**题目：简化考虑，假设饮料只有一种价格为2.5元。硬币有0.5元和1.0元两种，考虑找零，用Verilog描述其控制电路，并用FPGA实现**

**一、分析问题**

假设饮料只有一种价格为2.5元，简化问题为我们的目的为只买一瓶饮料，在已投入价格能出一瓶饮料之后就不再继续投币。所以我们最终得到的结果是刚好2.5元，不许找零，或者在已投入2元时再投入1个1元硬币，此时需找零0.5元。

而硬币只有0.5元和1元两种，时序电路的输出取决于输入信号和电路原有状态，在本问题中输入信号即为本次投入硬币，电路原有状态为已投入的硬币数。

**二、分析输入输出端口信号**

输入信号： clk，rst；

输入信号：操作开始: start； //定义1开始操作

输入信号：投币币值: coin\_val； //定义2’b01表示0.5元； 2’b10表示1元

输入信号：取消操作指示 : cancel; //定义1为取消操作

输出信号：机器是否占用: hold\_ind； //定义0为不占用，可以使用

输出信号：取饮料信号: drinktk\_ind； //定义1为取饮料

输出信号：找零与退币标志信号: charge\_ind; //定义1为找零

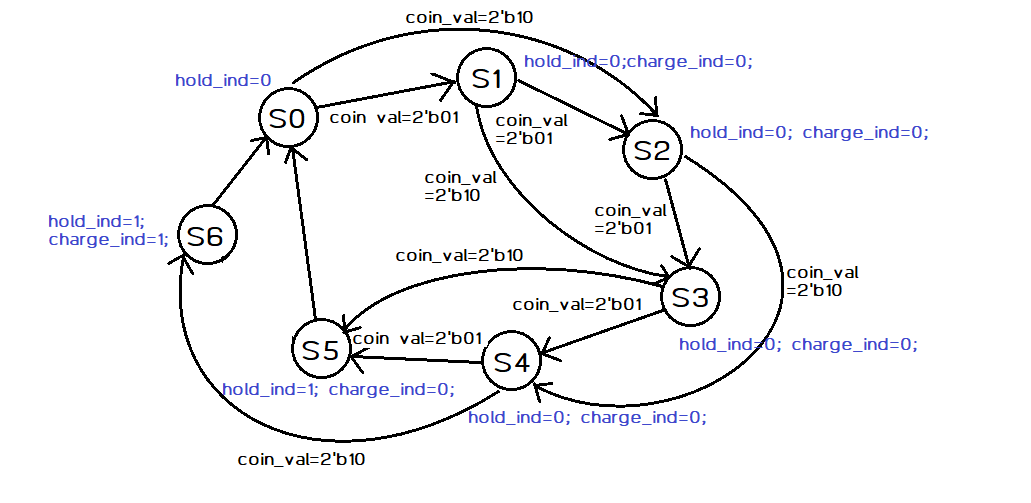
输出信号：找零与退币币值： charge\_val; //定义3’b001表示找0.5元； 3’b010表示找1元；3’b011表示找1.5元；3’b100表示找2.0元;

**二、分析状态转移**

状态定义说明如下：

|  |  |
| --- | --- |
| S0 | 初始态 |
| S1 | 已投币0.5元 |
| S2 | 已投币1.0元 |
| S3 | 已投币1.5元 |
| S4 | 已投币2.0元 |
| S5 | 已投币2.5元 |
| S6 | 已投币3.0元 |

画得状态转移图如下：



图意为：

在S0状态下，如果start=1则开始检测是否有投币，如果有，则开始售货操作；在S1,S2,S3,S4状态下，如果检测到cancel等于1，则退币并返回状态S0，再将cancel置于0；在状态S5卖出饮料不找零，返回状态S0在状态S6卖出饮料并找零，返回状态S。在S状态下，hold=0,可以发出新一轮操作，其他状态都等于1。

