**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

学生学号： PB20111642

完成日期： 2021.11.8

【实验题目】

我们通过之前的实验应该能够达到熟练使用 Logisim 工具以及阅读、编写简单组合逻辑 Verilog HDL 代码的程度。因此本次实验我们将使用 Logisim 设计简单时序电路，并学习用 Verilog 语言描述简单时序逻辑电路。

【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】

·vlab.ustc.edu.cn

·Logisim

·verilogoj.ustc.edu.cn

【实验过程】

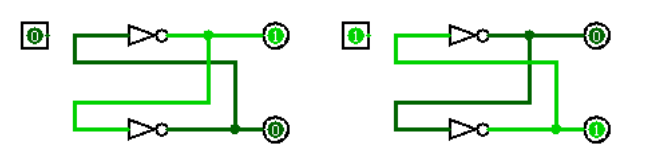
时序逻辑电路的输出受到电路当前输入和之前状态两种因素的影响。当输入信号完全一样时，时序逻辑电路的输出及状态可能不一样，因此说，时序逻辑电路具有记忆功能，能够记住电路之前的状态。下面我们将使用与或非三种基本门逐步搭建出各种时序逻辑电路的关键器件，在此过程中加深对时序逻辑器件结构和工作原理的理解。

**·搭建双稳态电路**

双稳态电路是由两个非门交叉耦合构成，如下图所示，完全一样

的电路结构，却可以具备两种完全不同的状态，这一点与组合逻辑电路存在本质的区别。双稳态电路是一种最简单的时序逻辑电路，没有

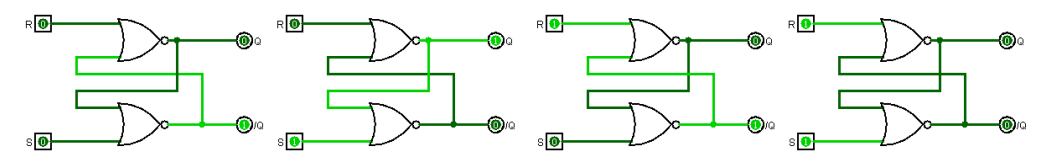
输入信号，状态一旦确定之后也无法改变，没有实际使用价值，但却是所有时序逻辑电路的基础。



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。

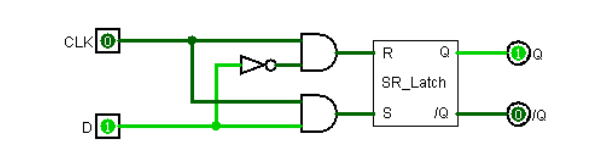
**·搭建SR锁存器**

双稳态电路没有输入信号，所以无法进行操作，我们对其进行修改，将两个非门用或非门代替。两个输入信号分别命名为 S 和 R，输出信号命名为 Q 和/Q，其中/Q 是 Q 取反的意思，S 信号负责对 Q Set，R 信号负责对 Q 信号Reset。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，在实际使用过程中应避免这种状态的出现。

