Seven-Segment Display

高政 11912015

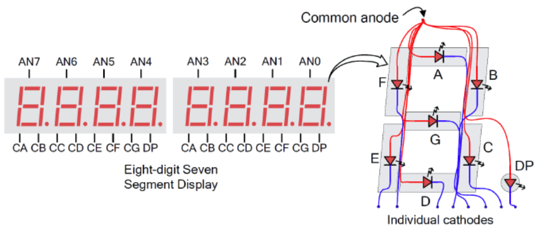
一、简述

a) 实验目的

通过本次实验了解七段数字显示器的工作原理，继续熟悉模块化设计的 VHDL 设计理念。

b) 七段显示码

本次实验使用的是共阳极的七段数字显示器，其电路结构如下



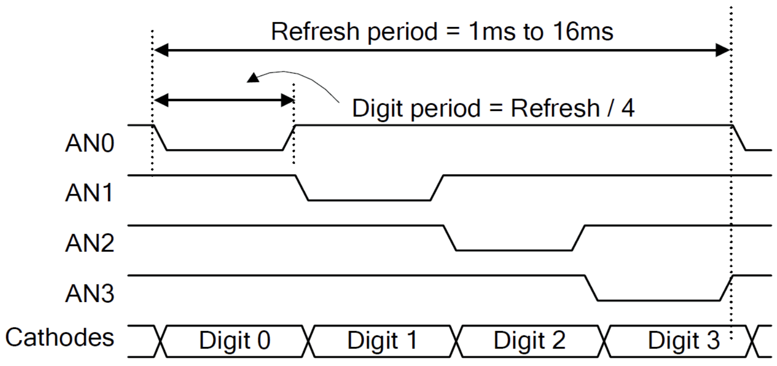
同一时刻拉高二极管阳极并将相应的阴极拉低即可显示数字，应用 4-8 解码器可实现对阴极型号的控制，解码格式如下

x	a	b	c	d	e	f	g
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0
A	0	0	0	1	0	0	0
B	1	1	0	0	0	0	0
C	0	1	1	0	0	0	1
D	1	0	0	0	0	1	0
E	0	1	1	0	0	0	0
F	0	1	1	1	0	0	0

1 = off

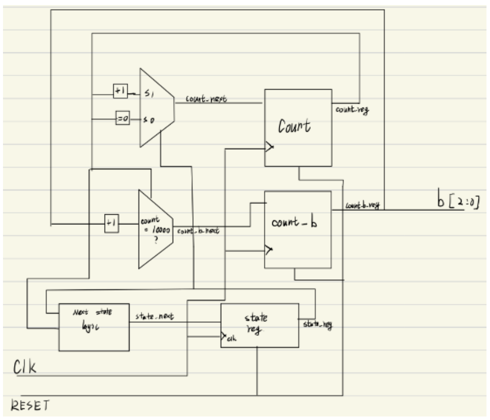
0 = on

为节省引脚数量，本实验板上的八个数字显示器共用一组阴极信号，为实现八个显示器显示不同的数字，采取同一时刻仅拉高一个显示器的阳极，利用人眼的视觉残留效应，以至少每秒 45Hz 的频率循环拉高各个显示器达到同时显示不同数字的效果，阳极信号循环示例如下



二、实验准备

计数器部分的 RTL 设计原理如下



三、VHDL 代码与 testbench 代码

a)　计数器模块

VHDL 代码：

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

entity counter is
    Port ( CLK : in STD_LOGIC;
           RESET : in STD_LOGIC;
           b : out STD_LOGIC_VECTOR (2 downto 0));
end counter;

architecture Behavioral of counter is
type state_type is(s0, s1);
signal state, state_next :state_type;
signal count, count_next : std_logic_vector(13 downto 0);
signal count_b, count_b_next: std_logic_vector(2 downto 0);
constant upcount :integer:=10000;
begin
clk_pro:process(CLK, RESET)is
begin
    if RESET = '1' then
        state <= s0;
        count <= (others => '0');
        count_b <= "000";
    elsif(CLK'event and CLK = '1') then
        state <= state_next;
        count <= count_next;
        count_b <= count_b_next;
    end if;
end process;
```

