

浙江大学

本科实验报告

| | |
|-------|--|
| 课程名称: | 计算机逻辑设计基础 |
| 姓 名: | 王晓宇 |
| 学 院: | 竺可桢学院 |
| 专 业: | 计算机 |
| 邮 箱: | 1657946908@qq.com |
| QQ 号: | 1657946908 |
| 电 话: | 19550222634 |
| 指导教师: | 洪奇军 |
| 报告日期: | 2023 年 12 月 15 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： 移位寄存器设计与应用 完成 P2S

学生姓名： 王晓宇 学号： 3220104364 同组学生姓名：

实验地点： 紫金港东四 509 室 实验日期： 2023 年 12 月 15 日

一、操作方法与实验步骤

任务 1：设计 8 位带并行输入的右移移位寄存器并设计跑马灯程序

►移位寄存器

每来一个时钟脉冲，寄存器中的数据按顺序向左或向右移动一位

✧必须采用主从触发器或边沿触发器

✧不能采用锁存器

数据移动方式：左移、右移、循环移位

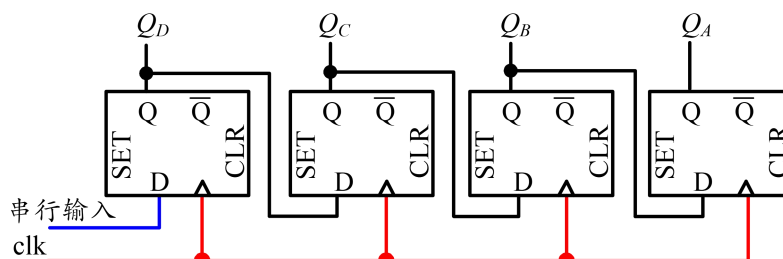
数据输入输出方式

✧串行输入，串行输出

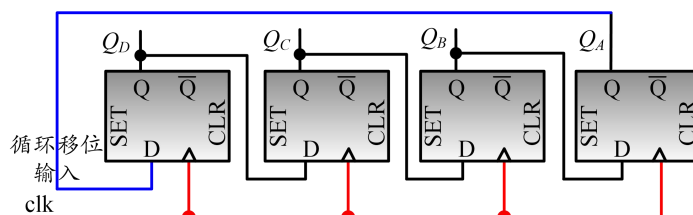
✧串行输入，并行输出

✧并行输入，串行输出

使用 D 触发器构成串行输入的右移移位寄存器

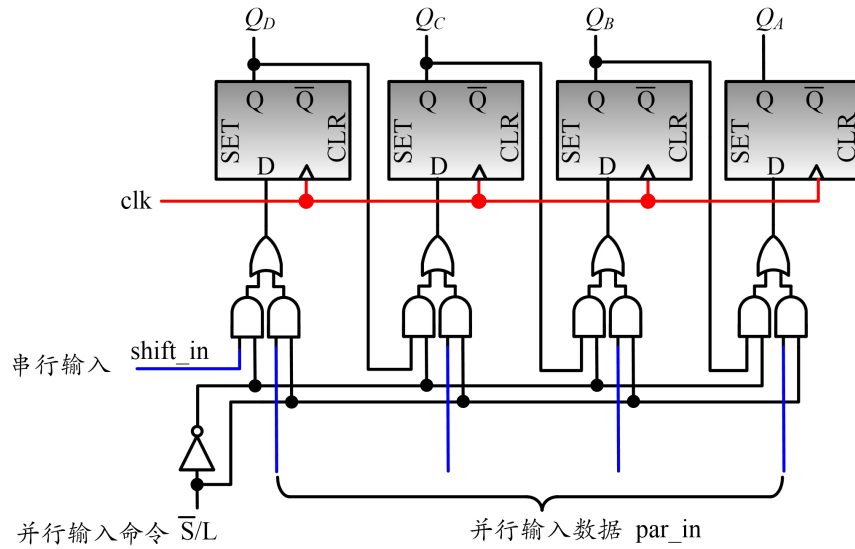


循环右移移位寄存器



►带并行输入的移位寄存器

数据输入方式：串行输入、并行输入



新建工程

工程名称用 ShfitReg8b。

Top Level Source Type 用 HDL

用结构化描述设计

```

19 //
20 ///////////////////////////////////////////////////
21 module MyshiftReg8b(
22     input wire clk,SL,sin,cycle,
23     input wire [7:0]pin,
24     output wire [7:0]Q
25 );
26 FD M1(.C(clk),.D((!SL & Q[1])|(SL & pin[0])),.Q(Q[0]));
27 FD M2(.C(clk),.D((!SL & Q[2])|(SL & pin[1])),.Q(Q[1]));