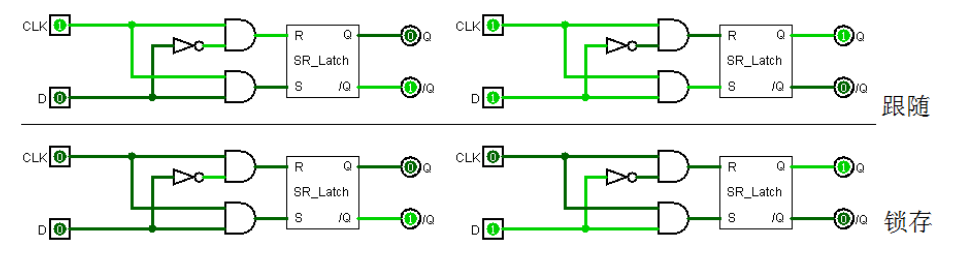


**·搭建D锁存器**

SR 锁存器两个输入都为 1 是一种未定义状态，我们不希望这种状态出现，为此我们在 SR 锁存器前面添加两个与门和一个非门，如下图所示，便构成了 D 锁存器。

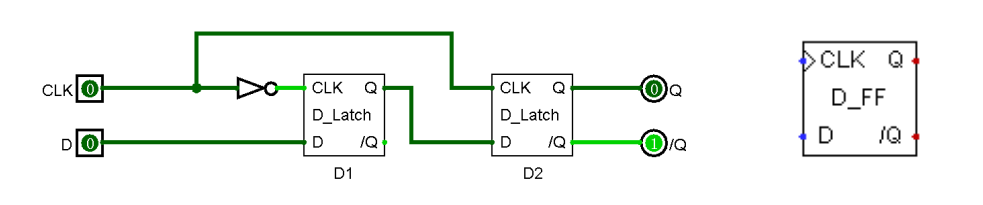


分析 D 锁存器电路可以发现，当 CLK 信号为高电平时，Q 信号将随着 D 端输入信号的变化而变化，称之为“跟随”状态。当 CLK 信号为低电平时，Q 信号将保持之前的值，不会收到 D 信号变化的影响，称之为“锁存”状态。D 锁存器是一种电平敏感的时序逻辑器件。

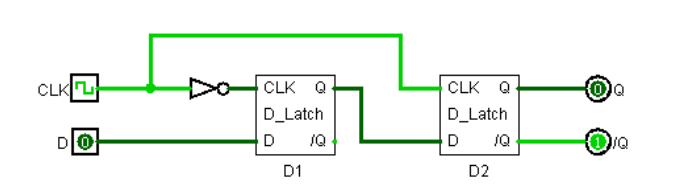


**·搭建D触发器**

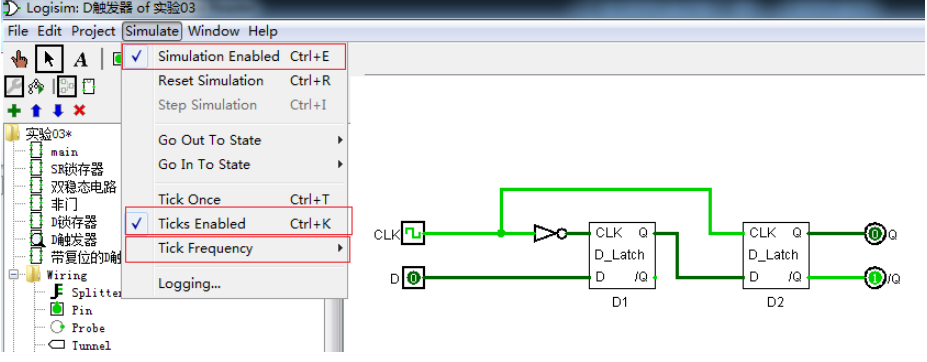
通过对 D 锁存器的行为特性分析，我们可以发现，D 锁存器在信号的传输过程中起到了类似于开关的作用，当开关（CLK 信号）打开的时候，信号能够传输过去，当开关（CLK 信号）关闭时信号无法通过。如果我们将两个 D 锁存器串起来，其控制信号有效值始终相反，会是什么样的情况呢？实际上这就构成了 D 触发器，如下图所示，CLK信号为低电平时，D 信号通过了 D1，当 CLK 信号由低电平变为高电平时，D1 关闭，D2 打开，信号到达 Q 端。



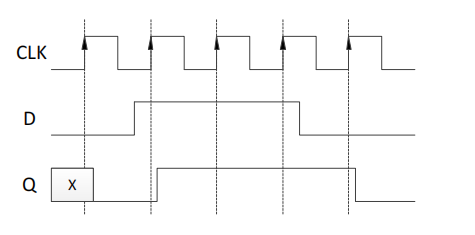
为了更直观的观察 D 触发器的行为特性，我们可以把 CLK 端口换成一个可自动变化的时钟信号，如下图所示。



修改完之后，我们可在 Logisim 菜单栏中点击“simulation”选项，首先将“Tick Frequency”设置为“1Hz”，然后使能仿真和触发功能，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。



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。将 D 触发器作为一个整体观察，该行为特性如下波形图所示。



其 Verilog 代码如下所示：

*module d\_ff(*

*input clk,d,*

*output reg q);*

*always@(posedge clk)*

*q <= d;*

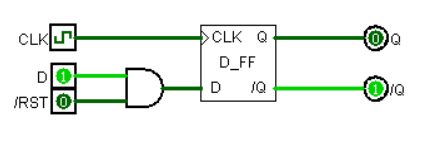
*endmodule*

*（说明：reg、always 和 posedge 是 Verilog 中的关键字，其中 always 表示其后是个过程语句块。reg 与前面学习到的 wire 关键字类似，是一种数据类型，称为寄存器类型。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。另外，在时序逻辑电路中，信号赋值采用“<=”（非阻塞赋值）,而不是“=”（阻塞赋值），这两种赋值方式的区别暂不介绍，读者只需记住一个原则：组合逻辑采用阻塞赋值“=”，*

