计数器的设计

2017011620 计73 李家昊

实验目的

1. 掌握时序逻辑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和分析。

实验内容

基础内容

使用两个未经译码的数码管显示计数，时钟上升沿计数一次，当计数到59时，要求复位到00状态重新计数。还要求实现一个复位按键，可随时复位到00状态重新计数。

1. 写出计数器的源程序，并且编译通过。
2. 使用软件进行功能仿真。
3. 根据要求进行实验的引脚绑定，编译并下载。
4. 观察实验结果。

研究内容

1. 使用实验平台上的 1MHz 时钟，将计数器改成秒表。
2. 在秒表中使用开关控制秒表启动、暂停。

代码分析

D触发器

```
-- flip flop

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

entity ff is
    port(
        enable: in std_logic:='0';
        clk: in std_logic;
        rst: in std_logic;
        d: in std_logic:='0';
        q: out std_logic:='0'
    );
```

```
end ff;

architecture clock of ff is
begin
    process(clk, rst)
    begin
        if(enable = '1') then -- if enable
            if(clk'event and clk = '1') then -- if clock rising
                q <= d;
            end if;
            if(rst = '1') then -- reset
                q <= '0';
            end if;
        end if;
    end process;
end clock;
```

实现了一个带有使能端的正边沿D触发器，为四位计数器做准备。

4位二进制计数器

```
-- counter 4 bit

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

entity counter_4bit is
    port(
        enable: in std_logic;
        clk: in std_logic;
        rst: in std_logic;
        num: out std_logic_vector(3 downto 0) -- output 4-bit number
    );
end counter_4bit;

architecture count of counter_4bit is
    component ff
        port(
            enable: in std_logic;
            clk: in std_logic;
            rst: in std_logic;
            d: in std_logic;
            q: out std_logic
        );
    end component;
    signal sig_q: std_logic_vector(3 downto 0);
    signal sig_d: std_logic_vector(3 downto 0);

begin
    ff_0: ff port map(enable=>enable, clk=>clk, rst=>rst, d=>sig_d(0), q=>sig_q(0));
    ff_1: ff port map(enable=>enable, clk=>clk, rst=>rst, d=>sig_d(1), q=>sig_q(1));
```

```
    ff_2: ff port map(enable=>enable, clk=>clk, rst=>rst, d=>sig_d(2), q=>sig_q(2));
    ff_3: ff port map(enable=>enable, clk=>clk, rst=>rst, d=>sig_d(3), q=>sig_q(3));
    process(clk)
    begin
        sig_d(0) <= not sig_q(0);
        sig_d(1) <= sig_q(1) xor sig_q(0);
        sig_d(2) <= sig_q(2) xor (sig_q(1) and sig_q(0));
        sig_d(3) <= sig_q(3) xor (sig_q(2) and sig_q(1) and sig_q(0));
        num <= sig_q;
    end process;
end count;
```

实现了同步4位二进制计数器，用元件例化的方式，例化4个D触发器，根据激励函数：

$$
\begin{aligned}
D_0 &= \overline{Q_{0n}} \\
D_1 &= Q_{1n} \oplus Q_{0n} \\
D_2 &= Q_{2n} \oplus (Q_{1n}Q_{0n}) \\
D_3 &= Q_{3n} \oplus (Q_{2n}Q_{1n}Q_{0n})
\end{aligned}
$$

实现同步计数器。

译码模块

```
-- display raw

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


entity display_raw is
    port(
        buf_num: in std_logic_vector(3 downto 0);
        display_7: out std_logic_vector(6 downto 0)
    );
end display_raw;

architecture display of display_raw is
begin
    process(buf_num)
    begin
        case buf_num is
            when "0000"=> display_7 <="1111110"; -- 0
            when "0001"=> display_7 <="0110000"; -- 1
            when "0010"=> display_7 <="1101101"; -- 2
            when "0011"=> display_7 <="1111001"; -- 3
            when "0100"=> display_7 <="0110011"; -- 4
            when "0101"=> display_7 <="1011011"; -- 5
            when "0110"=> display_7 <="1011111"; -- 6
            when "0111"=> display_7 <="1110000"; -- 7
            when "1000"=> display_7 <="1111111"; -- 8
```

```
            when "1001"=> display_7 <="1111011"; -- 9
            when "1010"=> display_7 <="1110111"; -- A
            when "1011"=> display_7 <="0011111"; -- B
            when "1100"=> display_7 <="1001110"; -- C
            when "1101"=> display_7 <="0111101"; -- D
            when "1110"=> display_7 <="1001111"; -- E
            when "1111"=> display_7 <="1000111"; -- F
            when others=> display_7 <="0000000"; -- else 0
        end case;
    end process;
end display;
```