28 FD M3(.C(clk),.D((!SL & Q[3])|(SL & pin[2])),.Q(Q[2]));
29 FD M4(.C(clk),.D((!SL & Q[4])|(SL & pin[3])),.Q(Q[3]));
30 FD M5(.C(clk),.D((!SL & Q[5])|(SL & pin[4])),.Q(Q[4]));
31 FD M6(.C(clk),.D((!SL & Q[6])|(SL & pin[5])),.Q(Q[5]));
32 FD M7(.C(clk),.D((!SL & Q[7])|(SL & pin[6])),.Q(Q[6]));
33 FD M8(.C(clk),.D((!SL & (cycle&Q[0]|!cycle & sin))|(SL & pin[7])),.Q(Q[7]));
34
35 endmodule
36

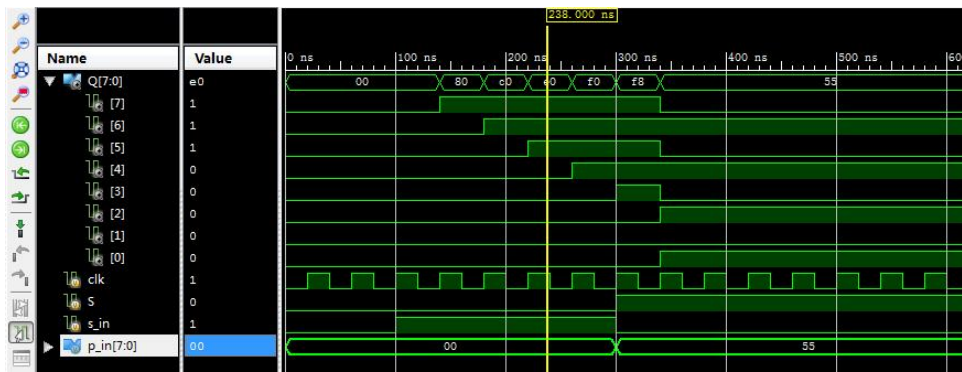
```

波形仿真

```

4 wire [7:0] Q;
5
6 // Instantiate the Unit Under Test (UUT)
7 MyshiftReg8b uut (
8     .clk(clk),
9     .SL(SL),
10    .sin(sin),
11    .pin(pin),
12    .Q(Q)
13 );
14
15 initial begin
16     // Initialize Inputs
17     clk = 0;
18     SL = 0;
19     sin = 0;
20     pin = 0;
21     // Wait 100 ns for global reset to finish
22     #100;
23     SL = 0;
24     sin = 1;
25     pin = 8'b00000000;
26     #200;
27     SL = 1;
28     sin = 0;
29     pin = 8'b01010101;
30     #500;
31 end
32
33 always begin
34     clk = 0;#20;
35     clk = 1;#20;
36 end
37
38 endmodule
39

```



设计跑马灯应用

新建工程

工程名称用 MyMarquee。

Top Level Source Type 用 HDL

New Project Wizard

Create New Project
Specify project location and type.

