**《数字逻辑》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | | 李佳明，明成 | | **年级** | | 2020 |
| **学号** | | 20204254，20204247 | | **专业、班级** | | 计算机科学与技术（卓越）01班 |
| **实验名称** | **实验十四 存储器实验** | | | | | |
| **实验时间** | **2021.12.2** | | **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  掌握随机存储器原理，学会FPGA内部存储器控制器的设计方法。  掌握单端口与双端口RAM（随机存储器）设计与实现。  掌握FIFO（先入先出存储队列）设计与实现。 | | | | | | |
| 二、实验项目内容   1. 利用BASYS3片内存储器单元实现单端口RAM设计（带异步读和同步读两种模式），在时钟（clk）上升沿，采集地址addr）、输入数据data\_in）、执行相关控制信息。当写使能（we）有效，则执行写操作，否则执行读取操作。同步与异步设计仅针对读操作：对于异步RAM而言，读操作为异步，即地址信号有效时，控制器直接读取RAM阵列；对于同步RAM而言，地址信号在时钟上升沿被采集。并保存在寄存器中，然后使用该地址信号读取RAM阵列。 2. 实现双端口（同步与异步）RAM设计，相对于单端口RAM而言，双端口RAM存在两个存取端口，并且可独立进行读写操作，具有自己的地址（addr\_a、addr\_b）、数据输入din\_a、din\_b输出端口（dout\_a、dout\_b以及控制信号。 3. 实现FIFO设计，FIFO由存储单元队列或阵列构成，和RAM不同的是FIFO没有地址，第一个被写入队列的数据也是第一个从队列中读出的数据。 | | | | | | |
| 三、实验设计  **单端口RAM设计原理：**  单端口RAM设计（带异步读和同步读两种模式），在时钟（clk）上沿采集地址（addr）、输入数据data\_in）、执行相关控制信息。当写使能we有效，则执行写操作，否则执行读取操作。同步与异步设计仅针对读操作：对于异步RAM而言，读操作为异步，即地址信号有效时，控制器直接读取RAM阵列；对于同步RAM而言，地址信号在时钟上升沿被采集并保存在寄存器中，然后使用该地址信号读取RAM阵列。单端口RAM框图如右：  （额外添加了读使能oe）  **单端口同步RAM：**  顶层模块：    divider：为七段数码管提供分频后的clk；  Syn\_SinglePortRAM：单端口同步RAM；  transformer：移位加三法，输入至七段数码管进行显示功能的实现；  display7seg：七段数码管显示模块；  schematic：    Syn\_SinglePortRAMsimulation：    **单端口异步RAM：**  顶层模块：    Asy\_SinglePortRAM：单端口异步RAM；  其余模块功能同上；  schematic：    Asy\_SinglePortRAMsimulation：与Syn\_SinglePortRAMsimulation一致  **双端口RAM设计原理：**  双端口（同步与异步）RAM，相对于单端口RAM而言，双端口RAM存在两个存取端口，并且可独立进行读写操作，具有自己的地址（addr\_a、addr\_b）、数据输入（din\_a、din\_b）/输出端口（dout\_a、dout\_b）以及控制信号。双端口RAM常用于视频/图像处理设计中。双端口RAM框图如右：  （额外添加了读使能oe\_a，oe\_b）  **双端口同步RAM：**  顶层模块：    Syn\_DoublePortRAM：双端口同步RAM；  其余模块功能同上  schematic：    **双端口异步RAM：**  顶层模块：    Asy\_DoublePortRAM：双端口同步RAM；  其余模块功能同上  schematic：    **FIFO：**  **设计原理：**  FIFO 是一个先入先出的存储队列，和RAM 不同的是FIFO 没有地址，第一个被写入队列的数据也是第一个从队列中读出的数据。FIFO 可以在输入输出速率不匹配时，作为临时存储单元；可用于不同时钟域中间的同步；输入数据路径和输出数据路径之间数据宽度不匹配时，可用于数据宽度调整电路。FIFO 的框图和信号功能如右：  **顶层模块：**    debkey：消抖模块；  FIFO：实现FIFO；  schematic：    FIFO simulation： | | | | | | |
| 四、实验过程或算法  单端口同步RAM：Project14-1-1  顶层模块：top\_Syn\_SinglePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 23:23:18*  *// Design Name:*  *// Module Name: top\_Syn\_SinglePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module top\_Syn\_SinglePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 4)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr,      input [DATA\_WIDTH-1:0]data\_in,      input we,oe,      output wire[3:0]an,      output wire[6:0]display      );      wire[DATA\_WIDTH-1:0]data\_out;      wire clk\_div;      reg [25:0]target = 50000;      wire [15:0]BCD;  *//divider*      divider d(.clk(clk),.rst(rst),.target(target),.clk\_div(clk\_div));  *//Syn\_SinglePortRAM*      Syn\_SinglePortRAM S(.clk(clk),.rst(rst),.addr(addr),.data\_in(data\_in),.we(we),.oe(oe),.data\_out(data\_out));  *//transformer*      transformer t(.data(data\_out),.BCD(BCD));  *//display7seg*      display7seg dis(.clk(clk\_div),.data3(BCD[15:12]),.data2(BCD[11:8]),.data1(BCD[7:4]),.data0(BCD[3:0]),.an(an),.display(display));  endmodule  divider.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 22:18:46*  *// Design Name:*  *// Module Name: divider*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module divider(      input clk,rst,      input [15:0] target,      output reg clk\_div      );      reg [15:0] counter;      always @(posedge clk) begin          if(rst)          begin              counter <= 0;              clk\_div <= 0;          end          else if(counter==target)          begin              counter <= 0;              clk\_div <= ~clk\_div;          end          else              counter <= counter+1;      end  endmodule  transformer.