浙江大学

本科实验报告

课程名称：	数字逻辑设计
姓　　名：	
学　　院：	计算机科学与技术学院
专　　业：	计算机科学与技术
学　　号：	
指导教师：	马德

2022 年　　1 月　　9 日

浙江大学实验报告

课程名称：数字逻辑设计　　　　　实验项目名称：移位寄存器设计与应用

学生姓名：　　　　　专业：计算机科学与技术　　　　学号：

指导老师：马德　　　　　　　　　　实验日期：2021 年 1 月 8 日

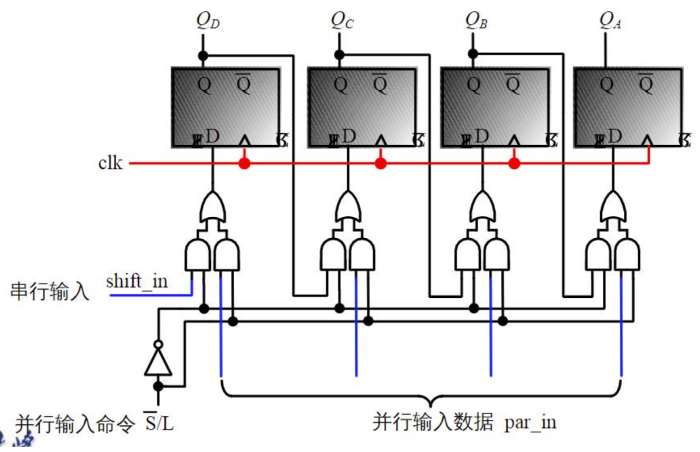
一、实验目的

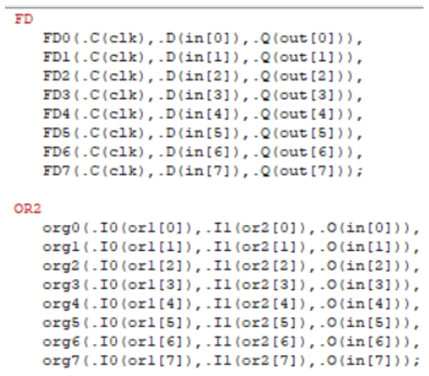
1. 掌握支持并行输入的移位寄存器的工作原理
2. 掌握支持并行输入的移位寄存器的设计方法

二、操作方法与实验步骤

1. 设计 8 位带并行输入的右移移位寄存器

① 建立 ShiftReg8b 工程，按照结构图，使用结构化描述设计 8 位右移移位寄存器。



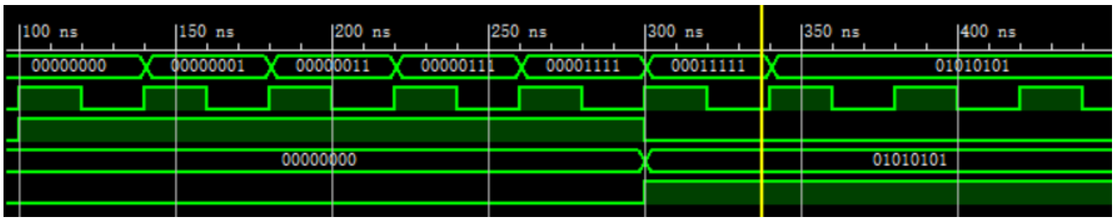


```
FD
    FD0(.C(clk),.D(in[0]),.Q(out[0])),
    FD1(.C(clk),.D(in[1]),.Q(out[1])),
    FD2(.C(clk),.D(in[2]),.Q(out[2])),
    FD3(.C(clk),.D(in[3]),.Q(out[3])),
    FD4(.C(clk),.D(in[4]),.Q(out[4])),
    FD5(.C(clk),.D(in[5]),.Q(out[5])),
    FD6(.C(clk),.D(in[6]),.Q(out[6])),
    FD7(.C(clk),.D(in[7]),.Q(out[7]));

OR2
    org0(.I0(or1[0]),.I1(or2[0]),.O(in[0])),
    org1(.I0(or1[1]),.I1(or2[1]),.O(in[1])),
    org2(.I0(or1[2]),.I1(or2[2]),.O(in[2])),
    org3(.I0(or1[3]),.I1(or2[3]),.O(in[3])),
    org4(.I0(or1[4]),.I1(or2[4]),.O(in[4])),
    org5(.I0(or1[5]),.I1(or2[5]),.O(in[5])),
    org6(.I0(or1[6]),.I1(or2[6]),.O(in[6])),
    org7(.I0(or1[7]),.I1(or2[7]),.O(in[7]));
```

```
AND2
   a0(.I0(Pbar),.I1(s_in),.O(or1[0])),
   a1(.I0(Pbar),.I1(out[0]),.O(or1[1])),
   a2(.I0(Pbar),.I1(out[1]),.O(or1[2])),
   a3(.I0(Pbar),.I1(out[2]),.O(or1[3])),
   a4(.I0(Pbar),.I1(out[3]),.O(or1[4])),
   a5(.I0(Pbar),.I1(out[4]),.O(or1[5])),
   a6(.I0(Pbar),.I1(out[5]),.O(or1[6])),
   a7(.I0(Pbar),.I1(out[6]),.O(or1[7])),
   b0(.I0(S_L),.I1(p_in[0]),.O(or2[0])),
   b1(.I0(S_L),.I1(p_in[1]),.O(or2[1])),
   b2(.I0(S_L),.I1(p_in[2]),.O(or2[2])),
   b3(.I0(S_L),.I1(p_in[3]),.O(or2[3])),
   b4(.I0(S_L),.I1(p_in[4]),.O(or2[4])),
   b5(.I0(S_L),.I1(p_in[5]),.O(or2[5])),
   b6(.I0(S_L),.I1(p_in[6]),.O(or2[6])),
   b7(.I0(S_L),.I1(p_in[7]),.O(or2[7]));

INV
   inv(.I(S_L),.O(Pbar));
```