Enter a name, locations, and comment for the project—

Name: MyMarquee

Location: /tmp/Logic/MyMarquee

Working Directory: /tmp/Logic/MyMarquee

Description:

Select the type of top-level source for the project—

Top-level source type: HDL

More Info Next > Cancel

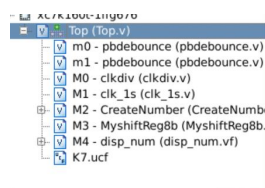
用结构化描述设计

调用 ShfitReg8b

调用分频模块，用 1s 作为移位寄存器驱动时钟

调用显示模块

调用 CreateNumber 模块



```

1 //////////////////////////////////////////////////
2 module Top(
3     input wire clk,
4     input wire [15:0]sw,
5     input wire [1:0]btn,
6     output wire [7:0]SEGMENT,
7     output wire [3:0] AN,
8     output wire [7:0]LED,
9     output wire BTNX4
10 );
11 assign BTNX4 = 0;
12 wire [15:0] num;
13 wire clk_ls;
14 wire [31:0]clk_div;
15 wire [1:0]btn_out;
16
17 pbdebounce m0(clk_div[17],btn[0],btn_out[0]);
18 pbdebounce m1(clk_div[17],btn[1],btn_out[1]);
19 clkdiv M0(.clk(clk),.clkdiv(clk_div),.rst(1'b0));
20 clk_ls M1(.clk(clk),.clk_ls(clk_ls));
21 CreateNumber M2(.btn({2'b0,btn_out}),.num(num),.sw({2'b00,sw[1:0]}));
22 MyshiftReg8b M3(.clk(clk_ls),.SL(sw[2]),.sin(sw[3]),.cycle(sw[4]),.pin(num[7:0]),.Q(LED));
23 disp_num M4(.clk(clk),.HEXS((num[7:0],LED)),.LES(4'b0000),.points(4'b0000),.RST(1'b0),.AN(AN),.segment(SEGMENT));
24
25 endmodule
26

```

```

NET "clk" LOC = AC18 | IOSTANDARD = LVCMOS18;
NET "clk" TNM_NET = TM_CLK;
TIMESPEC TS_CLK_100M = PERIOD "TM_CLK" 10ns HIGH 50%;
#NET "RSTN" LOC = W13 | IOSTANDARD = LVCMOS18;
NET "BTN[0]" LOC = V18 | IOSTANDARD = LVCMOS18;
NET "BTN[0]" CLOCK_DEDICATED_ROUTE = FALSE;
NET "BTN[1]" LOC = V19 | IOSTANDARD = LVCMOS18;
NET "BTN[1]" CLOCK_DEDICATED_ROUTE = FALSE;
//NET "BTN[2]" LOC = V14 | IOSTANDARD = LVCMOS18;
//NET "BTN[2]" CLOCK_DEDICATED_ROUTE = FALSE;
//NET "BTN[3]" LOC = W14 | IOSTANDARD = LVCMOS18;
//NET "BTN[3]" CLOCK_DEDICATED_ROUTE = FALSE;

NET "SW[0]" LOC = AA10 | IOSTANDARD = LVCMOS15;
NET "SW[1]" LOC = AB10 | IOSTANDARD = LVCMOS15;
NET "SW[2]" LOC = AA13 | IOSTANDARD = LVCMOS15;
NET "SW[3]" LOC = AA12 | IOSTANDARD = LVCMOS15;
NET "SW[4]" LOC = Y13 | IOSTANDARD = LVCMOS15;
NET "SW[5]" LOC = Y12 | IOSTANDARD = LVCMOS15;
NET "SW[6]" LOC = AD11 | IOSTANDARD = LVCMOS15;
NET "SW[7]" LOC = AD10 | IOSTANDARD = LVCMOS15;
NET "SW[8]" LOC = AE10 | IOSTANDARD = LVCMOS15;
NET "SW[9]" LOC = AE12 | IOSTANDARD = LVCMOS15;
NET "SW[10]" LOC = AF12 | IOSTANDARD = LVCMOS15;
NET "SW[11]" LOC = AE8 | IOSTANDARD = LVCMOS15;
NET "SW[12]" LOC = AF8 | IOSTANDARD = LVCMOS15;
NET "SW[13]" LOC = AE13 | IOSTANDARD = LVCMOS15;
NET "SW[14]" LOC = AF13 | IOSTANDARD = LVCMOS15;
NET "SW[15]" LOC = AF10 | IOSTANDARD = LVCMOS15;

