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：Ouedraogo Ezekiel B.

学生学号：PL19215001

完成日期：10/29/2020

计算机实验教学中心制

2020 年 10 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：Windows 操作系统

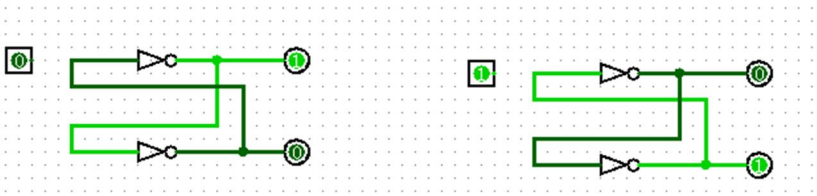
Logisim 仿真工具。

Notepad++ 文本编辑器。

【实验过程】

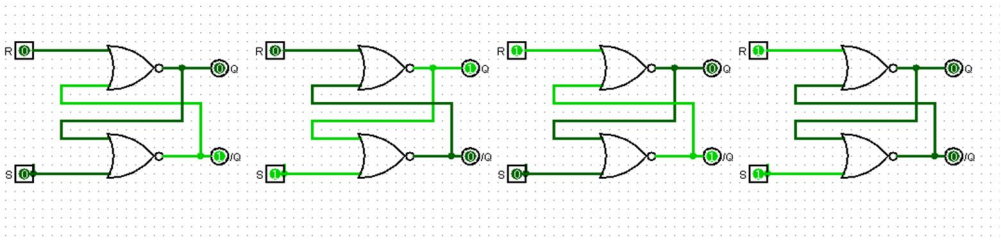
1. 搭建双稳态电路

双稳态电路是由两个非门交叉耦合构成，完全一样 的电路结构，却可以具备两种完全不同的状态。



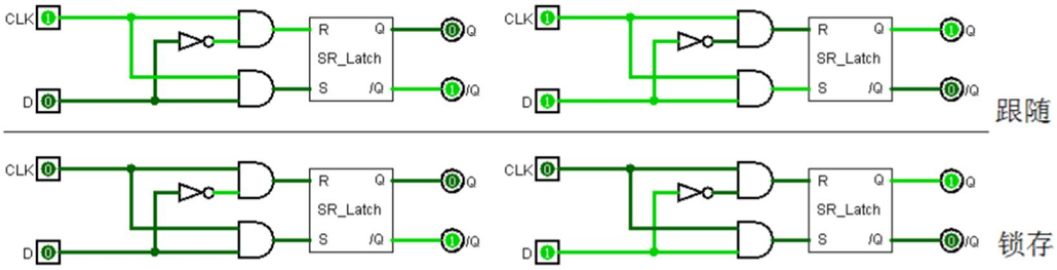
2. 搭建 SR 锁存器

当 SR 信号都无效（为 0）时，电路将保持之前的状态，即处于锁存状态，因此这种电路称 为 SR 锁存器。SR 信号都有效（为 1）时，Q 和/Q 信号都为零，虽然 也是一种确定状态，但不符合/Q 为 Q 取反的定义，因此我们将其看 成是一种未定义状态。



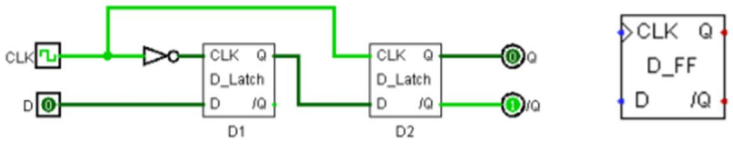
3. 搭建 D 锁存器

分析 D 锁存器电路可以发现，当 CLK 信号为高电平时，Q 信号将随着 D 端输入信号的变化而变化，称之为“跟随”状态。当 CLK 信号为低电平时，Q 信号将保持之前的值，不会收到 D 信号变化的影响， 称之为“锁存”状态。D 锁存器是一种电平敏感的时序逻辑器件。



