# 实验三：三位扭环计数器的设计与实现

【**实验要求**】：

1. 采用Moore（摩尔型）电路，利用D触发器，设计并实现三位扭环计数器。

【**实验目的**】

1. 掌握时序逻辑电路的设计方法；
2. 熟悉Vivado2014集成开发环境和Verilog编程语言；
3. 实现如下图所示的三位扭环计数器。
4. 

【**实验环境**】

* FPGA虚拟仿真平台。
* Vivado2014 集成开发环境。
* Verilog编程语言。

【**实验原理**】

功能描述

计数功能：主要功能是计数，每次触发一次计数器，二进制数字会递增。计数器可以在每次触发时增加一个二进制单位，例如从000递增到001，然后到010，再到011，以此类推。

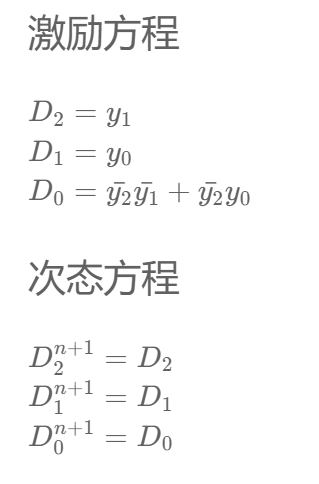
显示功能：有三个二进制数字显示位，用于显示当前的计数数值。每个数字位都能够独立显示0或1。当计数器计数达到111（二进制）时，通常会回到000重新计数，形成循环显示。

复位功能：通常，计数器配备有复位按钮或功能，用于将计数器的值重置为初始状态，即000。这样可以方便重新开始计数。

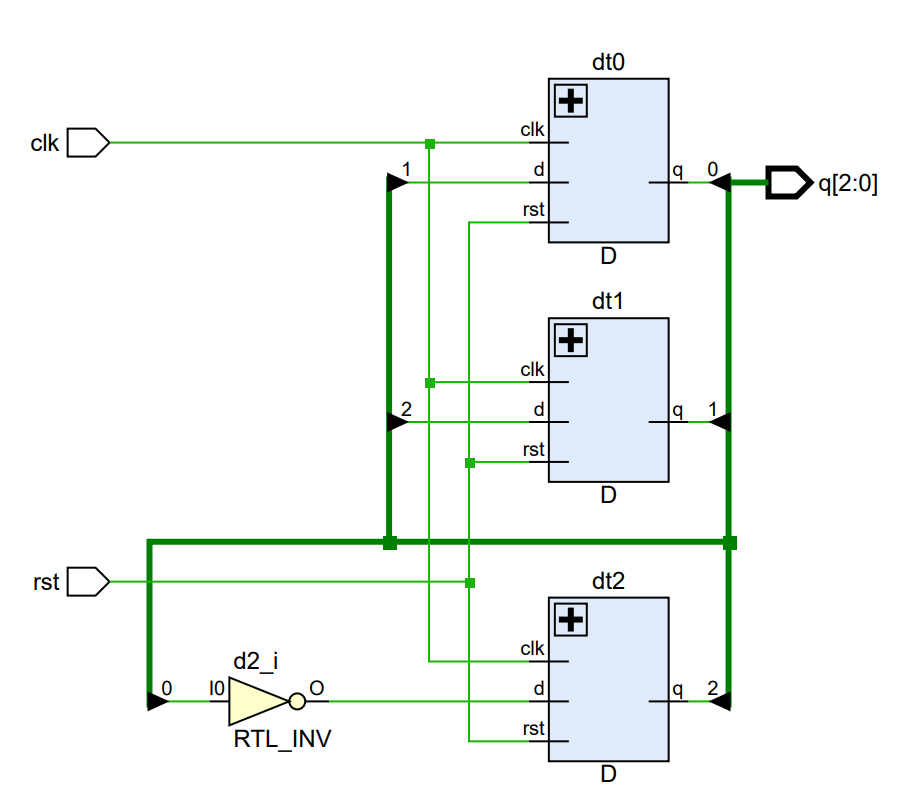
真值表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| y2 | y1 | y0 | D2 | D1 | D0 |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 |