将“点亮数字人生”实验中的译码代码封装成一个结构，供十位和个位数的显示使用，避免写两份重复的代码。

计数器实体

```
-- counter

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: in std_logic;
        rst: in std_logic;
        pause: in std_logic;
        display_unit: out std_logic_vector(6 downto 0);
        display_ten: out std_logic_vector(6 downto 0);

        -- for simulation
        buf_0: buffer std_logic_vector(3 downto 0);
        buf_1: buffer std_logic_vector(3 downto 0)
    );
end counter;
```

定义了计数器实体，输入为时钟信号clk，重置信号rst，暂停信号pause，输出为不带译码器的数码管的信号。

自动计数器

```
architecture auto of counter is
    component counter_4bit
        port(
            enable: in std_logic;
            clk: in std_logic;
            rst: in std_logic;
            num: out std_logic_vector(3 downto 0)
        );
    end component;
```

```
    component display_raw
        port(
            buf_num: in std_logic_vector(3 downto 0);
            display_7: out std_logic_vector(6 downto 0)
        );
    end component;

    signal cnt: integer:=0;
    signal rst_0: std_logic:='0';
    signal rst_1: std_logic:='0';
    signal enable_0: std_logic:='0';
    signal enable_1: std_logic:='0';

begin
    counter_4bit_0: counter_4bit port map(enable=>enable_0, clk=>clk, rst=>rst_0,
num=>buf_0);
    counter_4bit_1: counter_4bit port map(enable=>enable_1, clk=>clk, rst=>rst_1,
num=>buf_1);
    display_raw_0: display_raw port map(buf_num=>buf_0, display_7=>display_unit);
    display_raw_1: display_raw port map(buf_num=>buf_1, display_7=>display_ten);

    process(clk)
    begin
        if(pause = '0') then -- not pause
            if(clk'event and clk = '1') then
                if(rst = '1') then -- reset
                    rst_0 <= '1';
                    rst_1 <= '1';
                else -- not reset
                    if(cnt < 1000000) then
                        cnt <= cnt + 1;
                        enable_0 <= '0';
                        enable_1 <= '0';
                    else -- add one
                        cnt <= 0;
                        enable_0 <= '1';
                        if(buf_0 = "1001") then -- carry
                            rst_0 <= '1';
                            enable_1 <= '1';
                            if(buf_1 = "0101") then -- reset
                                rst_1 <= '1';
                            else
                                rst_1 <= '0';
                            end if;
                        else
                            rst_0 <= '0';
                            enable_1 <= '0';
                        end if;
                    end if;
                end if;
            end if;
        end if;
    end process;
```

```
end auto;
```

实现自动计数器，例化两个4位二进制计数器，个位和十位分别计数。计数器接受时钟信号，每当时钟为上升沿时，内部计数加一，计数到1M时，计数清零，并将个位的使能端置为1，使个位加一，若此时个位为9，则令十位的使能端置为1，使其进位，并将个位重置，若十位为5，说明当前状态为59，则将个位和十位清零，重新计数。

