实验名称：FPGA 步进电机脉冲分配器　姓名：____严旭铧_____学号：___3220101731_____

专业：电气工程及其自动化
姓名：严旭铧
学号：3220101731
日期：2024.6.4
地点：紫金港东三 406

浙江大学实验报告

课程名称：__电路与电子技术 2_实验___指导老师：__ 张伟___ _成绩：_____

实验名称：_____FPGA 步进电机脉冲分配器_____实验类型：_EDA 实验_____同组学生姓名：褚玘铖

实验 15　FPGA 步进电机脉冲分配器设计

一、 实验目的

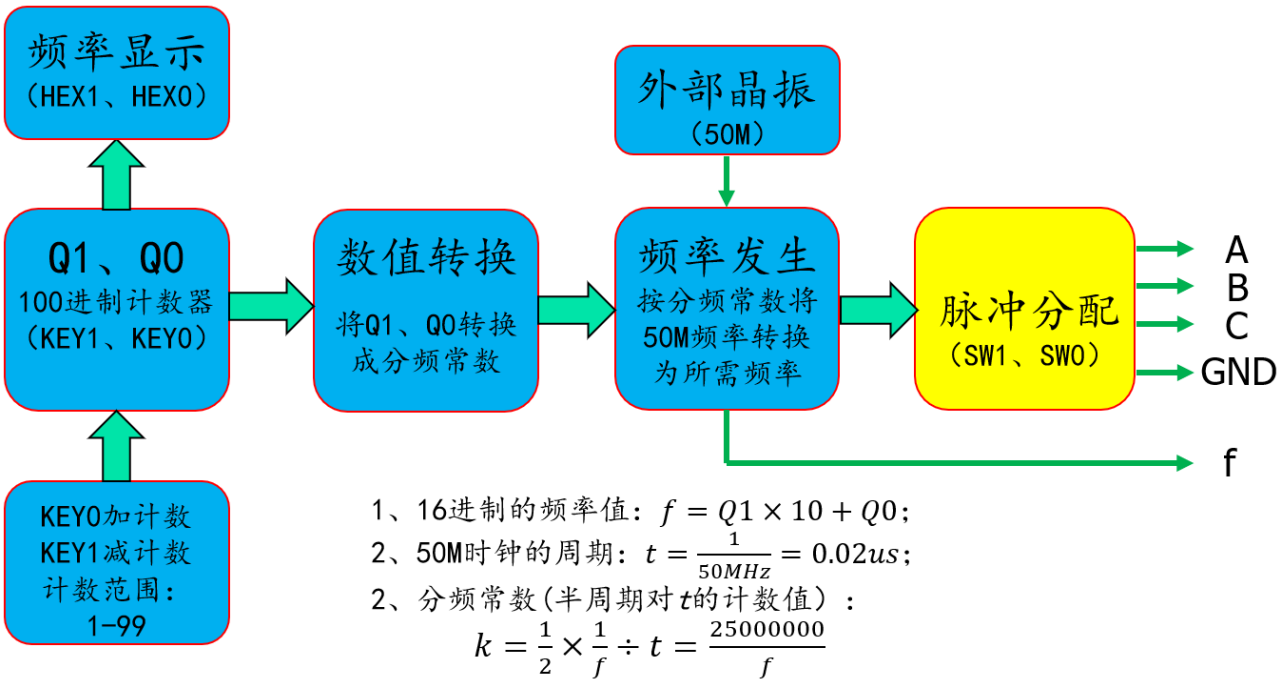
1. 进一步熟悉 FPGA 开发板和 Quartus 软件的联合使用
2. 进一步熟悉 VHDL 语言

二、 实验要求

1. 采用 DE10-Lite FPGA 学习板，用两位数码管（HEX1、 HEX0）显示脉冲频率，频率范围：1Hz -- 99Hz；频率调节按钮（KEY1， KEY0）；
2. 工作模式为三相六拍；
3. 滑动开关 SW0 可控制 运行/停止 状态；滑动开关 SW1 可控制 正转/反转 状态；
4. A、 B、 C 三相脉冲及频率 f 采用学习板上 GPIO（JP1）输出
   A - GPIO_[1]; B - GPIO_[5]; C - GPIO_[9]; f - GPIO_[11]; GND - PIN12 。

