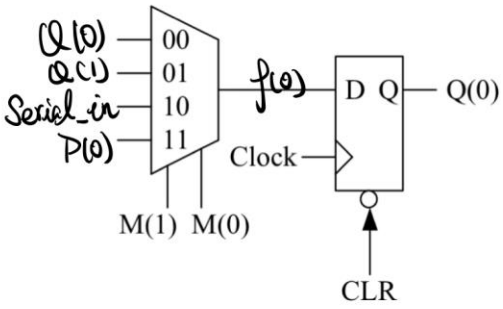
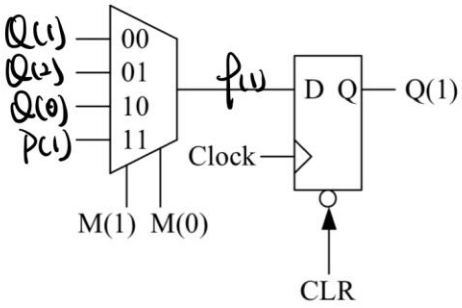
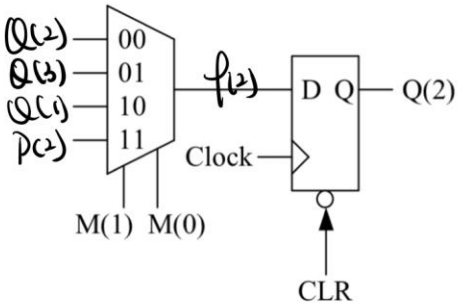
EE 115B, Fall 2021

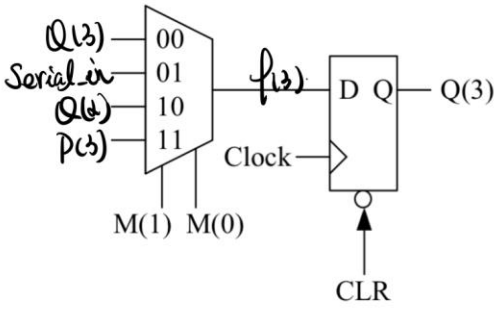
Student Name: 朱宇轩 2020531016 Lab 3

1 Schematic (20 points.)









2 Code for Components (10 points. 5 points each.)

2.1 D flip flop

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

entity Dflipflop is
   Port ( D : in STD_LOGIC;
          clr : in STD_LOGIC;
          clock : in STD_LOGIC;
          Q : buffer STD_LOGIC);
end Dflipflop;

architecture Behavioral of Dflipflop is

begin
   process (clr ,clock )
   begin
      if clr = '0' then
         Q <= '0';
      elsif clock'Event and clock = '1' then
         Q <= D;
      end if ;
   end process ;
end Behavioral;
```

2.2 mux4to1

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

entity mux4to1 is
   Port ( w0 : in STD_LOGIC;
          w1 : in STD_LOGIC;
          w2 : in STD_LOGIC;
          w3 : in STD_LOGIC;
          M : in STD_LOGIC_VECTOR (1 downto 0);
          f : buffer STD_LOGIC);
end mux4to1;

architecture Behavioral of mux4to1 is

begin
   with M select
      f <= w0 when "00",
           w1 when "01",
           w2 when "10",
           w3 when others;
end Behavioral;
```

3 Code for Register (30 points.)

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

entity shiftreg is
   Port (  P : IN std_logic_vector(3 downto 0);
         Q : buffer std_logic_vector(3 downto 0);
         clock : IN std_logic;
         clr : IN std_logic;
         serial_in : IN std_logic;
         M : IN std_logic_vector(1 downto 0)
         );
end shiftreg;

architecture Structure of shiftreg is
   signal f : std_logic_vector (3 downto 0 );

component mux4to1
   port ( w0 : in STD_LOGIC;
        w1 : in STD_LOGIC;
        w2 : in STD_LOGIC;
        w3 : in STD_LOGIC;
        M : in STD_LOGIC_VECTOR (1 downto 0);
        f : buffer STD_LOGIC);
end component ;

component Dflipflop
   port ( D : in STD_LOGIC;
        clr : in STD_LOGIC;
```

```
        clock : in STD_LOGIC;
        Q : buffer STD_LOGIC);
end component ;



begin
   mux0: mux4to1 port map
      (Q(0),Q(1),serial_in ,P(0),M(1 downto 0),f(0));
   mux1: mux4to1 port map
      (Q(1),Q(2),Q(0),P(1),M(1 downto 0),f(1));
   mux2: mux4to1 port map
      (Q(2),Q(3),Q(1) ,P(2),M(1 downto 0),f(2));
   mux3: mux4to1 port map
      (Q(3),serial_in ,Q(2),P(3),M(1 downto 0),f(3));