② 通过波形仿真进行检查。

```
initial begin
   // Initialize Inputs
   clk = 0;S_L = 0;s_in = 0;p_in = 0;#100;
   // Add stimulus here
   S_L = 0;s_in = 1;p_in =0;#200;
   S_L = 1;s_in = 0;p_in = 8'b0101_0101;#500;
end
always begin
   clk = 0; #20;
   clk = 1; #20;
end
```



2. 设计主板 LED 灯驱动模块

① 建立 LEDP2S 工程，通过行为描述方式设计 LED 驱动 LEDDRV 模块。时钟使用主板时钟取反表示。

```
initial begin
   fin=1'b1;
   con=4'b1111;
   EN=1;
   DO=0;
   CLR=1;
end
always@(posedge clk) begin
   if(en) begin
      fin<=0;
      con<=4'b1110;
      DO<=HEX[4'b1111];
      CLR<=1'b1;
      EN<=1;
   end
   else if(!fin) begin
      con<=con-1;
      DO<=HEX[con];
      CLR<=1'b1;
      EN<=1;
      if(con==4'b1111) fin<=1;
   end
end

assign LED_CLK=(fin)?0:~clk;
assign LED_CLR=CLR;
assign LED_DO=DO;
assign LED_EN=EN;
```

② 使用已有模块搭建 LED 灯显示器的 top 模块。使用四个按钮并加防抖动作为更改数值的输入，并以 1ms 时钟为上升沿时间。

```
wire [15:0]Hex;
wire [3:0]btn_out;
wire [31:0]clk_div;

clkdiv
   cldv(.clk(clk),.rst(1'b0),.clkdiv(clk_div));

pbdebounce
   p0(clk_div[17],btn[0],btn_out[0]),
   p1(clk_div[17],btn[1],btn_out[1]),
   p2(clk_div[17],btn[2],btn_out[2]),
   p3(clk_div[17],btn[3],btn_out[3]);

CreateNumber
   CN(.btn(btn_out),.num(Hex[15:0]));

DispNum
   DN(Hex[15:0],4'b0000,4'b0000,clk,1'b0,AN[3:0],SEGMENT[7:0]);

LED_DRV
   LDRV(clk,sw,Hex,LED_CLK,LED_CLR,LED_DO,LED_EN);

assign BTNX4=1'b0;
```

