**《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** | **实验三 4选1多路选择器** | | | | | |
| **实验时间** | **2021.10.30** | | **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  通过实验，学会多路选择器的功能，以及使用 Block Design 和 verilog  HDL 语言设计多路选择器。 | | | | | | |
| 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  Verilog HDL语言编写代码实现：  电路图  波形图    真值表   |  |  |  |  | | --- | --- | --- | --- | | In | Cntrl1 | Cntrl2 | out | | In1 | 0 | 0 | In1 | | In2 | 0 | 1 | In2 | | In3 | 1 | 0 | In3 | | In4 | 1 | 1 | In4 |   Block design：  将Verilog语言设计的代码块封装成ip核如下    完成接线后： | | | | | | |
| 四、实验过程或算法  Verilog HDL实现：  源代码：  module selecter4\_1(out, in1, in2, in3, in4, cntrl1, cntrl2);  parameter size = 3;  output [size-1:0] out; //三位输出  input [size-1:0] in1, in2, in3, in4; //三位值输入  input cntrl1, cntrl2;  reg [size-1:0] out;  always @(\*) //任一输入发生变化执行  case({cntrl1 , cntrl2}) //cntrl的排列  2'b00:out = in1;  2'b01:out = in2;  2'b10:out = in3;  2'b11:out = in4;  endcase  endmodule  仿真代码：  module sim\_selecter;  parameter size = 3;  reg[size-1:0] in1 = 3'b001; //此仿真文件先认定被选择值为常量不变  reg[size-1:0] in2 = 3'b011;  reg[size-1:0] in3 = 3'b101;  reg[size-1:0] in4 = 3'b111;  reg cntrl1; reg cntrl2;  wire[size-1:0] out;  selecter4\_1 uut( //实例化模块  .in1(in1),  .in2(in2),  .in3(in3),  .in4(in4),  .cntrl1(cntrl1),  .cntrl2(cntrl2),  .out(out)  );  initial //只执行一次  begin  cntrl1 = 0; cntrl2 = 0; //分别置到四种状态，观察输出结果，即可验证  #100  cntrl1 = 0; cntrl2 = 1;  #100  cntrl1 = 1; cntrl2 = 0;  #100  cntrl1 = 1; cntrl2 = 1;  end  endmodule  Block Design实现：  首先将HDL代码封装成ip核，在另一个项目中调用该ip核（封装过程文档有介绍）；  创建bd文件，添加ip并连线；  Generate the Output Products；  Create a HDL wrapper；  Add Constraints；  Generate Bitstream； | | | | | | |
| 五、实验过程中遇到的问题及解决情况  Verilog HDL：  最开始时代码为always @(in1 or in2 or in3 or in4)    导致无论怎么变化cntrl，最终输出都是一样的，之后改成always @(\*)，  只要任意输入变化就会执行，才和理论结果相同。(上图中cntrl为11，理论上应该输出in4为7，但实际输出为in1 1)  block design：  如果直接用HDL中的变量名字，会发现‘out’与官方关键字冲突，需相应修改约束文件中的变量名，否则会无法生成bit流。 | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程        *实验结果*:  输入in1-4分别置为100，010，001，110;  当cntrl为00，01，10，11时，对应led灯分别开启，与理论值和仿真  结果一样。  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 | | | | | | |