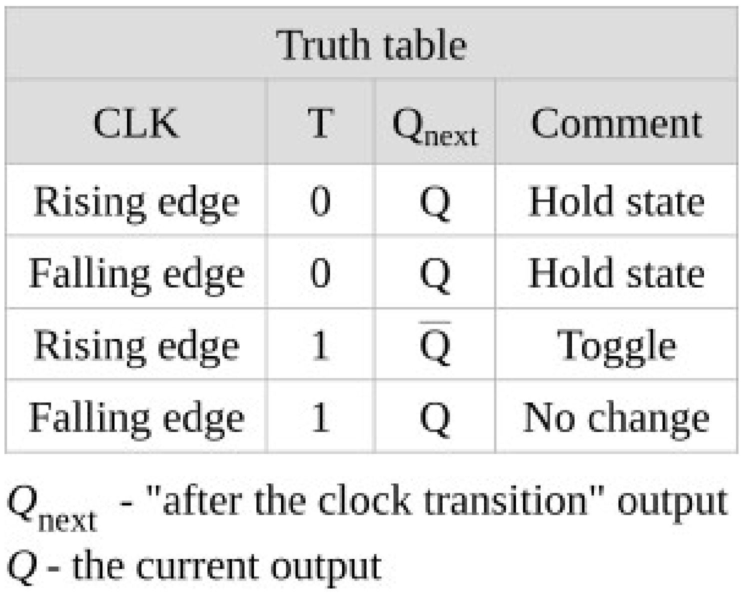
❖Test bench code:

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;


-- Define the test bench entity

entity TestBench_CLA is

-- Testbench does not have any ports

end TestBench_CLA;


architecture behavior of TestBench_CLA is

    -- Component Declaration for the Carry Adder

    component carry_adder

        Port (

            A    : in  STD_LOGIC;

            B    : in  STD_LOGIC;
            Cin   : in  STD_LOGIC;
            S     : out STD_LOGIC;
            G     : out STD_LOGIC
        );
    end component;

    -- Signals for interfacing with the component
    signal tb_A, tb_B, tb_Cin : STD_LOGIC := '0';
    signal tb_S, tb_G : STD_LOGIC;

begin
    -- Instantiate the Unit Under Test (UUT)
    uut: carry_adder port map (
        A => tb_A,
        B => tb_B,
        Cin => tb_Cin,
        S => tb_S,
        G => tb_G
    );

    -- Stimulus process
    stim_proc: process
    begin
        -- Test all input combinations
        tb_A <= '0'; tb_B <= '0'; tb_Cin <= '0';
        wait for 10 ns;
        tb_A <= '0'; tb_B <= '0'; tb_Cin <= '1';
        wait for 10 ns;
        tb_A <= '0'; tb_B <= '1'; tb_Cin <= '0';
        wait for 10 ns;

        tb_A <= '0'; tb_B <= '1'; tb_Cin <= '1';

        wait for 10 ns;

        tb_A <= '1'; tb_B <= '0'; tb_Cin <= '0';

        wait for 10 ns;

        tb_A <= '1'; tb_B <= '0'; tb_Cin <= '1';

        wait for 10 ns;

        tb_A <= '1'; tb_B <= '1'; tb_Cin <= '0';

        wait for 10 ns;

        tb_A <= '1'; tb_B <= '1'; tb_Cin <= '1';

        wait for 10 ns;


        -- Complete the simulation

        wait;

    end process;


end behavior;
```

Question 2

Q2. For a T flip flop, we the next state of flip flop driven by XOR T (input of circuit) and Q which is the current state of our circuit so based on this and based on t ff truth table I wrote a code for it. First here is a truth table of t ff:

❖**T FF truth table:**



Truth table			
CLK	T	Q_{next}	Comment
Rising edge	0	Q	Hold state
Falling edge	0	Q	Hold state
Rising edge	1	$\overline{Q}$	Toggle
Falling edge	1	Q	No change

Q_{next} - "after the clock transition" output

Q - the current output

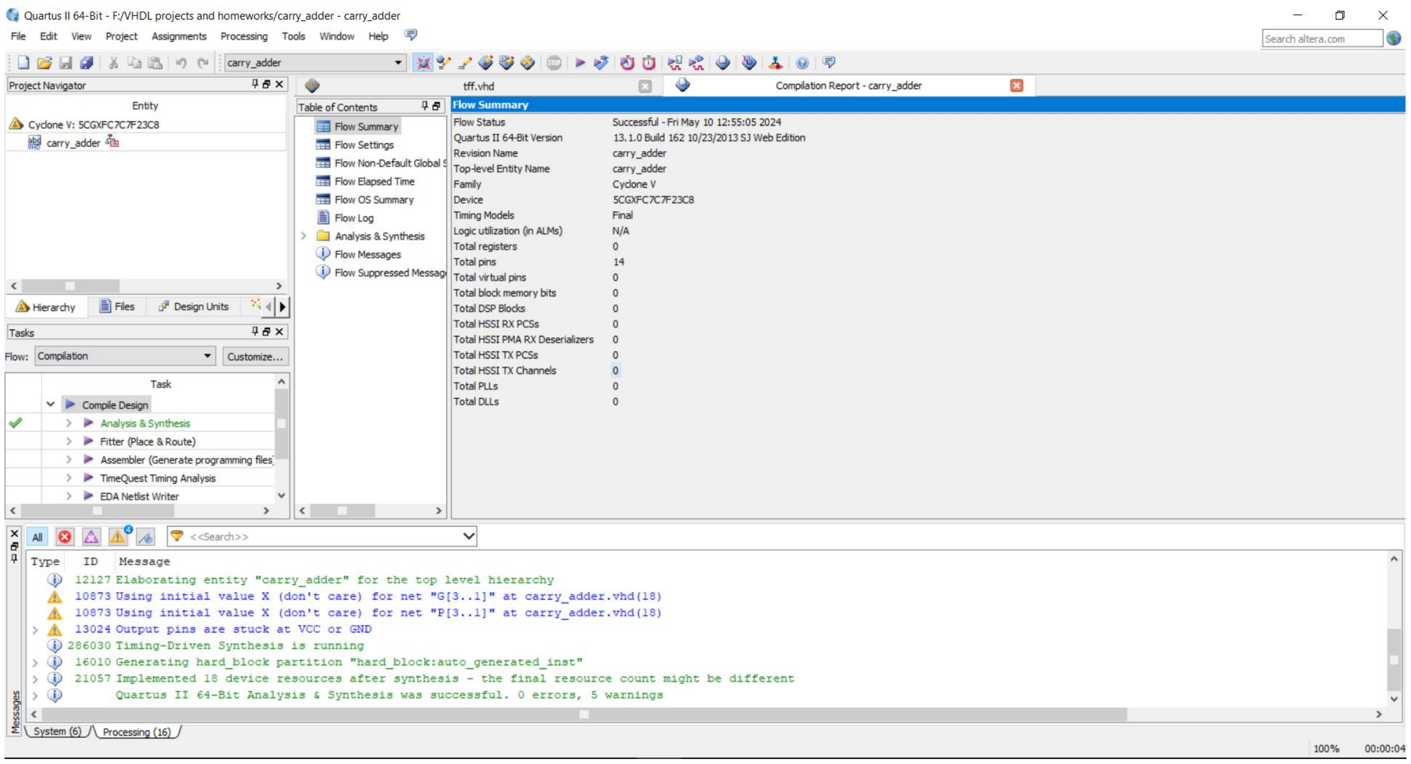
❖**Code:**

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;


entity tff is

   Port ( clk : in STD_LOGIC;

         async_reset : in STD_LOGIC;

         Q : out STD_LOGIC);

end entity tff;
```

```
architecture toggle of tff is
signal S_Q : std_LOGIC := '0';
begin
    process(clk, async_reset)
    begin
        if async_reset = '1' then
            S_Q <= '0';
        elsif rising_edge(clk) then
            S_Q <= not S_Q;
        end if;
    end process;
           Q <= S_Q;
end architecture toggle;
```