③ 完成引脚约束文件 K7.ucf。生成 bit 文件，并在实验板上验证功能。

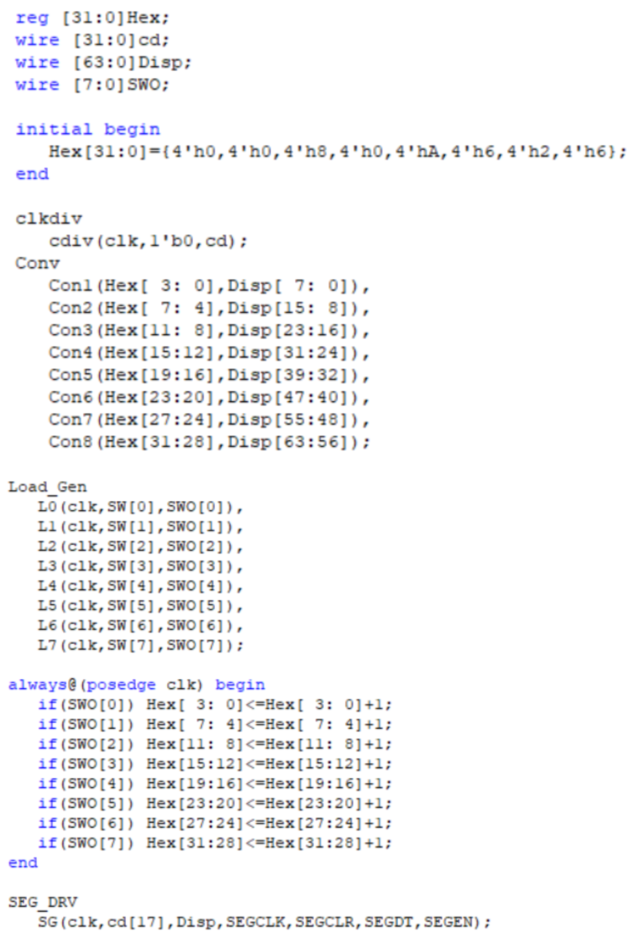
```
NET "clk" LOC = AC18 | IOSTANDARD = LVCMOS18;
NET "clk" TNM_NET = TM_CLK;
TIMESPEC TS_CLK_100M = PERIOD "TM_CLK" 10 ns HIGH 50%;

NET "SEGMENT[0]" LOC = AB22 | IOSTANDARD = LVCMOS33;#a
NET "SEGMENT[1]" LOC = AD24 | IOSTANDARD = LVCMOS33;#b
NET "SEGMENT[2]" LOC = AD23 | IOSTANDARD = LVCMOS33;#c
NET "SEGMENT[3]" LOC = Y21 | IOSTANDARD = LVCMOS33;#d
NET "SEGMENT[4]" LOC = W20 | IOSTANDARD = LVCMOS33;#e
NET "SEGMENT[5]" LOC = AC24 | IOSTANDARD = LVCMOS33;#f
NET "SEGMENT[6]" LOC = AC23 | IOSTANDARD = LVCMOS33;#g
NET "SEGMENT[7]" LOC = AA22 | IOSTANDARD = LVCMOS33;#point

NET "AN[0]" LOC = AD21 | IOSTANDARD = LVCMOS33;
NET "AN[1]" LOC = AC21 | IOSTANDARD = LVCMOS33;
NET "AN[2]" LOC = AB21 | IOSTANDARD = LVCMOS33;
NET "AN[3]" LOC = AC22 | IOSTANDARD = LVCMOS33;

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

3. 设计主板七段数码管驱动模块

① 建立 SEGP2S 工程，通过行为描述方式设计七段管驱动 SEGDRV 模块。时钟使用主板时钟取反表示。

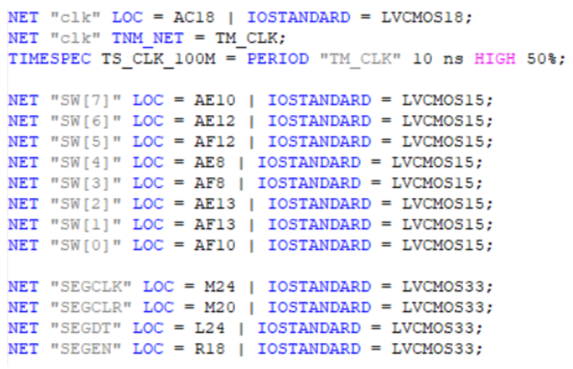
```
reg [5:0]con;
reg CLR,DO,EN;
reg fin;

initial begin
   fin=1'b1;
   con=6'b111111;
   EN=1;
   DO=0;
   CLR=1;
end
always@(posedge clk) begin
   if(en) begin
      fin<=0;
      con<=6'b111110;
      DO<=HEX[6'b111111];
      CLR<=1'b1;
      EN<=1;
   end
   else if(!fin) begin
      con<=con-1;
      DO<=HEX[con];
      CLR<=1'b1;
      EN<=1;
      if(con==6'b111111) fin<=1;
   end
end

assign SEGCLK=(fin)?0:~clk;
assign SEGCLR=CLR;
assign SEGDO=DO;
assign SEGEN=EN;
```

② 使用已有模块搭建七段管显示器的 top 模块。使用八个开关并加防抖动作为更改数值的输入，并以 1ms 时钟为刷新时间。



