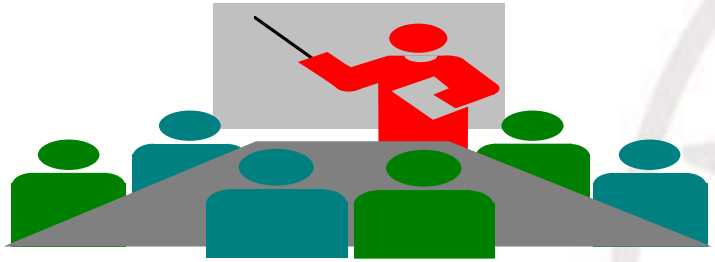


逻辑与计算机设计基础实验与课程设计

实验六

7段码显示译码器设计

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. 掌握七数码管显示原理
2. 掌握七段码显示译码设计
3. 掌握多位数码管扫描显示控制
4. 进一步熟悉ISE平台，利用原理图综合学习Verlog-HDL语言

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 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

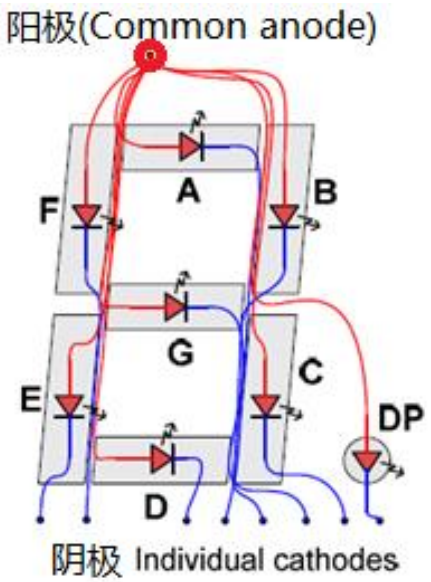
1. 设计十六进制通用七段显示译码电路
2. 仿真测试并封装MC14495兼容显示译码器
3. 设计实现四位十六进制数动态扫描显
4. 设计实现八位十六进制数静态显示*
5. 学习显示译码电路的HDL描述方法

Course Outline

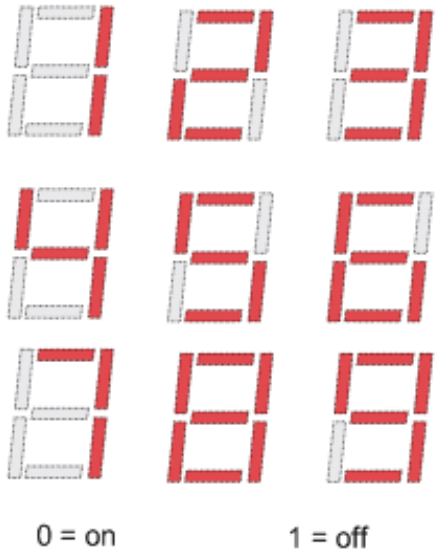
- 实验目的与实验环境
- 实验任务
- 实验原理
- 实验操作与实现

7段显示(器)结构

- 由7+1个LED构成的数字显示器件
- 每个LED显示数字的一段，另一个为小数点
- LED的正极(负极)连在一起，另一端作为点亮的控制
  - 共阳：正极连在一起，负极＝0，点亮
  - 共阴：负极连在一起，正极＝1，点亮



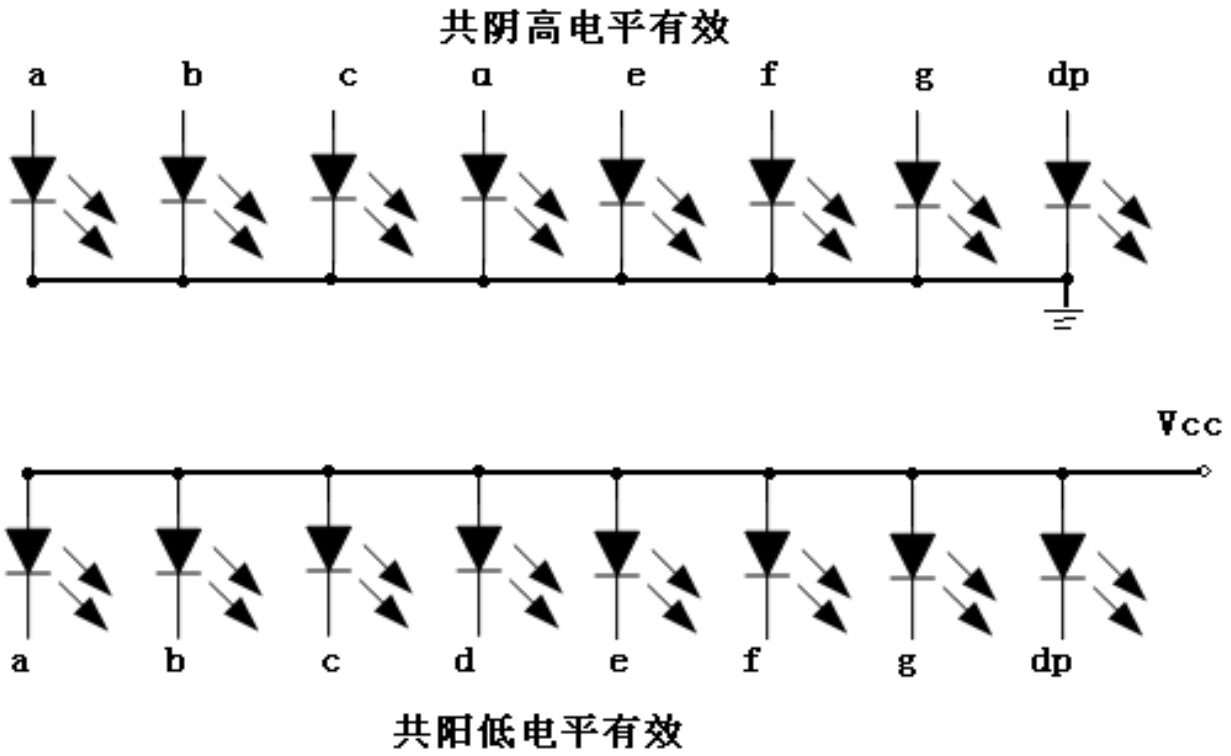
X	a	b	c	d	e	f	g
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	0	1	0	0
A	0	0	0	1	0	0	0
B	1	1	0	0	0	0	0
C	0	1	1	0	0	0	1
D	1	0	0	0	0	1	0
E	0	1	1	0	0	0	0
F	0	1	1	1	0	0	0



共阳7段码显示器结构和显示原理

共阳连接：8个LED正极连在一起，负极低电平时点亮

共阴连接：8个LED负极连在一起，正极高电平时点亮



Hex 7- segment decoder

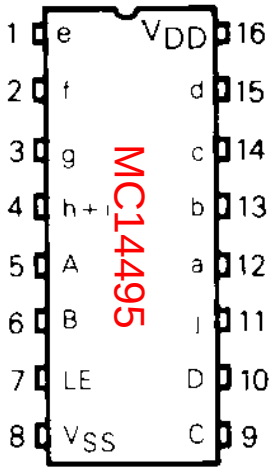
common anode

兼容MC14495

略掉：

Pin11=VCR

Pin4=h+i



其它

共阳：74LS46/47

共阴：74LS48/49

CMOS4511

Hex	$D_3D_2D_1D_0$	BI/LE	a	b	c	d	e	f	g	p
0	0 0 0 0	1	0	0	0	0	0	0	1	p
1	0 0 0 1	1	1	0	0	1	1	1	1	p
2	0 0 1 0	1	0	0	1	0	0	1	0	p
3	0 0 1 1	1	0	0	0	0	1	1	0	p
4	0 1 0 0	1	1	0	0	1	1	0	0	p
5	0 1 0 1	1	0	1	0	0	1	0	0	p
6	0 1 1 0	1	0	1	0	0	0	0	0	p
7	0 1 1 1	1	0	0	0	1	1	1	1	p
8	1 0 0 0	1	0	0	0	0	0	0	0	P
9	1 0 0 1	1	0	0	0	0	1	0	0	P
A	1 0 1 0	1	0	0	0	1	0	0	0	P
B	1 0 1 1	1	1	1	0	0	0	0	0	P
C	1 1 0 0	1	0	1	1	0	0	0	1	P
D	1 1 0 1	1	1	0	0	0	0	1	0	P
E	1 1 1 0	1	0	1	1	0	0	0	0	P
F	1 1 1 1	1	0	1	1	1	0	0	0	P
X	x x x x	0	1	1	1	1	1	1	1	1