三、 实验内容

1. 程序结构说明



1、16进制的频率值：$f = Q1 \times 10 + Q0$；

2、50M时钟的周期：$t = \frac{1}{50MHz} = 0.02us$；

2、分频常数（半周期对t的计数值）：

$$k = \frac{1}{2} \times \frac{1}{f} \div t = \frac{25000000}{f}$$

2. 分模块代码分析

(1) 库引用及主体定义

```
LIBRARY IEEE;
```

```
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
USE IEEE.STD_LOGIC_ARITH.ALL;

entity motor is
    port(
        clk_50Mhz : in std_logic;
        key0_up, key1_down : in std_logic;
        sw0_clr, sw1_x : in std_logic;
        f,a,b,c : out std_logic;
        hex0, hex1 : out STD_LOGIC_VECTOR(6 downto 0));
end motor;
```

(2) 结构体及内部信号定义

```
architecture Behavioral of motor is
--internal signal
signal  q1   : STD_LOGIC_VECTOR(3 downto 0) :="0000";
signal  q0   : STD_LOGIC_VECTOR(3 downto 0) :="0001";     --100 进制可逆计数器
signal  clk_10hz :     std_logic; --0.1s signal
signal  q2,q3,q4,q5 :     STD_LOGIC_VECTOR(7 downto 0) := "00000000";     --100 进制转 16 进制中间变量
signal  k : integer range 0 to 25000000;     --分频系数
signal  freq:std_logic; --频率信号
signal  qa   : std_logic := '1';
signal  qb,qc: std_logic := '0';     --脉冲信号 ABC，初值设定为“100”

constant      m : integer := 2500000;     --0.1 秒信号分频常数
```

(3) 0.1 秒定时信号程序

```
begin
--0.1 秒定时信号程序
process(clk_50Mhz)
variable cout : integer := 0;
begin
        if rising_edge(clk_50Mhz) then
            cout := cout+1;
             if cout <= m then clk_10hz <= '0';
                elsif cout < m*2 then clk_10hz <= '1';
                  else cout := 0;
             end if;
         end if;
         end process;
```

这里，课件所用的是 0.5s 的定时，但是这样的话，键盘 KEY0 和 KEY1 用起来实在是不方便，响应速度太慢。于是把分频常数改小，这样的话得到的是 0.1s 的定时，按键的时候体验更好了。

(4) 100 进制可逆计数器

```
process(clk_10hz)
begin
    if rising_edge(clk_10hz) then
        if key0_up = '0' then
            if(q1="1001" and q0="1001") then
            q1 <= "0000"; q0 <= "0001"; --99 返回 01
            elsif (q0="1001") then
            q1 <= q1+1; q0 <= "0000";    --个位为 9，个位清零，十位+1
            else     q0 <= q0+1;       --个位+1
            end if;
        elsif key1_down = '0' then   --减计数
            if(q1="0000" and q0="0001") then
            q1 <= "1001"; q0 <= "1001"; --十位为 0，个位为 1，置 99
            elsif (q0="0000") then
            q1 <= q1-1; q0 <="1001";     --个位为 0，十位-1，个位置 9
            else q0 <= q0-1;     --个位-1
            end if;
        end if;
    end if;
end process;
```

这里和之前的加法器、数字钟如出一辙，就是在特殊点做置数，这样可以实现特定进制的设定。

(5) 100 进制转换成 16 进制

```
q2(3 downto 0) <= q0;    --q2=q0
q3(4 downto 1) <= q1;    --q3=q1×2
q4(6 downto 3) <= q1;    --q4=q1×8
q5 <= q2+q3+q4;           --q5=q1×(2+8)+q0
k <= 25000000/conv_integer(q5); --分频系数
```

转换成 16 进制是为了方便点亮数码管。

这里由于二进制数的性质，左移一位相当于×2，右移一位相当于/2。最终是要实现 q5×10+q0，这里的×10 就用（×2+×8）来实现，也即分别左移 1 位和 3 位。

(6) 产生设定频率的方波信号

```
process(clk_50Mhz)
    variable cout :   integer :=0;
        begin
            if rising_edge(clk_50Mhz) then
                cout   := cout+1;
                if cout <=k then freq <='0';
                    elsif cout < k*2 then freq <= '1';
```

```
                    else cout := 0;
                    end if;
                end if;
    end process;
```

