**《数字逻辑》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | | *孙莹莹* | | **年级** | | *2020级* |
| **学号** | | *20204104* | | **专业、班级** | | *计算机科学与技术卓越一班* |
| **实验名称** | **实验十五 摩尔状态机序列检测器** | | | | | |
| **实验时间** | **2021.12.09** | | **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  通过实验，掌握有限状态机的设计方法，并用摩尔状态机设计实现“1101”序列检测。 | | | | | | |
| 二、实验项目内容   1. 设计“1101”序列检测的状态转换图; 2. 设计一个 8 位并转串输出模块 par2ser。该器件有 8 位输入 d[7:0]，1 位输出 q，另有一个 clk 端，一个 set 端。set端上升沿将 8位输入锁存到逻辑右移移位寄存器中。 3. 调用并转串输出模块，使用Verilog HDL语言的行为描述方式实现一个摩尔状态机，能检测一个8位的二进制数据中是否存在“1101”序列，如果检测到该序列则指定的LED灯亮； 4. 综合、实现、生成bit流，下载到Basys3开发板进行验证； | | | | | | |
| 1. 实验设计 2. 实验原理   8 位并转串输出模块 par2ser:  该模块实现并行载入、串行输出寄存器的功能。set端上升沿载入数据，在每个时钟上升沿移位，最低位输出，实现串行输出的功能。  Moore型状态机检测“1101”：  并转串模块先工作，载入输入的数据；然后在每一个clk上升沿对序列进行移位操作，并依次检测最低位，根据最低位数字发生状态转移，直到检测到了序列“1101”，则light（输出变量）为1，开发板上的灯亮；否则light为0，灯不亮。   1. 检测序列“1101”的状态表（真值表）和状态图 2. 状态图   91F43D24E5CEB988E40C2D7FE18227BB  （2）根据状态图列出状态表  979F68F5539FE402AA047D65765A75F5   1. 模块功能图   AF45C4CABC5E180DE260B5B2FDA8E1C8   1. 所用器件   RTL Memory,RTL Register,RTL Multiplexer | | | | | | |
| 四、实验过程或算法(关键步骤、核心代码注解等）  （1）par2ser模块核心代码及注解如下  module parser#(parameter size=8)(data,clk,set,out);  input [size-1:0] data; //载入的数据  input clk;  input set;  output reg out;  //串行输出最低位  reg [size-1:0] load; //临时寄存器存储载入的值  always@(posedge clk)  begin  if (set)  load <= data;  else  load <= {1'b0,load[7:1]};  end  always@(\*)  begin  out <= load[0];  end  endmodule  （2）Moore型“1101”序列检测器核心代码和注解如下  module Moore\_1101(clk,set,rst\_n,data,light);  input clk;  input set;  input rst\_n;  input [7:0] data;  output reg light;    wire z;  parser tryone(.data(data),.clk(clk),.set(set),.out(z));  //定义五个状态  parameter s0 = 5'b00001;  parameter s1 = 5'b00010;  parameter s2 = 5'b00100;  parameter s3 = 5'b01000;  parameter s4 = 5'b10000;    reg [4:0] currState;  reg [4:0] nextState;    always@(posedge clk or negedge rst\_n)  begin  if(!rst\_n)  currState <= s0;  else  currState <= nextState;  end    //状态转移  always@(\*)  begin  if(!rst\_n)  nextState <= s0;  else  begin  case(currState)  s0: nextState<=(z==1)?s1:s0;  s1: nextState<=(z==1)?s2:s0;  s2: nextState<=(z==1)?s2:s3;  s3: nextState<=(z==1)?s4:s0;  s4: nextState<=(z==1)?s2:s0;  default:nextState <= s0;  endcase  end  end    //亮灯模块  always@(\*)  begin  if(!rst\_n)  light <= 0;  else if(currState==s4)  light <= 1;  end  endmodule  （5）RTL电路如下 | | | | | | |
| 五、实验过程中遇到的问题及解决情况  1.状态机的三段式写法中，时序逻辑和组合逻辑没有分清，导致仿真结果不正确 – 将输出赋值由时序逻辑改为组合逻辑，仿真结果正确。 | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  1.仿真文件如下  `define clk\_period 20  module try();  reg clk,set,rst\_n;  reg [7:0] data;  wire light;  Moore\_1101 ugg(clk,set,rst\_n,data,light);  always #(`clk\_period/2) clk = ~clk;  initial  begin  clk = 0;  set = 0;  rst\_n = 0;  data = 8'b0000\_0000;  #(`clk\_period\*2)  rst\_n = 1;  data = 8'b1011\_1011;  #(`clk\_period\*2)  set = 1;  #(`clk\_period\*2)  set = 0;  #(`clk\_period\*10)  rst\_n = 0;  #(`clk\_period\*2)  rst\_n = 1;  #(`clk\_period\*2)  data = 8'b0011\_0110;  set = 1;  #(`clk\_period\*2)  set = 0;  #(`clk\_period\*10)  $stop;  end  endmodule  2.仿真结果如下    由仿真结果可知，序列中存在“1101”可以成功被检测出来；序列中没有“1101”则灯不亮。  3.资源利用如下     1. 上板测试 2. 输入1101\_1101，指示灯亮   C:/Users/LUY/AppData/Local/Temp/picturecompress_20211125100350/output_1.jpgoutput_1   1. 输入1111\_1111，指示灯不亮   C:/Users/LUY/AppData/Local/Temp/picturecompress_20211125100442/output_1.jpgoutput_1  （3）1100\_1011 灯不亮      (4)0011\_0110 灯亮    七、小组分工情况说明  *孙莹莹：状态机的设计，状态图的绘制，Moore状态机的设计,仿真文件的编写，实验报告的编写，上板测试。*  *刘淑文：par2ser并转串的设计与代码实现。* | | | | | | |