矩阵键盘扫描设计

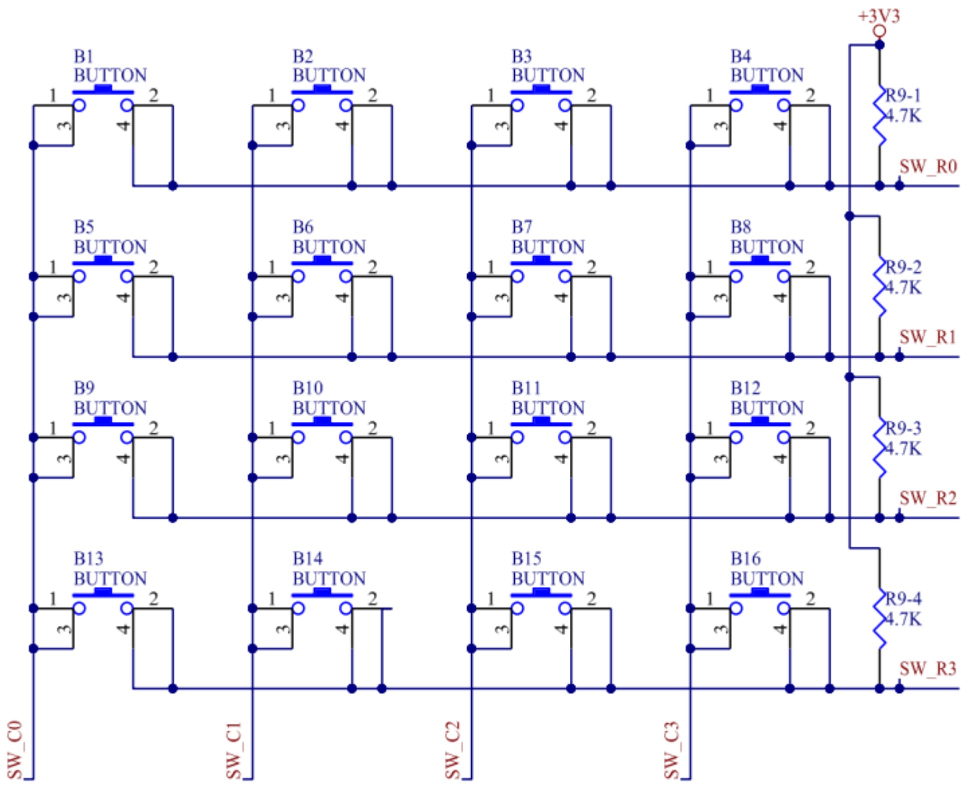
PB16060240
黄骏达

实验目的

1. 进一步学习并掌握 Quartus 设计的方法及步骤
2. 熟悉 VHDL 语言电路设计方法
3. 熟悉 VHDL test bench 的设计
4. 学习并掌握利用 VHDL 描述并设计电路的方法及步骤
5. 学习并掌握健盘阵列的扫描输入的方法及实现过程