这里和上面类似，但是将 m 换成了分频系数 k，以产生指定频率的方波。

(7) 脉冲分配信号产生和输出程序

```
    process(freq)
    begin
        if rising_edge(freq) then     --在设定频率信号的上升沿进行换相
            qa <= not((sw1_x and qb) or ((not sw1_x) and qc) );
            qb <= not((sw1_x and qc) or ((not sw1_x) and qa) );
            qc <= not((sw1_x and qa) or ((not sw1_x) and qb) );
        end if;
    end process;
--信号输出
    f <= freq;
    a <= qa and sw0_clr;
    b <= qb and sw0_clr;
    c <= qc and sw0_clr;
```

$$qa = \overline{SW1\_x \cdot qb} + \overline{SW1\_x \cdot qc}$$
$$qb = \overline{SW1\_x \cdot qc} + \overline{SW1\_x \cdot qa}$$
$$qc = \overline{SW1\_x \cdot qa} + \overline{SW1\_x \cdot qb}$$

这里实现了脉冲分配器的功能，按照 f, SW0, SW1 产生三相六拍步进电机控制信号。这里，SW1 置 1 为正转，否则反转。

(8) 100 进制个位数七段显示码译码程序

```
    process(q0) --个位译码
    begin
            case q0 is
            when   "0000" => hex0 <= "1000000";   --0
            when   "0001" => hex0 <= "1111001";   --1
            when   "0010" => hex0 <= "0100100";   --2
            when   "0011" => hex0 <= "0110000";   --3
            when   "0100" => hex0 <= "0011001";   --4
            when   "0101" => hex0 <= "0010010";   --5
            when   "0110" => hex0 <= "0000010";   --6
            when   "0111" => hex0 <= "1111000";   --7
            when   "1000" => hex0 <= "0000000";   --8
            when   "1001" => hex0 <= "0010000";   --9
            when others   => hex0 <= "1111111";   --灭
        end case;
    end process;

    process(q1) --十位译码
```

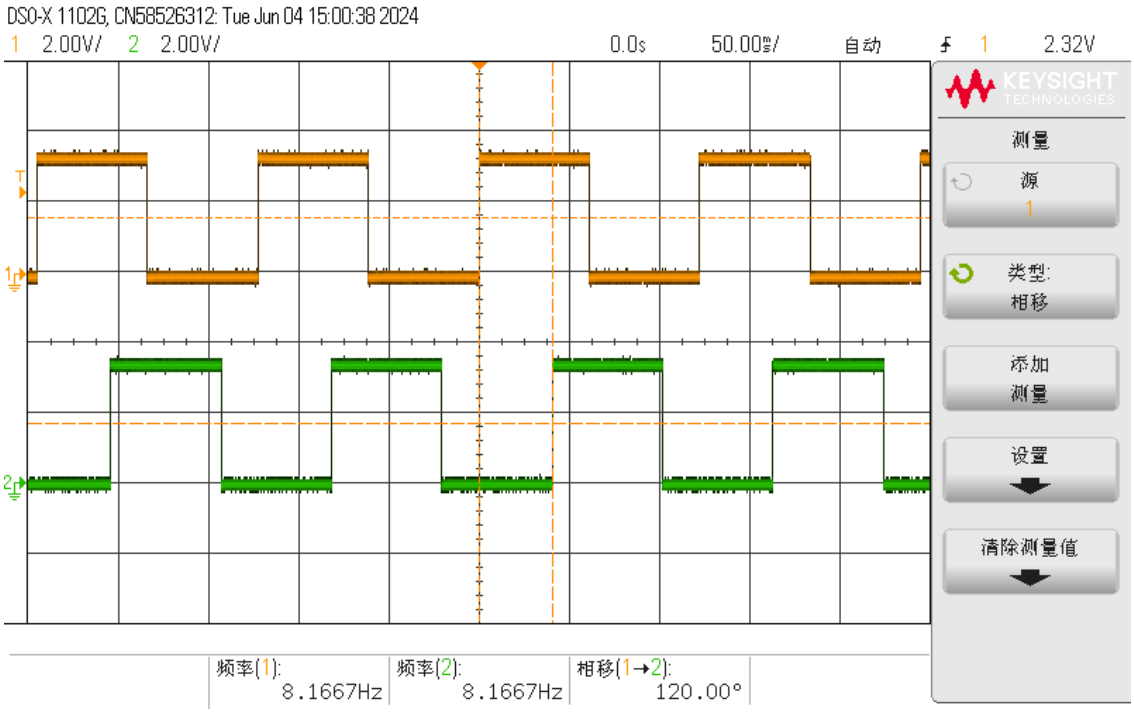
```
begin
        case q1 is
       when   "0000" => hex1 <= "1000000";   --0
        when   "0001" => hex1 <= "1111001";   --1
        when   "0010" => hex1 <= "0100100";   --2
        when   "0011" => hex1 <= "0110000";   --3
        when   "0100" => hex1 <= "0011001";   --4
        when   "0101" => hex1 <= "0010010";   --5
        when   "0110" => hex1 <= "0000010";   --6
        when   "0111" => hex1 <= "1111000";   --7
        when   "1000" => hex1 <= "0000000";   --8
        when   "1001" => hex1 <= "0010000";   --9
        when others   => hex1 <= "1111111";   --灭

        end case;
end process;
```