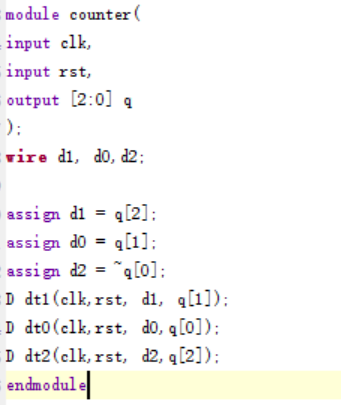
逻辑方程

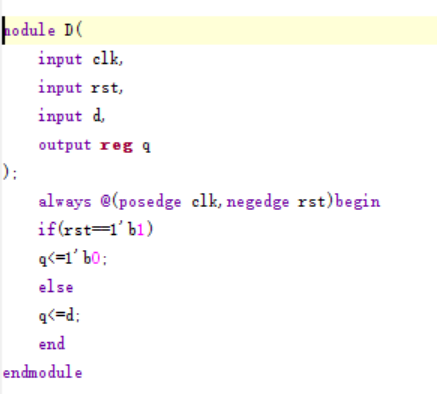


电路图

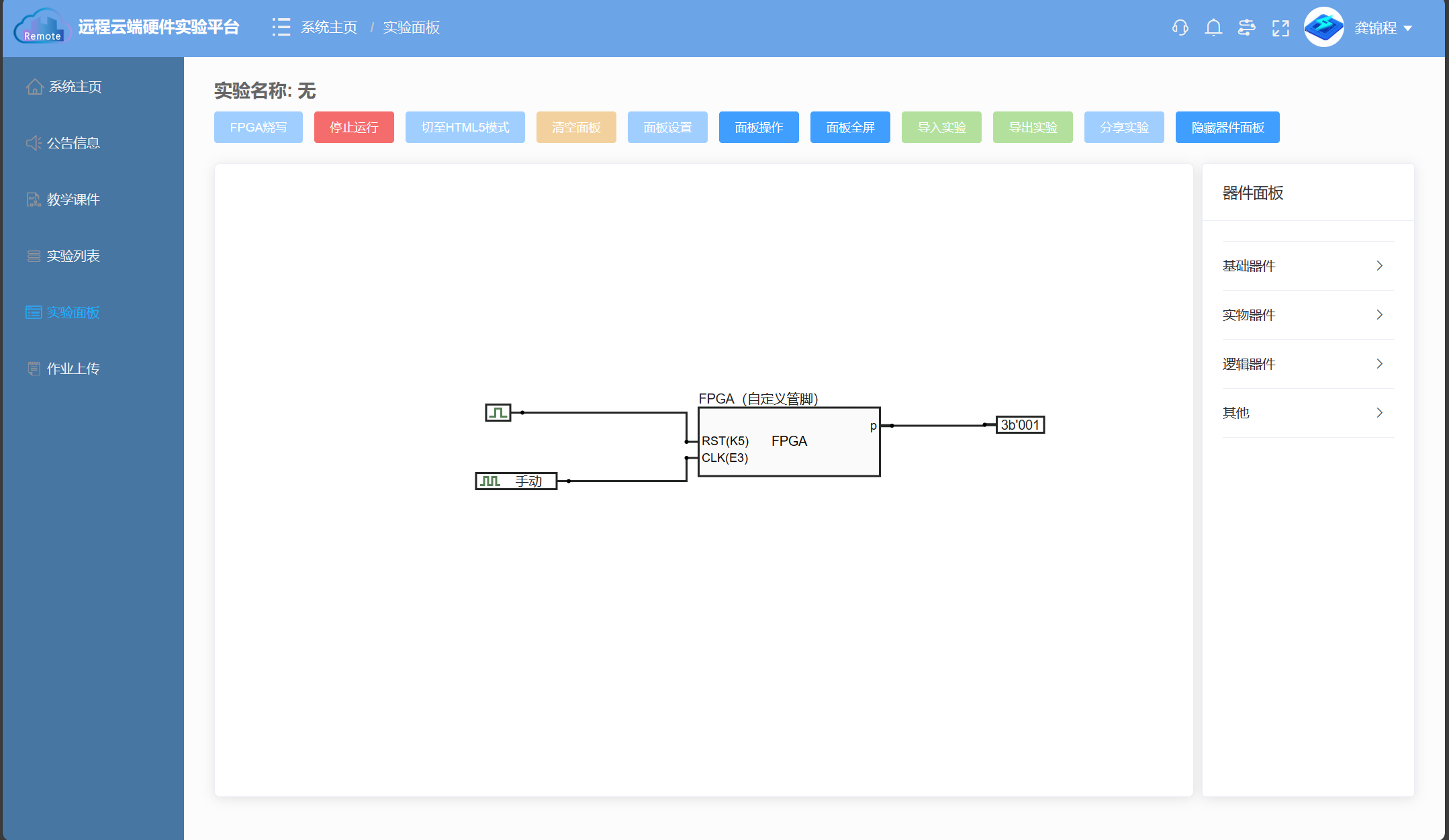


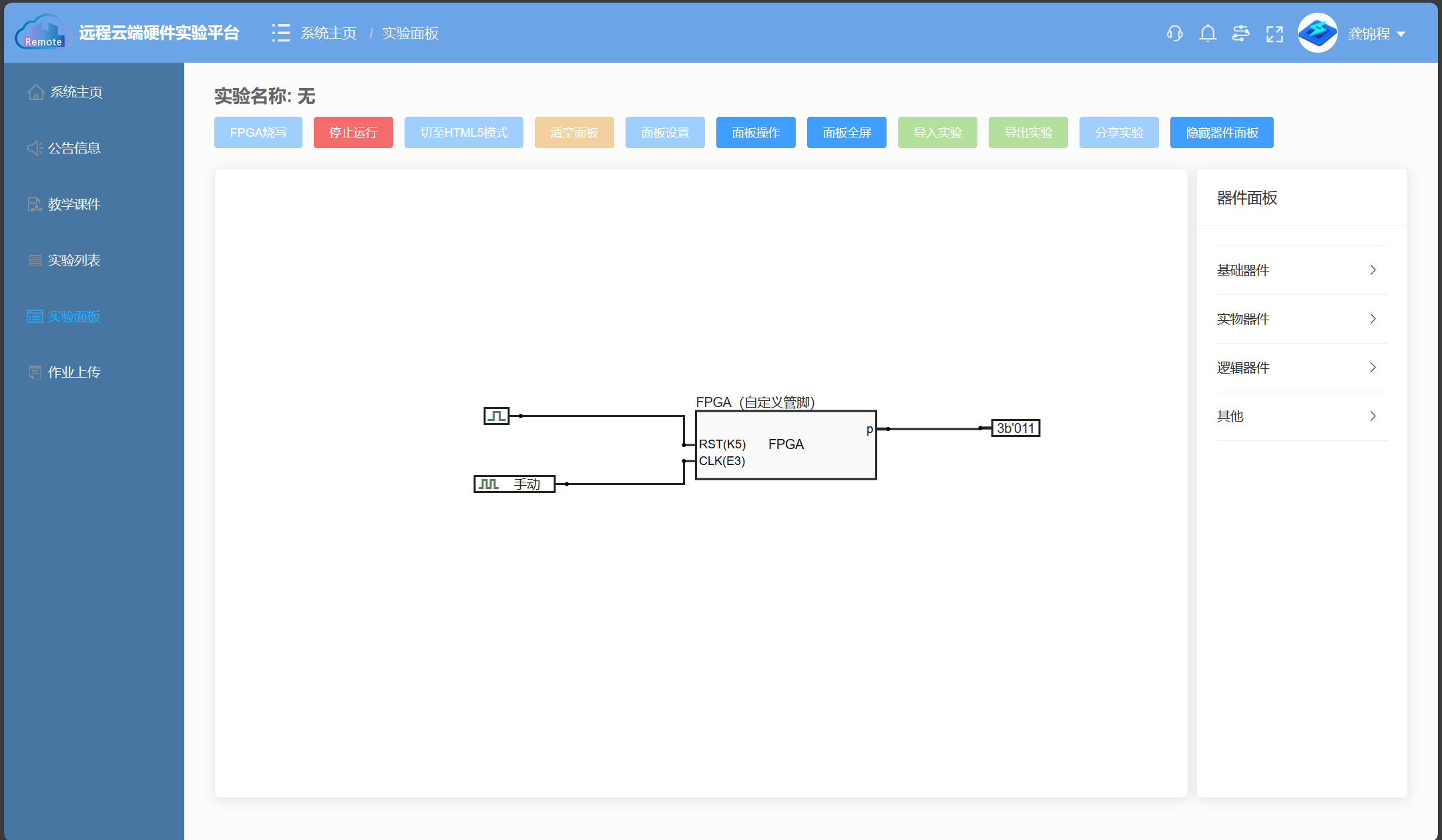
Verilog代码实现





实验结果与仿真结果





# 实验四：四位移位寄存器的设计与实现

【**实验要求**】：

1. 设计4位串行移位寄存器，使其能够实现串行移位功能，即串行输入信号S1输入，依次经过四个触发器串行输出，要求能够进行左移和右移。
