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：王章瀚

学生学号：PB18111697

完成日期：2019/10/20

计算机实验教学中心制

2019 年 09 月

1 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

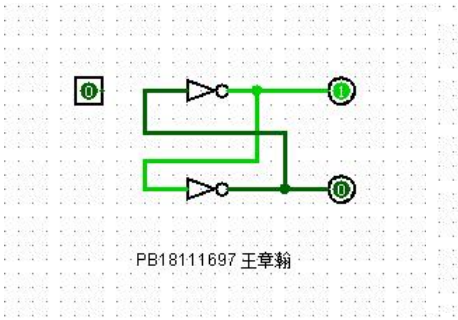
2 实验环境

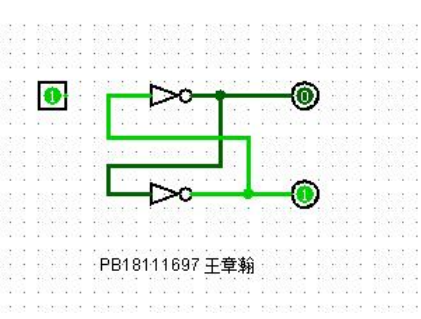
PC 一台
Windows 或 Linux 操作系统
Java 运行环境（jre）
Logisim 仿真工具
vlab.ustc.edu.cn (jre 和 Logisim 工具都可在此网站获取)

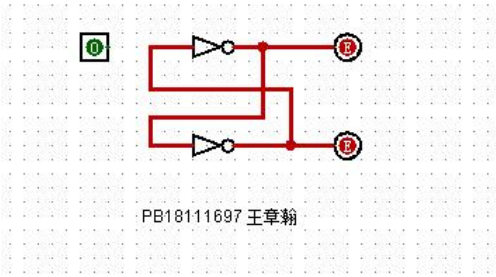
3 实验过程

3.1 搭建双稳态电路

此处值得注意的是，在 Logisim 中搭建此电路时，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。否则电路将处于一种不确定状态。

