**《数字逻辑》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | | **陈鹏宇，**  **帕肉合·帕尔哈提** | | **年级** | | **2020级，**  **2020级** |
| **学号** | | **20204227，**  **20201641** | | **专业、班级** | | **计算机科学与技术01班，**  **计算机科学与技术05班** |
| **实验名称** | **七段数码管显示，数字钟设计** | | | | | |
| **实验时间** | **2021.11.13** | | **实验地点** | | **Ds1410** | |
| **实验成绩** |  | | **实验性质** | | **□验证性 □设计性 □综合性** | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  评语：  评价教师签名（电子签名）： | | | | | | |
| 一、实验目的  1、掌握 7 段数码管的显示原理，学会设计八进制数、十六进制数的七段数  码管显示；   1. 学会使用显示模块和多路选择器设计组合逻辑电路。   3、通过实验，充分理解和掌握数字钟的原理，理解多进制计时器和数码管的工作原理，学会设计自己的数字钟。 | | | | | | |
| 二、实验项目内容  1、将拨码开关 SW3~SW0 以十六进制数显示在 1 个七段数码管上。  2、设计一个 3 位的 4 选 1 多路选择器，并利用拨码开关 SW15~SW14 控制选  择将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 以八进制数显示在  1 个七段数码管上。  3、将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 分别以八进制数“同  时”显示在 4 个七段数码管上。  4、封装一个显示模块 IP 核，该模块能将 16 位的二进制数以十六进制形式  显示到 4 个七段数码管上，并编写顶层模块调用该 IP 核，下载至开发板  验证其正确性。  5、实现一个六十进制数字时钟，秒到 60 则归零重加，同时让分加 1，分加  到 60 归零重加，并让小时加 1，小时加到 24 归零重加。  6、设计用于数字钟的计数器（60 进制，24 进制）并编写仿真文件，观察其  波形图并验证期正确性。  7、将设计好的计数器连接数码管并显示数值，数码管 1,0 显示秒值，数码  管 2,3 显示分值，数码管 4,5 显示小时（可以十六进制形式显示在 led 灯  上）。  8、综合、实验、生成 bit 流，下载到开发板进行验证。 | | | | | | |
| 三、实验设计  1、 实现十六进制数显示模块,将拨码开关 SW3~SW0 以十六进制数显示在1 个七段数码管上。    2、设计一个 3 位的 4 选 1 多路选择器，并利用拨码开关 SW15~SW14 控制选  择将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 以八进制数显示在  1 个七段数码管上。    3、将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 分别以八进制数“同  时”显示在 4 个七段数码管上。    4、封装一个显示模块 IP 核，该模块能将 16 位的二进制数以十六进制形式  显示到 4 个七段数码管上，并编写顶层模块调用该 IP 核，下载至开发板  验证其正确性。       1. 数字钟设计 | | | | | | |
| 四、实验过程或算法(关键步骤、核心代码注解等）  1、 实现十六进制数显示模块,将拨码开关 SW3~SW0 以十六进制数显示在1 个七段数码管上。  module hexseg(  input [3:0] hex,  input en, //使能控制输出  output an0, //使能输出  output an1,  output an2,  output an3,  output reg [0:6] segs  );  assign an1 = ~en , an2 = ~en , an3 = ~en , an0 = en; //使能赋值  always @(\*) //输入一旦变化即执行  case(hex) //abc\_defg  4'h0: segs = 7'b000\_0001;  4'h1: segs = 7'b100\_1111;  4'h2: segs = 7'b001\_0010;  4'h3: segs = 7'b000\_0110;  4'h4: segs = 7'b100\_1100;  4'h5: segs = 7'b010\_0100;  4'h6: segs = 7'b010\_0000;  4'h7: segs = 7'b000\_1111;  4'h8: segs = 7'b000\_0000;  4'h9: segs = 7'b000\_1100;  4'ha: segs = 7'b000\_1000;  4'hb: segs = 7'b110\_0000;  4'hc: segs = 7'b111\_0010;  4'hd: segs = 7'b100\_0010;  4'he: segs = 7'b011\_0000;  4'hf: segs = 7'b011\_1000;  default:  segs = 7'b111\_1111;  endcase  Endmodule  2、设计一个 3 位的 4 选 1 多路选择器，并利用拨码开关 SW15~SW14 控制选  择将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 以八进制数显示在  1 个七段数码管上。  