**汽车尾灯控制电路设计**

学院：数据科学与计算机学院 专业：计算机类

学号：16337341 姓名：朱志儒

电话：15989184223 Email:739741104@qq.com

指导老师：陈云洽 完成时间：2017/6/20

1. **设计目的**

设计一个汽车尾灯控制电路。已知汽车左右两侧各有3个尾灯，要求控制尾灯按如下规则亮灭。

（1）汽车沿直线行驶时，两侧的指示灯全灭；

（2）右转弯时，左侧的指示灯全灭，右侧的指示灯按 000，100，010，001，000 循环顺序点亮；

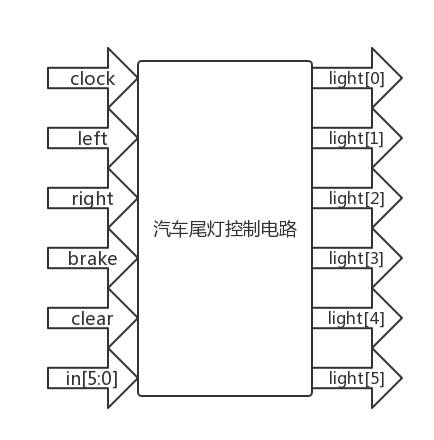
（3）左转弯时，右侧的指示灯全灭，左侧的指示灯按与右侧同样的循环顺序点亮；

（4）如果在直行时刹车，两侧的指示灯全亮；如果在转弯时刹车，转弯这一侧 的指示灯按上述的循环顺序点亮，另一侧的指示灯全亮。

1. **设计路线**

本次的课程设计我选择使用 Verilog 语言来完成。原因有以下：A、老师建议使用 HDL 语言；B、硬件描述语言能在各个抽象级别上进行设计，比画原理图方便；C、使用硬件描述语言设计起来使脉络显得更加清晰易懂。硬件描述语言主要两种，其中一种是 VHDL，另一种就是 Verilog。其实对两种HDL语言，我们的课程都有所涉及。针对我的实际情况，我选择使用 Verilog。因为VHDL比较严谨，更适合设计复杂的设计，而 Verilog 语言很灵活，与C语言很相近，学习起来很简单，能在比较短的时间里面掌握，最重要的是跟我们的编程习惯比较相近。 在之前的实验课中，我们都是使用 vivado2017.1来完成的，因此对 vivado2017.1软件的使用较为熟悉，而且 vivado2017.1功能强大，所以我选择在vivado2017.1环境下进行设计与仿真。

1. **实现过程**
2. **引脚简介**



汽车尾灯控制电路引脚图

1. **汽车尾灯控制状态转换描述**

汽车沿直线行驶时，light[5:0]全为0；

右转弯时，light[2:0]全为0，light[5:3]按000，100，010，001，000循环；

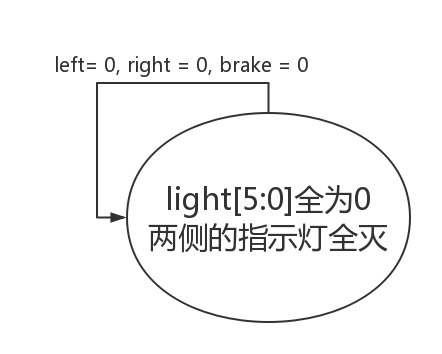
左转弯时,light[2:0]按000，100，010，001，000循环,light[5:3]全为0；

在直行时刹车,light[5:0]全为1；

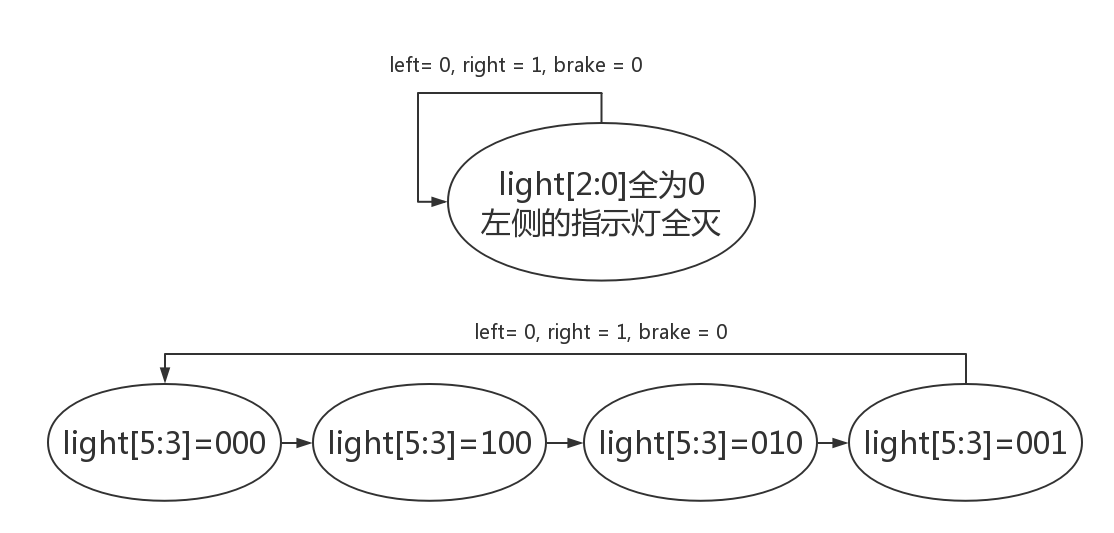
在右转时刹车，light[2:0]全为1，light[5:3]按000，100，010，001，000循环；

