Lab Report 4
D-Latch, Flip Flops, 8-bit Register
ECE 238L

Kenneth Cox

April 9, 2021

Date Performed:	April 9, 2021
Instructor:	Professor Hamke

1 Items

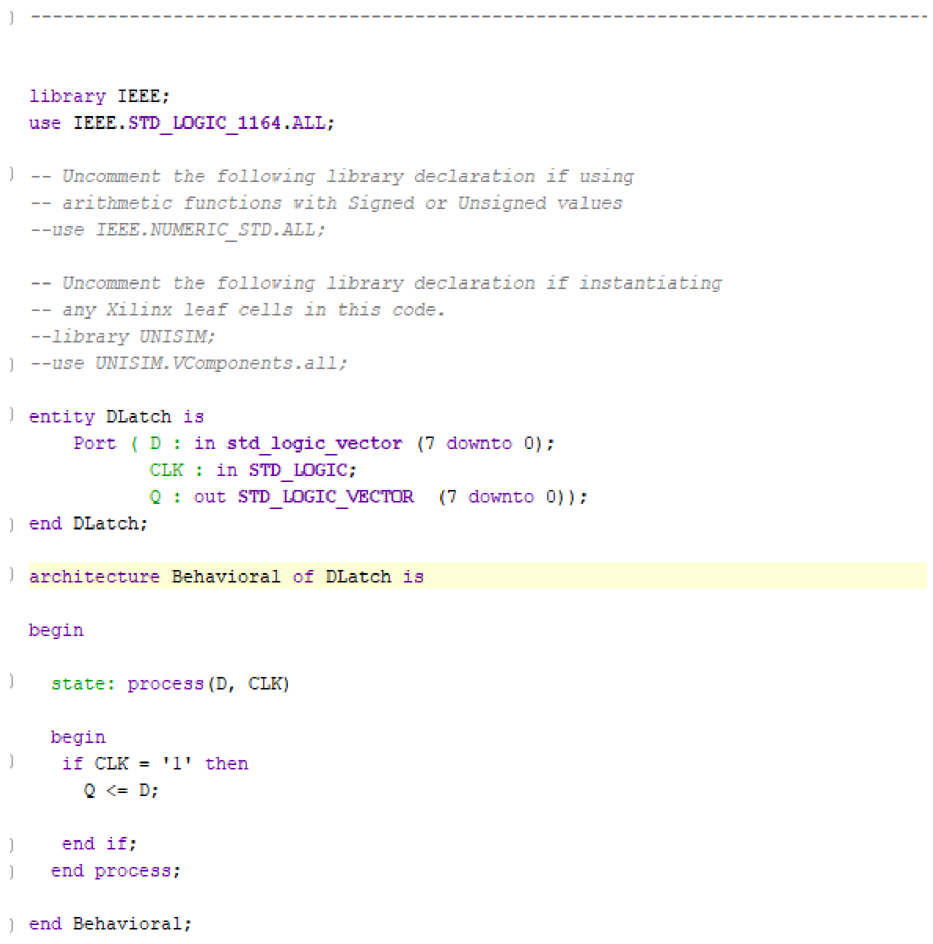
D-Latch

Flip Flop

8-Bit Register

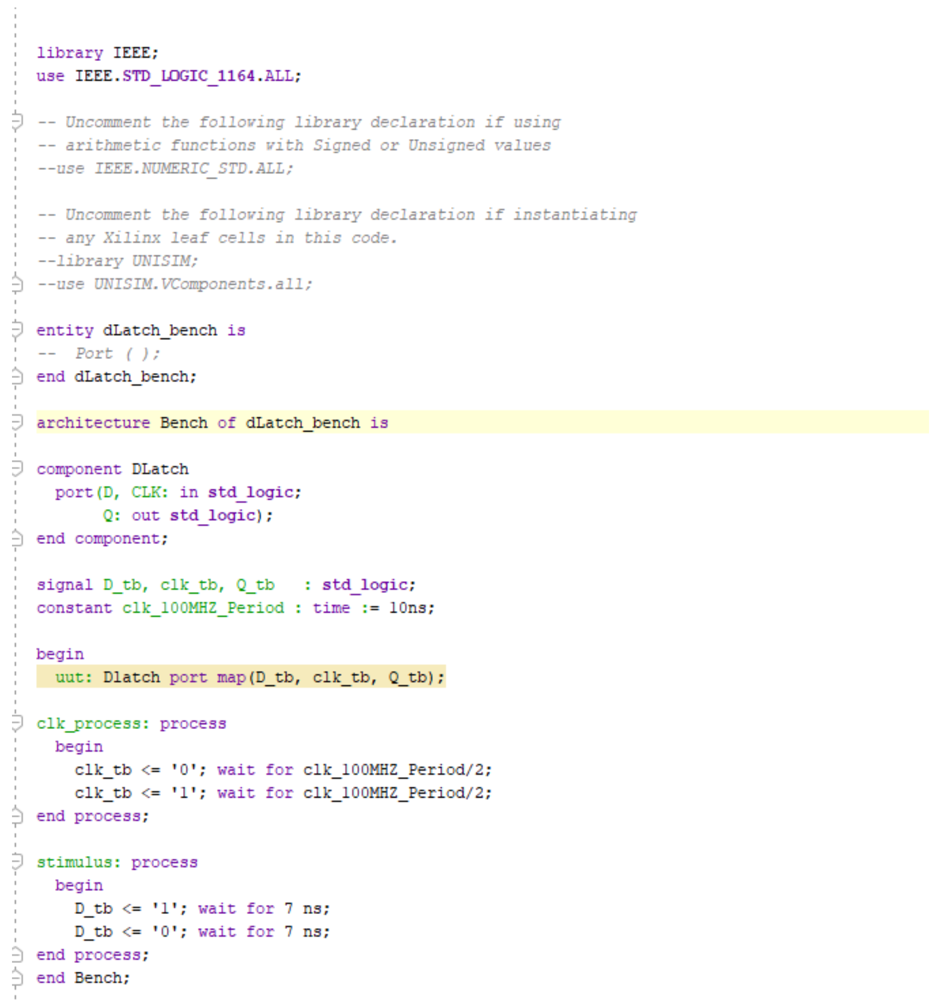
2 Feedback

The biggest problem I had was introducing my board into this lab. I have the BASYS 3, so the pins were different. Also, I wasnt able to readily identify which project part I needed, it was xc7a35tcpg236-1. Another obstucle I had to overcome was the abilty to set the clock to my SW0 pin/ override the default clock.



```
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity DLatch is
    Port ( D : in std_logic_vector (7 downto 0);
           CLK : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR  (7 downto 0));
end DLatch;

architecture Behavioral of DLatch is

begin

  state: process(D, CLK)

  begin
   if CLK = '1' then
     Q <= D;

   end if;
  end process;

end Behavioral;
```

Figure 1: D-Latch Source



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

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity dLatch_bench is
--  Port ( );
end dLatch_bench;

architecture Bench of dLatch_bench is

component DLatch
  port(D, CLK: in std_logic;
       Q: out std_logic);
end component;

signal D_tb, clk_tb, Q_tb   : std_logic;
constant clk_100MHZ_Period : time := 10ns;

begin
  uut: Dlatch port map(D_tb, clk_tb, Q_tb);

clk_process: process
  begin
    clk_tb <= '0'; wait for clk_100MHZ_Period/2;
    clk_tb <= '1'; wait for clk_100MHZ_Period/2;
end process;