键盘扫描原理：

1. 此次矩阵键盘是一个由 4X4 个按键开关组成的阵列，可实现 16 中状态输入，矩阵键盘阵列如下：



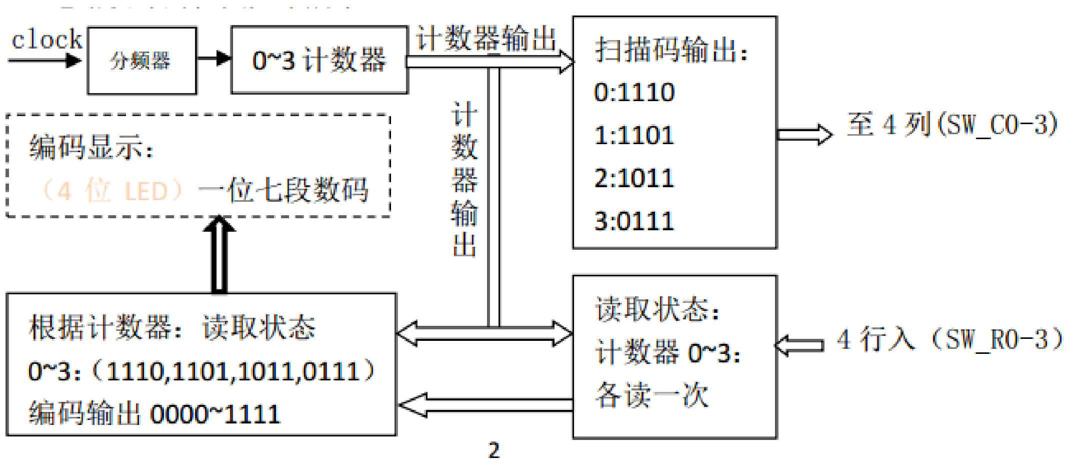
4X4 键盘阵列

2. 在行输入端，由上拉电阻接到 3.3v 标准 TTL 电平上，在列输入端，列电平输出直接由控制器进行控制。

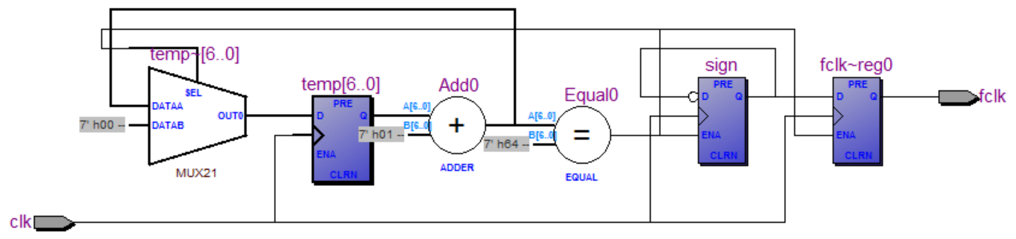
3. 当按下按键，如 B1，此时，若列输出为 0111，则在行输入端能够检测到输入为 0111，这是因为，对于按键按下的行，由于电路接通，使得此行输入电平变为 0 电平，而对于没有按键按下的行，由于电路断开，因此行电平依旧被外部电压拉高在高电平状态；同时，若列输出为 1011，因为在第二列并无按键按下，因此行输入全部为高电平，即在输入端能够检测到 1111，其他列输出同理。

实验设计：

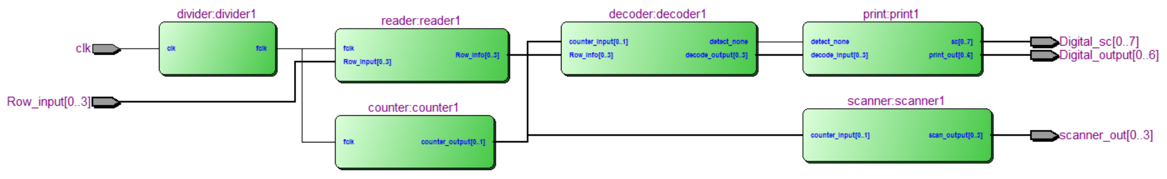
1. 模块化电路逻辑简图如下：



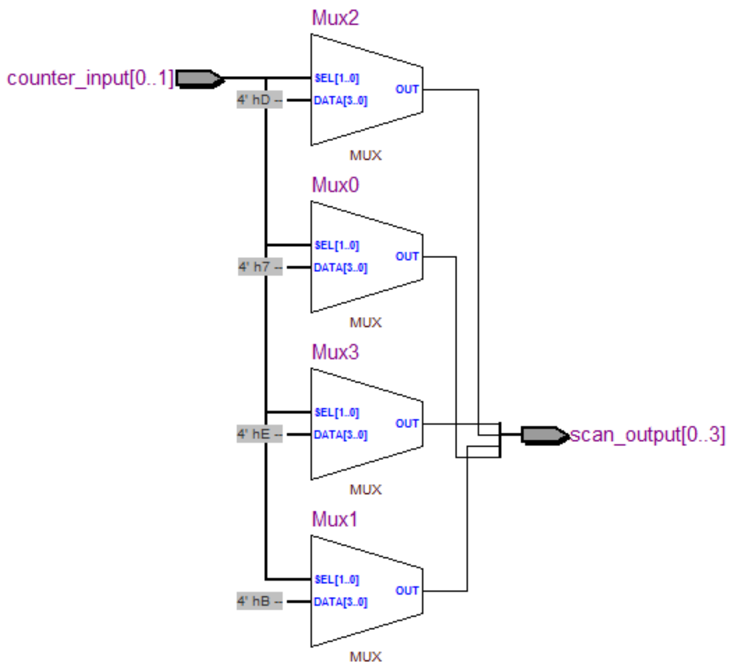
2. 由于时钟 50MHz 太快，因此需要一个分频器进行分频，分频器电路图如下，为 200 分频：