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 22:25:06*  *// Design Name:*  *// Module Name: transformer*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module transformer(      input [3:0] data,      output [15:0]BCD*//四位，方便输入至数码管*      );  *//移位加3，转换成BCD*      reg [19:0] transfor\_data;      always @(\*)      begin          transfor\_data = 16'b0;          transfor\_data[3:0] = data;          repeat(4)          begin              if(transfor\_data[19:16]>4)                  transfor\_data[19:16] = transfor\_data[19:16]+2'b11;              if(transfor\_data[15:12]>4)                  transfor\_data[15:12] = transfor\_data[15:12]+2'b11;              if(transfor\_data[11:8]>4)                  transfor\_data[11:8] = transfor\_data[11:8]+2'b11;              if(transfor\_data[7:4]>4)                  transfor\_data[7:4] = transfor\_data[7:4]+2'b11;              transfor\_data[19:1] = transfor\_data[18:0];          end      end      assign BCD = transfor\_data[19:4];  endmodule  display7seg.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 23:07:38*  *// Design Name:*  *// Module Name: display7seg*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module display7seg(      input clk,      input [3:0]data3,data2,data1,data0,      output reg[3:0]an,      output reg[6:0]display      );      reg [1:0] count;      always @(posedge clk) begin          if(count == 'b11)              count <= 0;          else              count <= count +'b1;      end      always @(posedge clk) begin          case(count)          2'b00: an <= 4'b1110;          2'b01: an <= 4'b1101;          2'b10: an <= 4'b1011;          2'b11: an <= 4'b0111;          endcase      end      always @(posedge clk) begin          case(count)              2'b00:              case (data0)                  4'b0000:display = 7'b0000001;                  4'b0001:display = 7'b1001111;                  4'b0010:display = 7'b0010010;                  4'b0011:display = 7'b0000110;                  4'b0100:display = 7'b1001100;                  4'b0101:display = 7'b0100100;                  4'b0110:display = 7'b0100000;                  4'b0111:display = 7'b0001111;                  4'b1000:display = 7'b0000000;                  4'b1001:display = 7'b0000100;              endcase              2'b01:              case (data1)                  4'b0000:display = 7'b0000001;                  4'b0001:display = 7'b1001111;                  4'b0010:display = 7'b0010010;                  4'b0011:display = 7'b0000110;                  4'b0100:display = 7'b1001100;                  4'b0101:display = 7'b0100100;                  4'b0110:display = 7'b0100000;                  4'b0111:display = 7'b0001111;                  4'b1000:display = 7'b0000000;                  4'b1001:display = 7'b0000100;              endcase              2'b10:              case (data2)                  4'b0000:display = 7'b0000001;                  4'b0001:display = 7'b1001111;                  4'b0010:display = 7'b0010010;                  4'b0011:display = 7'b0000110;                  4'b0100:display = 7'b1001100;                  4'b0101:display = 7'b0100100;                  4'b0110:display = 7'b0100000;                  4'b0111:display = 7'b0001111;                  4'b1000:display = 7'b0000000;                  4'b1001:display = 7'b0000100;              endcase              2'b11:              case (data3)                  4'b0000:display = 7'b0000001;                  4'b0001:display = 7'b1001111;                  4'b0010:display = 7'b0010010;                  4'b0011:display = 7'b0000110;                  4'b0100:display = 7'b1001100;                  4'b0101:display = 7'b0100100;                  4'b0110:display = 7'b0100000;                  4'b0111:display = 7'b0001111;                  4'b1000:display = 7'b0000000;                  4'b1001:display = 7'b0000100;              endcase          endcase      end  endmodule  Syn\_SinglePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 22:06:52*  *// Design Name:*  *// Module Name: Syn\_SinglePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module Syn\_SinglePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 4)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr,      input [DATA\_WIDTH-1:0]data\_in,      input we,      input oe,      output reg[DATA\_WIDTH-1:0]data\_out      );      reg [DATA\_WIDTH-1:0] RAM[(1<<ADDR\_DEPTH)-1:0];  *//write*      always @(posedge clk)      begin          if(rst)          begin:init\_RAM              integer i;*//必须声明在有名字的块中，或写在外面*              for(i=0;i<(1<<ADDR\_DEPTH);i = i+1)              begin                  RAM[i] <= 0;              end          end          else if(we)          begin              RAM[addr] <= data\_in;          end      end  *//syn\_read*      always @(posedge clk) begin          if(rst)          begin              data\_out <= 0;          end          else if(!we && oe)          begin              data\_out <= RAM[addr];          end          else              data\_out <= 0;      end  endmodule  约束文件：  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports clk]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[4]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[5]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[6]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports rst]  set\_property IOSTANDARD LVCMOS33 [get\_ports we]  set\_property PACKAGE\_PIN W5 [get\_ports clk]  set\_property PACKAGE\_PIN R2 [get\_ports we]  set\_property PACKAGE\_PIN U18 [get\_ports rst]  set\_property PACKAGE\_PIN W7 [get\_ports {display[6]}]  set\_property PACKAGE\_PIN W6 [get\_ports {display[5]}]  set\_property PACKAGE\_PIN U8 [get\_ports {display[4]}]  set\_property PACKAGE\_PIN V8 [get\_ports {display[3]}]  set\_property PACKAGE\_PIN U5 [get\_ports {display[2]}]  set\_property PACKAGE\_PIN V5 [get\_ports {display[1]}]  set\_property PACKAGE\_PIN U7 [get\_ports {display[0]}]  set\_property PACKAGE\_PIN T1 [get\_ports {data\_in[3]}]  set\_property PACKAGE\_PIN U1 [get\_ports {data\_in[2]}]  set\_property PACKAGE\_PIN W2 [get\_ports {data\_in[1]}]  set\_property PACKAGE\_PIN R3 [get\_ports {data\_in[0]}]  set\_property PACKAGE\_PIN W4 [get\_ports {an[3]}]  set\_property PACKAGE\_PIN V4 [get\_ports {an[2]}]  set\_property PACKAGE\_PIN U4 [get\_ports {an[1]}]  set\_property PACKAGE\_PIN U2 [get\_ports {an[0]}]  set\_property PACKAGE\_PIN T2 [get\_ports {addr[3]}]  set\_property PACKAGE\_PIN T3 [get\_ports {addr[2]}]  set\_property PACKAGE\_PIN V2 [get\_ports {addr[1]}]  set\_property PACKAGE\_PIN W13 [get\_ports {addr[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports oe]  set\_property PACKAGE\_PIN V17 [get\_ports oe]  单端口异步RAM：Project14-1-2  顶层模块：top\_Asy\_SinglePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 23:23:18*  *// Design Name:*  *// Module Name: top\_Syn\_SinglePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module top\_Asy\_SinglePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 4)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr,      input [DATA\_WIDTH-1:0]data\_in,      input we,oe,      output wire[3:0]an,      output wire[6:0]display      );      wire[DATA\_WIDTH-1:0]data\_out;      wire clk\_div;      reg [25:0]target = 50000;      wire [15:0]BCD;  *//divider*      divider d(.clk(clk),.rst(rst),.target(target),.clk\_div(clk\_div));  *//Syn\_SinglePortRAM*      Asy\_SinglePortRAM S(.clk(clk),.rst(rst),.addr(addr),.data\_in(data\_in),.we(we),.oe(oe),.data\_out(data\_out));  *//transformer*      transformer t(.data(data\_out),.BCD(BCD));  *//display7seg*      display7seg dis(.clk(clk\_div),.data3(BCD[15:12]),.data2(BCD[11:8]),.data1(BCD[7:4]),.data0(BCD[3:0]),.an(an),.display(display));  endmodule  Asy\_SinglePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 14:24:08*  *// Design Name:*  *// Module Name: SinglePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module Asy\_SinglePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 4)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr,      input [DATA\_WIDTH-1:0]data\_in,      input we,oe,      output reg[DATA\_WIDTH-1:0]data\_out      );      reg [DATA\_WIDTH-1:0] RAM[(1<<ADDR\_DEPTH)-1:0];  *//write*      always @(posedge clk)      begin          if(rst)          begin:init\_RAM              integer i;*//必须声明在有名字的块中，或写在外面*              for(i=0;i<(1<<ADDR\_DEPTH);i = i+1)              begin                  RAM[i] <= 0;              end          end          else if(we)          begin              RAM[addr] <= data\_in;          end      end  *//read*  *//asy*      always @(addr)      begin          if(!we && oe)              data\_out = RAM[addr];          else          begin              data\_out = 0;          end      end  endmodule  其余文件与单端口同步RAM相同  双端口同步RAM：Project14-2-1  顶层模块：top\_Syn\_DoublePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/02 08:45:46*  *// Design Name:*  *// Module Name: top\_Syn\_DoublePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module top\_Syn\_DoublePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 3)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr\_a,addr\_b,      input [DATA\_WIDTH-1:0]din\_a,din\_b,      input we\_a,we\_b,      input oe\_a,oe\_b,      output wire[3:0]an,      output wire[6:0]display,      output wire error*//错误检测灯*      );      wire[DATA\_WIDTH-1:0]dout\_a,dout\_b;      wire clk\_div;      reg [25:0]target = 50000;      wire [15:0]BCD\_a;      wire [15:0]BCD\_b;  *//divider*      divider d(.clk(clk),.rst(rst),.target(target),.clk\_div(clk\_div));  *//Syn\_DoublePortRAM*      Syn\_DoublePortRAM S(.clk(clk),.rst(rst),.addr\_a(addr\_a),.addr\_b(addr\_b),.din\_a(din\_a),.din\_b(din\_b),.we\_a(we\_a),.we\_b(we\_b),.oe\_a(oe\_a),.oe\_b(oe\_b),.dout\_a(dout\_a),.dout\_b(dout\_b),.error(error));  *//transformer*      transformer t\_a(.data(dout\_a),.BCD(BCD\_a));      transformer t\_b(.data(dout\_b),.BCD(BCD\_b));  *//display7seg*      display7seg dis(.clk(clk\_div),.data3(BCD\_a[7:4]),.data2(BCD\_a[3:0]),.data1(BCD\_b[7:4]),.data0(BCD\_b[3:0]),.an(an),.display(display));  endmodule  Syn\_DoublePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/01 22:06:52*  *// Design Name:*  *// Module Name: Syn\_DoublePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module Syn\_DoublePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 3)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr\_a,addr\_b,      input [DATA\_WIDTH-1:0]din\_a,din\_b,      input we\_a,we\_b,      input oe\_a,oe\_b,      output reg[DATA\_WIDTH-1:0]dout\_a,dout\_b,      output reg error*//错误检测*      );      reg [DATA\_WIDTH-1:0] RAM[(1<<ADDR\_DEPTH)-1:0];  *//error*      always @(posedge clk)      begin          if(rst)          begin              error <= 0;          end          else if(!we\_a&&!we\_b&&(addr\_a==addr\_b))*//地址相同时只能进行read*              error <= 0;          else if(addr\_a!