   D0: Dflipflop  port map
      (f(0),clr ,clock ,Q(0));
   D1: Dflipflop  port map
      (f(1),clr ,clock ,Q(1));
   D2: Dflipflop  port map
      (f(2),clr ,clock ,Q(2));
   D3: Dflipflop  port map
      (f(3),clr ,clock ,Q(3));
end Structure;
```

4 Code for Testbench (20 points.)

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

ENTITY shiftreg_test IS
END shiftreg_test;

ARCHITECTURE behavior OF shiftreg_test IS
-- Component Declaration for the Unit Under Test (UUT)
COMPONENT shiftreg
  PORT(
       P : IN std_logic_vector(3 downto 0);
       Q : buffer std_logic_vector(3 downto 0);
       clock : IN std_logic;
       clr : IN std_logic;
       serial_in : IN std_logic;
       m : IN std_logic_vector(1 downto 0)
       );
END COMPONENT;

--Inputs
signal P : std_logic_vector(3 downto 0) := (others => '0');
signal clock : std_logic := '0';
signal clr : std_logic := '0';
signal serial_in : std_logic := '0';
signal m : std_logic_vector(1 downto 0) := (others => '0');
--Outputs
signal Q : std_logic_vector(3 downto 0);
```

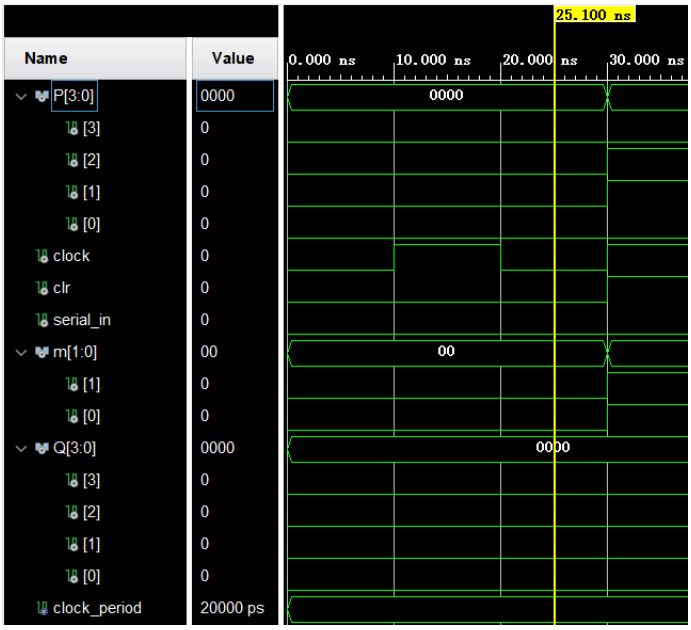
```
-- Clock period definitions
constant clock_period : time := 20ns;
BEGIN
-- Instantiate the Unit Under Test (UUT)
uut: shiftreg PORT MAP (
                P => P,
                Q => Q,
                clock => clock,
                clr => clr,
                serial_in => serial_in,
                m => m
                );

-- Clock process definitions
clock_process :process
begin
   clock <= '0';
   wait for clock_period/2;
   clock <= '1';
   wait for clock_period/2;
end process;

-- Stimulus process
stim_proc: process
begin
   -- test CLR
   clr<='0';
   wait for 30ns;
```

```
    -- test parallel load
    clr<='1';
    m<="11";
    P<="0110";
    wait for 40ns;

    -- test right shift
    m<="01";
    serial_in<='1';
    wait for 80ns; -- shift 4 bits

    -- test left shift
    m<="10";
    serial_in <='0';
    wait for 80ns; -- shift 4 bits

    -- test no change
    m<="00";
    wait for 10ns;
    serial_in <='1';
    P<="1111";
    wait for 80ns;
    wait;

end process;
END;
```

5 Timing Diagram (20 points.)

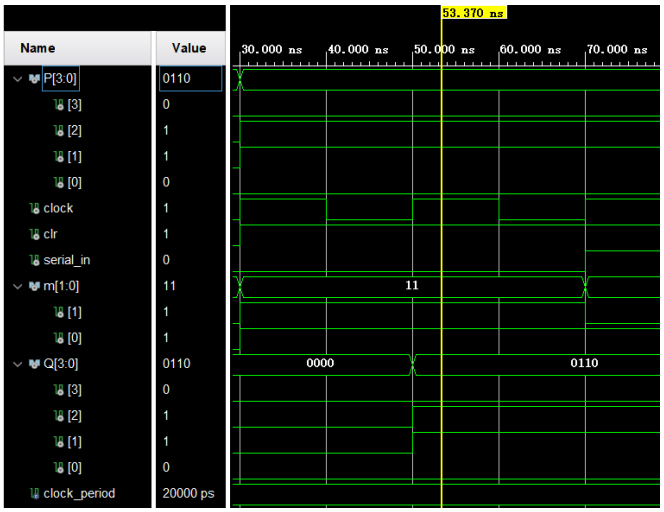
0-30ns: test CLR

All the outputs (Q(3:0)) maintains the value of 0 when clr=0.



