IT 451- Computer Organization and Architecture LAB

Assignment 3

ISE Design Suite has been used for the simulation and synthesis of the design. VHDL is used as the hardware description language.

Suprotik Dey

IT 4th Semester

Roll: Hx 31

510815050

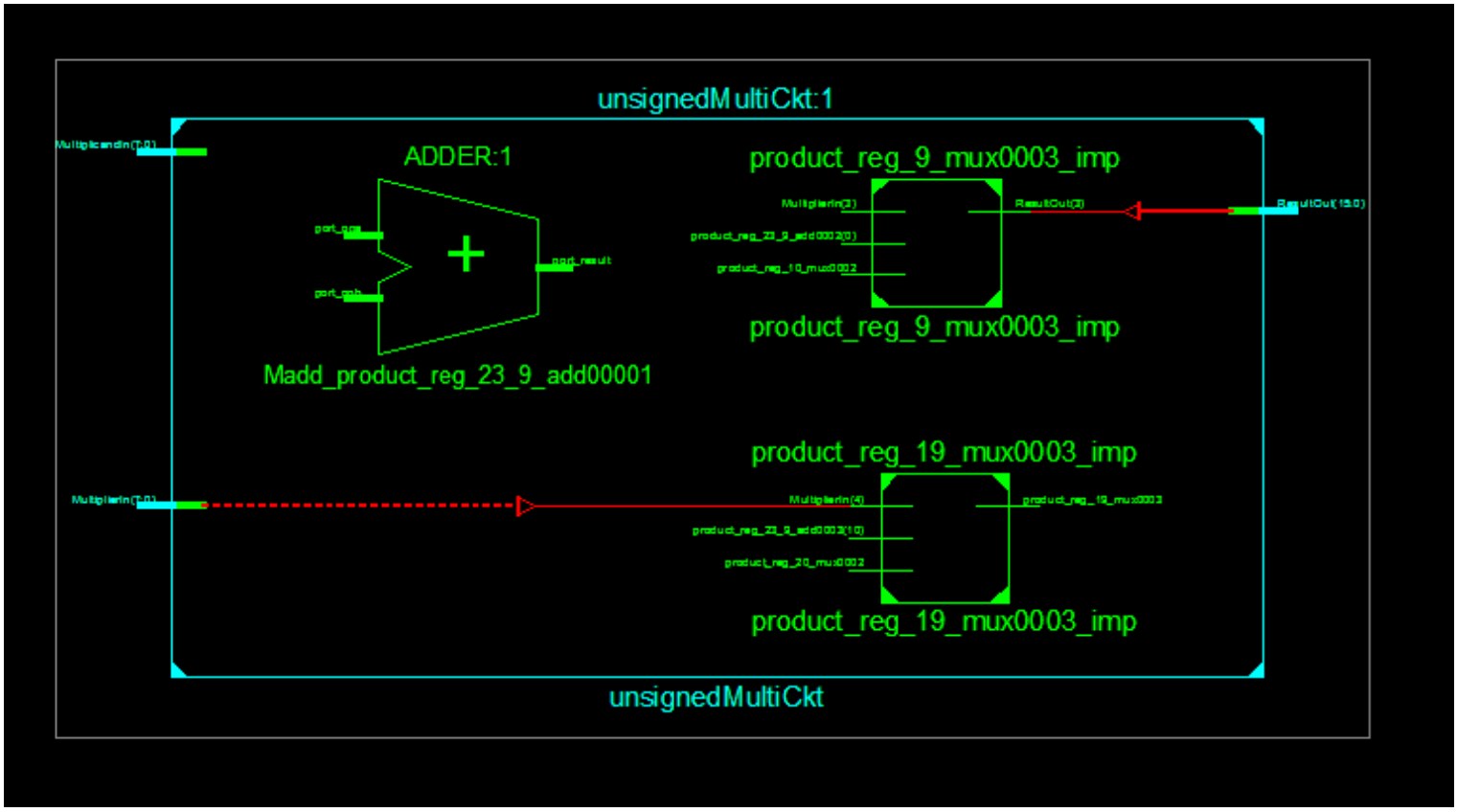
3. **Question:**

Design and simulate 8 bit unsigned binary multiplier.

VHDL Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;

entity unsignedMultiCkt is
   Port ( MultiplicandIn : in  STD_LOGIC_VECTOR (7 downto 0);
          MultiplierIn : in  STD_LOGIC_VECTOR (7 downto 0);
          ResultOut : out  STD_LOGIC_VECTOR (15 downto 0));
end unsignedMultiCkt;

architecture Behavioral of unsignedMultiCkt is

begin
process(MultiplicandIn, MultiplierIn)

variable Acc : std_logic_vector(8 downto 0);
variable product_reg : std_logic_vector(23 downto 0);

begin

Acc := '0' & MultiplicandIn;
product_reg := "0000000000000000" & MultiplierIn;