=addr\_b)*//地址不同*              error <= 0;          else              error <= 1;*//error指示灯亮*      end  *//write*      integer i;      always @(posedge clk)      begin          if(rst)*//init*          begin              for(i=0;i<(1<<ADDR\_DEPTH);i = i+1)              begin                  RAM[i] <= 0;              end          end          else if(we\_a&&!we\_b&&(addr\_a!=addr\_b))              RAM[addr\_a] = din\_a;          else if(!we\_a&&we\_b&&(addr\_a!=addr\_b))              RAM[addr\_b] = din\_b;          else if(we\_a&&we\_b&&(addr\_a!=addr\_b))          begin              RAM[addr\_a] = din\_a;              RAM[addr\_b] = din\_b;          end      end  *//read*  *//syn\_a*      always @(posedge clk)      begin          if(rst)          begin              dout\_a <= 0;          end          else if(!we\_a && oe\_a)          begin              dout\_a <= RAM[addr\_a];          end          else              dout\_a <= 0;      end      always @(posedge clk)      begin          if(rst)          begin              dout\_b <= 0;          end          else if(!we\_b && oe\_b)          begin              dout\_b <= RAM[addr\_b];          end          else              dout\_b <= 0;      end  endmodule  约束文件：  set\_property IOSTANDARD LVCMOS33 [get\_ports we\_b]  set\_property IOSTANDARD LVCMOS33 [get\_ports we\_a]  set\_property IOSTANDARD LVCMOS33 [get\_ports rst]  set\_property IOSTANDARD LVCMOS33 [get\_ports error]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[6]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[5]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[4]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_b[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_b[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_b[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_b[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_a[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_a[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_a[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {din\_a[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports clk]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr\_b[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr\_b[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr\_a[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr\_a[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr\_a[2]}]  set\_property PACKAGE\_PIN U18 [get\_ports rst]  set\_property PACKAGE\_PIN R2 [get\_ports we\_a]  set\_property PACKAGE\_PIN L1 [get\_ports error]  set\_property PACKAGE\_PIN W7 [get\_ports {display[6]}]  set\_property PACKAGE\_PIN W6 [get\_ports {display[5]}]  set\_property PACKAGE\_PIN U8 [get\_ports {display[4]}]  set\_property PACKAGE\_PIN V8 [get\_ports {display[3]}]  set\_property PACKAGE\_PIN U5 [get\_ports {display[2]}]  set\_property PACKAGE\_PIN V5 [get\_ports {display[1]}]  set\_property PACKAGE\_PIN U7 [get\_ports {display[0]}]  set\_property PACKAGE\_PIN U1 [get\_ports {din\_a[3]}]  set\_property PACKAGE\_PIN W4 [get\_ports {an[3]}]  set\_property PACKAGE\_PIN V4 [get\_ports {an[2]}]  set\_property PACKAGE\_PIN U4 [get\_ports {an[1]}]  set\_property PACKAGE\_PIN U2 [get\_ports {an[0]}]  set\_property PACKAGE\_PIN T3 [get\_ports {din\_b[3]}]  set\_property PACKAGE\_PIN V2 [get\_ports {din\_b[2]}]  set\_property PACKAGE\_PIN W13 [get\_ports {din\_b[1]}]  set\_property PACKAGE\_PIN W14 [get\_ports {din\_b[0]}]  set\_property PACKAGE\_PIN W5 [get\_ports clk]  set\_property PACKAGE\_PIN W16 [get\_ports {addr\_b[2]}]  set\_property PACKAGE\_PIN V15 [get\_ports {addr\_a[2]}]  set\_property PACKAGE\_PIN W15 [get\_ports {addr\_a[1]}]  set\_property PACKAGE\_PIN W17 [get\_ports {addr\_a[0]}]  set\_property PACKAGE\_PIN V17 [get\_ports {addr\_b[0]}]  set\_property PACKAGE\_PIN