4. 搭建 D 触发器

在 D 触发器中当 CLK 信号由低电平变为高电平 时 Q 信号才能收到 D 端输入的信号。



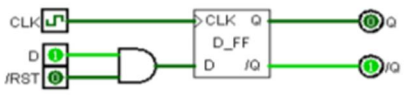
其 Verilog 代码如下

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

always 表示其后 是个过程语句块。reg 与 wire 关键字类似。对于初学者，可以简单的理解为凡是在 always 语句块内被赋值 的信号，都应定义为 reg 类型。posedge 为事件控制关键字，例如代码中的“posedge clk”表示“clk 信号的上升沿”这一事件。对应“<=”只需记住一个原则：组合逻辑采用阻塞赋值“=”， 时序逻辑采用非阻塞赋值“<=”。

- 带同步复位信号触发器

当 复位信号有效（低电平有效）时，输出信号 Q 始终为零。



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。其 Verilog 代码如下所示：

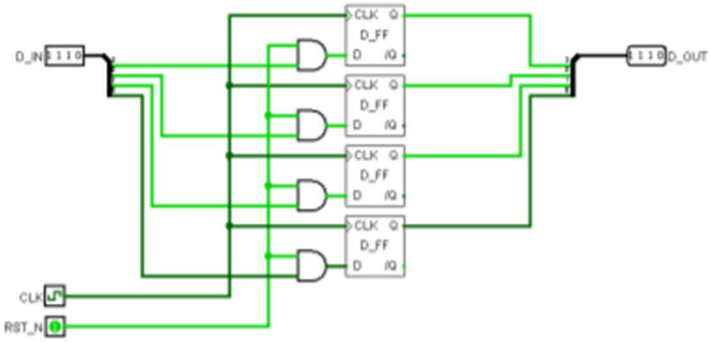
```
module d_ff_r(
input clk,rst_n,d,
output reg q);
always@(posedge clk)
begin
    if(rst_n==0)
        q <= 1'b0;
    else
        q <= d;
end
endmodule
```

begin/end 必须成对出现，用于表征语句块的作用区间，如上述例子中，begin/end 之间的 代码都属于同一 always 块。

5. 搭建寄存器

用 4 个 D 触 发器构成一个能够存储 4bit 数据的寄存器，带有低电

平有效的同 步复位信号。



其 Verilog 代码为：

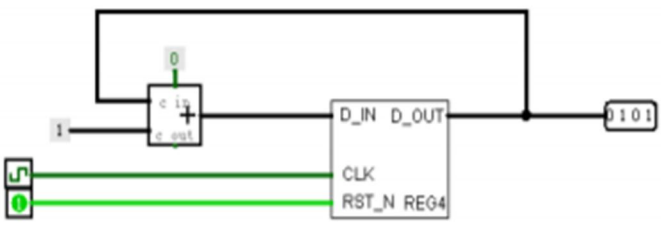
```
module REG4(
input CLK,RST_N,
input [3:0] D_IN,
output reg [3:0] q);
always@(posedge CLK)
begin
    if(RST_N==0)
        D_OUT <= 4'b0;
    else
        D_OUT <= D_IN;
end
endmodule
```

“[x:y]” 用于 bit 位宽信号的声明。上述 D_IN 为 4 bit 的信号。

D_IN[3]为 MSB，D_IN[0]为 LSB。

6. 搭建简单时序逻辑电路

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



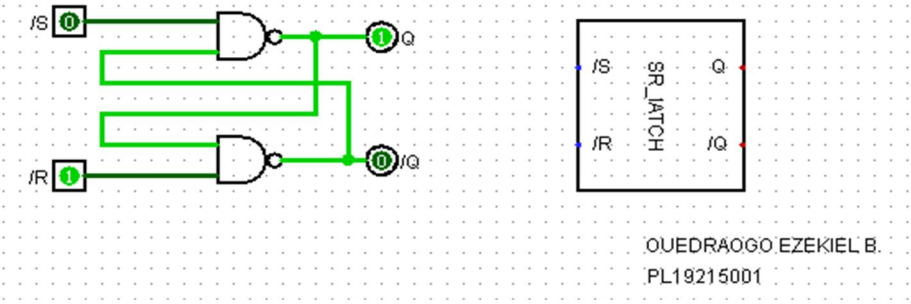
其 Verilog 代码为：

```
module REG4(
input CLK,RST_N,
output reg [3:0] CNT);
always@(posedge CLK)
begin
    if(RST_N==0)
        CNT <= 4'b0;
    else
        CNT <= CNT + 4'b1;
end
endmodule
```