这里直接借用先前的共阳极数码管的显示程序，让频率值的个位和十位分别显示在 hex0 和 hex1 两个数码管上。

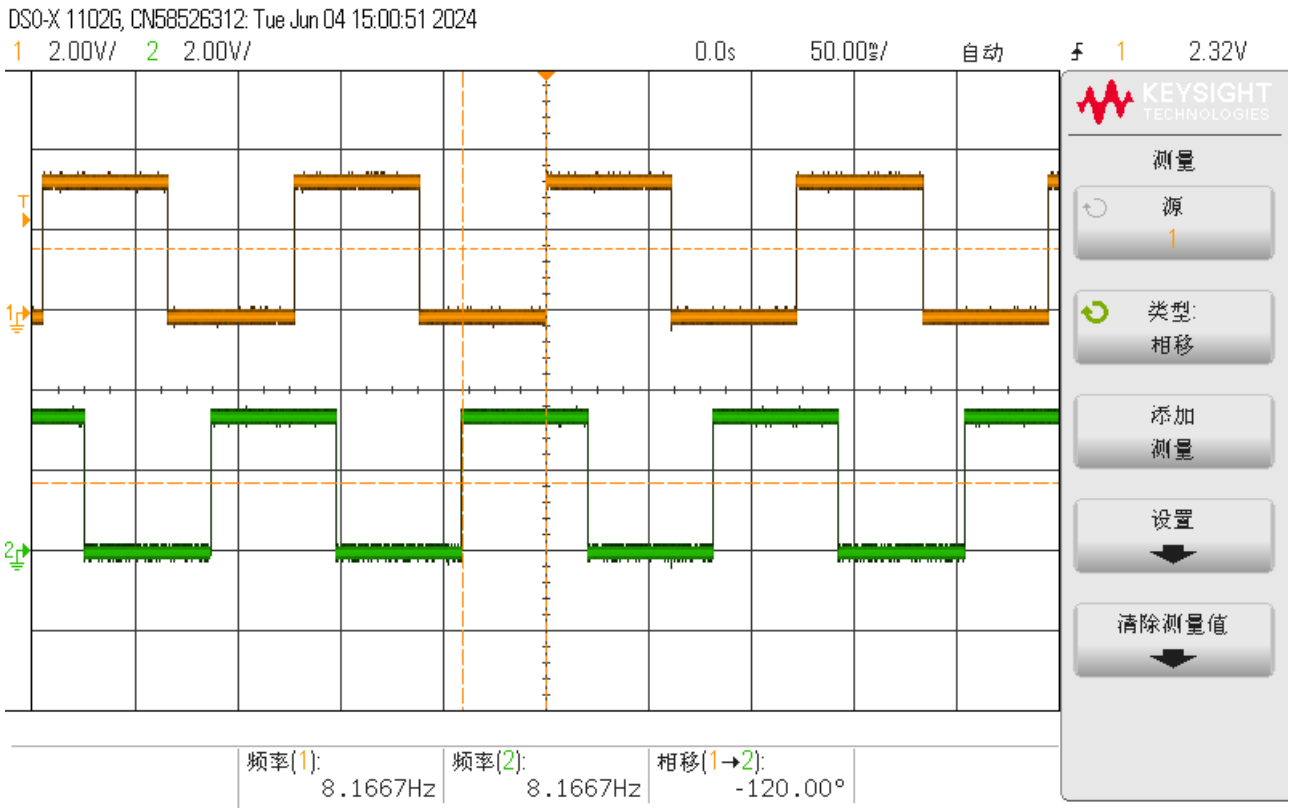
3. 实验数据和结果

A-B 正转

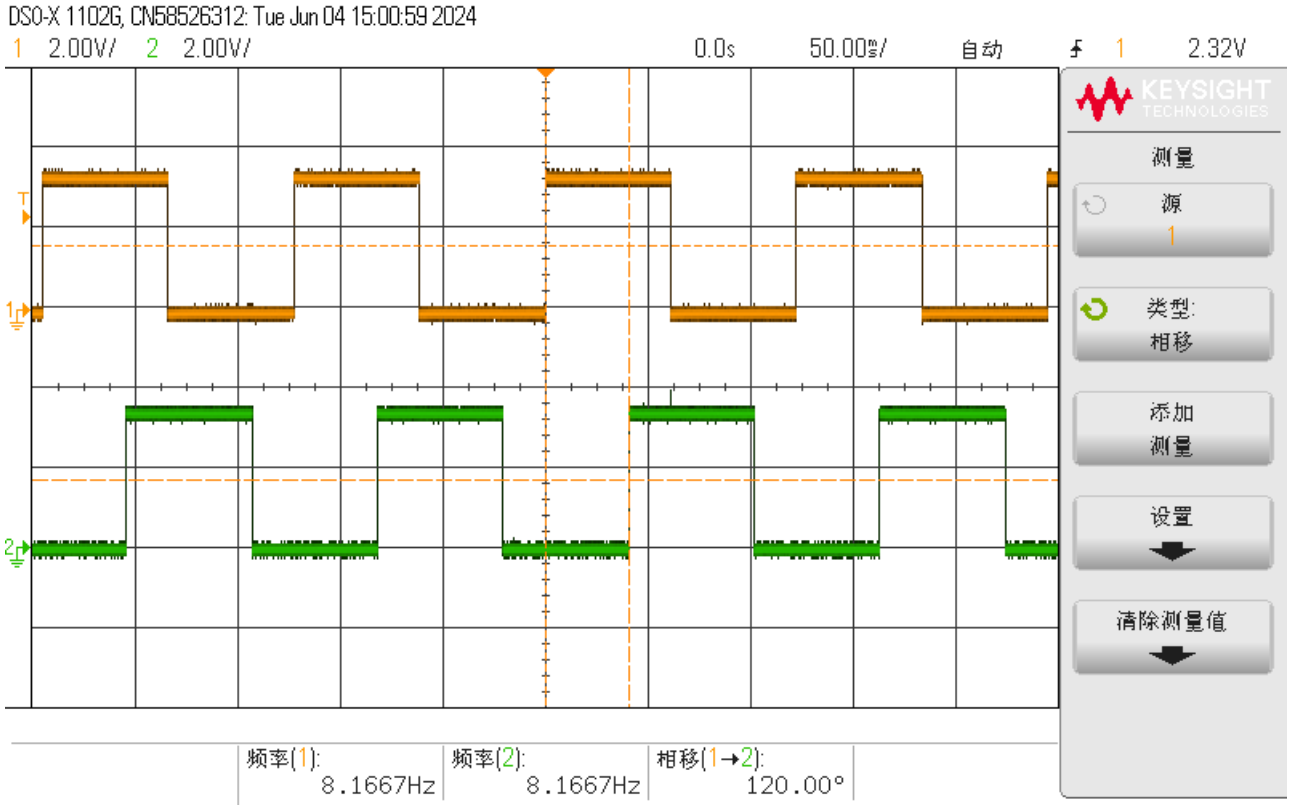


实验名称：FPGA 步进电机脉冲分配器　姓名：____严旭铧_____学号：___3220101731_____