module oct4\_1(  input [2:0] in0, in1, in2, in3,  input cn1, cn2,  input en,  output an0, an1, an2, an3,//一个数码管显示  output reg [6:0] segs  );  reg [2:0] temp; //储存临时输出变量  assign an0 = en, an1 = ~en, an2 = ~en, an3 = ~en;  always @(\*)  begin  if (!cn1 && !cn2) begin temp = in0; end //四选一  else if(!cn1 && cn2) begin temp = in1; end  else if( cn1 && !cn2) begin temp = in2; end  else begin temp = in3; end    case(temp) //数码管  3'o0 : segs = 7'b000\_0001;  3'o1 : segs = 7'b100\_1111;  3'o2 : segs = 7'b001\_0010;  3'o3 : segs = 7'b000\_0110;  3'o4 : segs = 7'b100\_1100;  3'o5 : segs = 7'b010\_0100;  3'o6 : segs = 7'b010\_0000;  3'o7 : segs = 7'b000\_1111;  default:  segs = 7'b111\_1111;  endcase  end  Endmodule  3、将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 分别以八进制数“同  时”显示在 4 个七段数码管上。  module pro2\_3(  input [2:0] in0, in1, in2, in3,  input clk,  output reg [3:0] an,  output reg [6:0] segs  );    reg [15:0] count = 21'b000;  wire sel = count[15:14];//sel信号决定亮起来的是四个数码管中的哪一个  reg [2:0] in;    parameter \_0 = 7'b000\_0001;  parameter \_1 = 7'b100\_1111;  parameter \_2 = 7'b001\_0010;  parameter \_3 = 7'b000\_0110;  parameter \_4 = 7'b100\_1100;  parameter \_5 = 7'b010\_0100;  parameter \_6 = 7'b010\_0000;  parameter \_7 = 7'b000\_1111;  always @(posedge clk)  begin  case(count[15:14])  2'b00: begin in <= in0; an <= 4'b1110; end  2'b01: begin in <= in1; an <= 4'b1101; end  2'b10: begin in <= in2; an <= 4'b1011; end  2'b11: begin in <= in3; an <= 4'b0111; end  endcase  end  always @(posedge clk)  begin  case(in)  0: segs = \_0;  1: segs = \_1;  2: segs = \_2;  3: segs = \_3;  4: segs = \_4;  5: segs = \_5;  6: segs = \_6;  7: segs = \_7;  default: segs = 7'b111\_1111;  endcase  end  always @(posedge clk)  begin  count <= count + 1;  end  endmodule  4、封装一个显示模块 IP 核，该模块能将 16 位的二进制数以十六进制形式  显示到 4 个七段数码管上，并编写顶层模块调用该 IP 核，下载至开发板  验证其正确性。  module \_2to\_16(in, an, segs, clk);  parameter len = 15;  input [len:0] in;  input clk;  output reg [3:0] an;  output reg [6:0] segs;    reg [15:0] count = 16'b000;  reg [3:0] display;    always @(posedge clk) begin  count <= count + 1;  end  always @(posedge clk) begin  case(count[15:14])  0: begin an <= 4'b1110; display <= in[3:0]; end  1: begin an <= 4'b1101; display <= in[7:4]; end  2: begin an <= 4'b1011; display <= in[11:8]; end  3: begin an <= 4'b0111; display <= in[15:12]; end  endcase  end  always @(posedge clk) begin  case(display)  4'h0: segs = 7'b000\_0001;  4'h1: segs = 7'b100\_1111;  4'h2: segs = 7'b001\_0010;  4'h3: segs = 7'b000\_0110;  4'h4: segs = 7'b100\_1100;  4'h5: segs = 7'b010\_0100;  4'h6: segs = 7'b010\_0000;  4'h7: segs = 7'b000\_1111;  4'h8: segs = 7'b000\_0000;  4'h9: segs = 7'b000\_1100;  4'ha: segs = 7'b000\_1000;  4'hb: segs = 7'b110\_0000;  4'hc: segs = 7'b111\_0010;  4'hd: segs = 7'b100\_0010;  4'he: segs = 7'b011\_0000;  4'hf: segs = 7'b011\_1000;  endcase  end  Endmodule  5、数字钟  module dig\_clk(  input clk, //100MHz  input rst, //重置  input hold, //暂停  input addm0, addh0, //加一分，加一时  output reg [3:0] an, //选通  output reg [6:0] segs //段选  );  reg sec\_clk;  reg [31:0] count = 31'b000;//分频成1Hz  reg cnt = 0;  reg [1:0] sel; //sel信号决定亮起来的是四个数码管中的哪一个  parameter //七段数码管显示数字0-9的段选值  \_0=7'b000\_0001,  \_1=7'b100\_1111,  \_2=7'b001\_0010,  \_3=7'b000\_0110,  \_4=7'b100\_1100,  \_5=7'b010\_0100,  \_6=7'b010\_0000,  \_7=7'b000\_1111,  \_8=7'b000\_0000,  \_9=7'b000\_1100;  reg [3:0] dis;  reg [3:0] m0 = 4'b0011; // 分钟和小时的个位十位  reg [3:0] m1 = 4'b0001;  reg [3:0] h0 = 4'b0011;  reg [3:0] h1 = 4'b0001;  always @(posedge clk) //分频模块1Hz  begin  if(count == 50000000)  begin  count <= 0;  sec\_clk <= ~sec\_clk;  end  else  count = count + 1;  end  always @(posedge clk) //分时复用  begin  cnt <= cnt + 1;  end  always @(posedge sec\_clk or negedge rst) //进位模块  if(rst) begin  m0 <= 0; m1 <= 0; h0 <= 0; h1 <= 0;  end  else if(hold) begin end  else begin  if(addm0) begin  if(h1 == 2 & h0 == 3 & m1 == 5 & m0 == 9) begin  h1 <= 0;  h0 <= 0;  m1 <= 0;  m0 <= 0;  end  else if(h0 == 9 & m1 == 5 & m0 == 9) begin  h1 <= h1 + 1;  h0 <= 0;  m1 <= 0;  m0 <= 0;  end  else if(m1 == 5 & m0 == 9) begin  h0 <= h0 + 1;  m1 <= 0;  m0 <= 0;  end  else if(m0 == 9) begin  m1 <= m1 + 1;  m0 <= 0;  end  else  m0 <= m0 + 1;  end  if(addh0) begin  if(h1 == 2 & h0 == 3) begin  h1 <= 0;  h0 <= 1;  end  else if(h0 == 9) begin  h1 <= h1 + 1;  h0 <= 0;  end  else  h0 <= h0 + 1;  end  m0 <= m0 + 1;  if(m0 == 9) begin  m0 <= 0; m1 <= m1 + 1;  end  if(m1 == 5 & m0 == 9) begin  m1 <= 0; h0 <= h0 + 1;  end  if(h1 < 2 & h0 == 9 & m1 == 5 & m0 == 9) begin  h0 <= 0; h1 <= h1 + 1;  end  else if(h1 == 2 & h0 == 3 & m1 == 5 & m0 == 9) begin  h0 <= 0; h1 <= 0;  end  end  always @(posedge clk) begin //选通模块  case(count[15:14])  2'b00: begin dis <= m0; an <= 4'b1110; end  2'b01: begin dis <= m1; an <= 4'b1101; end  2'b10: begin dis <= h0; an <= 4'b1011; end  2'b11: begin dis <= h1; an <= 4'b0111; end  endcase  end  always @(posedge clk) begin //显示模块  case(dis)  0: segs = \_0;  1: segs = \_1;  2: segs = \_2;  3: segs = \_3;  4: segs = \_4;  5: segs = \_5;  6: segs = \_6;  7: segs = \_7;  8: segs = \_8;  9: segs = \_9;  default: segs = 7'b111\_1111;  endcase  end  endmodule | | | | | | |
| 五、实验过程中遇到的问题及解决情况(主要问题及解决情况)  1、 实现十六进制数显示模块,将拨码开关 SW3~SW0 以十六进制数显示在1 个七段数码管上。  在第一次生成bit流时，报错，原因：约束文件管脚没有配置完全；  第一次上板，四个数码管都显示数字，数码管所显示数字成镜像，且数字0和字母c显示出现问题，a和g管脚反接，初步判断管脚分配顺序颠倒；  第二次上板，颠倒管脚分配顺序，讲segs[6]分配到a管脚，删除使能控制，四个数码管都显示数字，数字显示正常  第三次上板，将segs[6:0]改为segs[0:6]，将segs[0]分配到a管脚（数码管的段控制信号的顺序一定要和管脚约束相匹配），新增三个使能控制（对四个an同时控制才能只让一个数码管显示），只有一个数码管显示，显示正常。  2、设计一个 3 位的 4 选 1 多路选择器，并利用拨码开关 SW15~SW14 控制选  择将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 以八进制数显示在  1 个七段数码管上。  问题主要出现在编写源代码时对reg，wire类和部分语法理解不熟悉，经常混用两类或者在always里面用assign赋值，对wire进行逻辑操作导致报错。  3、将拨码开关 SW11~SW9、SW8~SW6、SW5~SW3、SW2~SW0 分别以八进制数“同  时”显示在 4 个七段数码管上。  一开始并不理解分时复用的概念，看了一些不同的分频实现代码才逐渐理解。  5、数字钟  在掌握以上实验的基础上，要注意的是时钟的分频，开发板上的时钟频率为100mhz，为让时钟每一秒跳动一次，即频率为1hz，需让上下边沿之间跳动50000000次。对分时的进位也要注意判断顺序和严谨性。 | | | | | | |
| 1. 实验结果及分析和（或）源程序调试过程 2. 实现十六进制数显示模块,将拨码开关 SW3~SW0 以十六进制数显示在1 个七段数码管上。   965AD82B14889B1A0BC76FF068C592A2AA80113015C6BF9DCF1471D66A140A911210318111982971187708D7D6454E16  显示正常。   1. 数字钟   qq_pic_merged_1636814067689  时钟显示和进位正常，长按1会持续使分加1，长按3会持续时加1，按下2会将时钟重置为0，长按4会使时钟暂停。  七、小组分工情况说明  陈鹏宇：自主完成实验开发，完成实验报告  帕肉合·帕尔哈提：自主完成实验开发 | | | | | | |