在左转时刹车,light[2:0]按000，100，010，001，000循环,light[5:3]全为1；

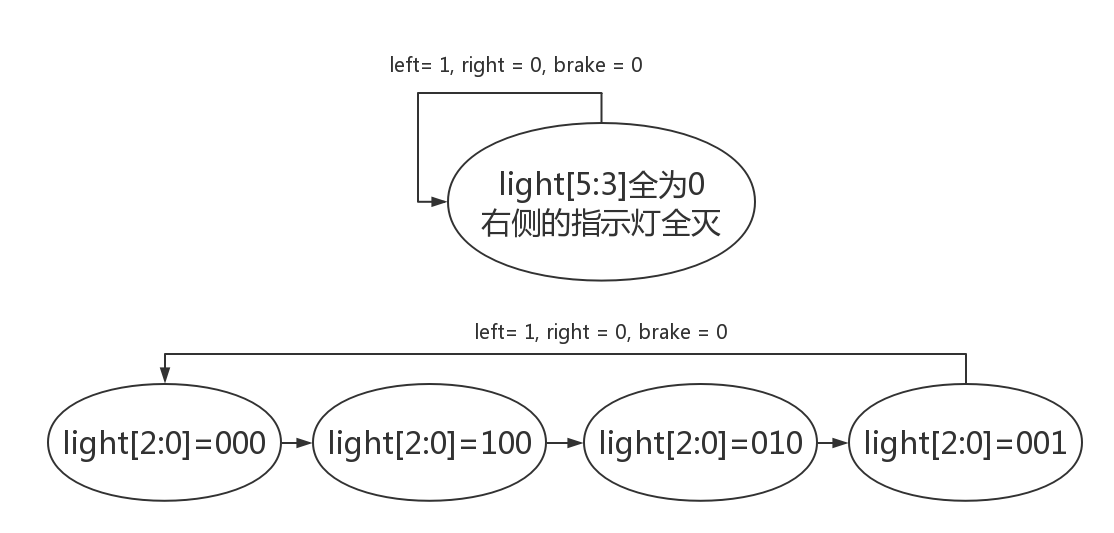
1. **汽车尾灯控制的状态转移图**
2. 直线行驶时



1. 右转弯时



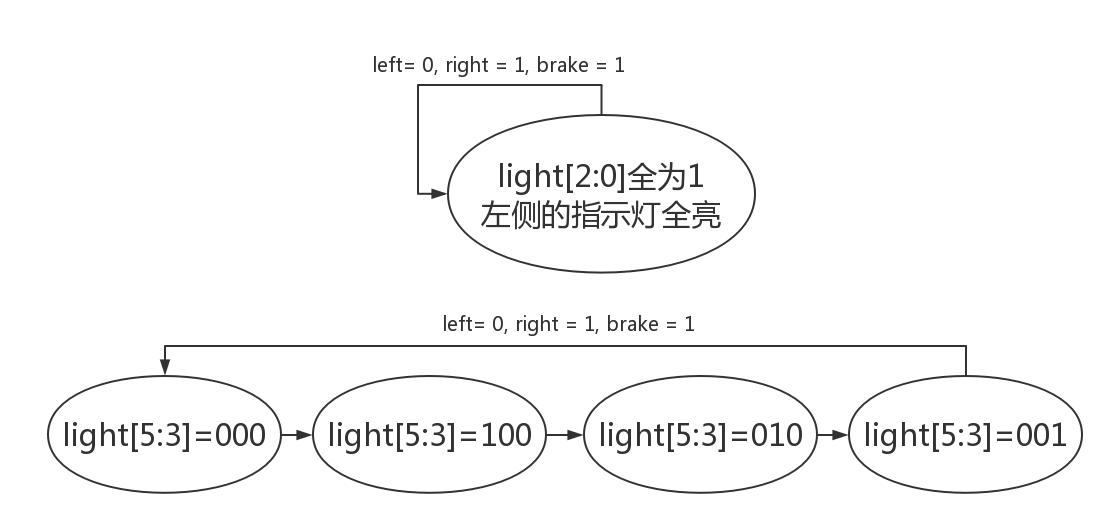
1. 左转弯时



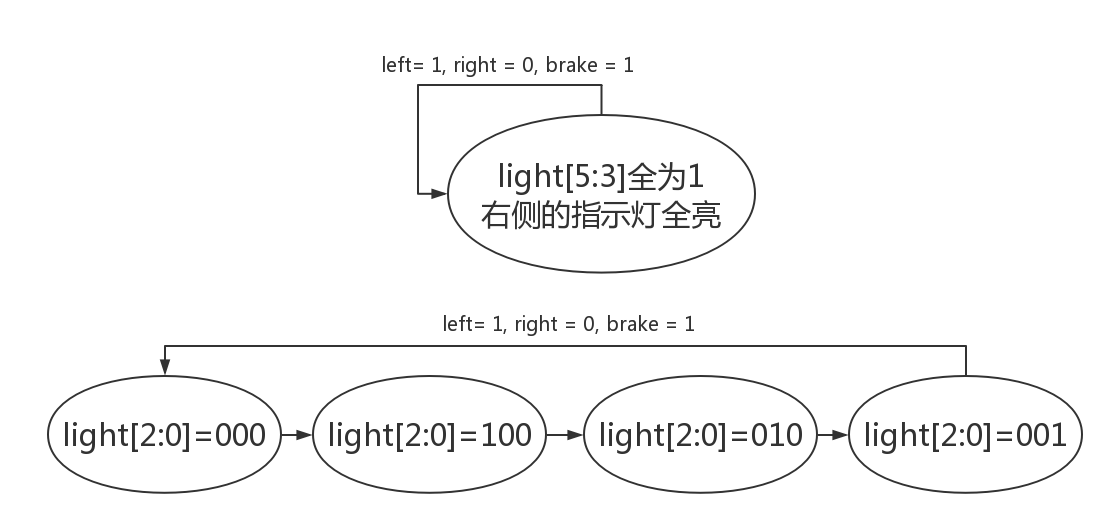
1. 在直行时刹车



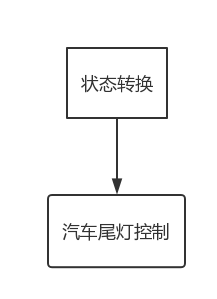
1. 在右转时刹车



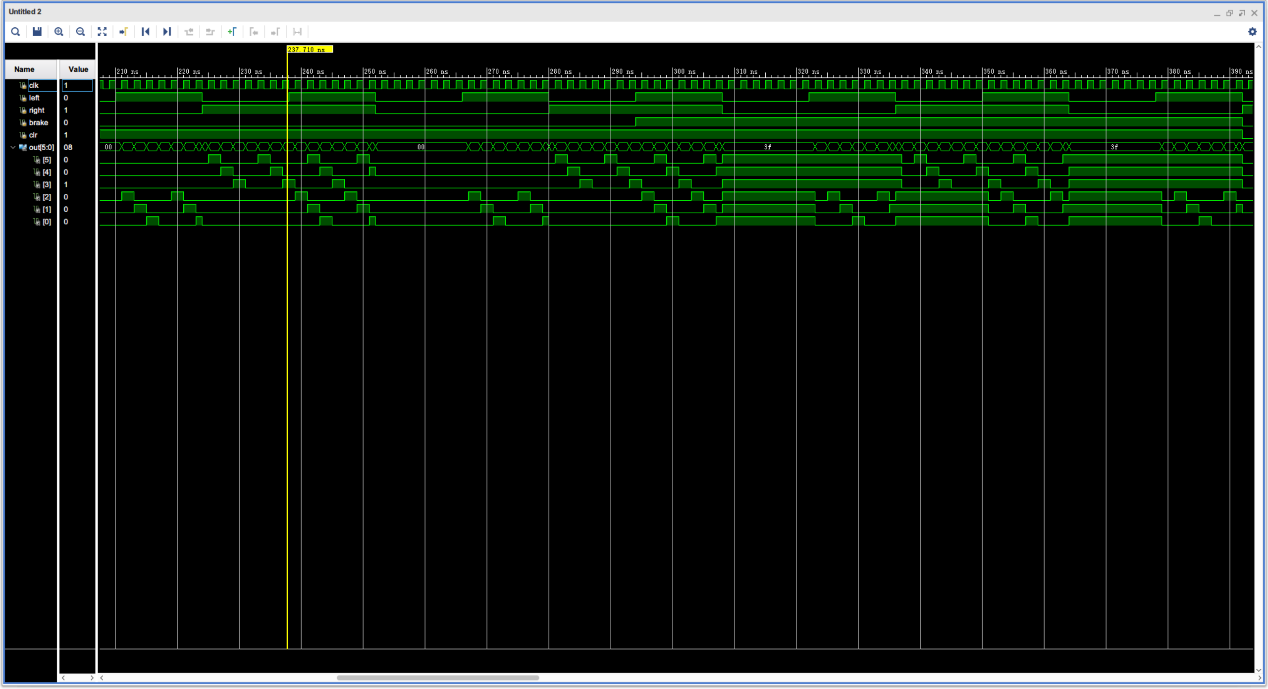
1. 在左转时刹车



1. **汽车尾灯控制的总体框图及功能分割图**

****

1. **仿真波形分析**

注：由于vivado的仿真波形时间长度局限在1000000ps，如果我们使用周期为1s的clk，则会导致波形无法全部显示，所以我们便在模拟波形写project的simu文件时，将clk由1ns为周期来代替，因此在以下的波形中，1ns均代表实际的1s。波形显示的不足之处在于，我们无法通过 button控制left, right和brake三个输入信号，所以只能选择让其作为有周期的时钟信号输入，在观察波形图的时候，要注意，只有当clear=0时，系统才正常工作。

图中出现left和right均为1的输入，这是一个无效输入请无视。

1. **效果分析**

本次课程设计，成功达到了预期效果，汽车沿直线行驶时，两侧的指示灯全灭；右转弯时，左侧的指示灯全灭，右侧的指示灯按 000，100，010，001，000循环顺序点亮；左转弯时，右侧的指示灯全灭，左侧的指示灯按与右侧同样的循环顺序点亮；如果在直行时刹车，两侧的指示灯全亮；如果在转弯时刹车，转弯这一侧的指示灯按上述的循环顺序点亮，另一侧的指示灯全亮。

在vivado2017.1环境下进行波形仿真后发现其波形（时间间隔与显示状态）与预期效果一致！

1. **总结**

1、由于总体设计思路较为清晰，此次基于verilog语言的汽车尾灯控制电路的设计过程虽多有波折，但总体较为顺利。