*: in this code I declared 2 inputs: clk for clock pin of circuit and the async_reset for reset pin of circuit. The output is Q which is next state of circuit. Reset pin Is asynchronous from clock pin and it is active high so it means when pin becomes 1 the output of circuit be 0. I used signals to work with outputs Q because we can not directly work with Q. signal is a wired in VHDL.

❖Output after compile:

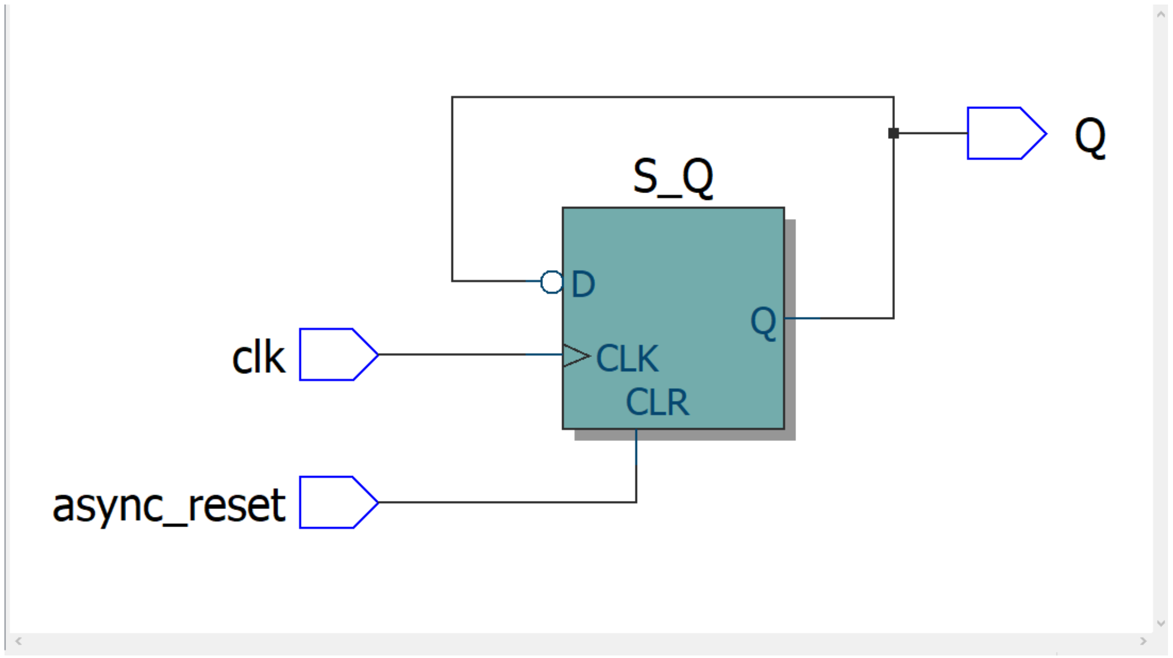


Quartus II 64-Bit - F:/VHDL projects and homeworks/carry_adder - carry_adder

Flow Status	Successful - Fri May 10 12:55:05 2024
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	carry_adder
Top-level Entity Name	carry_adder
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	0
Total pins	14
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI TX Channels	0
Total PLLs	0
Total DLLs	0

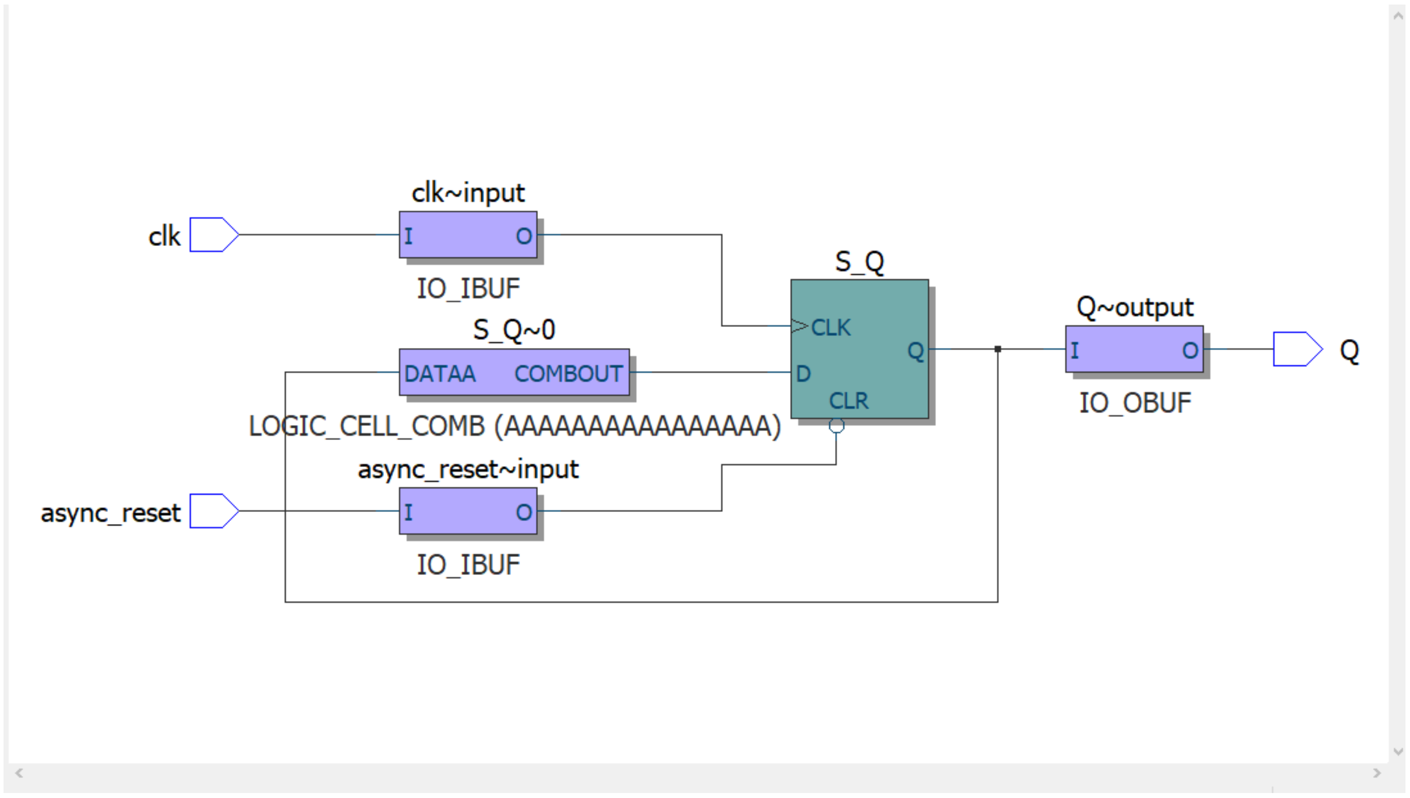
```
Type  ID     Message
      12127  Elaborating entity "carry_adder" for the top level hierarchy
      10873  Using initial value X (don't care) for net "G[3..1]" at carry_adder.vhd(18)
      10873  Using initial value X (don't care) for net "P[3..1]" at carry_adder.vhd(18)
      13024  Output pins are stuck at VCC or GND
     286030  Timing-Driven Synthesis is running
      16010  Generating hard_block partition "hard_block:auto_generated_inst"
      21057  Implemented 18 device resources after synthesis - the final resource count might be different
             Quartus II 64-Bit Analysis & Synthesis was successful. 0 errors, 5 warnings
```