Hex to 7-segment decoder: Simplifying

a

0	1	0	0
1	0	0	0
0	1	0	0
0	0	1	0

b

0	0	0	0
0	1	0	1
1	0	1	1
0	0	1	0

c

0	0	0	1
0	0	0	0
1	0	1	1
0	0	0	0

d

0	1	0	0
1	0	1	0
0	0	1	0
0	0	0	1

e

0	1	1	0
1	1	1	0
0	0	0	0
0	1	0	0

$$a = \bar{D}_3\bar{D}_2\bar{D}_1D_0+\bar{D}_3D_2\bar{D}_1\bar{D}_0+D_3\bar{D}_2D_1D_0+D_3\bar{D}_2D_1\bar{D}_0$$

$$b = \bar{D}_3D_2\bar{D}_1D_0+D_2D_1\bar{D}_0+D_3D_2\bar{D}_0+D_3D_1D_0$$

$$c = \bar{D}_3\bar{D}_2D_1\bar{D}_0+D_3D_2\bar{D}_0+D_3D_2D_1$$

有错误请修整

f

0	1	1	1
0	0	1	0
0	1	0	0
0	0	0	0

g

1	1	0	0
0	0	1	0
1	0	0	0
0	0	0	0

$$d = \bar{D}_3\bar{D}_2\bar{D}_1D_0+\bar{D}_3D_2\bar{D}_1\bar{D}_0+D_2D_1D_0+\bar{D}_3D_2D_1\bar{D}_0$$

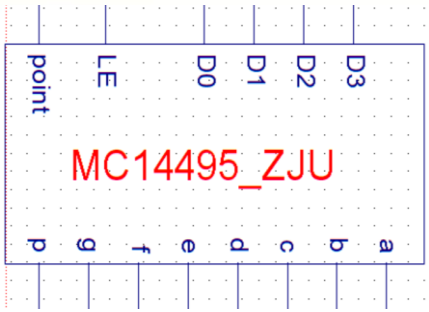
$$e = \bar{D}_3D_0+\bar{D}_3D_2\bar{D}_1+\bar{D}_2\bar{D}_1D_0$$

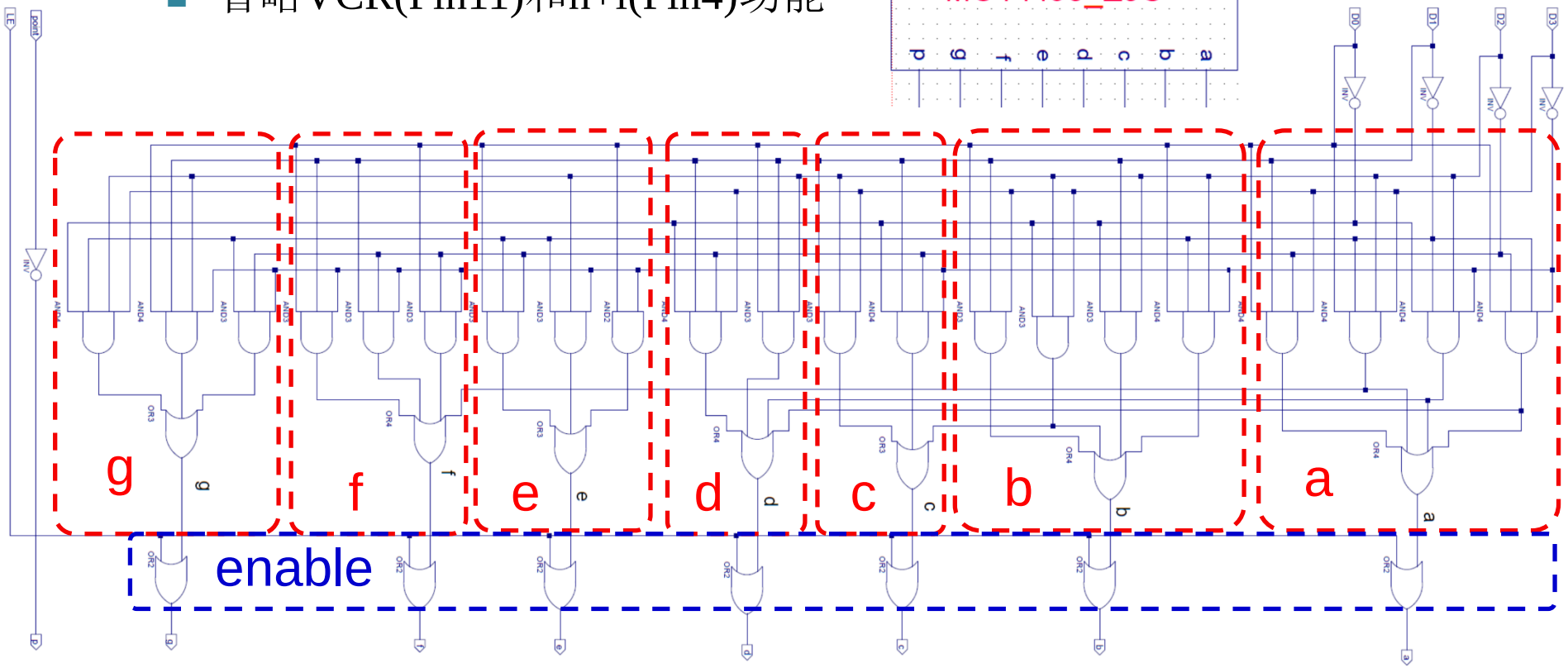
$$f = \bar{D}_3\bar{D}_2D_0+\bar{D}_3\bar{D}_2D_1+\bar{D}_3D_1D_0+D_3D_2\bar{D}_1D_0$$

$$g = \bar{D}_3\bar{D}_2\bar{D}_1+\bar{D}_3D_2D_1D_0+D_3D_2\bar{D}_1\bar{D}_0$$

Hex to 7-segment decoder Schematic

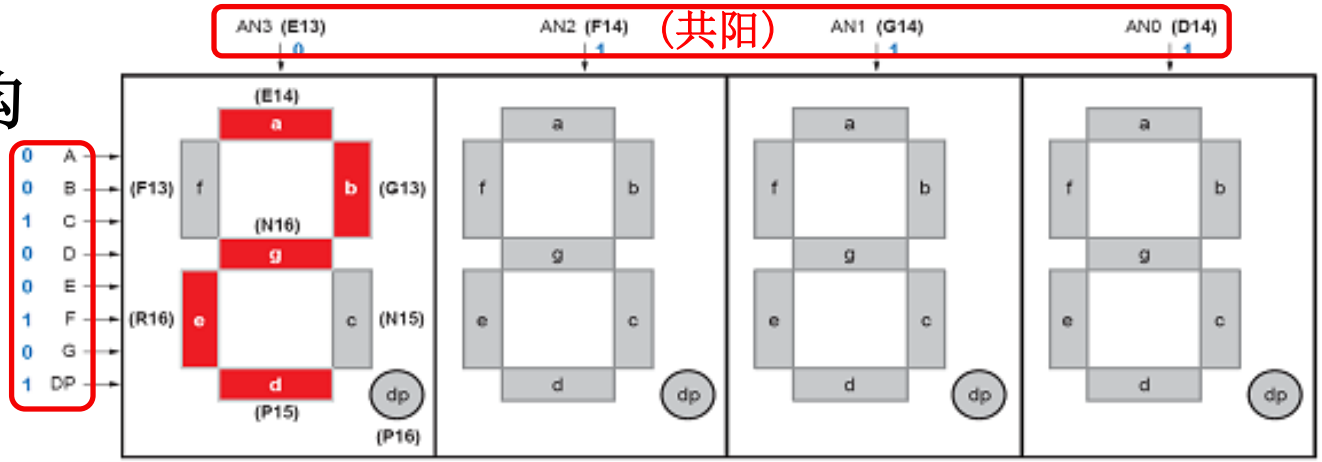
- 兼容MC14495
  - 省略VCR(Pin11)和h+i(Pin4)功能





