实验四：键盘阵列

一. 实验内容

实现一个健盘阵列：按下某个按键后，数码管上显示对应的数字（符号），松开按键后，显示内容保持不变。

二. 设计分析

编程思路：

由于矩阵键盘中有16个按键，而键盘与FPGA的连接只有8根线，所以不能用一根信号线来判断一个按键是否被按下，所以需要采用动态扫描的方法。

对FPGA而言，可以把行信号作为一组输入信号，列信号作为一组输出信号，分别进行控制。通过不断给列信号输出一个扫描序列，再通过读取行信号来判断哪一个按键被按下，即FPGA向列信号SW_C0、SW_C1、SW_C2、SW_C3循环输出“1110、1101、1011、0111”来驱动键盘阵列，每输出一个列序列后，紧接着读取相应的4个行信号。通过读取的SW_R0、SW_R1、SW_R2、SW_R3的数据或状态来判断16个按键中哪个按键被按下，并对其状态做编码输出，并将对应按键的号码显示在一个数码管上。

三. VHDL源代码

顶层实体：(FPGA_EXP4_dxh.vhd)

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

entity FPGA_EXP4_dxh is
    port(clk: in std_logic;
    segment: out std_logic_vector(6 downto 0);
    row: in std_logic_vector(3 downto 0);
    column: out std_logic_vector(3 downto 0);
    COM: out std_logic
    );
end FPGA_EXP4_dxh;

architecture arch_FPGA_EXP4_dxh of FPGA_EXP4_dxh is

--  component Div50MHz
--      port(clk: in std_logic;
--      clkout: out std_logic
--      );
--  end component;

--  component debounce
--      port(raw: in std_logic_vector(3 downto 0);
--      clk: in std_logic;
--      de: out std_logic_vector(3 downto 0)
--      );
--  end component;

    component Counter4
        port (clk: in std_logic;
        q: out std_logic_vector (1 downto 0)
        );
    end component;

    component decoder24
        port(
        A: in std_logic_vector (1 downto 0);
        B:out std_logic_vector (3 downto 0));
    end component;

    component driver
        port(
        column: in std_logic_vector(3 downto 0);
        row: in std_logic_vector(3 downto 0);
        state: out std_logic_vector(3 downto 0)
        );
     end component;

     component decoder47
        port(
        A: in std_logic_vector (3 downto 0);
        B: out std_logic_vector (6 downto 0));
     end component;

--   signal newclk: std_logic;
--  signal row_de: std_logic_vector(3 downto 0);
    signal num: std_logic_vector(1 downto 0);
    signal state: std_logic_vector(3 downto 0);
    signal column_in: std_logic_vector(3 downto 0);

begin
--  u1: Div50MHz port map (clk, newclk);
--  u2: debounce port map (row, clk, row_de);
    u3: Counter4 port map (clk, num);
    u4: decoder24 port map (num, column_in);
    u5: driver port map (column_in, row, state);
    u6: decoder47 port map (state, segment);
    column <= column_in;
    COM <= '1';
end arch_FPGA_EXP4_dxh;
```

分频器：（Div50MHz.vhd）

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

```
entity Div50MHz is
    port(clk: in std_logic;
    clkout: out std_logic
    );
end Div50MHz;

architecture arch_Div50MHz of Div50MHz is
    signal cnt: integer range 0 to 24999;
    signal tmp_clk: std_logic;
begin
    process(clk)
    begin
        if(clk'event and clk = '1') then
            if (cnt = 24999) then
                cnt <= 0;
                tmp_clk <= NOT tmp_clk;
            else
                cnt <= cnt + 1;
            end if;
        end if;
    end process;
    clkout <= tmp_clk;
end arch_Div50MHz;
```

计数器：（Counter4.vhd）

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

entity Counter4 is
    port (clk: in std_logic;
    q: out std_logic_vector (1 downto 0)
    );
end Counter4;

architecture arch_Counter4 of Counter4 is
    signal tmp_q: std_logic_vector(1 downto 0) := "00";
begin
    q <= tmp_q;
    process(clk)
    begin
        if (clk'event and clk='1') then
            if (tmp_q="11") then
                tmp_q <= "00";
            else
                tmp_q <= tmp_q + 1;
            end if;
        end if;
    end process;
end arch_Counter4;
```

