数字电路与逻辑设计实验报告

猜拳游戏的设计与实现

目录

[**摘要 3**](#_bookmark0)

[**一、任务要求 4**](#_bookmark1)

[（一）基本要求 4](#_bookmark2)

[（二）提高要求 5](#_bookmark3)

[**二、系统设计 5**](#_bookmark4)

[（一）设计思路 5](#_bookmark5)

（二）总体框图 6

（三）分块设计 7

1. 状态机 7
2. 点阵显示 8
3. 倒计时 9
4. 分频器模块 10
5. 音乐播放模块 11
6. 消抖模块 12

**三、仿真波形及波形分析 13**

（一）状态机模块 14

（二）倒计时模块 14

（三）分频模块 15

（四）消抖模块 15

**四、源程序 16**

**五、功能说明及资源利用情况 16**

1. 功能说明 16
2. 资源利用情况 16
3. 管脚分配情况 17

**六、故障及问题分析 18**

1. 倒计时模块不能重复使用 18
2. 蜂鸣器没有声音 18
3. 编译错误 18

**七、总结与结论 19**

**参考文献 20**

**附件(源程序) 21**

摘要

本次实验课题是用硬件描述语言 Verilog 来设计实现猜拳游戏，通过不同的按键控制，选择多种出拳方式，显示猜拳的结果，实现猜拳游戏，防止了作弊的可能。通过按键，可实现猜拳、防作弊、计分统计共三个操作。通过拨码开关，可实现复位、清零、播放音乐、显示动画四个附加功能。通过LCD液晶屏显示猜拳比分。本次模块划分方法，将整个游戏机系统分为八个模块：状态控制模块、分频模块、消抖模块、点阵显示模块、计分模块、数码管显示模块、蜂鸣器模块和LCD显示模块。最后通过对程序的调试、相应部分功能的仿真以及在电路板的烧写，验证了整个系统的原理和本设计方案的正确性。

**Abstract**

The topic of this lab is to implement a rock-paper-scissors game on an FPGA board using Verilog HDL. By pressing different buttons, the user can choose among rock, paper and scissors, and the scores will be displayed by an LCD screen and segments, together with the winning side to be displayed by arrays. In addition, with the judge controlling the display of the choices of both sides, the rock-paper-scissors game can be played between two players without cheating. The program consists of three basic functions: the scoreboard, the anti-cheating mechanism, and the game itself. Using the switches on the FPGA board, the user can reset the program, play some music or show some animation, which are some extra functions. This program is divided into eight modules: the state controller module, the clock divider module, the button debouncing module, the array display module, the scoreboard module, the segment display module, the beep module and the LCD display module. This program was simulated with a computer and tested on an FPGA board, and proves to be correct and usable.

关键词: 猜拳、Verilog 语言、QuartusII 、蜂鸣器、LCD1602

## 一、任务要求

利用 EPM1270T144C5N 实验开发板，设计并实现一个猜拳游戏机。

（一）基本要求

1. 甲乙双方各用三个按键模拟“石头”、“剪刀”、“布”，BTN7、BTN6、BTN5为甲，BTN2、BTN1、BTN0为乙。
2. 裁判用BTN3表示“准备”，BTN4表示“开”。
3. 每局比赛开始前裁判先宣布“准备”，点阵显示全灭，然后甲乙双方分别选择出拳方式（以最后一次选择为准）；
4. 裁判按“开”以后，用点阵的左右三列同时显示甲乙双方的猜拳选择（如下图所示），并用两个数码管显示甲乙的猜拳比分。

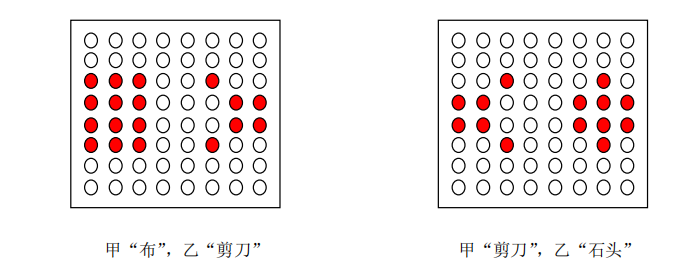


图1 点阵左右三列同时显示双方猜拳

1. 猜拳游戏为五局三胜制。若甲乙双方出拳一致，则比分保持不变，双方重新出拳。
2. 比赛结束后，用8\*8点阵显示获胜方。
3. 用拨码开关作为复位键，复位后游戏重新开始。