```
process (state, state_next, count, count_next) is
begin
        state_next <= s0;
        count_next <= count;
        case state is
                when s0 =>
                        count_next <= (others => '0');
                        count_b_next <= count_b + 1;
                        state_next <= s1;
                when s1 =>
                        if count = upcount - 2 then
                                count_next <= (others => '0');
                                state_next <= s0;
                        else
                                count_next <= count + 1;
                                state_next <= s1;
                        end if;
        end case;
end process;
b <= count_b;
end Behavioral;
```

testbench 代码：

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

entity tb_count is
end tb_count;

architecture Behavioral of tb_count is
        component counter
        port(
                CLK : in STD_LOGIC;
                RESET : in STD_LOGIC;
                b : out STD_LOGIC_VECTOR (2 downto 0)
        );
        end component;

        signal CLK : std_logic:='0';
        signal RESET : std_logic:='0';

        signal b : std_logic_vector(2 downto 0):="000";

        constant per : time:= 100 ns;
```

```vhdl
begin
uut:counter
port map(
        CLK => CLK,
        RESET => RESET,
        b => b
);

clk_pro:process
begin
        CLK <= '0';
        wait for per/2;
        CLK <= '1';
        wait for per/2;
end process;

reset_pro:process
begin
        RESET <= '1';
        for i in 1 to 2 loop
                wait until CLK = '1';
        end loop;
        RESET <= '0';
        wait;
end process;
end Behavioral;
```

b) 阳极信号产生模块

VHDL 代码：

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

entity anode_driver is
        Port ( b : in std_logic_vector(2 downto 0);
                      AN0 : out STD_LOGIC;
                      AN1 : out STD_LOGIC;
                      AN2 : out STD_LOGIC;
                      AN3 : out STD_LOGIC;
                      AN4 : out STD_LOGIC;
                      AN5 : out STD_LOGIC;
                      AN6 : out STD_LOGIC;
                      AN7 : out STD_LOGIC);
end anode_driver;

architecture Behavioral of anode_driver is
```

```
begin
process(b) is
begin
	AN0 <= '1';
	AN1 <= '1';
	AN2 <= '1';
	AN3 <= '1';
	AN4 <= '1';
	AN5 <= '1';
	AN6 <= '1';
	AN7 <= '1';
	case b is
	when "000" =>
		AN0 <= '0';
	when "001" =>
		AN1 <= '0';
	when "010" =>
		AN2 <= '0';
	when "011" =>
		AN3 <= '0';
	when "100" =>
		AN4 <= '0';
	when "101" =>
		AN5 <= '0';
	when "110" =>
		AN6 <= '0';
	when "111" =>
		AN7 <= '0';
	when others =>
	end case;
end process;
end Behavioral;
```

testbench 代码：

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_signed.all;
use IEEE.std_logic_unsigned.all;

entity tb_anode_driver is
end tb_anode_driver;

architecture Behavioral of tb_anode_driver is
	component anode_driver is
	Port ( b : in std_logic_vector(2 downto 0);
			AN0 : out STD_LOGIC;
			AN1 : out STD_LOGIC;
			AN2 : out STD_LOGIC;
			AN3 : out STD_LOGIC;
			AN4 : out STD_LOGIC;
			AN5 : out STD_LOGIC;
			AN6 : out STD_LOGIC;
			AN7 : out STD_LOGIC);
	end component;

	signal b : std_logic_vector(2 downto 0):="000";

	signal AN0, AN1, AN2, AN3, AN4, AN5, AN6, AN7: std_logic:='1';

	constant per : time:= 100 ns;

begin
uut: anode_driver
port map(
	b => b,
	AN0 => AN0,
	AN1 => AN1,
	AN2 => AN2,
	AN3 => AN3,
	AN4 => AN4,
	AN5 => AN5,
	AN6 => AN6,
	AN7 => AN7
);

b_pro : process
begin
	b <= b + 1;
	wait for per;
end process;
end Behavioral;
```