2、在电路具体框架的具体设立上，我在查阅多方相关资料、进行一定的知识储备积累后，结合自身实际水平，最终选定了汽车尾灯控制电路的设计系统为设计方向，并在仔细推敲、讨论、反思以及改进之后，制订了较为详细和清晰的总体思路。

3、基于汽车尾灯控制电路的设计思路，我们充分考虑了汽车尾灯控制电路在现实生活中的实用性、可行性和科学性，仔细制订了的汽车尾灯的有限状态机，并以此为基础开展 verilog主体程序的编写。

4、由于 reset 是周期信号，所以有大篇幅的无效波形，这是我需要改进的地方， 我会继续考虑如何进行优化改进！

1. **参考资料**

《精通Verilog HDL语言编程》刘波编著 电子工业出版社

1. **附录**

//源程序代码

`timescale 1ns / 1ps

module state\_machine(

input clk, //1Hz时钟信号

input clr, //清零信号输入

input pre, //置数信号输入

input [2:0] prefore, //前一状态

output reg [2:0] later //后一状态

);

always @ (posedge clk or negedge clr or posedge pre) //状态转换

begin

if (clr == 0) later = 3'b000;

else if (pre == 1) later = 3'b111;

else

case(prefore)

3'b000 : later = 3'b100;

3'b100 : later = 3'b010;

3'b010 : later = 3'b001;

3'b001 : later = 3'b000;

default : later = 3'b000;

endcase

end

endmodule

module project(

input clk, //1Hz时钟信号

input left, //左转信号输入

input right, //右转信号输入

input brake, //刹车信号输入