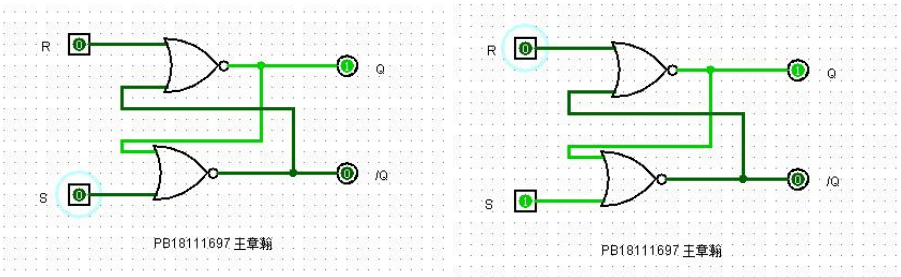


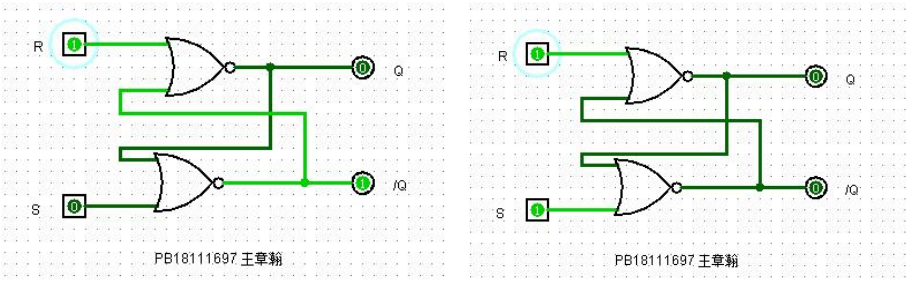




3.2 搭建 SR 锁存器

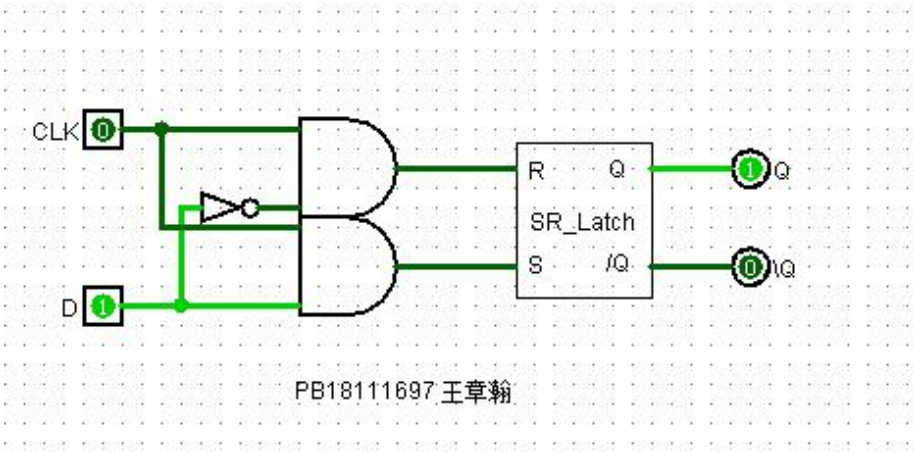
由于刚才搭建的双稳态电路没有输入信号，无法进行操作。故将两非门用或非门代替。输入信号为 S 和 R，输出信号为 Q 和/Q(用以表示 $\overline{Q}$)。如下四图所示：





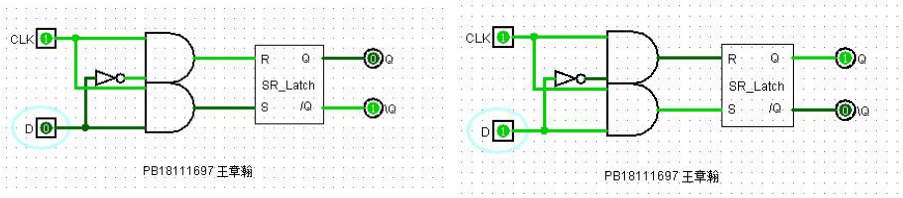
3.3 搭建 D 锁存器

前面的 SR 锁存器中，两个输入为 1 是一种未定义状态，我们不希望它出现。为此在 SR 锁存器前面添加两个与门和一个非门。如下图：

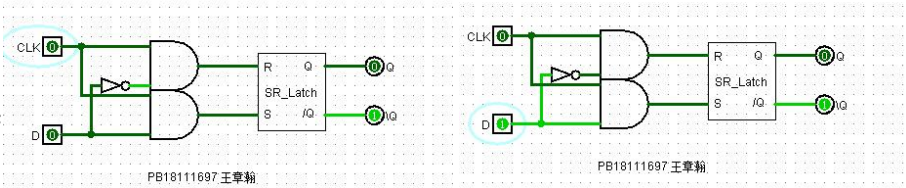


只有 CLK 为高电平时，锁存器的值才会随 D 变化，如下图:

跟随：

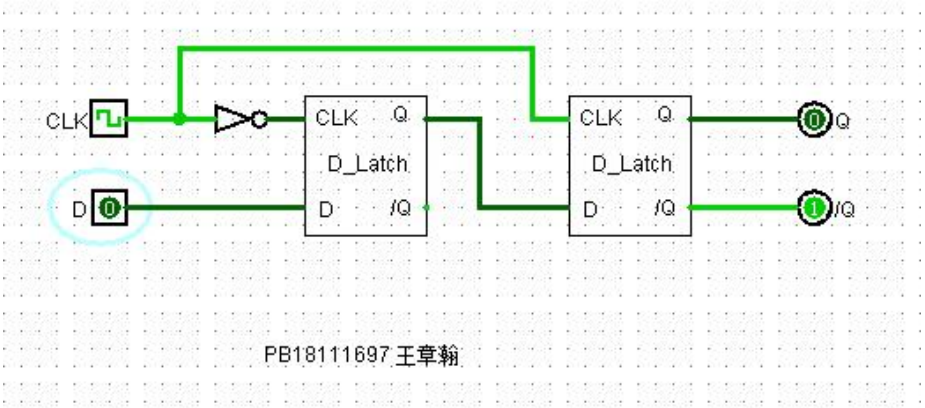


锁存：



3.4 搭建 D 触发器

如果我们将两个 D 锁存器串起来，其控制信号有效值始终相反，实际上这就构成了触发器。



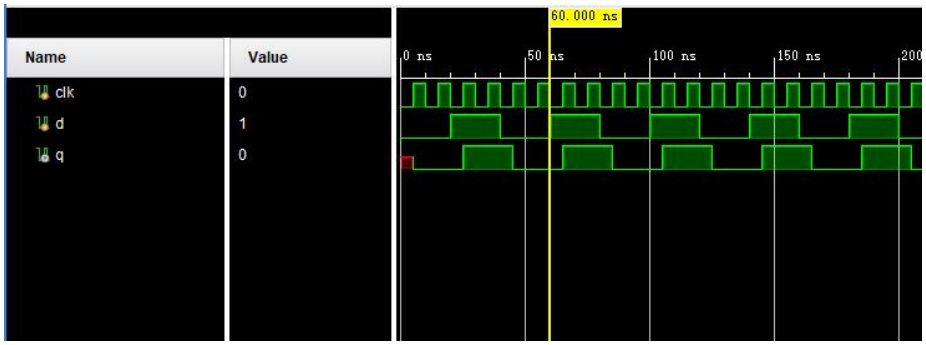
利用 Logisim 的仿真功能，设置频率为 1HZ，通过分析我们可以发现，只有在 CLK 信号由低电平变为高电平的瞬间，D 信号才会传播到 Q 端，其余时刻 Q 端的值都保持不变。

写出该过程的代码，

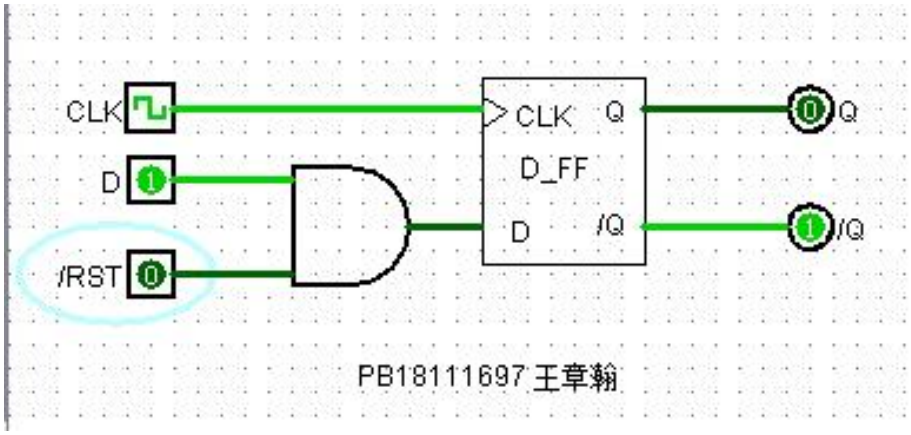
```
1  module D_FF(
2      input  clk ,
```

```
3           input d,
4           output reg q
5           );
6           always @(posedge clk)
7               q <= d;
8       endmodule
9
```

利用 Vivado 的仿真得到波形图，如下：



此外，我们还可以添加复位信号，当复位信号有效（低电平邮箱）时，输出信号 Q 始终为 0，如下图：

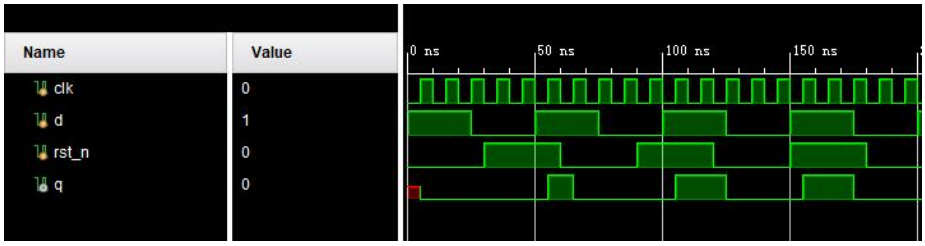


带复位信号的电路代码如下,

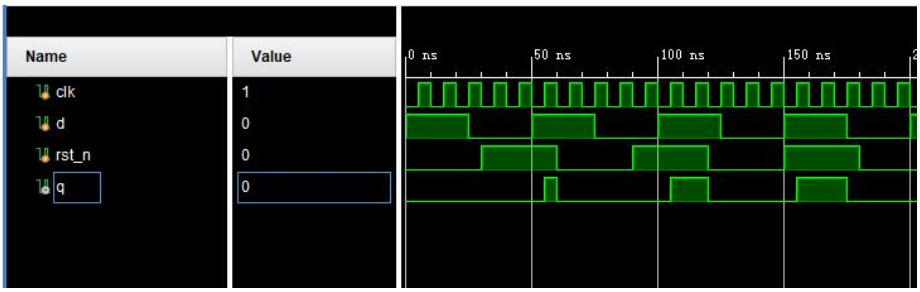
```
module D_FF_RST(
    input clk,
    input d,
    input rst_n,
    output reg q
    );
    always@(posedge clk)
    begin
        if(rst_n==0)
            q <= 1'b0;
        else
            q <= d;
    end
endmodule