V16 [get\_ports {addr\_b[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {addr\_b[1]}]  set\_property PACKAGE\_PIN T1 [get\_ports oe\_a]  set\_property PACKAGE\_PIN U1 [get\_ports we\_b]  set\_property PACKAGE\_PIN W2 [get\_ports oe\_b]  set\_property PACKAGE\_PIN R3 [get\_ports {din\_a[2]}]  set\_property PACKAGE\_PIN T2 [get\_ports {din\_a[1]}]  set\_property PACKAGE\_PIN T3 [get\_ports {din\_a[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports oe\_a]  set\_property IOSTANDARD LVCMOS33 [get\_ports oe\_b]  双端口异步RAM：Project14-2-2  顶层模块：top\_Asy\_DoublePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/02 09:08:09*  *// Design Name:*  *// Module Name: top\_Asy\_DoublePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module top\_Asy\_DoublePortRAM#(parameter DATA\_WIDTH = 3,parameter ADDR\_DEPTH = 3)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr\_a,addr\_b,      input [DATA\_WIDTH-1:0]din\_a,din\_b,      input we\_a,we\_b,      input oe\_a,oe\_b,      output wire[3:0]an,      output wire[6:0]display,      output wire error*//错误检测灯*      );      wire[DATA\_WIDTH-1:0]dout\_a,dout\_b;      wire clk\_div;      reg [25:0]target = 50000;      wire [15:0]BCD\_a;      wire [15:0]BCD\_b;  *//divider*      divider d(.clk(clk),                .rst(rst),                .target(target),                .clk\_div(clk\_div));  *//Syn\_DoublePortRAM*      Asy\_DoublePortRAM S(.clk(clk),                          .rst(rst),                          .addr\_a(addr\_a),                          .addr\_b(addr\_b),                          .din\_a(din\_a),                          .din\_b(din\_b),                          .we\_a(we\_a),                          .we\_b(we\_b),                          .oe\_a(oe\_a),                          .oe\_b(oe\_b),                          .dout\_a(dout\_a),                          .dout\_b(dout\_b),                          .error(error));  *//transformer*      transformer t\_a(.data(dout\_a),.BCD(BCD\_a));      transformer t\_b(.data(dout\_b),.BCD(BCD\_b));  *//display7seg*      display7seg dis(.clk(clk\_div),.data3(BCD\_a[7:4]),.data2(BCD\_a[3:0]),.data1(BCD\_b[7:4]),.data0(BCD\_b[3:0]),.an(an),.display(display));  endmodule  Asy\_DoublePortRAM.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/02 09:07:48*  *// Design Name:*  *// Module Name: Asy\_DoublePortRAM*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module Asy\_DoublePortRAM#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 3)(      input clk,rst,      input [ADDR\_DEPTH-1:0]addr\_a,addr\_b,      input [DATA\_WIDTH-1:0]din\_a,din\_b,      input we\_a,we\_b,      input oe\_a,oe\_b,      output reg[DATA\_WIDTH-1:0]dout\_a,dout\_b,      output reg error*//错误检测*      );        reg [DATA\_WIDTH-1:0] RAM[(1<<ADDR\_DEPTH)-1:0];  *//error*      always @(posedge clk)      begin          if(rst)          begin              error <= 0;          end          else if(!we\_a&&!we\_b&&(addr\_a==addr\_b))*//地址相同时只能进行read*              error <= 0;          else if(addr\_a!=addr\_b)*//地址不同*              error <= 0;          else              error <= 1;*//error指示灯亮*      end  *//write*      integer i;      always @(posedge clk)      begin          if(rst)*//init*          begin              for(i=0;i<(1<<ADDR\_DEPTH);i = i+1)              begin                  RAM[i] <= 0;              end          end          else if(we\_a&&!we\_b&&(addr\_a!=addr\_b))              RAM[addr\_a] = din\_a;          else if(!we\_a&&we\_b&&(addr\_a!=addr\_b))              RAM[addr\_b] = din\_b;          else if(we\_a&&we\_b&&(addr\_a!