A-C 正转

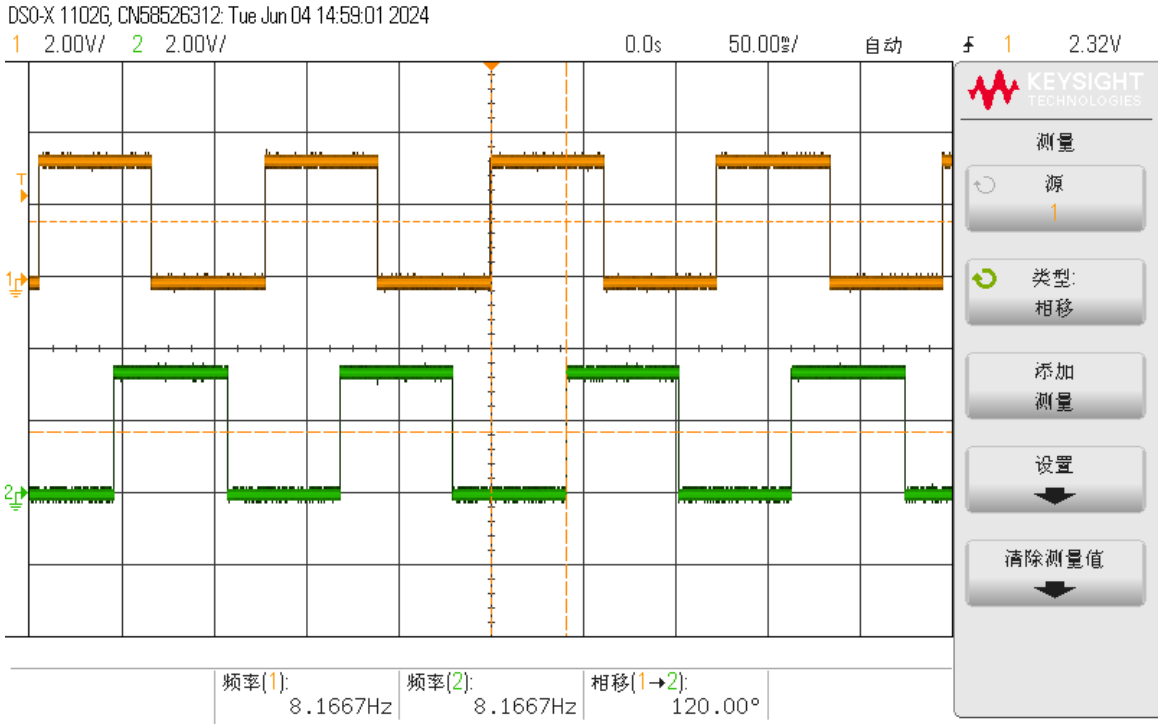


B-C 正转

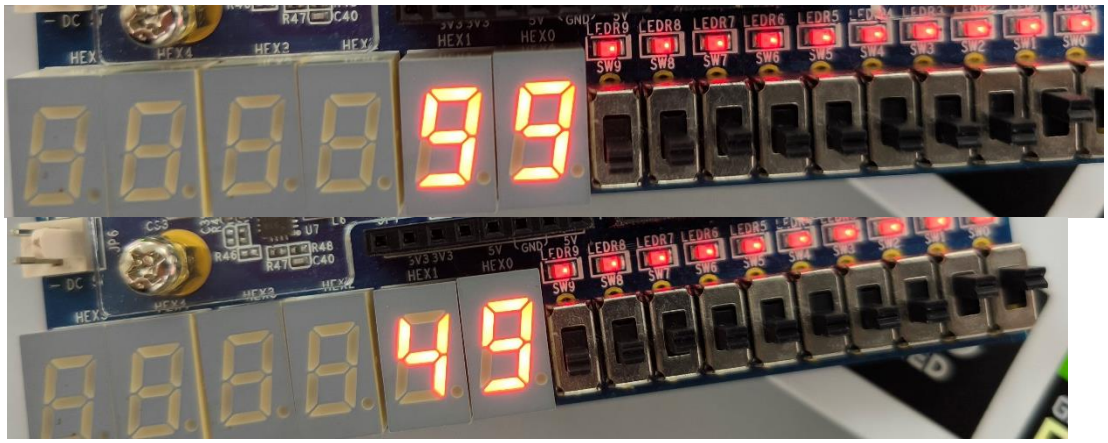


实验名称：FPGA 步进电机脉冲分配器　姓名：____严旭铧_____学号：___3220101731_____

AB 反转



1-99 计数正常



四、 实验体会与思考

1. 本次实验使用了 FPGA 来实现步进电机的脉冲分配，这个功能在前面的课程中用逻辑芯片做出来过。相比来说，FPGA 具有可编程的巨大优势，可以非常方便地对脉冲进行分配和控制。
2. 本实验中依然体现了模块化编程的思想。实验代码中的许多部分都是由前面实验中的代码组合而成，例如数码管显示和 1-99 计数。这说明还是要把这些基本的代码熟练运用。