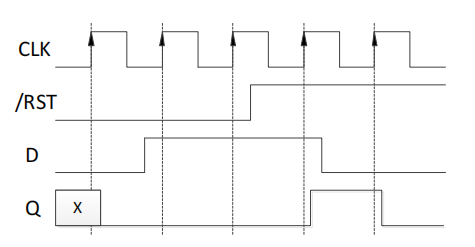
*时序逻辑采用非阻塞赋值“<=”。）*

我们还可以为触发器添加复位信号，如下图所示，可以看出，当

复位信号有效（低电平有效）时，输出信号 Q 始终为零。



其波形图如下图所示。



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*

*（说明：这段代码中又新出现了 begin、end、if、else 四个关键字，其中 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。if、else 用于条件判断，在很多其它语言中都有出现，其含义也都一样，此处不再赘述。“1’b0”是一种数据表示方式，一般格式为“数据位宽’进制数值”，本例中表示这是一个 1bit 的数据，用二进制表示，其值为 0。）*

与此同步复位相对应的，还有一种异步复位方式，即不论时钟和

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（一般为低电平），读者可考虑一下这种触发器电路结构。其 Verilog 代码为：

*module d\_ff\_r(*

*input clk,rst\_n,d,*

*output reg q);*

*always@(posedge clk or negedge rst\_n)*

*begin*

*if(rst\_n==0)*

*q <= 1’b0;*

*else*

*q <= d;*

*end*

*endmodule*

*（说明：negedge 是与 posedge 同类型的一个关键字，只不过它表示信号的下降沿事件。关键字“or”表示“或”操作）*

可以看出，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，在没有时钟上

升沿的情况下，复位信号也能够起作用。因为复位操作不再完全与时

钟信号的上升沿同步，因此称为异步复位。

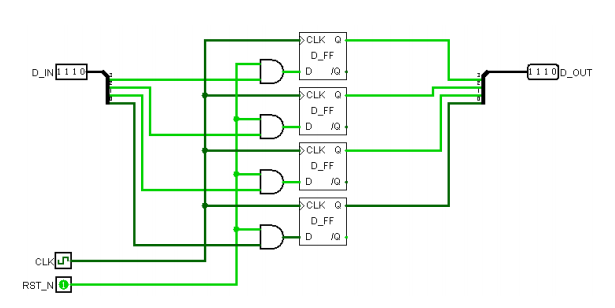
D 触发器与 D 锁存器的最大不同在于它是边沿敏感的器件，电路输出状态只在时钟信号的边沿（一般来说是上升沿）发生（异步复位除外），时钟成为整个电路的同步信号，因此由 D 触发器为核心构成的电路一般称为同步时序逻辑电路，而锁存器构成的一般都是异步时序逻辑电路。

同步时序逻辑电路在电路设计中非常重要，绝大部分的电路都是

同步时序逻辑电路，而 D 触发器又是同步时序逻辑电路的核心器件，D 触发器的重要性不言而喻。

**·搭建寄存器**

寄存器本质上来说就是 D 触发器，如下图所示，我们用 4 个 D 触发器构成了一个能够存储 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*

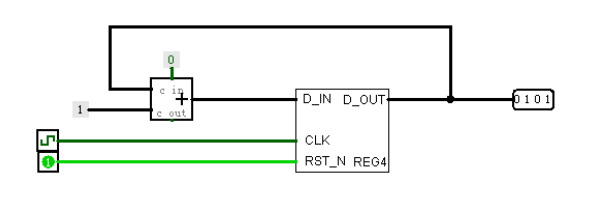
*endmodule*

*（说明：对于多 bit 位宽的信号，在 Verilog 中使用“[x:y]”这种方式声明，例如上述代码中，D\_OUT 就是一个 4bit 的信号，它包含了 D\_OUT[0]、D\_OUT[1]、D\_OUT[2]、D\_OUT[3]四个单 bit 信号。）*

