电子系统设计-第二次作业

PB20051061 牟真伟

1.编写一个四位锁存器，结构如图1所示，功能如表1所示。

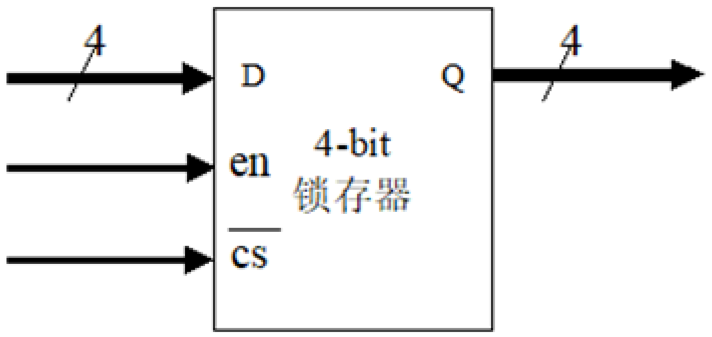
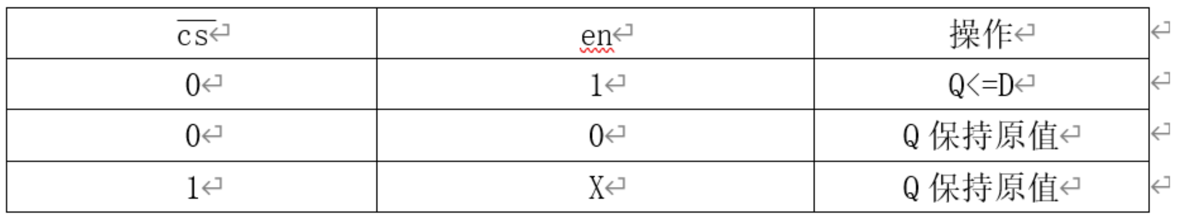