c)　输入选择模块

VHDL 代码：

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

entity mux is
```

```
        Port ( char1 : in STD_LOGIC_VECTOR (3 downto 0);
                      char2 : in STD_LOGIC_VECTOR (3 downto 0);
                      char3 : in STD_LOGIC_VECTOR (3 downto 0);
                      char4 : in STD_LOGIC_VECTOR (3 downto 0);
                      b : in STD_LOGIC_VECTOR (2 downto 0);
                      x : out STD_LOGIC_VECTOR (3 downto 0));
end mux;

architecture Behavioral of mux is
begin
mux_pro:process(char1, char2, char3, char4, b)is
begin
        x <= "0000";
        case b is
        when "000" =>
                x <= char1;
        when "001" =>
                x <= char2;
        when "010" =>
                x <= char3;
        when "011" =>
                x <= char4;
        when "100" =>
                x <= char1 + char2;
        when "101" =>
                x <= char1 - char2;
        when "110" =>
                x<= char3 + char4;
        when "111" =>
                x <= char3 - char4;
        when others =>
        end case;
end process;
end Behavioral;
```

testbench 代码：

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_signed.all;
use IEEE.std_logic_unsigned.all;

entity tb_mux is
end tb_mux;

architecture Behavioral of tb_mux is
```

```
    component mux is
    Port (
        char1 : in STD_LOGIC_VECTOR (3 downto 0);
        char2 : in STD_LOGIC_VECTOR (3 downto 0);
        char3 : in STD_LOGIC_VECTOR (3 downto 0);
        char4 : in STD_LOGIC_VECTOR (3 downto 0);
        b : in STD_LOGIC_VECTOR (2 downto 0);
        x : out STD_LOGIC_VECTOR (3 downto 0)
    );
    end component;

    signal char1: std_logic_vector(3 downto 0):="0001";
    signal char2: std_logic_vector(3 downto 0):="0010";
    signal char3: std_logic_vector(3 downto 0):="0011";
    signal char4: std_logic_vector(3 downto 0):="0100";
    signal b: std_logic_vector(2 downto 0):="000";

    signal x: std_logic_vector(3 downto 0):="0000";

    constant per : time:= 100 ns;

begin
uut:mux
port map(
    char1 => char1,
    char2 => char2,
    char3 => char3,
    char4 => char4,
    b => b,
    x => x
);

b_pro : process
begin
    b <= b + 1;
    wait for per;
end process;

end Behavioral;
```

d)　阴极信号产生模块

VHDL 代码：

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_signed.all;
use IEEE.std_logic_unsigned.all;

entity hex7seg is
        Port ( x : in STD_LOGIC_VECTOR (3 downto 0);
                      LED : out STD_LOGIC_VECTOR (7 downto 0));
end hex7seg;

architecture Behavioral of hex7seg is
begin
process(x) is
begin
        LED <= "00000000";
        case x is
        when "0000" =>
                LED <= "00000011";
        when "0001" =>
                LED <= "10011111";
        when "0010" =>
                LED <= "00100101";
        when "0011" =>
                LED <= "00001101";
        when "0100" =>
                LED <= "10011001";
        when "0101" =>
                LED <= "01001001";
        when "0110" =>
                LED <= "01000001";
        when "0111" =>
                LED <= "00011111";
        when "1000" =>
                LED <= "00000001";
        when "1001" =>
                LED <= "00001001";
        when "1010" =>
                LED <= "00010001";
        when "1011" =>
                LED <= "11000001";
        when "1100" =>
                LED <= "01100011";
        when "1101" =>
                LED <= "10000101";
        when "1110" =>
                LED <= "01100001";
        when "1111" =>
```

