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：牛庆源

学生学号：PB21111733

完成日期：2022.10.27

【实验题目】

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。

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。

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。

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。

题目 5. 手册中给出的示例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。

【实验目的】

1.进一步熟悉 logisim 中的元件。

2.对时序逻辑电路有一个基本的认识，学会基本的操作。

3.编写 verilog 代码。

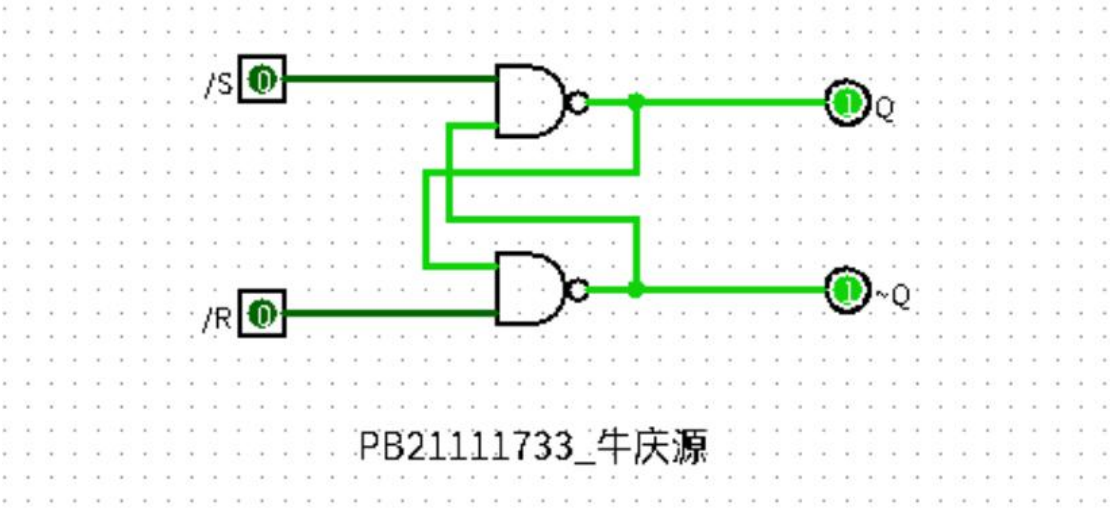
【实验环境】

https://vlab.ustc.edu.cn/

【实验练习】

题目一

直接搭建与非门 S-R 锁存器即可

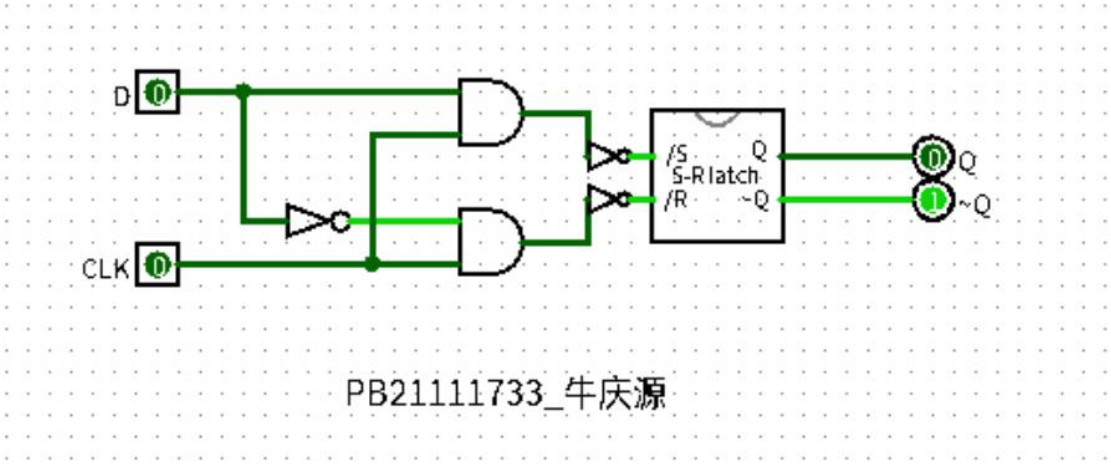


真值表如下，反映了行为特性。S R 输入为 0 0 时输出为 1 1，为非定义状态；输入为 0 1 时输出为 1 0，表示置一；输入为 1 0 时输出为 0 1，表示置零；输入为 1 1 时保持上一次输出。

