TaaviSalum Small error fix at exercise 4 ... 18 seconds ago History

..

Pictures	4 minutes ago
flip_flops	4 minutes ago
README.md	18 seconds ago

README.md

Digital-electronics-1

Laboratory #7

Exercise 1: Preparation tasks

D-type flip-flop equation and truth table

$q^D_{n+1} = d$

clk	d	q(n)	q(n+1)	Comments
↑	0	0	0	Remembers values, no change
↑	0	1	0	Remembers values, no change
↑	1	0	1	Sampled abd stored

clk	d	q(n)	q(n+1)	Comments
↑	1	1	1	Sampled and stored

JK-type flip-flop equation and truth table

$q_{n+1}^{JK} = j \cdot \overline{q_n} + \overline{k} \cdot q_n$

clk	j	k	q(n)	q(n+1)	Comments
↑	0	0	0	0	No change
↑	0	0	1	1	No change
↑	0	1	0	0	Reset
↑	0	1	1	0	Reset
↑	1	0	0	1	Set
↑	1	0	1	1	Set
↑	1	1	0	1	Toggle (Invert)
↑	1	1	1	0	Toggle (Invert)

T-type flip-flop equation and truth table

$q_{n+1}^{T} = t \cdot \overline{q_n} + \overline{t} \cdot q_n$

clk	t	q(n)	q(n+1)	Comments
↑	0	0	0	No change
↑	0	1	1	No change
↑	1	0	1	Invert (Toggle)
↑	1	1	0	Invert (Toggle)

Exercise 2: D latch

VHDL code of the process `p_d_latch`:

```
    p_d_latch : process (en, d, arst)
       begin
            if (arst = '1') then
                q       <= '0';
                q_bar   <= '1';
            elsif (en = '1') then
                q       <= d;
                q_bar   <= not d;
        end if;
    end process p_d_latch;
```

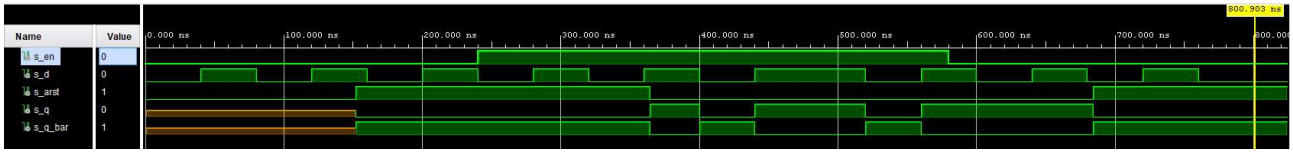
VHDL testbench file `tb_d_latch` :

```
    ----------------------------------------------------------------------
    -- Reset generation process
    ----------------------------------------------------------------------
        p_reset_gen : process
            begin
               s_arst <= '0';
               wait for 152 ns;
               s_arst <= '1';  -- Reset
               wait for 212 ns;
               s_arst <= '0';
               wait for 320 ns;
               s_arst <= '1';  -- Reset
               wait;
        end process p_reset_gen;

    ----------------------------------------------------------------------
    -- Data generation process
    ----------------------------------------------------------------------
        p_stimulus : process
            begin
               report "Stimulus process started" severity note;

               s_en <= '0';
               s_d  <= '0';
               wait for 40 ns;
               s_d  <= '1';
               wait for 40 ns;
               s_d  <= '0';
               wait for 40 ns;
               s_d  <= '1';
               wait for 40 ns;
               s_d  <= '0';
               wait for 20 ns;

               assert ((s_arst = '1') and (s_en = '0') and (s_q = '0') and (s_q_bar = '1
```

```
        report "Test failed when RESET is '1', EN is '0' and D is '0'" severity e

        wait for 20 ns;
        s_d  <= '1';
        wait for 20 ns;

        assert ((s_arst = '1') and (s_en = '0') and (s_q = '0') and (s_q_bar = '1
        report "Test failed when RESET is '1', EN is '0' and D is '1'" severity e

        wait for 20 ns;
        s_d  <= '0';
        s_en <= '1';
        wait for 40 ns;
        s_d  <= '1';
        wait for 20 ns;

        assert ((s_arst = '1') and (s_en = '1') and (s_q = '0') and (s_q_bar = '1
        report "Test failed when RESET is '1', EN is '1' and D is '1'" severity e

        wait for 20 ns;
        s_d  <= '0';
        wait for 20 ns;

        assert ((s_arst = '1') and (s_en = '1') and (s_q = '0') and (s_q_bar = '1
        report "Test failed when RESET is '1', EN is '1' and D is '0'" severity e

        wait for 20 ns;
        s_d  <= '1';
        wait for 40 ns;
        s_d  <= '0';
        wait for 40 ns;
        s_d  <= '1';
        wait for 20 ns;

        assert ((s_arst = '0') and (s_en = '1') and (s_q = '1') and (s_q_bar = '0
        report "Test failed when RESET is '0', EN is '1' and D is '1'" severity e

        wait for 60 ns;
        s_d  <= '0';
        wait for 20 ns;

        assert ((s_arst = '0') and (s_en = '1') and (s_q = '0') and (s_q_bar = '1
        report "Test failed when RESET is '0', EN is '1' and D i '0'" severity er

        wait for 20 ns;
        s_d  <= '1';
        wait for 20 ns;
        s_en <= '0';
        wait for 20 ns;
        s_d  <= '0';
        wait for 40 ns;
```

```
        s_d  <= '1';
        wait for 40 ns;
        s_d  <= '0';
        wait for 40 ns;
        s_d  <= '1';
        wait for 40 ns;
        s_d  <= '0';

        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;
```

