

中国科学技术大学计算机学院  
《数字电路实验》报告



实验题目： 简单时序逻辑电路

学生姓名： 黄瑞轩

学生学号： PB20111686

完成日期： 2021. 10. 24

计算机实验教学中心制

2020 年 09 月

## 实验题目

简单时序逻辑电路

## 实验目的

- 掌握时序逻辑相关器件的原理及底层结构
- 能够用基本逻辑门搭建各类时序逻辑器件
- 能够使用 Verilog HDL 设计简单逻辑电路

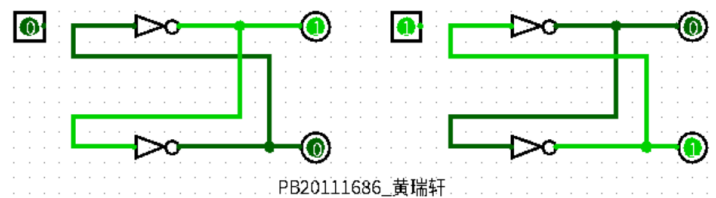
## 实验环境

- PC 一台：安装了 Linux 操作系统的一台虚拟 PC
- VLAB 实验中心平台（vlab.ustc.edu.cn）
- Logisim Version 2.7.1

## 实验过程

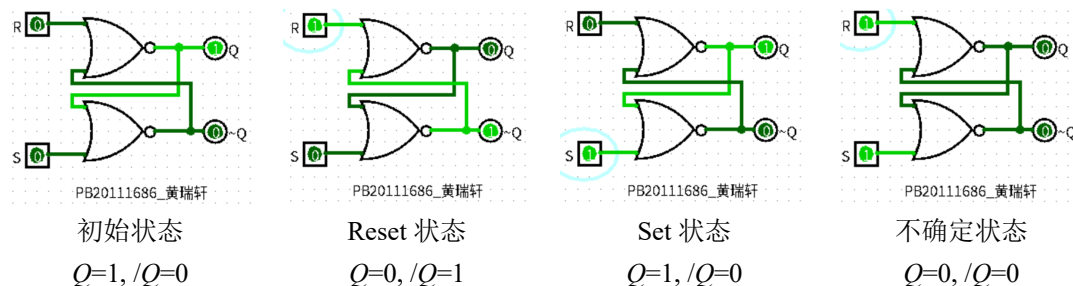
### 1. 搭建双稳态电路

- 双稳态电路由两个非门交叉耦合构成，是一种最简单的时序逻辑电路，是所有时序逻辑电路的基础。
- 在 Logisim 中搭建电路时应先将两条交叉耦合线断开一条，等输入信号将其状态初始到确定状态后再将耦合线连上。否则电路将处不确定状态。



### 2. 搭建 SR 锁存器

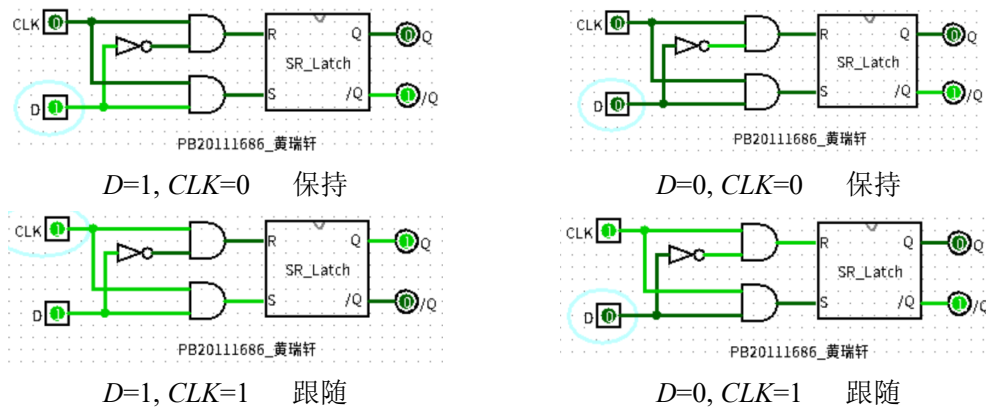
- SR 锁存器的功能已在理论课上说明，这里不多记录。
- 用 Logisim 搭建电路，并改变输入输出，观察电路仿真情况是否与理论一致。
- 理论分析得出 R 和 S 同时置 1 时电路会处于不确定状态，因此要避免这种情况。



- 将此电路封装为“SR\_Latch”以便复用。

### 3. 搭建 D 锁存器

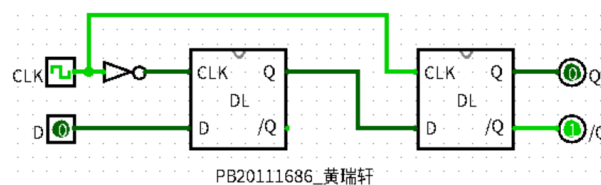
- SR 锁存器两个输入都为 1 是一种未定义状态，为了避免这种情况出现设计了下面这种 D 锁存器。D 锁存器的理论分析也已在理论课上说明，这里不多记录。
- 用 Logisim 搭建电路，并改变输入输出，观察电路仿真情况是否与理论一致。



- 将此电路封装为“DL”以便复用。

### 4. 搭建 D 触发器

- 用 D 锁存器搭建 D 触发器的电路如下图所示。



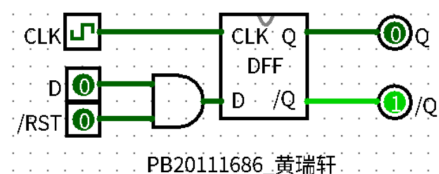
- Logisim 菜单栏—“simulation”—“Tick Frequency” 设置为“1Hz”，然后 Enable 仿真和触发功能，在“CLK”信号以 1Hz 频率跳变过程中，改变 D 信号的输入值，观察 Q 信号的输出。
- 其 Verilog 代码如下所示，“posedge CLK”表示“CLK 信号的上升沿”这一事件。

```

module DFF(input clk, d, output reg q);
    always@(posedge clk) q <= d;
endmodule

```

- [同步复位] 添加复位信号/RST，当/RST 有效时，Q 始终为 0。这种复位信号只有在时钟信号的上升沿才起作用，在非上升沿时刻不起作用。其电路和 Verilog 代码如下所示。

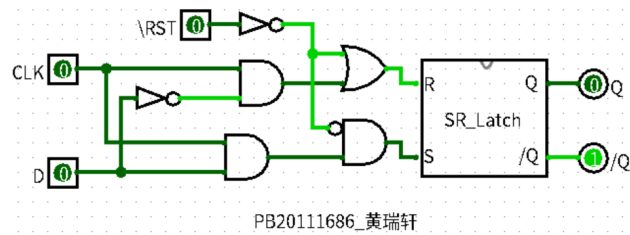


```

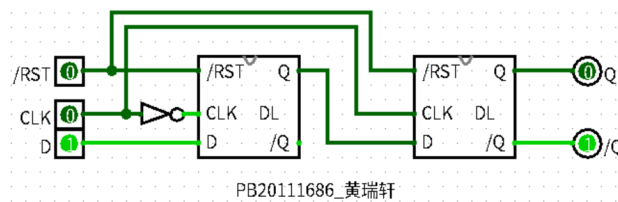
module DFF_R(input clk, d, rst, output reg q);
    always@(posedge clk)
    begin
        if(rst == 0) q <= 1'b0;
        else q <= d;
    end
endmodule

```

- [异步复位] 添加复位信号/RST，当/RST 变为有效时(负电平)，Q 立即变为 0，无论 CLK 和 D 如何。
- 首先，制作一个/RST 负有效的 D 锁存器，电路图如下。



- 再用相似的结构做 D 触发器，电路图和 Verilog 代码如下。



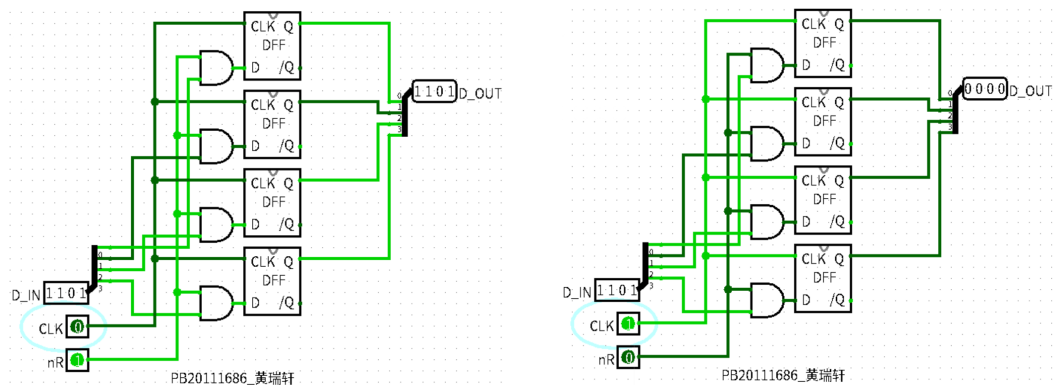
```

module DFF_R(input clk, d, r, output reg q);
    always@(posedge clk or negedge r)
    begin
        if(r == 0) q <= 1'b0;
        else q <= d;
    end
endmodule

```

## 5. 搭建寄存器


- 用 4 个 D 触发器组成一个能够存储 4bit 数据的寄存器，带有低电平有效的同步复位信号，其电路结构和 Verilog 代码如下所示，可以看出寄存器本质上就是 D 触发器。



```
module REG4(input clk, [3:0] d_in, rst, output reg [3:0] q);
    always@(posedge clk)
    begin
        if(rst == 0) q <= 4'b0;
        else q <= d;
    end
endmodule
```

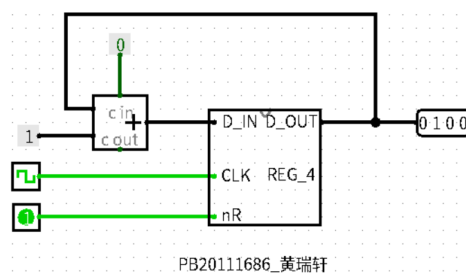
- 有些情况下信号的复位值需要为 1，如将上述代码复位部分改成

```
if(rst == 0) q <= 4'b0011;
```

后电路应将低两位的 D 触发器的 D 部分改成 ，此时如果 nR 有效，D 将成为 1，电路稳定后相应输出位将为 1。如果 nR 无效，D 仍是 D\_IN 中的相应位，这是一种同步复位方法。

## 6. 搭建简单时序逻辑电路

- 利用 4 bit 寄存器，搭建一个 4 bit 的计数器，该计数器在 0~15 之间循环计数，复位时输出值为 0，其电路结构和 Verilog 代码如下所示。

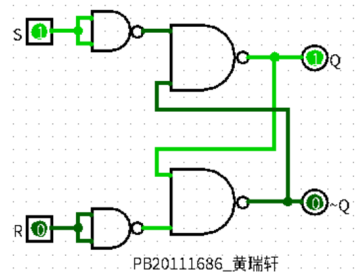


```
module CTR(input clk, rst, output reg [3:0] cnt);
    always@(posedge clk)
    begin
        if(rst == 0) cnt <= 4'b0;
        else cnt <= cnt + 4'b1;
    end
endmodule
```

## 实验练习

### 【题目 1】

- 在 Logisim 中用与非门搭建 SR 锁存器，其电路图如下所示。



- 写出逻辑表达式： $Q = \overline{\sim Q \cdot \bar{S}}$ ,  $\sim Q = \overline{Q \cdot \bar{R}}$ 。
- 当  $S = 0, R = 0$  时，电路将保持原来的状态；  
当  $S = 1, R = 0$  时， $Q$  将先变成 1，随后  $\sim Q$  随其变成 0（置 1）；  
当  $S = 0, R = 1$  时， $\sim Q$  将先变成 1，随后  $Q$  随其变成 0（置 0）；  
当  $S = 1, R = 1$  时，电路下一个状态不确定。

以表列出如下，这与普通 SR 锁存器的功能表相同。

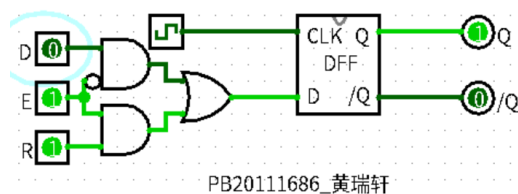
$S = 0, R = 0$	$S = 1, R = 0$	$S = 0, R = 1$	$S = 1, R = 1$
保持	置 1	置 0	不确定

### 【题目 2】

- 设计同步置位功能的 D 触发器只需要修改 D 的逻辑。新增输入端 E 和 R，当 E 为 0 时置位功能无效，当 E 为 1 时启动置位功能，D 输入端将输入 R。写成逻辑式就是：

$$Q^{n+1} = D\bar{E} + RE$$

设计成电路图如下：

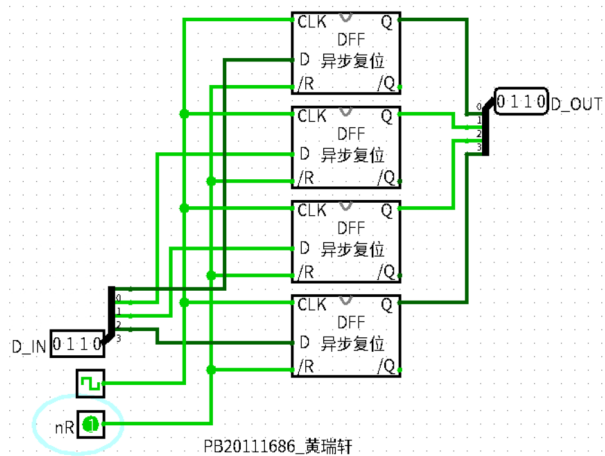


其 Verilog 代码如下：

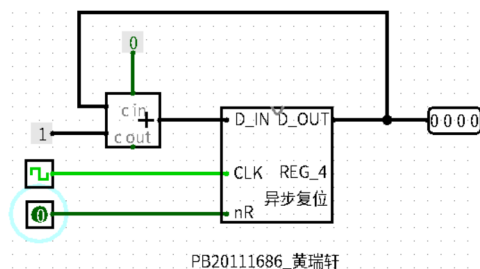
```
module DFF_S(input clk, d, e, r, output reg q);
    always@(posedge clk)
    begin
        if(e == 1'b1) q <= r;
        else q <= d;
    end
endmodule
```

### 【题目 3】

- 电路设计和 Verilog 代码已在实验过程(4.)[异步复位]中体现，这里不再重复，仅完成后半部分。先将寄存器中所有的触发器换成异步复位的，如下图所示。



- 计数器的电路与 Verilog 代码如下：



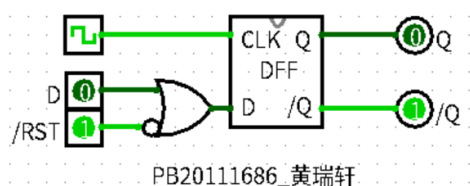
```

module CTR_R(input clk, rst, output reg [3:0] cnt);
    always@(posedge clk or negedge r)
    begin
        if(rst == 0) cnt <= 4'b0;
        else cnt <= cnt + 4'b1;
    end
endmodule

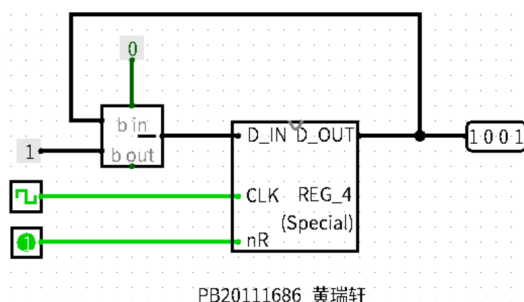
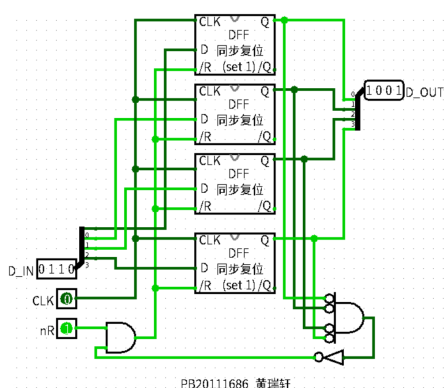
```

#### 【题目 4】

- 首先做一个低电平有效且复位置 1 的触发器，电路图如下：

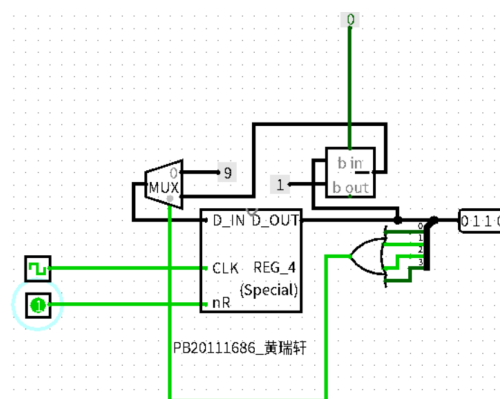
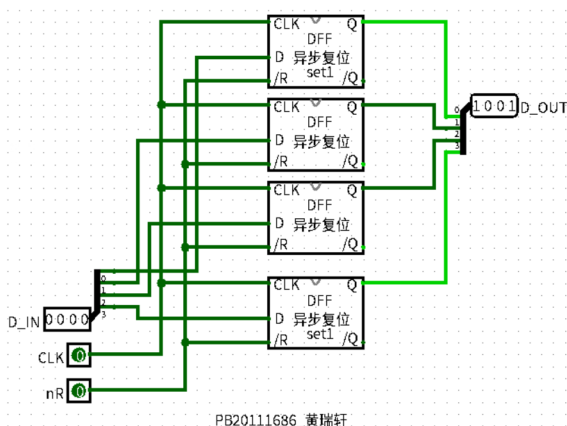


然后用此触发器实现 4 位寄存器。由于需要的数据范围是 0000 ~ 1001，所以只要复位信号 nR 为 0 或者当前状态是 0000 时，下一个状态就要复位，寄存器设计如下左图所示，最终计数器电路如下右图所示，其 Verilog 代码附后。



```
module MNR(input clk, rst, input [3:0] d, output reg [3:0] q);
    always@(posedge clk)
        begin
            if(rst == 0 || d == 4'b0000) q <= 4'b1001;
            else q <= q - 4'b1;
        end
    endmodule
```

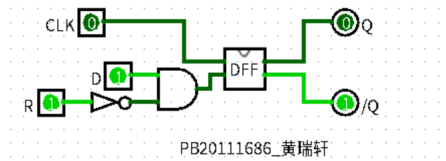
如果要求异步复位，就不能通过当前状态来锁住 nR，这样会导致计数器卡在 1001 的状态。可以通过在实现计数器时加一个数据选择器来完成，当前状态为 0000 时给 D\_IN 传入 0x9，其他时刻传入  $q - 4'b1$ 。这种情况的寄存器电路如下左图所示，计数器电路如下右图所示。（因为 Logisim 的数据选择器不能改变方向，导致电路图些许不美观）





### 【题目 5】

- 以之前的同步复位 D 触发器为例，将复位信号后加一个非门即可，电路图如下：



Verilog 代码为:

```
module DFF_R(input clk, d, r, output reg q);  
    always@(posedge clk)  
    begin  
        if(r == 1) q <= 1'b0;  
        else q <= d;  
    end  
endmodule
```

## 总结与思考

- 本次实验中我了解到了在仿真情况下如何搭建时序逻辑电路的方法，并且搭建了锁存器、触发器和寄存器等各种常用的元件。我对时序逻辑电路的理解有了更深刻的认识。
- 本次实验的难度比之前稍有增加，主要在于时序逻辑电路本身比组合逻辑电路要更复杂，实现起来的思维容量比较大，但是经过思考我仍完成了所有练习题。
- 本次实验的任务量较大，主要在实验过程部分，但是这些都是时序逻辑电路的基础。关于任务量的改进建议写在后面。
- 改进建议：
  - ① 理论课的进度还未讲到寄存器，建议将实验部分的进度和理论部分进行统一。
  - ② 题目 5 适合作为思考题，让学生简要回答一下即可，不需要给出示例和代码。