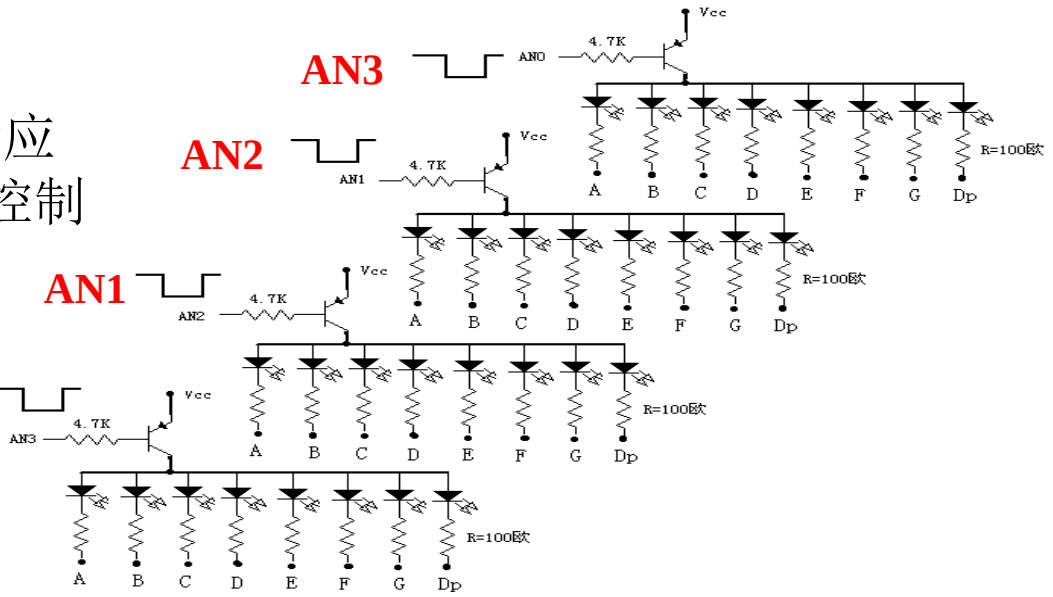
多位七段数码管显示原理

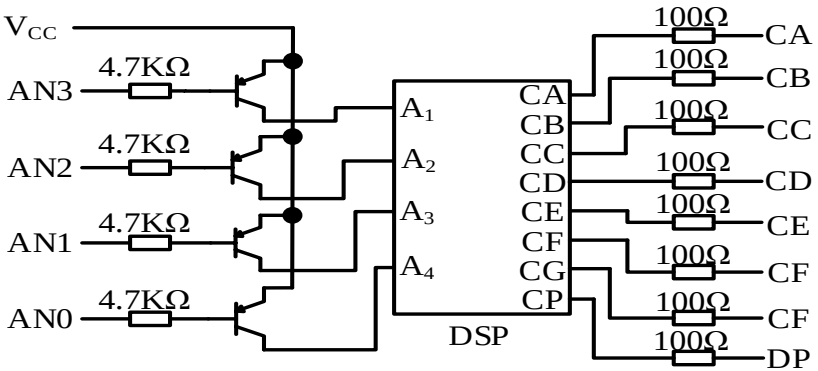
- **静态显示**
  - 每个7段码对应一个显示译码电路
- **动态扫描显示：时分复用显示**
  - 利用人眼视觉残留
  - 一个7段码译码电路分时为每个7段码提供译码
- **控制时序**
  - 用定时计数信号控制公共极，分时输出对应七段码的显示信号：
    动态扫描
- **4位七段码结构**
  - 正极：公共端
  - 七段信号并联



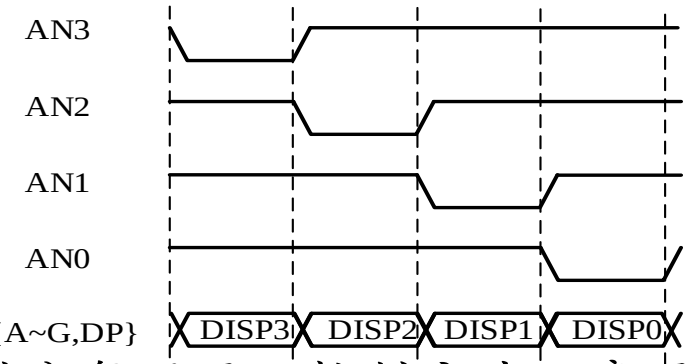
分时控制示意

- 动态扫描
  - 低电平与输入显示对应
    - 共阳：但低电平控制
  - 分时送***a~g，p***
  - 可用序列信号控制





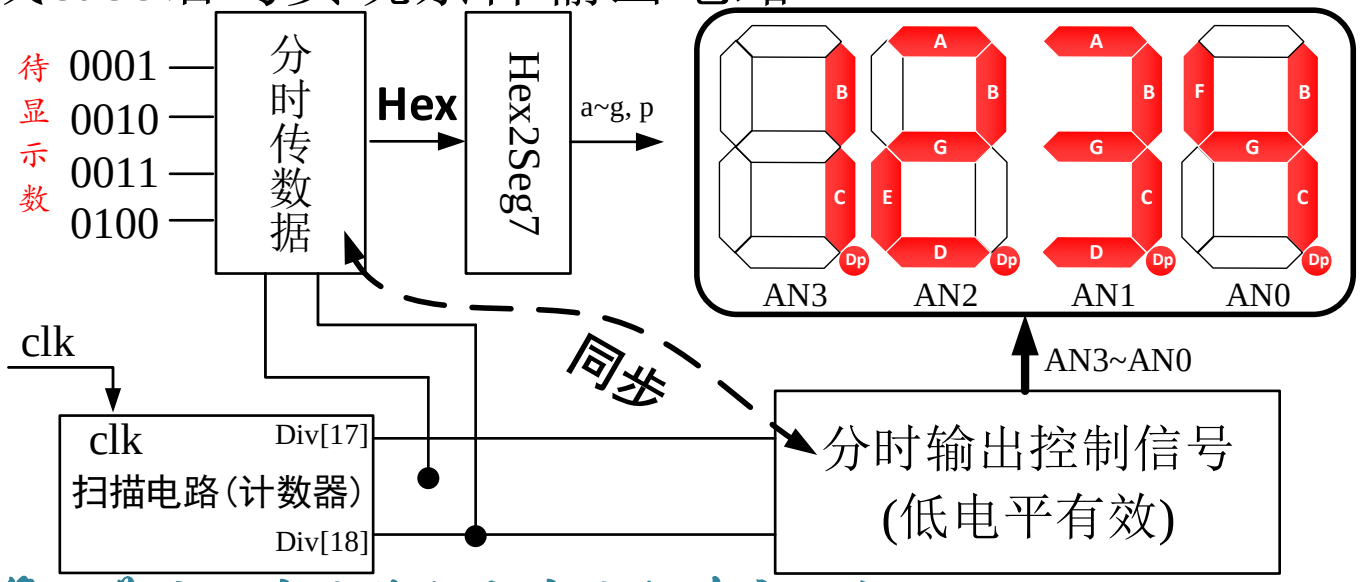
七段数码管显示控制电路图



时分复用显示控制方法示意图

四位七段动态显示控制

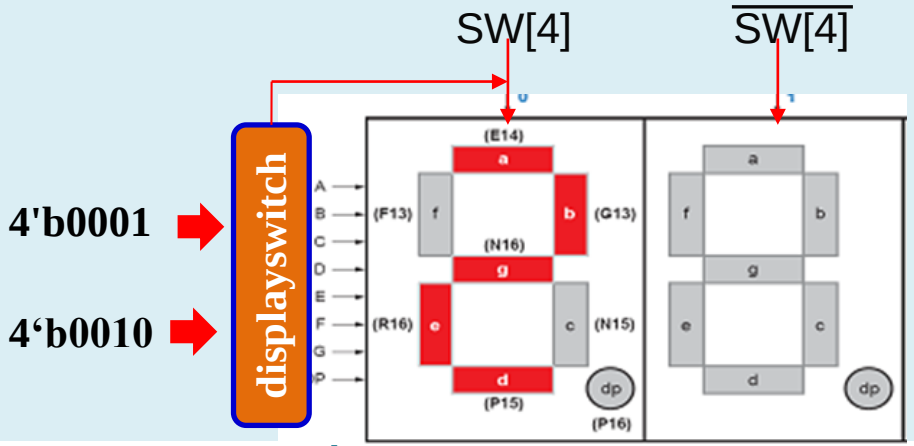
- 动态扫描显示方案
  - 扫描信号来自计数器：**时序转化为组合电路**
  - 由板载时钟clk(50MHz)作为计数器时钟，分频后输入到数据选择器的控制端，作为数码管扫描信号
  - 计数器的分频系数要适当，眼睛舒适即可
- 条件语句实现
  - if_then 或case语句实现条件输出电路