-- algorithm is to repeat shifting/adding
for i in 1 to 8 loop
if product_reg(0)='1' then
product_reg(23 downto 9) := product_reg(23 downto 9)
+ Acc;
end if;
product_reg(23 downto 0) := '0' & product_reg(23 downto 1);
end loop;

ResultOut <= product_reg(16 downto 1);

end process;
end Behavioral;
```

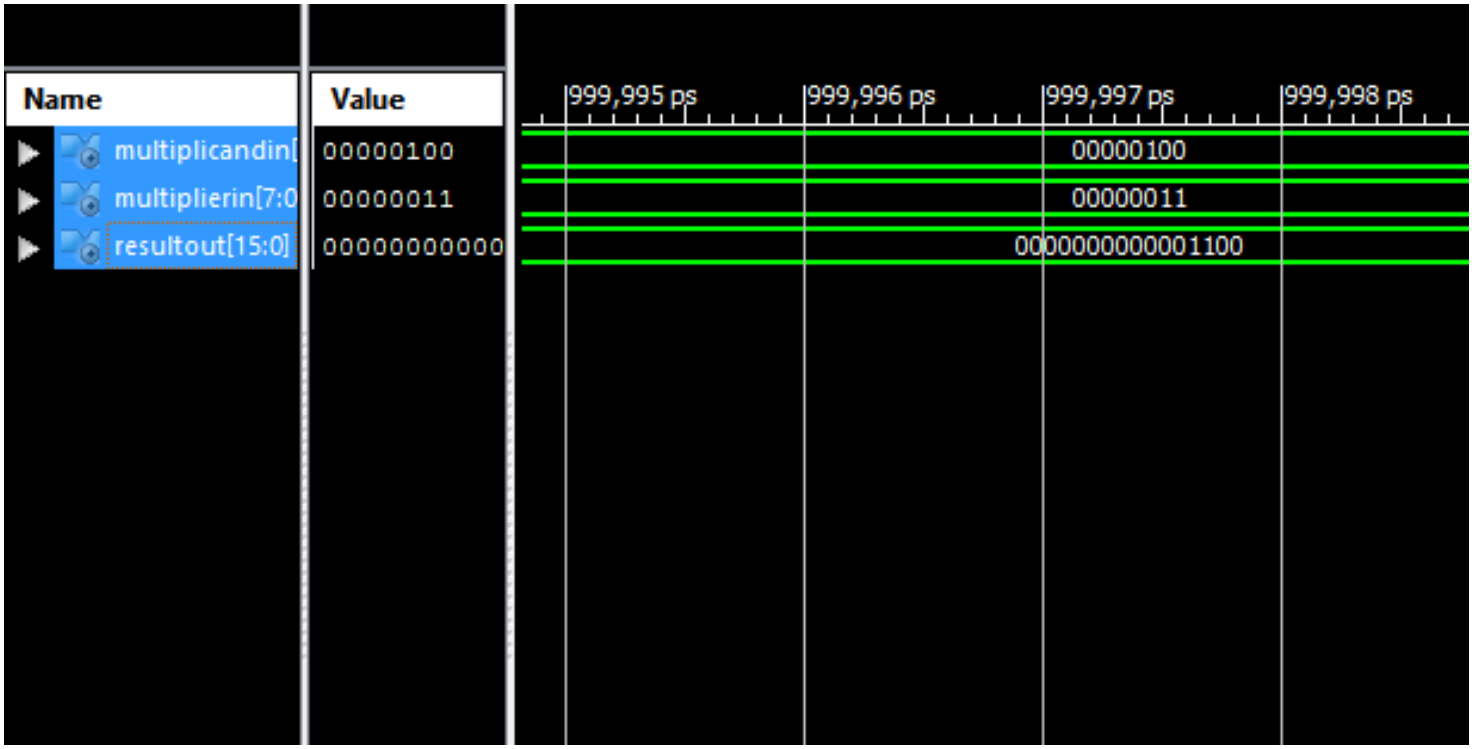
RTL template:



Test Bench Code:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
 ENTITY multitest IS
END multitest;
 ARCHITECTURE behavior OF multitest IS
    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT unsignedMultiCkt
   PORT(
        MultiplicandIn : IN  std_logic_vector(7 downto 0);
        MultiplierIn : IN  std_logic_vector(7 downto 0);
        ResultOut : OUT  std_logic_vector(15 downto 0)
       );
   END COMPONENT;
      --Inputs
  signal MultiplicandIn : std_logic_vector(7 downto 0) := (others => '0');
  signal MultiplierIn : std_logic_vector(7 downto 0) := (others => '0');

    --Outputs
  signal ResultOut : std_logic_vector(15 downto 0);

BEGIN
    -- Instantiate the Unit Under Test (UUT)
  uut: unsignedMultiCkt PORT MAP (
        MultiplicandIn => MultiplicandIn,
        MultiplierIn => MultiplierIn,
        ResultOut => ResultOut
       );
  -- Stimulus process
  stim_proc: process
  begin
    MultiplicandIn <= "00000100";
      MultiplierIn <= "00000011";

     wait;
  end process;

END;
```

Output:



The End