开放实验 最大公约数计算器

2018011365 张鹤潇

不同意将本工程代码开源。

实验目的

用VHDL实现简单数值算法，体会它与通用编程语言(C/Python)的区别。

实验任务

实现一个四位最大公约数计算器。每次输入一个四位二进制数，输出它与之前所有输入的数的最大公约数。

实验原理

代码的关键部分是最大公约数的求解，使用辗转相减算法。伪代码如下：

```
while a != b:
    if a > b:
        a -= b
    else:
        b -= a
gcd = a
```

考虑到数据范围很小，用VHDL简单实现如下。实现中固定了循环的次数。

```
    variable tmp_X, tmp_Y: std_logic_vector(3 downto 0);
    -- do something
    tmp_X := X;
    tmp_Y := gcd_prev;
    for i in 0 to 15 loop  -- 辗转相减法
        if (tmp_X /= tmp_Y) then
            if (tmp_X < tmp_Y) then
                tmp_Y := tmp_Y - tmp_X;
            else
                tmp_X := tmp_X - tmp_Y;
            end if;
        else
            gcd_prev <= tmp_X;
        end if;
    end loop;
```

实现方法

用实验平台上两个带译码的七段数码管表示输入和输出4位二进制数。

设计过程

在最大公约数求解算法的基础上增加逻辑判断。完整实验代码如下：

```
entity gcd is
port(
    clk, rst: in std_logic;  -- clk, 复位
    X:  in std_logic_vector(3 downto 0);  -- 输入
    Z:  out std_logic_vector(3 downto 0) -- 输出
);
end gcd;

architecture behv of gcd is
signal mark: std_logic := '0';  -- 判断是否是第一个输入
signal gcd_prev: std_logic_vector(3 downto 0):= "0000";  -- 记录之前所有数的GCD
begin
    process(clk, rst)
    variable tmp_X, tmp_Y: std_logic_vector(3 downto 0);
    begin
    if (rst = '1') then -- 复位
        mark <= '0';
        gcd_prev <= "0000";
    elsif (clk'event and clk = '1') then  -- 输入
        if (mark = '0') then -- 第一个数字
            mark <= '1';
            gcd_prev <= X;
        else -- 计算新输入数字和之前数字的GCD
            tmp_X := X;
            tmp_Y := gcd_prev;
            for i in 0 to 15 loop -- 辗转相减法，保证循环次数确定
                if (tmp_X /= tmp_Y) then
                    if (tmp_X < tmp_Y) then
                        tmp_Y := tmp_Y - tmp_X;
                    else
                        tmp_X := tmp_X - tmp_Y;
                    end if;
                else -- 计算结束，更新输出
                    gcd_prev <= tmp_X;
                end if;
            end loop;
        end if;
    end if;
    end process;

    process(gcd_prev)
    begin -- 更新输出
        Z <= gcd_prev;
    end process;
end;
```

实验结果

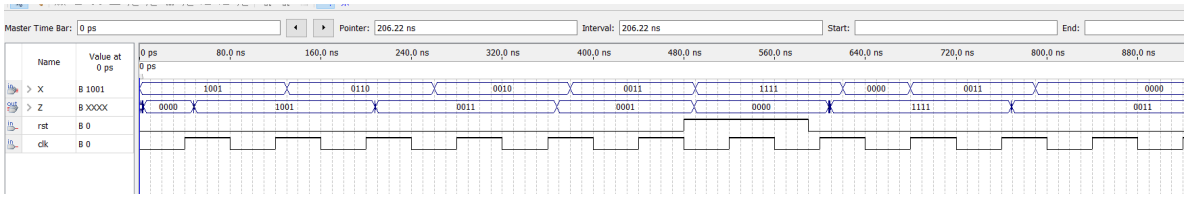
管脚分配

信号	管脚	说明
clk	pin12	输入，对应计算按钮
rst	pin44	输入复位，对应开关RST
X[0]-X[3]	pin1-pin4	输入
Z[0]-Z[3]	pin5-pin8	输出