```
reg [31:0]Hex;
wire [31:0]cd;
wire [63:0]Disp;
wire [7:0]SWO;

initial begin
   Hex[31:0]={4'h0,4'h0,4'h8,4'h0,4'hA,4'h6,4'h2,4'h6};
end

clkdiv
   cdiv(clk,1'b0,cd);
Conv
   Con1(Hex[ 3: 0],Disp[ 7: 0]),
   Con2(Hex[ 7: 4],Disp[15: 8]),
   Con3(Hex[11: 8],Disp[23:16]),
   Con4(Hex[15:12],Disp[31:24]),
   Con5(Hex[19:16],Disp[39:32]),
   Con6(Hex[23:20],Disp[47:40]),
   Con7(Hex[27:24],Disp[55:48]),
   Con8(Hex[31:28],Disp[63:56]);

Load_Gen
   L0(clk,SW[0],SWO[0]),
   L1(clk,SW[1],SWO[1]),
   L2(clk,SW[2],SWO[2]),
   L3(clk,SW[3],SWO[3]),
   L4(clk,SW[4],SWO[4]),
   L5(clk,SW[5],SWO[5]),
   L6(clk,SW[6],SWO[6]),
   L7(clk,SW[7],SWO[7]);

always@(posedge clk) begin
   if(SWO[0]) Hex[ 3: 0]<=Hex[ 3: 0]+1;
   if(SWO[1]) Hex[ 7: 4]<=Hex[ 7: 4]+1;
   if(SWO[2]) Hex[11: 8]<=Hex[11: 8]+1;
   if(SWO[3]) Hex[15:12]<=Hex[15:12]+1;
   if(SWO[4]) Hex[19:16]<=Hex[19:16]+1;
   if(SWO[5]) Hex[23:20]<=Hex[23:20]+1;
   if(SWO[6]) Hex[27:24]<=Hex[27:24]+1;
   if(SWO[7]) Hex[31:28]<=Hex[31:28]+1;
end

SEG_DRV
   SG(clk,cd[17],Disp,SEGCLK,SEGCLR,SEGDT,SEGEN);
```

③ 完成引脚约束文件 K7.ucf。生成 bit 文件，上板测试。



```
NET "clk" LOC = AC18 | IOSTANDARD = LVCMOS18;
NET "clk" TNM_NET = TM_CLK;
TIMESPEC TS_CLK_100M = PERIOD "TM_CLK" 10 ns HIGH 50%;

NET "SW[7]" LOC = AE10 | IOSTANDARD = LVCMOS15;
NET "SW[6]" LOC = AE12 | IOSTANDARD = LVCMOS15;
NET "SW[5]" LOC = AF12 | IOSTANDARD = LVCMOS15;
NET "SW[4]" LOC = AE8 | IOSTANDARD = LVCMOS15;
NET "SW[3]" LOC = AF8 | IOSTANDARD = LVCMOS15;
NET "SW[2]" LOC = AE13 | IOSTANDARD = LVCMOS15;
NET "SW[1]" LOC = AF13 | IOSTANDARD = LVCMOS15;
NET "SW[0]" LOC = AF10 | IOSTANDARD = LVCMOS15;

NET "SEGCLK" LOC = M24 | IOSTANDARD = LVCMOS33;
NET "SEGCLR" LOC = M20 | IOSTANDARD = LVCMOS33;
NET "SEGDT" LOC = L24 | IOSTANDARD = LVCMOS33;
NET "SEGEN" LOC = R18 | IOSTANDARD = LVCMOS33;
```

三、实验数据记录和处理

在 LEDP2S 实验中，实验板上的四个按钮分别对应 LED 灯四个显示数码的增加。在 SEGP2S 实验中，八个开关分别对应数码管八个显示数码的增加。显示正常，刷新频率正常。

四、实验结果与分析

1. 实验结果正常，已经过验收。

五、心得

在调试过程中，先写好大致框架，手动通过开关模拟时钟信号，再通过时钟分频器加上较低频率的时钟信号有助于调试。