```

```

36
37 NET "LED[0]" LOC = W23 | IOSTANDARD = LVCMOS33;
38 NET "LED[1]" LOC = AB26 | IOSTANDARD = LVCMOS33;
39 NET "LED[2]" LOC = Y25 | IOSTANDARD = LVCMOS33;
40 NET "LED[3]" LOC = AA23 | IOSTANDARD = LVCMOS33;
41 NET "LED[4]" LOC = Y23 | IOSTANDARD = LVCMOS33;
42 NET "LED[5]" LOC = Y22 | IOSTANDARD = LVCMOS33;
43 NET "LED[6]" LOC = AE21 | IOSTANDARD = LVCMOS33;
44 NET "LED[7]" LOC = AF24 | IOSTANDARD = LVCMOS33;
45 //NET "BTNX3" LOC = W15 | IOSTANDARD = LVCMOS18;
46 NET "BTNX4" LOC = W16 | IOSTANDARD = LVCMOS18;
47 NET "SEGMENT[0]" LOC = AB22 | IOSTANDARD = LVCMOS33;
48 NET "SEGMENT[1]" LOC = AD24 | IOSTANDARD = LVCMOS33;
49 NET "SEGMENT[2]" LOC = AD23 | IOSTANDARD = LVCMOS33;
50 NET "SEGMENT[3]" LOC = Y21 | IOSTANDARD = LVCMOS33;
51 NET "SEGMENT[4]" LOC = W20 | IOSTANDARD = LVCMOS33;
52 NET "SEGMENT[5]" LOC = AC24 | IOSTANDARD = LVCMOS33;
53 NET "SEGMENT[6]" LOC = AC23 | IOSTANDARD = LVCMOS33;
54 NET "SEGMENT[7]" LOC = AA22 | IOSTANDARD = LVCMOS33;
55 NET "AN[0]" LOC = AD21 | IOSTANDARD = LVCMOS33;
56 NET "AN[1]" LOC = AC21 | IOSTANDARD = LVCMOS33;
57 NET "AN[2]" LOC = AB21 | IOSTANDARD = LVCMOS33;
58 NET "AN[3]" LOC = AC22 | IOSTANDARD = LVCMOS33;
59

```

下载验证

用 sw[0] 和 sw[1] 作为 regA 和 regB 的按键自增控制输入

sw[2]=1, 并行输入, 将 {RegA, RegB} 赋给移位寄存器

sw[2]=0, 串行/循环右移移位

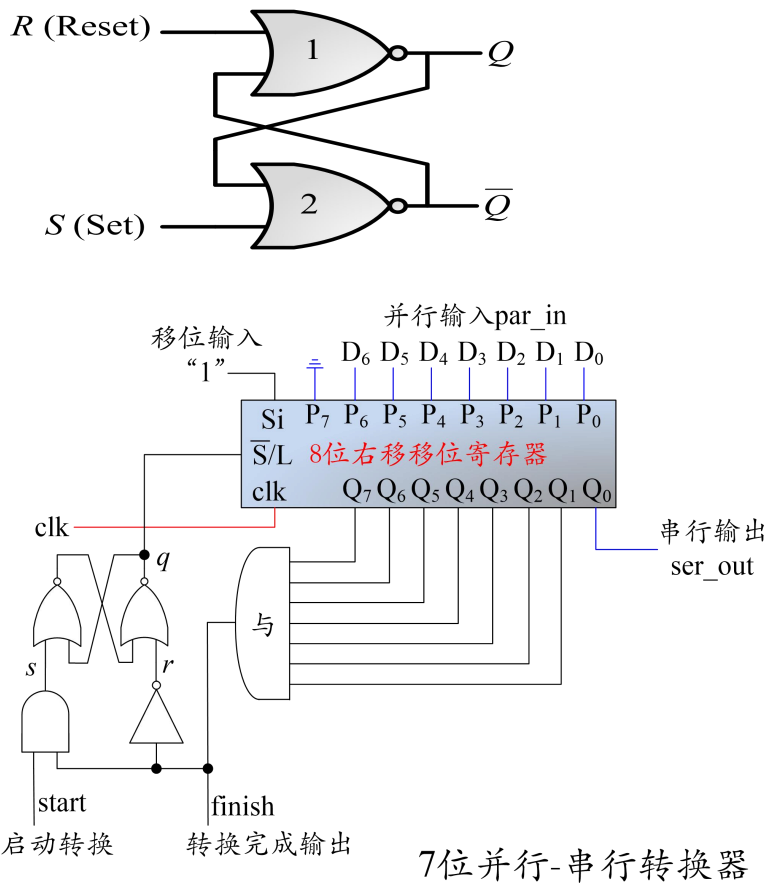
sw[4] 作为移位寄存器的模式选择:

sw[4]=0, 串行右移, 串行输入值为 sw[3]

sw[4]=1, 循环右移

8 位的移位寄存器的值用 LED 灯表示

任务 2: 并行—串行转换器



7位并行-串行转换器

并行-串行转换器设计原理

当没有启动命令（低电平）时，电路上电后经过若干个时钟脉冲后将会稳定在 RS 触发器输出 $q=0$ ，移位寄存器 $Q_7-Q_0=11111111$ 的状态。

当启动命令（高电平）加至启动输入端时，RS 触发器的输出端 q 被置 1，7 位并行数据及标志码“0”在第一个 clk 的作用下同时置入移位寄存器。此时，由于 $P_7=0$ 导致 $Q_7=0$ ，使得七输入与门的输出变成 0，

一方面封锁启动命令的输入，

另一方面通过非门在 RS 的触发器的 r 端输入 1，使 RS 触发器的输出 $q=0$ ，移位寄存器进入移位状态。

再在时钟脉冲作用下，一方面使并行数据串行移出，另一方面又不断将“1”移入寄存器。

等第 7 个脉冲来到后，七输入与门的输入已全为“1”，使得其输出变为 1，标志着转换完成，同时解除对启动信号的封锁

当再来一个启动命令时又可以再次进行并行-串行转换

接口说明：主板 LED 灯



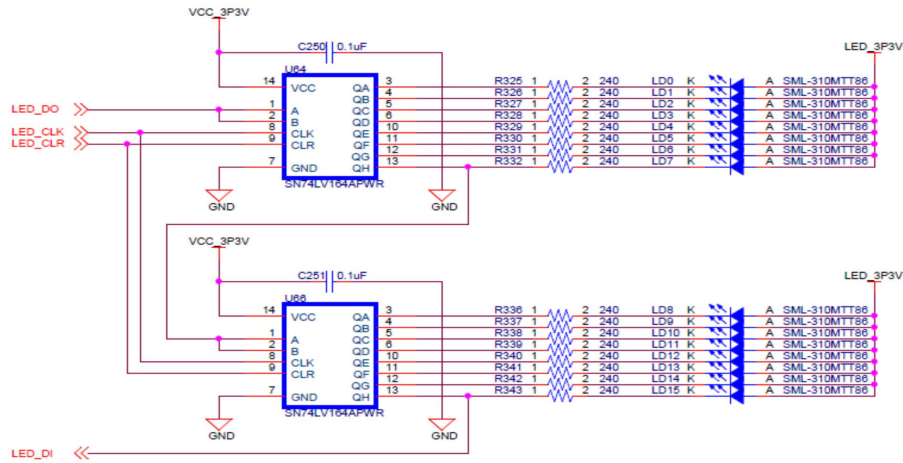
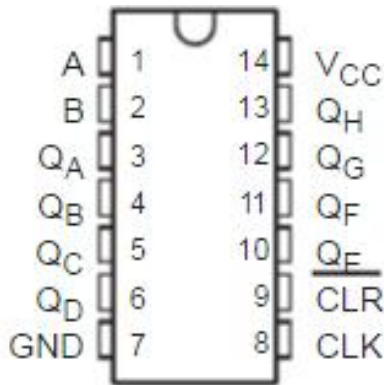
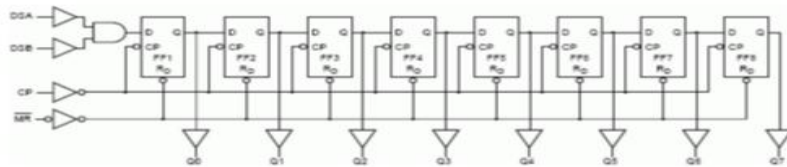