❖RTL view:

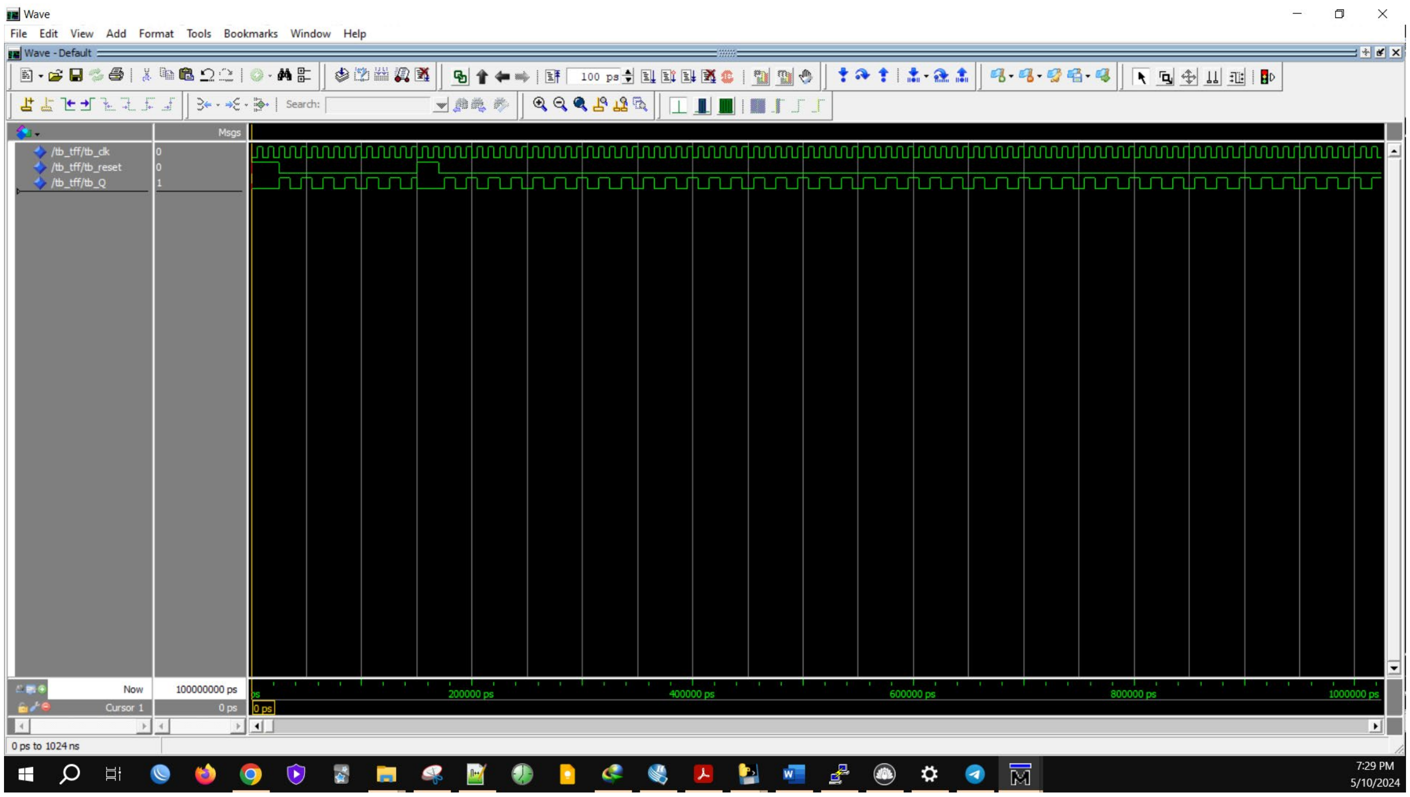


*: notice that in this code we built a T flip flop using D flip flop. This is achieved by feeding the output of the D Flip flop to its input through an XOR gate.

❖Post-mapping view:



❖Wave Form:



❖Test bench code:

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;


entity tb_tff is

-- Testbench does not have any ports

end tb_tff;


architecture behavior of tb_tff is

    -- Component Declaration for the T Flip-Flop

    component tff

        Port ( clk : in STD_LOGIC;

               async_reset : in STD_LOGIC;

               Q : out STD_LOGIC);
    end component;


    -- Signals for interfacing with the component
    signal tb_clk : STD_LOGIC := '0';
    signal tb_reset : STD_LOGIC := '0';
    signal tb_Q : STD_LOGIC;


    -- Clock period definition
    constant clk_period : time := 10 ns;


begin
    -- Instantiate the T Flip-Flop
    uut: tff port map (
        clk => tb_clk,
        async_reset => tb_reset,
        Q => tb_Q
    );


    -- Clock process definitions
    clk_process : process
    begin
        while true loop
            tb_clk <= '0';
            wait for clk_period/2;
            tb_clk <= '1';
            wait for clk_period/2;
        end loop;
    end process;
    -- Stimulus process
    stim_proc: process
    begin
        -- Test 1: Reset the flip-flop
        tb_reset <= '1';
        wait for 25 ns;
        tb_reset <= '0';
        wait for 25 ns;

        -- Test 2: Allow some clock pulses to toggle the flip-flop
        tb_reset <= '0';
        wait for 100 ns;

        -- Test 3: Assert reset while clock is running
        tb_reset <= '1';
        wait for 20 ns;
        tb_reset <= '0';
        wait for 100 ns;