2-4译码器：（decoder24.vhd）

译码器的输出信号作为列信号输出。

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

entity decoder24 is
    port(
    A: in std_logic_vector (1 downto 0);
    B:out std_logic_vector (3 downto 0));
end decoder24;

architecture arch_decoder24 of decoder24 is
begin
    process(A)
    begin
        case A is
            when "00" => B <= "1110";
            when "01" => B <= "1101";
            when "10" => B <= "1011";
            when "11" => B <= "0111";
            when others => null;
        end case;
    end process;
end arch_decoder24;
```

driver.vhd:

读取行信号，根据行信号和列信号判断哪个按键被按下，并编码成对应的状态。

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

entity driver is
    port(
    column: in std_logic_vector(3 downto 0);
    row: in std_logic_vector(3 downto 0);
    state: out std_logic_vector(3 downto 0)
    );
end driver;

architecture arch_driver of driver is
begin
    process(column, row,  clk)
    begin
        if (clk'event and clk='1') then
            case row is
                when "1110" =>
                    case column is
                        when "1110" => state <= "0000";
                        when "1101" => state <= "0001";
                        when "1011" => state <= "0010";
                        when "0111" => state <= "0011";
                        when others => null;
                    end case;
                when "1101" =>
                    case column is
                        when "1110" => state <= "0100";
                        when "1101" => state <= "0101";
                        when "1011" => state <= "0110";
                        when "0111" => state <= "0111";
                        when others => null;
                    end case;
                when "1011" =>
                    case column is
                        when "1110" => state <= "1000";
                        when "1101" => state <= "1001";
                        when "1011" => state <= "1010";
                        when "0111" => state <= "1011";
                        when others => null;
                    end case;
                when "0111" =>
                    case column is
                        when "1110" => state <= "1100";
                        when "1101" => state <= "1101";
                        when "1011" => state <= "1110";
                        when "0111" => state <= "1111";
                        when others => null;
                    end case;
                when others => null;
            end case;
        end if;
    end process;
end arch_driver;
```

4-7译码器：（decoder47.vhd）

根据driver输出的四位状态，译码生成7位数码管的段控制信号，将对应号码显示在数码管上。

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

entity decoder47 is
    port(
    A: in std_logic_vector (3 downto 0);
    B: out std_logic_vector (6 downto 0));
end decoder47;

architecture arch_decoder47 of decoder47 is
begin
    process(A)
    begin
        case A is
            when "0000" => B <= "1001111";
            when "0001" => B <= "0010010";
            when "0010" => B <= "0000110";
            when "0011" => B <= "0001000";
            when "0100" => B <= "1001100";
            when "0101" => B <= "0100100";
```

```
                when "0110" => B <= "0100000";
                when "0111" => B <= "1100000";
                when "1000" => B <= "0001111";
                when "1001" => B <= "0000000";
                when "1010" => B <= "0000100";
                when "1011" => B <= "0110001";
                when "1100" => B <= "1111110";
                when "1101" => B <= "0000001";
                when "1110" => B <= "1111110";
                when "1111" => B <= "1000010";
                when others => null;
            end case;
        end process;
    end arch_decoder47;
```

消抖电路：（debounce.vhd）

对按键控制的行信号进行消抖处理，当低电平（高电平）数量到达一定阈值后，才输出低电平（高电平）

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

entity debounce is
    port(raw: in std_logic_vector(3 downto 0);
    clk: in std_logic;
    de: out std_logic_vector(3 downto 0)
    );
end debounce;

architecture arch_debounce of debounce is
signal KL3, KL2, KL1, KL0: integer range 0 to 49;
signal KH3, KH2 ,KH1, KH0: integer range 0 to 49;
begin
    p3: process(clk)
    begin
        if (clk'event and clk='1') then
            if(raw(3)='1') then
                if (KH3 = 49) then
                    KH3 <= 0;
                    de(3) <= '1';
                else
                    KH3 <= KH3 + 1;
                end if;
            else
                if (KL3 = 49) then
                    KL3 <= 0;
                    de(3) <= '0';
                else
                    KL3 <= KL3 + 1;
                end if;
            end if;
        end if;
    end process;
```

```vhdl
    p2: process(clk)
    begin
        if (clk'event and clk='1') then
            if(raw(2)='1') then
                if (KH2 = 49) then
                    KH2 <= 0;
                    de(2) <= '1';
                else
                    KH2 <= KH2 + 1;
                end if;
            else
                if (KL2 = 49) then
                    KL2 <= 0;
                    de(2) <= '0';
                else
                    KL2 <= KL2 + 1;
                end if;
            end if;
        end if;
    end process;

    p1: process(clk)
    begin
        if (clk'event and clk='1') then
            if(raw(1)='1') then
                if (KH1 = 49) then
                    KH1 <= 0;
                    de(1) <= '1';
                else
                    KH1 <= KH1 + 1;
                end if;
            else
                if (KL1 = 49) then
                    KL1 <= 0;
                    de(1) <= '0';
                else
                    KL1 <= KL1 + 1;
                end if;
            end if;
        end if;
    end process;

    p0: process(clk)
    begin
        if (clk'event and clk='1') then
            if(raw(0)='1') then
                if (KH0 = 49) then
                    KH0 <= 0;
                    de(0) <= '1';
                else
                    KH0 <= KH0 + 1;
                end if;
            else
                if (KL0 = 49) then
```