```
                LED <= "01110001";
        when others =>
        end case;
end process;
end Behavioral;
```

testbench 代码：

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_signed.all;
use IEEE.std_logic_unsigned.all;

entity tb_hex7seg is
end tb_hex7seg;

architecture Behavioral of tb_hex7seg is
        component hex7seg
        port(
                x : in std_logic_vector (3 downto 0);
                LED : out std_logic_vector (7 downto 0)
                );
        end component;

        signal x : std_logic_vector(3 downto 0):="0000";

        signal LED : std_logic_vector(7 downto 0):="00000000";

        constant per : time:= 100 ns;

begin
uut: hex7seg
port map(
        x => x,
        LED => LED
);

x_pro:process
begin
        x <= x + 1;
        wait for per;
end process;
end Behavioral;
```

e)　顶层设计

VHDL 代码：

```
library IEEE;
```

```
use IEEE.STD_LOGIC_1164.ALL;

entity seven_segment_display is
    Port ( CLK : in STD_LOGIC;
           RESET : in STD_LOGIC;
           char1 ,char2 ,char3 ,char4 : in STD_LOGIC_VECTOR (3 downto 0);
           LED : out STD_LOGIC_VECTOR (7 downto 0);
           AN0 ,AN1 ,AN2 ,AN3 ,AN4 ,AN5 ,AN6 ,AN7 : out STD_LOGIC);
end seven_segment_display;

architecture Behavioral of seven_segment_display is
    component  mux is
    Port (
        char1 ,char2 ,char3 ,char4 : in STD_LOGIC_VECTOR (3 downto 0);
        b : in STD_LOGIC_VECTOR (2 downto 0);
        x : out STD_LOGIC_VECTOR (3 downto 0)
    );
    end component;

    component hex7seg
    port(
        x : in STD_LOGIC_VECTOR (3 downto 0);
        LED : out STD_LOGIC_VECTOR (7 downto 0)
    );
    end component;

    component counter is
    Port (
        CLK : in STD_LOGIC;
        RESET : in STD_LOGIC;
        b : out STD_LOGIC_VECTOR (2 downto 0)
    );
    end component;

    component anode_driver is
    Port (
        b : in std_logic_vector(2 downto 0);
        AN0 ,AN1 ,AN2 ,AN3 ,AN4 ,AN5 ,AN6 ,AN7 : out STD_LOGIC
    );
    end component;

    signal b:std_logic_vector(2 downto 0);
    signal x:std_logic_vector(3 downto 0);
```

```
begin
mux_pro:mux
port map(
        char1 => char1,
        char2 => char2,
        char3 => char3,
        char4 => char4,
        b => b,
        x => x
);

hex7seg_pro:hex7seg
port map(
        x => x,
        LED => LED
);

counter_pro: counter
port map(
        CLK => CLK,
        RESET => RESET,
        b => b
);

anode_driver_pro: anode_driver
port map(
        b => b,
        AN0 => AN0,
        AN1 => AN1,
        AN2 => AN2,
        AN3 => AN3,
        AN4 => AN4,
        AN5 => AN5,
        AN6 => AN6,
        AN7 => AN7
);
end Behavioral;
```