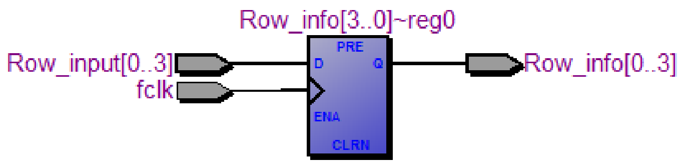
3. 分频器的分频信号输出至计数器，计数器再将输出输出给译码器、扫描器和读取设备，作为信号触发源，总电路图如下所示：



4. 每个计数器输出到来时，扫描器输出对应的电平，如模块逻辑图所示。同时读取设备读入行电平信号，扫描器的电路图如下所示：



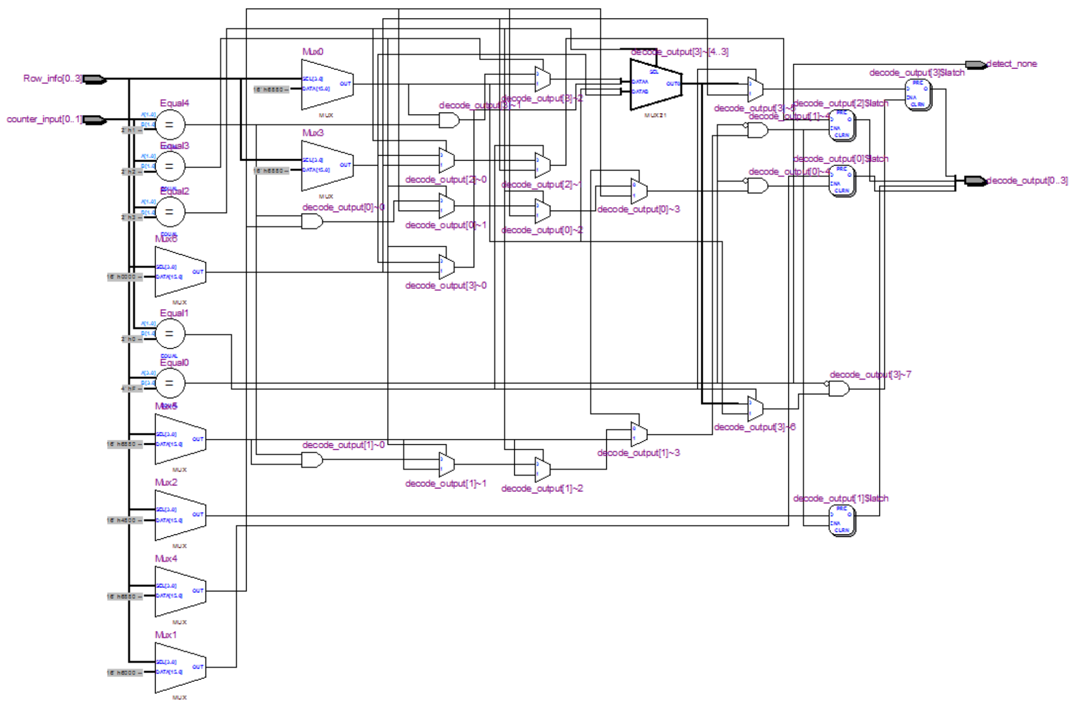
5. 然后读取设备将从键盘获取的行信息输出给译码器，译码器根据计数器的扫描信号和行信号进行译码,行输入电路图如下所示：