=addr\_b))          begin              RAM[addr\_a] = din\_a;              RAM[addr\_b] = din\_b;          end      end  *//read*  *//asy\_a*      always @(addr\_a)      begin          if (!we\_a && oe\_a)              dout\_a <= RAM[addr\_a];          else              dout\_a <= 0;      end  *//asy\_b*      always @(addr\_b)      begin          if(!we\_b && oe\_b)              dout\_b <= RAM[addr\_b];          else              dout\_b <= 0;      end  endmodule  FIFO：Project14-3  顶层模块：top\_FIFO  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/02 11:59:36*  *// Design Name:*  *// Module Name: top\_FIFO*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module top\_FIFO#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 4)(      input button,clk,rst,wr\_en,rd\_en,      input [DATA\_WIDTH-1:0]data\_in,      output empty,full,      output wire[3:0]an,      output wire[6:0]display      );      wire button\_deb;      wire[DATA\_WIDTH-1:0]data\_out;      wire clk\_div;      reg [25:0]target = 50000;      wire [15:0]BCD;  *//divider*      divider d(.clk(clk),.rst(rst),.target(target),.clk\_div(clk\_div));  *//debkey*      debkey deb(.clk(clk),.rst(rst),.key\_in(button),.key\_out(button\_deb));  *//FIFO*      FIFO F(.clk(button\_deb),.rst(rst),.wr\_en(wr\_en),.rd\_en(rd\_en),.data\_in(data\_in),.empty(empty),.full(full),.data\_out(data\_out));  *//transformer*      transformer t(.data(data\_out),.BCD(BCD));  *//display7seg*      display7seg dis(.clk(clk\_div),.data3(BCD[15:12]),.data2(BCD[11:8]),.data1(BCD[7:4]),.data0(BCD[3:0]),.an(an),.display(display));  endmodule  debkey.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/02 19:23:22*  *// Design Name:*  *// Module Name: debkey*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module debkey(      input clk,      input rst,      input key\_in,      output key\_out      );      parameter T100Hz = 249999;      integer cnt\_100Hz;      reg clk\_100Hz;      always @(posedge clk) begin          if(rst)              cnt\_100Hz<=32'b0;          else          begin              cnt\_100Hz<=cnt\_100Hz+1'b1;              if(cnt\_100Hz==T100Hz)              begin                  cnt\_100Hz<=32'b0;                  clk\_100Hz<=~clk\_100Hz;              end          end      end      reg[2:0]key\_rrr,key\_rr,key\_r;      always @(posedge clk\_100Hz) begin          if(rst)              begin                  key\_rrr<=1'b1;                  key\_rr<=1'b1;                  key\_r<=1'b1;              end          else          begin              key\_rrr<=key\_rr;              key\_rr<=key\_r;              key\_r<=key\_in;          end      end      assign key\_out = key\_rrr&key\_rr&key\_r;  endmodule  FIFO.v  `timescale 1ns / 1ps  *//////////////////////////////////////////////////////////////////////////////////*  *// Company:*  *// Engineer:*  *//*  *// Create Date: 2021/12/02 09:16:16*  *// Design Name:*  *// Module Name: FIFO*  *// Project Name:*  *// Target Devices:*  *// Tool Versions:*  *// Description:*  *//*  *// Dependencies:*  *//*  *// Revision:*  *// Revision 0.01 - File Created*  *// Additional Comments:*  *//*  *//////////////////////////////////////////////////////////////////////////////////*  module FIFO#(parameter DATA\_WIDTH = 4,parameter ADDR\_DEPTH = 4)(      input clk,rst,wr\_en,rd\_en,      input [DATA\_WIDTH-1:0] data\_in,      output reg empty,full,      output reg[DATA\_WIDTH-1:0] data\_out      );      reg [DATA\_WIDTH-1:0] FIFO[(1<<ADDR\_DEPTH) - 1:0];      reg [ADDR\_DEPTH-1:0]head;      reg [ADDR\_DEPTH-1:0]rear;      reg [ADDR\_DEPTH:0]NUM;  *//empty*      always @(\*) begin          if(NUM==0)              empty<=1;          else              empty<=0;      end  *//full*      always @(\*) begin          if(NUM==(1<<ADDR\_DEPTH))              