（二）提高要求

1. 点阵显示增加游戏开机动画、结束动画；
2. 为游戏增加音效；
3. 在LCD1602液晶屏上显示甲乙双方的猜拳比分；
4. 自拟其他功能。

## 二、系统设计

（一）设计思路

首先对整个实验进行思路梳理，功能核心为点阵显示猜拳选择，主要控制方式为按键控制。所以为实现按键控制出拳，将模块主要分为：状态控制模块，点阵显示模块，计分模块，蜂鸣器模块，分频模块，消抖模块，数码管显示模块，LCD显示模块，动画模块。

接下来考虑显示分数与胜负问题：将分数分为计分与显示两部分进行思考，利用数码管与点阵都是通过人眼暂留原理进行显示，编写数码管显示模块与LCD显示模块。根据逻辑思路编写计分模块。

最后，在完成以上功能后，再使用BTN3、BTN4实现裁判功能，拨码开关添加音频模块与复位清零功能。根据状态之间的逻辑关系进行编写。

1. .状态控制模块(状态机)：根据出拳情况将点阵图案显示分有（石头石头/石头剪刀/石头布/剪刀剪刀/剪刀石头/剪刀布/布布/布剪刀/布石头）九种情况。每个人出拳情况有（石头/剪刀/布）三种情况。故可用一个三种状态的状态机来描述出拳的工作状态，输出为一个3bit型的二进制数（001/010/100），分别表示三种对应情况。因为输出信号不仅与当前状态有关，且直接受BTN输入影响，因此为 Mealy 型状态机。
2. .点阵显示模块：当闪烁显示的发光二极管闪烁频率较高时，我们将观察到持续点亮状态。同理，当点阵的每一行依次显示，当切换速度足够快时，我们将观察到点阵的每一行都是在同时显示。点阵的某一行要稳定显示要求显示频率大于 50Hz，那么 8 行需要 400Hz 以上的频率才能看到稳定点亮的现象。故使用500Hz 的时钟对点阵进逐行扫描，得到稳定的点阵显示状态。通过裁判的“准备”“开始”控制按键进行选择输出点阵图案，再通过状态机输出的甲乙二人出拳状态，选择显示输出的点阵图案。
3. .计分模块：输入出拳状态，对甲乙9种出拳状态进行一一列些，每种情况对应甲加一分/乙加一分/不加分。最后分别输出甲乙的分数。
4. .蜂鸣器模块：蜂鸣器的输出为背景音效。背景音乐模块，可用50MHz时钟信号作为蜂鸣器的输入信号。此外，我又设计实现了提示音效，实现了裁判提示时间即将结束的提示音乐。此模块利用了两个时钟信号，其中一个时钟信号来完成节拍(某一音调的持续时间)、另一个时钟信号来控制音调。
5. .分频模块：整个系统需要 1Hz、100Hz、1000Hz、500Hz 和蜂鸣器部分需要的多种频率的时钟信号，可由板子上的 50MHz 进行分频得到。

应该注意的是，由于按键为机械弹性开关，为了保证按键识别的准确性，上述所有的按钮需要进行按键消抖处理。这里利用 100Hz 的时钟信号进行按键消抖处理。

1. .消抖模块：因为实验中使用BTN0到BTN7共8个按键进行输入，存在按键抖动，会造成误识别，所以必须进行去抖动处理，消除抖动部分的信号。
2. .数码管显示模块：数码管动态显示原理和点阵显示相似，利用共阴极选通信号，某一个数码管要稳定显示要求显示频率大于 50Hz，所以模块频率设置为1000Hz。将从计分模块输出得到的甲乙得分输入，进行显示。
3. .LCD显示模块：利用LCD1602液晶显示原理，将分数输入并转化为相应的ASCII码，从而在液晶屏上进行输出。
4. .动画模块：原理与点阵显示模块相同。即利用 500Hz 的时钟信号对点阵进行扫描，利用 1Hz 的时钟信号切换点阵显示图案，从而达到动画的目的。
5. 总体框图
6. 分块设计
7. 点阵显示

点阵能稳定显示某一图形是利用了人眼的视觉暂留现象。点阵的扫描方式主

要有三种：按行扫描、按列扫描、按点扫描，这里选用了按行扫描。对于每一个

显示状态，利用 500Hz 的时钟信号对点阵进行逐行扫描，可以得到稳定的点阵显

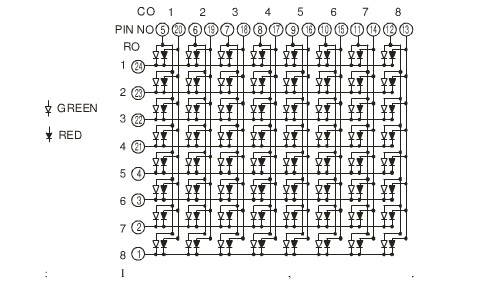
示画面。为区分甲乙，使用双色点阵进行显示点阵图案：甲为绿色，乙为红色。实验板上的行列坐标，左下角为(0,0) ，右上角为(7,7) 双色点阵结构图如图3.2所示。

图3.2双色点阵结构图

点亮点阵上某一点的条件是对应的 row 管脚为低电平，col 管脚为高电平，故可用如下 Verilog 语言对点阵进行逐行扫描：

1. always @(posedge clk)
2. begin
3. **if**(count>=14'd50000)
4. begin
5. count<=14'b0;
6. judge = judge+3'b001;
7. end
8. **else**
9. count<=count+1'b1;
10. end

不同的显示状态对应的画面不同、不同行对应列电平也不同，故采用 case

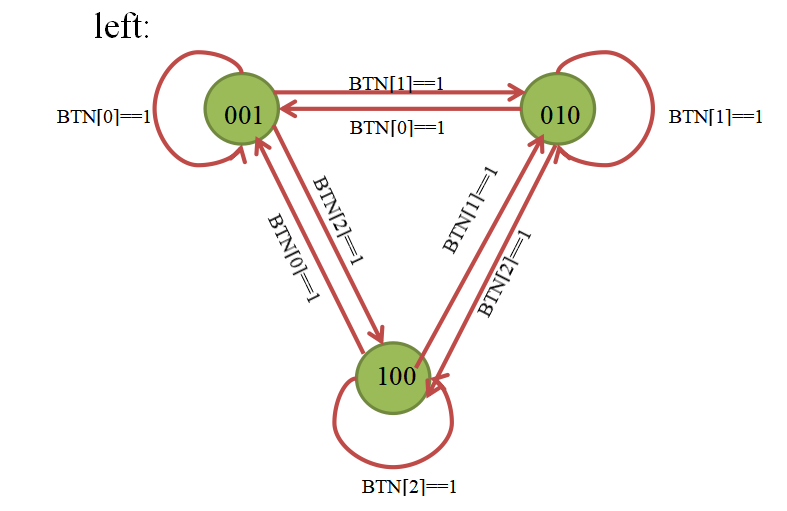
语句的嵌套，外层 case 语句控制不同的 picture，里层 case 语句控制不同的 row。 row为0触发 col为1触发 从右往左读。

对应的 Verilog 语言描述如下：

1. begin
2. **case**({new\_button\_left,new\_button\_right})
3. 6'b001001:begin//石头石头
4. …………
5. end
6. 6'b010001:begin//剪刀石头
7. …………
8. end
9. 6'b100001:begin//布石头
10. …………
11. end
12. 6'b001010:begin//石头剪刀
13. …………
14. end
15. 6'b010010:begin//剪刀剪刀
16. …………
17. end
18. 6'b100010:begin//布剪刀
19. …………
20. end
21. 6'b001100:begin//石头布
22. …………
23. end
24. 6'b010100:begin//剪刀布
25. …………
26. end
27. 6'b100100:begin//布布
28. …………
29. end
30. 状态机

在设计思路中已经分析过，在这里需要创建一个Mealy型有限状态机。设置

三个状态 001、010、100，分别代表石头、剪刀、布。输入信号为BTN[0]-BTN[2]（乙）,BTN[5]-BTN[7]（甲）。输出信号为left(甲的出拳状态)，right（乙的出拳状态），输入的时钟信号为系统时钟信号。



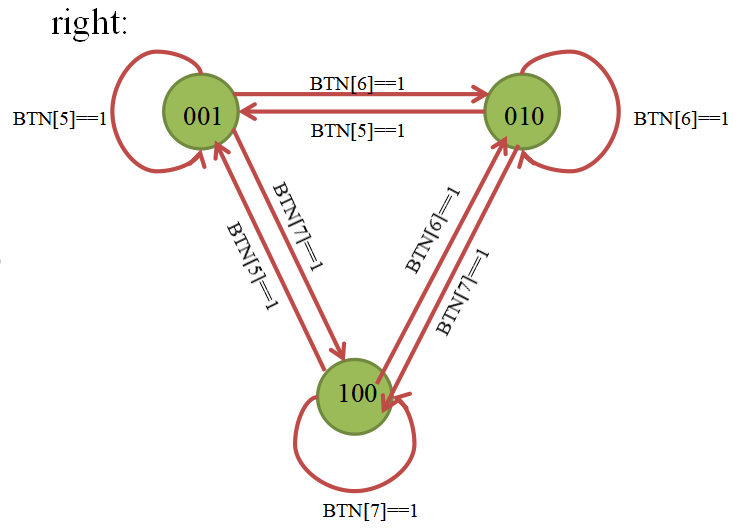


图3.1 状态转移图

由图3.1知，left与right的状态转移方式为：

1. left：遇到BTN[0]变为或保持001，遇到BTN[1]变为或保持010，遇到BTN[2]变为或保持100；
2. right：遇到BTN[5]变为或保持001，遇到BTN[6]变为或保持010，遇到BTN[7]变为或保持100；

对应的Verilog语言表示如下：

1. reg[2:0] left;
2. always @ (posedge clk)
3. begin     //key\_pulse\_left[n]为BTN[n]消抖后的状态
4. **if**(key\_pulse\_left[0]==1)
5. left <= 3'b001;
6. **else** **if**(key\_pulse\_left[1]==1)
7. left <= 3'b010;
8. **else** **if**(key\_pulse\_left[2]==1)
9. left <= 3'b100;
10. end
12. reg[2:0] right;
13. always @ (posedge clk)
14. begin    //key\_pulse\_right[n]为BTN[n]消抖后的状态
15. **if**(key\_pulse\_right[0]==1)
16. right <= 3'b001;
17. **else** **if** (key\_pulse\_right[1]==1)
18. right <= 3'b010;
19. **else** **if** (key\_pulse\_right[2]==1)
20. right <= 3'b100;
22. end

3.计分模块

计分模块在获得消抖后的出拳状态，进行逻辑运算获得得分，并将甲乙的得分输出给数码管显示模块与LCD显示模块。

整个模块需要注意的问题有：

（1）当一局得分已经出现3分及以上的一方，游戏应该终止，数值清零，所以应设置清零键。

（2）按键需要消抖

4.分数/数码管显示

数码管初始状态为第1个数码管与第7个数码管为0，出拳完成后，按下BTN4 ，显示得分。当拨码开关sw2与sw6同为1时，按下BTN4，得分清零。

整个模块需要注意的问题有：

(1)按键需要消抖。

(2)BTN4必须在BTN3（准备）按下后开始按，否则无效。

(3)当一局已经出现3分获胜者时，应设置积分清零键。

按键消抖原理将在后续部分介绍，这里首先围绕（2）、（3）进行设计与实现。

clkout 为分频器分到的1000Hz时钟，grade\_left和grade\_right为输入信号，place 为计数信号，DIG\_r、Y\_r 为数码管的输出信号。

当按下BTN4 时，start='1'；当回到空档(包括倒计时结束、按下立即关闭键)