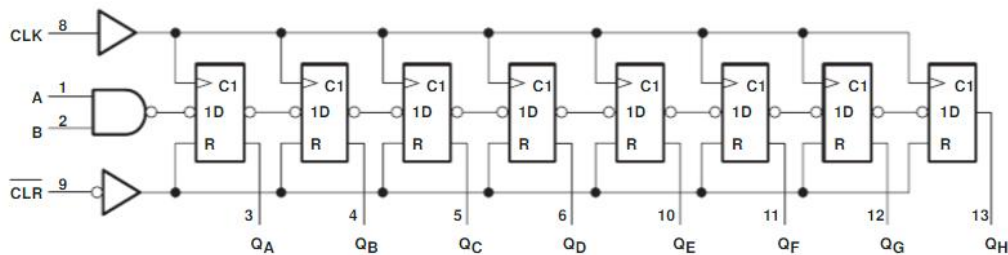
图 3. 逻辑图



| FUNCTION TABLE | | | | | | |
|----------------|-----|---|---|-----------------|-----------------------------------|-----------------|
| INPUTS | | | | OUTPUTS | | |
| CLR | CLK | A | B | Q _A | Q _B ... Q _H | |
| L | X | X | X | L | L | L |
| H | L | X | X | Q _{A0} | Q _{B0} | Q _{H0} |
| H | ↑ | H | H | H | Q _{An} | Q _{Gn} |
| H | ↑ | L | X | L | Q _{An} | Q _{Gn} |
| H | ↑ | X | L | L | Q _{An} | Q _{Gn} |

QA0, QB0, QH0 = the level of QA, QB, or QH, respectively, before the indicated steady-state input conditions were established.

QA_n, QG_n = the level of QA or QG before the most recent ↑ transition of the clock; indicates a 1-bit shift.



```

20 //////////////////////////////////////////////////
21 module My74LV164A(
22     input wire clk,A,B,CLR,
23     output wire [7:0]Q
24 );
25 FDR M1(.C(clk),.D(A & B),.Q(Q[0]),.R(!CLR));
26 FDR M2(.C(clk),.D(Q[0]),.Q(Q[1]),.R(!CLR));
27 FDR M3(.C(clk),.D(Q[1]),.Q(Q[2]),.R(!CLR));
28 FDR M4(.C(clk),.D(Q[2]),.Q(Q[3]),.R(!CLR));
29 FDR M5(.C(clk),.D(Q[3]),.Q(Q[4]),.R(!CLR));
30 FDR M6(.C(clk),.D(Q[4]),.Q(Q[5]),.R(!CLR));
31 FDR M7(.C(clk),.D(Q[5]),.Q(Q[6]),.R(!CLR));
32 FDR M8(.C(clk),.D(Q[6]),.Q(Q[7]),.R(!CLR));
33
34 endmodule
35