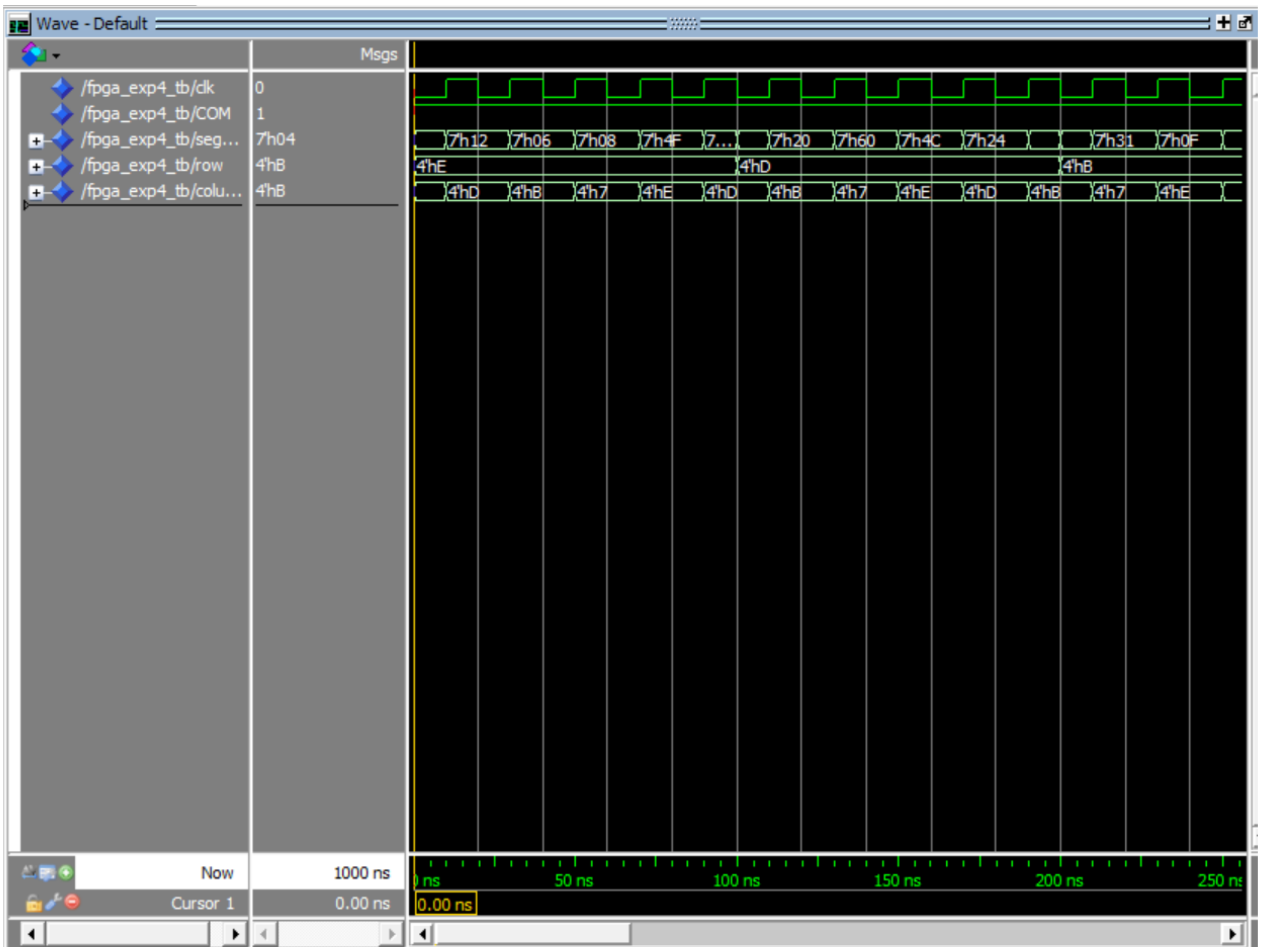
```
                    KL0 <= 0;
                    de(0) <= '0';
                else
                    KL0 <= KL0 + 1;
                end if;
            end if;
        end if;
    end process;
end arch_debounce;
```