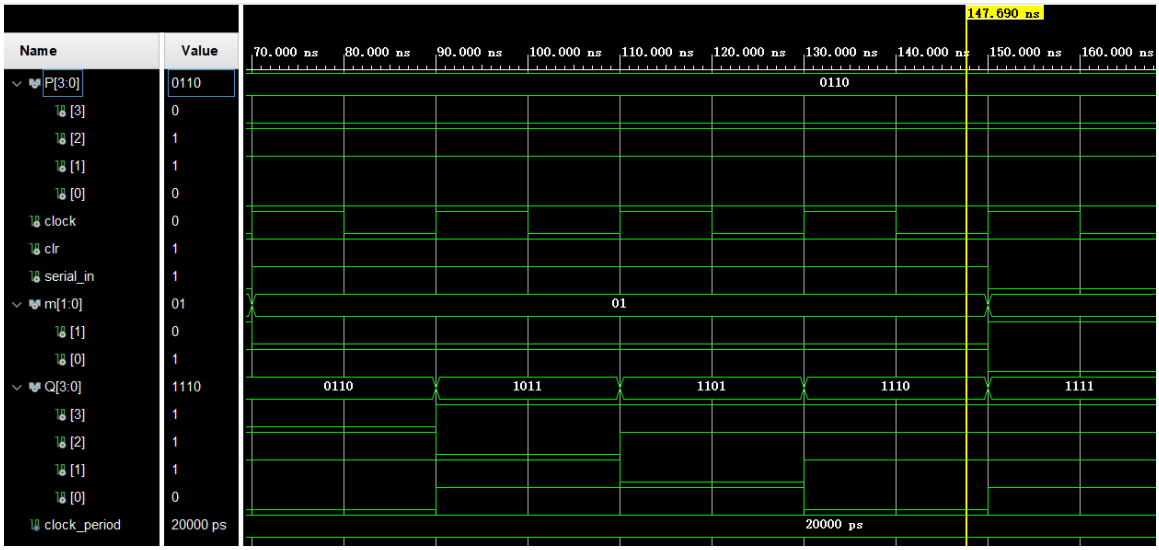
30-70ns: test parallel load

After the inputs have been set to the expected value, which is P(3:0)=”0110”, and waiting for a rising edge of the clock, the outputs turn into Q(3:0)=P(3:0)=”0110”, which verify the function of parallel load.



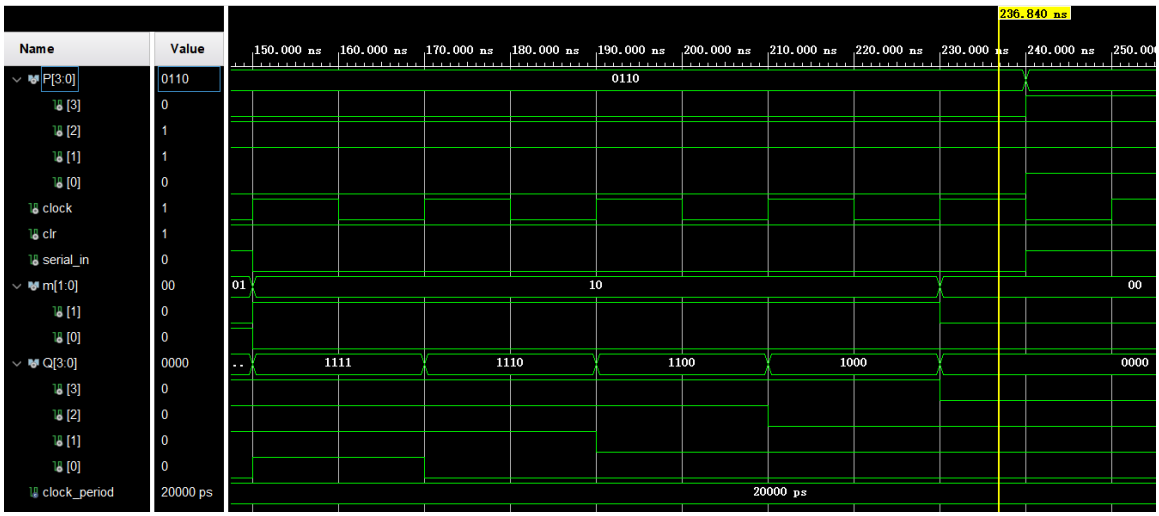
70-150ns: test right shift

Each rising edge of the clock makes all the data shifts to its right bit. And for Q(3), it uses the value of serial_in, which always equal to 1. Moreover, the last rising edge happens at 150ns, thus the 4th shift result is shown in 150-170ns. Then from the simulation, we can see that the value of Q(3:0) changes as we expected, thus we verify the function of right shift.



150-230ns: test left shift

Each rising edge of the clock makes all the data shifts to its left bit. And for Q(0), it uses the value of serial_in, which always equal to 0. Moreover, the last rising edge happens at230ns, thus the 4th shift result is shown in 230-250ns. Then from the simulation, we can see that the value of Q(3:0) changes as we expected, thus we verify the function of left shift.



230- ∞ ns: test no change

After setting M(1:0)=”00”, we change the value of P(3:0) and serial_in for example, and we can see the value of Q(3:0) hold it original value at 230ns, thus we verify the function of no change.