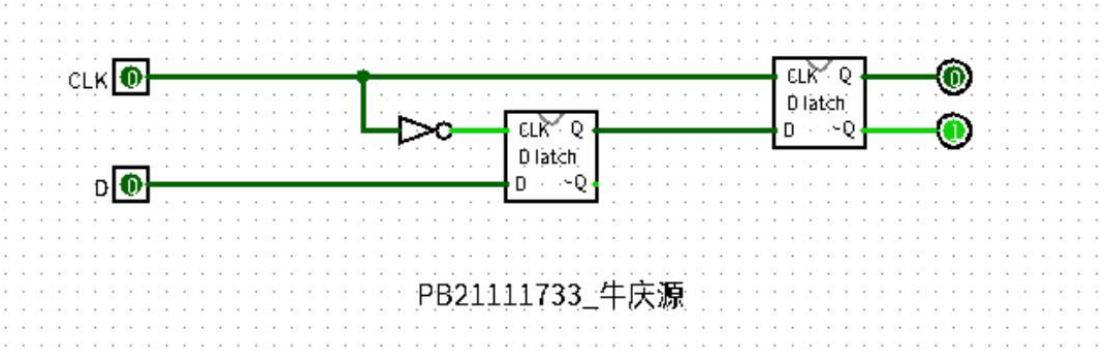
S	R	Q	Q2
0	0	1	1
0	1	1	0
1	0	0	1
1	1	!!	!!

题目二

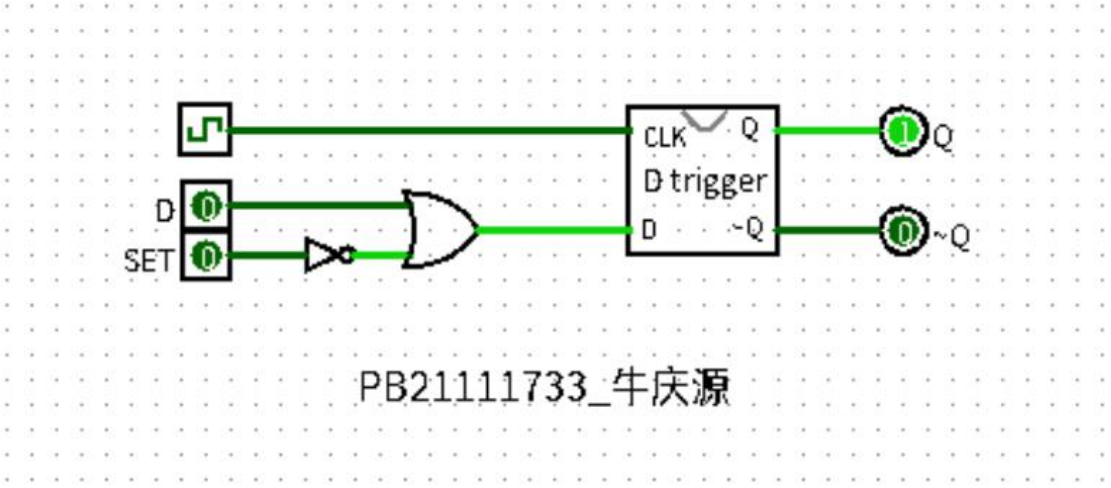
（1） 首先通过 S-R 锁存器连接门电路得到 D 锁存器如下。



（2） 通过 D 锁存器可以得到最基本的 D 触发器如下。



（3） 由于同步置位信号的生效与时钟边沿对齐，则只需把 D 与置位信号 SET 进行门级的连接输入到 D 触发器的 D 端口，即可实现（SET 信号低电平有效）SET 信号有效时直接将输出置一。