仿真代码：（FPGA_EXP4_dxh_tb.vhd）

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

entity FPGA_EXP4_tb is
end FPGA_EXP4_tb;

architecture arch_FPGA_EXP4_tb of FPGA_EXP4_tb is

    component FPGA_EXP4
        port(clk: in std_logic;
        segment: out std_logic_vector(6 downto 0);
        row: in std_logic_vector(3 downto 0);
        column: out std_logic_vector(3 downto 0);
        COM: out std_logic
        );
    end component;

    signal clk: std_logic;
    signal COM: std_logic := '1';
    signal segment: std_logic_vector(6 downto 0);
    signal row: std_logic_vector(3 downto 0);
    signal column: std_logic_vector(3 downto 0);
begin
    u_tb: FPGA_EXP4 port map (clk, segment, row, column, COM);
    p1: process
    begin
        clk <= '0';
        wait for 10 ns;
        clk <= '1';
        wait for 10 ns;
    end process;

    p2: process
    begin
        row <= "1110";
        wait for 100 ns;
        row <= "1101";
        wait for 100 ns;
        row <= "1011";
        wait for 100 ns;
        row <= "0111";
        wait for 100 ns;
    end process;
end arch_FPGA_EXP4_tb;
```

四. 仿真结果FPGA_EXP4_dxh_tb.vhd

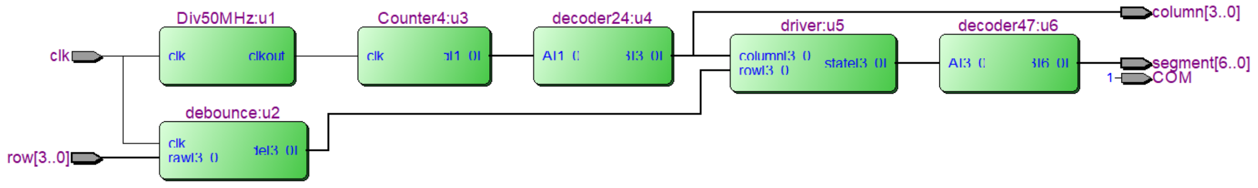
仿真结果：（仿真时未带分频器和防抖电路）


Wave - Default
Msgs
/fpga_exp4_tb/clk 0
/fpga_exp4_tb/COM 1
/fpga_exp4_tb/seg... 7'h04
/fpga_exp4_tb/row 4'hB
/fpga_exp4_tb/colu... 4'hB
7'h12 7'h06 7'h08 7'h4F 7... 7'h20 7'h60 7'h4C 7'h24 7'h31 7'h0F
4'hE 4'hD 4'hB
4'hD 4'hB 4'h7 4'hE 4'hD 4'hB 4'h7 4'hE 4'hD 4'hB 4'h7 4'hE
Now 1000 ns
Cursor 1 0.00 ns
0 ns 50 ns 100 ns 150 ns 200 ns 250 ns
0.00 ns

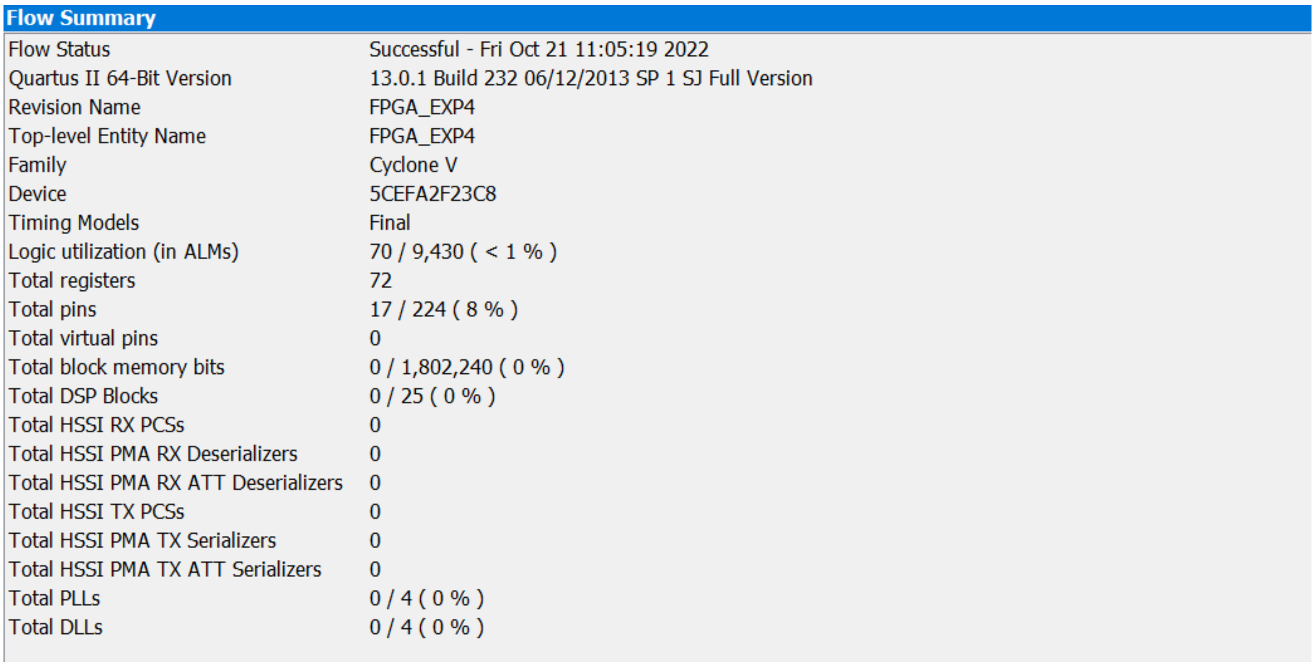
五. 中间结果

1.RTL电路图

整体结构：



2.资源占用信息



Flow Summary	
Flow Status	Successful - Fri Oct 21 11:05:19 2022
Quartus II 64-Bit Version	13.0.1 Build 232 06/12/2013 SP 1 SJ Full Version
Revision Name	FPGA_EXP4
Top-level Entity Name	FPGA_EXP4
Family	Cyclone V
Device	5CEFA2F23C8
Timing Models	Final
Logic utilization (in ALMs)	70 / 9,430 (< 1 %)
Total registers	72
Total pins	17 / 224 (8 %)