功能仿真

演示视频

仿真结果如图，可以验证程序的正确性。

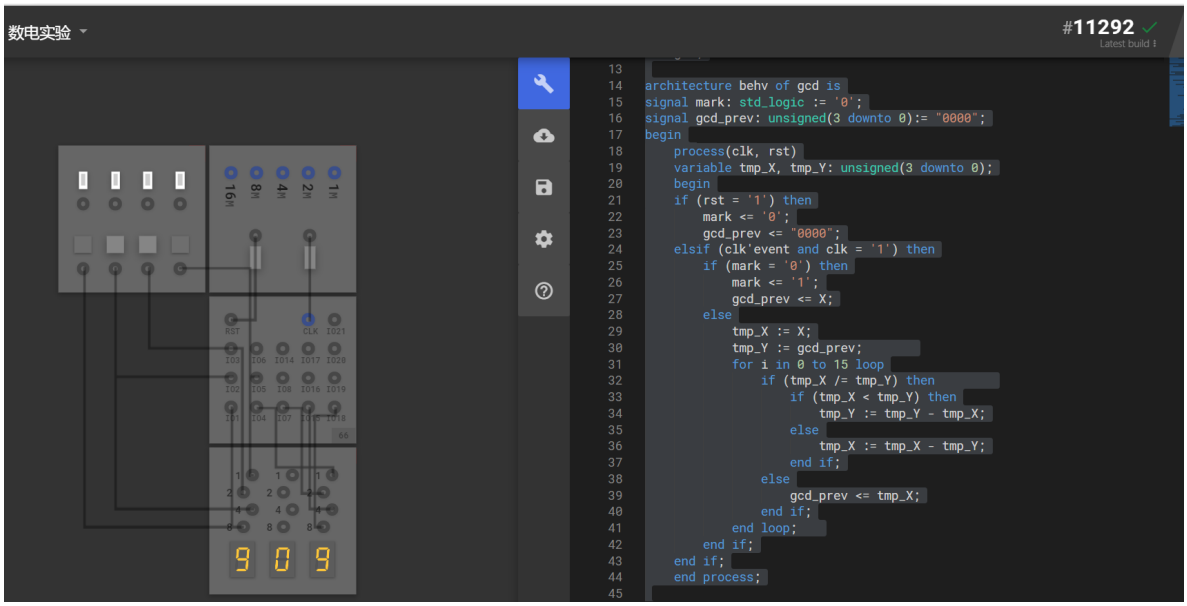


先输入1001=9，程序输出9，再输入6，程序输出3；再输入2，程序输出1；然后将rst置1，输出复位；再输入15，程序输出15；再在clk的上升沿输入3，程序输出3.