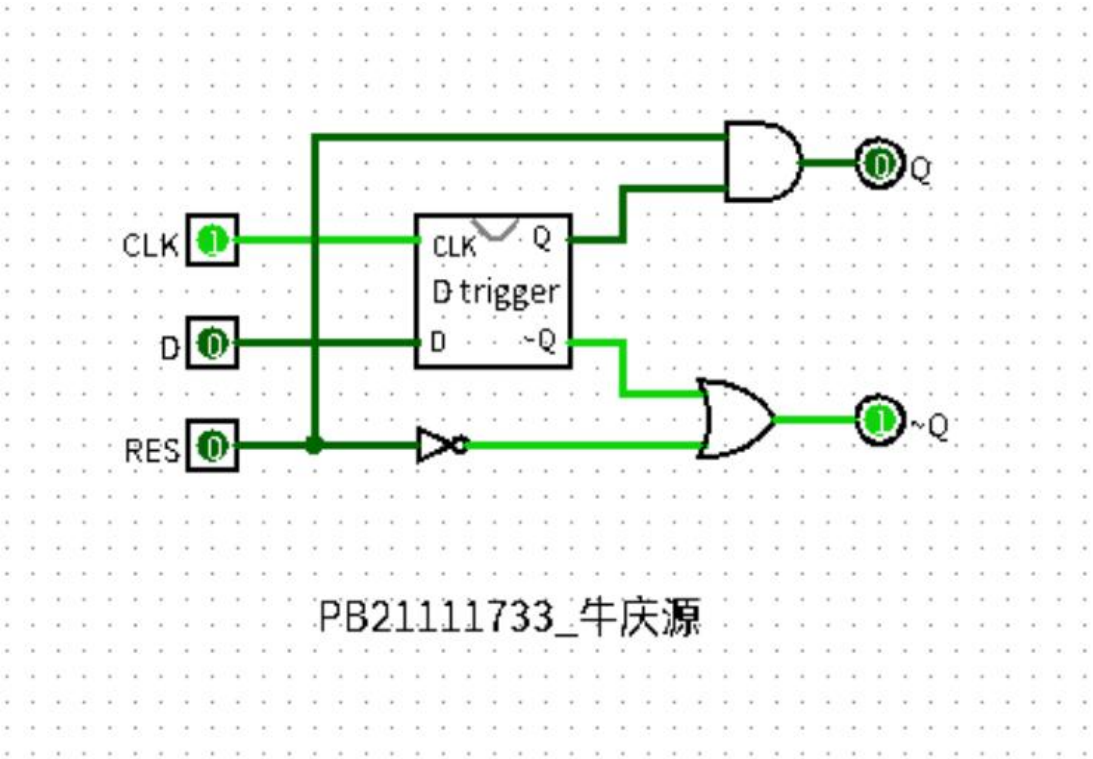


（4） 编写 verilog 代码，时钟一直控制输出，输入的 SET 作为输入 D 是否有效的判断条件，编写代码如下。

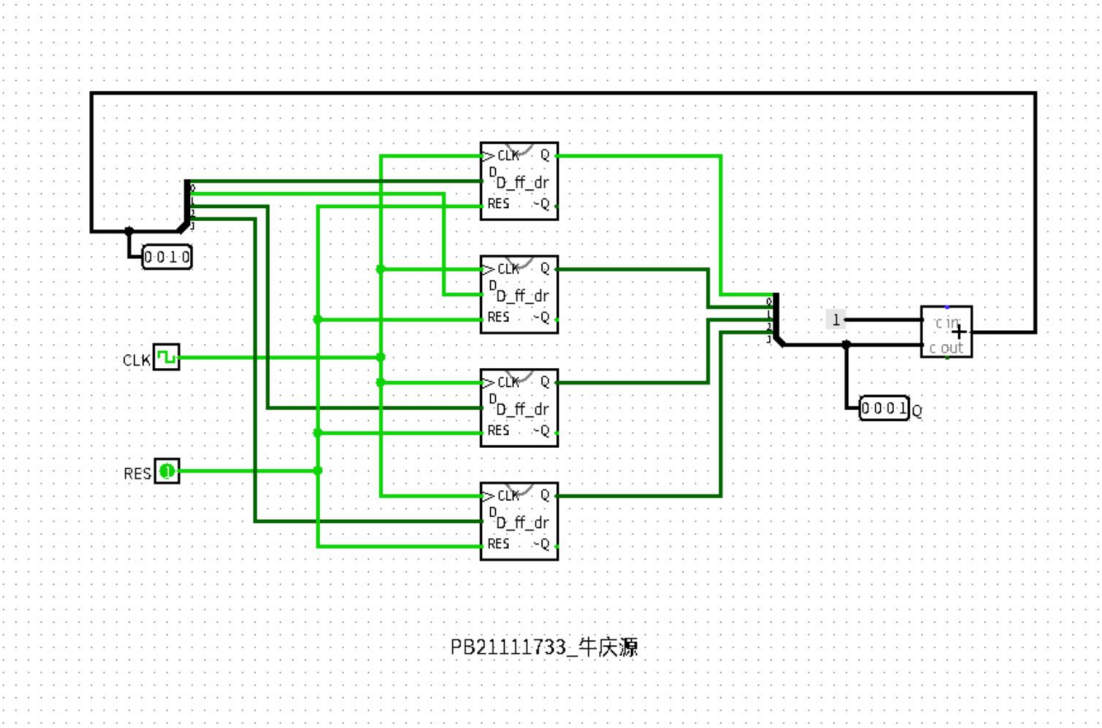
```
module d_ff_sr (
    input CLK, SET, D
    output reg Q);

    always @(posedge CLK)
    begin
        if(SET == 0)
            Q <= 1'b1;
        else
            Q <= D;
    end

endmodule
```