```

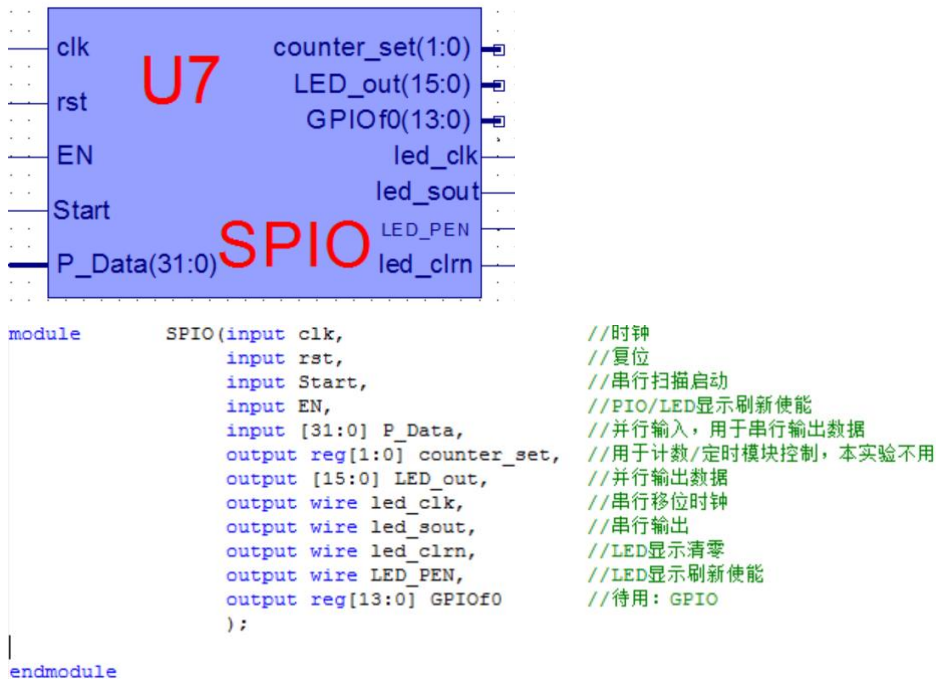

LED 并行显示模块 M6: SPI0

15 位 LED 指示灯控制 (IP Core)，逻辑实验的输出 LED 显示模块，相当于通用输入输出接口：GPIO15 位用于 LED 指示控制，其余用于扩展

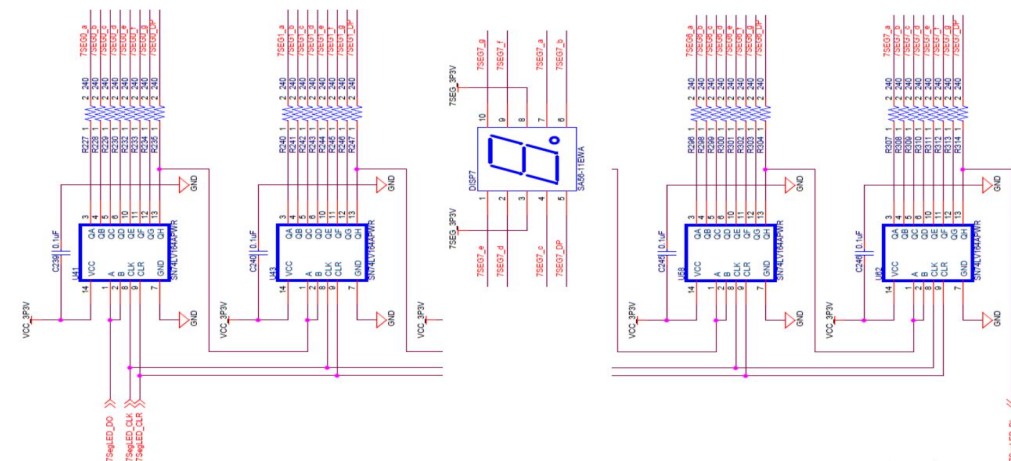
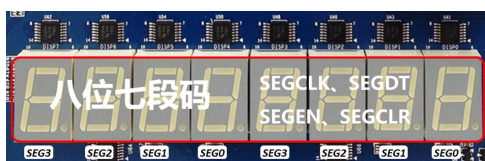
器件编号改为 U7

本课程用于调试显示和 CPU 的简单外设

基本功能：输入 32 位二进制数据：P_Data，clk=时钟，EN：输出使能，Start：串行扫描启动，rst=复位，串行输出：led_clk=时钟，led_sout=串行输出数据，LED_PEN=使能，led_clrn=清零，并行输出：LED_out、counter_set、GPIOf0，核模块符号文档：SPI0.sym
本实验提供 U7 的 IP 核