2. 串行左移举例：串行输入信号S1=1，现态Q3nQ2nQ1nQ0n为0000时，当时钟上升沿到来时，次态（输出）Q3nQ2nQ1nQ0n为0010；下一时钟上升沿到来时，次态（输出）Q3nQ2nQ1nQ0n为0100；下一时钟上升沿到来时，次态（输出）Q3nQ2nQ1nQ0n为1000。

【**实验目的**】

1. 掌握时序逻辑电路的设计方法；
2. 熟悉Vivado2014集成开发环境和Verilog编程语言；
3. 实现四位串行移位寄存器，包括左移和右移。

【**实验环境**】

* FPGA虚拟仿真平台。
* Vivado2014 集成开发环境。
* Verilog编程语言。

【**实验原理**】

功能描述

寄存器结构： 由四个触发器（Flip-Flop）组成，分别存储四个位（Q0到Q3）。

控制信号： 引入一个控制信号，比如说"Shift\_Left"和"Shift\_Right"。这个信号决定了在时钟上升沿到来时是左移还是右移。

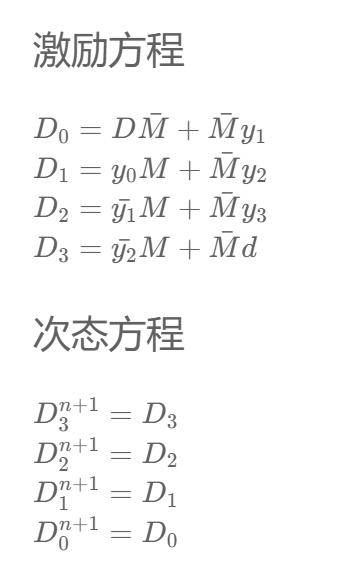
左移操作： 如果控制信号为"Shift\_Left"，那么在每个时钟上升沿到来时，数据从Q0传递到Q1，Q1传递到Q2，Q2传递到Q3，同时Q3传递到一个辅助输出，比如说Q3\_out。这样就完成了左移一位的操作。

右移操作： 如果控制信号为"Shift\_Right"，那么在每个时钟上升沿到来时，数据从Q3传递到Q2，Q2传递到Q1，Q1传递到Q0，同时Q0传递到一个辅助输出，比如说Q0\_out。这样就完成了右移一位的操作。