Simulated time waveforms



Exercise 3: Flip-flops

VHDL code of the process `p_d_ff_arst` :

```
    p_d_ff_arst : process (clk, arst)
        begin
           if (arst = '1') then
              q     <= '0';
              q_bar <= '1';
           elsif rising_edge(clk) then
              q     <= d;
              q_bar <= not d;
           end if;
    end process p_d_ff_arst;
```

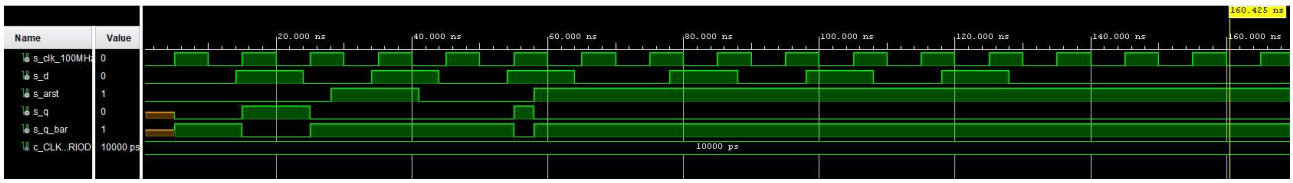
VHDL testbench file `tb_d_ff_arst` :

```
  ----------------------------------------------------------------------
  -- Clock generation process
  ----------------------------------------------------------------------
      p_clk_gen : process
          begin
              while now < 40 ms loop
```

```
                s_clk_100MHz <= '0';
                wait for c_CLK_100MHZ_PERIOD / 2;
                s_clk_100MHz <= '1';
                wait for c_CLK_100MHZ_PERIOD / 2;
            end loop;
            wait;
    end process p_clk_gen;

    --------------------------------------------------------------------
    -- Reset generation process
    --------------------------------------------------------------------

     p_reset_gen : process
        begin
            s_arst <= '0';
            wait for 28 ns;
            s_arst <= '1';  -- Reset
            wait for 13 ns;
            s_arst <= '0';
            wait for 17 ns;
            s_arst <= '1';
            wait for 693 ns;
            s_arst <= '1';  -- Reset
            wait;
     end process p_reset_gen;

    --------------------------------------------------------------------
    -- Data generation process
    --------------------------------------------------------------------
    p_stimulus : process
        begin
            report "Stimulus process started" severity note;

            s_d  <= '0';
            wait for 14 ns;
            s_d  <= '1';
            wait for 2 ns;

            assert ((s_arst = '0') and (s_q = '1') and (s_q_bar = '0'))
            report "Test failed when RESET is '0', CLK is RISING and D is '1'" sever

            wait for 8 ns;
            s_d  <= '0';
            wait for 10 ns;
            s_d  <= '1';
            wait for 10 ns;
            s_d  <= '0';
            wait for 10 ns;
            s_d  <= '1';
            wait for 5 ns;
```

```
        assert ((s_arst = '1') and (s_q = '0') and (s_q_bar = '1'))
        report "Test failed when RESET is '1', CLK is FALLING and D is '1'" seve

        wait for 5 ns;
        s_d  <= '0';
        wait for 14 ns;
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';
        wait for 10 ns;
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';
        wait for 10 ns;
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';

        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;
```