【实验练习】

1. 搭建 SR 锁存器

用与非门搭建 SR 锁存器

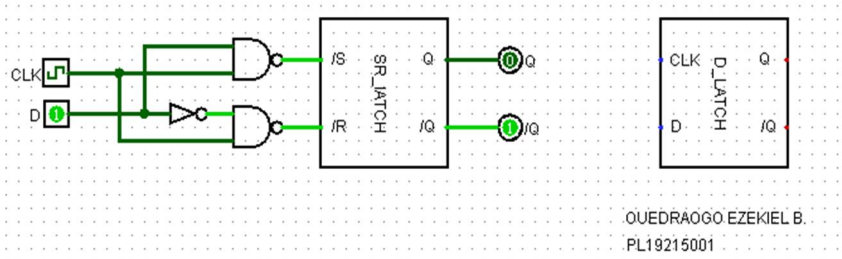


两个输入信号分别命名为 /S 和 /R，输 出信号命名为 Q 和/Q。/S 信号负责对 Q 置位 （Set），/R 信号负责对 Q 信号置位（Reset）。当 SR 信号都无效（为 1）时，电路将保持之前的状态，即处于锁存状态。SR 信号都有效（为 0）时，Q 和/Q 信号都为 1，不符合/Q 为 Q 取反的定义，因此是一种未定义状态。

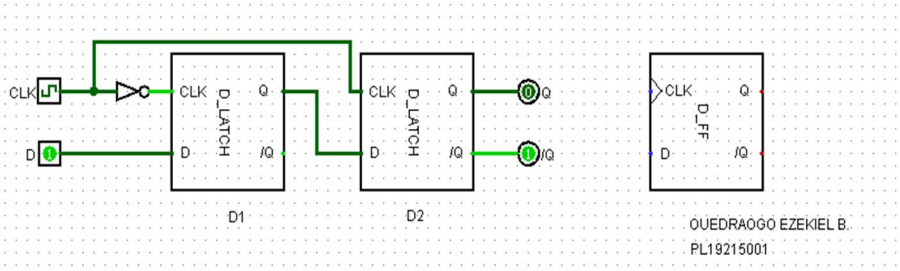
S	R	Q^{n+1}	**说明**
0	0	1*	禁止
0	1	1	置 1
1	0	0	清 0
1	1	Q^n	保持

2. 搭建同步置位 D 触发器

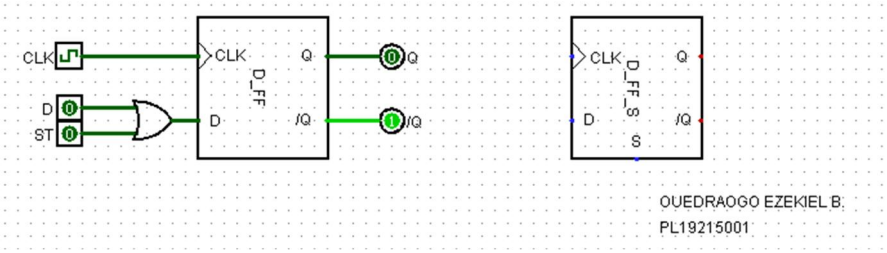
首先用上面搭建的 SR 锁存器搭建 D 锁存器



再利用两个 D 锁存器搭建 D 触发器



最后搭建高电平有效的同步置位 D 触发器

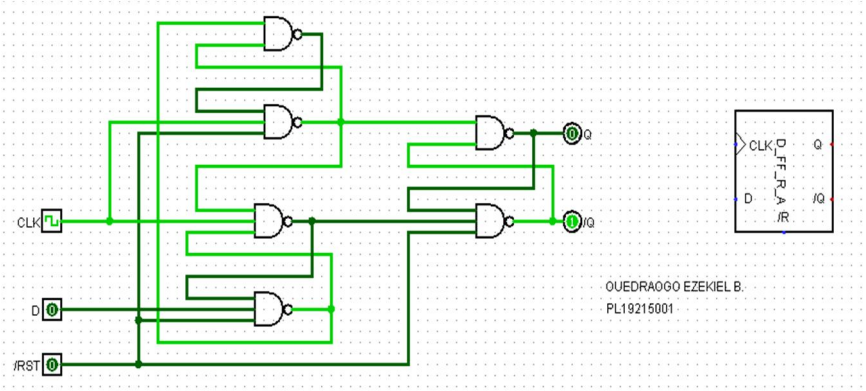


Verilog 代码

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

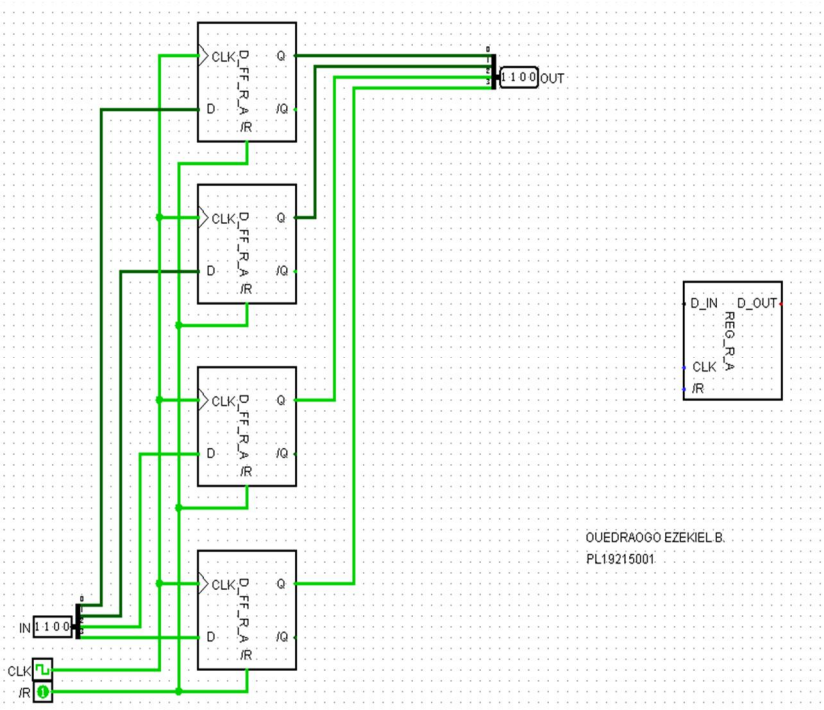
3. 搭建异步复位 D 触发器和 0~15 循环计数器

搭建异步复位 D 触发器

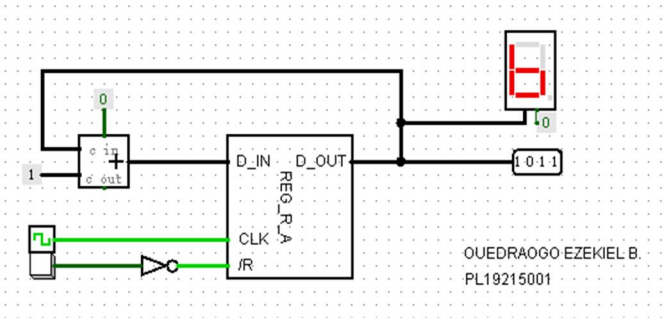


调用该异步复位 D 触发器设计一个从 0~15 循环计数的 4bit 计数器

首先用该触发器设计一个寄存器



再设计计算器

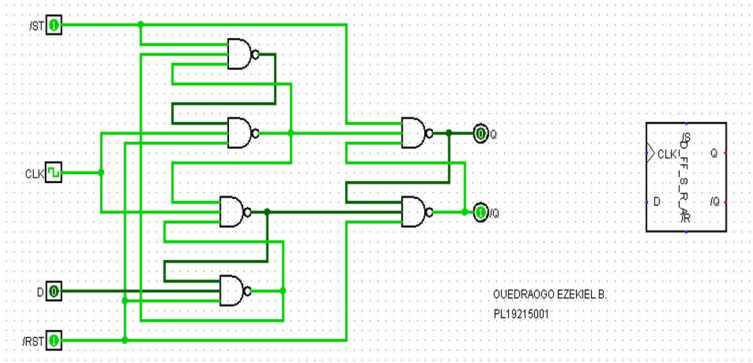


计数器的 Verilog 代码。

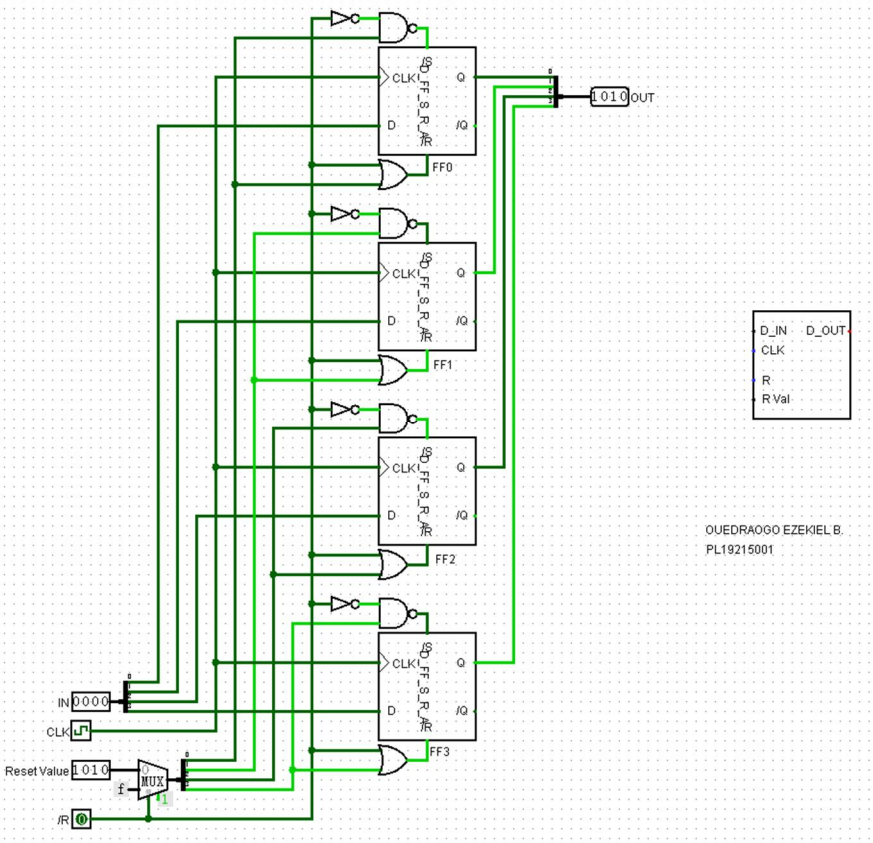
```
module REG4(
    input CLK,RST_N,
    output reg [3:0] CNT);

    always@(posedge CLK or negedge RST_N)
    begin
    if(RST_N==0)
        CNT <= 4'b0;
    else
        CNT <= CNT + 4'b1;
    end

endmodule
```