stimulus: process
  begin
    D_tb <= '1'; wait for 7 ns;
    D_tb <= '0'; wait for 7 ns;
end process;
end Bench;
```

Figure 2: D-Latch Test Bench

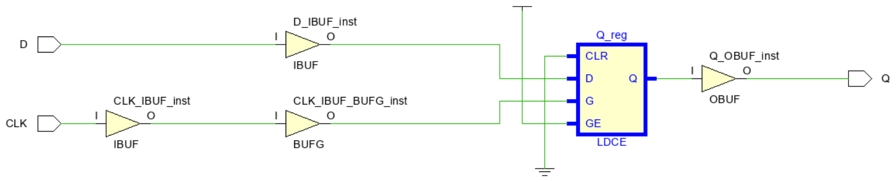
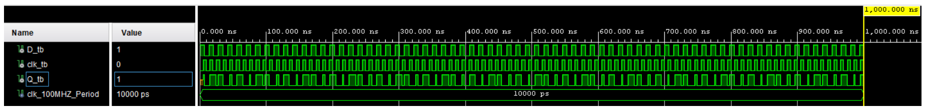
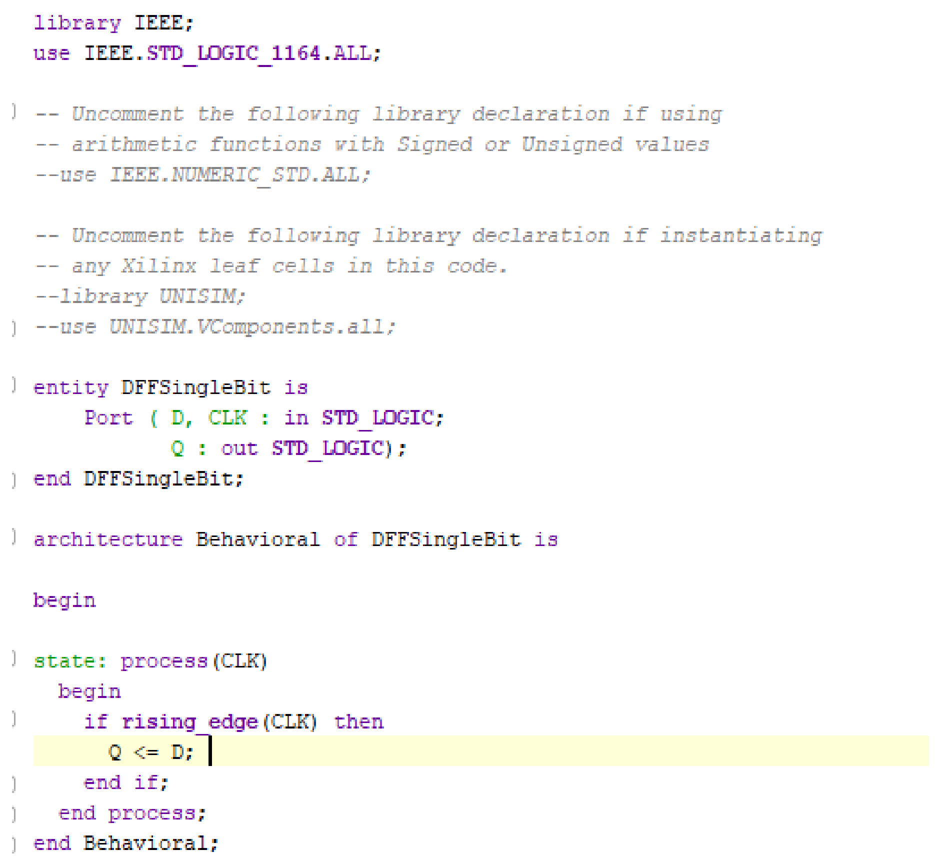


Figure 3: D-Latch Schematic



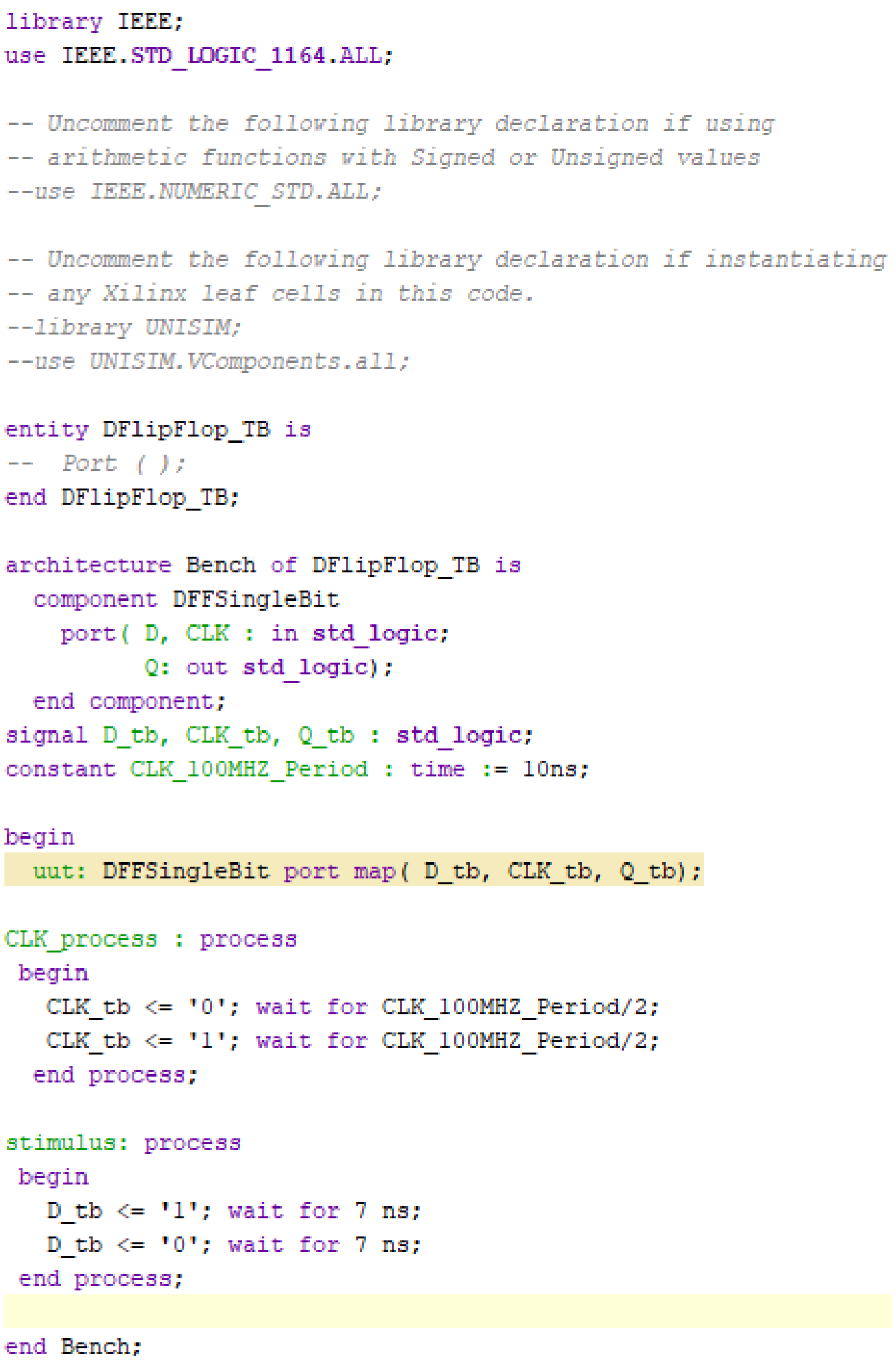
Name	Value
D_tb	1
clk_tb	0
Q_tb	1
clk_100MHZ_Period	10000 ps

Figure 4: D-Latch Wavefrom



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

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity DFFSingleBit is
    Port ( D, CLK : in STD_LOGIC;
           Q : out STD_LOGIC);
end DFFSingleBit;

architecture Behavioral of DFFSingleBit is

begin

state: process(CLK)
  begin
    if rising_edge(CLK) then
      Q <= D;
    end if;
  end process;
end Behavioral;
```