Simulated time waveforms



VHDL code of the process `p_d_ff_rst` :

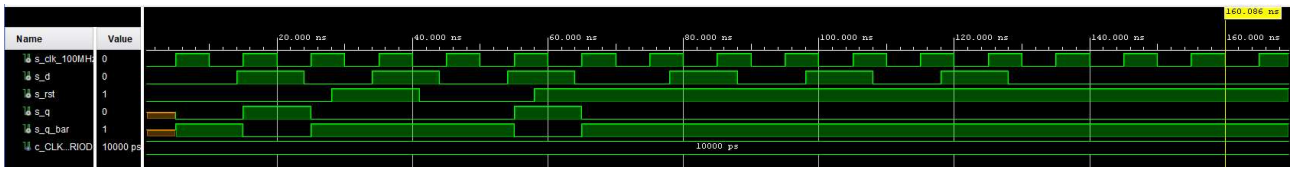
```
    p_d_ff_rst : process (clk)
        begin
            if rising_edge(clk) then
                if (rst = '1') then
                    q     <= '0';
                    q_bar <= '1';
                else
                    q     <= d;
                    q_bar <= not d;
                end if;
            end if;
    end process p_d_ff_rst;
```

VHDL testbench file `tb_d_ff_rst` :

```
    ----------------------------------------------------------------------
    -- Clock generation process
    ----------------------------------------------------------------------
        p_clk_gen : process
            begin
               while now < 40 ms loop
                  s_clk_100MHz <= '0';
                  wait for c_CLK_100MHZ_PERIOD / 2;
                  s_clk_100MHz <= '1';
                  wait for c_CLK_100MHZ_PERIOD / 2;
               end loop;
               wait;
        end process p_clk_gen;

    ----------------------------------------------------------------------
    -- Reset generation process
    ----------------------------------------------------------------------

        p_reset_gen : process
           begin
              s_rst <= '0';
              wait for 28 ns;
              s_rst <= '1';  -- Reset
              wait for 13 ns;
              s_rst <= '0';
              wait for 17 ns;
              s_rst <= '1';  -- Reset
              wait for 693 ns;
              s_rst <= '1';  -- Reset
              wait;
        end process p_reset_gen;

    ----------------------------------------------------------------------
    -- Data generation process
    ----------------------------------------------------------------------
        p_stimulus : process
            begin
               report "Stimulus process started" severity note;

               s_d  <= '0';
               wait for 14 ns;
               s_d  <= '1';
               wait for 2 ns;

               assert ((s_rst = '0') and (s_q = '1') and (s_q_bar = '0'))
               report "Test failed when RESET is '0', CLK is RISING and D is '1'" severi

               wait for 8 ns;
               s_d  <= '0';
               wait for 10 ns;
```

```
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';
        wait for 10 ns;
        s_d  <= '1';
        wait for 5 ns;

        assert ((s_rst = '1') and (s_q = '1') and (s_q_bar = '0'))
        report "Test failed when RESET is '1', CLK is FALLING and D is '1'" sever

        wait for 5 ns;
        s_d  <= '0';
        wait for 14 ns;
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';
        wait for 10 ns;
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';
        wait for 10 ns;
        s_d  <= '1';
        wait for 10 ns;
        s_d  <= '0';

        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;
```