二位七段动态显示实现

- 使用if_then语句实现条件输出

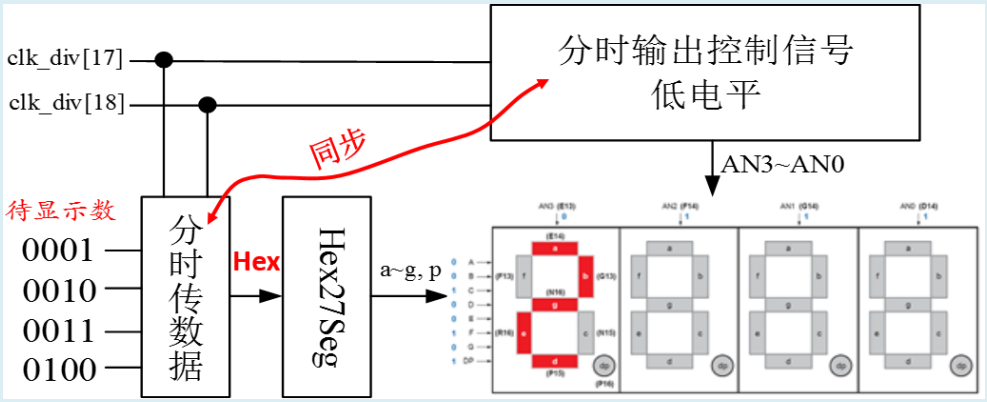
```
module displayswitch( input clk,                    //端口变量说明与定义合并
                      input[7:0]Hexs,
                      output reg[3:0]AN,
                      output reg[3:0]Hex);
  assign AN[0] = SW[0];
  assign AN[1] = ~SW[0];
  assign AN[2] = 1;                    //不用
  assign AN[3] = 1;                    //不用
  always @ (posedge clk) begin         //时钟触发(也可以信号变化触发)
    if (SW[4])  Hex <= Hexs[7:4];      //显示"1" Hexs[7:4]=4'b0001
    else        Hex <= Hexs[3:0];      //显示"0" Hexs[3:0]=4'b0010
    end
endmodule
```



使用Case语句实现条件输出

四位七段动态显示实现

```
module dispsync(input   [15:0] Hexs,          //端口变量说明与定义合并
                input   [1:0] Scan,
                input   [3:0] point,
                input   [3:0] blink,
                output reg[3:0] Hex,
                output reg p,LE,
                output reg[3:0] AN);
  always @* begin                             //信号变化触发 (组合电路不用时钟触发)
      case (Scan)
          2'b00 : begin Hex <= Hexs[3:0];    AN <= 4'b 1110; …   //同步输出
          2'b01 : begin Hex <= Hexs[7:4];    AN <= 4'b 1101; …   //同步输出
          2'b10 : begin Hex <= Hexs[11:8];   AN <= 4'b 1011; …   //同步输出
          2'b11 : begin Hex <= Hexs[15:12]; AN <= 4'b 0111; …   //同步输出
      endcase
  end

endmodule
```



always 深入学习

- 过程描述语句always
  - 对于复杂的电路行为，用assign赋值无法描述
  - always可以用来描述复杂的信号传输过程
  - 赋值必须是reg变量，但结果由电路综合决定，不一定是reg

```
Always @ (触发表达式)begin
    //过程赋值
    //条件语句
    ……
end
```

- 触发事件表达式有：
  - 组全电路使用变量：
    (a)、(a or b)，可用“*”代替
  - 时序电路用时钟边沿：
    (posedge clk)　　//上升沿
    (negdge clk)　　//下降沿
    (posedge clk or negdge res)
    //上下边沿

- 赋值
  - 非阻塞赋值：<=，触发边沿同时赋值(并行)
  - 阻塞赋值：　　=，先赋值再同步，较难把握(组合电路)
- 辅助模块：时钟分频模块
  - 实验经常需要用到各种不同频率的时钟
  - 在要求不高时可以用计数器分频获取

辅助模块：时钟计数分频器

- **32位时钟计数分频器**
  - 可输出2-2^{32}分频信号，可用于一般非同步类时钟信号
  - 延时较高，要求不高的时钟也可以用
  - 本实验多位七段显示器动态扫描可用

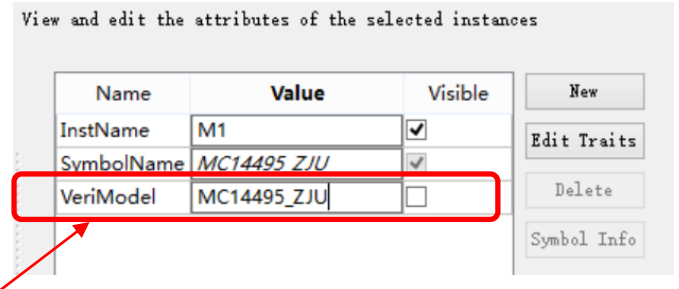
```
module  clkdiv(input clk,
               input rst,
               output reg[31:0]clkdiv
               );

// Clock divider-时钟分频器

   always @ (posedge clk or posedge rst) begin
      if (rst) clkdiv <= 0;
      else clkdiv <= clkdiv + 1'b1;
   end

endmodule
```

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：Hex27Seg

- **设计实现十六进制七段显示译码器**
  - 兼容MC14495
  - 省略Pin11=VCR和Pin4=h+i功能
- **仿真验证**
  - 设计时序仿真激励代码
  - 仿真通过封装，名称： MC14495_ZJU
- **说明**
  - 电路描述文档命名约定(非后缀)
    - 原理：???_sch
    - 代码：???
    - 测试：???_test
    - 封闭：原理图模块封装模块符号去掉“_sch”
      - 符号改名用另存
      - 注意校对模块调用名称



设计要点

◎新建工程： Hex27Seg

- Hex to seven segment decoder

◎设计七段显示模块

- 模块命名：MC14495_ZJU
- 验证表达式并原理图输入

$$a = \bar{D}_3\bar{D}_2\bar{D}_1D_0+\bar{D}_3D_2\bar{D}_1\bar{D}_0+D_3\bar{D}_2D_1D_0+D_3\bar{D}_2D_1\bar{D}_0$$

$$b = \bar{D}_3D_2\bar{D}_1D_0+D_2D_1\bar{D}_0+D_3D_2\bar{D}_0+D_3D_1D_0$$

$$c = \bar{D}_3\bar{D}_2D_1\bar{D}_0+D_3D_2\bar{D}_0+D_3D_2D_1$$

$$d = \bar{D}_3\bar{D}_2\bar{D}_1D_0+\bar{D}_3D_2\bar{D}_1\bar{D}_0+D_2D_1D_0+\bar{D}_3D_2D_1\bar{D}_0$$