**三、根据状态转移图进行Verilog 语言描述**

Module softdrinkFSM (clk,rst,op\_start,cancel\_flag,coin\_val,hold\_ind,charge\_ind,

drinktk\_ind,charge\_val);

input clk,rst； //输入端口声明

input op\_start，cancel\_flag;

input [1:0] coin\_val;

output hold\_ind，charge\_ind,drinktk\_ind; //输出端口声明

output [2:0] charge\_val;

reg hold\_ind，charge\_ind, drinktk\_ind; /信号类型声明

reg[2:0] charge\_val;

reg[2:0] currentstate, nextstate;

parameter S0=3'b000; //初始化

parameter S1=3'b001;

parameter S2=3'b010;

parameter S3=3'b011;

parameter S4=3'b100;

parameter S5=3'b101;

parameter S6=3'b110

always@(posedge clk or posedge rst)

if (rst) currentstate<= S0;

else currentstate<= nextstate;

always@ (currentstate or rst or op\_start or cancel\_flag or coin\_val)

if(rst) nextstate=S0;

else case (currentstate)

S0: if(op\_start)

if(coin\_val==2'b01) nextstate=S1; //投币功能实现

else if(coin\_val==2'b10) nextstate=S2;

S1: if(cancel\_flag) nextstate=S0;

else if(coin\_val==2'b01) nextstate=S2;

else if(coin\_val==2'b10) nextstate=S3;

S2: if(cancel\_flag) nextstate=S0;

else if(coin\_val==2'b01) nextstate=S3;

else if(coin\_val==2'b10) nextstate=S4;

S3: if(cancel\_flag) nextstate=S0;

else if(coin\_val==2'b01) nextstate=S4;

else if(coin\_val==2'b10) nextstate=S5;

S4: if(cancel\_flag) nextstate=S0;

else if(coin\_val==2'b01) nextstate=S5;

else if(coin\_val==2'b10) nextstate=S6;

S5: nextstate=S0;

S6: nextstate=S0;

default: nextstate=S0;

endcase

always @ (currentstate)

if (currentstate== S0) hold\_ind= 1'b0; //确定hold\_ind

else hold\_ind= 1'b1;

always@ (currentstate)

if ((currentstate== S5 )||(currentstate== S6)) //确定是否出饮料（S5S6）

drinktk\_ind= 1'b1;

else drinktk\_ind= 1'b0;

always@ (currentstateor cancel\_flag) //确定是否找零（S6、中途取消）

if(currentstate== S0) charge\_ind= 1'b0;

else if(currentstate== S6) charge\_ind= 1'b1;

else if(cancel\_flag) charge\_ind= 1'b1;

else charge\_ind= 1'b0;

always @ (currentstateor cancel\_flag) //确定找零钱数

if(currentstate== S0) charge\_val= 3'b000

else if(currentstate== S6) charge\_val= 3'b001

else if(cancel\_flag)

begin

case (currentstate)

S1: charge\_val= 3'b001;

S2: charge\_val= 3'b010;

S3: charge\_val= 3'b011;

S4: charge\_val= 3'b100;

default: charge\_val= 3'b000;

endcase

end

else charge\_val= 3'b000;

endmodul

**四、测试代码**

module softdrink\_testbench;

reg rst, clk;

reg op\_start;

reg cancel\_flag;

reg [1:0] coin\_val;

wire hold\_ind;

wire charge\_ind;

wire drinktk\_ind;

wire [2:0] charge\_val;

initial clk=0；

always #500 clk=~clk;

initial

begin

rst=0;

op\_start=0;

cancel\_flag=0;

coin\_val=2'b00;

#25 rst=1;

#25 rst=0;

//第一次：依次投入一元、一元、一元的硬币

#50 op\_start=1;

#300 coin\_val=2'b10;

#1000 coin\_val=2'b10;

#1000 coin\_val=2'b10;

#1000 op\_start=0;

//第二次：依次投入0.5元、一元的硬币，然后取消操作

#2000 op\_start=1; coin\_val=2'b01;

#1000 coin\_val=2'b10;

#1000 cancel\_flag=1'b1; op\_start=0;

#1000 cancel\_flag=0;

//第三次：依次投入0.5元、0.5元、0.5元、0.5元、一元 的硬币

#2000 op\_start=1; coin\_val=2'b01;

#1000 coin\_val=2'b01;

#1000 coin\_val=2'b01;

#1000 coin\_val=2'b01;

#1000 coin\_val=2'b10;

#1000 op\_start=0;

#1000000 $stop;

end //end initial

softdrinkFSM u0 (rst,clk,op\_start,coin\_val,cancel\_flag, hold\_ind,charge\_ind,drinktk\_ind,charge\_val);

endmodule