接口说明：主板七段数码管



#七段码移位输出引脚约束

```
NET "SEGCLK" LOC = M24 | IOSTANDARD = LVCMOS33 ;
```

```
NET "SEGCLR" LOC = M20 | IOSTANDARD = LVCMOS33 ;
```

```
NET "SEGDT" LOC = L24 | IOSTANDARD = LVCMOS33 ;
```

```
NET "SEGEN" LOC = R18 | IOSTANDARD = LVCMOS33 ;
```

SEGCLK: 74LV164A 的时钟

SEGCLR: 清零

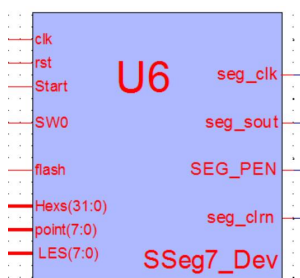
SEGDT: 数据串行输入

SEGEN: 控制数码管电源, 1 为使能

七段码显示器 IP 核 M3: SSeg7_Dev

8 位七段码显示器 (IP Core), 逻辑实验的输出显示模块, 本课程用于调试显示和 CPU 的简单外设, 器件编号改为 U6。





基本功能: 输入 32 位二进制数据: Hexs, SW[0]=1, 显示 8 位 16 进制数, SW[0]=0, 显示七段码 LED 点阵, SW[0]=1 时: SW[1]=1 高 16 位, SW[1]=0 低 16 位, , flash 七码闪烁频率, 由通用分频器 U8 (Div[25]) 提供, Start 串行扫描启动, point: 七段小数点, LES: 七段码使能, 闪烁指示, 串行输出: seg_clk=时钟, seg_out=串行七段显示数据, SEG_PEN=使能, seg_clrn=清零, 核模块符号文档: SSeg7_Dev.sym, 由实验二优化扩展, 本实验提供 U6 的 IP 核











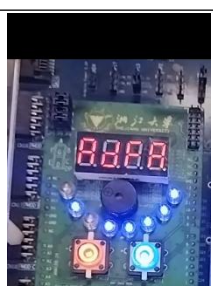



```
module SSeg7_Dev(input clk,           // 时钟
                 input rst,           // 复位
                 input Start,         // 串行扫描启动
                 input SW0,           // 文本(16进制)/图型(点阵)切换
                 input flash,         // 七段码闪烁频率
                 input[31:0]Hexs,     // 32位待显示输入数据
                 input[7:0]point,     // 七段码小数点: 8个
                 input[7:0]LES,       // 七段码使能: =1时闪烁
                 output seg_clk,       // 串行移位时钟
                 output seg_sout,     // 七段显示数据(串行输出)
                 output SEG_PEN,      // 七段码显示刷新使能
                 output seg_clrn      // 七段码显示清零
                );

endmodule
```

二、实验结果与分析 (完成 P2S)

| 并行输入 | | | |
|---|---|--|---|
| AA | AB | AC | AD |
|  |  |  |  |

| 循环跑马灯 | | | |
|--|--|---|--|
| 1 | 2 | 3 | 4 |
|  |  |  |  |

| 串行输入跑马灯 | | | |
|---|---|--|---|
| 1 | 2 | 3 | 4 |
|  |  |  |  |
| 5 | 6 | 7 | 8 |
|  |  |  |  |

三、讨论、心得

本次的实验同样地先看一下本次实验的部分函数。8 位右移寄存器的制作，利用门级电路描述，可以很方便的免去原理图中繁琐的连线，可以规避连错线的错误；另一方面，利用门级电路来写函数，可以很方便地写出与或非三种逻辑门，不必重新应用逻辑门。

这是最后一次实验了，不知不觉已经经过了十三周的学习，从一开始的完全不知所以然到现在的熟练打开 ISE 软件平台，很难说这段难忘的学习时光。

感谢《计算机逻辑设计基础》课程中博学的洪老师与王老师，以及帮助过我的助教和各位热心的同学，如春风吹动我知识的荒芜。

于途各兼程，千里自同风。