Seven-Segment Display

高政 11912015

1. 简述
   1. 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。

* 1. 图形用户界面, 图示, 应用程序

     中度可信度描述已自动生成七段显示码

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

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

1. 实验准备

计数器部分的RTL设计原理如下

图示, 示意图

描述已自动生成

1. VHDL代码与testbench代码
   1. 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;

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;

* 1. 阳极信号产生模块

**VHDL代码：**

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;

* 1. 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;

* 1. 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;

* 1. 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;

1. 实验结果与分析
   1. 行为仿真

由图可知图形用户界面

描述已自动生成，程序功能正常。

* 1. 综合后功能仿真

图形用户界面, 树状图

描述已自动生成日程表

中度可信度描述已自动生成

* 1. 综合后时序仿真
  2. 图表

     描述已自动生成布线后功能仿真
  3. 图片包含 日程表

     描述已自动生成布线后时序仿真
  4. RTL原理图

图表, 瀑布图

描述已自动生成

* 1. 图片包含 室内, 桌子, 男人, 大

     描述已自动生成上板实验结果

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。

1. 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。