

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | 计算机逻辑设计基础 |
| 姓 名： | 汪珉凯 |
| 学 院： | 计算机科学与技术学院 |
| 专 业： | 人工智能 |
| 邮 箱： | 904738557@qq.com |
| QQ 号： | 904738557 |
| 电 话： | 18157421318 |
| 指导教师： | 王总辉 |
| 报告日期： | 2023年 12月 10 日 |

**浙江大学实验报告**

课程名称： 计算机逻辑设计基础 实验类型： 综合

实验项目名称： 加法器、加减法器和ALU基本原理与设计

学生姓名： 汪珉凯 学号： 3220100975 同组学生姓名： 岳晟稼

实验地点： 紫金港东四509室 实验日期： 2023 年 11 月 9 日

# 实验目的和要求

1. 掌握一位全加器的工作原理和逻辑功能
2. 掌握串行进位加法器的工作原理和进位延迟
3. 掌握减法器的实现原理
4. 掌握加减法器的设计方法
5. 掌握 ALU 基本原理及在 CPU 中的作用
6. 掌握 ALU 的设计方法

# 实验内容和原理

## 2.1实验设备

* 1. 装有Xilinx ISE 14.7的计算机 1台
  2. SWORD开发板 1套

## 2.2实验内容

任务1：原理图方式设计4位加减法器

任务2：实现4位ALU及应用设计

## 2.3实验原理

**2.3.1 1位全加器**

三个输入位：数据位 Ai 和 Bi，低位进位输入 Ci

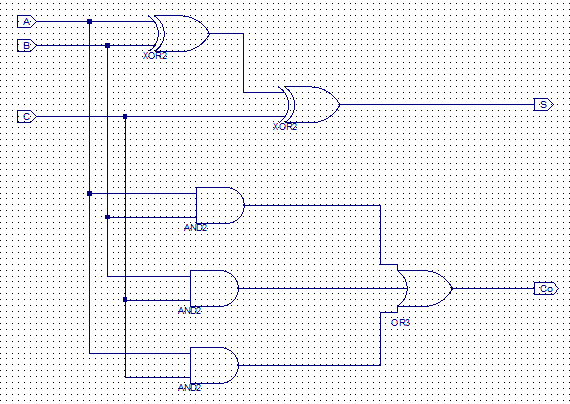
二个输出位：全加和 Si，进位输出 Ci+1

逻辑表达式如下所示：

根据逻辑表达式，得到真值表：



根据一位全加器的输入输出关系，得到电路图：



一位全加器电路图

其逻辑符号表达式如下：



一位全加器逻辑符号图

对应的verilog代码如下：

module adder\_1bit(

    input wire a, b, ci,

    output wire s, co);

    and m0(c1,a,b);

    and m1(c2,b,ci);

    and m2(c3,a,ci);

    xor m3(s1,a,b);

    xor m4(s,s1,ci);

    or  m5(co,c1,c2,c3);

endmodule

**2.3.2 多位串行进位加法器**

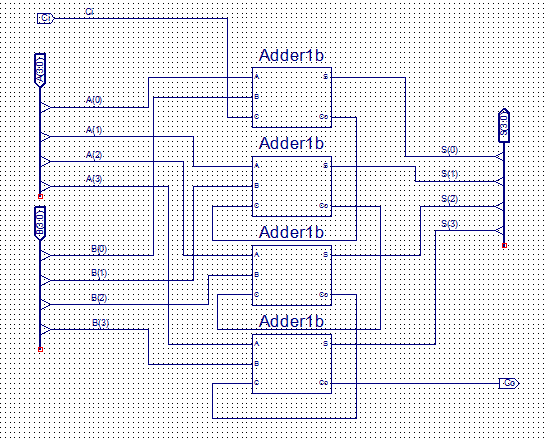
组成：

* 由一位全加器将进位串接构成
* 低位进位为0，为高位进位输出

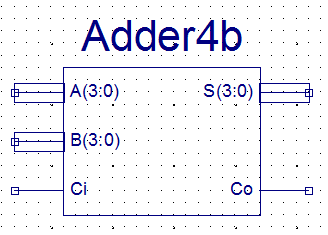


**实例：4位全加器**

原理图：



Symbol图示：



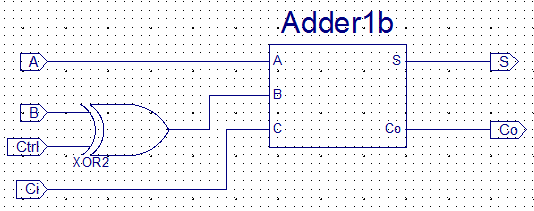
**2.3.3一位加减法器：**

**用负数补码加法实现，减数当作负数求补码**

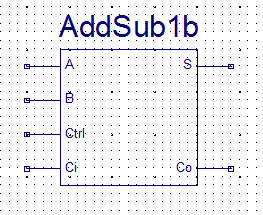
**共用加法器**

**用“异或”门控制求反，低位进位C0为1**

原理图：

****

Symbol图：



**2.3.4多为串行进位全减器**

**用复数补码加法实现，减数当作负数求补码**

**公用加法器**

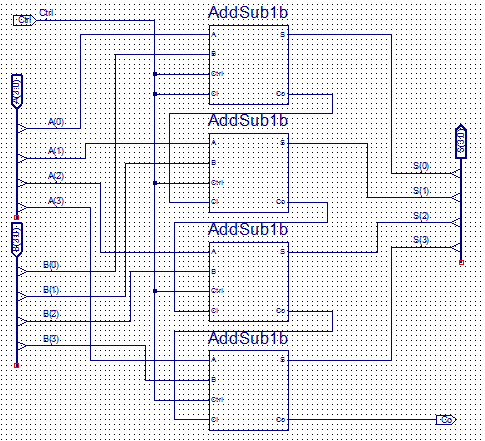
**用“异或”门控制求反，地位进位C0为1**

****

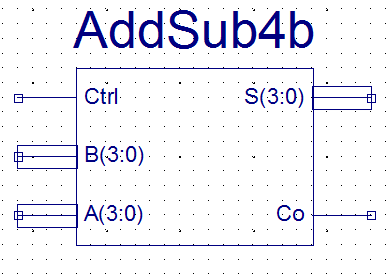
**关于表达式：**

****

1. **4位加减法器的图：**

****

**Symbol：**

****

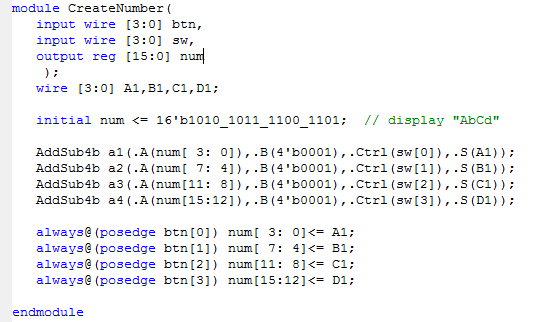
**2)8位加减法器代码实现：**

****

**设计案件数据输入模块**

**使用行为描述设计**

**在实验7基础上，更新Abber4b为AddSub4b模块**

****

**2.4 按键去抖动原理**

**抖动原因：按键按下或放开时，存在机械振动**

**抖动时间一般在10~20ms**

**按键去抖动方法：延时，以避开机械抖动**

****

****

**2.4.1防抖动模块 + 分频器**

**代码：**module pbdebounce(

    input wire clk\_1ms,

    input wire button,

    output reg pbreg

    );

    reg [7:0] pbshift;

    always@(posedge clk\_1ms) begin

        pbshift=pbshift<<1;

        pbshift[0]=button;

        if (pbshift==8'b0)

            pbreg=0;

        if (pbshift==8'hFF)

            pbreg=1;

    end

endmodule //button要连续按8ms才有效输出一个pbreg.