        -- End of testing
        wait;
    end process;
end behavior;
```

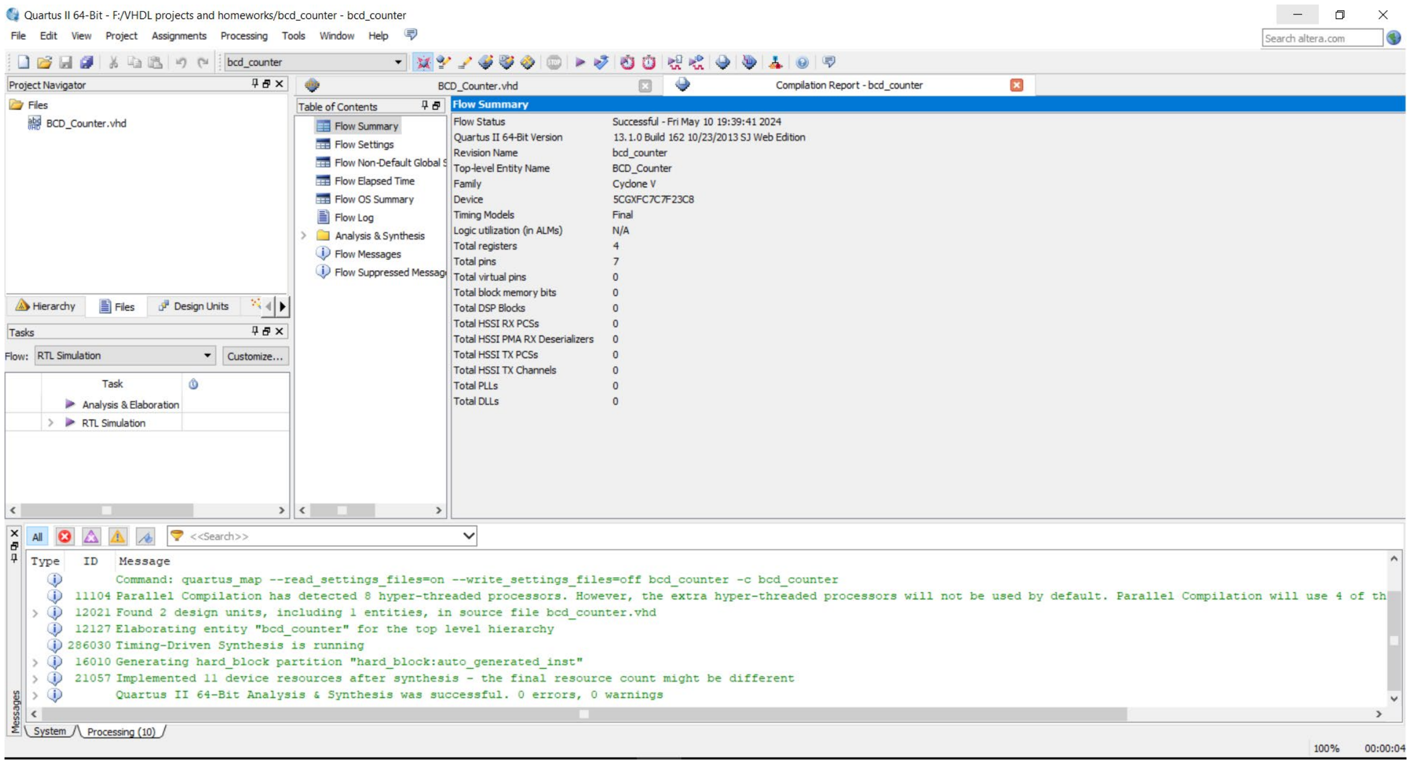
question 3

Q3. Designing a 4-bit BCD (Binary-Coded Decimal) up/down counter in VHDL involves creating a circuit that counts from 0 to 9 and back again, with the count direction controlled by an input signal **Dir**.

❖Code:

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

entity BCD_Counter is
    Port (
        clk : in STD_LOGIC;
        reset : in STD_LOGIC;
        Dir : in STD_LOGIC;  -- '0' for down, '1' for up
        count : out STD_LOGIC_VECTOR(3 downto 0)
    );
end BCD_Counter;
architecture Behavioral of BCD_Counter is
    signal r_count : STD_LOGIC_VECTOR(3 downto 0) := "0000";
begin
    process(clk, reset)
    begin
        if reset = '1' then
            r_count <= "0000";  -- Reset the counter
        elsif rising_edge(clk) then
            if Dir = '1' then  -- Count up
            if r_count = "1001" then  -- Max value for BCD
                r_count <= "0000";  -- Wrap around
            else
                r_count <= r_count + 1;
            end if;
        else  -- Count down
            if r_count = "0000" then
                r_count <= "1001";  -- Wrap around from 0 to 9
            else
                r_count <= r_count - 1;
            end if;
        end if;
    end if;
  end process;

  count <= r_count;  -- Output the current count
end Behavioral;
```

❖compilation report:

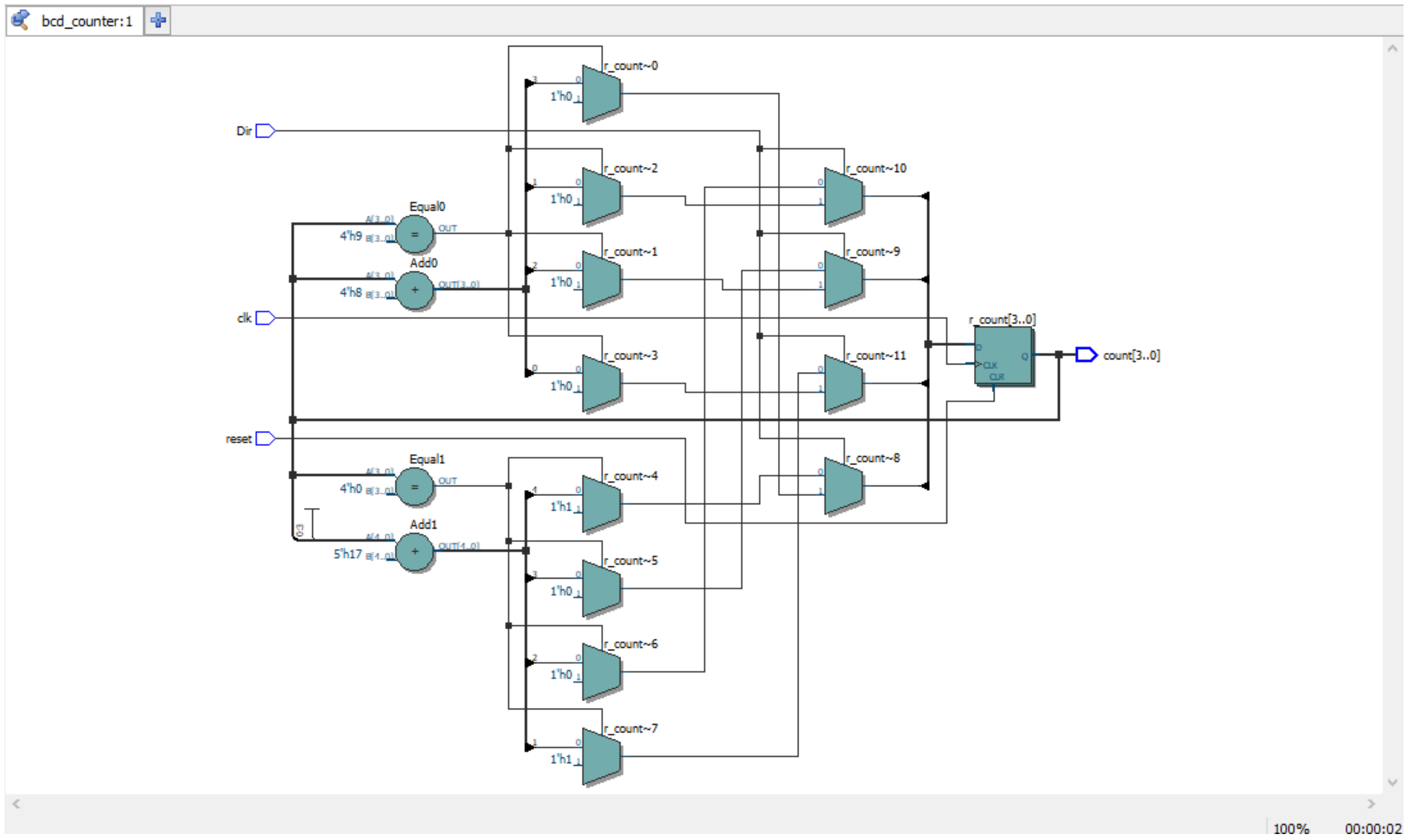


Quartus II 64-Bit - F:/VHDL projects and homeworks/bcd_counter - bcd_counter

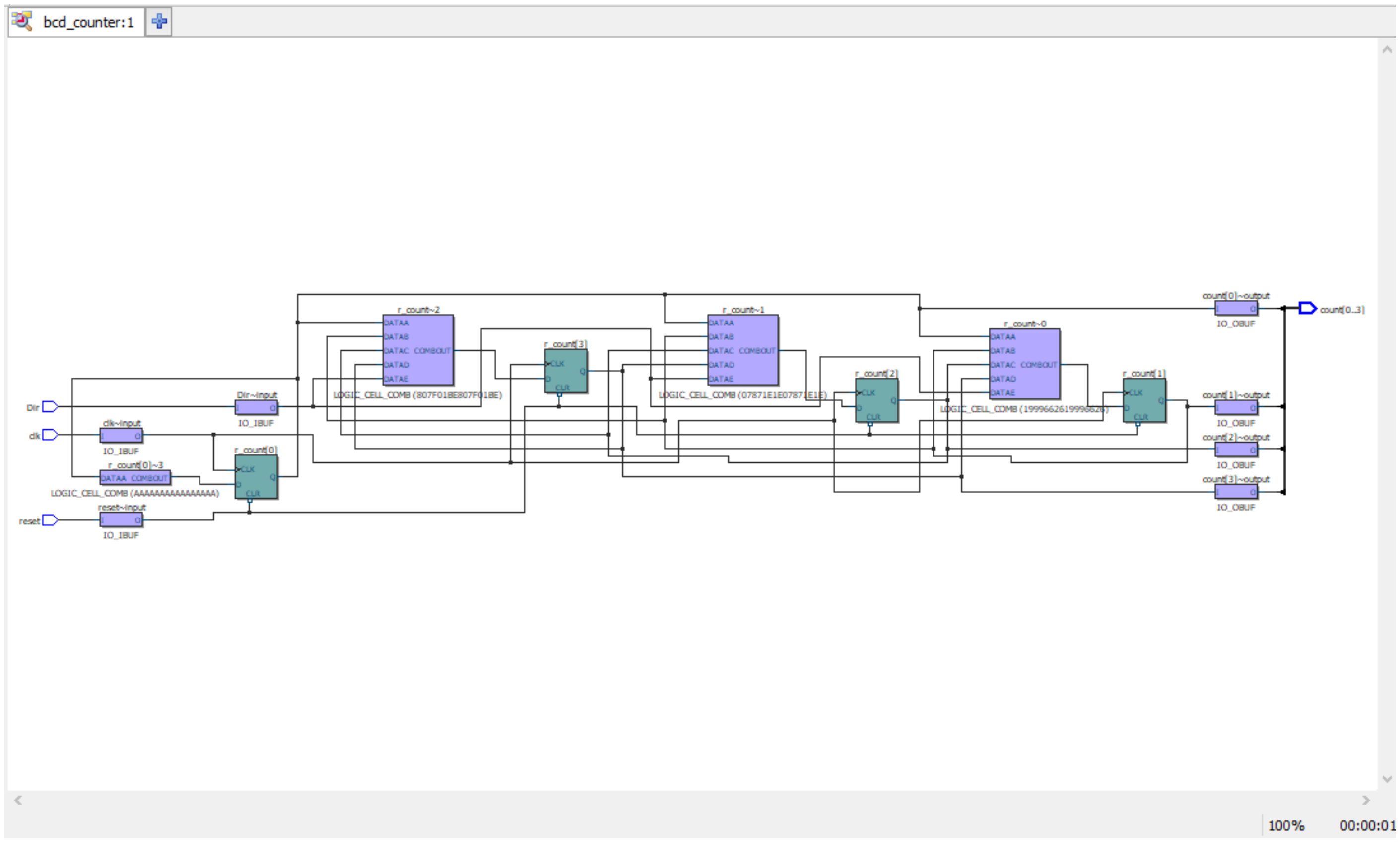
Flow Summary	
Flow Status	Successful - Fri May 10 19:39:41 2024
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	bcd_counter
Top-level Entity Name	BCD_Counter
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	4
Total pins	7
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI TX Channels	0
Total PLLs	0
Total DLLs	0

```
Type  ID     Message
             Command: quartus_map --read_settings_files=on --write_settings_files=off bcd_counter -c bcd_counter
      11104  Parallel Compilation has detected 8 hyper-threaded processors. However, the extra hyper-threaded processors will not be used by default. Parallel Compilation will use 4 of th
      12021  Found 2 design units, including 1 entities, in source file bcd_counter.vhd
      12127  Elaborating entity "bcd_counter" for the top level hierarchy
      286030 Timing-Driven Synthesis is running
      16010  Generating hard_block partition "hard_block:auto_generated_inst"
      21057  Implemented 11 device resources after synthesis - the final resource count might be different
             Quartus II 64-Bit Analysis & Synthesis was successful. 0 errors, 0 warnings
```