```

波形图如下，



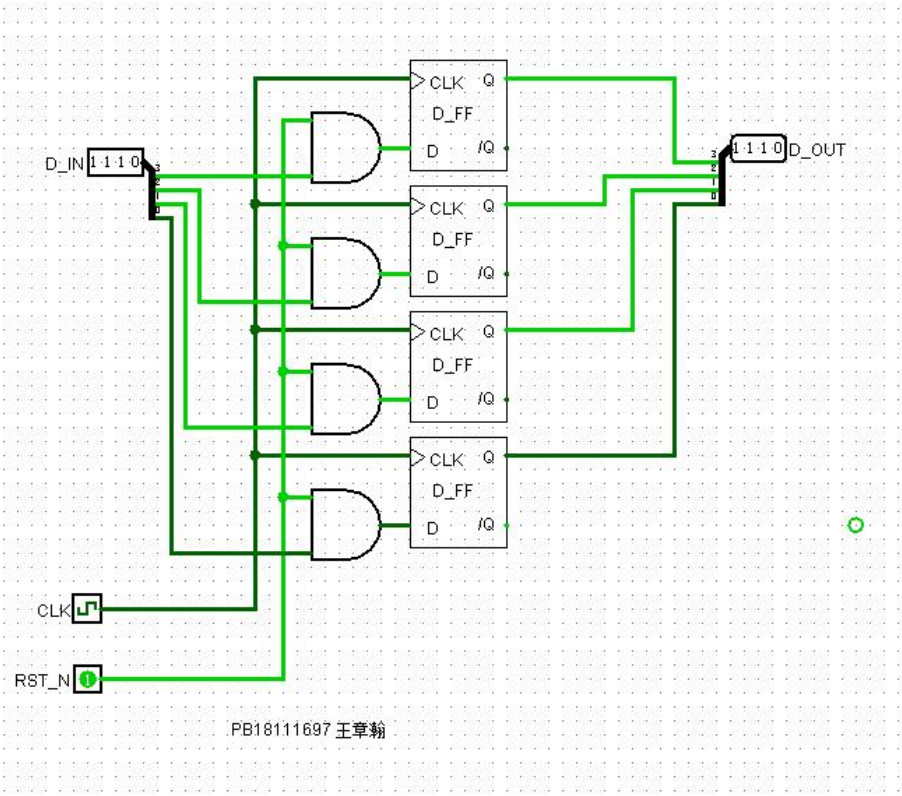
上述这种触发器的复位信号只有在时钟信号的上升沿才起作用，在非上升沿时刻，复位信号不起作用。这种复位方式称为同步复位。另外还有一种异步复位的方式，可以使得不论时钟和 D 信号如何，一旦复位信号有效，输出端 Q 立即变为确定的复位值。而异步复位与同步复位最大的区别在于，复位信号与时钟信号同时出现在了 always 语句的敏感变量列表中。其波形图如下：



D 触发器是一种边沿敏感器件，因此由 D 触发器为核心构成的电路一般称为同步时序逻辑电路，而锁存器构成的一般都是异步时序电路。

3.5 搭建寄存器

寄存器本质上就是触发器，如下图，用 4 个 D 触发器构成一个储存 4bit 数据的寄存器，并有低电平有有效的复位信号。



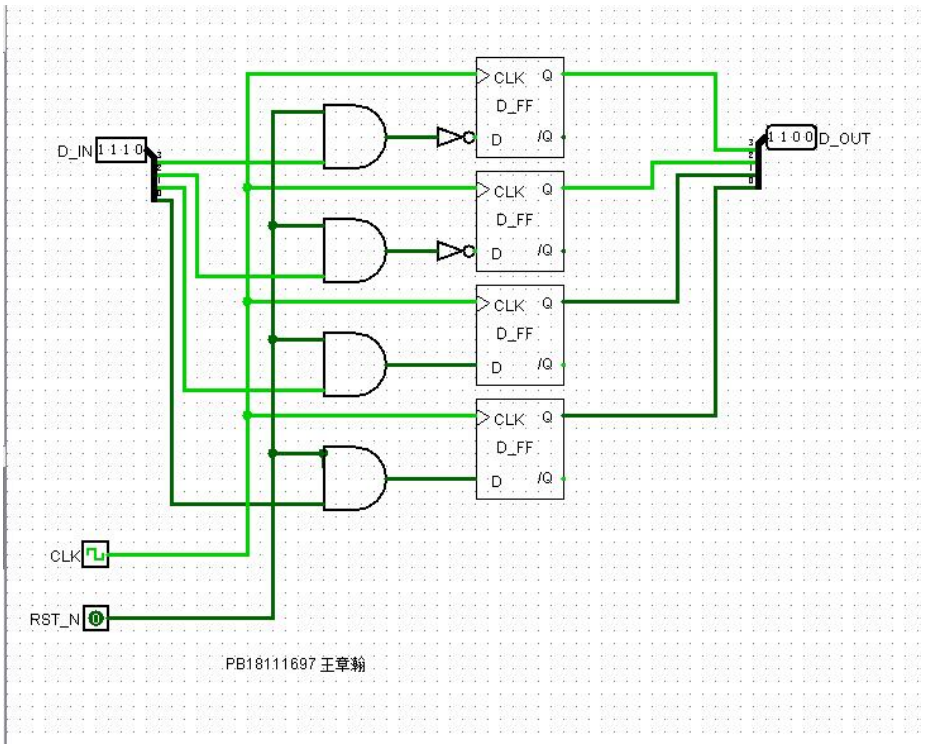
该寄存器的行为表现可以总结如下：若复位信号有效，则在时钟上升沿时，输出全部复位为 0，否则在上升沿时，更新为输入 D_IN 的值。

其 Verilog 代码如下：

```
module REG4(
    input CLK,
    input RST_N,
    input [3:0] D_IN,
    output reg [3:0] D_OUT
    );
    always @(posedge CLK)
    begin
        if(RST_N == 0)
            D_OUT <= 4'b0;
        else
            D_OUT <= D_IN;
    end
endmodule
```

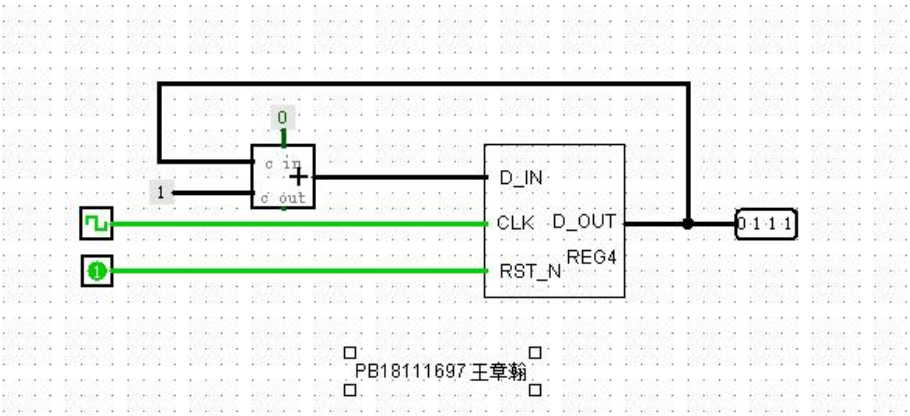
15

如要使复位信号将输出置为 0011，其 verilog 代码实现显然。电路图实现可以如下：



3.6 搭建简单时序逻辑电路

我们利用 4bit 寄存器，搭建一个 4bit 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，电路图如下所示：

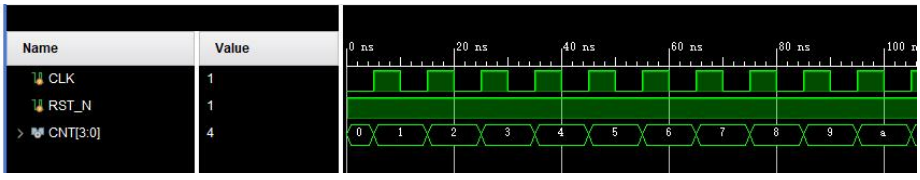


其 Verilog 代码如下：

```
module counter(
    input CLK, RST_N,
    output reg [3:0] CNT = 0
    );
    always @(posedge CLK)
    begin
        if(RST_N == 0)
            CNT <= 4'b0;
        else
            CNT <= CNT + 4'b1;
    end
endmodule

```

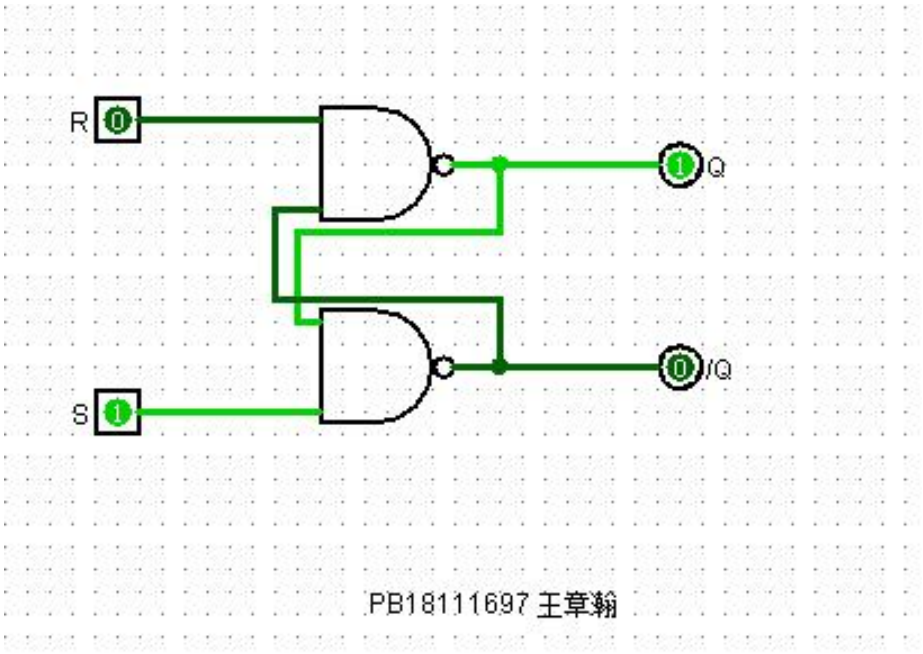
通过 Vivado 仿真，得到如下波形图：



4 实验练习

4.1 题目 1

在 Logisim 中用与非门搭建 SR 锁存器，画出电路图，并分析其行为特性，列出电路在不同输入时的状态。



该电路的行为特性描述如下：当 S,R=1,0 时，Q 置 1；当 S,R=0,1 时，Q 置 0；当 S,R=1,1 时，Q 保持不变；应当避免 S,R=0,0，这是未定义状态。

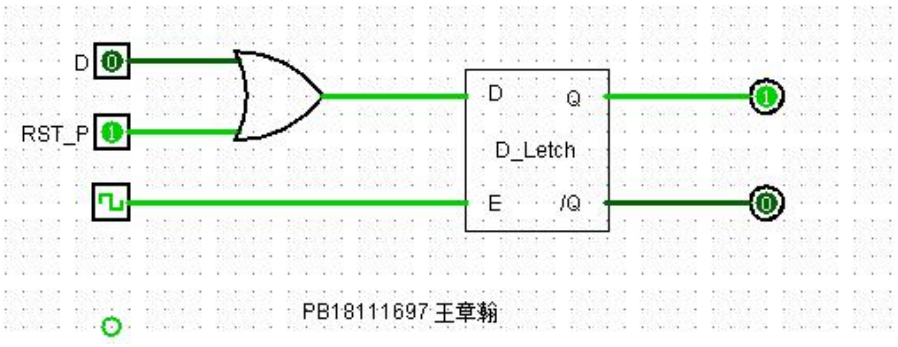
电路在不同输入时的状态如下表所示：

S	R	Q	$\overline{Q}$	功能
0	0	1	1	未定义状态
0	1	0	1	置 0
1	0	1	0	置 1
1	1	不变	不变	保持

4.2 题目 2

在 Logisim 中搭建一个支持同步置位功能的 D 触发器，画出其电路图，并编写对应的 Verilog 代码。

搭建结果的电路图如下图所示：



其对应的 Verilog 代码如下：

```
module D_FF_SYN_RST1(
    input D, CLK, RST_P,
    output reg q
    );

    always @(posedge CLK)
    begin
        if(RST_P == 1)
            q <= 1'b1;
        else
            q <= D;
    end
endmodule

```

4.3 题目 3

在 Logisim 中搭建一个带有异步复位功能的 D 触发器，画出其完整电路图，并进一步调用该触发器设计一个从 0 15 循环计数的 4bit 计数器

（可使用 Logisim 中的加法器模块，也可自行设计计数 器），写出计数器的 Verilog 代码

搭建结果的电路图如下图所示（各模块中只有带复位的 D 锁存器较为特殊，故作展示）：

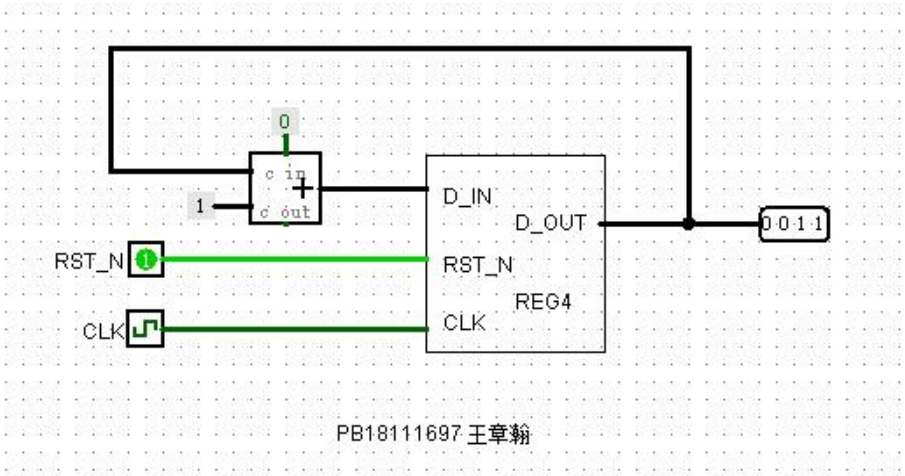


图 1: 总体电路

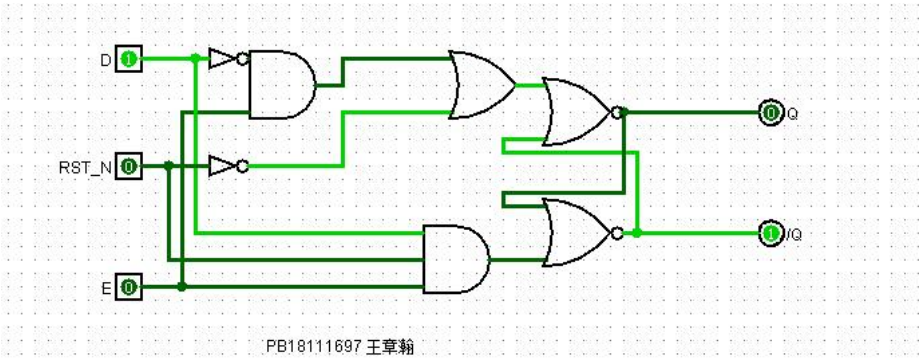


图 2: 最底层的 D 锁存器

其对应的 Verilog 代码如下：

```
1  module D_FF_ASYN_RSTN(
2      input d, CLK, RST_N,
3      output reg q
4      );
```

```
5
6               always@(posedge CLK or negedge RST_N)
7               begin
8                   if(RST_N==0)
9                       q <= 1'b0;
10                  else
11                      q <= d;
12              end
13          endmodule
14
```

D 触发器

```
1       module REG4(
2           input [3:0] D_IN,
3           input CLK, RST_N,
4           output wire[3:0] D_OUT
5           );
6
7           D_FF_ASYN_RSTN D_FF_inst0(D_IN[0], CLK, RST_N, D_OUT[0]);
8           D_FF_ASYN_RSTN D_FF_inst1(D_IN[1], CLK, RST_N, D_OUT[1]);
9           D_FF_ASYN_RSTN D_FF_inst2(D_IN[2], CLK, RST_N, D_OUT[2]);
10          D_FF_ASYN_RSTN D_FF_inst3(D_IN[3], CLK, RST_N, D_OUT[3]);
11      endmodule
12
```

4bit 寄存器

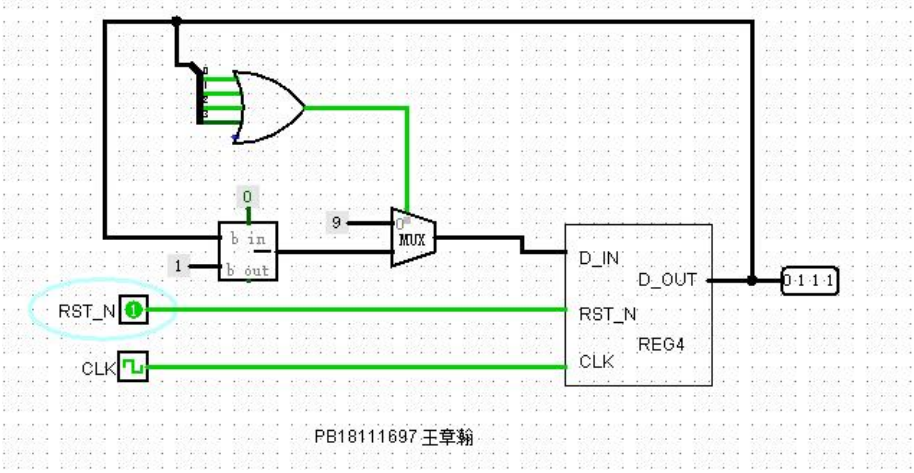
```
1           module counter_up(
2               output[3:0] c);
3               reg CLK;
4               reg RST_N = 0;
5               reg [3:0] num = 4'b0000;
6
7               REG4 REG4_init0(num, CLK, RST_N, c);
8
9               always
10              begin
11                  RST_N = 1;
12                  CLK = 0; #5;
13                  CLK = 1; #5;
14                  num = c + 1;
15              end
16          endmodule
17
```

向上 0 到 15 计数

4.4 题目 4

在 Logisim 中搭建一个 9 0 循环递减的计数器，复位值为 9，每个周期减一（可使用 Logisim 中的减法器模块，也可自行设计计数器），画出电路图，进行正确性测试，并写出其对应的 Verilog 代码。

搭建结果的电路图如下图所示，



其对应的 Verilog 代码如下（只展示与题目 3 不同的代码部分）：

```
module counter_down(
    output[3:0] c);
    reg CLK;
    reg RST_N = 0;
    reg [3:0] num = 4'b0000;

    REG4 REG4_init0(num, CLK, RST_N, c);

    always
    begin
        RST_N = 1;
        CLK = 0; #5;
        CLK = 1; #5;
        if(num == 0)
            num = 9;
        else
            num = c - 1;
    end
endmodule
```

```
20
```

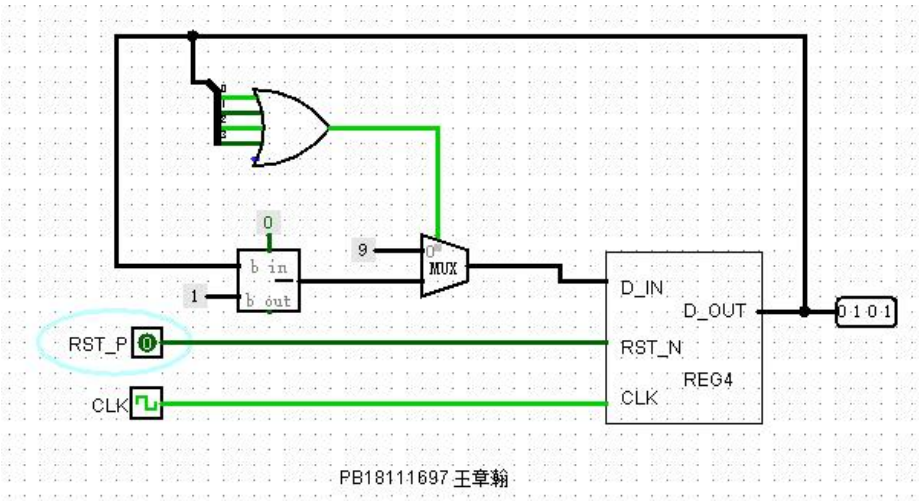
向下 9 到 0 计数

4.5 题目 5

前面所有电路的复位信号都是低电平有效，如要使复位信号高电平有效，应如何实现？试用 Logisim 画出一个示例电路，并编写 Verilog 代码。

选取题目 4 进行修改。

修改的部分的电路图如下：



而程序相较于题目 4 只需做如下修改。

```
1   module D_FF_ASYN_RSTP(
2   input d, CLK, RST_P,
3   output reg q
4   );
5
6   always@(posedge CLK or posedge RST_P)
7   begin
8   if(RST_P==0)
9   q <= 1'b0;
10  else
11  q <= d;
12  end
13  endmodule
```

```
14
```

向下 9 到 0 计数 (RST 高电平有效)

5 总结与思考

5.1 本次实验的收获

在本次实验中，体验了使用 Logisim 作时序逻辑电路，并在 Vivado 中写出对应代码，并仿真模拟来验证。初步理解了时序逻辑电路在 Verilog 中的表达。

5.2 评价本次实验的难易程度

本次实验内容难度适中，基本上是可以自主完成的。

5.3 评价本次实验的任务量

本次实验任务量较大，需要课后花费较长时间才能完成。

5.4 为本次实验提供改进建议

暂无建议。