Simulated time waveforms



VHDL code of the process `p_jk_ff_rst` :

```
    p_jk_ff_rst : process (clk)
        begin
            if rising_edge(clk) then
                if (rst = '1') then
                    s_q <= '0';
                else
                    if (j = '0' AND k = '0') then
                        s_q <= s_q;
                    elsif (j = '0' AND k = '1') then
```

```
                    s_q <= '0';
                elsif (j = '1' AND k = '0') then
                    s_q <= '1';
                elsif (j = '1' AND k = '1') then
                    s_q <= not s_q;
                end if;
            end if;
        end if;
    end process p_jk_ff_rst;

    q <= s_q;
    q_bar <= not s_q;
```

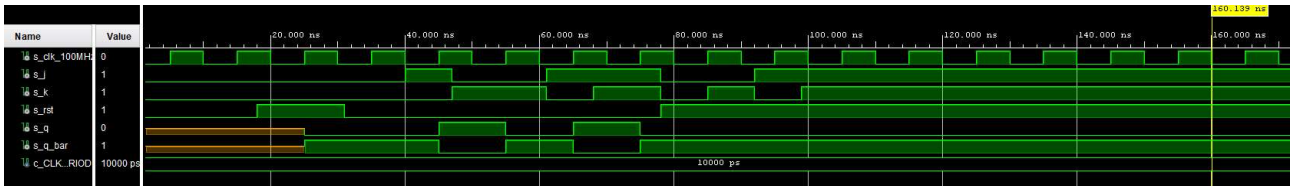
VHDL testbench file `tb_jk_ff_rst` :

```
    ----------------------------------------------------------------------
    -- Clock generation process
    ----------------------------------------------------------------------
        p_clk_gen : process
            begin
               while now < 40 ms loop
                  s_clk_100MHz <= '0';
                  wait for c_CLK_100MHZ_PERIOD / 2;
                  s_clk_100MHz <= '1';
                  wait for c_CLK_100MHZ_PERIOD / 2;
               end loop;
               wait;
        end process p_clk_gen;

    ----------------------------------------------------------------------
    -- Reset generation process
    ----------------------------------------------------------------------

        p_reset_gen : process
           begin
              s_rst <= '0';
              wait for 18 ns;
              s_rst <= '1';  -- Reset
              wait for 13 ns;
              s_rst <= '0';
              wait for 47 ns;
              s_rst <= '1';  -- Reset
              wait for 693 ns;
              s_rst <= '1';  -- Reset
              wait;
        end process p_reset_gen;

    ----------------------------------------------------------------------
    -- Data generation process
```

```
    ----------------------------------------------------------------------
        p_stimulus : process
            begin
                report "Stimulus process started" severity note;

                s_j  <= '0';
                s_k  <= '0';
                wait for 38 ns;

                assert ((s_rst = '0') and (s_j = '0') and (s_k = '0') and (s_q = '0') and
                report "Test failed when RESET is '0', CLK is RISING, J is '0' and K is '

                wait for 2 ns;
                s_j  <= '1';
                s_k  <= '0';
                wait for 6 ns;

                assert ((s_rst = '0') and (s_j = '1') and (s_k = '0') and (s_q = '1') and
                report "Test failed when RESET is '0', CLK is RISING, J is '1' and K is '

                wait for 1 ns;
                s_j  <= '0';
                s_k  <= '1';
                wait for 13 ns;

                assert ((s_rst = '0') and (s_j = '0') and (s_k = '1') and (s_q = '0') and
                report "Test failed when RESET is '0', CLK is RISING, J is '0' and K is '

                wait for 1 ns;
                s_j  <= '1';
                s_k  <= '0';
                wait for 7 ns;
                s_j  <= '1';
                s_k  <= '1';
                wait for 8 ns;

                assert ((s_rst = '0') and (s_j = '1') and (s_k = '1') and (s_q = '0') and
                report "Test failed when RESET is '0', CLK is RISING, J is '1' and K is '

                wait for 2 ns;
                s_j  <= '0';
                s_k  <= '0';
                wait for 7 ns;
                s_j  <= '0';
                s_k  <= '1';
                wait for 7 ns;
                s_j  <= '1';
                s_k  <= '0';
                wait for 7 ns;
                s_j  <= '1';
                s_k  <= '1';
```

```
        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;
```