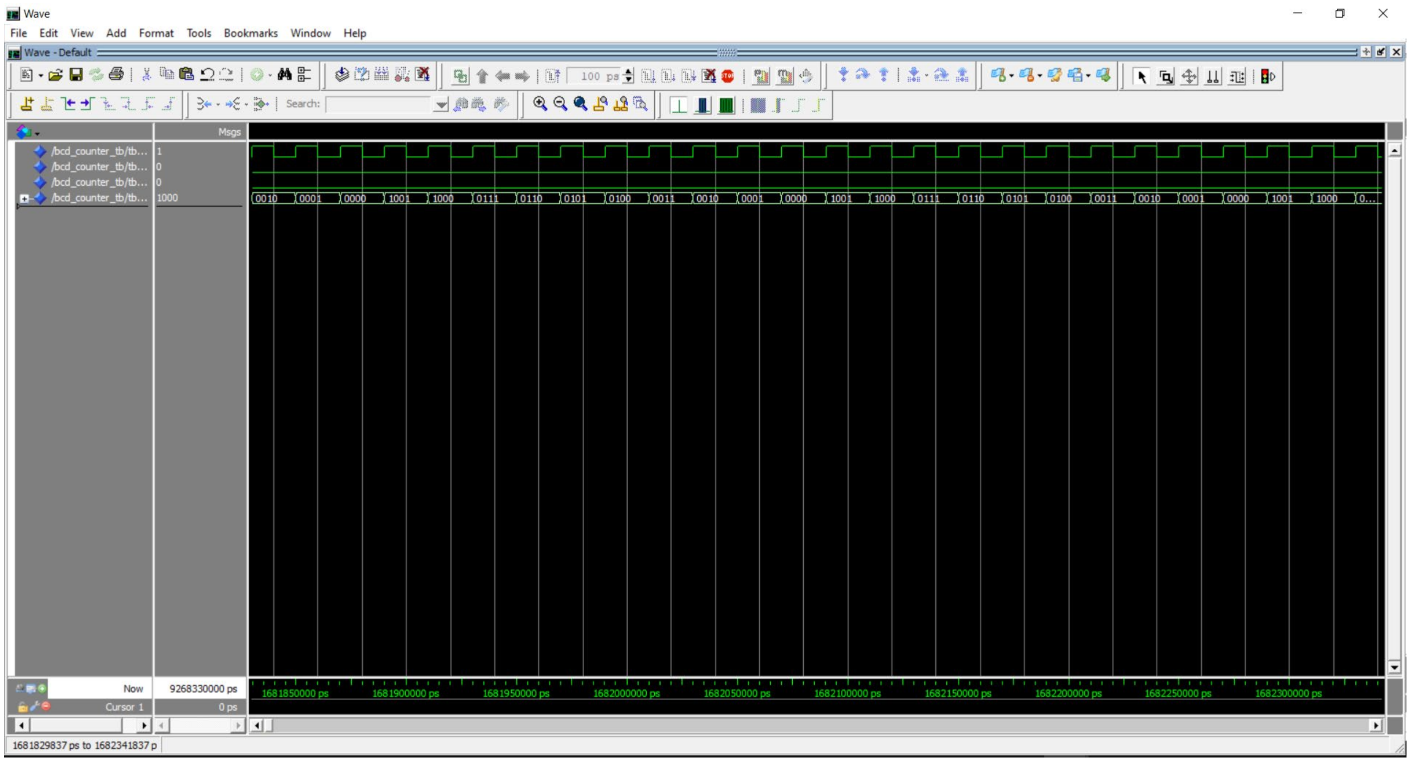
❖RTL view:



❖Post-mapping view:



❖Wave form:



❖Test bench code:

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

entity BCD_Counter_TB is
-- Testbench has no ports
end BCD_Counter_TB;

architecture behavior of BCD_Counter_TB is
    -- Component Declaration
    component BCD_Counter
        Port (
            clk : in STD_LOGIC;
            reset : in STD_LOGIC;
            Dir : in STD_LOGIC;
            count : out STD_LOGIC_VECTOR(3 downto 0));
    end component;

    -- Test Signals
    signal tb_clk : STD_LOGIC := '0';
    signal tb_reset : STD_LOGIC := '0';
    signal tb_Dir : STD_LOGIC := '0';
    signal tb_count : STD_LOGIC_VECTOR(3 downto 0);

begin
    -- Clock process
    clk_process : process
```

```
begin
    tb_clk <= '0';
    wait for 10 ns;  -- Clock period of 20 ns
    tb_clk <= '1';
    wait for 10 ns;
end process;



-- Instantiate the Unit Under Test (UUT)
uut: BCD_Counter port map (
    clk => tb_clk,
    reset => tb_reset,
    Dir => tb_Dir,
    count => tb_count);

-- Test Stimulus Process
stim_proc: process
begin
    -- Initialize Inputs
    tb_reset <= '1';  -- Assert reset initially
    wait for 40 ns;
    tb_reset <= '0';  -- De-assert reset
    wait for 20 ns;

    -- Test Count Up
    tb_Dir <= '1';  -- Set direction to up
    wait for 220 ns;  -- Allow enough time to observe the count

    -- Test Count Down
```

```
        tb_Dir <= '0';  -- Change direction to down
        wait for 220 ns;  -- Allow enough time to observe the count


        -- Finish simulation
        wait;
    end process;


end behavior;
```

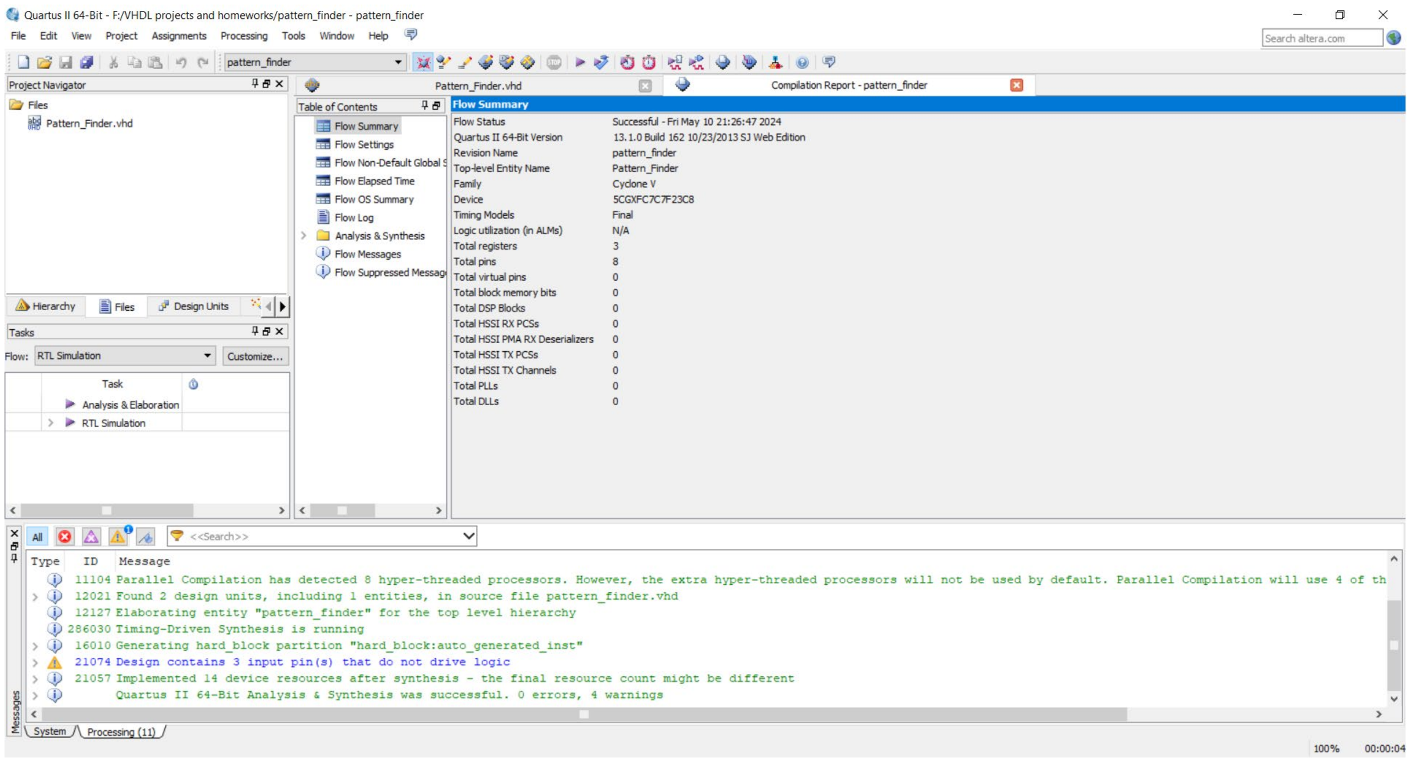
Question 4

Q4. to write a VHDL code that implement pattern transformation for the given table I'll create a state machine where each state corresponds to one of the input patterns and transition to the corresponding next pattern.

❖**Code:**

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


-- Define the entity for the pattern transformation
entity Pattern_Finder is
    Port (
        clk : in STD_LOGIC;
        reset : in STD_LOGIC;
        input_pattern : in STD_LOGIC_VECTOR(2 downto 0);
        next_pattern : out STD_LOGIC_VECTOR(2 downto 0)
    );
end Pattern_Finder;

-- Architecture declaration using a behavioral model
architecture Behavioral of Pattern_Finder is
    signal current_state, next_state : STD_LOGIC_VECTOR(2 downto 0);
begin

    -- State register process
    process(clk, reset)
    begin
        if reset = '1' then
            current_state <= "000"; -- Default state
        elsif rising_edge(clk) then
            current_state <= next_state;
        end if;
    end process;

    -- Next state logic based on current state
    process(current_state)
    begin
        case current_state is
            when "000" => next_state <= "011";
            when "011" => next_state <= "110";
            when "110" => next_state <= "101";
            when "101" => next_state <= "111";
            when "111" => next_state <= "000";
            when others => next_state <= "000"; -- Safety net
        end case;
    end process;


    -- Output logic
    next_pattern <= next_state; -- Directly map next state to output


end Behavioral;
```