4. 搭建 9~0 循环递减计数器

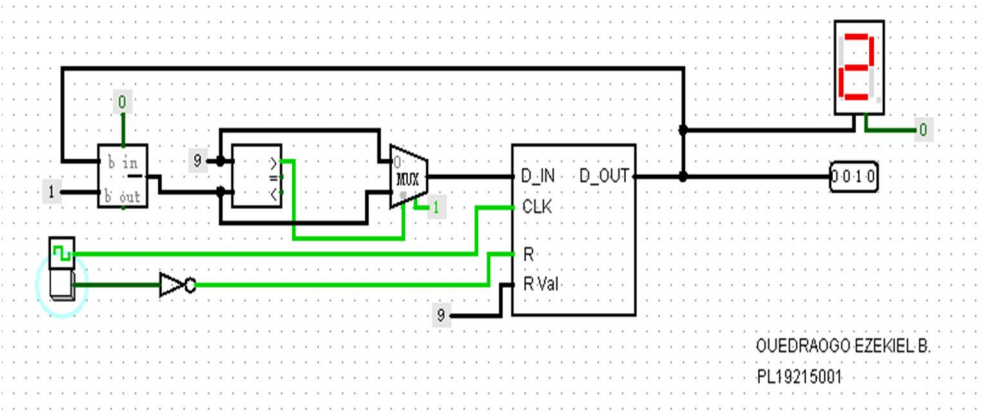
首先搭建带异步复位和异步置位的 D 触发器



利用该触发器搭建一个寄存器。该寄存器有定值复位的功能



最后搭建计算器



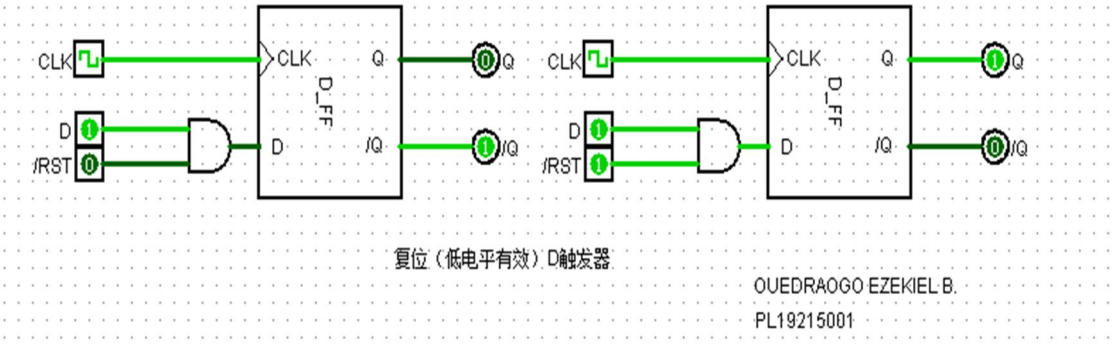
其 Verilog 代码如下

```
module REG(
    input CLK,RST_N,
    output reg [3:0] CNT);

    always@(posedge CLK or negedge RST_N)
    begin
    if(RST_N==0 || CNT==4'b0 || CNT>4'b1001)
        CNT <= 4'b1001;
    else
        CNT <= CNT - 4'b1;
    end

endmodule
```

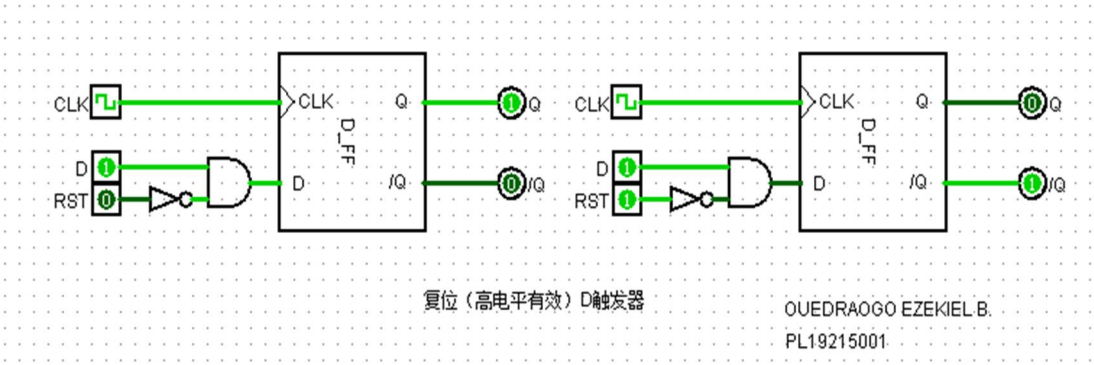
5. 低电平换高电平

如要使复位信号 高电平有效只要在复位信号的入口加一个非门就可以。例如:

同步复位（低电平有效）D 触发器



同步复位（高电平有效）D 触发器



其（高电平有效）Verilog 代码如下

```
module d_ff_r(
    input clk,rst,d,
    output reg q
    );

    always@(posedge clk)
    begin
        if(rst==1)
            q <= 1'b0;
        else
            q <= d;
    end

endmodule
```

【总结与思考】

本次实验中咱们学到了如何用基本逻辑门搭建时序逻辑器件。因为时序逻辑电路受到当前的输入和之前的状态影响，构造这种逻辑电路更加困难。