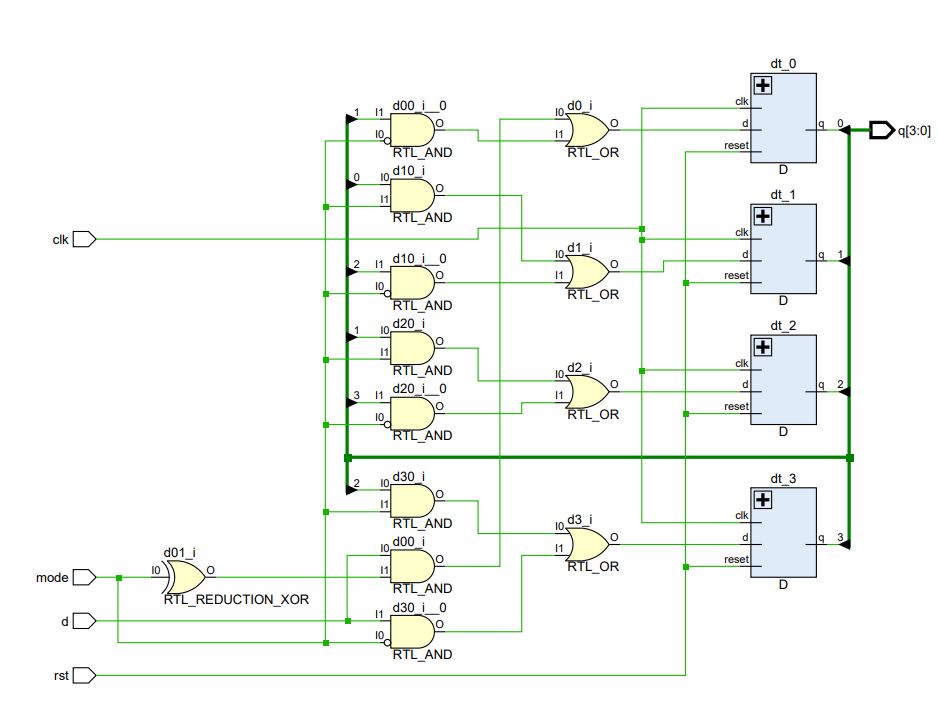
真值表

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| mode | d | Q1 | Q2 | Q3 | Q4 | Qn1 | Qn2 | Qn3 | Qn4 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |

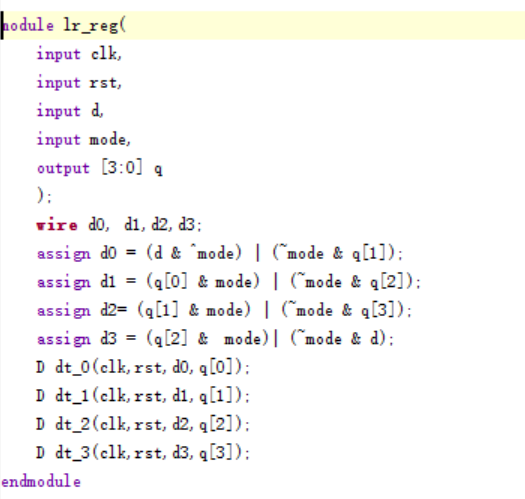
逻辑方程

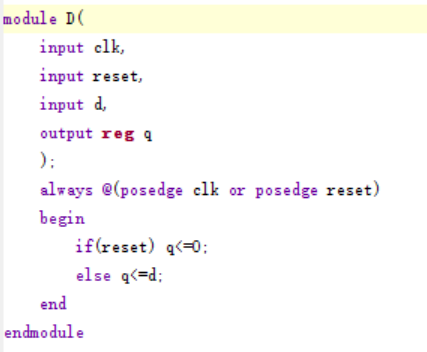


电路图



Verilog代码实现





实验结果与仿真结果