testbench 代码：

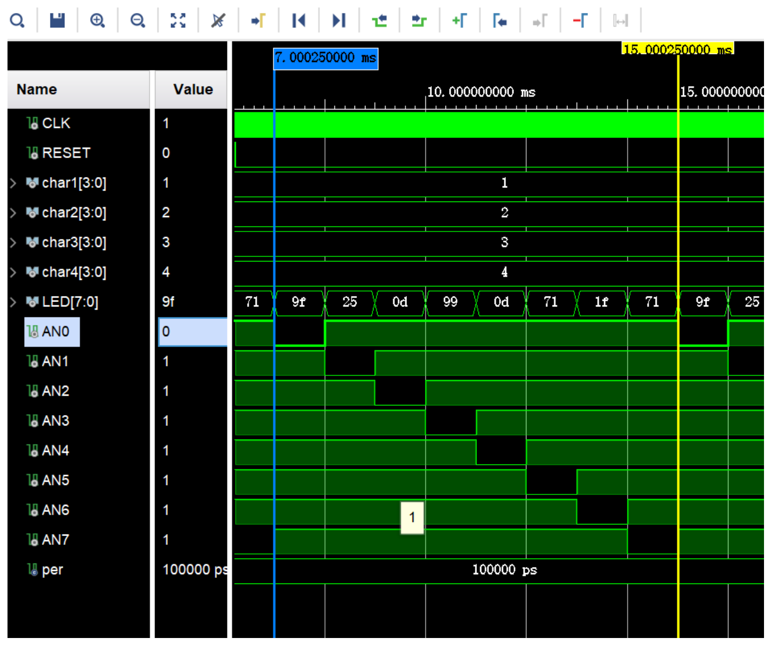
```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.std_logic_signed.all;
use IEEE.std_logic_unsigned.all;

entity tb_seven_segment_display is
```

```
end tb_seven_segment_display;

architecture Behavioral of tb_seven_segment_display is
	component seven_segment_display is
	Port (
		CLK : in STD_LOGIC;
		RESET : in STD_LOGIC;
		char1 ,char2 ,char3 ,char4 : in STD_LOGIC_VECTOR (3 downto 0);
		LED : out STD_LOGIC_VECTOR (7 downto 0);
		AN0 ,AN1 ,AN2 ,AN3 ,AN4 ,AN5 ,AN6 ,AN7 : out STD_LOGIC
	);
	end component;

	signal CLK, RESET: std_logic:='0';
	signal char1: std_logic_vector(3 downto 0):="0001";
	signal char2: std_logic_vector(3 downto 0):="0010";
	signal char3: std_logic_vector(3 downto 0):="0011";
	signal char4: std_logic_vector(3 downto 0):="0100";

	signal LED: std_logic_vector(7 downto 0):="00000000";
	signal AN0, AN1, AN2, AN3, AN4, AN5, AN6, AN7: std_logic:='1';

	constant per : time:= 100 ns;

begin
uut: seven_segment_display
port map(
	CLK => CLK,
	RESET => RESET,
	char1 => char1,
	char2 => char2,
	char3 => char3,
	char4 => char4,
	LED => LED,
	AN0 => AN0,
	AN1 => AN1,
	AN2 => AN2,
	AN3 => AN3,
	AN4 => AN4,
	AN5 => AN5,
	AN6 => AN6,
	AN7 => AN7
);
```

```
clk_pro:process
begin
        CLK <= '0';
        wait for per/2;
        CLK <= '1';
        wait for per/2;
end process;

reset_pro:process
begin
        RESET <= '1';
        for i in 1 to 2 loop
                wait until CLK = '1';
        end loop;
        RESET <= '0';
        wait;
end process;
end Behavioral;
```