由于上述的电路存在复位问题，所以仅供理解。

**·搭建简单的时序逻辑电路**

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;*

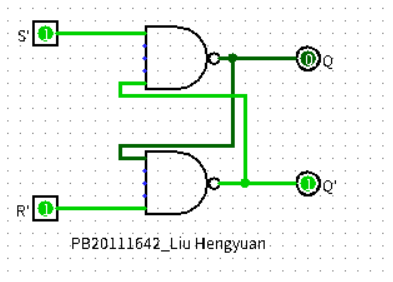
*end*

*endmodule*

【实验练习】

·题目一：

搭建SR锁存器，如图所示：

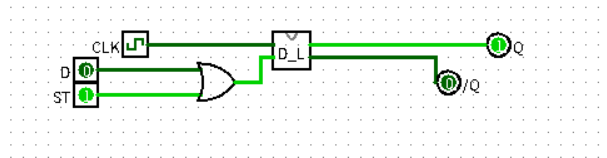


电路输入状态如下表：

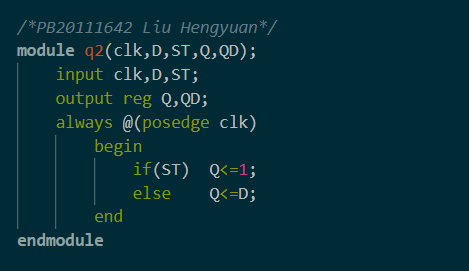


·题目二：

搭建同步置位的D 触发器如图所示：

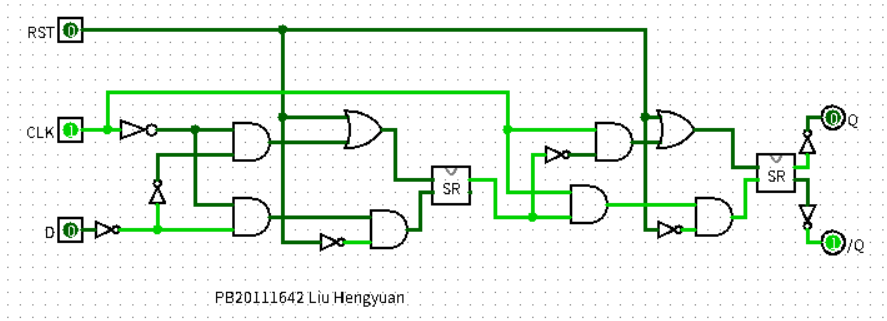


代码如下：

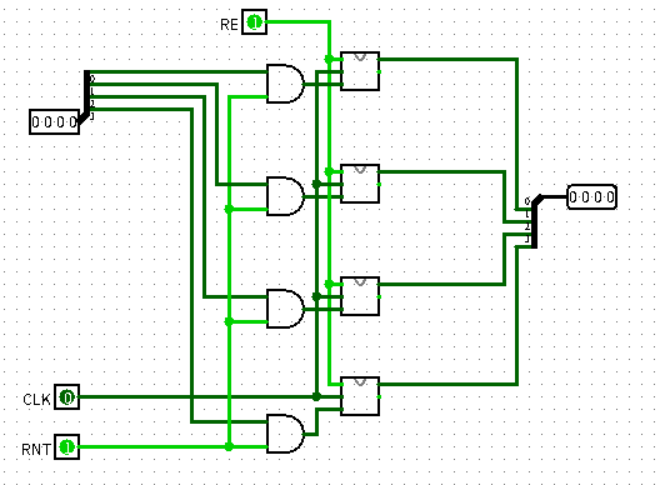


·题目三：

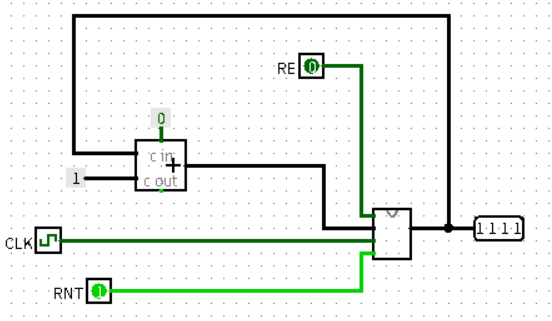
搭建异步复位的D触发器如图所示：