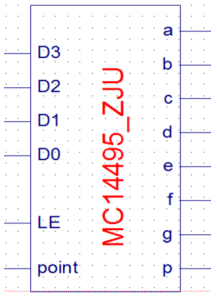
$$e = \bar{D}_3D_0+\bar{D}_3D_2\bar{D}_1+\bar{D}_2\bar{D}_1D_0$$

$$f = \bar{D}_3\bar{D}_2D_0+\bar{D}_3\bar{D}_2D_1+\bar{D}_3D_1D_0+D_3D_2\bar{D}_1D_0$$

$$g = \bar{D}_3\bar{D}_2\bar{D}_1+\bar{D}_3D_2D_1D_0+D_3D_2\bar{D}_1\bar{D}_0$$

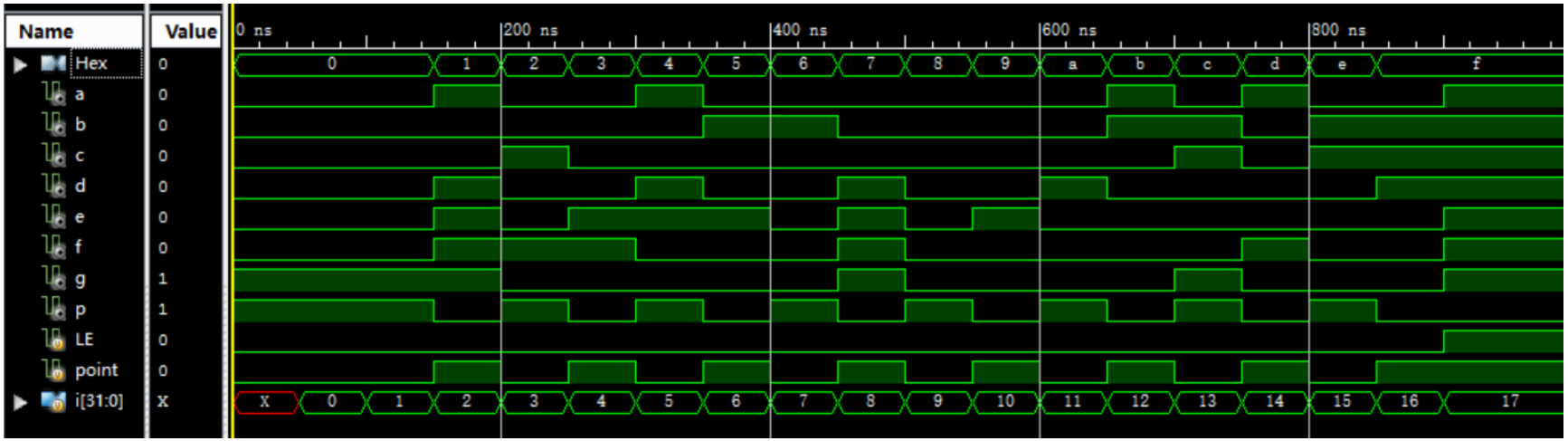
◎ 仿真测试

- 设计测试代码
- 封装符号图并修改




```
for (i=0; i<=15;i=i+1) begin
#50;
{D3,D2,D1,D0}=i;
point = i;
end

#50;
i=i+1;
assign LE = 1;
end
```