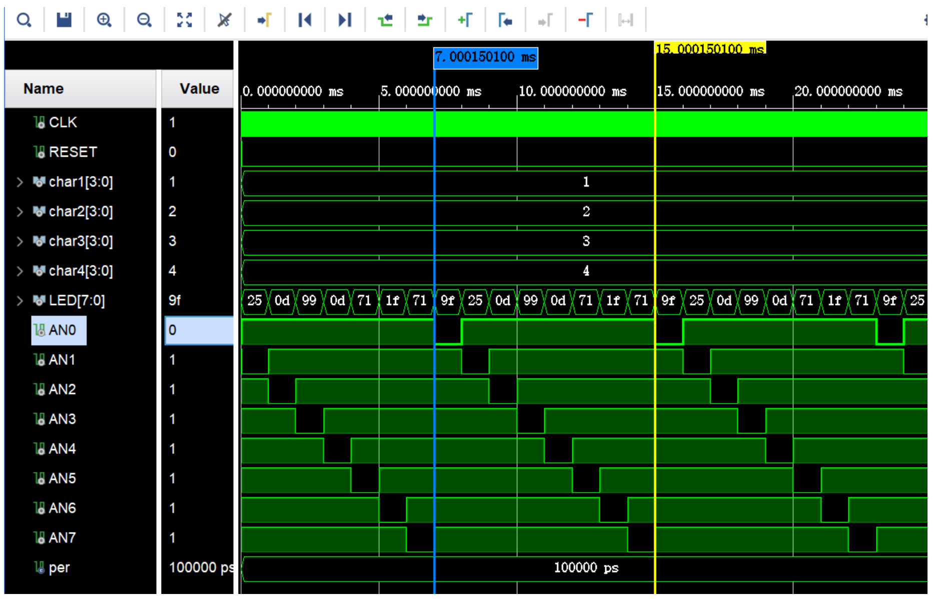
四、实验结果与分析

a)　行为仿真

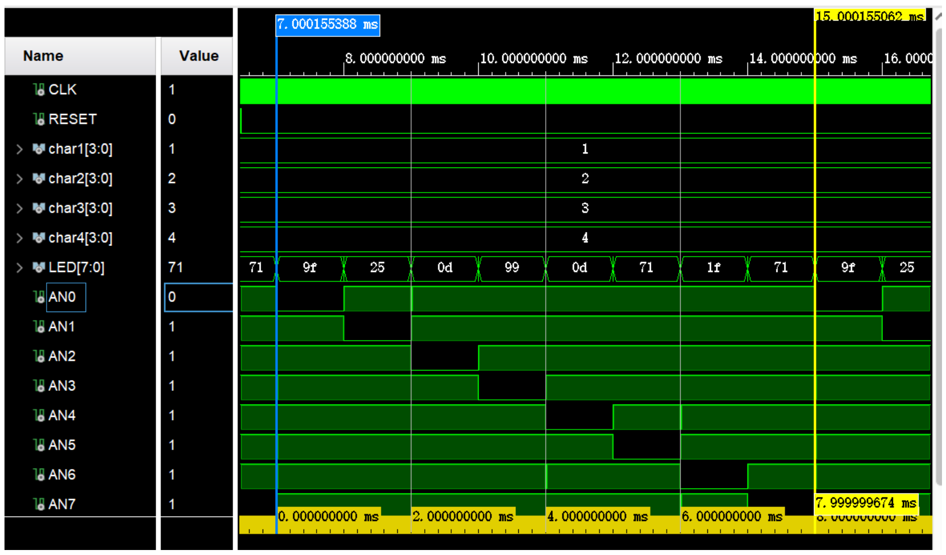


由图可知，程序功能正常。

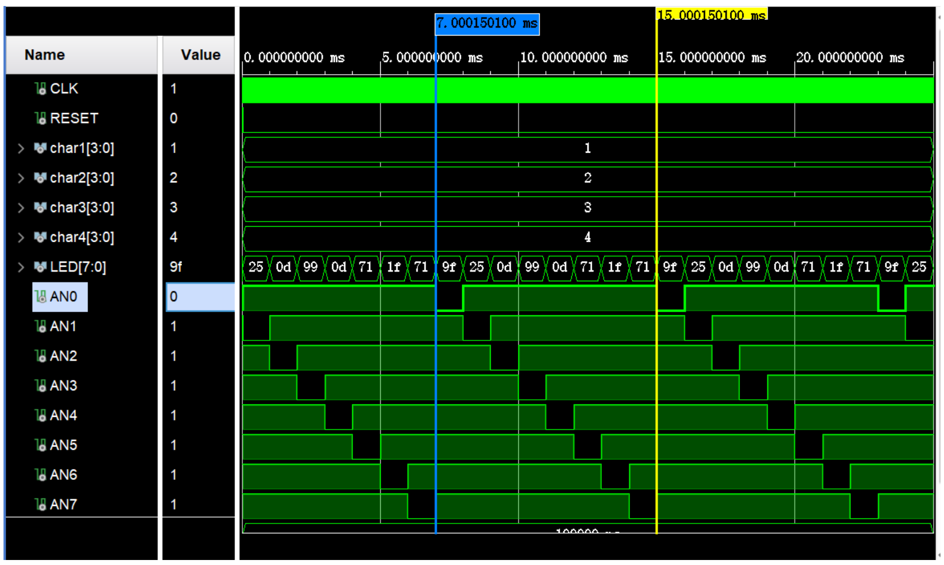
b)　综合后功能仿真


7.000150100 ms
15.000150100 ms
Name
Value
0.000000000 ms
5.000000000 ms
10.000000000 ms
15.000000000 ms
20.000000000 ms
CLK 1
RESET 0
char1[3:0] 1
char2[3:0] 2
char3[3:0] 3
char4[3:0] 4
LED[7:0] 9f
AN0 0
AN1 1
AN2 1
AN3 1
AN4 1
AN5 1
AN6 1
AN7 1
per 100000 ps