Figure 5: Flip Flop Source



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

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity DFlipFlop_TB is
--  Port ( );
end DFlipFlop_TB;

architecture Bench of DFlipFlop_TB is
  component DFFSingleBit
    port( D, CLK : in std_logic;
          Q: out std_logic);
  end component;
signal D_tb, CLK_tb, Q_tb : std_logic;
constant CLK_100MHZ_Period : time := 10ns;

begin
  uut: DFFSingleBit port map( D_tb, CLK_tb, Q_tb);

CLK_process : process
 begin
   CLK_tb <= '0'; wait for CLK_100MHZ_Period/2;
   CLK_tb <= '1'; wait for CLK_100MHZ_Period/2;
  end process;

stimulus: process
 begin
   D_tb <= '1'; wait for 7 ns;
   D_tb <= '0'; wait for 7 ns;
 end process;

end Bench;
```

Figure 6: Flip Flop Test bench

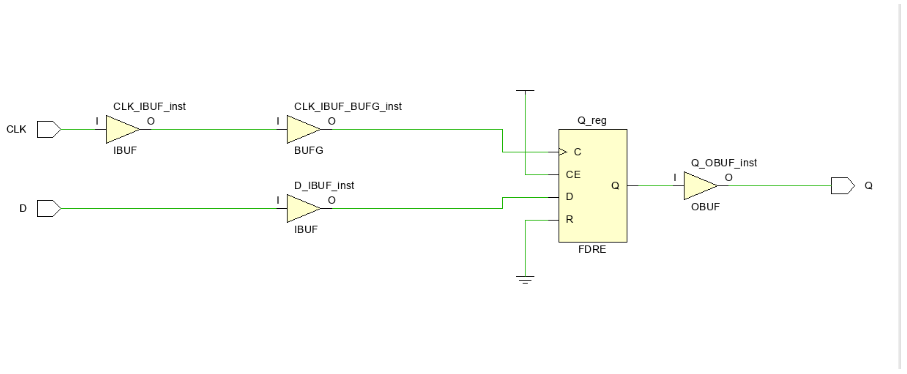


Figure 7: Flip Flop Schematic

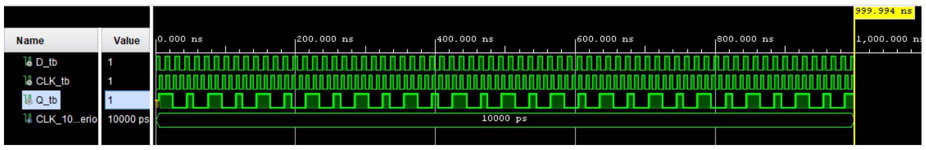
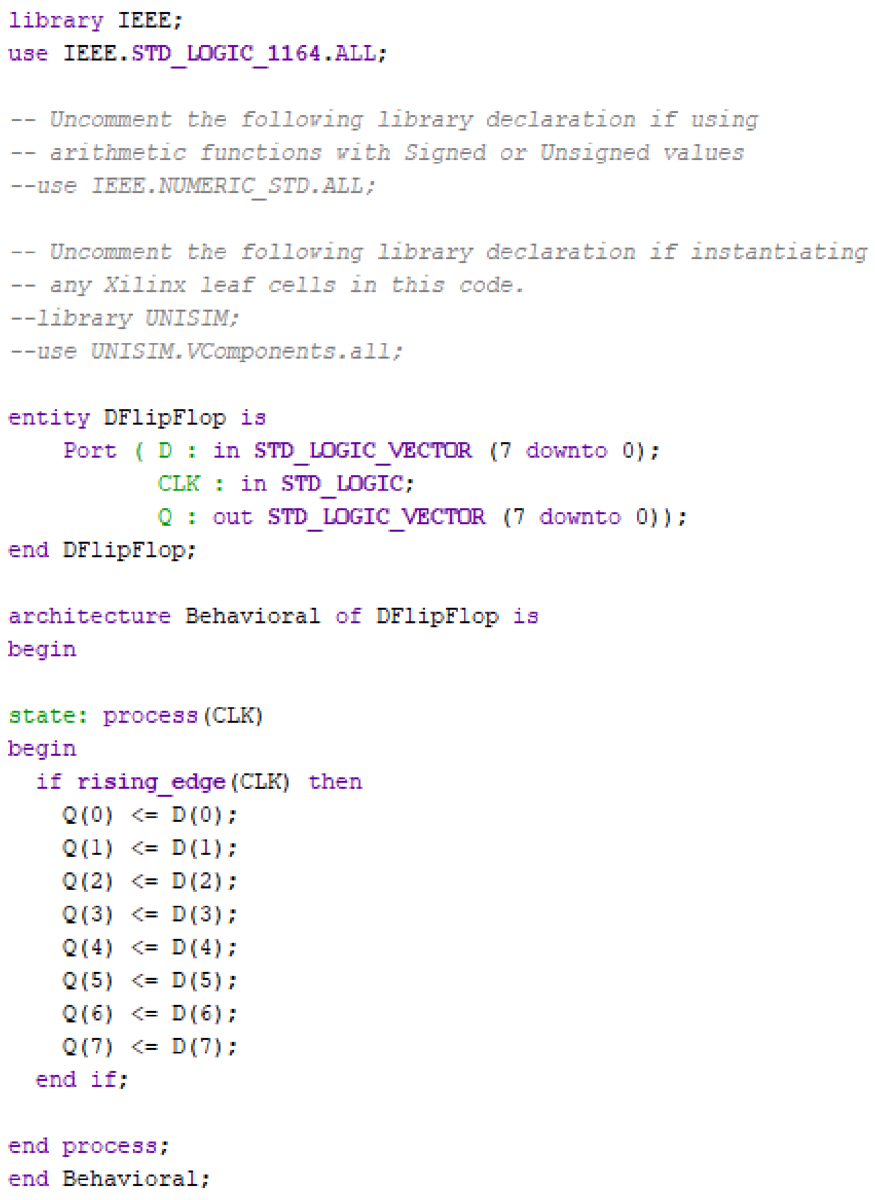
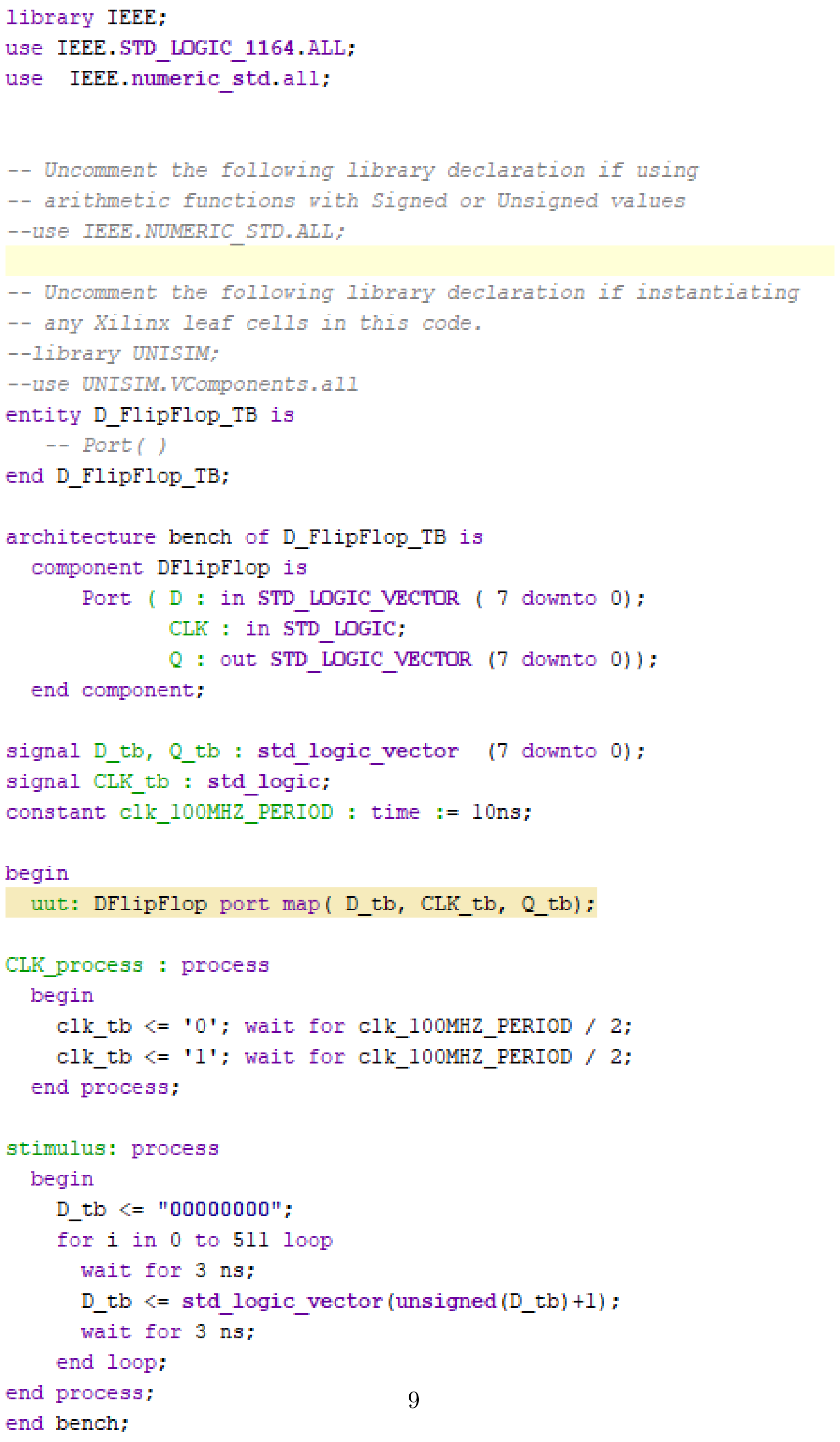


Figure 8: Flip Flop Waveform



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

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity DFlipFlop is
    Port ( D : in STD_LOGIC_VECTOR (7 downto 0);
           CLK : in STD_LOGIC;
           Q : out STD_LOGIC_VECTOR (7 downto 0));
end DFlipFlop;

architecture Behavioral of DFlipFlop is
begin

state: process(CLK)
begin
  if rising_edge(CLK) then
    Q(0) <= D(0);
    Q(1) <= D(1);
    Q(2) <= D(2);
    Q(3) <= D(3);
    Q(4) <= D(4);
    Q(5) <= D(5);
    Q(6) <= D(6);
    Q(7) <= D(7);
  end if;

end process;
end Behavioral;
```

