点亮数字人生

2017011620 计73 李家昊

实验目的

1. 通过数码管点亮程序，熟悉VHDL语言，了解掌握硬件程序的编写规范。
2. 掌握EDA软件的使用方法和工作流程。
3. 进一步理解可编程芯片的工作原理。

实验任务

1. 同时点亮一个经过译码的数码管和一个未经过译码的数码管。数码管根据输入显示从 0 到 F （带译码的显示 0 到 9 ）。
2. 点亮三个数码管（至少要使用一个不带译码的数码管），这三个数码管分别显示奇数列、偶数列和自然数列（通过CLK信号控制数列变化并且有RST复位功能）

基本要求

代码如下

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

entity kindle_life is
    port(
        key:in std_logic_vector(3 downto 0);
        display:out std_logic_vector(6 downto 0);
        display_4:out std_logic_vector(3 downto 0)
    );
end kindle_life;

architecture bhv of kindle_life is
begin
    display_4<=key;

    process(key)
    begin
        case key is
            when "0000"=>display<="1111110"; -- 0
            when "0001"=>display<="0110000"; -- 1
            when "0010"=>display<="1101101"; -- 2
            when "0011"=>display<="1111001"; -- 3
            when "0100"=>display<="0110011"; -- 4
            when "0101"=>display<="1011011"; -- 5
            when "0110"=>display<="1011111"; -- 6
```

```
            when "0111"=>display<="1110000"; -- 7
            when "1000"=>display<="1111111"; -- 8
            when "1001"=>display<="1111011"; -- 9
            when "1010"=>display<="1110111"; -- A
            when "1011"=>display<="0011111"; -- B
            when "1100"=>display<="1001110"; -- C
            when "1101"=>display<="0111101"; -- D
            when "1110"=>display<="1001111"; -- E
            when "1111"=>display<="1000111"; -- F
            when others=>display<="0000000"; -- else 0
        end case;
    end process;
end bhv;
```

工作原理

设置 1 个输入端口：开关输入端口 key。

设置 2 个输出端口：输出端口 display（不带译码器）和 display_4（带译码器）。

通过控制 4 个开关，控制输入信号 key 的值，将 key 的值直接赋给 display_4，点亮带译码器的数码管，将 key 的值解码后赋值给display，点亮不带译码器的数码管。

测试结果

数码管显示的数字与开关表示的二进制数完全相同。

提高要求

代码如下

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

entity kindle_life is
    port(
        display: out std_logic_vector(6 downto 0); -- without decoder
        display_4_odd: out std_logic_vector(3 downto 0); -- odd num with decoder
        display_4_even: out std_logic_vector(3 downto 0); -- even num with decoder
        clk: in std_logic; -- clock input
        rst: in std_logic -- reset input
    );
end kindle_life;

architecture bhv of kindle_life is
    signal bin_4_natural: std_logic_vector(3 downto 0):="0000"; -- binary natural num
    signal bin_4_odd: std_logic_vector(3 downto 0):="0001"; -- binary odd num
    signal bin_4_even: std_logic_vector(3 downto 0):="0000"; -- binary even num
begin
    process(clk)
    begin
```

```
        display_4_odd <= bin_4_odd; -- output without decoder
        display_4_even <= bin_4_even; -- output without decoder

        if(clk'event and clk = '1') then
            -- process natural
            if(bin_4_natural = "1001") then
                bin_4_natural <= "0000";
            else
                bin_4_natural <= bin_4_natural + 1;
            end if;
            -- process odd
            if(bin_4_odd = "1001") then
                bin_4_odd <= "0001";
            else
                bin_4_odd <= bin_4_odd + 2;
            end if;
            -- process even
            if(bin_4_even = "1000") then
                bin_4_even <= "0000";
            else
                bin_4_even <= bin_4_even + 2;
            end if;
        end if;
        if(rst='1') then -- reset pressed
            bin_4_natural <= "0000";
            bin_4_odd <= "0001";
            bin_4_even <= "0000";
        end if;
    end process;

    process(bin_4_natural)  -- decode natural num
    begin
        case bin_4_natural is
            when "0000"=> display<="1111110"; -- 0
            when "0001"=> display<="0110000"; -- 1
            when "0010"=> display<="1101101"; -- 2
            when "0011"=> display<="1111001"; -- 3
            when "0100"=> display<="0110011"; -- 4
            when "0101"=> display<="1011011"; -- 5
            when "0110"=> display<="1011111"; -- 6
            when "0111"=> display<="1110000"; -- 7
            when "1000"=> display<="1111111"; -- 8
            when "1001"=> display<="1111011"; -- 9
            when "1010"=> display<="1110111"; -- A
            when "1011"=> display<="0011111"; -- B
            when "1100"=> display<="1001110"; -- C
            when "1101"=> display<="0111101"; -- D
            when "1110"=> display<="1001111"; -- E
            when "1111"=> display<="1000111"; -- F
            when others=> display<="0000000"; -- else 0
        end case;
    end process;
end bhv;
```

工作原理

设置 2 个输入端口：CLK输入端口 clk，RST输入端口 rst。

设置 3 个输出端口：自然数列输出端口 display （不带译码器），奇数列输出端口 display_4_odd （带译码器），偶数列输出端口 display_4_even （带译码器）。

设置变量 bin_4_natural 记录自然数值，bin_4_odd 记录奇数值，bin_4_even 记录偶数值，并初始化。

当 CLK 按下时，变量 bin_4_natural 加一，bin_4_odd 和 bin_4_even 分别加二。并将 bin_4_natural 解码后赋值给 display，点亮自然数列，将 bin_4_odd 和 bin_4_even 直接赋值给 display_4_odd 和 display_4_even，点亮奇数列和偶数列。

当 RST 按下时，变量 bin_4_natural 和 bin_4_even 归零，变量 bin_4_odd 归一，并重置数码管。

测试结果

随着 CLK 不断按下，三个数码管分别显示自然数列、奇数列和偶数列。

当按下 RST 时，自然数列归零，奇数列归一，偶数列归零。

遇到的问题及解决方法

1. - Problem: 如何安装和破解 Quartus II 、如何通过 VHDL 语言开发、如何分配引脚、如何配置 USB-Blaster 驱动、如何烧写程序?
   - Solution: 通过上网查找资料解决。
2. - Problem: 如何调试 VHDL 语言?
   - Solution: 通过在可疑代码块中给数码管赋值，通过观察数码管输出调试程序。

实验小结

1. 这是我第一次接触可编程器件，遇到了不少问题，在互联网的帮助下成功解决，这极大的锻炼了我的自学能力，以及搜集资料的能力。
2. 通过本次实验，我感觉自己还有很多东西要学，今后要在硬件编程上更加努力。
3. 感谢老师和助教，我成功的点亮了我的数字人生。