25 0d 99 0d 71 1f 71 9f 25 0d 99 0d 71 1f 71 9f 25 0d 99 0d 71 1f 71 9f 25
100000 ps

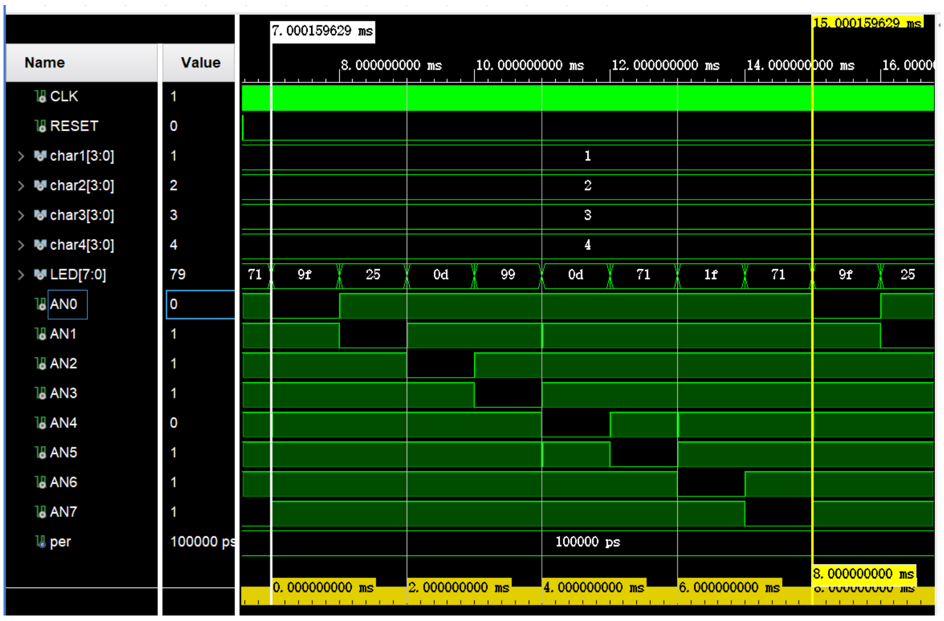
c)　综合后时序仿真


7.000155388 ms
15.000155062 ms
Name
Value
8.000000000 ms
10.000000000 ms
12.000000000 ms
14.000000000 ms
16.0000
CLK 1
RESET 0
char1[3:0] 1
char2[3:0] 2
char3[3:0] 3
char4[3:0] 4
LED[7:0] 71
AN0 0
AN1 1
AN2 1
AN3 1
AN4 1
AN5 1
AN6 1
AN7 1
71 9f 25 0d 99 0d 71 1f 71 9f 25
0.000000000 ms
2.000000000 ms
4.000000000 ms
6.000000000 ms
7.999999674 ms
8.000000000 ms