Figure 9: 8-Bit Register Source



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


-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all
entity D_FlipFlop_TB is
   -- Port ( )
end D_FlipFlop_TB;

architecture bench of D_FlipFlop_TB is
  component DFlipFlop is
      Port ( D : in STD_LOGIC_VECTOR ( 7 downto 0);
             CLK : in STD_LOGIC;
             Q : out STD_LOGIC_VECTOR (7 downto 0));
  end component;

signal D_tb, Q_tb : std_logic_vector  (7 downto 0);
signal CLK_tb : std_logic;
constant clk_100MHZ_PERIOD : time := 10ns;

begin
  uut: DFlipFlop port map( D_tb, CLK_tb, Q_tb);

CLK_process : process
  begin
    clk_tb <= '0'; wait for clk_100MHZ_PERIOD / 2;
    clk_tb <= '1'; wait for clk_100MHZ_PERIOD / 2;
  end process;

stimulus: process
  begin
    D_tb <= "00000000";
    for i in 0 to 511 loop
      wait for 3 ns;
      D_tb <= std_logic_vector(unsigned(D_tb)+1);
      wait for 3 ns;
    end loop;
end process;
end bench;
```

Figure 10: 8-Bit Register Test Bench

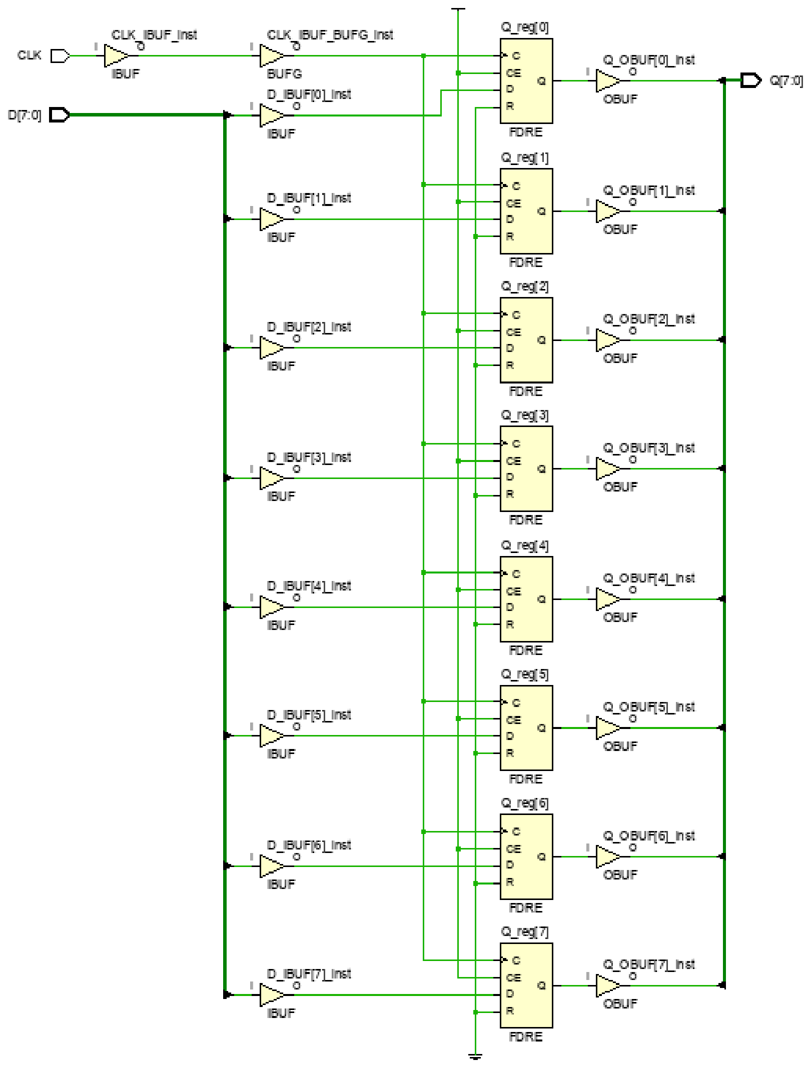
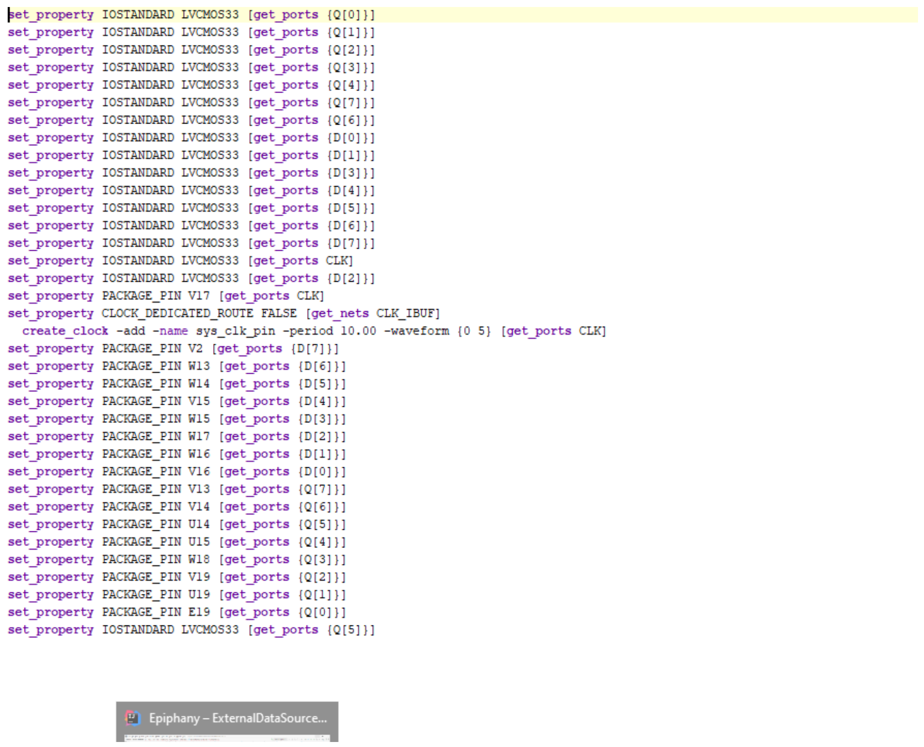


Figure 11: 8-Bit Register Schematic