题目三

（1） RES 信号为异步复位信号，低电平有效，有效时不管时钟如何直接将输出复位为 0，于是将 RES 信号越过时钟判断直接影响输出结果即可。（采用基本的 D 触发器元件搭建）



（2） 实现 0 到 15 循环计数，首先使用四个 D 触发器构成一个四位的寄存器，每一个触发器输入的 D 为一位信号，输出为该时钟周期下寄存的信号。要实现从 0 到 15 的循环计数，只需要将寄存器的输出信号经过加法器进行加一操作后重新回到寄存器输入端，此时时钟信号不有效，待下一个上升沿到来时，输入的四位信号即为上一个时钟周期输出信号加一后的信号，如此循环。又由于四位恰好存 0 到 15 的数字，当二进制数为 1111 时加一变为 10000，截断成为 0000，满足继续循环的条件。Logisim 作图如下。



PB21111733_牛庆源

（3） 由（2）的解释可以写出该循环计数器的 verilog 代码。输入端为时钟和复位信号，寄存的信号为四位信号，每次循环加一。若复位信号有效则直接复位为 0000。代码如下。

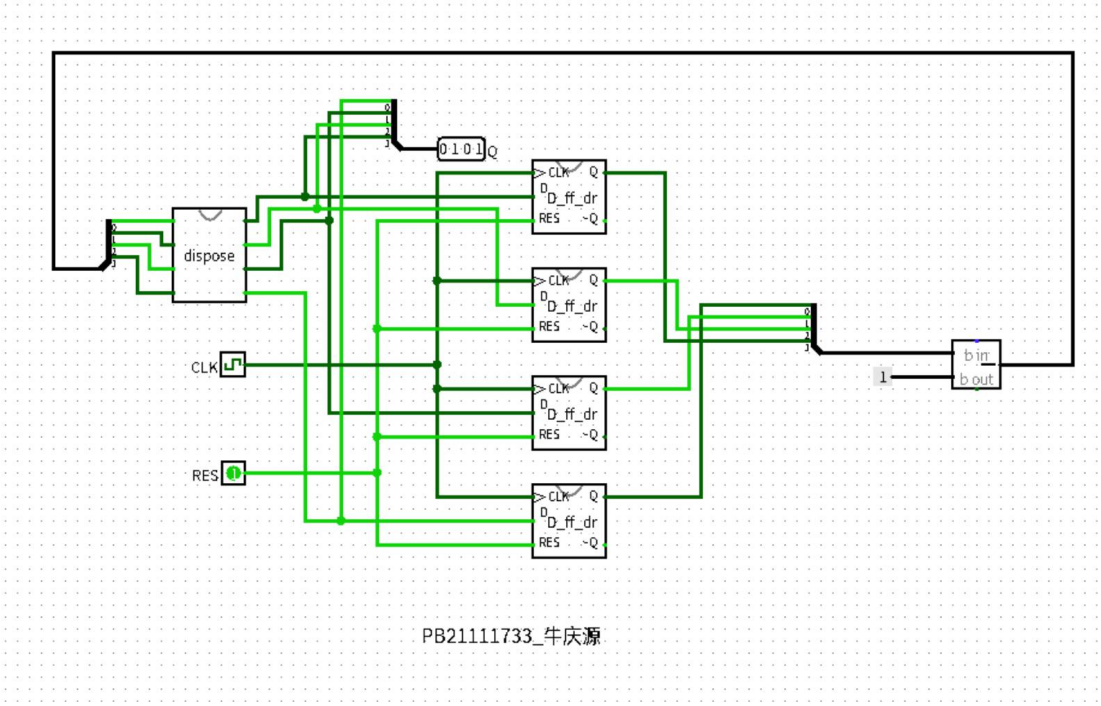
```
module cycle_15 (
    input CLK, RES,
    output reg [3:0] Q);

    always @(posedge CLK or negedge RES)
    begin
        if(RES == 0)
            begin
                Q <= 4'b0000;
            end
        else
            begin
                Q <= Q + 4'b0001;
            end
    end

endmodule
```

题目四

（1） 需要从 9 到 0 的计数，由于三位只能表示到 7，四位表示到 15，于是采用四位寄存器实现，初始信号为 9，每一次寄存后经过一个减法器减一后回到输入端。注意四位寄存器会从 15 到 0 循环，所以在输出为 0 时经过减法器会变成 15，此时采用自己画的模块，经过该模块时，15 会变成 9，从而将 9 输入到寄存器信号中，输出为 9，输入为 8，再次进行循环。自己画得模块采用真值表生成，输入为 0000 到 1001 时输出为 0000 到 1001，输入为 1111 时输出为 1001，其余情况输入为 x 即可，这里采用输出 0001。以下是操作模块的真值表的部分。

I1	I2	I3	I4	O1	O2	O3	O4
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	1
0	1	1	0	0	1	1	0
0	1	1	1	0	1	1	1
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	1
1	1	1	1	1	0	0	1

由于复位信号需要复位到 9，所以将输出信号的位置调至该模块后，即复位信号传入时，寄存器输出 0000，通过减法器得到 1111，再经过操作模块得到 1001，输出信号为 1001。以下为电路图。



（2） Verilog 代码根据（1）的描述也很好得出，如下。当复位信号为 0（有效）时，输出 Q 复位为 1001，当 Q 的值为 0000 时，输出

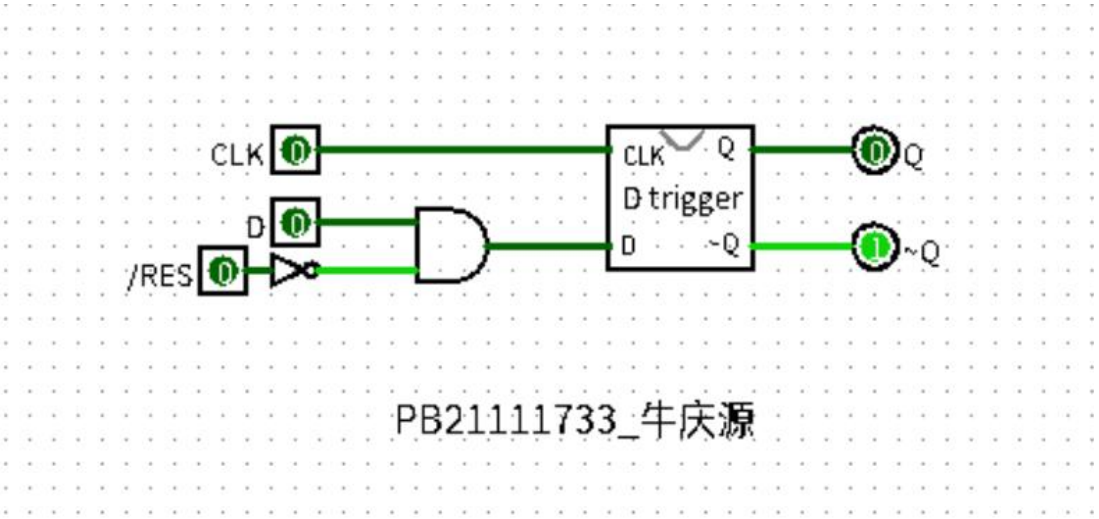
信号为 1001，当 Q 的值小于 1001 时，输出信号减一，其余情况为防止出现不确定态，均将 Q 值赋值为 0000（事实上其余情况也不会发生）。

```
module cycle_sub_9 (
    input CLK, RES,
    output reg [3:0] Q);

    always @(posedge CLK or negedge RES)
    begin
        if(RES == 0)
            begin
                Q <= 4'b1001;
            end
        else if(Q == 4'b0000)
            begin
                Q <= 4'b1001;
            end
        else if(Q <= 4'b1001)
            begin
                Q <= Q - 4'b0001;
            end
        else
            begin
                Q <= 4'b0000;
            end
    end

endmodule
```

题目五

要使得复位信号高电平有效，只需要在原来低电平有效的复位信号后加入一个非门即可实现。（下图为高电平有效复位信号的同步复位 D

触发器）



【总结与思考】

1. 进一步熟悉了 logisim。

2. 用 logisim 实现了最初步的时序逻辑电路，同时又巩固了上一次实验中用真值表生成电路的操作。

2. 对 verilog 语言中的 always 语句进行了初步的了解。