Simulated time waveforms



VHDL code of the process `p_t_ff_rst` :

```
    p_t_ff_rst : process (clk)
        begin
            if rising_edge(clk) then
                if (rst = '1') then
                    s_q <= '0';
                elsif (t = '1') then
                    s_q <= not s_q;
                end if;
            end if;
    end process p_t_ff_rst;

    q <= s_q;
    q_bar <= not s_q;
```

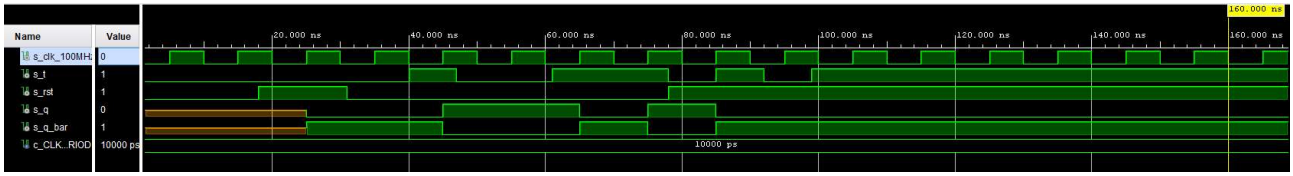
VHDL testbench file `tb_t_ff_rst` :

```
  ----------------------------------------------------------------------
  -- Clock generation process
  ----------------------------------------------------------------------
      p_clk_gen : process
          begin
             while now < 40 ms loop
                s_clk_100MHz <= '0';
                wait for c_CLK_100MHZ_PERIOD / 2;
                s_clk_100MHz <= '1';
                wait for c_CLK_100MHZ_PERIOD / 2;
             end loop;
             wait;
      end process p_clk_gen;
```

```
    ----------------------------------------------------------------------
    -- Reset generation process
    ----------------------------------------------------------------------

        p_reset_gen : process
           begin
              s_rst <= '0';
              wait for 18 ns;
              s_rst <= '1';  -- Reset
              wait for 13 ns;
              s_rst <= '0';
              wait for 47 ns;
              s_rst <= '1';  -- Reset
              wait for 693 ns;
              s_rst <= '1';  -- Reset
              wait;
        end process p_reset_gen;

    ----------------------------------------------------------------------
    -- Data generation process
    ----------------------------------------------------------------------
        p_stimulus : process
            begin
               report "Stimulus process started" severity note;

               s_t  <= '0';
               wait for 38 ns;

               assert ((s_rst = '0') and (s_t = '0') and (s_q = '0') and (s_q_bar = '1')
               report "Test failed when RESET is '0', CLK is RISING and T is '0'" severi

               wait for 2 ns;
               s_t  <= '1';
               wait for 6 ns;

               assert ((s_rst = '0') and (s_t = '1') and (s_q = '1') and (s_q_bar = '0')
               report "Test failed when RESET is '0', CLK is RISING and T is '1'" severi

               wait for 1 ns;
               s_t  <= '0';
               wait for 13 ns;

               assert ((s_rst = '0') and (s_t = '0') and (s_q = '1') and (s_q_bar = '0')
               report "Test failed when RESET is '0', CLK is RISING and T is '0'" severi

               wait for 1 ns;
               s_t  <= '1';
               wait for 5 ns;

               assert ((s_rst = '0') and (s_t = '1') and (s_q = '0') and (s_q_bar = '1')
               report "Test failed when RESET is '0', CLK is RISING and T is '1'" severi
```

```
        wait for 12 ns;
        s_t  <= '0';
        wait for 7 ns;
        s_t  <= '1';
        wait for 7 ns;
        s_t  <= '0';
        wait for 7 ns;
        s_t  <= '1';

        report "Stimulus process finished" severity note;
        wait;
    end process p_stimulus;
```

Simulated time waveforms



Exercise 4: Shift register schematic

Sketch of the shift register schematic: