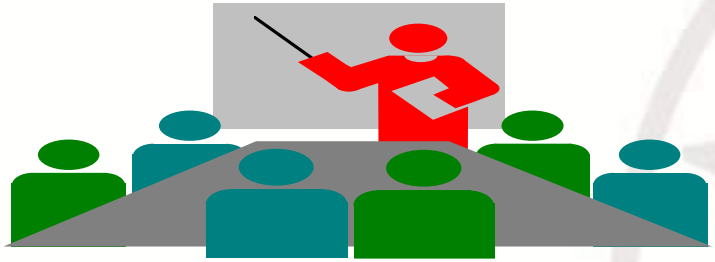


逻辑与计算机设计基础实验与课程设计

实验九

锁存器与触发器

施青松
Asso. Prof. Shi Qingsong
College of Computer Science and Technology, Zhejiang University
zjsqs@zju.edu.cn

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

实验目的

1. 掌握锁存器与触发器构成的条件和工作原理。
2. 掌握锁存器与触发器的区别。
3. 了解静态存储器SRAM存储单元结构。
4. 掌握基本RS锁存器的基本功能与使用要点。
5. 掌握RS、D触发器的基本功能及基本应用。
6. 掌握集成触发器的使用和异步清零的作用
7. 了解用D触发器实现分频电路；
8. 了解用D触发器构成单稳态电路(开关去抖动)。

实验环境

实验设备

1. 计算机（Intel Core i3以上，1GB内存以上）系统
2. SWORD Board开发板
3. Xilinx ISE12.4及以上开发工具

材料

无

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

实验任务

1. 用原理图实现RS锁存器并仿真验证；
2. 实现门控RS锁存器、D锁存器并仿真验证；
3. 用RS锁存器实现RS主从触发器并仿真验证；
4. 用D锁存器和用RS锁存器实现主从D触发器并仿真验证；
5. 用原理图实现维持阻塞型D触发器；
6. 触发器物理测试

Course Outline

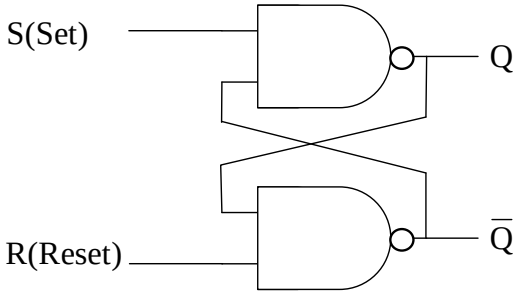
- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

什么是锁存器

- 锁存器三个基本条件
  - 能长期保持给定的某个稳定状态；
  - 有两个稳定状态，“0”、“1”；
  - 在一定条件下能随时改变状态
    - 即：set 置“1”或 reset 置“0”。
- 最基本的锁存器有R-S锁存器和D锁存器
  - 用反向门互锁：实现长期保持稳定状态
    - 使能(enable)NAND或NOR可实现
  - 破坏互锁实现set 或 reset
    - 禁止(disable) NAND或NOR可实现
  - 只有二个状态

最基本RS锁存器

NAND结构R-$\bar{S}$锁存器

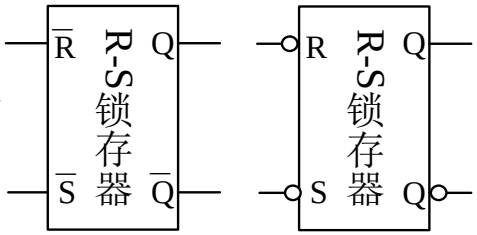


(a) 逻辑图

R S	Q $\bar{Q}$	S R
0 0	1 1	无定义
0 1	0 1	置 0
1 0	1 0	置 1
1 1	Q $\bar{Q}$	保持

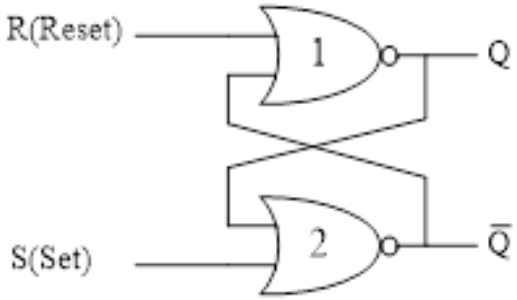
不确定

(b) 功能表



(c) 逻辑符号

NOR结构R-S锁存器

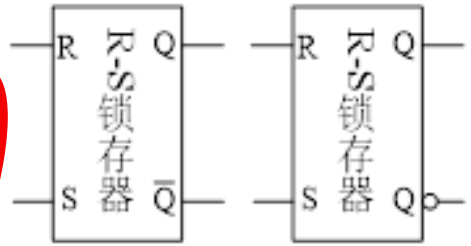


(a) 逻辑图

R S	Q $\bar{Q}$	S R
0 0	Q $\bar{Q}$	保持
0 1	1 0	置 1
1 0	0 1	置 0
1 1	0 0	无定义

不确定

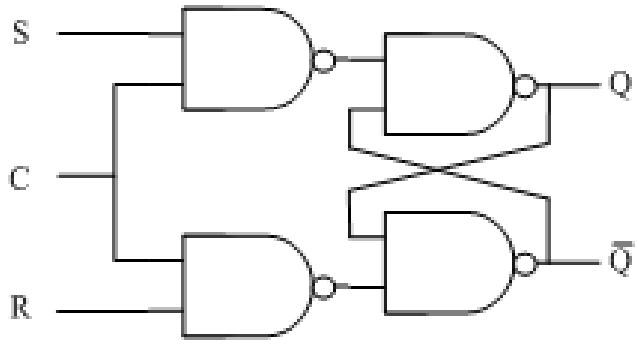
(b) 功能表



(c) 逻辑符号

使能控制

门控RS锁存器(电平使能)

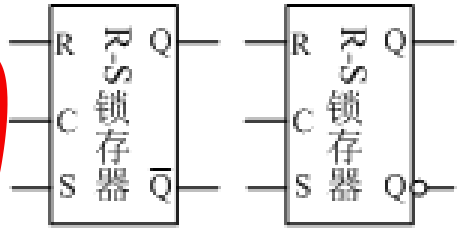


(a) 逻辑图

C R S	Q $\overline{Q}$	说明
0 × ×	Q $\overline{Q}$	保持
1 0 0	Q $\overline{Q}$	保持
1 0 1	1 0	置 1
1 1 0	0 1	置 0
1 1 1	1 1	无定义

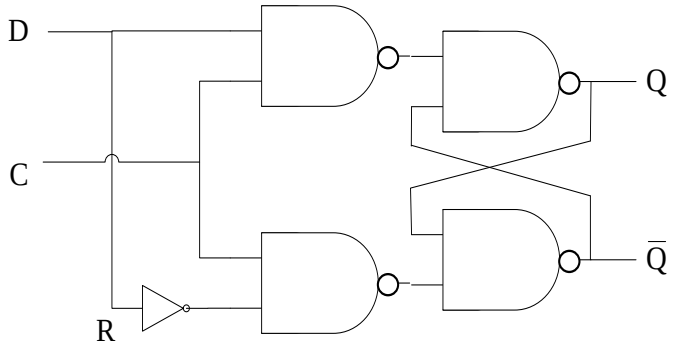
不确定

(b) 功能表



(c) 逻辑符号

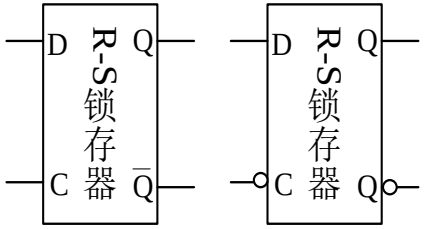
电平使能D锁存器



(a) 逻辑图

C D	Q $\overline{Q}$	说明
0 ×	Q $\overline{Q}$	保持
1 0	0 1	置 0
1 1	1 0	置 1

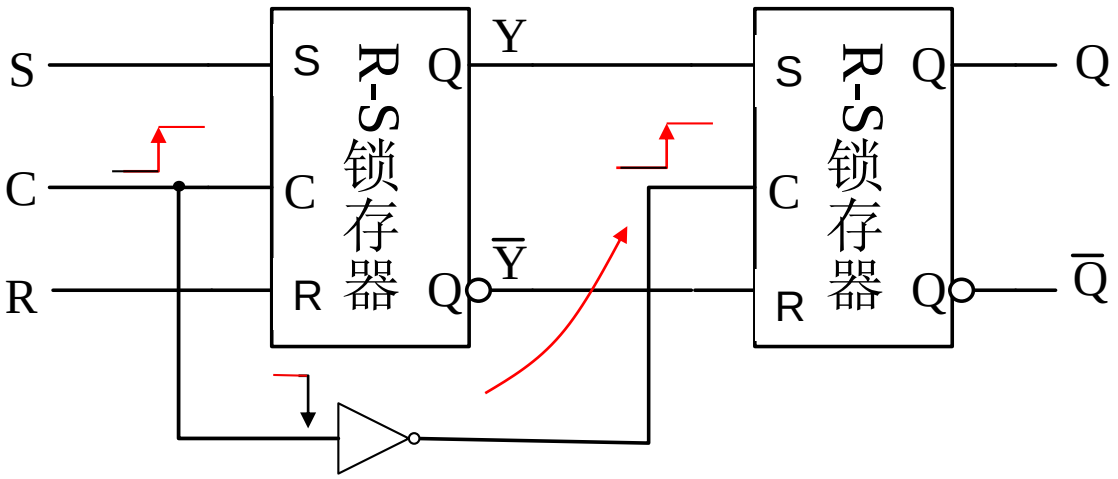
(b) 功能表



(c) 逻辑符号

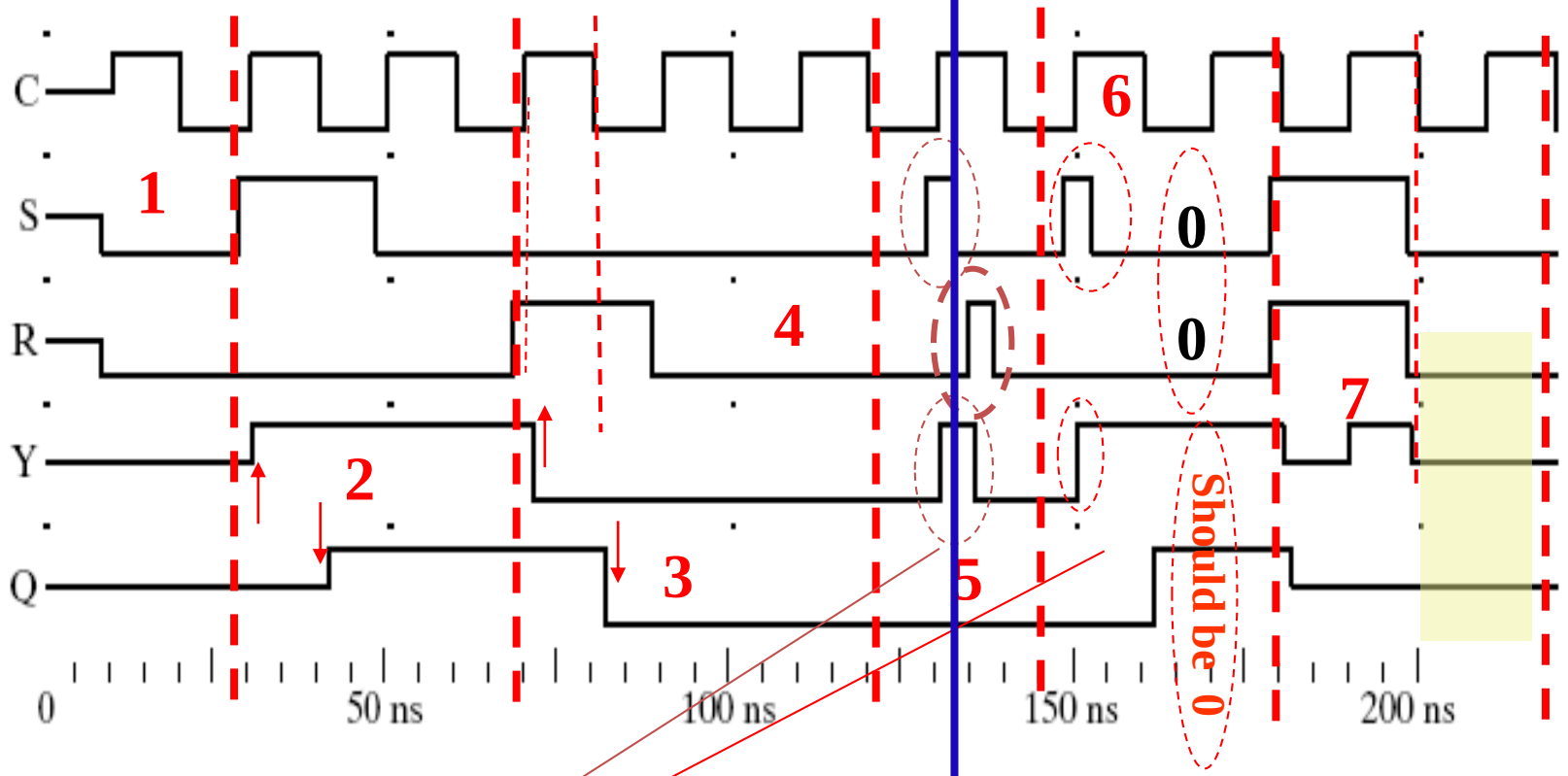
主从RS触发器

- 锁存器存在的问题
  - 锁存器的输出端可以直接反应出输入端的数据
  - 在时序电路存储状态时出现“空翻”无法控制
  - 解决思想：切断回路，同步变化一次→触发
- 主从触发器
  - 用使能信号控制回路，切断直接通路
    - 高电平采样
      - 正脉冲窗口
    - 低电平阻断采样
      - 保持



RS主从触发器存在的问题

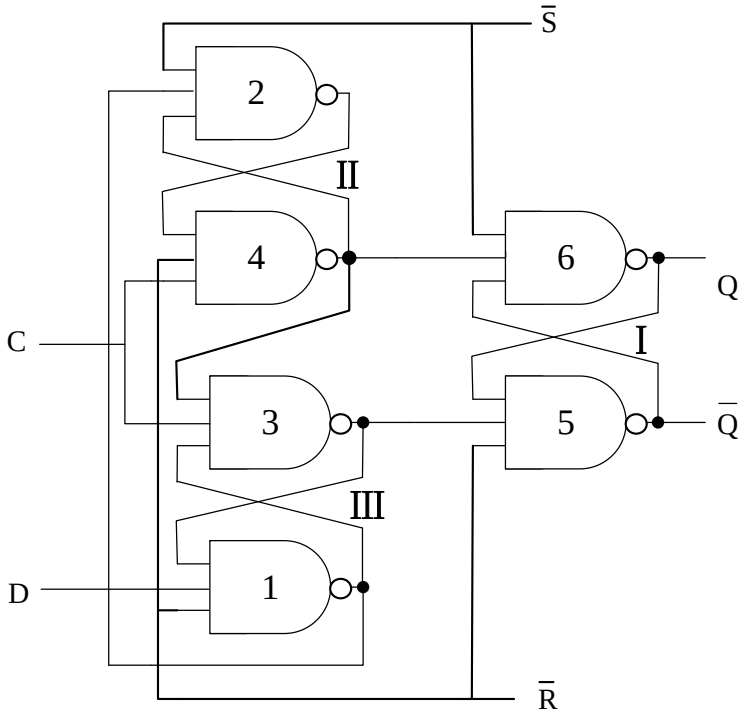
- 一次性采样（1st-Catching）

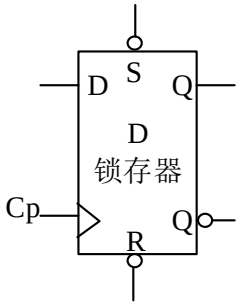


边沿触发器

维持阻塞型边沿触发器(正边沿)

- 缩小采用窗口：边沿采样





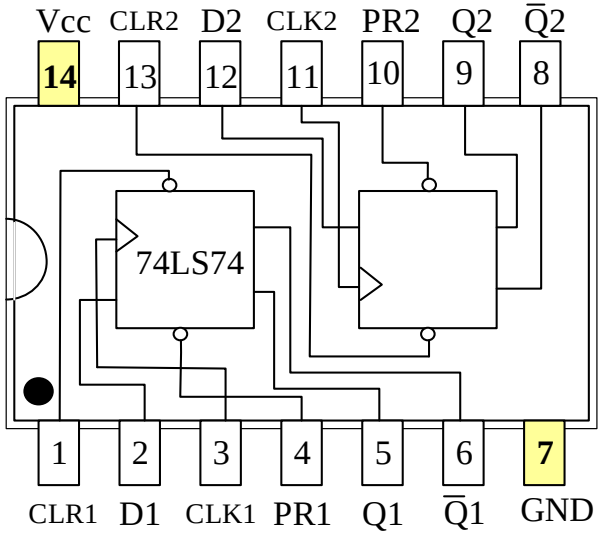
异步控制		上升沿触发			
$\overline{R}$	$\overline{S}$	Cp	D	Q	$\overline{Q}$
0	1	X	X	0	1
1	0	X	X	1	0
1	1	↑	0	0	1
1	1	↑	1	1	0

- 正边沿维持阻塞型D触发器原语描述

```
NAND3   and1(.I0(Rb), .I1(c), .I2(Q), .O(Qbar) ),
        and2(.I0(Qbar), .I1(d), .I2(Sb), .O(Q) );
NAND3   and3(.I0(Rb), .I1(c), .I2(Di), .O(a) ),
        and4(.I0(clk), .I1(a), .I2(d), .O(c) );
NAND3   and5(.I0(a), .I1(d), .I2(Sb), .O(b) ),
        and6(.I0(clk), .I1(b), .I2(Rb), .O(d) );
```

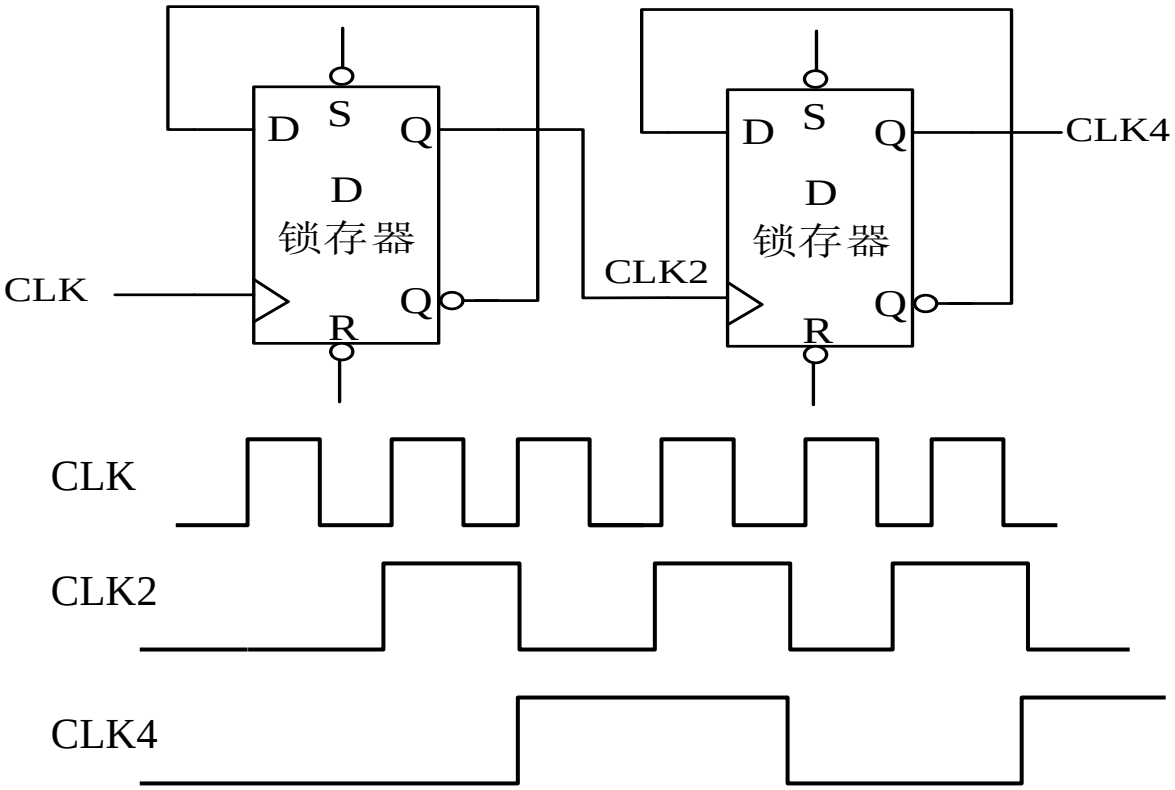
- 74LS74双D触发器

Inputs				Outputs	
PR	CLR	CLK	D	Q	$\overline{Q}$
L	H	X	X	H	L
H	L	X	X	L	H
L	L	X	X	H*	H*
H	H	↑	H	H	L
H	H	↑	L	L	H
H	H	L	X	Q_0	$\overline{Q}_0$



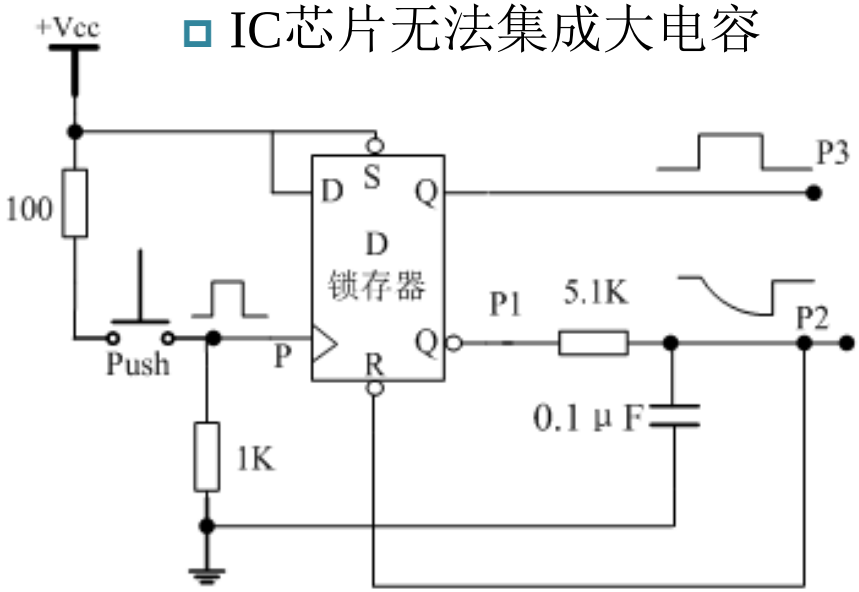
D触发器应用：分频电路

- 多个触发器一维调用(串联)
  - 时钟作为边界条件：前面输出Q接入后面时钟输入
    - 异步工作

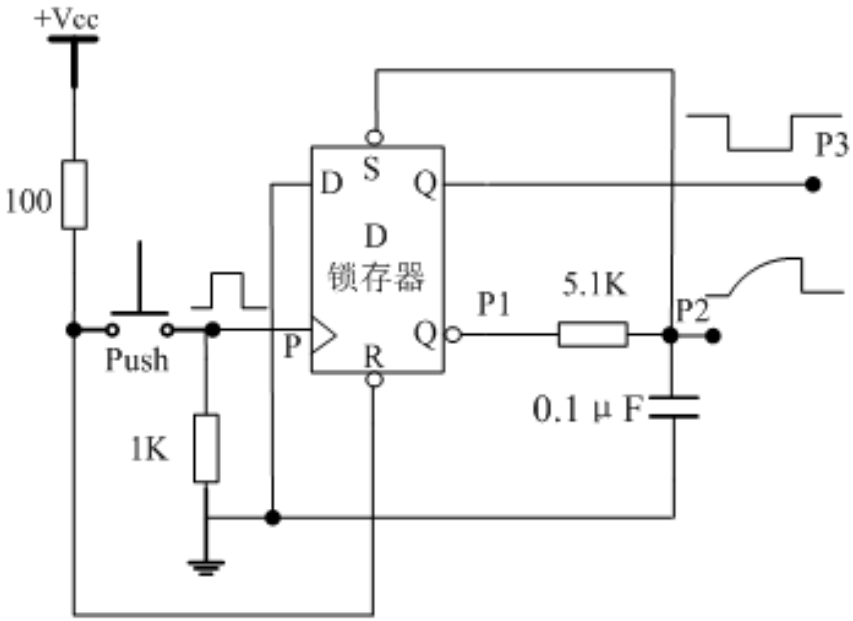


D触发器应用：单稳态电路

- 开头去抖动
  - 需要电阻、电容构成充放电延时
    - 称RC延时(中学物理知识)：时常数τ = RC
  - FPGA只能用计数延时
    - IC芯片无法集成大电容



(a)低稳态



(b)高稳态

Course Outline

- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

设计工程一：Locker

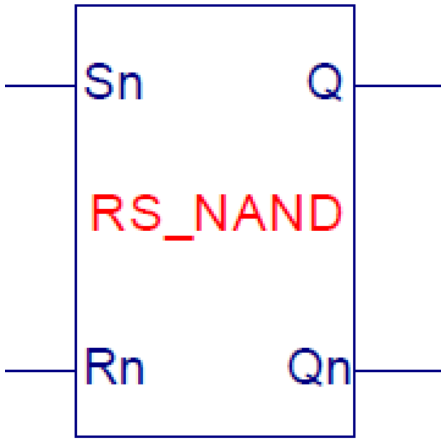
- **设计RS锁存器**
  - 用与非门实现并仿真验证
  - 增加使能控制并仿真验证
- **设计电平控制D锁存器**
  - 调用RS锁存器实现并仿真验证
- **此工程不下板做物理调试和验证**

设计要点

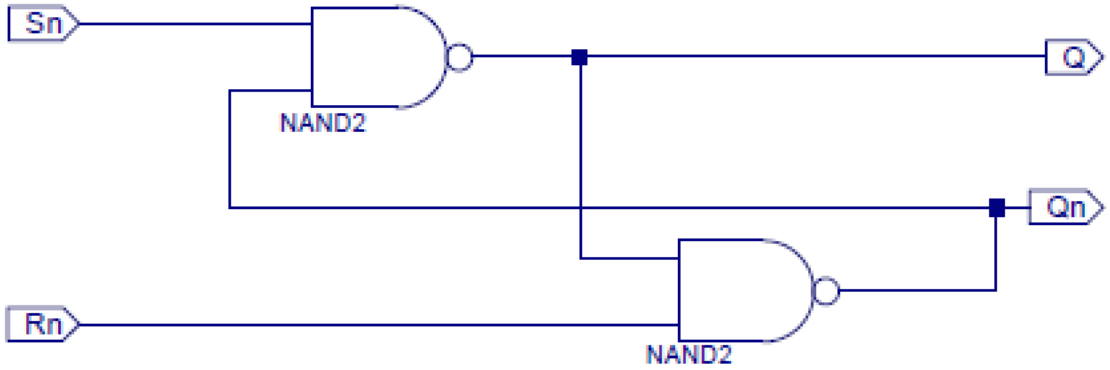
◎新建工程：Locker

◎设计RS锁存器

- 调用与非门(NAND2)用原理图设计实现
- 命名：RS_NAND
- 设计激励代码并仿真测试
- 仿真测试通过后封装逻辑符号



封装后逻辑符号



NAND RS锁存器仿真激励

◎ 激励设计考虑：

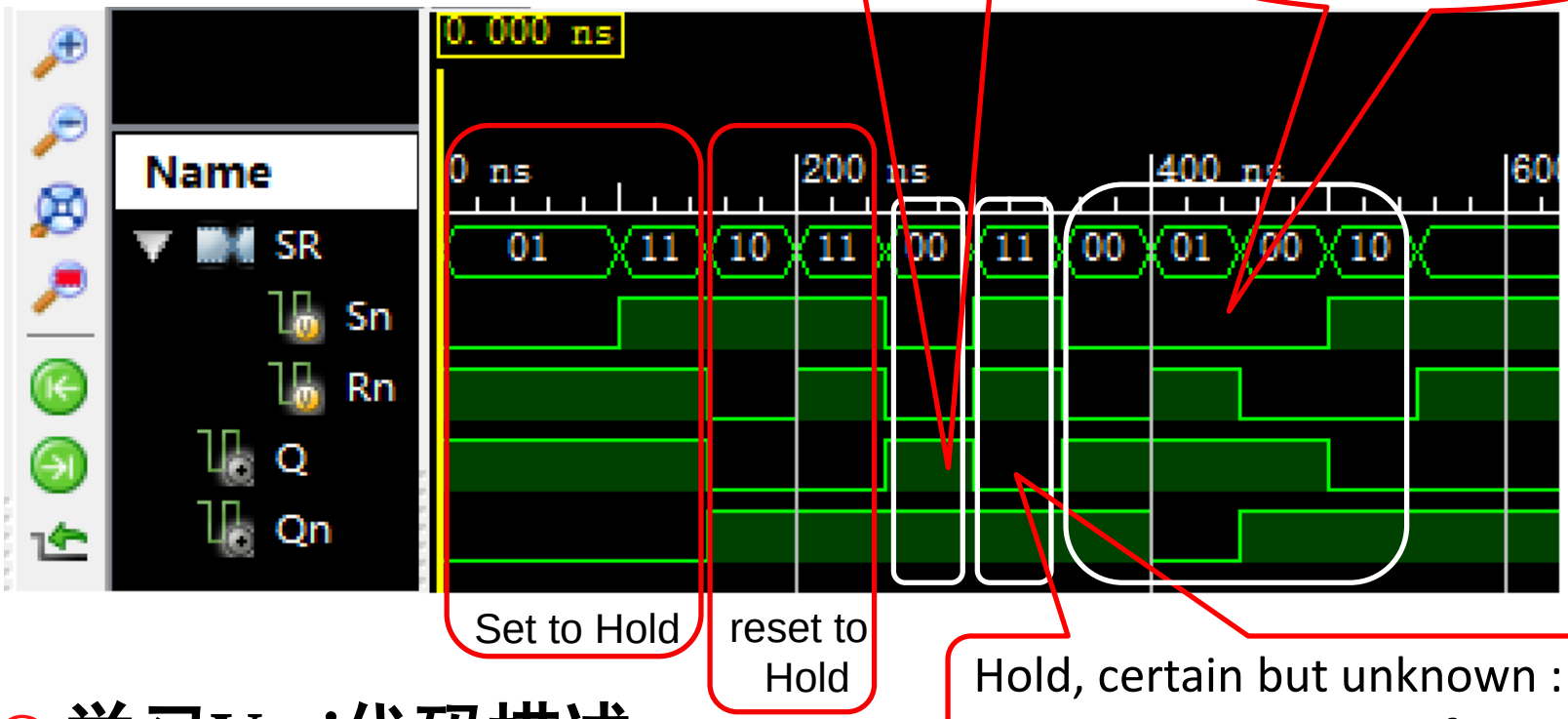
- 功能测试输入：set、reset、Hold
- 特殊状态输入：undefild、unknown

◎ 参考激励

```
25    initial begin
26       Rn = 1;
27       Sn = 0;
28       #50;              //From set to Hold
29       Sn = 0;           //set
30       Rn = 1;
31       #50;
32       Sn = 1;           //Hold
33       Rn = 1;
34       #50;              //From reset to Hold
35       Sn = 1;
36       Rn = 0;           //reset
37       #50;
38       Sn = 1;           //Hold
39       Rn = 1;
40       #50;              //From RS=00 to Hold
41       Sn = 0;           //undefild
42       Rn = 0;
43       #50;
44       Sn = 1;           //Hold: What Value?
45       Rn = 1;
46       #50;              //From RS=00 to set
47       Sn = 0;
48       Rn = 0;
49       #50;
50       Sn = 0;           //set
51       Rn = 1;
52       #50;              //From RS=00 to reset
53       Sn = 0;
54       Rn = 0;
55       #50;
56       Sn = 1;           //reset
57       Rn = 0;
58       #50;
59       Sn = 1;           //Hold
60       Rn = 1;
61    end
```

NAND RS锁存器仿真结果分析

◎ 仿真结果分析：

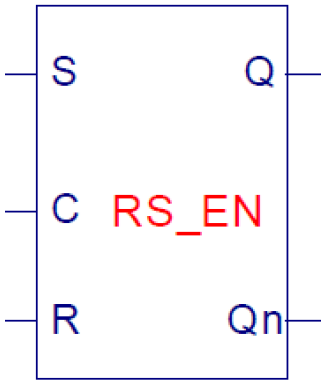


◎ 学习Veri代码描述

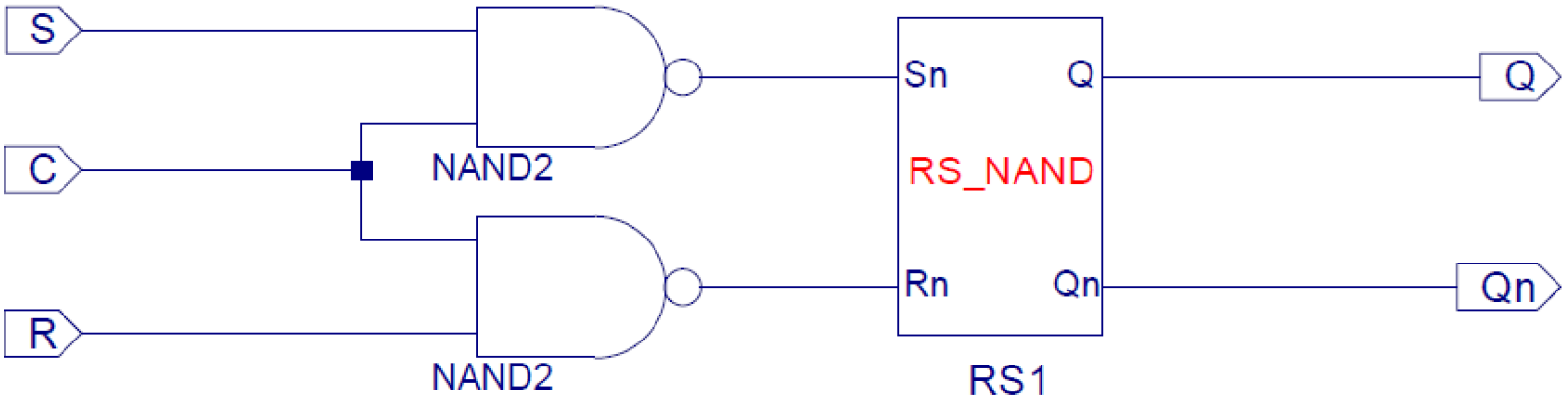
- 打开*View HDL Functional Model*分析学习模块的代码描述

◎ RS锁存器增加电平使能控制

- 命名为：RS_EN.sch
- 调用RS_NAND实现
- 仿真验证
- 验证后封装



封装后逻辑符号



带使能控制C的RS锁存器仿真激励

◎ 激励设计考虑：

- 使能控制采用方波信号
- 功能测试输入：set、reset、Hold
- 特殊状态输入：undefild、unknown

◎ 参考激励

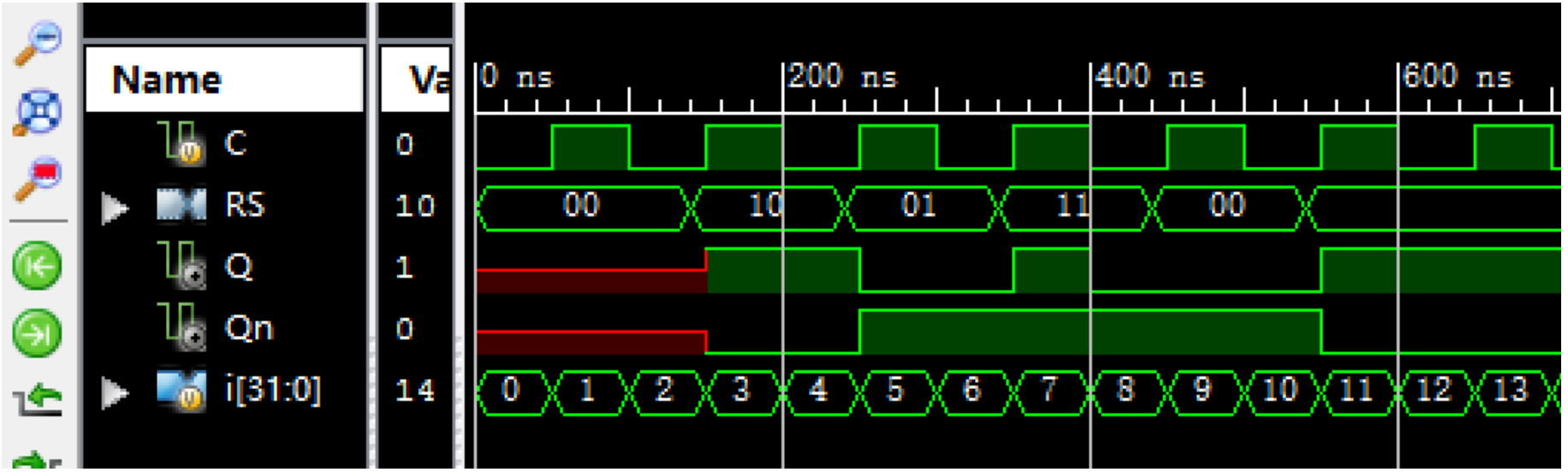
```
integer i=0;
// Initialize Inputs
        initial begin
        C = 0;
        S = 0;
        R = 0;
        #40;
        S = 0;        //Hold
        R = 0;
        S = 1;        //set
        R = 0;
        #100;
        S = 0;        //reset
        R = 1;
        #100;
        S = 1;        //undefild
        R = 1;
        #100;
        S = 0;        //Hold: What Value?
        R = 0;
        #100;
        S = 1;        //set
        R = 0;
        end

always@ * //Pulse, Square wave
        for(i=0;  i<20; i=i+1)begin
        #50;
        C <= ~C;
end
```

时钟

带使能控制C的RS锁存器仿真：

◎ 参考仿真结果

- 请分析你的仿真结果：

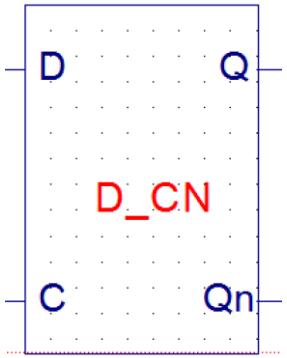


◎ 学习Veri代码描述

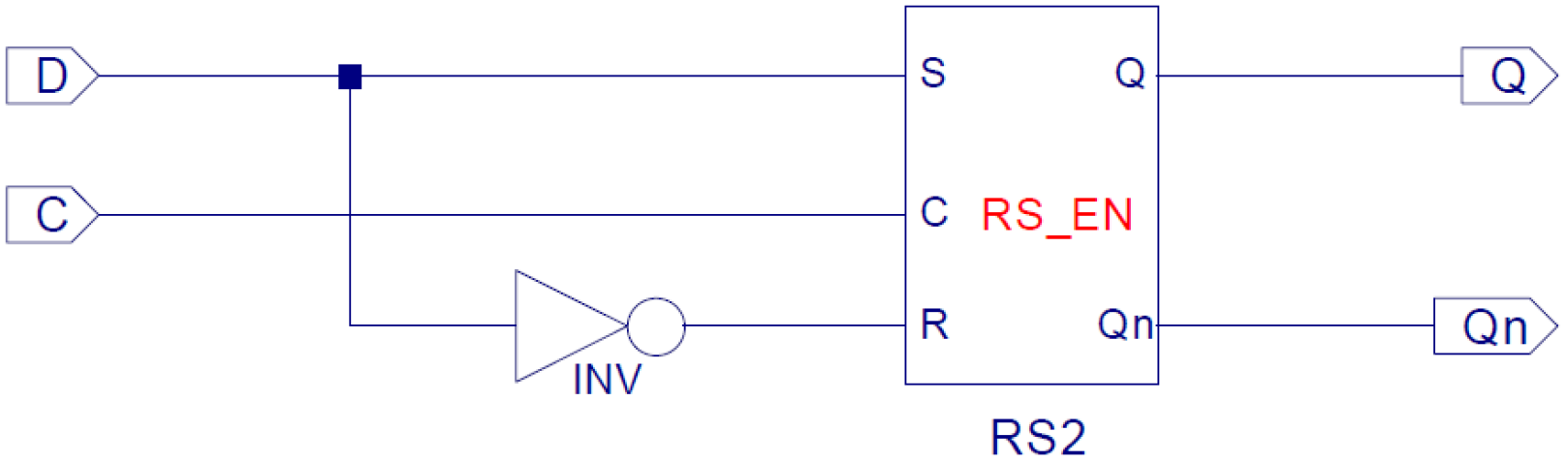
- 打开*View HDL Functional Model*分析学习模块的代码描述

◎ 设计电平控制D锁存器

- 命名为：D_EN.sch
- 调用RS_EN实现
- 仿真验证
- 验证后封装



封装后逻辑符号

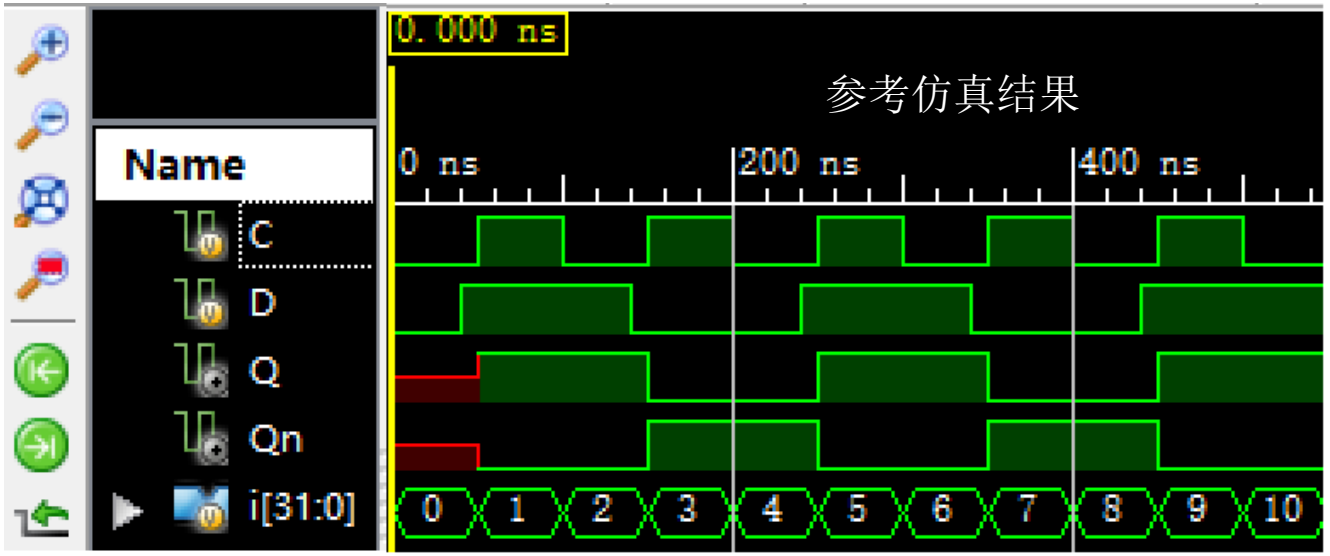


电平控制D锁存器仿真：

◎ 激励设计考虑：

- 使能控制采用方波信号
- 功能测试输入：D、Hold
- 这个太简单了，从现在开始自己设计激励测试代码

◎ 请分析你的仿真结果



◎ 学习Veri代码描述

- 打开*View HDL Functional Model*分析学习模块的代码描述

设计工程二：Trig

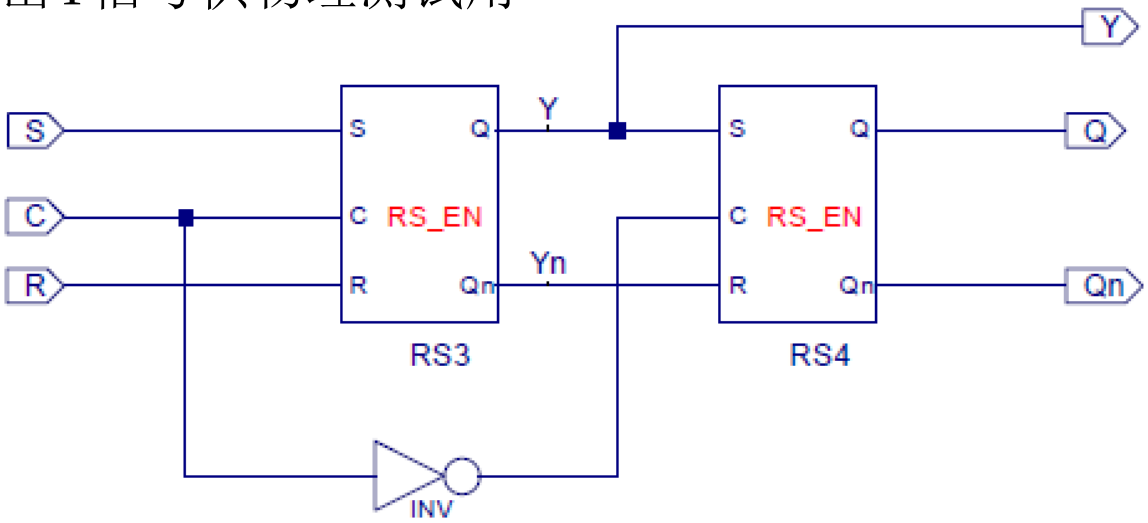
- **设计RS主从触发器**
  - 调用工程一RS_EN锁存器实现
  - 仿真并物理验证
- **设计D主从触发器(接近边沿功能)**
  - 调用工程一D_EN和RS_EN锁存器实现
  - 仿真并物理验证
- **设计维持阻塞型正边沿D触发器(习题5-3)**
  - 仿真并物理验证

设计要点

◎新建工程： Trigger

◎设计RS主从触发器

- 调用RS_EN用原理图设计实现
- 命名：RS_Trig
- 设计测试激励代码并仿真测试
- 增加主锁存器输出Y信号供物理测试用

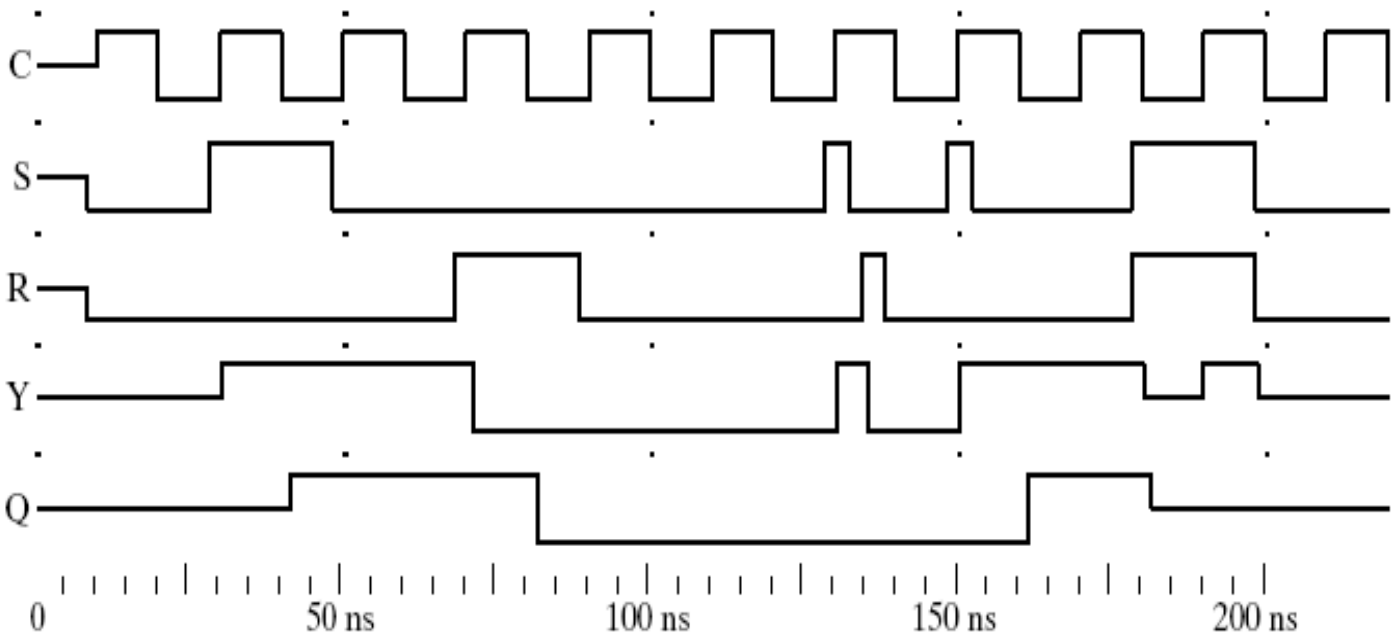


RS主从触发器仿真激励

◎ **激励设计考虑：**

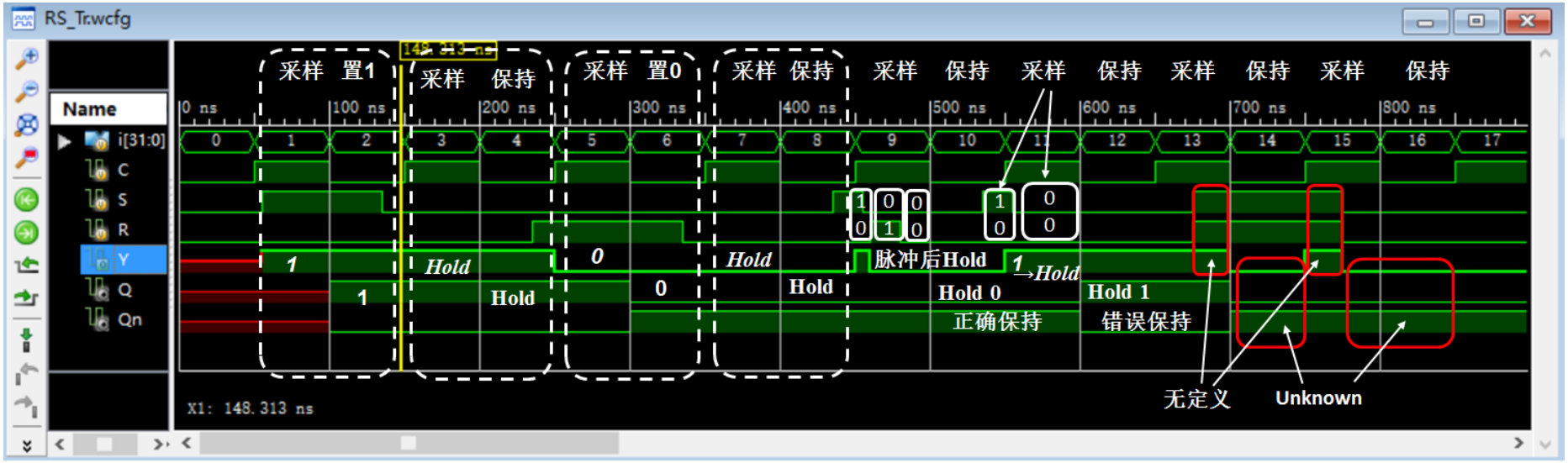
- 功能测试输入： set、reset、Hold
- 特殊状态输入：1s catching(一次性采用)

◎ **参考激励：（原版教材P217 Fig 5-10)**



RS主从触发器仿真结果

◎ 仿真结果分析：



◎ 仿真通过后下载做物理测试

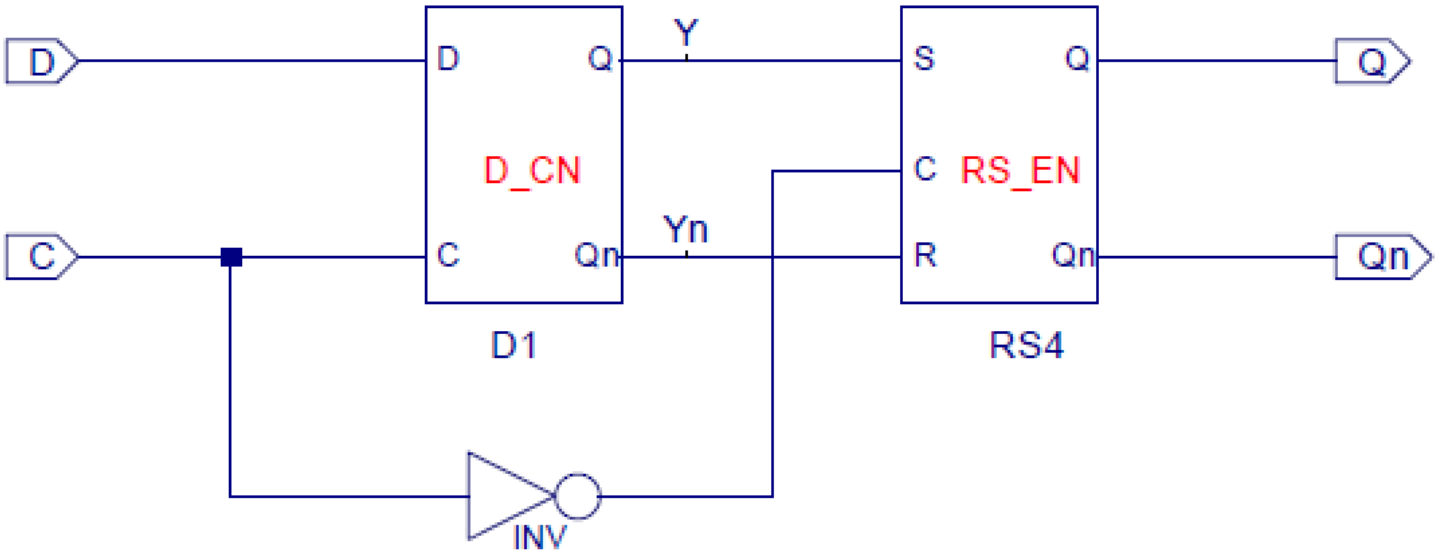
- ◎ RS_Trig：输入RS=SW[1:0]；输出{Y，Q，Qn}=LED[2:0]

◎ 学习Veri代码描述

- 打开*View HDL Functional Model*分析学习模块的代码描述

◎设计主从D触发器

- 调用D_EN和RS_EN用原理图设计实现
- 命名：D_Trig
- 设计测试激励代码并仿真测试

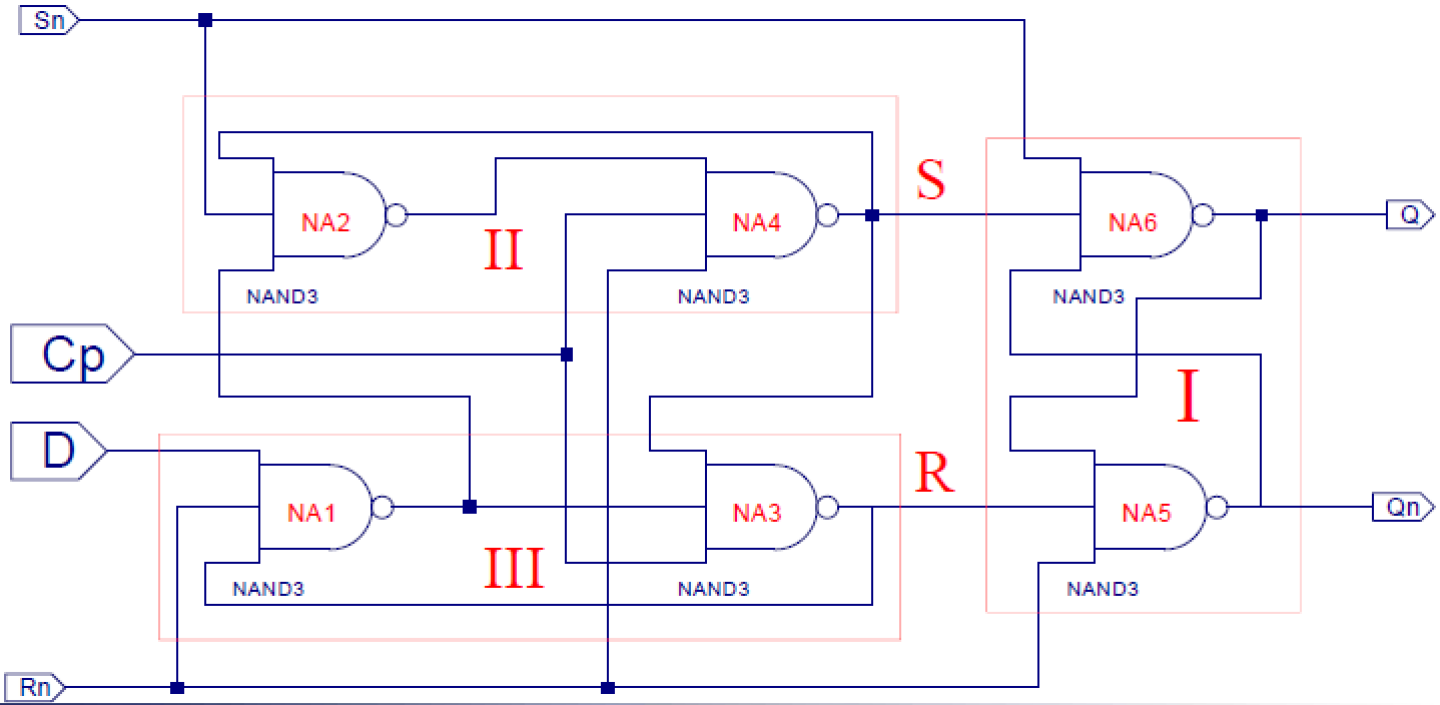


主从D触发器仿真激励

- **激励设计考虑：**
  - 功能测试输入：D=0、D=1
- **自行设计激励代码做仿真**
  - 分析你的仿真结果
- **仿真通过后下载做物理测试**
  - D_Trig：输入D=SW[3]；输出{Q，Qn}=LED[4:3]
- **学习Veri代码描述**
  - 打开*View HDL Functional Model*分析学习模块的代码描述

◎设计维持阻塞正边沿D触发器(习题5-3)

- 增加异步初始输入功能：Rn、Sn
- 调用与非门(NAND3)用原理图设计实现
- 命名：MB_DFF
  - Maintain Block DFF
- 设计测试激励代码并仿真测试



维持阻塞正边沿D触发器仿真激励

- **激励设计考虑：**
  - 功能测试输入(RnSn=11)：D=0、D=1
  - 异步初始化设置：RnSn=10、RnSn=01
- **自行设计激励代码做仿真**
  - 分析你的仿真结果
- **仿真通过后下载做物理测试**
  - MB_DFF：
    输入{Sn,Rn,D}=SW[6:4]；输出{Q，Qn}=LED[7:6]
- **学习Veri代码描述**
  - 打开*View HDL Functional Model*分析学习模块的代码描述

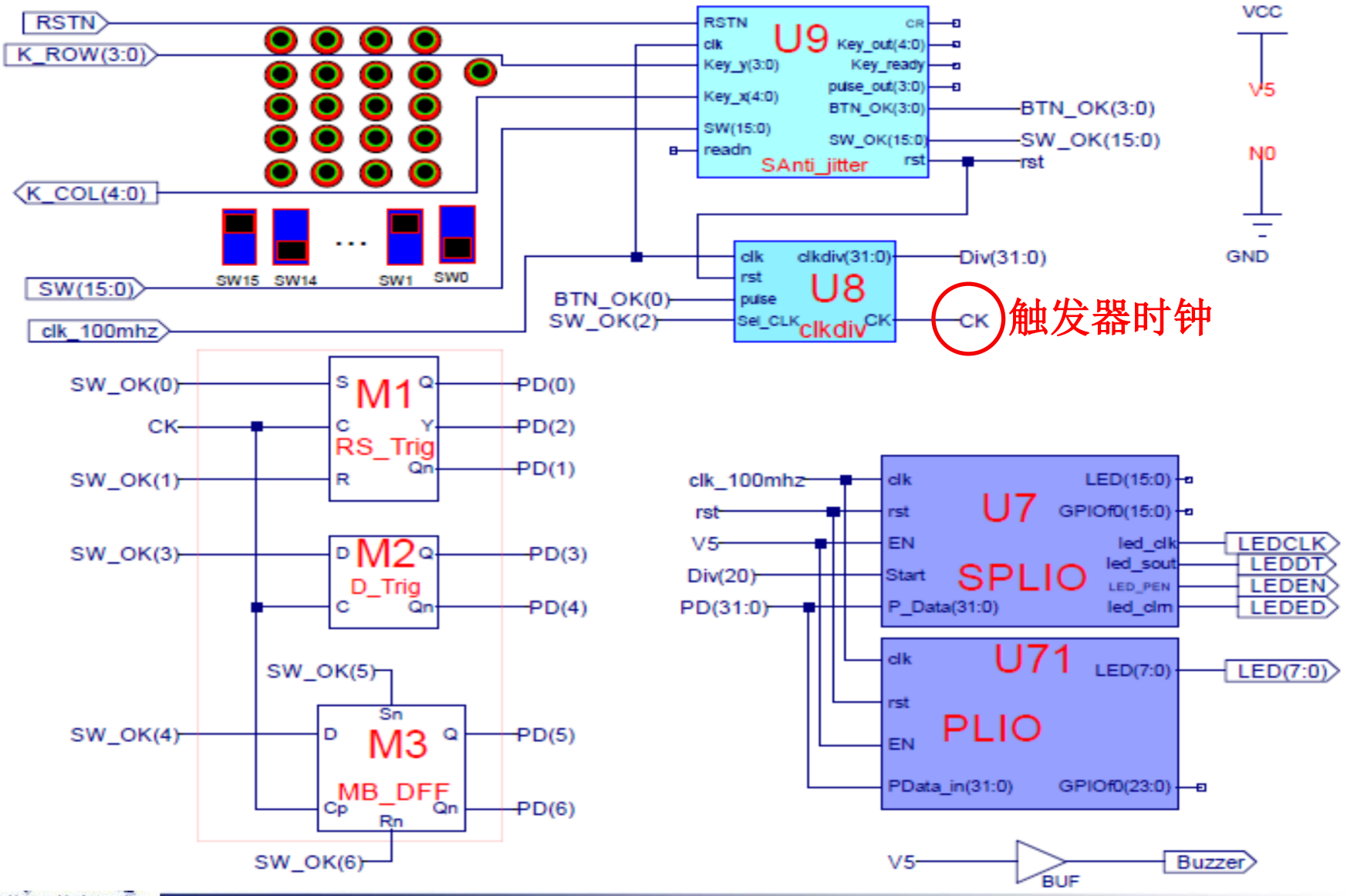
物理验证：顶层模块设计

◎ 触发器物理验证顶层模块

- 三个触发器用一个顶层模块：Top_Trigger
- 用Verilog HDL语言结构模块调用描述
  - ⊙ 也可以直接集成到实验八的顶层模块
- 开头去抖动模块与实验八相同或采用IP核：SAnti_jitter
- 触发器使能控制采用时钟，
  - ⊙ 修改分频模块(divclk)增加单步或连续时钟输出：CK
  - ⊙ 选择信号采用SW_OK[2]

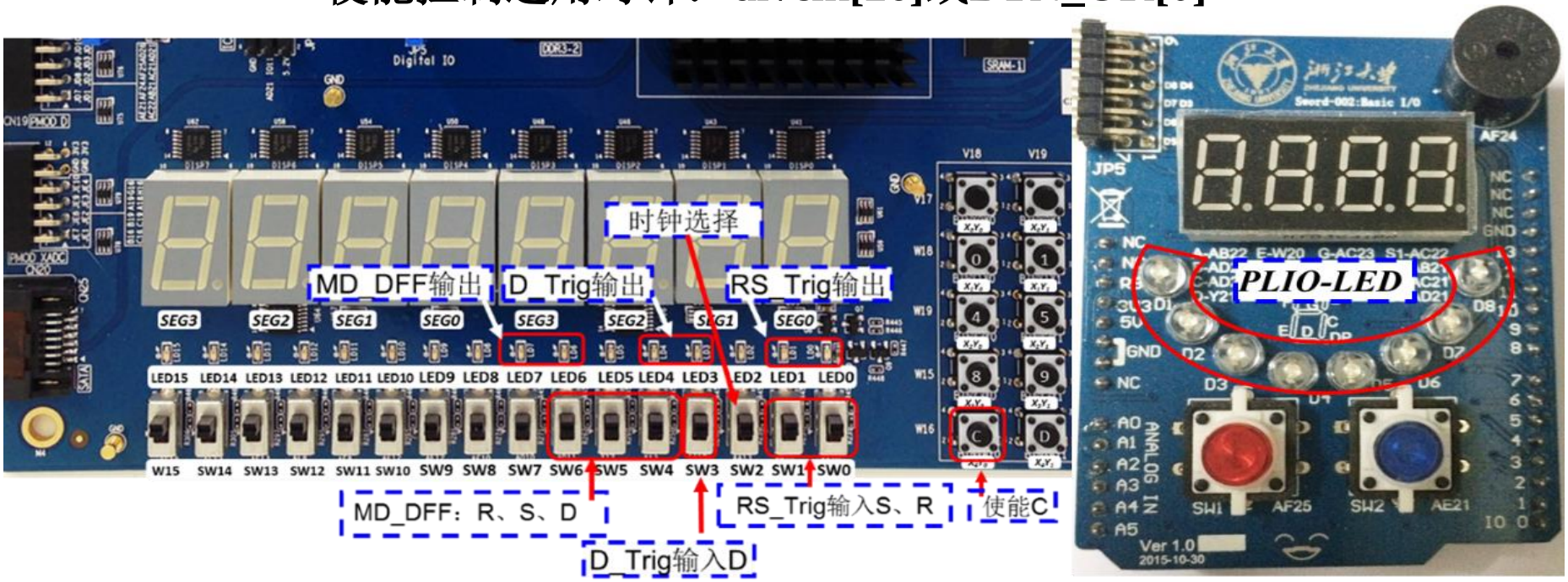
```
1  module  clkdiv(input clk,
2                 input rst,
3                 input Sel_CLK,
4                 input pulse,
5                 output CK,
6                 output reg[31:0]clkdiv
7                 );
8
9  // Clock divider-时钟分频器
10
11    always @ (posedge clk or posedge rst) begin
12       if (rst) clkdiv <= 0;
13       else clkdiv <= clkdiv + 1'b1;
14    end
15    assign CK = (Sel_CLK) ? pulse :  clkdiv[26];
16
17 endmodule
```

触发器验证顶层结构


RSTN
K_ROW(3:0)
K_COL(4:0)
SW(15:0)
SW15 SW14 SW1 SW0
clk_100mhz
U9
SAnti_jitter
RSTN
clk
Key_y(3:0)
Key_x(4:0)
SW(15:0)
readn
CR
Key_out(4:0)
Key_ready
pulse_out(3:0)
BTN_OK(3:0)
SW_OK(15:0)
rst
BTN_OK(3:0)
SW_OK(15:0)
rst
VCC
V5
N0
GND
U8
clkdiv
clk
rst
pulse
Sel_CLK
clkdiv(31:0)
CK
Div(31:0)
BTN_OK(0)
SW_OK(2)
CK
触发器时钟
M1
RS_Trig
S
C
R
Q
Y
Qn
SW_OK(0)
CK
SW_OK(1)
PD(0)
PD(2)
PD(1)
M2
D_Trig
D
C
Q
Qn
SW_OK(3)
PD(3)
PD(4)
M3
MB_DFF
Sn
D
Cp
Rn
Q
Qn
SW_OK(5)
SW_OK(4)
SW_OK(6)
PD(5)
PD(6)
U7
SPLIO
clk
rst
EN
Start
P_Data(31:0)
LED(15:0)
GPIOf0(15:0)
led_clk
led_sout
LED_PEN
led_clm
clk_100mhz
rst
V5
Div(20)
PD(31:0)
LEDCLK
LEDDT
LEDEN
LEDED
U71
PLIO
clk
rst
EN
PData_in(31:0)
LED(7:0)
GPIOf0(23:0)
LED(7:0)
V5
BUF
Buzzer

物理验证

- **UCF引脚定义**
  - RS_Trig： **输入**RS=SW[1:0]； 输出{Y，Q，Qn}=LED[2:0]
  - D_Trig： 输入D=SW[3]； 输出{Q，Qn}=LED[4:3]
  - MB_DFF：输入{Sn,Rn,D}=SW[6:4]；输出{Q，Qn}=LED[7:6]
  - 公共控制接口信号
    - 时钟选择SW_OK[2]：单步=1，连续=0
    - **使能控制选用时钟：divclk[26]或BTN_OK[0]**



顶层模块HDL描述参考结构

```
1   module   Top_Trigger(input clk_100mhz,
2                         input wire RSTN,
3                         input wire [3:0]K_COL,
4                         output wire[4:0]K_ROW,
5                         input wire[15:0]SW,
6
7                         output wire LEDCLK,          //串行移位时钟
8                         output wire LEDDT,           //串行输出
9                         output wire LEDCLR,          //LED显示清零
10                        output wire LEDEN,           //LED显示刷新使能
11                        output [7:0] LED,
12                        output Buzzer                //蜂鸣器
13                        );
14
15  wire[31:0]Div,PD;
16  wire [15:0]SW_OK;
17  wire [3:0]BTN_OK, pulse_out;
18  wire rst, CK;
19
20     assign clk = clk_100mhz;                        //时钟信号
21     assign Buzzer = 1'b1;                           //关闭蜂鸣器
```

触发器输入输出

```
23    RS_Trig      M1(.S(???),                    //"???"处请读者根据约束分配填写
24                    .R(???),
25                    .C(CK),
26                    .Y(???),
27                    .Q(???),
28                    .Qn(???)
29                    );
30
31    D_Trig       M2(.D(???),
32                    .C(CK),
33                    .Q(???),
34                    .Qn(???)
35                  );
36
37    MB_DFF        M3(.D(???),
38                  .Cp(CK),
39                  .Sn(???),
40                  .Rn(???),
41                  .Q(???),
42                  .Qn(???)
43                  );
```

RS_Trig：　输入RS=SW[1:0]；　　输出{Y，Q，Qn}=LED[2:0]
D_Trig：　输入D=SW[3]；　　输出{Q，Qn}=LED[4:3]
MB_DFF：输入{Sn,Rn,D}=SW[6:4]；输出{Q，Qn}=LED[7:6]

```
45  //--------------以下是辅助模块    ，不使用接口可以用空括号
46     SAnti_jitter    U8(.clk(clk),                    //去抖动模块
47                        .RSTN(RSTN),
48                        .readn(),                    //不使用的信号用空括号
49                        .Key_y(K_COL),
50                        .Key_x(K_ROW),
51                        .SW(SW),
52                        .Key_out(),
53                        .Key_ready(),
54                        .pulse_out(),
55                        .BTN_OK(BTN_OK),
56                        .SW_OK(SW_OK),
57                        .CR(),
58                        .rst(rst)
59                        );
60
61     clkdiv       U9(.clk(clk),                    // 时钟分频模块
62                     .rst(rst),
63                     .Sel_CLK(SW_OK[2]),
64                     .pulse(BTN_OK[0]),            //也可用button_pulse，有什么区别
65                     .clkdiv(Div),
66                     .CK(CK)                       //触发器时钟
67                     );
89     SPLIO        U7(.clk(clk),                     //GPIO模块串行输出，LED静态显示
90                     .rst(rst),
91                     .Start(Div[20]),              //移位输出启动信号
92                     .EN(1'b1),                    //PIO输出&LED显示刷新使能
93                     .P_Data(PD),                  //GPIO输出数据，通过移位输出
94                     .LED(),                       //16位LED灯状态
95                     .led_clk(LEDCLK),             //串行移位时钟
96                     .led_sout(LEDDT),             //移位输出数据
97                     .led_clrn(LEDCLR),            //LED(74LS164)显示清零
98                     .LED_PEN(LEDEN),              //LED(74LS164)显示刷新使能
99                     .GPIOf0()                     //保留GPIO位
100                    );
101 endmodule
```

```
PLIO      U71(.clk(clk),                    //GPIO模块串行输出，LED动态显示
                .rst(rst),
                .EN(1'b1),                  //PIO输出&LED显示刷新使能
                .PData_in(PD),              //GPIO输出数据，直接输出
                .LED(LED),                  //接LED指示灯
                .GPIOf0()                   //保留GPIO位
                );
```

Arduino

选一个

触发器物理测试

- **结合仿真时的激励用SW和BTN输入**
  - 在LED上观测测试结果
  - 用真值表记录测试结果
    - 功能测试
    - 一次性采用测试
    - 初始化功能测试(RS主从触发器)
    - 观测边沿与主从输出变化时间(单步时钟)
      - 改变触发器输入，**button_out[0]**按下、释放
      - 观测二类触发器Q是如何变化的
  - 分析触发器物理测试结果
    - 针对真值表无法表达的用文字详细叙述

主从触发器测试

主从RS触发器(RS_Trig，C=0时改变RS)							
功能	R	S	C单步	Y	Q	Qn	备注
置1			0				
			1→0				
保持			0				
			1→0				
置0			0				
			1→0				
保持			0				
			1→0				
无定义	1	1	0				
			1→0				
保持	**0**	**0**	0				
			1→0				

一次性采样测试

主从RS触发器(RS_Trig，C=0时改变RS)								
功能		R	S	C单步	Y	Q	Qn	备注
一次性采样	置1	0	1	1→0				
	采样RS	0	0	1				
	R=脉冲	1	0	1				
	保持	0	0	1→0				
一次性采样	置1	0	1	1→0				
	采样RS	0	0	1				
	S=脉冲	0	1	1				
	保持	0	0	1→0				

D触发器测试

主从D触发器(D_Trig)				边沿D触发器(MD_DFF)					
D	C	Q	Qn	Rn	Sn	D	C	Q	Qn

思考题

- 触发器unknown能仿真吗？为什么？
- 触发器unknown能物理测试吗？为什么？
- 通过物理测试说明：
  - RS主从触发器一次性采样的原因和使用影响
  - 边沿触发器和主从D触发器改变状态过程有什么不同？
  - 根据RS主从触发器Y输出测试对比，说明主从和边沿在状态变化过程的区别对使用的影响(不是指一次性采样)

同学们：每次做完实验请整理好实验台，放好仪器，理清桌面。