**《数字逻辑》实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** | **王映淞** | | **年级** | | **2020级** |
| **学号** | **20204202** | | **专业、班级** | | **计算机科学与技术 卓越02** |
| **实验名称** | **实验三 多路选择器** | | | | |
| **实验时间** | **2021.10.23** | **实验地点** | | **DS1404** | |
| **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.掌握门电路的设计开发。  2.学习掌握IP核的封装与调用。  3. 调用封装的IP核，用Block Design设计一个三位的4选1多路选择器。  4.熟悉仿真文件的写法，并熟悉将程序写入开发板的流程。 | | | | | |
| 二、实验项目内容  设计一个三位4选1多路选择器电路，分别用Block Design和verilog HDL语言编写代码两种方式实现，通过仿真、看RTL电路图、下载到板子验证其正确性，并比较不同方法实现的异同（逻辑资源、RTL电路图、仿真波形等方面）。 | | | | | |
| 三、实验设计    实验原理图。  实验原理是，利用四个与门和一个或门进行综合，用两个输入信号和两个非门来控制输入的情况。  但是考虑到我们需要做出三位数据的多路选择器，所以我们需要控制w信号和f信号为3位。  根据    逻辑函数，w若为3为，那么为了完成正确的位运算，s必须也为3位。考虑到开发板案件数量。我们仍然输入1位的s0和s1,但是在输入之后利用译码器拓展位宽，实现逻辑功能即可。  真值表如图： | | | | | |
| 四、实验过程或算法  Block design语言编写：首先实现门电路的设计与IP封装，注意实现门电路的位数即可。然后编写转换器，将1位拓展为3位，只需1到111，0到000就可以。  然后建立block design项目，导入并调用IP核，画出电路图。    进行RTL分析，观察结果与自己所连电路的差异。    检查无误后，编写仿真文件，看是否与真值表相对应。  module mux4x1\_3\_sim();  reg s0=0,s1=0;  reg [2:0]w0=3'b111;  reg [2:0]w1=3'b000;  reg [2:0]w2=3'b000;  reg [2:0]w3=3'b000;  wire [2:0]f;  mux4x1\_3 u(.s0(s0),.s1(s1),.w0(w0),.w1(w1),.w2(w2),.w3(w3),.f(f));  initial begin  #10 s0=1;s1=1;w0=3'b000;w1=3'b000;w2=3'b000;w3=3'b111;  #10 s0=1;s1=0;w0=3'b000;w1=3'b101;w2=3'b000;w3=3'b000;  #10 s0=0;s1=1;w0=3'b000;w1=3'b000;w2=3'b110;w3=3'b000;  #10 s0=1;s1=1;w0=3'b001;w1=3'b010;w2=3'b100;w3=3'b000;  end  endmodule    以下为仿真图：    之后连接开发板，并综合程序原文件，输入引脚并产生约束文件。    确认后，保存位xdc文件。之后生成比特流。接着打开硬件程序并打开一个新的硬件目标。然后向开发板中编入文件即可。在开发板上验证自己的逻辑与程序是否正确。    E6D4807C889122B48F3BF7803EAC8321  输入w0=001,w1=110,w2=011,w3=100  当s1=0,s0=0时，选择的是w0，w0=001，所以f输出001  F03DD0118FB96C3470CAD2918F04BB26  s1=0,s1=1,f=w1=110  41FE62FAB11B156A6ED7D2187113DBA6  s1=1,s0=0,f=w2=011  26785836BA35EAC0C57D865214131753  s1=1,s0=1,f=w3=100  81F6F69ADC3C63F63B66C732BC5B0A49  经检验，与真值表和仿真生成图相符。  Verilog语言程序设计：  利用了case语句实现，相比于block design设计，此处添加了一个rst信号来复位0。代码如下：  module mux4\_1(f, rst, w0, w1, w2, w3, s0,s1);  output [2:0]f;  input s0,s1;  input rst;  input [2:0]w0;  input [2:0]w1;  input [2:0]w2;  input [2:0]w3;  reg [2:0]f;  always @ (w0, w1, w2, w3, s0,s1, rst) begin  if(rst ==1 )  f = 3'b000;  else begin  case({s0,s1})  2'b00: f = w0;  2'b01: f = w2;  2'b10: f = w1;  2'b11: f = w3;  endcase  end  end  endmodule  接下来和block design基本一致，RTL分析：    激励仿真：  module mux4\_1\_sim();  reg s0=0,s1=0;  reg rst=0;  reg [2:0]w0=3'b100;  reg [2:0]w1=3'b010;  reg [2:0]w2=3'b001;  reg [2:0]w3=3'b000;  wire [2:0]f;  mux4\_1 u(.s0(s0),.rst(rst),.s1(s1),.w0(w0),.w1(w1),.w2(w2),.w3(w3),.f(f));  initial begin  #10 s0=1;s1=0;rst=0;w0=3'b111;w1=3'b000;w2=3'b000;w3=3'b111;  #10 s0=0;s1=1;rst=0;w0=3'b111;w1=3'b000;w2=3'b000;w3=3'b111;  #10 s0=1;s1=1;rst=0;w0=3'b000;w1=3'b000;w2=3'b000;w3=3'b111;  #10 s0=1;s1=1;rst=0;w0=3'b000;w1=3'b000;w2=3'b000;w3=3'b111;  #10 s0=1;s1=0;rst=1;w0=3'b000;w1=3'b101;w2=3'b000;w3=3'b000;  #10 s0=0;s1=1;rst=0;w0=3'b000;w1=3'b000;w2=3'b110;w3=3'b000;  #10 s0=1;s1=1;rst=0;w0=3'b001;w1=3'b010;w2=3'b100;w3=3'b000;  end  endmodule    综合源文件，设置引脚，生成约束文件保存为xdc。打开硬件目标，并向开发板中编程。在开发板上验证正确性。按键位置与block design一致，只不过在最右面加了rst信号。 | | | | | |
| 五、实验过程中遇到的问题及解决情况  封装IP核时，没有考虑到输入输出的信号个数与信号位数，所以重新写了一次。粗心的毛病以后一定注意。  还有就是激励文件的编写及测试时出现输出zzz和xxx现象，但是程序写到开发板上是正确的。经过仔细地分析，发现是激励文件地编写除了差错，在更改后仿真正确 | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  Block design设计：仿真测试和开发板验证      Verilog 设计：仿真测试和开发板验证      经过电路分析级及真值表情况，判断知结果正确。  七、小组分工情况说明  魏永森：完成门电路IP核的封装。完成Verilog代码。以及编写两程序的激励文件，并进行仿真验证。  张鑫：完成IP核的调用及block design的图形构建。实现综合电路与引脚设置，以及开发板的写入验证。 | | | | | |