时，start<='0'，确保计数停止。

当start='1'时，利用1Hz 的时钟计数，遇到一个1Hz 时钟上升沿时，计数加

一。num 共设计了8 个状态，000~111，对应不显示、6~1、不显示(0)8 个状态。

当计数结束后，即num="111"时，输出倒计时结束信号count\_end<='1';当回到空

档时，计数信号、计数结束信号重置，即num<="000"、count\_end<='0'。综上，

回到空档后需要重置的信号为倒计时开始信号、计数信号、计数结束信号。

接下来考虑(4)，上述状态机模块，需要实现“倒计时期间切换按键禁用”

的功能，num 从000~110 均为倒计时状态，111 为初始状态(不显示)，故可用

num="111"作为判断是否是在“倒计时期间”的条件。

倒计时模块中的计数器对应的Verilog 语言描述如下:

1. always @(posedge clkout or negedge rst)
2. begin
3. **if** (!rst)
4. place <= 0 ;
5. **else**  begin
6. place <= place + 1;
7. **if**(place==3'd7)  place <= 0;
8. end
9. end
11. always @(place)         //数码管选择
12. begin
13. **case** (place)
14. 3'b000 : DIG\_r <= 8'b0000\_0001;
15. //        3'b001 : DIG\_r <= 8'b0000\_0010;
16. //        3'b010 : DIG\_r <= 8'b0000\_0100;
17. 3'b011 : DIG\_r <= 8'b0000\_1000;
18. //        3'b100 : DIG\_r <= 8'b0001\_0000;
19. //        3'b101 : DIG\_r <= 8'b0010\_0000;
20. //        3'b110 : DIG\_r <= 8'b0100\_0000;
21. 3'b111 : DIG\_r <= 8'b1000\_0000;
22. **default** :DIG\_r <= 8'b0000\_0000;
23. endcase
24. end
25. always @ ({grade\_left, grade\_right}) //译码
26. begin
27. **case** (place)
28. 3'b000:
29. **case** (grade\_right)
30. 3'b000: Y\_r = 7'b1000000; // 0  1000000
31. 3'b001: Y\_r = 7'b1111001; // 1
32. 3'b010: Y\_r = 7'b0100100; // 2
33. 3'b011: Y\_r = 7'b0110000; // 3
34. //3'b100: Y\_r = 7'b0011001; // 4
35. //3'b101: Y\_r = 7'b0010010; // 5
36. //3'b110: Y\_r = 7'b0000010; // 6
37. //3'b111: Y\_r = 7'b0000001; // 7        从右向左为abcdefg 1为不亮 0-7为数码管从右向左排序
38. **default**: Y\_r = 7'b0000000;//8
39. endcase
40. 3'b011:
41. **case**(round)
42. 3'b000: Y\_r = 7'b1000000; // 0
43. 3'b001: Y\_r = 7'b1111001; // 1
44. 3'b010: Y\_r = 7'b0100100; // 2
45. 3'b011: Y\_r = 7'b0110000; // 3
46. 3'b100: Y\_r = 7'b0011001; // 4
47. 3'b101: Y\_r = 7'b0010010; // 5
48. 3'b100: Y\_r = 7'b0011001; // 4
49. 3'b101: Y\_r = 7'b0010010; // 5
50. 3'b110: Y\_r = 7'b0000010; // 6
51. 3'b111: Y\_r = 7'b0000001; // 7
52. **default**: Y\_r = 7'b0000000;//8
53. endcase
54. 3'b111:
55. **case** (grade\_left)
56. 3'b000: Y\_r = 7'b1000000; // 0
57. 3'b001: Y\_r = 7'b1111001; // 1
58. 3'b010: Y\_r = 7'b0100100; // 2
59. 3'b011: Y\_r = 7'b0110000; // 3
60. //3'b100: Y\_r = 7'b0011001; // 4
61. //3'b101: Y\_r = 7'b0010010; // 5
62. //3'b110: Y\_r = 7'b0000010; // 6
63. //3'b111: Y\_r = 7'b1111000; // 7        从右向左为abcdefg 1为不亮 0-7为数码管从右向左排序
64. **default**: Y\_r = 7'b0000000;
65. endcase
66. endcase
68. end