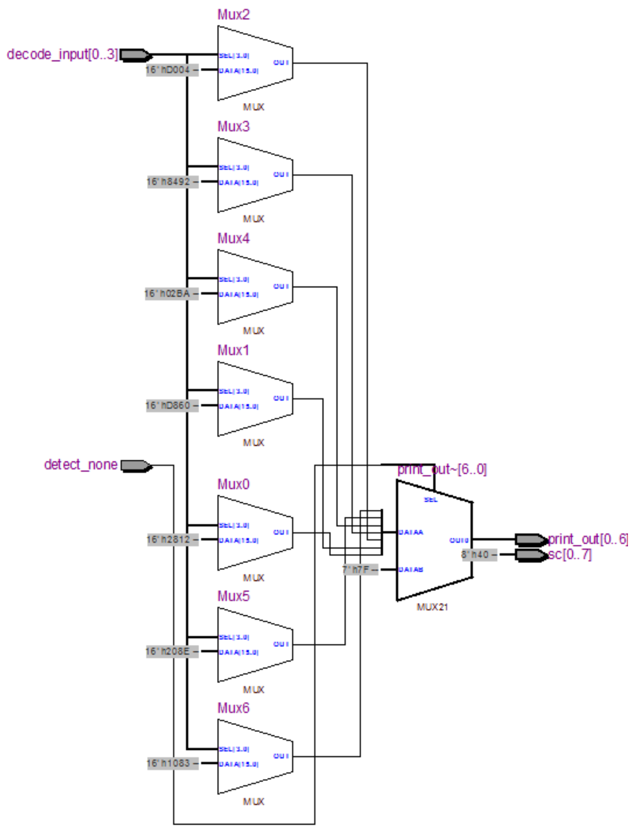
6. 译码器译码后将显示信息输出给数码管，同时若为读取到有效的信号，则另数码管为熄灭，译码器与数码管显示电路图在后面贴出。

电路图：

译码器电路图：



数码管控制电路：



VHDL 源代码：

(scanner.vhd) 列扫描

```
library ieee;
use ieee.std_logic_1164.all;

entity scanner is
port(counter_input : in std_logic_vector(0 to 1);
        scan_output : out std_logic_vector(0 to 3));
end entity scanner;

architecture behave of scanner is
begin
    process(counter_input)
        begin
        case counter_input is
            when "00" => scan_output <= "1110";
            when "01" => scan_output <= "1101";
            when "10" => scan_output <= "1011";
            when "11" => scan_output <= "0111";
            when others => NULL;
        end case;
    end process;
end behave;
```

(reader.vhd) 行读入

```
library ieee;
use ieee.std_logic_1164.all;

entity reader is
port(fclk : in std_logic;
        Row_input : in std_logic_vector(0 to 3);
        Row_info : out std_logic_vector(0 to 3));
end entity reader;

architecture behave of reader is
begin
    process(fclk)
    begin
        if rising_edge(fclk) then
            Row_info <= Row_input;
        end if;
    end process;
end behave;
```

(print.vhd) 数码管控制

```
library ieee;
use ieee.std_logic_1164.all;

entity print is
port(decode_input : in std_logic_vector(0 to 3);
        sc : out std_logic_vector(0 to 7);
        print_out : out std_logic_vector(0 to 6);
        detect_none : in std_logic);
end entity print;

architecture behave of print is
begin
    sc <= "01000000";
    process(decode_input,detect_none)
    variable a0:std_logic_vector(0 to 6) := "0000001";
    variable a1:std_logic_vector(0 to 6) := "1001111";
    variable a2:std_logic_vector(0 to 6) := "0010010";
    variable a3:std_logic_vector(0 to 6) := "0000110";
    variable a4:std_logic_vector(0 to 6) := "1001100";
    variable a5:std_logic_vector(0 to 6) := "0100100";
    variable a6:std_logic_vector(0 to 6) := "0100000";
    variable a7:std_logic_vector(0 to 6) := "0001111";
    variable a8:std_logic_vector(0 to 6) := "0000000";
    variable a9:std_logic_vector(0 to 6) := "0000100";
    variable aA:std_logic_vector(0 to 6) := "0001000";
    variable aB:std_logic_vector(0 to 6) := "1100000";
    variable aC:std_logic_vector(0 to 6) := "0110001";
    variable aD:std_logic_vector(0 to 6) := "1000010";
    variable aE:std_logic_vector(0 to 6) := "0110000";
    variable aF:std_logic_vector(0 to 6) := "0111000";
    begin
        if(detect_none = '1') then
            print_out <= "1111111";
        else
            case decode_input is
                when "0000" => print_out <= a0;
                when "0001" => print_out <= a1;
                when "0010" => print_out <= a2;
                when "0011" => print_out <= a3;
                when "0100" => print_out <= a4;
                when "0101" => print_out <= a5;
                when "0110" => print_out <= a6;
```

```
                    when "0111" => print_out <= a7;
                    when "1000" => print_out <= a8;
                    when "1001" => print_out <= a9;
                    when "1010" => print_out <= aA;
                    when "1011" => print_out <= aB;
                    when "1100" => print_out <= aC;
                    when "1101" => print_out <= aD;
                    when "1110" => print_out <= aE;
                    when "1111" => print_out <= aF;
                    when others => print_out <= "1111111";
                end case;
            end if;
        end process;
end behave;
```