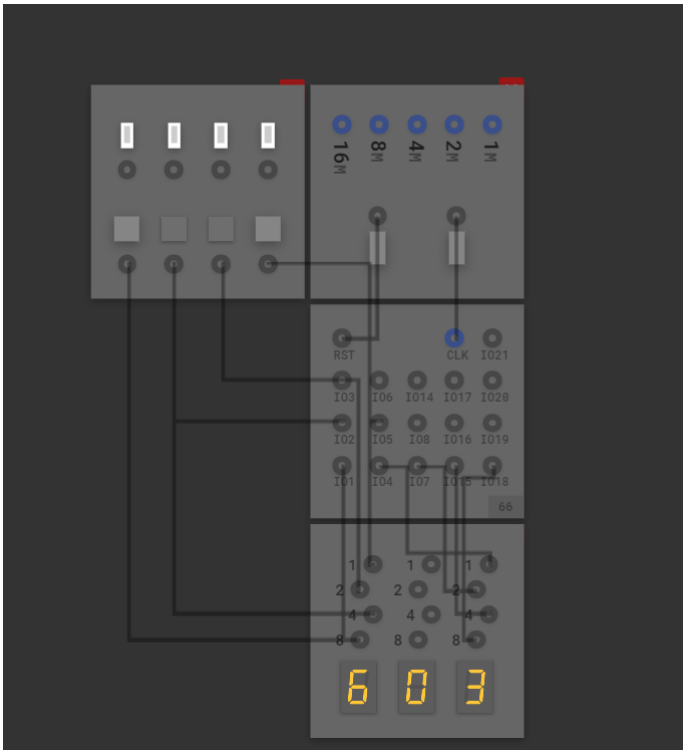
实验操作

演示视频

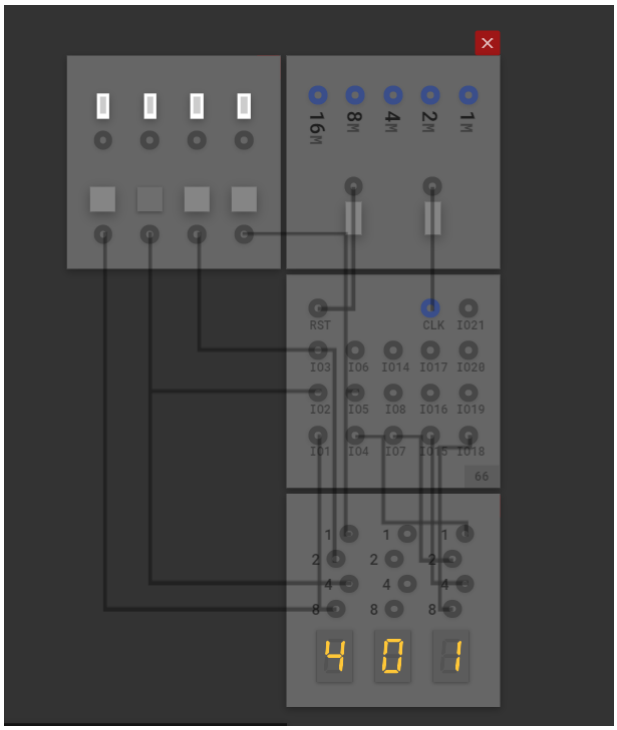
在Jielab平台实验如图，先输入第一个数字$1001=9$，按下clk，输出端也显示9；



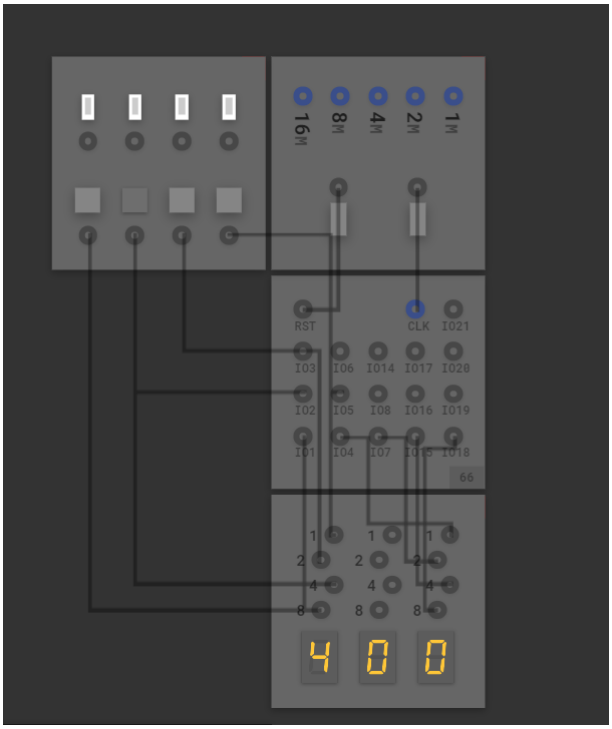
再输入第二个数字$0110=6$，按下clk，输出端显示$\gcd(9,6)=3$；

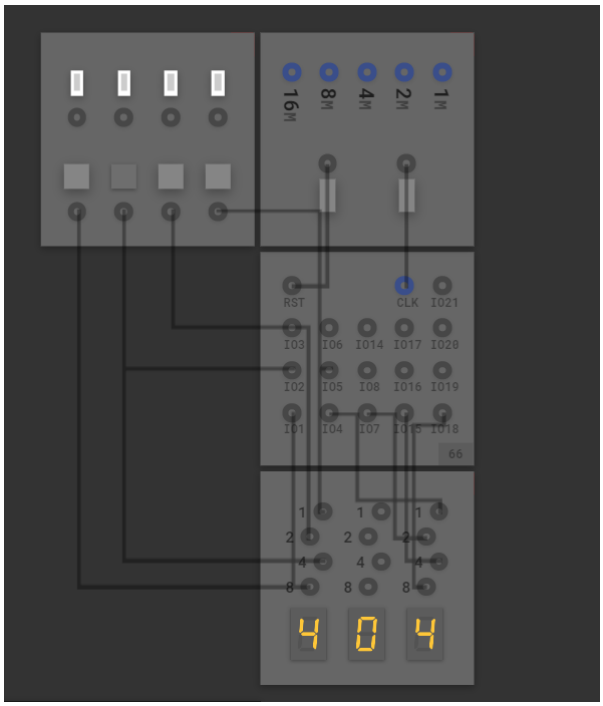


输入第三个数字4，按下clk，显示$\gcd(9,6,4)=1$.



保持输入为4，按下rst，输出端变为0；再次按下clk，输出端显示4.





总结反思

本次实验是我最后一次数电实验，最大公约数这个题目虽然简单，但也加深了我对VHDL和通用编程语言区别的理解。求多个数的最大公约数，每次输出的结果既与输入有关，也与现态有关，可以说是结合了组合逻辑与时序逻辑。因为功能简单，所以没有结合元件例化进行设计。

实验中，因为EPM240的逻辑单元不足，不得已使用了有更多逻辑单元的板子，针对这个问题，目前我还没有找到解决办法。

感谢助教和老师，本学期的数电实验令我受益匪浅。