计数器报告

项晨东 2019011831

实验目的

1. 掌握时序电路的基本分析和设计方法
2. 理解同步时序电路和异步时序电路的区别
3. 掌握计数器电路的设计原理, 用硬件描述语言实现指定功能的计数器设计
4. 学会利用软件仿真实现对数字电路的逻辑功能进行验证分析

实验内容

1. 使用实验平台上两个未经译码处理的数码管显示计数, 手动单次时钟进行计数, 时钟 上升沿计数一次, 当计数到 59 的时候, 要求两个数码管都能复位到 00 的状态, 重 新计数, 实验还要求设置一个复位键, 可以随时重新恢复到 00 的状态继续计数。
2. 使用实验平台上的 $1\mathrm{MHz}$ 时钟, 将计数器改成秒表, 在秒表中使用开关控制秒表启动和暂停。

实验代码

根据模块化设计, 我们将计数器设计为计数模块和现实模块, 其中计数模块为`count.vhd`,电路的具体工作原理在代码注释中。

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

entity count is
  port (
    clk, rst, pause: in std_logic;
    lown, highn: buffer std_logic_vector(3 downto 0)
    );
end count;

architecture arc of count is
  signal cnt: integer :=0;
begin
  process(clk, rst)
  begin
    if (rst = '1') then --在这里判断是否重置
      lown <= "0000";
      highn <= "0000";
      cnt <= 0;
```

```
  --在这里判断pause来判断是否进行暂停
    elsif (clk'event and clk = '1' and pause = '0') then
        if (cnt < 1000000) then --进行时钟的更新
          cnt <= cnt +1;
        else
          cnt <= 0;
        end if;

        if (cnt  = 0) then --一秒后
          if ( lown = "1001") then --低位到9后归零，并进位
            lown <= "0000";
            if ( highn = "0101") then --高位到5之后归零
              highn <= "0000";
            else
              highn <= highn +1;
            end if;
          else --个位数计数
            lown <= lown + 1;
          end if;
        end if;

    end if;
  end process;
end arc;
```

显示模块可以复用点亮数字人生中的代码, 电路的具体工作原理在代码注释中:

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

ENTITY digital_7 IS
  PORT (
    key : IN STD_LOGIC_VECTOR(3 DOWNTO 0); --控制开关
    display : OUT STD_LOGIC_VECTOR(6 DOWNTO 0) --不带译码器
  );
END digital_7;

ARCHITECTURE bhv OF digital_7 IS

BEGIN
  PROCESS (key) --不带译码器的需要进行译码处理
  BEGIN
    CASE key IS --以下是0/8/F 的编码规则
      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 <= "0011111"; --6
      WHEN"0111" => display <= "1110000"; --7
      WHEN"1000" => display <= "1111111"; --8
      WHEN"1001" => display <= "1110011"; --9
      WHEN"1010" => display <= "1110111"; --10
      WHEN"1011" => display <= "0011111"; --11
      WHEN"1100" => display <= "1001110"; --12
      WHEN"1101" => display <= "0111101"; --13
      WHEN"1110" => display <= "1001111"; --14
      WHEN"1111" => display <= "1000111"; --15
      WHEN OTHERS => display <= "0000000"; --其他情况全灭
    END CASE;
  END PROCESS;
END bhv;
```

最后`conter.vhd`将两个模块组合，通过引脚绑定来实现最后的功能。

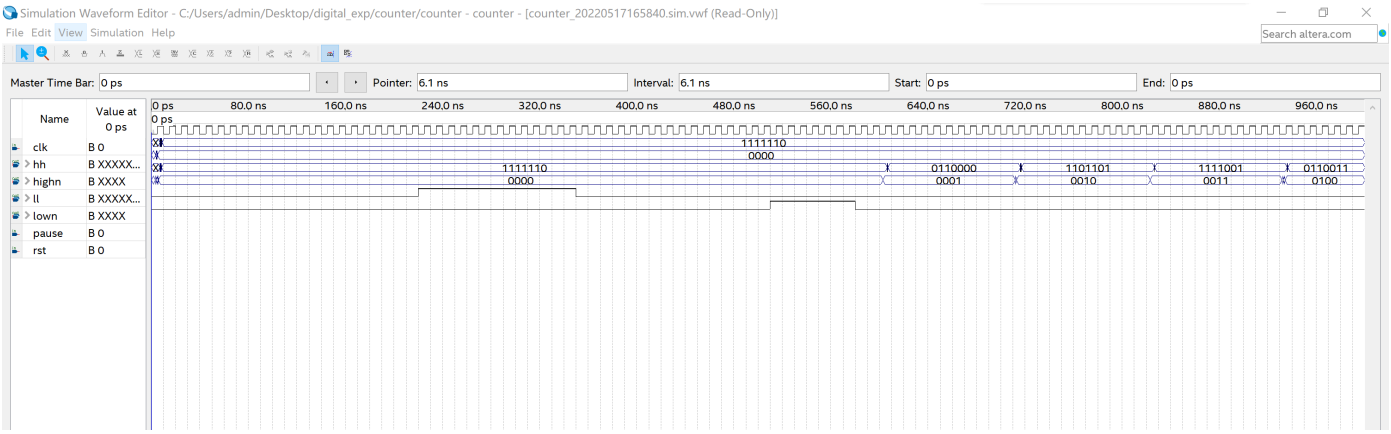
```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
entity counter is
  port(
    clk, rst, pause: in std_logic;
    lown, highn: buffer std_logic_vector(3 downto 0);
    ll, hh: out std_logic_vector(6 downto 0)
  );
end counter;
architecture  arc of counter is
  component count  --计数模块
    port(
      clk, rst, pause: in std_logic;
      lown,highn :buffer std_logic_vector(3 downto 0)
      );
  end component;

  component digital_7  --显示模块
    port(
      key: in std_logic_vector(3 downto 0);
      display: out std_logic_vector(6 downto 0)
      );
  end component;
begin --引脚绑定
  tmp0: count port map(clk=>clk, rst=>rst,pause=>pause, lown=>lown, highn=>highn);
  tmp1: digital_7 port map(key=>lown, display=>ll);
  tmp2: digital_7 port map(key=>highn, display=>hh);
```

```
end arc;
```

仿真部分

仿真结果如下, 这里的仿真结果有修改源代码`if (cnt < 1000000) then`为`if (cnt < 10) then`,不然会因为时间周期太长看不到波形的变化。仿真部分同时展示了归零和暂停的功能。



实验小结

通过本次实验, 我对CPLD的编写, 编译, 烧写的过程更加熟悉, 同时对一些bug的处理也更加熟练。

感谢老师和助教的辛苦付出~