(decoder.vhd) 译码器

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;

entity decoder is
port(counter_input : in std_logic_vector(0 to 1);
        Row_info : in std_logic_vector(0 to 3);
        decode_output: out std_logic_vector(0 to 3);
        detect_none : out std_logic);
end entity decoder;

architecture behave of decoder is

begin

    process(counter_input,Row_info)

    begin
        if (Row_info = "1111") then
            detect_none <= '1';
        else
            detect_none <= '0';
            if (counter_input = "00") then --10
                case Row_info is
                    when "0111" => decode_output <= conv_std_logic_vector(0,4);
                    when "1011" => decode_output <= conv_std_logic_vector(4,4);
                    when "1101" => decode_output <= conv_std_logic_vector(8,4);
                    when "1110" => decode_output <= conv_std_logic_vector(12,4);
```

```
                        when others => NULL;
                    end case;
                elsif (counter_input = "11") then --01
                    case Row_info is
                        when "0111" => decode_output <= conv_std_logic_vector(1,4);
                        when "1011" => decode_output <= conv_std_logic_vector(5,4);
                        when "1101" => decode_output <= conv_std_logic_vector(9,4);
                        when "1110" => decode_output <= conv_std_logic_vector(13,4);
                        when others => NULL;
                    end case;
                elsif (counter_input = "10") then    --00
                    case Row_info is
                        when "0111" => decode_output <= conv_std_logic_vector(2,4);
                        when "1011" => decode_output <= conv_std_logic_vector(6,4);
                        when "1101" => decode_output <= conv_std_logic_vector(10,4);
                        when "1110" => decode_output <= conv_std_logic_vector(14,4);
                        when others => NULL;
                    end case;
                elsif (counter_input = "01") then    --11
                    case Row_info is
                        when "0111" => decode_output <= conv_std_logic_vector(3,4);
                        when "1011" => decode_output <= conv_std_logic_vector(7,4);
                        when "1101" => decode_output <= conv_std_logic_vector(11,4);
                        when "1110" => decode_output <= conv_std_logic_vector(15,4);
                        when others => NULL;
                    end case;
                end if;
            end if;
        end process;

end behave;
```

(counter.vhd) 计数器

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity counter is
port(fclk : in std_logic;
        counter_output : out std_logic_vector(0 to 1));
end entity counter;

architecture behave of counter is
```

```
begin
        process(fclk)
        variable temp : std_logic_vector(0 to 1) := "00";
        begin
                if (rising_edge(fclk)) then
                        temp := temp + 1;
                end if;
                counter_output <= temp;
        end process;
end behave;
```

(Keyboard.vhd) 顶层实体

```
library ieee;
use ieee.std_logic_1164.all;

entity Keyboard is
port(clk : in std_logic;
                scanner_out : out std_logic_vector(0 to 3);
                Row_input : in std_logic_vector(0 to 3);
                Digital_sc : out std_logic_vector(0 to 7);
                Digital_output : out std_logic_vector(0 to 6));
end entity Keyboard;

architecture behave of Keyboard is

component divider is
port(clk : in std_logic;
                fclk : out std_logic);
end component divider;

component counter is
port(fclk : in std_logic;
                counter_output : out std_logic_vector(0 to 1));
end component counter;

component scanner is
port(counter_input : in std_logic_vector(0 to 1);
                scan_output : out std_logic_vector(0 to 3));
end component scanner;

component reader is
port(fclk : in std_logic;
                Row_input : in std_logic_vector(0 to 3);
                Row_info : out std_logic_vector(0 to 3));
```

```
end component reader;

component decoder is
port(counter_input : in std_logic_vector(0 to 1);
		Row_info : in std_logic_vector(0 to 3);
		decode_output: out std_logic_vector(0 to 3);
		detect_none : out std_logic);
end component decoder;

component print is
port(decode_input : in std_logic_vector(0 to 3);
		sc : out std_logic_vector(0 to 7);
		print_out : out std_logic_vector(0 to 6);
		detect_none : in std_logic);
end component print;

signal counter_bus : std_logic_vector(0 to 1);
signal Row_info_bus : std_logic_vector(0 to 3);
signal code_bus : std_logic_vector(0 to 3);
signal detect_none : std_logic;
signal fclk : std_logic;


begin
divider1 : divider port map(clk, fclk);
counter1 : counter port map(fclk , counter_bus);
scanner1 : scanner port map(counter_bus, scanner_out);
reader1 : reader port map(fclk, Row_input, Row_info_bus);
decoder1 : decoder port map(counter_bus, Row_info_bus, code_bus, detect_none);
print1 : print port map(code_bus, Digital_sc, Digital_output, detect_none);

end behave;
```