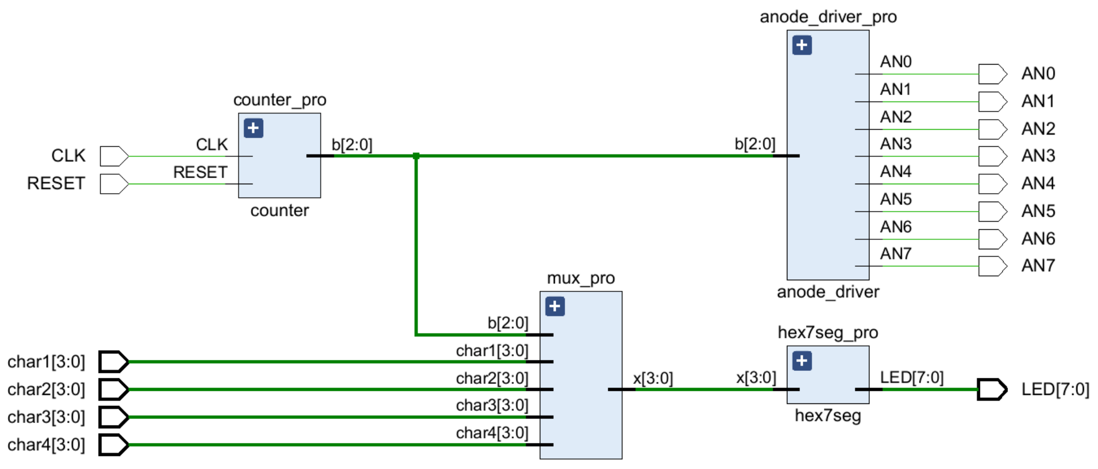
d)　布线后功能仿真



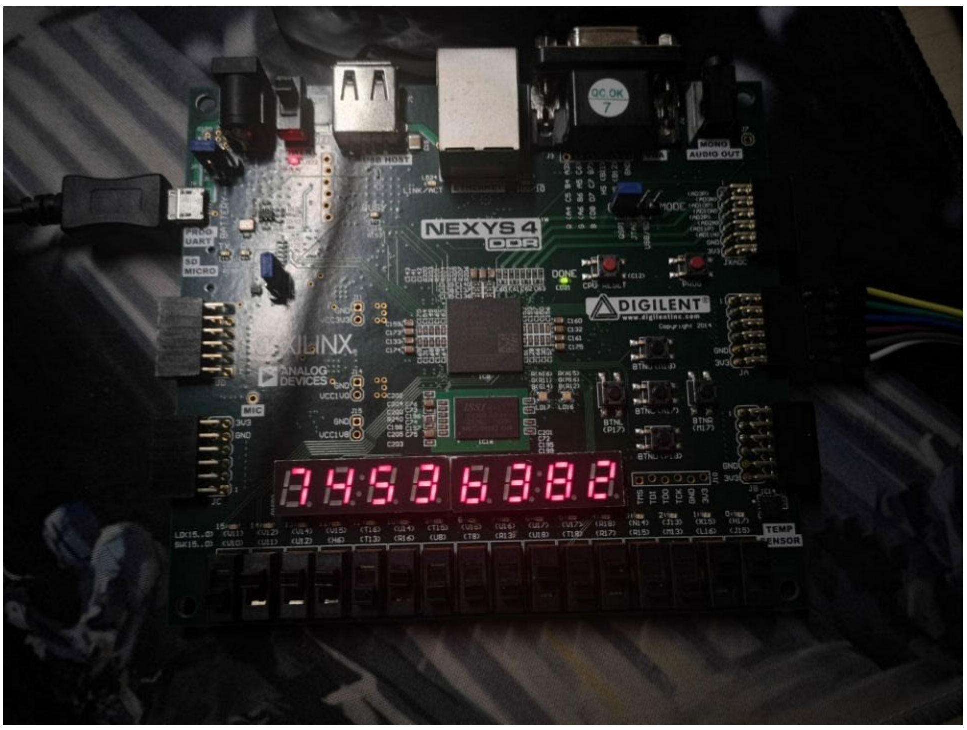
e)　布线后时序仿真



f)　RTL 原理图



g)　上板实验结果



实验板实际效果如上，从左至右八个显示器分别显示：a, b, c, d, a + b, a - b, c + d, c – d。下方 16 个开关依次对应 a, b, c, d。

五、实验结论

实验结果符合预期，通过本次实验我站我了七段数字显示器的使用方式，对时序电路的设计更为熟练，对模块化设计的理念也有了更为深入的理解。