full<=1;          else              full<=0;      end  *//NUM*      always @(posedge clk or posedge rst) begin          if(rst)              NUM<=0;          else if(!wr\_en&&!rd\_en)*//no write no read*              NUM<=NUM;          else if(wr\_en&&!rd\_en&&(NUM<(1<<ADDR\_DEPTH)))*//wirte no read*              NUM<=NUM+'b1;          else if(!wr\_en&&rd\_en&&(NUM>0))*//read no write*              NUM<=NUM-'b1;          else if(wr\_en&&rd\_en)              NUM<=NUM;      end  *//write*      integer i;      always @(posedge clk or posedge rst) begin          if(rst)          begin              rear<='b0;              for(i=0;i<(1<<ADDR\_DEPTH);i=i+1)                  FIFO[i]<=0;          end          else if(wr\_en&&(NUM<(1<<ADDR\_DEPTH)))*//not full*          begin              FIFO[rear]<=data\_in;              rear<=(rear+1)%(1<<ADDR\_DEPTH);          end          else if(wr\_en&&(NUM>(1<<ADDR\_DEPTH)-1))*//full*              rear<=rear;      end  *//read*      always @(posedge clk or posedge rst) begin          if(rst)          begin              head<='b0;              data\_out<=0;          end            else if(rd\_en&&(NUM!=0))          begin              data\_out<=FIFO[head];              head<=(head+1)%(1<<ADDR\_DEPTH);          end          else if(NUM==0)              data\_out<=0;      end  endmodule  约束文件：  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets button\_IBUF]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {an[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports button]  set\_property IOSTANDARD LVCMOS33 [get\_ports clk]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {data\_in[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[0]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[1]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[2]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[3]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[4]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[5]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports {display[6]}]  set\_property IOSTANDARD LVCMOS33 [get\_ports full]  set\_property IOSTANDARD LVCMOS33 [get\_ports rd\_en]  set\_property IOSTANDARD LVCMOS33 [get\_ports rst]  set\_property IOSTANDARD LVCMOS33 [get\_ports wr\_en]  set\_property IOSTANDARD LVCMOS33 [get\_ports empty]  set\_property PACKAGE\_PIN R2 [get\_ports wr\_en]  set\_property PACKAGE\_PIN U18 [get\_ports rst]  set\_property PACKAGE\_PIN T1 [get\_ports rd\_en]  set\_property PACKAGE\_PIN L1 [get\_ports full]  set\_property PACKAGE\_PIN P1 [get\_ports empty]  set\_property PACKAGE\_PIN W7 [get\_ports {display[6]}]  set\_property PACKAGE\_PIN W6 [get\_ports {display[5]}]  set\_property PACKAGE\_PIN U8 [get\_ports {display[4]}]  set\_property PACKAGE\_PIN V8 [get\_ports {display[3]}]  set\_property PACKAGE\_PIN U5 [get\_ports {display[2]}]  set\_property PACKAGE\_PIN V5 [get\_ports {display[1]}]  set\_property PACKAGE\_PIN U7 [get\_ports {display[0]}]  set\_property PACKAGE\_PIN U1 [get\_ports {data\_in[3]}]  set\_property PACKAGE\_PIN W2 [get\_ports {data\_in[2]}]  set\_property PACKAGE\_PIN R3 [get\_ports {data\_in[1]}]  set\_property PACKAGE\_PIN T2 [get\_ports {data\_in[0]}]  set\_property PACKAGE\_PIN W5 [get\_ports clk]  set\_property PACKAGE\_PIN V17 [get\_ports button]  set\_property PACKAGE\_PIN W4 [get\_ports {an[3]}]  set\_property PACKAGE\_PIN V4 [get\_ports {an[2]}]  set\_property PACKAGE\_PIN U4 [get\_ports {an[1]}]  set\_property PACKAGE\_PIN U2 [get\_ports {an[0]}] | | | | | | |
| 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) | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  七、小组分工情况说明  李佳明：  明成： | | | | | | |