input clr, // 清零信号输入

input [5:0] in, //前一状态

output [5:0] light //后一状态

);

wire lc;

assign lc = (left | right | brake) & (left | ~right | brake) & clr; //左侧指示灯清零信号

wire rc;

assign rc = (left | right | brake) & (~left | right | brake) & clr; //右侧指示灯清零信号

wire lp;

assign lp = ~left & ~right & brake | ~left & right & brake; //左侧指示灯置数信号

wire rp;

assign rp = ~left & ~right & brake | left & ~right & brake; //右侧指示灯置数信号

state\_machine mysm1(clk, lc, lp, in[2:0], light[2:0]); //控制右侧指示灯

state\_machine mysm2(clk, rc, rp, in[5:3], light[5:3]); //控制左侧指示灯

endmodule

module simu();

reg clk = 0;

always #1 clk = ~clk; //1Hz时钟信号产生

reg left = 0;

reg right = 1;

always #14 left = ~left; //左转信号产生

always #28 right = ~right; //右转信号产生

reg brake = 0;

always #98 brake = ~brake; //刹车信号产生

reg clr = 0;

always #196 clr = ~clr; //清零信号产生

wire [5:0] out; //指示灯输出

project myp(clk, left, right, brake, clr, out[5:0], out[5:0]); //控制指示灯

endmodule