### 实验四 基于FPGA的自动售货机

2019010175 孔瑞阳 计科91

### 设计思路

**计算模块：**

根据输入处理售货机中的钱数。分出一个很慢的CLK在上升沿处进行判断，如果与上次结果不同则再进行计算来消除抖动、去重。

**选择模块：**

根据cur的值确定在哪个数码管上输出。

cur为00时第四个数码管显示小数点后第一位。

cur为01时第三个数码管显示个位数和小数点。

cur为10且十位数不为0时第二个数码管显示十位数。

cur为00时第一个数码管不显示（永远不显示）。

**输出模块：**

通过data的值确定数码管的各个部分的亮暗。

### 实现代码

module experiment(

input CLK,

input [5:0] in, // 投币5/1/0.5，购买2.5/1.5，退币

output reg [7:0] R, // 数码管输出

output reg [3:0] sel, // 选择在哪个数码管输出

output reg lack, coin, goods

);

integer money; // 保存钱数

reg [5:0] lastin;

reg [7:0] data;

reg [1:0] cur; // 当前显示的位

integer cnt, cnt2; // 统计时间周期个数

parameter cntmax = 50000000 / 400; // 400Hz

parameter cntmax2 = 50000000 / 10; // 10Hz

always @ (posedge CLK) // 计算模块

begin

cnt2 <= cnt2 + 1;

if (cnt2 == cntmax2)

cnt2 <= 0;

if ((cnt2 == 0) && (in != lastin))

begin

lack <= 0;

coin <= 0;

goods <= 0;

if (in[0] == 1) // 退币

begin

coin <= 1;

money <= 0;

end

if ((in[2] == 1) && (lastin[2] == 0)) // 购买2.5元商品

if (money >= 25)

begin

money <= money - 25;

goods <= 1;

end

else

lack <= 1;

if ((in[1] == 1) && (lastin[1] == 0)) // 购买1.5元商品

if (money >= 15)

begin

money <= money - 15;

goods <= 1;

end

else

lack <= 1;

always @ (posedge CLK)

begin

cnt <= cnt + 1;

if (cnt == cntmax)

begin

cnt <= 0;

cur <= cur + 1;

end

end

always @ (\*) // 输出模块

begin

case (data[7:0]) // 输出

always @ (\*) // 选择模块，十进制输出

begin

sel <= 4'b1111;

case (cur)

2'b00: // 第四个数码管保存小数点后第一位

begin

data <= money % 10;

sel <= 4'b1110;

end

2'b01: // 第三个数码管保存个位，并显示小数点

begin

data <= money / 10 % 10 + 8'h10;

sel <= 4'b1101;

end

2'b10: // 当十位有数时第二个数码管输出

begin

data <= (money >= 100 ? money / 100 % 10 : 8'h20);

sel <= 4'b1011;

end

2'b11: // 第一个数码管不输出

begin

data <= 8'h20;

sel <= 4'b0111;

end

endcase

end

if ((in[5] == 1) && (lastin[5] == 0)) // 投币5元

money <= money + 50;

if ((in[4] == 1) && (lastin[4] == 0)) // 投币1元

money <= money + 10;

if ((in[3] == 1) && (lastin[3] == 0)) // 投币0.5元

money <= money + 5;

lastin <= in;

end

end

8'h00: R[7:0] <= 8'b11111100;

8'h01: R[7:0] <= 8'b01100000;

8'h02: R[7:0] <= 8'b11011010;

8'h03: R[7:0] <= 8'b11110010;

8'h04: R[7:0] <= 8'b01100110;

8'h05: R[7:0] <= 8'b10110110;

8'h06: R[7:0] <= 8'b10111110;

8'h07: R[7:0] <= 8'b11100000;

8'h08: R[7:0] <= 8'b11111110;

8'h09: R[7:0] <= 8'b11110110;

8'h10: R[7:0] <= 8'b11111101; // 加上小数点

8'h11: R[7:0] <= 8'b01100001;

8'h12: R[7:0] <= 8'b11011011;

8'h13: R[7:0] <= 8'b11110011;

8'h14: R[7:0] <= 8'b01100111;

8'h15: R[7:0] <= 8'b10110111;

8'h16: R[7:0] <= 8'b10111111;

8'h17: R[7:0] <= 8'b11100001;

8'h18: R[7:0] <= 8'b11111111;

8'h19: R[7:0] <= 8'b11110111;

default:

R[7:0] <= 8'b00000000;

endcase

end

endmodule