**pbshift寄存器数据变化情况**

****

**（1/100mhz）\*X=1ms-->x=100MHz\*1ms=100\*106\*1\*10-3 =105s**

**217 =131072,=1.3\*105=1.3ms**

**clkdiv[16]=217**

**辅助模块：时钟计数分频器**

module clkdiv(input wire clk,output reg[31:0] clkdiv=0);

    always @(posedge clk) //clkdiv[0] 第1个L->h,第2个H->L

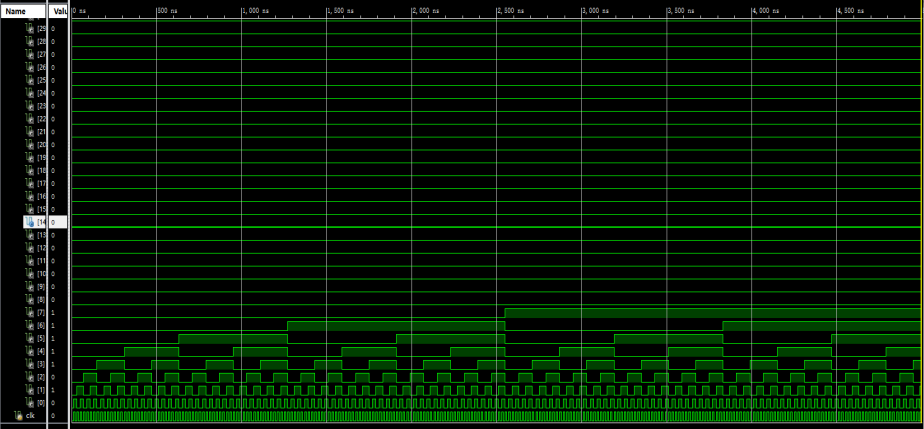
   begin

     clkdiv <=clkdiv + 1'b1;

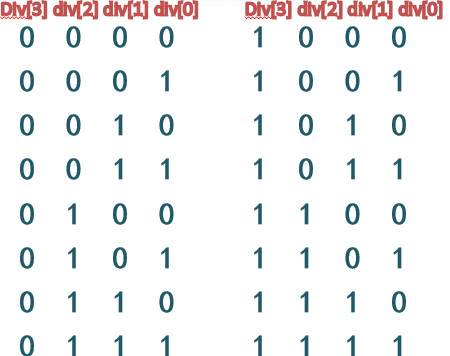
   end

endmodule

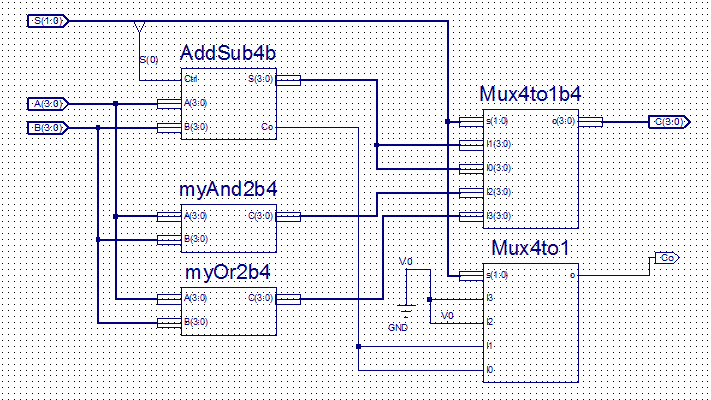
**仿真图：**

****

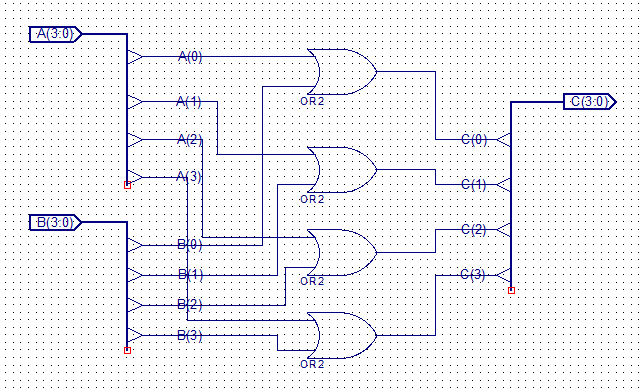
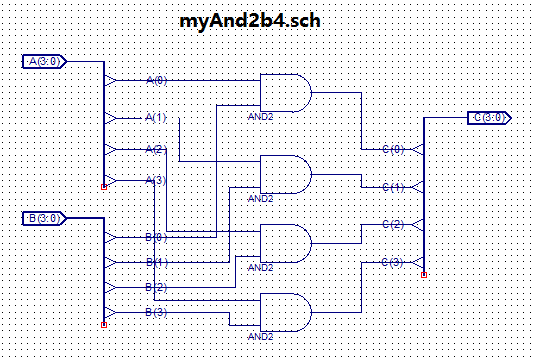
**时钟计数分频器-clkdiv[3:0]**

****

**4位ALU原理图**

****

**四位与门或门模块图：**

****

**仿真代码：**

integer i;

initial begin

A=4'b1010;B=4'b0111;#100;

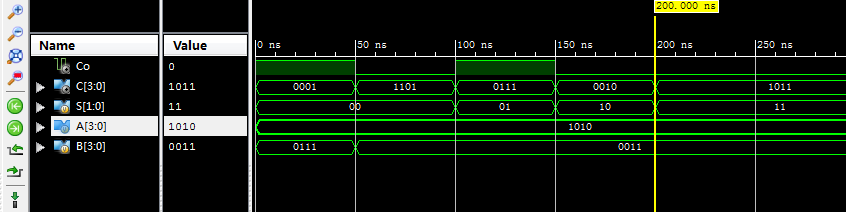
B=4'b0011;

for (i=0; i<=3;i=i+1) begin

     #100;S=i;end

end

**仿真结果：**

****

# 三、实验过程和数据记录

3.1实验内容：

任务1：原理图方式设计4位加减法器：

* **新建工程**
  + **工程名称用MyALU。**
  + **Top Level Source Type用HDL**
* **新建源文件**
  + **类型是Schematic**
  + **文件名称用AddSub1b。**
* **原理图方式进行设计**
* **新建源文件**
  + **类型是Schematic**
  + **文件名称用AddSub4b**
* **原理图方式进行设计，调用前面设计的AddSub1b**
* **进行波形仿真，激励输入至少4组**

任务2：实现4位ALU及应用设计

ALU设计：

* **新建源文件**
  + **类型是Verilog或Schematic**
  + **文件名称用ALU**
* **原理图方式进行设计**
* **进行波形仿真**
  + **激励输入至少4组**
  + **覆盖4种操作**

ALU应用设计：

* **新建源文件**
  + **类型是Verilog，文件名Top。**
  + **右键设为“Set as Top Module”**
* **代码输入进行设计**
  + **调用pbdebounce模块**
  + **调用AddSub4b模块**
  + **调用pbdebounce、clkdiv模块**
  + **调用DispNum模块**
  + **调用CreateNumber模块**
* **业务逻辑要求**
  + **两个4位操作数A，B**
  + **可用两个按键进行自增/减**
  + **得到结果C和进位Co**
  + **把A、B、C和Co动态显示**

Verilog代码：

integer i;

initial begin

A=4'b1010;B=4'b0111;#100;

B=4'b0011;

for (i=0; i<=3;i=i+1) begin

     #100;S=i;end

end

    module top(

    input wire clk,

             Input wire [1:0] BTN,//实现A,B两个操作数每按一下加1

    input wire [1:0]SW1,//SW1[1]:控制B数的增或减

                                                   //SW1[0]:控制A数的增或减

    input wire [1:0]SW2,//SW2 00:加,01:减,10:AND,11:OR

             input wire [11:0] SW,

    output wire [3:0]AN,

    output wire [7:0]SEGMENT,

             output wire BTNX4,

                output wire seg\_clk,

    output wire seg\_clrn,

    output wire seg\_sout,

    output wire SEG\_PEN

    );

               wire [15:0] num;

    wire [1:0] btn\_out;

    wire [3:0] C;

    wire Co;

    wire [31:0] clk\_div;

               wire [15:0] disp\_hexs;

            assign disp\_hexs[15:12] = num[7:4];//B

    assign disp\_hexs[11:8] = num[3:0]; //A

    assign disp\_hexs[7:4] = {3'b000, Co};

    assign disp\_hexs[3:0] = C[3:0];    //结果C

                assign BTNX4 = 1'b0;

  clkdiv m2(clk,0,clk\_div);

  pbdebounce m0(clk\_div[17],BTN[0],btn\_out[0]);//BTN[0]去抖动

   pbdebounce m1(clk\_div[17],BTN[1],btn\_out[1]);

   CreateNumber m3(btn\_out, SW1, num);

   myALU m5(.S(?), .A(num[3:0]), .B(?), .C(?), .Co(Co));

                       //A,B,C,C0不可以用disp\_hexs，否则有多个源问题

Disp\_Number  m6(.clk(clk), .HEXS(disp\_hexs), .LES(4'b????), .point(4'b????),.RST(1'b?), .AN(AN), .Segment(SEGMENT));

Sseg\_Devm7(.clk(clk),.rst(1'b0),.Start(clk\_div[20]),.flash(clk\_div[25]),.Hexs({disp\_hexs,disp\_hexs}),.point({4'b0000,SW[3:0]}),.LES(SW[11:4]),.seg\_clk(seg\_clk),.seg\_clrn(seg\_clrn),.seg\_sout(seg\_sout),.SEG\_PEN(SEG\_PEN));

//默认point,LES参数是传0，LES传1则a,b,c,d,e,f,g都不亮了

UCF定义：

* 输入
* 时钟: clk
* 按键控制输入: BTN11 控制 A，BTN[O] 控制 B，关联到 BTNX4Y[0:1]按键加/减方向控制: SW11] 控制 A，SW1(0] 控制 B，关联到 DSW[1:0ALU 运算控制:SW21:01,00-加，01-减，10-与，11-或，关联到 DSW115:14
* 输出
* 数码管[0]: A - num3:0
* 数码管[1]: B - num[7:4]
* 数码管 [2]: Co- Co
* 数码管3: C -C

BTNX4:按键使能输出

#七段码串行移位接口

NET "seg\_clk"        LOC = M24 | IOSTANDARD = LVCMOS33 ;

NET "seg\_clrn"       LOC = M20 | IOSTANDARD = LVCMOS33 ;

NET "seg\_sout"       LOC = L24 | IOSTANDARD = LVCMOS33 ;

NET "SEG\_PEN"        LOC = R18 | IOSTANDARD = LVCMOS33 ;

3.2实验结果：

演示A的自增功能：



演示B的自增功能：



演示A的自减功能：



演示B的自减功能：



# 讨论心得：

这次实验被同组队员带飞，主要收获点在于研究了ucf的具体写法，然后对alu本身有了更加透彻的了解。

**浙江大学实验报告**

课程名称： 计算机逻辑设计基础 实验类型： 综合

实验项目名称： 锁存器与触发器基本原理

学生姓名： 汪珉凯 学号：3220100975 同组学生姓名： 岳晟稼

实验地点： 紫金港东四509室 实验日期： 2023 年 11 月 23 日

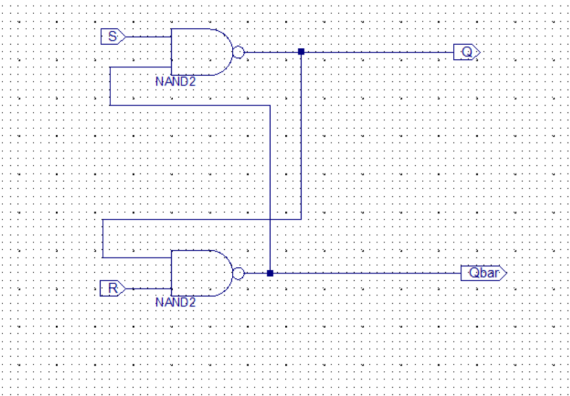
1. **操作方法与实验步骤**
2. **实现基本SR锁存器，验证功能和存在的时序问题**

**1、新建工程**

新建工程，工程名命名为 experiment9。

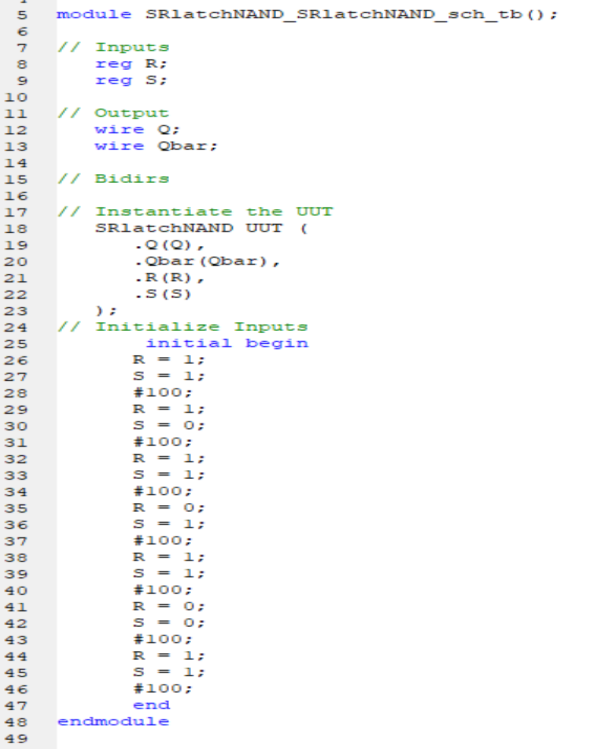
**2、新建 Schematic 源文件设计电路**

新建 Schematic 源文件，文件命名为 SRlatchNAND，并根据电路图绘图。新建好的源文件如下：



**3. 仿真**

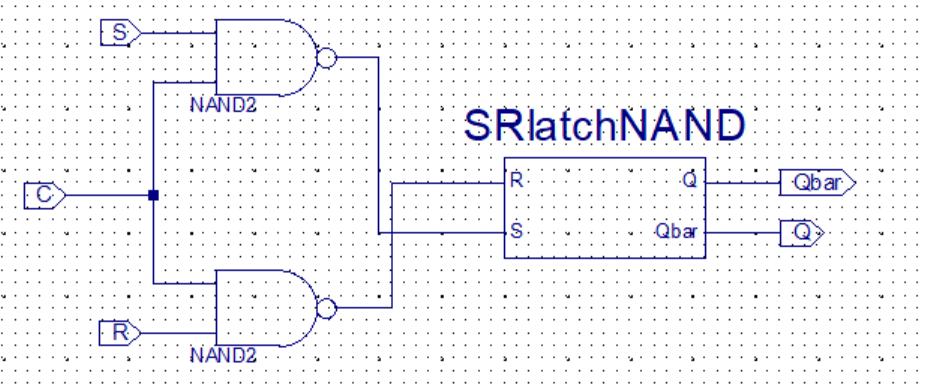
建立仿真模拟文件：testSR.v，代码如下：



1. **实现门控SR锁存器，并验证功能和存在的时序问题。**

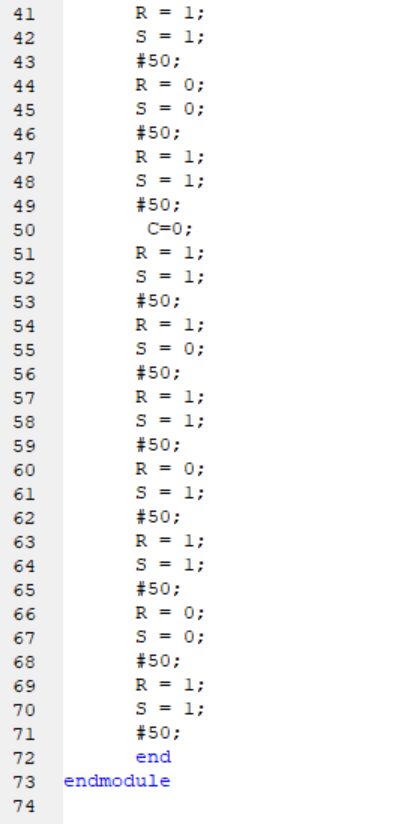
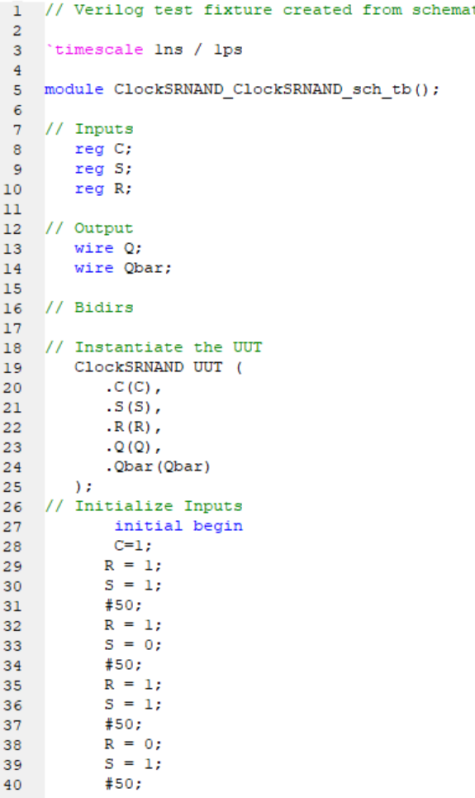
**1.新建 Schematic 源文件设计电路**

新建 Schematic 源文件，文件命名为 ClockSRNAND，并根据电路图绘图。源文件如下：



1. **仿真**

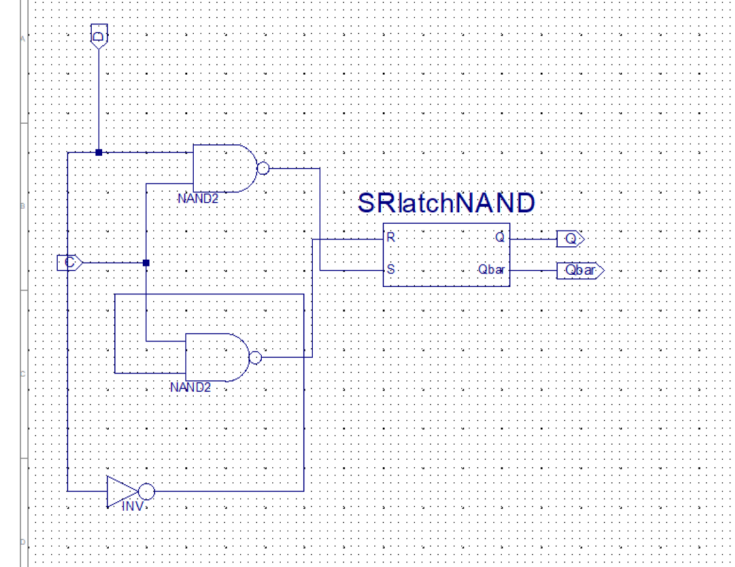
建立仿真模拟文件：CSRtest.v，代码如下：



1. **D锁存器，并验证功能和存在的时序问题**

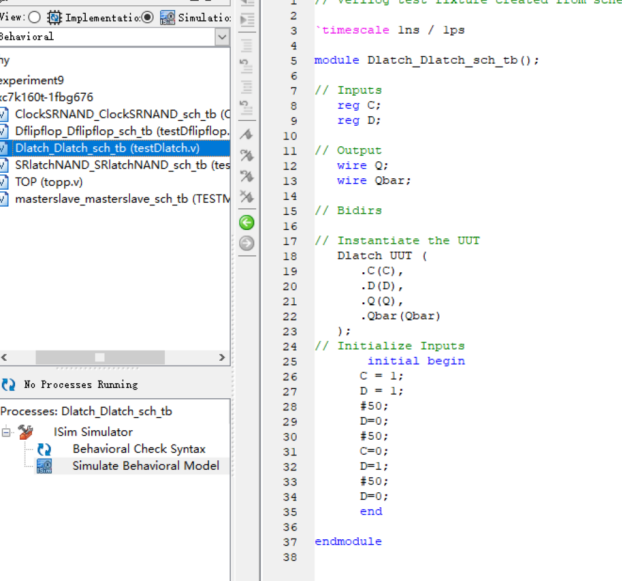
**1.新建 Schematic 源文件设计电路**

新建 Schematic 源文件，文件命名为 Dlatch，并根据电路图绘图。源文件如下：

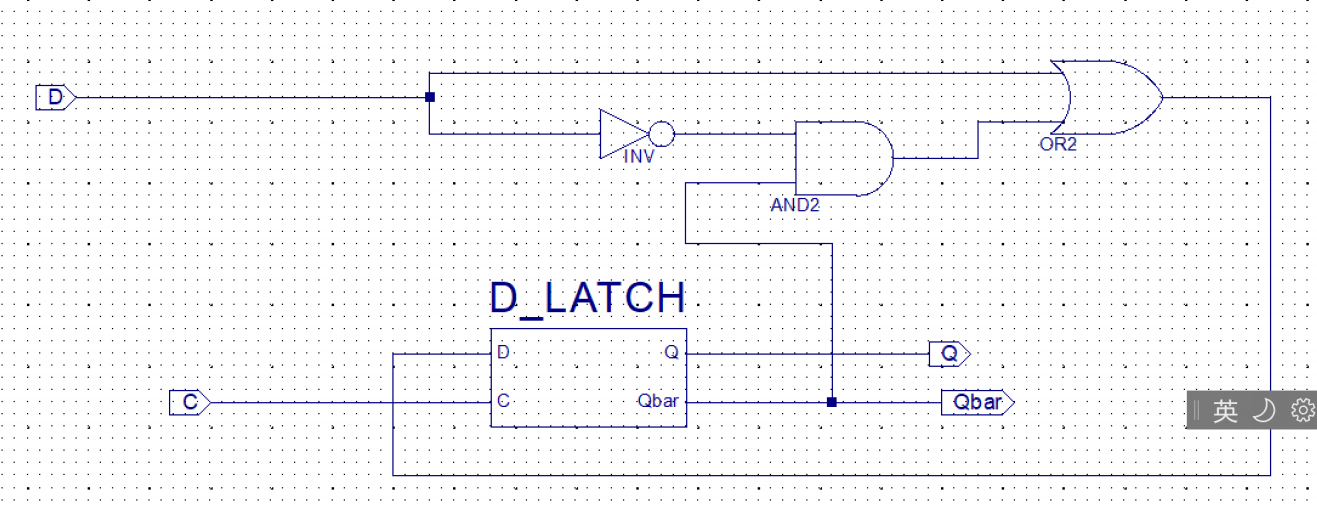


**2.仿真**

建立仿真模拟文件：testDlatch.v，代码如下：



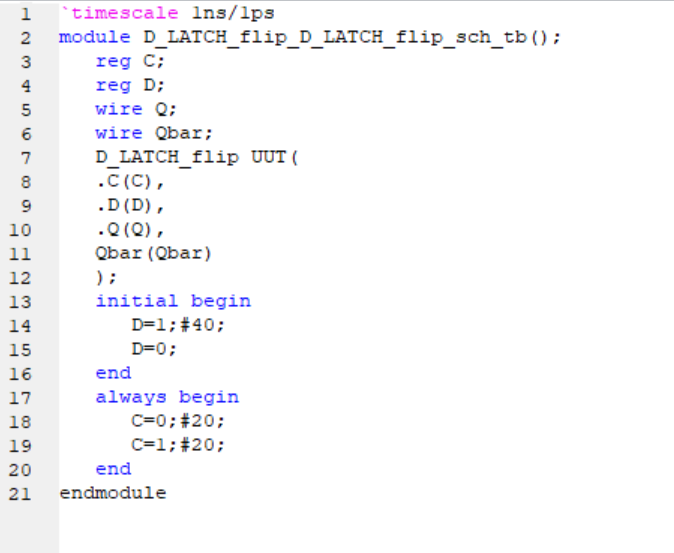
1. **搭建电路进行D锁存器的空翻验证**

新建 Schematic 源文件，文件命名为 Dlatch，并根据电路图绘图。源文件如下：

先令D=1，则Qbar=0；根据此电路，D与Qbar相连，两者的值始终相反，会产生一个振荡电路，因此可用于验证D的空翻现象。

**4.仿真**

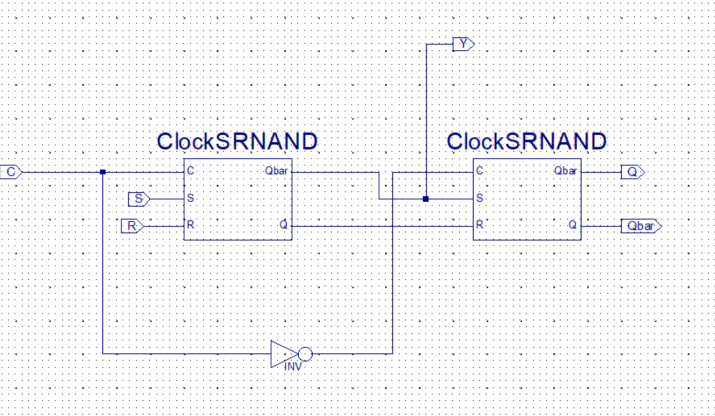
建立仿真模拟文件：testDlatch\_flip.v，代码如下:



1. **SR主从触发器，并验证功能和存在的时序问题**

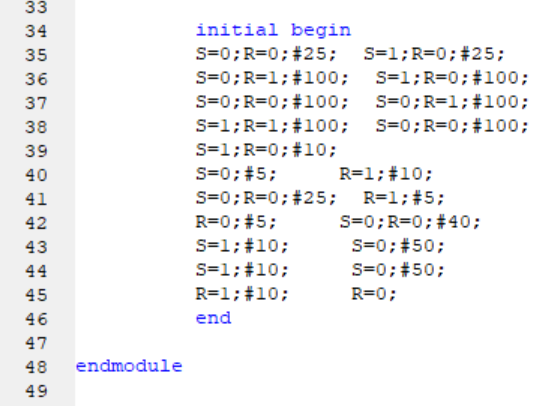
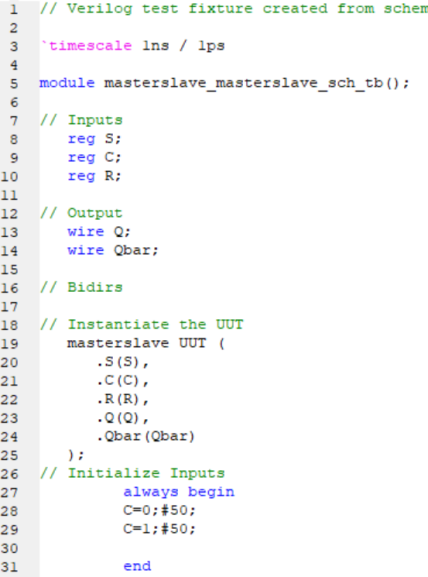
**1.新建 Schematic 源文件设计电路**

新建 Schematic 源文件，文件命名为 masterslave，并根据电路图绘图。源文件如下：



**2.仿真**

建立仿真模拟文件：TESTMASTERSLAVE.v，代码如下：



1. **D触发器，并验证功能**

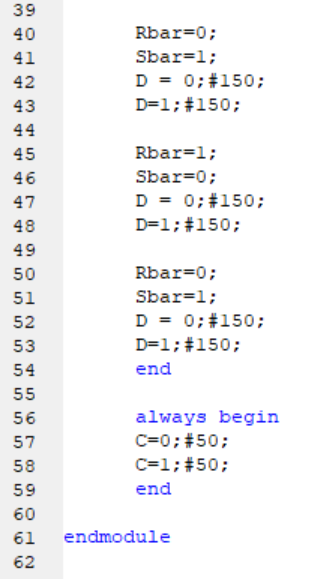
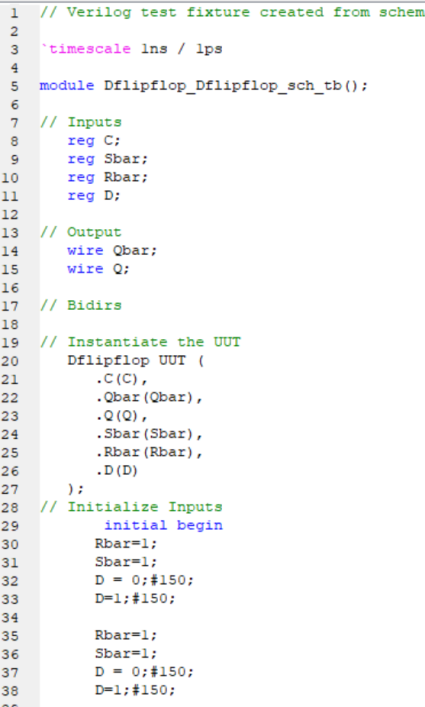
**1.新建 Schematic 源文件设计电路**

新建 Schematic 源文件，文件命名为 Dflipflop，并根据电路图绘图。源文件如下：



**2.仿真**

建立仿真模拟文件：testDflipflop.v，代码如下：



1. **设计顶层模块**

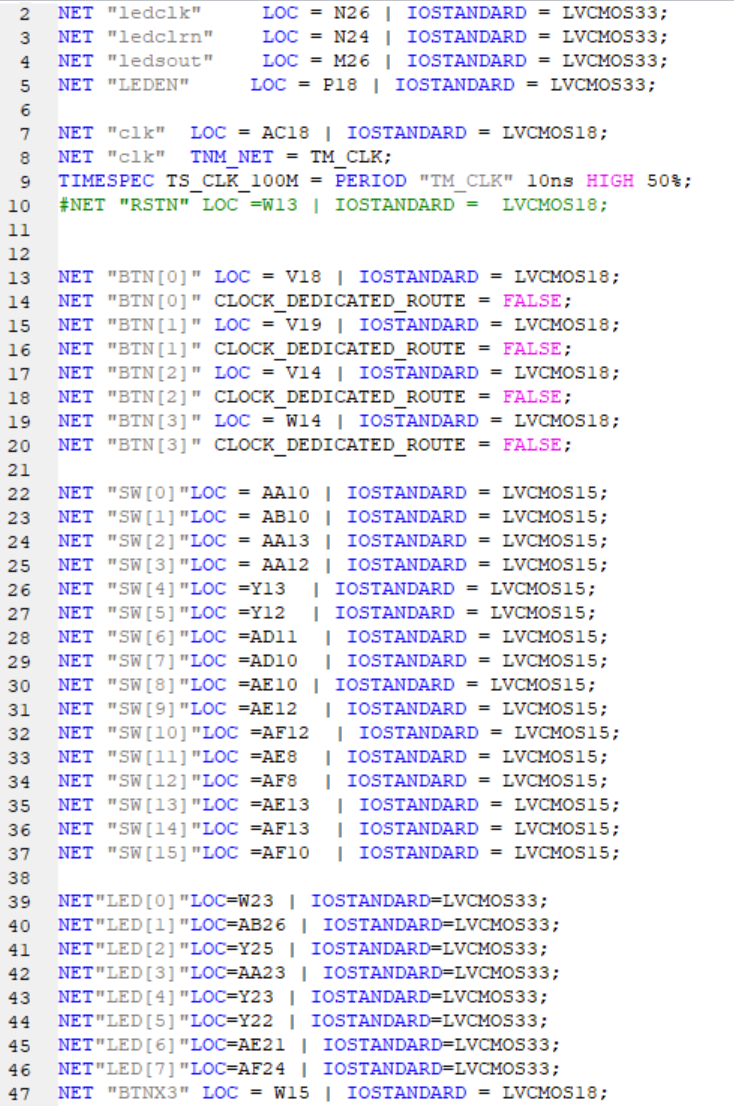
**1.新建 HDL源文件设计电路**

新建 HDL源文件，文件命名为 TOP.v，并根据电路图绘图。源文件如下：



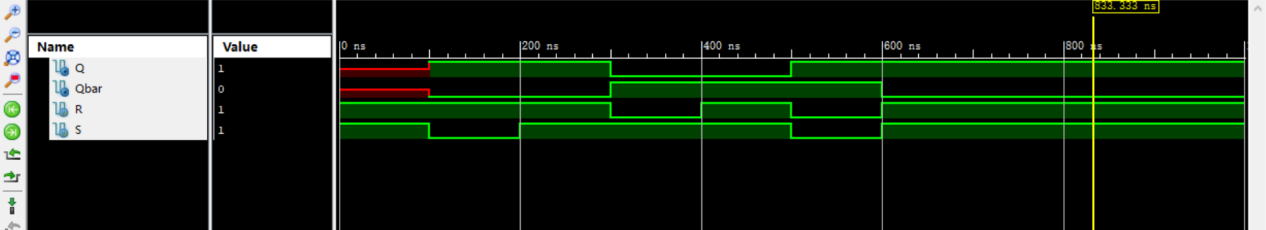
1. 设计UCF引脚并上板

Ucf文件代码如图所示：



1. **实验结果与分析**

**（一）基本SR锁存器的仿真结果**



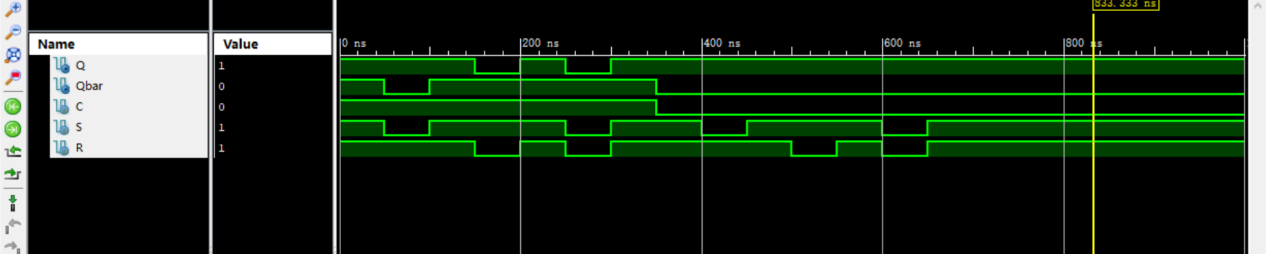
开始时锁存器中的存储值是未知的。

S=0,R=1时，置位Q=1；S=1,R=0时，复位Q=0.

S=0,R=0时，Q与Qbar结果取决于门延迟。

仿真结果和SR锁存器真值表相同，符合预期。

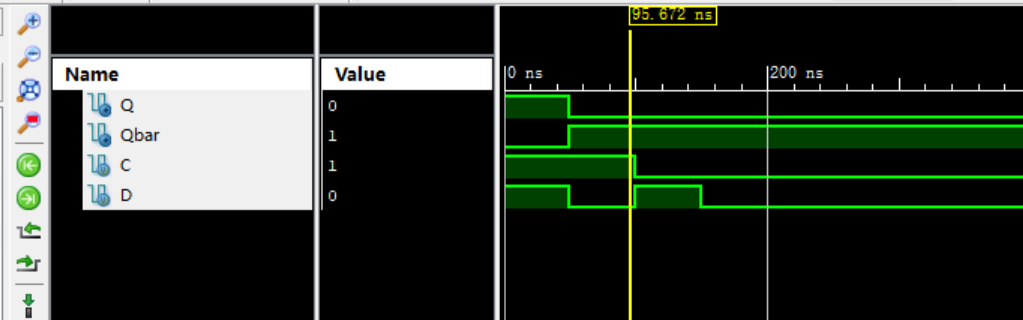
**（二）门控SR锁存器的仿真结果**



C=1时，门控SR锁存器等效于基本SR锁存器，即输出Q由SR决定；

C=0时，输出Q不会随着S和R的变化而发生改变。

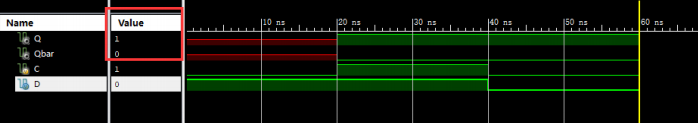
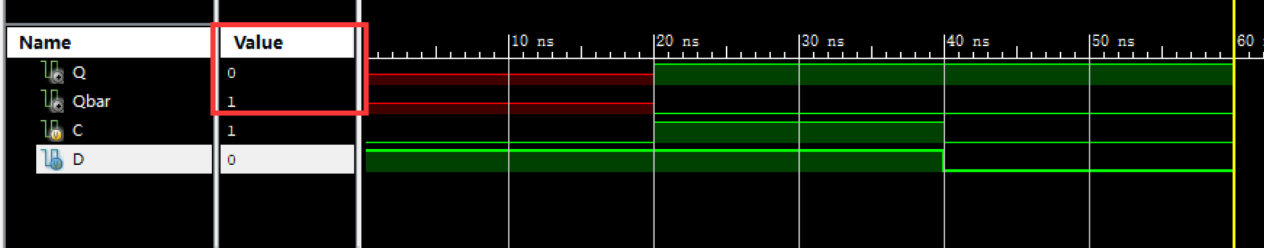
仿真结果和门控SR锁存器真值表相同，符合预期。

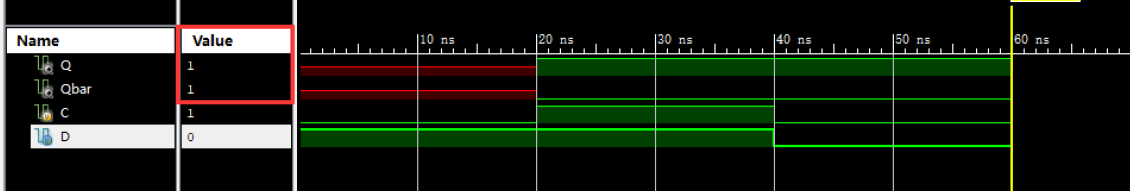
1. **D锁存器的仿真结果**

C=1时，若Q=D；C=0时，Q不随D的改变而变化。

仿真结果和D锁存器真值表相同，符合预期。

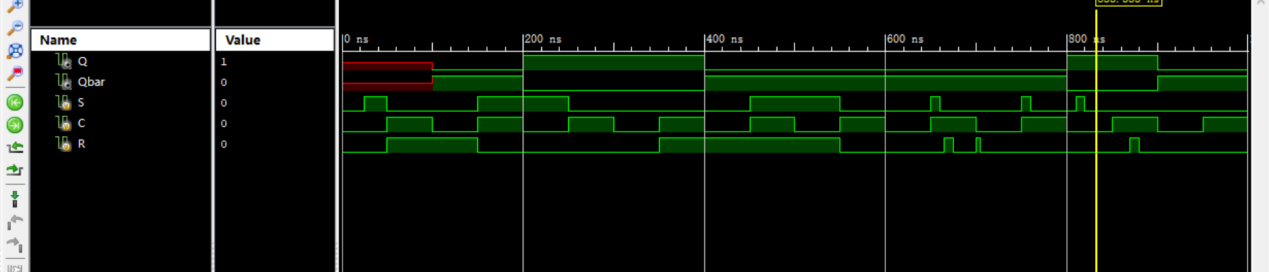
1. **D锁存器的空翻仿真结果**





60ns后，电路进入了震荡状态，在某一时刻点击查看Q的值，会得到不同的情况，符合对空翻现象的预期。

1. **SR主从触发器的仿真结果**

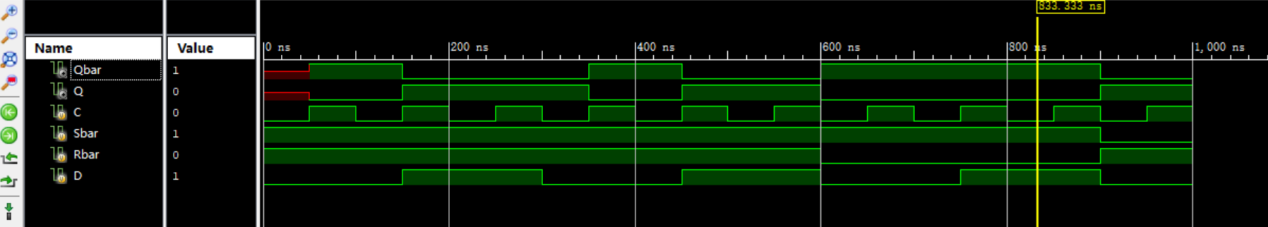


本次仿真同时完成了对SR主从触发器功能的验证以及对一次性采样问题的验证。

当S=0，R=1时，Q=0；当S=1，R =0时Q=1；当S=1，R=1时，Q可能是0或1。

然后验证一次性改样问题：在800ns时S=R=0，给S一个时间极短的正边沿小脉冲，则master部分由0变1，即使在S很快恢复原样时依然如此。这就是SR主从触发器的一次性采样问题。

1. **D触发器的仿真结果**



R=S=1时，Q=D；R=0，S=1时，Q=0；R=1，S=0时，Q=1。

仿真结果和D触发器真值表相同，符合预期。

1. **顶层模块上板验证**

前提：默认开关均置0.

|  |  |
| --- | --- |
| 0号开关置1、1号开关置0 | 0号开关置0、1号开关置1 |
|  |  |
| 0号灯亮、1号灯灭 | 0号灯灭、1号灯亮 |

|  |  |
| --- | --- |
| 0号开关、1号开关均置1 | 2号开关置1 |
|  |  |
| 0号灯常量，1号灯闪烁 | 2号灯灭，3、6号灯亮，5号灯闪烁  （原来2号开关置0时：  2号灯亮，三号灯灭，5、6号灯亮。） |

|  |  |
| --- | --- |
| 2号开关、3号开关均置1 | 4号开关置0 |
|  |  |
| 2号灯亮，3、5、6号灯均熄灭 | 5、6、7号灯全部熄灭 |

1. **讨论、心得**

本次实验上板内容较少，以仿真模拟为主，内容相对比较简单，只需要验证几个寄存器和触发器的功能以及空翻和一次性采样的现象。我认为本次实验的难点在于引脚意义的理解。希望下次实验前能够做好预习工作，这样课上的效率会得到提高。再接再厉。

**浙江大学实验报告**

课程名称： 计算机逻辑设计基础 实验类型： 综合

实验项目名称： 七段数码管显示译码器设计与应用

学生姓名： 岳晟稼 学号： 3220105673 同组学生姓名： 汪珉凯

实验地点： 紫金港东四509室 实验日期： 2023 年 11 月 16 日

# 实验目的和要求

1. 掌握典型同步时序电路的工作原理和设计方法
2. 掌握时序电路的激励函数、状态图、状态方程的运用
3. 掌握用Verilog进行有限状态机的设计、调试、仿真
4. 掌握用FPGA实现时序电路功能

# 实验内容和原理

## 2.1实验设备

* 1. 装有Xilinx ISE 14.7的计算机 1台
  2. SWORD开发板 1套

## 2.2实验内容

任务1：原理图方式设计4位同步二进制计数器

任务2：以Verilog行为描述方式设计16位可逆二进制同步计数器

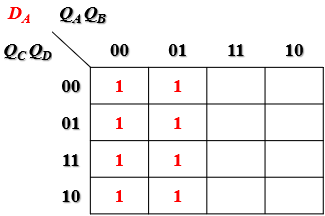
## 2.3实验原理

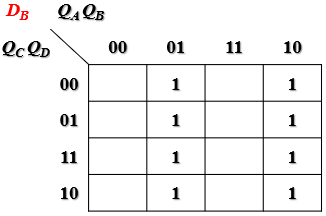
**2.3.1 4位同步二进制计数器**

4位同步二进制计数器状态表



根据D触发器原理，在clk作用下Q=D,4位计数器的Q和D的关系如左表:





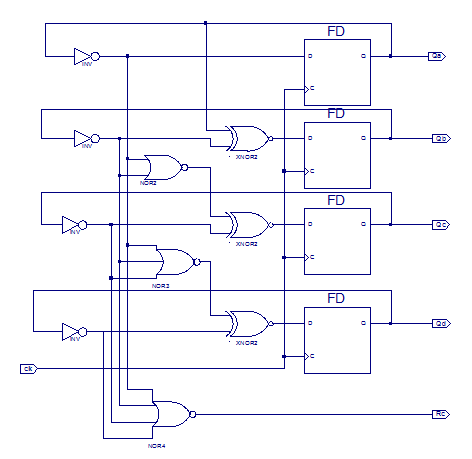




进位Rc的输出函数：



原理图：



Verilog代码：

module counter\_4bit(clk, Qa, Qb, Qc, Qd, Rc);

input wire clk;

output wire Qa, Qb, Qc, Qd, Rc;

wire Nor\_nQa\_nQb, Nor\_nQa\_nQb\_nQc;

//FD原程序的输入输出变量名：C,D,Q

FD FD\_A(.C(clk), .D(Da), .Q(Qa)),

   FD\_B(.C(clk), .D(Db), .Q(Qb)),

   FD\_C(.C(clk), .D(Dc), .Q(Qc)),

   FD\_D(.C(clk), .D(Dd), .Q(Qd));

defparam FD\_A.INIT = 1'b0, FD\_B.INIT = 1'b0;//对应parameter

defparam FD\_C.INIT = 1'b0, FD\_D.INIT = 1'b0;

INV nQa\_L(.I(Qa), .O(nQa)), nQb\_L(.I(Qb), .O(nQb)),

    nQc\_L(.I(Qc), .O(nQc)), nQd\_L(.I(Qd), .O(nQd));

assign Da = nQa;//数据流描述

XNOR2 Db\_L(.I0(Qa),              .I1(nQb), .O(Db)),

      Dc\_L(.I0(Nor\_nQa\_nQb),     .I1(nQc), .O(Dc)),

      Dd\_L(.I0(Nor\_nQa\_nQb\_nQc), .I1(nQd), .O(Dd));

NOR4  Rc\_L(.I0(nQa), .I1(nQb), .I2(nQc), .I3(nQd), .O(Rc));

NOR2  Nor\_nQa\_nQb\_L    (.I0(nQa), .I1(nQb), .O(Nor\_nQa\_nQb));

NOR3  Nor\_nQa\_nQb\_nQc\_L(.I0(nQa), .I1(nQb), .I2(nQc), .O(Nor\_nQa\_nQb\_nQc));

endmodule

4位二进制同步计数器仿真：

initial forever begin

    clk = 1’b0; #100;

    clk = 1’b1; #100;

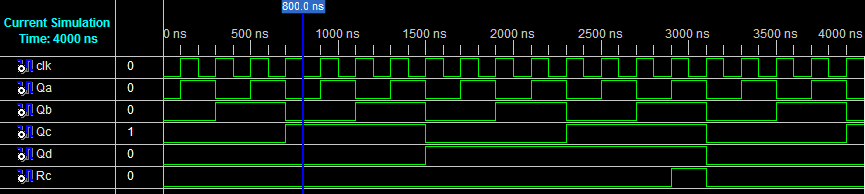
end

/\*  always begin

        #100 clk = 0;