Total virtual pins	0
Total block memory bits	0 / 1,802,240 (0 %)
Total DSP Blocks	0 / 25 (0 %)
Total HSSI RX PCSs	0
Total HSSI PMA RX Deserializers	0
Total HSSI PMA RX ATT Deserializers	0
Total HSSI TX PCSs	0
Total HSSI PMA TX Serializers	0
Total HSSI PMA TX ATT Serializers	0
Total PLLs	0 / 4 (0 %)
Total DLLs	0 / 4 (0 %)

六. 硬件验证

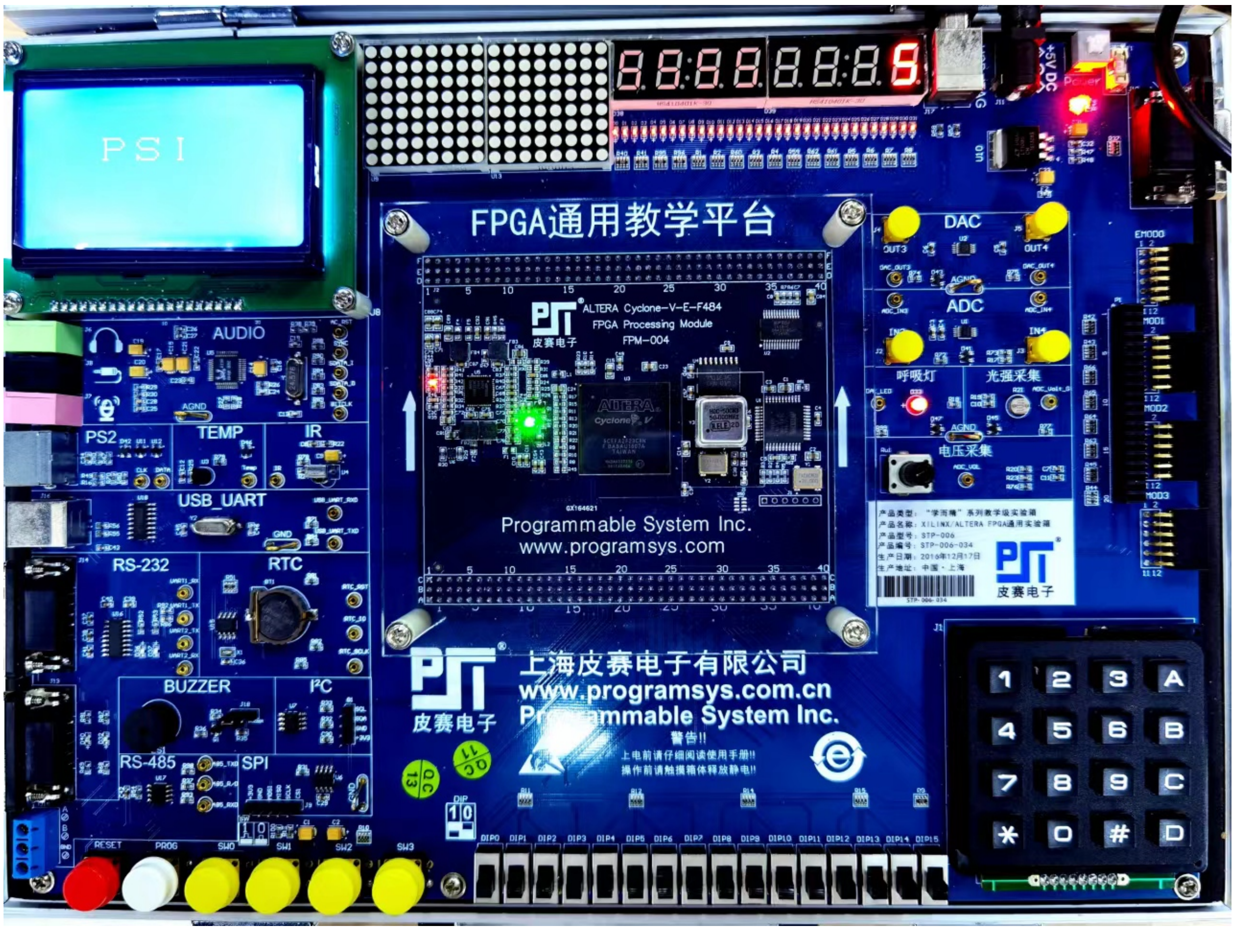
1. 管脚锁定情况:

端口	管脚
clk	PIN_W16
row[3]	PIN_V21
row[2]	PIN_W21
row[1]	PIN_Y19
row[0]	PIN_Y21
COM	PIN_B16
column[3]	PIN_AB22
column[2]	PIN_AB20
column[1]	PIN_AA20
column[0]	PIN_Y17
segment[6]	PIN_K19
segment[5]	PIN_H18
segment[4]	PIN_K20

端口	管脚
segment[3]	PIN_M18
segment[2]	PIN_E16
segment[1]	PIN_G13
segment[0]	PIN_G17

2. 实验效果图

   按下矩阵键盘中的按钮，最右侧数码管显示对应符号（1~9显示对应数字，"a~b"显示对应字母，"*"和"#"显示为"-"），且放开按钮后，数字保持不变。



七. 实验总结

程序设计代码在仿真时未出现任何错误。但在硬件验证时，按下对应按键后数码管显示错误的数字，且不稳定。这是因为仿真过程并不考虑物理器件可能引起的延时。在实际工作过程中，会因为延时而导致系统前后部分不同步，所以不能达到理想的实验效果。

解决方法：在系统后半部分的组合逻辑电路中，加入语句`if(clk'event and clk='1')`，使系统的所有部件能够同步工作。