```
set_property IOSTANDARD LVCMOS33 [get_ports {Q[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[2]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[3]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[4]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[5]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[6]}]
set_property IOSTANDARD LVCMOS33 [get_ports {D[7]}]
set_property IOSTANDARD LVCMOS33 [get_ports CLK]
set_property IOSTANDARD LVCMOS33 [get_ports {D[2]}]
set_property PACKAGE_PIN V17 [get_ports CLK]
set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets CLK_IBUF]
  create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports CLK]
set_property PACKAGE_PIN V2 [get_ports {D[7]}]
set_property PACKAGE_PIN W13 [get_ports {D[6]}]
set_property PACKAGE_PIN W14 [get_ports {D[5]}]
set_property PACKAGE_PIN V15 [get_ports {D[4]}]
set_property PACKAGE_PIN W15 [get_ports {D[3]}]
set_property PACKAGE_PIN W17 [get_ports {D[2]}]
set_property PACKAGE_PIN W16 [get_ports {D[1]}]
set_property PACKAGE_PIN V16 [get_ports {D[0]}]
set_property PACKAGE_PIN V13 [get_ports {Q[7]}]
set_property PACKAGE_PIN V14 [get_ports {Q[6]}]
set_property PACKAGE_PIN U14 [get_ports {Q[5]}]
set_property PACKAGE_PIN U15 [get_ports {Q[4]}]
set_property PACKAGE_PIN W18 [get_ports {Q[3]}]
set_property PACKAGE_PIN V19 [get_ports {Q[2]}]
set_property PACKAGE_PIN U19 [get_ports {Q[1]}]
set_property PACKAGE_PIN E19 [get_ports {Q[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {Q[5]}]
```

Figure 12: 8-Bit Register Constraints

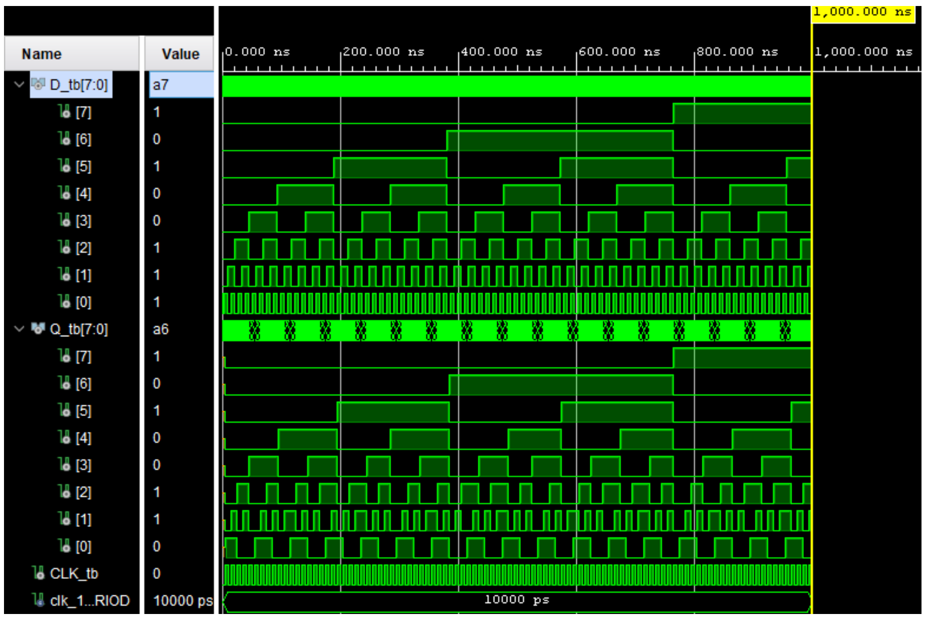


Figure 13: 8-Bit Register Wavefrom