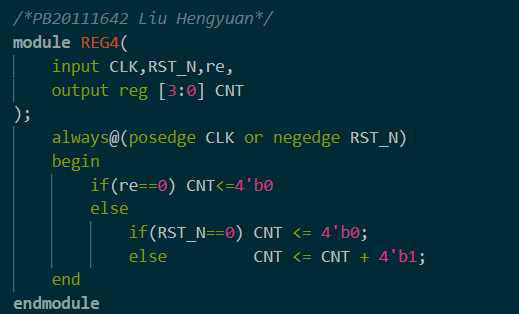
用上述触发器构建寄存器如图：



用寄存器构建的计数器如图：

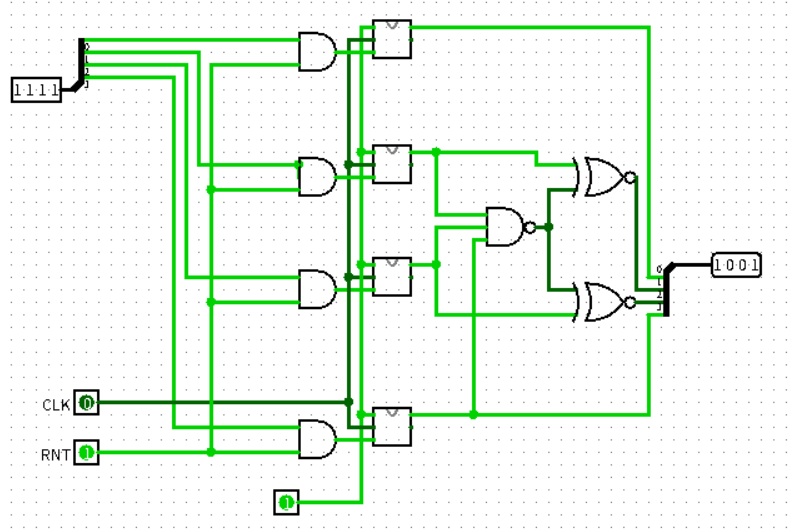


代码如下：



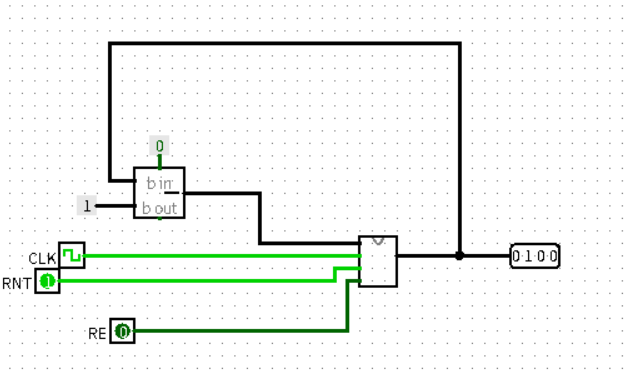
·题目四：

构建寄存器（仍使用题目三中异步复位触发器构建）：

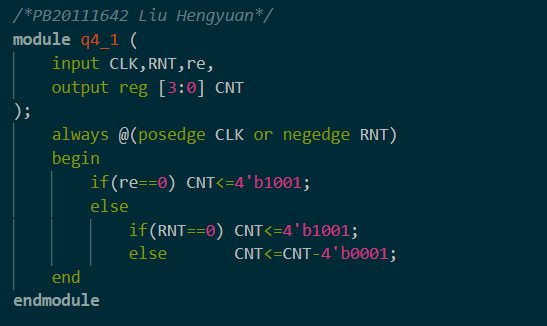


原理：当递减至0000时，返回输入1111；此时1111通过寄存器，输出1001，进行下一个循环。

构建计数器，如图：

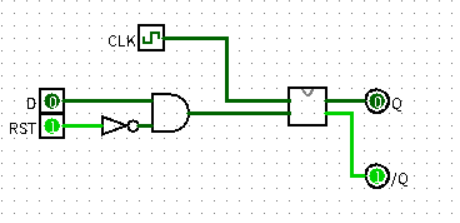


代码如下：



·题目五：

为触发器添加复位信号：



当复位信号有效时（高电平有效时），输出信号Q始终为0.

代码如下：



【总结与思考】

个人认为本次实验难度较大，在各种概念的理解上会有比较大的困难，所以操作时可能会出现各种问题。所以必须深刻地理解这些相似的概念，并且加以区分，才能熟练地运用。