(Keyboard_tb.vhd) 测试板

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;

entity Keyboard_tb is
end entity;

architecture behavior of Keyboard_tb is
```

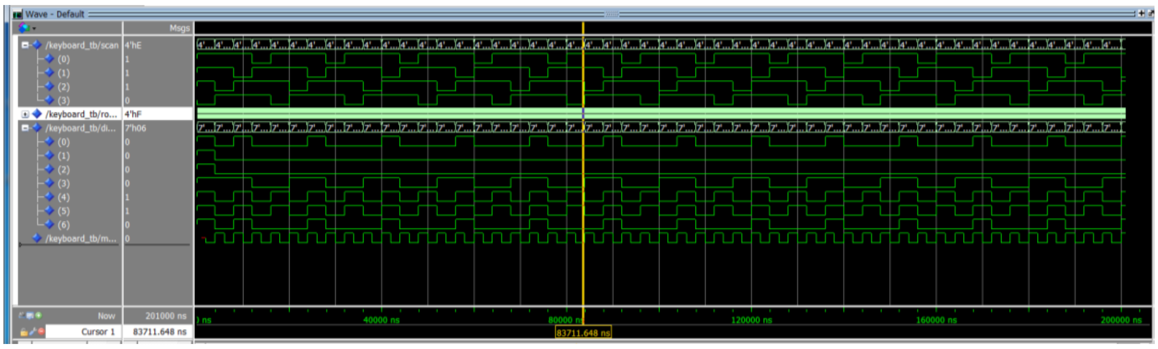
```
component Keyboard is
port(clk : in std_logic;
            scanner_out : out std_logic_vector(0 to 3);
            Row_input : in std_logic_vector(0 to 3);
            Digital_sc : out std_logic_vector(0 to 7);
            Digital_output : out std_logic_vector(0 to 6));
end component;

signal clk : std_logic;
signal scan : std_logic_vector(0 to 3);
signal row_input : std_logic_vector(0 to 3);
signal digital_sc : std_logic_vector(0 to 7);
signal digital_output : std_logic_vector(0 to 6);

begin
main : Keyboard port map(clk, scan, row_input,digital_sc, digital_output);

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

     process
     begin
          for j in 0 to 3 loop
               row_input <= (j => '0' , others => '1');
               wait for 20 ns;
               for i in 0 to 3 loop
                    if (i /= j) then
                         row_input <= (others => '1');
                         wait for 20 ns;
                    end if;
               end loop;
          end loop;
     end process;

end behavior;
```

(divider.vhd) 分频器

```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity divider is
port(clk : in std_logic;
		fclk : out std_logic);
end entity divider;
architecture behave of divider is
begin
	process(clk)
	variable temp : integer range 0 to 100 := 0;
	variable sign : std_logic :='0';
	begin
		if (rising_edge(clk)) then
			temp := temp + 1;
			if(temp = 100) then
				temp := 0;
				fclk <= sign;
				sign := not sign;
			end if;
		end if;
	end process;
end behave;
```

仿真结果记录：



FPGA 验证结果：

按下按键，在数码管上显示对应按键的编码，若未按下按键，按键不显示。

实验总结：

1. 前几次测试没有加分频器，直接将 50MHz 时钟信号引入计数器。但是这样会出现问题，具体原因还不是很清楚，表现是，数码管显示 0，并且按下按键，部分按键按下后数码管的显示会有轻微变化，但是主要还是显示 0。加上分频器之后问题自动解决。
2. 程序写入时出现 fail，主要是 USB 端口的连线问题导致。