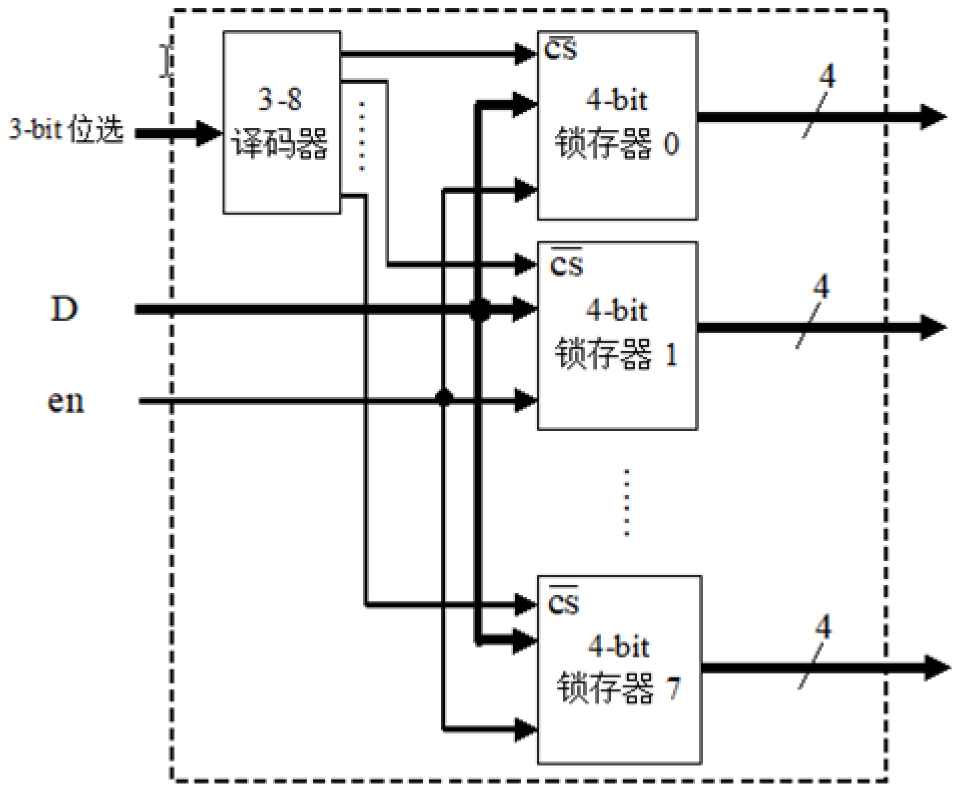
表 1 四位寄存器功能



$\overline{cs}$	en	操作
0	1	Q<=D
0	0	Q 保持原值
1	X	Q 保持原值

```vhdl
-- 四位锁存器latch @ PB20051061 牟真伟

library ieee;
use ieee.std_logic_1164.all;

entity LATCH_4_mzw is
    port(
        D :IN std_logic_vector(3 downto 0);
        Q :OUT std_logic_vector(3 downto 0);
        en :IN std_logic;
        cs :IN std_logic
    );
end LATCH_4_mzw;

architecture arch_LATCH_4 of LATCH_4_mzw is
begin
    process(cs,en,D)
    begin
        if(cs = '0' and en = '1')   then
            Q <= D;
        end if;
    end process;

end arch_LATCH_4;
```

2.以上述设计为元件，采用元件例化方式，设计八个四位锁存器，如图2所示。每个锁存器片选#CS信号需要由三位输入信号进行3-8译码获得，可以把数据输入到不同的锁存器。

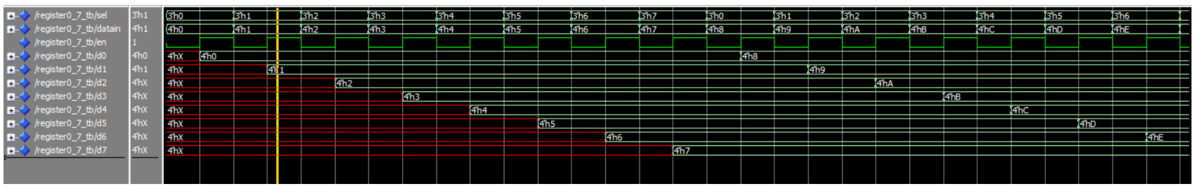


```
-- 38译码器
library ieee;
use ieee.std_logic_1164.all;

entity decoder38_0_mzw is
    port(
        A :IN std_logic_vector(2 downto 0);
        Y :OUT std_logic_vector(7 downto 0)
    );
end decoder38_0_mzw;

architecture arch_decoder38 of decoder38_0_mzw is

begin
    process(A)
    begin
        case A is
            when "000" => Y <= "11111110";
            when "001" => Y <= "11111101";
            when "010" => Y <= "11111011";
            when "011" => Y <= "11110111";
            when "100" => Y <= "11101111";
            when "101" => Y <= "11011111";
            when "110" => Y <= "10111111";
            when "111" => Y <= "01111111";
            when others => null;
        end case;
    end process;
end arch_decoder38;

-- 寄存器组

library ieee;
use ieee.std_logic_1164.all;

entity register0_7_mzw is
    port(
        sel :IN std_logic_vector(2 downto 0);
        datain :IN std_logic_vector(3 downto 0);
        en :IN std_logic;
        d0 :OUT std_logic_vector(3 downto 0);
        d1 :OUT std_logic_vector(3 downto 0);
        d2 :OUT std_logic_vector(3 downto 0);
        d3 :OUT std_logic_vector(3 downto 0);
        d4 :OUT std_logic_vector(3 downto 0);
        d5 :OUT std_logic_vector(3 downto 0);
        d6 :OUT std_logic_vector(3 downto 0);
        d7 :OUT std_logic_vector(3 downto 0)
    );
end register0_7_mzw;

architecture arch_register0_7_mzw of register0_7_mzw is
    component LATCH_4_mzw port(
        D :IN std_logic_vector(3 downto 0);
        Q :OUT std_logic_vector(3 downto 0);
        en :IN std_logic;
        cs :IN std_logic
    );
    end component;

    component decoder38_0_mzw port(
        A :IN std_logic_vector(2 downto 0);
        Y :OUT std_logic_vector(7 downto 0)
    );
    end component;

    signal cs7_0 :std_logic_vector(7 downto 0);

begin
    l0 :LATCH_4_mzw port map(datain,d0,en,cs7_0(0));
    l1 :LATCH_4_mzw port map(datain,d1,en,cs7_0(1));
    l2 :LATCH_4_mzw port map(datain,d2,en,cs7_0(2));
    l3 :LATCH_4_mzw port map(datain,d3,en,cs7_0(3));
    l4 :LATCH_4_mzw port map(datain,d4,en,cs7_0(4));
    l5 :LATCH_4_mzw port map(datain,d5,en,cs7_0(5));
    l6 :LATCH_4_mzw port map(datain,d6,en,cs7_0(6));
    l7 :LATCH_4_mzw port map(datain,d7,en,cs7_0(7));
    decoder38 :decoder38_0_mzw port map(sel,cs7_0);
end arch_register0_7_mzw;
```