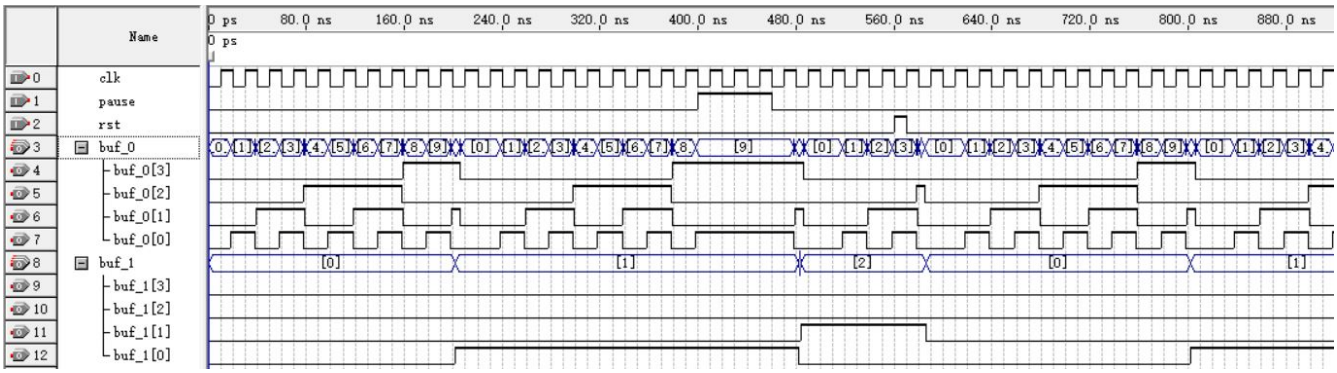
手动计数器

```
architecture manual of counter is
    component counter_4bit
        port(
            enable: in std_logic;
            clk: in std_logic;
            rst: in std_logic;
            num: out std_logic_vector(3 downto 0)
        );
    end component;

    component display_raw
        port(
            buf_num: in std_logic_vector(3 downto 0);
            display_7: out std_logic_vector(6 downto 0)
        );
    end component;

    signal rst_0: std_logic:='0';
    signal rst_1: std_logic:='0';
    signal carry: std_logic;
    signal enable_0: std_logic;
    signal enable_1: std_logic;

begin
    counter_4bit_0: counter_4bit port map(enable=>enable_0, clk=>clk, rst=>rst_0,
num=>buf_0);
    counter_4bit_1: counter_4bit port map(enable=>enable_1, clk=>carry, rst=>rst_1,
num=>buf_1);
    display_raw_0: display_raw port map(buf_num=>buf_0, display_7=>display_unit);
    display_raw_1: display_raw port map(buf_num=>buf_1, display_7=>display_ten);

    process(clk)
    begin
        if(pause = '0') then -- not pause
            enable_0 <= '1';
            enable_1 <= '1';
            if(clk'event and clk = '1') then
                if(rst = '1') then -- reset
                    rst_0 <= '1';
                    rst_1 <= '1';
                else -- not reset
                    if(buf_0 = "1001") then -- carry
                        rst_0 <= '1';
                        carry <= '1';
```

```
                        if(buf_1 = "0101") then -- reset
                            rst_1 <= '1';
                        else
                            rst_1 <= '0';
                        end if;
                    else
                        carry <= '0';
                        rst_0 <= '0';
                        rst_1 <= '0';
                    end if;
                end if;
            end if;
        else
            enable_0 <= '0';
            enable_1 <= '0';
        end if;
    end process;
end manual;
```

实现手动计数器，个位的时钟接受输入信号，十位的时钟接受进位信号，进位逻辑同自动计数器，这里不再赘述。

仿真结果



由图看出：正常工作时，计数器在每一个时钟上升沿处加一，当暂停信号为1时，计数暂停，当重置信号为1时，计数器重置，并重新计数。

功能测试结果

自动计数器：时钟信号接到1MHz输入上，每隔1秒计数加一，当计数到59后重置，重新计数，符合仿真结果。

手动计数器：当按下微动开关时，计数加一，当计数到59后重置，重新计数，符合仿真结果。

实验小结

1. 通过本次实验，我实现了异步计数器和同步计数器，对计数器以及同步时序逻辑有了更深刻的理解。
2. 本次CPLD实验中，我三次用到了元件例化的方法（D触发器，四位二进制计数器，译码模块），深刻体会到模块化设计的优点，模块化既方便了调试，还可以进行代码复用。
3. 最初我通过修改时钟实现异步计数电路，后来在助教的提醒下，我通过在D触发器内添加使能端实现了同步计数器，使得电路在高频下更加兼容。感谢助教的耐心指导！