参考激励

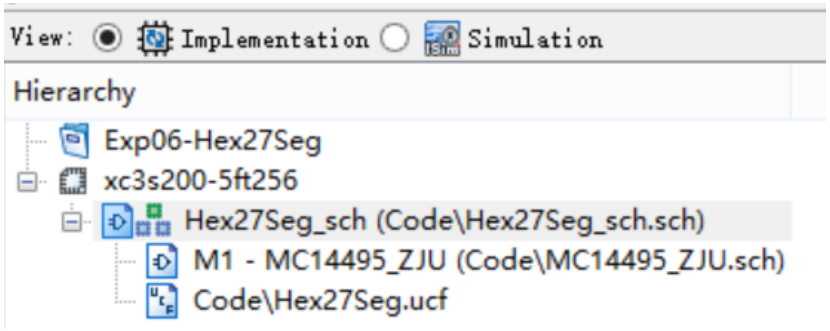


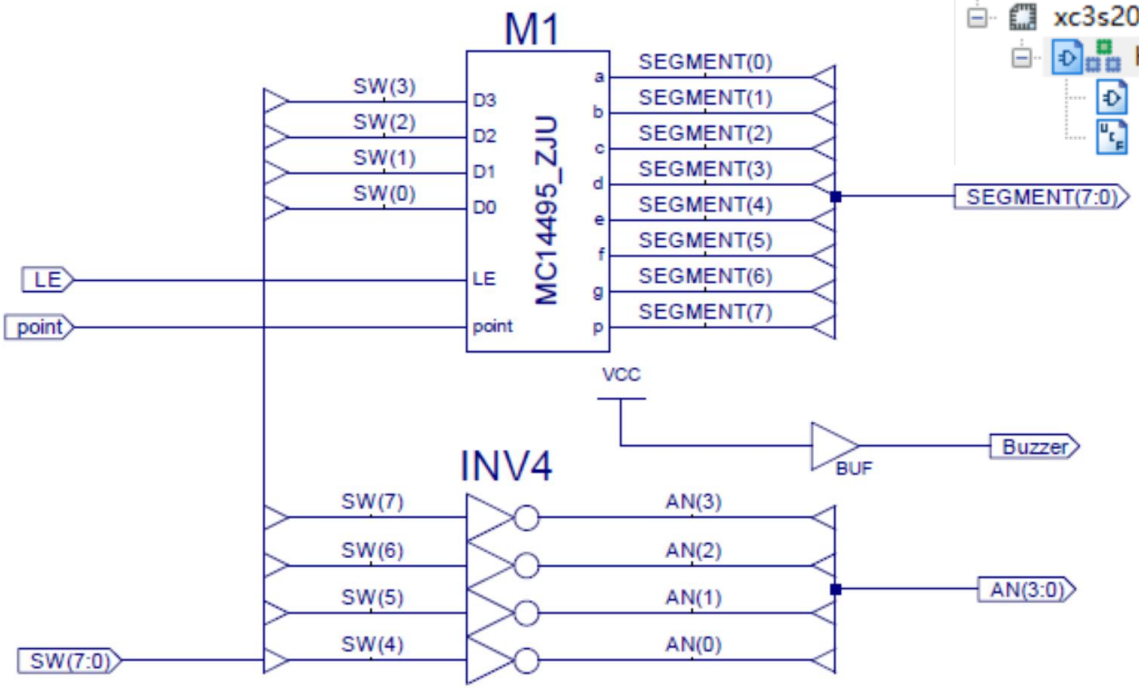
◎ 学习Veri代码描述

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

◎ 设计顶层模块

- Hex27Seg_sch，设置为顶层模块
- 调用MC14495_ZJU模块

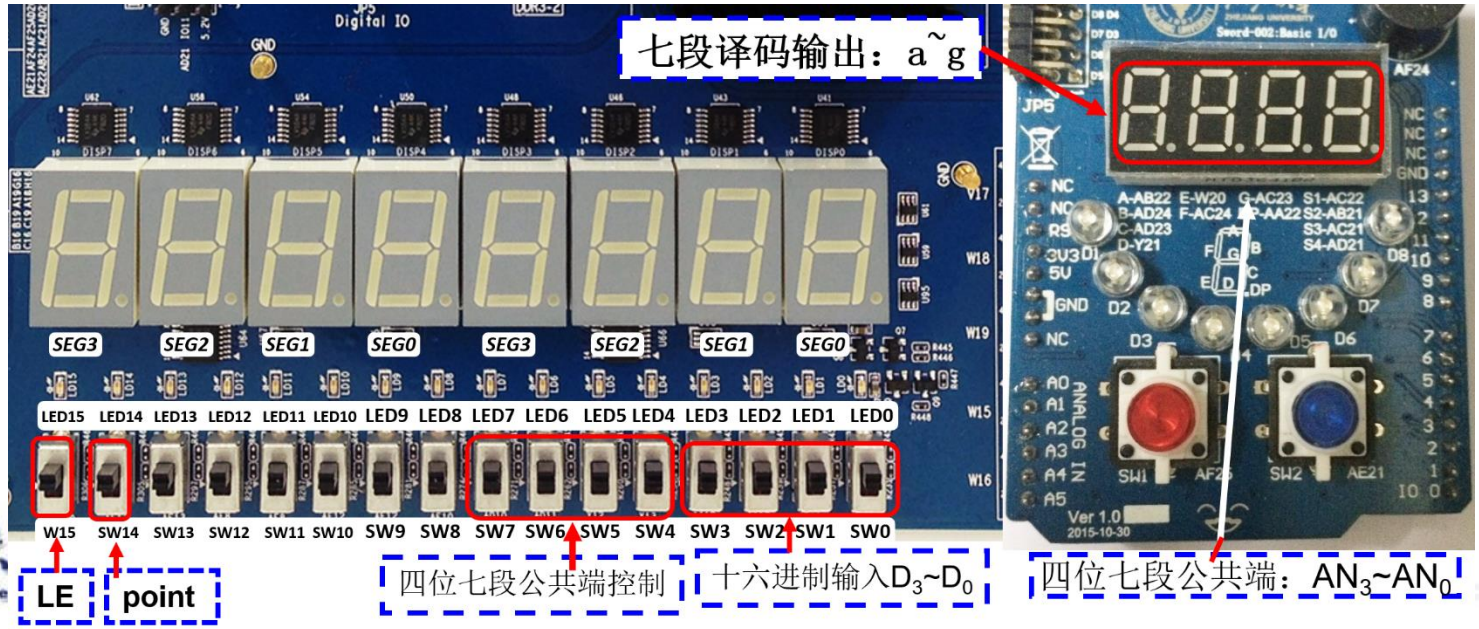




物理验证

- UCF引脚定义
  - 输入
    - SW[3:0]=AN[3:0]
    - SW[7:4]=$D_3D_2D_1D_0$
    - SW[14]=p
    - SW[15]=LE
  - 输出
    - SEGMENT=g~a，p，AN=AN
- 根据设计修改UCF

```
#MC14495-ZJU
#switch
NET "SW[0]"          LOC = AA10  | IOSTANDARD = LVCMOS15; #to AN[3]
NET "SW[1]"          LOC = AB10  | IOSTANDARD = LVCMOS15; #to AN[2]
NET "SW[2]"          LOC = AA13  | IOSTANDARD = LVCMOS15; #to AN[1]
NET "SW[3]"          LOC = AA12  | IOSTANDARD = LVCMOS15; #to AN[0]
NET "SW[4]"          LOC = Y13   | IOSTANDARD = LVCMOS15; #D3
NET "SW[5]"          LOC = Y12   | IOSTANDARD = LVCMOS15; #D2
NET "SW[6]"          LOC = AD11  | IOSTANDARD = LVCMOS15; #D1
NET "SW[7]"          LOC = AD10  | IOSTANDARD = LVCMOS15; #D0
#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 "point"          LOC = AF13  | IOSTANDARD = LVCMOS15 ;#SW[14]
NET "LE"             LOC = AF10  | IOSTANDARD = LVCMOS15 ;#SW[15]
#Arduino-Sword-002-Basic IO
NET "Buzzer"         LOC = AF24      | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[0]"     LOC = AB22      | IOSTANDARD = LVCMOS33 ;#a
NET "SEGMENT[1]"     LOC = AD24      | IOSTANDARD = LVCMOS33 ;#b
NET "SEGMENT[2]"     LOC = AD23      | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[3]"     LOC = Y21       | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[4]"     LOC = W20       | IOSTANDARD = LVCMOS33 ;
NET "SEGMENT[5]"     LOC = AC24      | IOSTANDARD = LVCMOS33 ;
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 ;
```

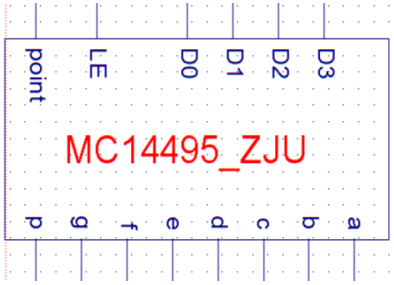


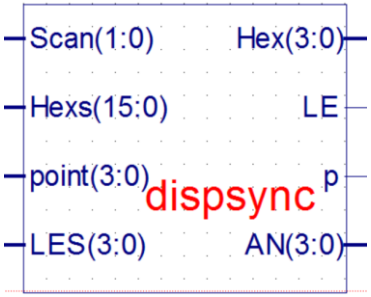
设计工程二： Hex427Seg

- 设计4位七段码显示模块
- 说明
  - 顶层模块名：Hex427Seg_sch
    - 原理图输入
    - 调用模块实现
      - 调用MC14495(从Hex27Seg工程复制)
      - 设计调用扫描同步输出模块，符号：dispsync.sym(制作)
      - 设计调用辅助时钟分频模块，符号：clkdiv.sym(制作)

设计要点

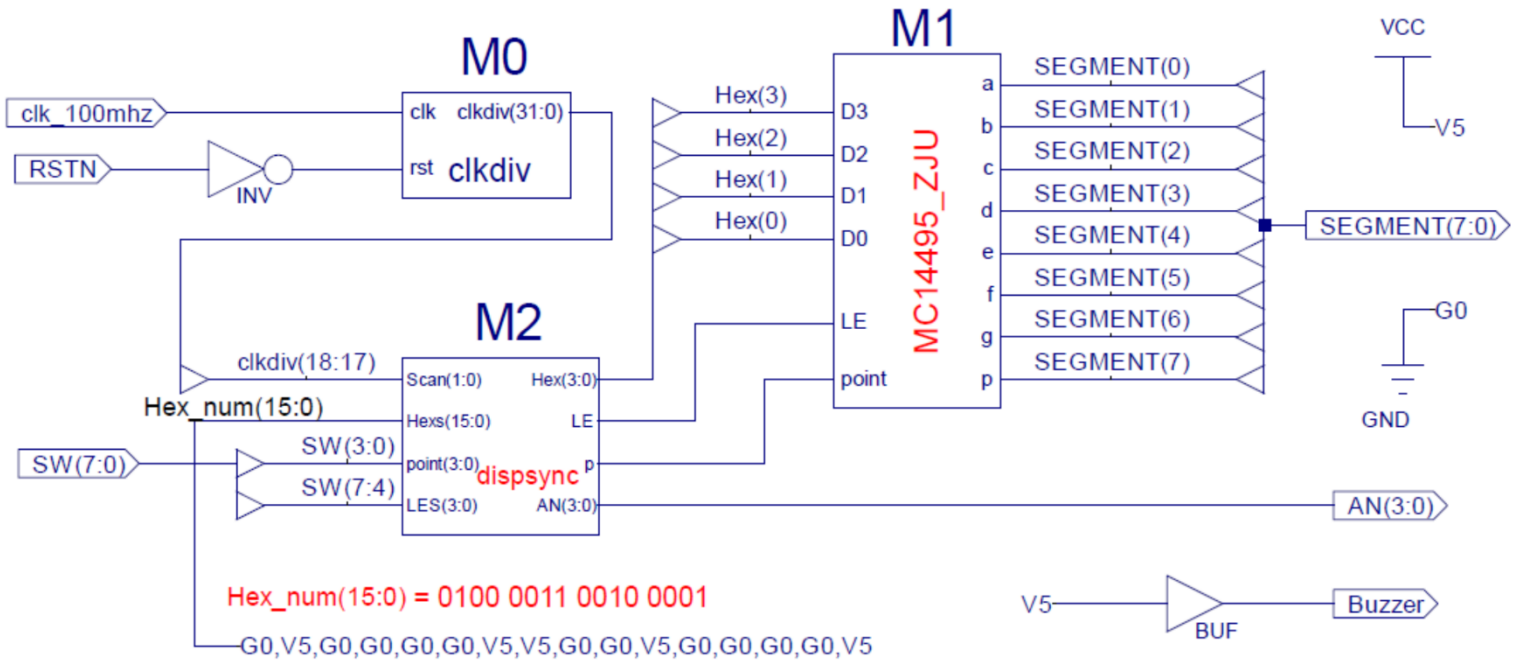
- 新建工程
  - 复制MC14495_ZJU138逻辑符号到当前工程根目录
    - MC14495.sym
  - 复制MC14495.sch代码存放目录
- **设计动态扫描同步输出模块**
  - 模块名：dispsync.v
  - 用Verilog HDL设计
  - 制作逻辑符号并修改：dispsync.syn
- **设计通用计数分频模块**
  - 模块名：clkdiv.v
  - 用Verilog HDL设计
  - 制作逻辑符号并修改：clkdiv.sym





设计要点

- 设计顶层模块: **Hex427Seg_sch.sch**
  - VCC是电源实例，用于逻辑画图，实现逻辑“1”
  - GND是地线实例，用于逻辑画图，实现逻辑“0”
  - 固定显示1234=0100-0011-0010-0001
    - 输入Hex_num= G0,V5,G0,G0,G0,G0,V5,V5,G0,G0,V5,G0,G0,G0,G0,V5

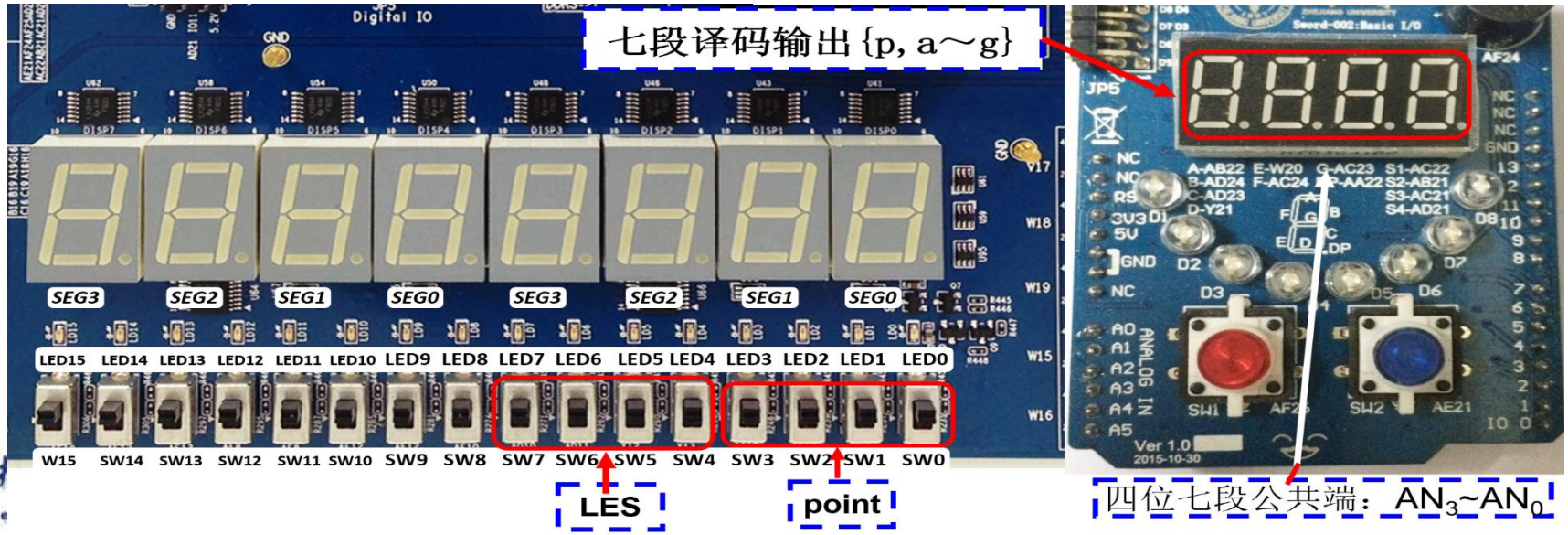


物理验证

- UCF引脚定义
  - 输入
    - 使能控制：SW[7:4]=LE[3:0]
    - 小数点输入：SW[3:0]=point[3:0]
    - 复位：RSTN(阵列键盘右边第二个红键)
  - 输出
    - a~g，p=SEGMENT
    - AN[3:0]
- 根据设计修改UCF

```
#Created by Constraints Editor (xc7k160t-ffg676-2l) - 2015/06/20
#系统时钟
NET "clk_100mhz"      LOC = AC18       | IOSTANDARD = LVCMOS18 ;
NET "clk_100mhz"      TNM_NET = TM_CLK ;
TIMESPEC TS_CLK_100M = PERIOD "TM_CLK"  10 ns HIGH 50%;

#Reset or CR
NET "RSTN"            LOC = W13        | IOSTANDARD = LVCMOS18 ;
```

新增ucf引脚

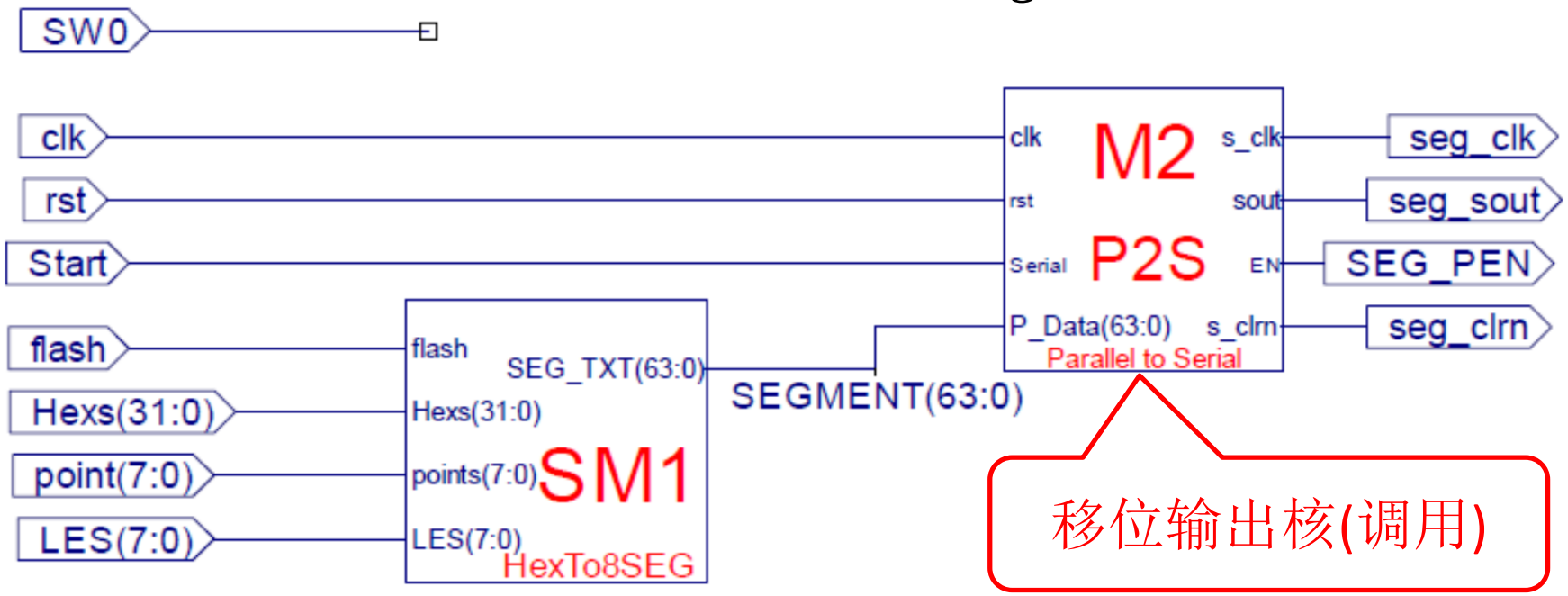


设计工程三(选修)： Hex827Seg

- 设计8位七段码显示模块
  - 二种实验方式
    - 用Arduino Sword-002子板四位动态扫描扩展
    - 用主板调用P2S模块输出静态显示
- 说明
  - 顶层模块名：Hex827Seg_sch
    - 原理图输入
    - 调用模块实现
      - 调用MC14495(Hex27Seg工程复制)
      - 调用辅助时钟分频模块，符号：clkdiv.sym(制作)
      - 修改4位七段扫描同步输出模块，符号：dispsync32.sym
        或设计八位七段静态译码模块(HexTo8SEG8)，调用P2S输出

参考设计：静态译码-调用P2S输出

- 八位七段显示器结构
  - 静态译码移位输出模块结构：Sseg_Dev




```
#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 ;
```

HexTo8SEG模块结构

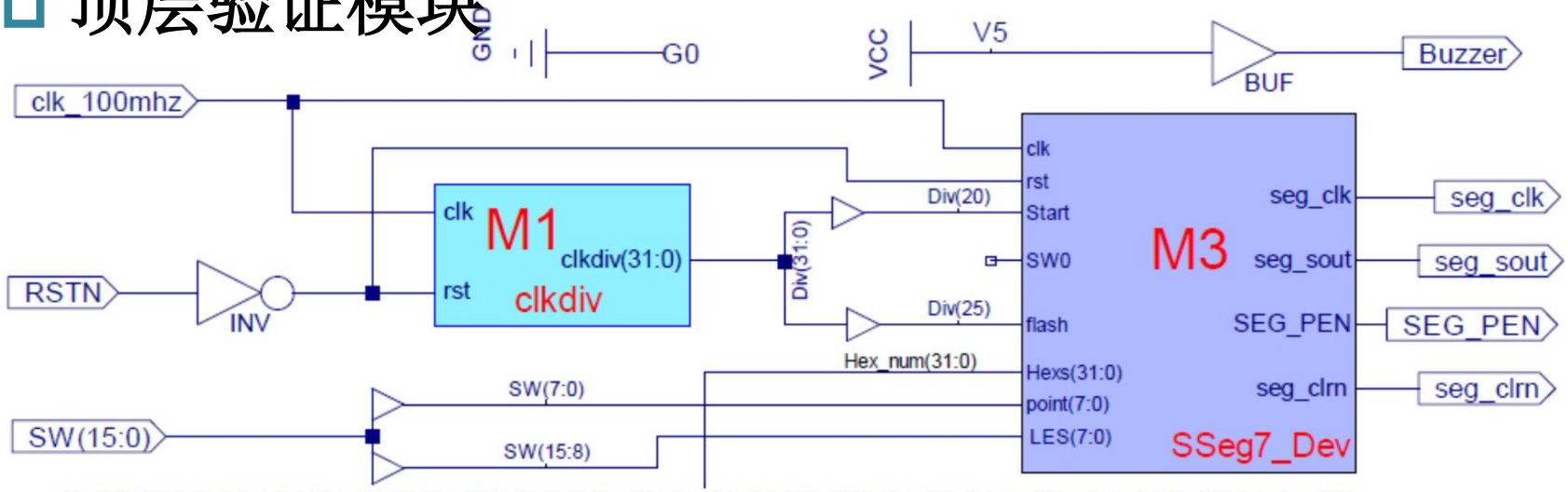
```
module  HexTo8SEG(input [31:0] Hexs,       //端口变量说明与定义合并
//                input [2:0] Scan,
                  input [7:0] points,
                  input [7:0] LES,
                  input flash,
                  output[63:0] SEG_TXT
                  );

   Hex2Seg HTS0(Hexs[31:28],LES[7],points[7],flash,SEG_TXT[7:0]);
   Hex2Seg HTS1(Hexs[27:24],LES[6],points[6],flash,SEG_TXT[15:8]);
   Hex2Seg HTS2(Hexs[23:20],LES[5],points[5],flash,SEG_TXT[23:16]);
   Hex2Seg HTS3(Hexs[19:16],LES[4],points[4],flash,SEG_TXT[31:24]);

   Hex2Seg HTS4(Hexs[15:12],LES[3],points[3],flash,SEG_TXT[39:32]);
   Hex2Seg HTS5(Hexs[11:8], LES[2],points[2],flash,SEG_TXT[47:40]);
   Hex2Seg HTS6(Hexs[7:4],  LES[1],points[1],flash,SEG_TXT[55:48]);
   Hex2Seg HTS7(Hexs[3:0],  LES[0],points[0],flash,SEG_TXT[63:56]);

endmodule
```

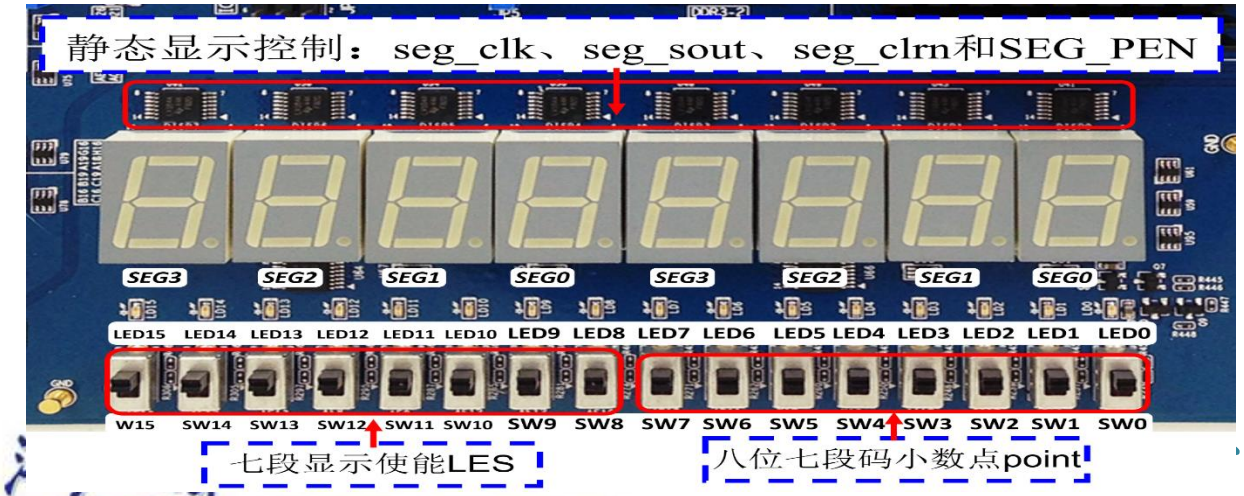
什么用途？
应输入什么？

```
module    Hex2Seg(input[3:0]Hex,
                  input LE,
                  input point,
                  input flash,
                  output[7:0]Segment
                  );
      wire en = LE & flash;
      MC14495_ZJU  MSEG(.D3(Hex[3]),.D2(Hex[2]),.D1(Hex[1]),.D0(Hex[0]),.LE(en),.point(point),
                        .a(a),.b(b),.c(c),.d(d),.e(e),.f(f),.g(g),.p(p));
      assign Segment = {a,b,c,d,e,f,g,p}; //p,g,f,e,d,c,b,a
endmodule
```

物理验证参考

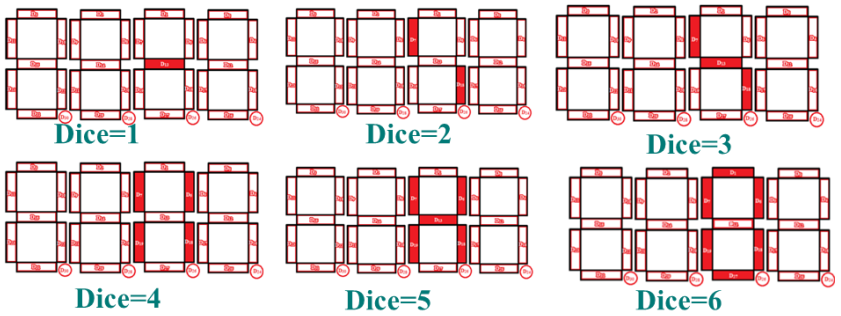
- 顶层验证模块





思考题

- 如何最简单地修改工程实现共阴七段译码？
- 如何实现多于4位十六进制数显示？
- 怎么实现七段码特殊符号显示？
  - 如习题3-32：



- 在多位七段显示中，如何使其中某位闪烁？
  - 如交通信号闪烁倒计数(秒)
  - 提示：利用使能信号LE
- 如何将一些特殊的时序电路转化为计数+组合电路？
  - 如本实验动态扫描，习题3-7也是

同学们：每次做完实验请整理好实验台，放好仪器，理清桌面。


Thank you!