3.编写题2模块的仿真测试程序（参考PPT），并根据输入信号的设置方式，说明要检测模块的哪些功能。

```
-- register0_7_tb  @ PB20051061

library ieee;
use ieee.std_logic_1164.all;

entity register0_7_tb is
end register0_7_tb;

architecture arch_register0_7_tb of register0_7_tb is
    component register0_7_mzw port(
        sel :IN std_logic_vector(2 downto 0);
        datain :IN std_logic_vector(3 downto 0);
        en :IN std_logic;
        d0 :OUT std_logic_vector(3 downto 0);
        d1 :OUT std_logic_vector(3 downto 0);
        d2 :OUT std_logic_vector(3 downto 0);
        d3 :OUT std_logic_vector(3 downto 0);
        d4 :OUT std_logic_vector(3 downto 0);
        d5 :OUT std_logic_vector(3 downto 0);
        d6 :OUT std_logic_vector(3 downto 0);
        d7 :OUT std_logic_vector(3 downto 0)
    );
    end component;
    signal  sel :std_logic_vector(2 downto 0);
    signal  datain :std_logic_vector(3 downto 0);
    signal  en :std_logic;
    signal  d0 :std_logic_vector(3 downto 0);
    signal  d1 :std_logic_vector(3 downto 0);
    signal  d2 :std_logic_vector(3 downto 0);
    signal  d3 :std_logic_vector(3 downto 0);
    signal  d4 :std_logic_vector(3 downto 0);
    signal  d5 :std_logic_vector(3 downto 0);
    signal  d6 :std_logic_vector(3 downto 0);
    signal  d7 :std_logic_vector(3 downto 0);
begin
    reg :register0_7_mzw port map(sel,datain,en,d0,d1,d2,d3,d4,d5,d6,d7);

    process
    begin
        en <= '0';
        wait for 10 ns;
        en <= '1';
        wait for 10 ns;
    end process;

    process
    begin
        datain <= "0000";
        wait for 20 ns;
        datain <= "0001";
        wait for 20 ns;
        datain <= "0010";
        wait for 20 ns;
        datain <= "0011";
        wait for 20 ns;
        datain <= "0100";
        wait for 20 ns;
        datain <= "0101";
        wait for 20 ns;
        datain <= "0110";
        wait for 20 ns;
        datain <= "0111";
        wait for 20 ns;
        datain <= "1000";
        wait for 20 ns;
        datain <= "1001";
        wait for 20 ns;
        datain <= "1010";
        wait for 20 ns;
        datain <= "1011";
        wait for 20 ns;
        datain <= "1100";
        wait for 20 ns;
        datain <= "1101";
        wait for 20 ns;
        datain <= "1110";
        wait for 20 ns;
        datain <= "1111";
        wait for 20 ns;
    end process;

    process
    begin
        sel <= "000";
        wait for 20 ns;
        sel <= "001";
        wait for 20 ns;
        sel <= "010";
        wait for 20 ns;
        sel <= "011";
        wait for 20 ns;
        sel <= "100";
        wait for 20 ns;
        sel <= "101";
        wait for 20 ns;
        sel <= "110";
        wait for 20 ns;
        sel <= "111";
        wait for 20 ns;
    end process;
end arch_register0_7_tb;
```

仿真图为：



仿真说明：以20ns为一个小循环，依次在8个20ns中sel信号选择从寄存器0到寄存器7，datain从0000到0111，再8个20ns中sel信号选择从寄存器0到寄存器7，datain从1000到0111，每个小循环中前10ns en信号无效，后10ns en信号有效，在每个小循环中间改变寄存器的值。此方法可以检查模块的选择功能，使能功能，数据输入功能，锁存功能。