❖output after compilation:



Quartus II 64-Bit - F:/VHDL projects and homeworks/pattern_finder - pattern_finder

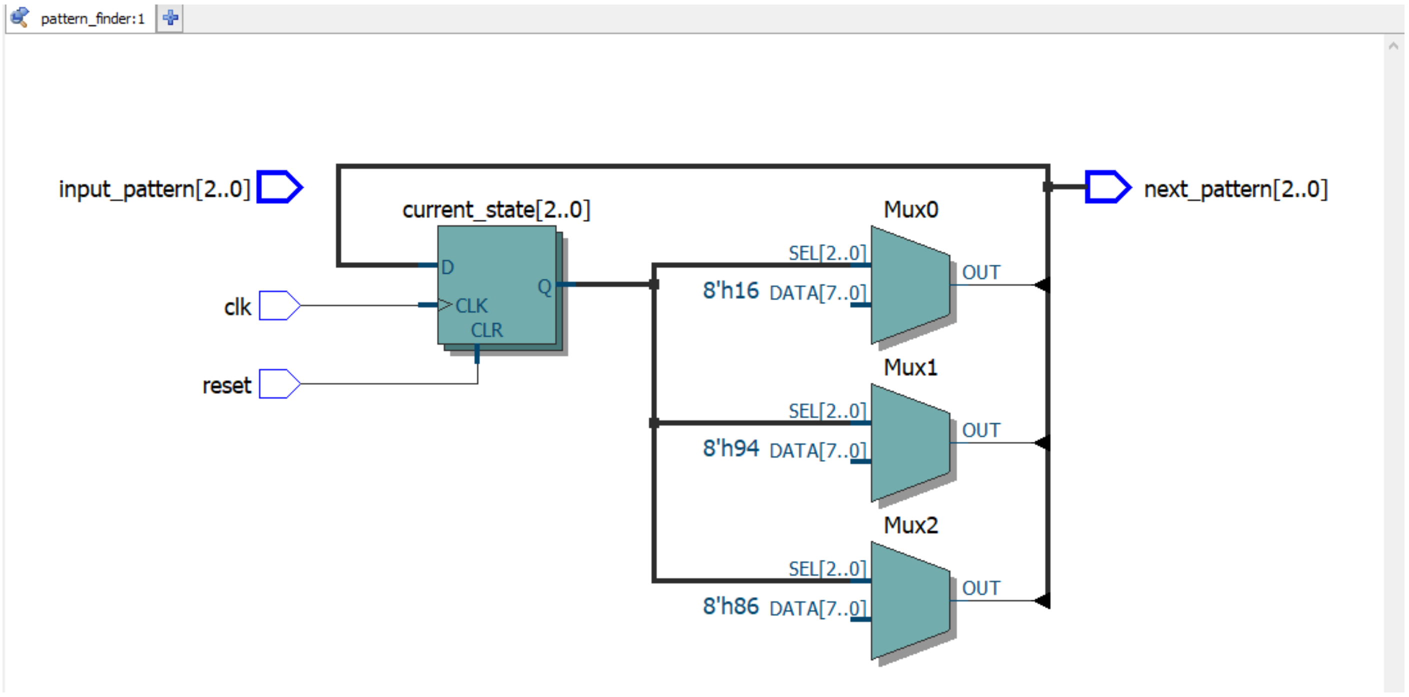
Compilation Report - pattern_finder

Flow Summary

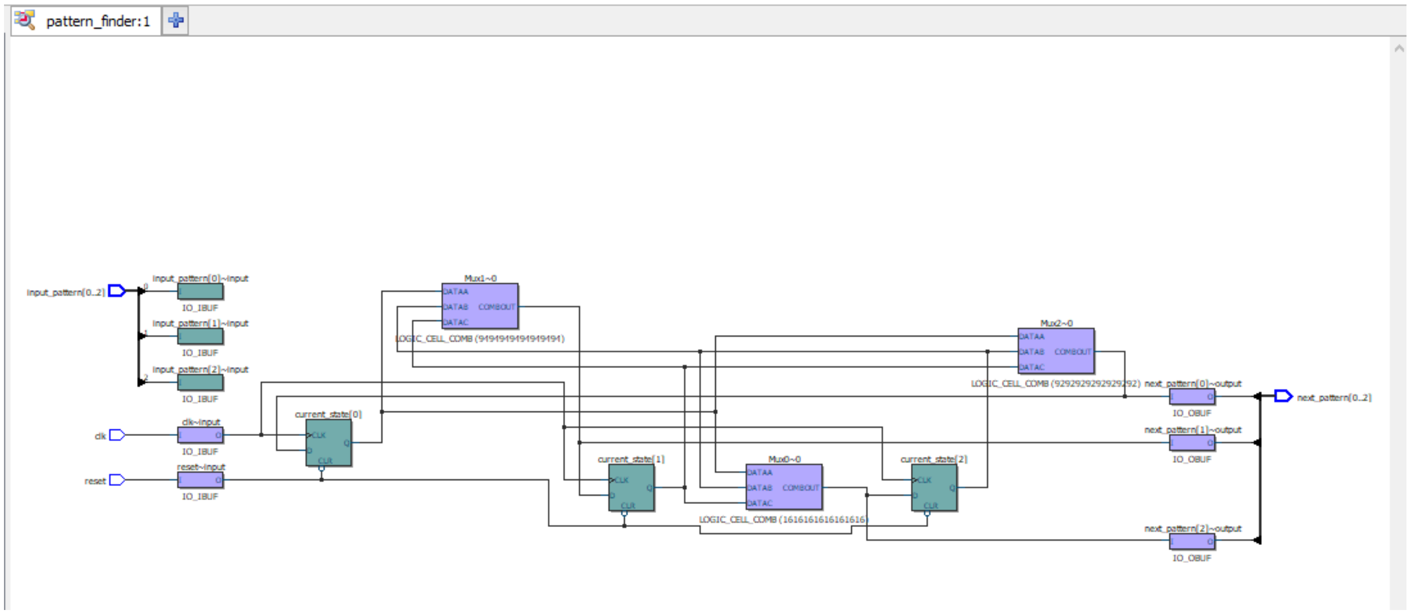
Flow Status	Successful - Fri May 10 21:26:47 2024
Quartus II 64-Bit Version	13.1.0 Build 162 10/23/2013 SJ Web Edition
Revision Name	pattern_finder
Top-level Entity Name	Pattern_Finder
Family	Cyclone V
Device	5CGXFC7C7F23C8
Timing Models	Final
Logic utilization (in ALMs)	N/A
Total registers	3
Total pins	8
Total virtual pins	0
Total block memory bits	0
Total DSP Blocks	0
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI TX PCSs	0
Total HSSI TX Channels	0
Total PLLs	0
Total DLLs	0

```
Type   ID     Message
       11104 Parallel Compilation has detected 8 hyper-threaded processors. However, the extra hyper-threaded processors will not be used by default. Parallel Compilation will use 4 of th
       12021 Found 2 design units, including 1 entities, in source file pattern_finder.vhd
       12127 Elaborating entity "pattern_finder" for the top level hierarchy
      286030 Timing-Driven Synthesis is running
       16010 Generating hard_block partition "hard_block:auto_generated_inst"
       21074 Design contains 3 input pin(s) that do not drive logic
       21057 Implemented 14 device resources after synthesis - the final resource count might be different
             Quartus II 64-Bit Analysis & Synthesis was successful. 0 errors, 4 warnings
```

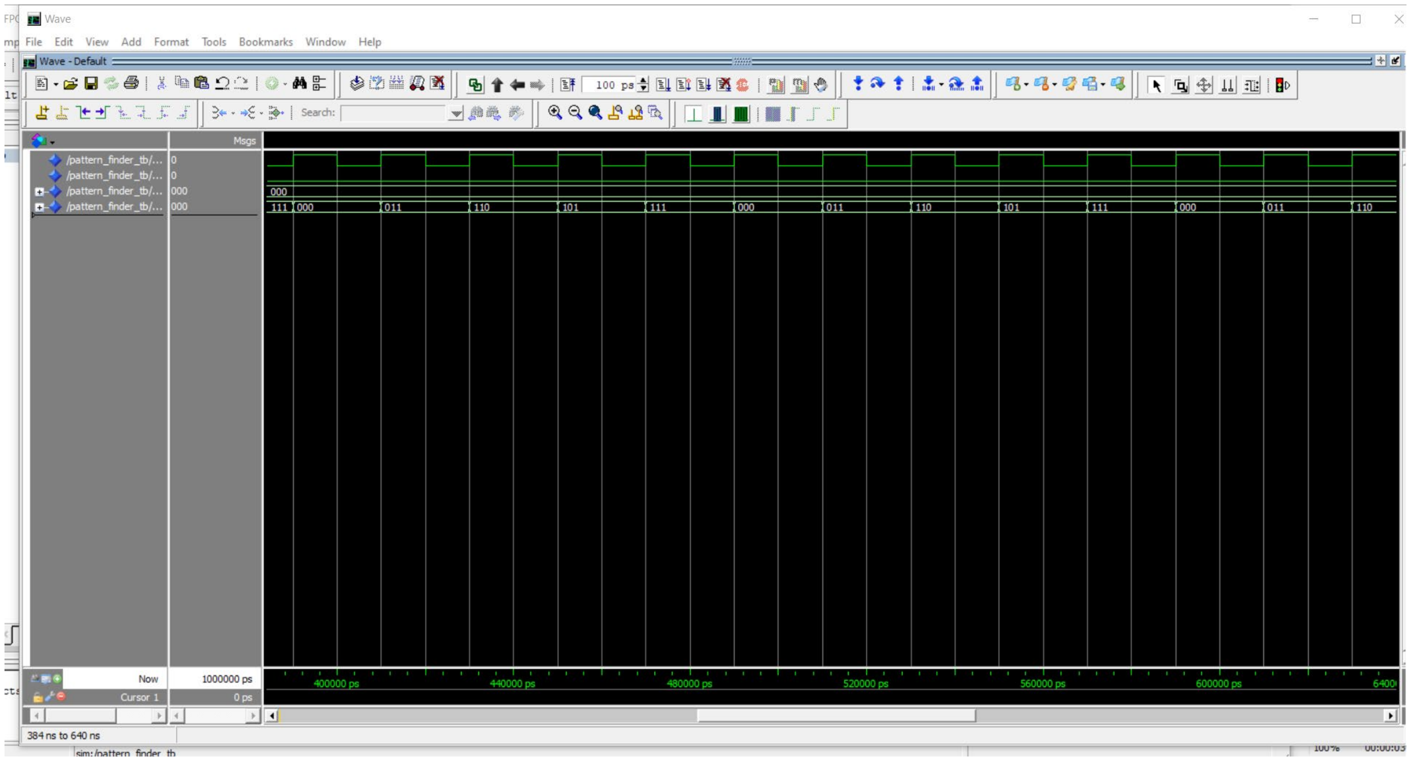
❖RTL view:



❖Post-mapping view:



❖Wave form:



❖Test bench code:

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;


-- Define the test bench entity

entity Pattern_Finder_TB is

-- Testbench does not have any ports

end Pattern_Finder_TB;


architecture behavior of Pattern_Finder_TB is

    -- Component Declaration for the Pattern_Finder

    component Pattern_Finder

        Port (

            clk : in STD_LOGIC;

            reset : in STD_LOGIC;
            input_pattern : in STD_LOGIC_VECTOR(2 downto 0);
            next_pattern : out STD_LOGIC_VECTOR(2 downto 0)
        );
    end component;

    -- Signals for interfacing with the component
    signal tb_clk : STD_LOGIC := '0';
    signal tb_reset : STD_LOGIC := '0';
    signal tb_input_pattern : STD_LOGIC_VECTOR(2 downto 0) := (others => '0');
    signal tb_next_pattern : STD_LOGIC_VECTOR(2 downto 0);

begin
    -- Clock process
    clk_process: process
    begin
        while true loop
            tb_clk <= '0';
            wait for 10 ns;  -- Clock period is 20 ns
            tb_clk <= '1';
            wait for 10 ns;
        end loop;
    end process;

    -- Instantiate the Unit Under Test (UUT)
    uut: Pattern_Finder port map (
        clk => tb_clk,
        reset => tb_reset,
        input_pattern => tb_input_pattern,
        next_pattern => tb_next_pattern
    );

    -- Stimulus process
    stim_proc: process
    begin
        -- Reset the device
        tb_reset <= '1';
        wait for 25 ns; -- wait more than one clock cycle
        tb_reset <= '0';
        wait for 15 ns;

        -- Loop through pattern states
        tb_input_pattern <= "000";  -- Start at pattern "000"
        wait for 20 ns; -- wait for one cycle
        tb_input_pattern <= "011";
        wait for 20 ns;
        tb_input_pattern <= "110";
        wait for 20 ns;
        tb_input_pattern <= "101";
        wait for 20 ns;
        tb_input_pattern <= "111";
        wait for 20 ns;
        tb_input_pattern <= "000";  -- Wrap around
        wait for 20 ns;

        -- Finish the simulation
        wait;
    end process;
end behavior;
```