        #100 clk = 1;

end \*/



**2.3.2可逆二进制同步计数器**

可逆二进制同步计数器通过控制端S选择正向或者反向计数

* S = 1时，正向计数，各触发器逻辑表达式同前面
* S = 0时，反向计数，各触发器逻辑表达式如下式

****

Verilog代码行为描述：

module counter\_4bit\_rev(clk, s, cnt, Rc);

input wire clk, s;

output reg [3:0] cnt;

output wire Rc;

Initial begin cnt = 0;

assign Rc = (~s & (~|cnt)) | (s & (&cnt));//数据流描述

always @ (posedge clk) begin              //行为描述

    if (s)

        cnt <= cnt + 1;

  //always中被赋值的变量cnt一定为reg量型。

    else

        cnt <= cnt - 1;

end

endmodule

2.3.3分频器设计

100MHz信号通过550000000次分频后，得到1Hz的秒脉搏方波，作为计数器的脉冲输入

module counter\_1s(clk, clk\_1s);

input wire clk;

output reg clk\_1s;

reg [31:0] cnt;//问题：reg [15:0]cnt可以吗？

always @ (posedge clk) begin

   if (cnt < 50\_000\_000) begin

       cnt <= cnt + 1;

   end else begin

       cnt <= 0;

       clk\_1s <= ~clk\_1s;

   end

end

endmodule

# 三、实验内容与步骤

任务1：原理图方式设计4位同步二进制计数器

* **新建工程**
  + **工程名称用MyCounter。**
  + **Top Level Source Type用HDL**
* **新建源文件**
  + **类型是Schematic**
  + **文件名称用Counter4b。**
* **原理图方式进行设计**
* **进行波形仿真**

波形仿真图：

* **新建源文件，用作时钟**
  + **类型是Verilog**
  + **文件名称用clk\_1s**
* **Verilog行为描述**
* **新建源文件**
  + **类型是Verilog**
  + **文件名称用Top。**
  + **右键设为“Set as Top Module”**
* **输入为clk（100MHZ）时钟**
* **每秒自增1//根据“分频器设计”程序得到1s时钟**
* **显示在1位数码管上//AN0,AN1,AN2,AN3中选一个**

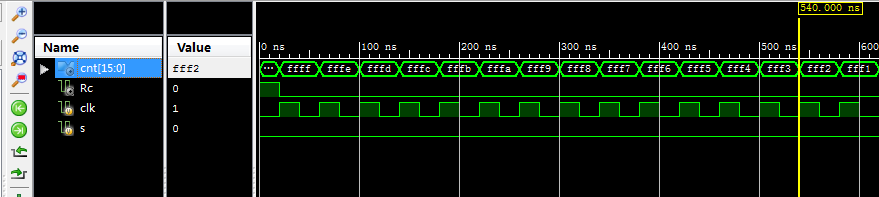
**//其他数码管位可以显示0或者不显示，最好不显示**

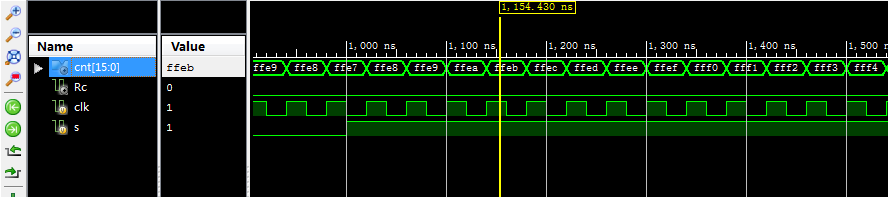
* **Rc显示在LED灯上**

任务2：以Verilog行为描述方式设计16位可你二进制同步计数器

* **新建工程**
  + **工程名称用myRevCounter。**
  + **Top Level Source Type用HDL**
* **新建源文件**
  + **类型是Verilog**
  + **文件名称用RevCounter。**
* **结构化描述方式进行设计**
* **波形仿真（包含正向计数和反向计数）**

波形仿真图：





* **新建源文件，设计100ms时钟**
  + **类型是Verilog**
  + **文件名称用clk\_100ms**
* **Verilog行为描述**
* **新建源文件**
  + **类型是Verilog，文件名称用Top。**
  + **右键设为“Set as Top Module”**
* **输入为clk（100MHZ）时钟**
* **用sw[0]控制自增/自减1（每0.1秒）**
* **显示在4位数码管上**
* **Rc状态用LED灯来显示**

1. 引脚约束

对输入输出进行引脚约束。

#七段码串行移位接口

NET "seg\_clk"       LOC = M24 | IOSTANDARD = LVCMOS33 ;

NET "seg\_clrn"      LOC = M20 | IOSTANDARD = LVCMOS33 ;

NET "seg\_sout"      LOC = L24 | IOSTANDARD = LVCMOS33 ;

NET "SEG\_PEN"       LOC = R18 | IOSTANDARD = LVCMOS33 ;

NET "led\_clk"       LOC = N26 | IOSTANDARD = LVCMOS33;

NET "led\_clrn"      LOC = N24 | IOSTANDARD = LVCMOS33;

NET "led\_sout"      LOC = M26 | IOSTANDARD = LVCMOS33;

NET "LED\_PEN"       LOC = P18 | IOSTANDARD = LVCMOS33;

NET "AN[0]" LOC = AC21 | IOSTANDARD = LVCMOS33;

NET "AN[1]" LOC = AD21 | IOSTANDARD = LVCMOS33;

NET "AN[2]" LOC = AB21 | IOSTANDARD = LVCMOS33;

NET "AN[3]" LOC = AC22 | IOSTANDARD = LVCMOS33;

NET "LED[0]" LOC = W22 | IOSTANDARD = LVCMOS33;

NET "LED[1]" LOC = AB26 | IOSTANDARD = LVCMOS33;

NET "LED[2]" LOC = Y25 | IOSTANDARD = LVCMOS33;

NET "LED[3]" LOC = AA23 | IOSTANDARD = LVCMOS33;

NET "LED[4]" LOC = Y23 | IOSTANDARD = LVCMOS33;

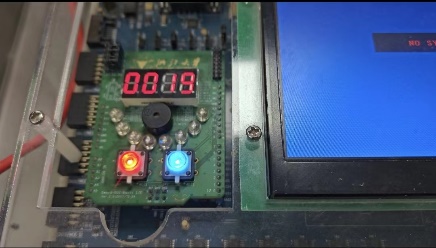
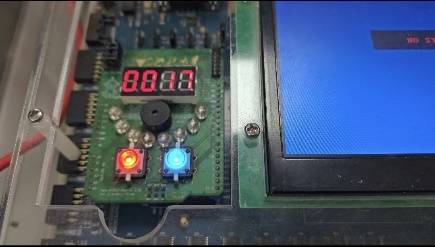
NET "LED[5]" LOC = Y22 | IOSTANDARD = LVCMOS33;

NET "LED[6]" LOC = AE21 | IOSTANDARD = LVCMOS33;

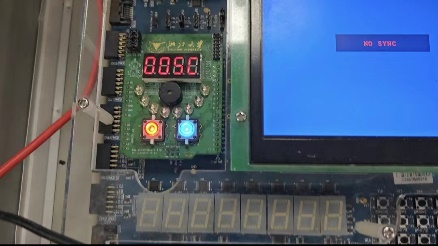
NET "LED[7]" LOC = AF24 | IOSTANDARD = LVCMOS33;

# 四、实验结果与分析

验证自增计数器：



验证自减计数器：



# 五、讨论、心得

本次实验我的主要收获在于深入学习了时序电路的设计与搭建过程，对它有了更加深入的理解。此外，在和同学的讨论过程中我也意识到了verilog本身作为一门硬件语言的重要性，我也在课后自己进行了相关语法的深入学习，希望以后再接再厉。

**六、个人生活照片展示**

****