《数字逻辑与数字系统》实验报告

学院 软件工程 年级 2019 级 班级 1 姓名 俞林昊

学号 3019207450 课程名称 数字逻辑与数字系统 实验日期 2021.6.5

成绩 同组实验者

实验项目名称 自动贩售机的设计与实现

一. 实验目的

1. 掌握有限状态机的设计方法。;

2. 能够使用 SystemVerilog 进行三段式状态机的建模。

二. 实验内容

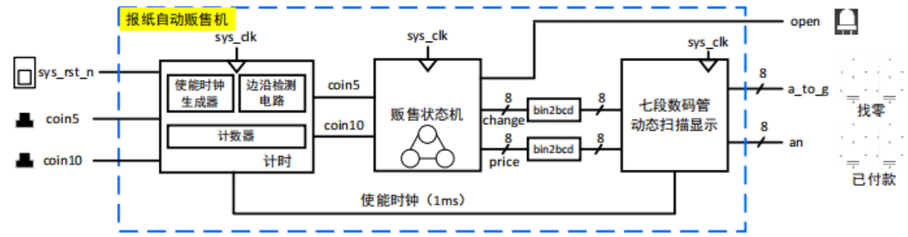
采用有限状态机，基于 SystemVerilog HDL 设计并实现一个报纸自动贩售机。 整个工程的顶层模块如图 4-3 所示，输入/输出端口如表 4-1 所示。使用 **4 个七 段数码**管实时显示已付款和找零情况。其中，两个数码管对应“已付款”，另两个 数码管对应“找零”，单位为分。通过 **1 个拨动开关**对数字钟进行复位控制。使用 **两个按键**模拟投币，其中一个按键对应 5 分，另一个按键对应 1 角。使用 **1 个 LED 灯**标识出售是否成功，灯亮表示出售成功，否则表示已付款不够，出售失败。

假设报纸价格为 15 分，合法的投币组合包括:

- 1 个 5 分的硬币和一个 1 角的硬币，不找零
- 3 个五分的硬币，不找零
- 1 个 1 角的硬币和一个 5 分的硬币，不找零
- 两个 1 角的硬币是合法的，找零 5 分。

天津大学本科生实验报告专用纸

当投入硬币的组合为上面 4 种之一时，则购买成功，LED 灯亮。**购买成功后，*LED* 灯持续亮 10 秒，然后自动熄灭，同时 4 个数码管也恢复为 0**。



报纸自动贩售机由 4 部分构成。

★ 第一部分是**计时器模块**，该模块又由 3 个子模块构成，分别是计数器电路、使能时钟生成电路和边沿检测电路。

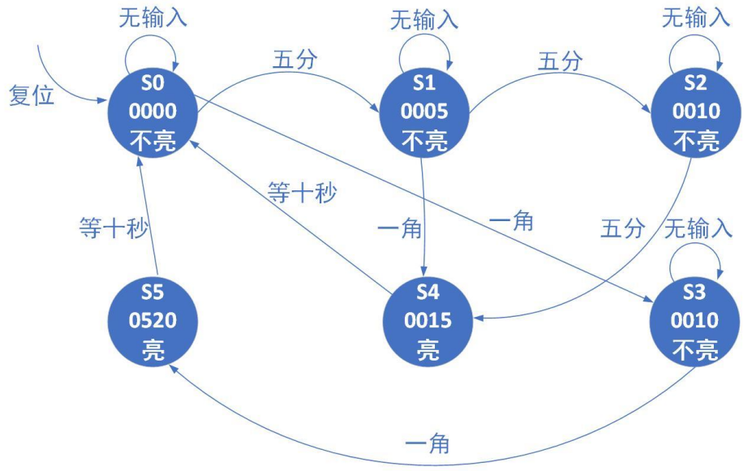
★ 第二部分是整个自动贩售机电路的核心——**贩售机状态机**。状态机根据投币情况产生“已付款”和“找零”输出。此外，如果已付款超过 15 分，则将 LED 灯点亮，表示出售成功。

★ 第三部分是两个 **8 位二进制转 *BCD* 模块**，分别将二进制的“已付款”和“找零”值转化为 BCD 编码，即 10 进制数。**本实验中，该模块不需要实现，由教师直接提供 *IP* 使用**。

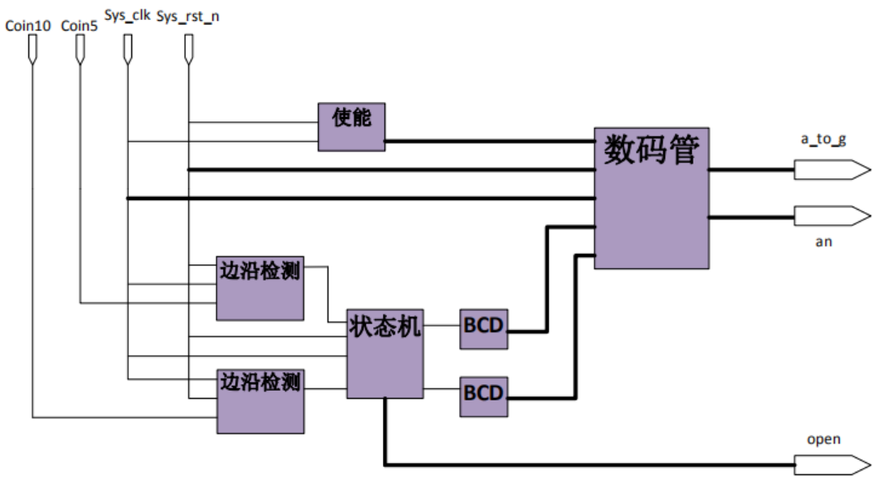
★ 第四部分是 **7 段数码管动态扫描显示模块**，它实现“已付款”和“找零”值的最终显示.

三． 实验原理与步骤（注：步骤不用写工具的操作步骤，而是设计步骤）

1. 画出自动贩售机的状态转换图。



2. 画出自动贩售机电路的原理图（模块级别即可，如使能时钟模块、边沿检测模块等）。



3. 报纸自动贩售机的 SystemVerilog 代码。

- **边沿检测模块：**

```
module pressCheck(
input logic sys_clk,
input logic sys_rst_n,
input logic i_btn,
output logic pose_edge
    );
 logic flag1 = 0;
 logic flag2 = 0;
   always_ff@(posedge sys_clk,posedge sys_rst_n
   begin
       if(!sys_rst_n)
       begin
           flag1 <= 0;
           flag2 <= 0;
       end
       else
       begin
           flag1 <= i_btn;
           flag2 <= flag1;
       end
   end
   assign pose_edge = (~flag2) & flag1;
   endmodule
```

- **使能时钟模块：**

```
module enClock(
input logic sys_clk,
input logic sys_rst_n,
output logic flag
    );
    logic[15:0]cnt=0;
    always_ff@(posedge sys_clk,posedge sys_rst_n)
    begin
        if(!sys_rst_n)
        begin
            cnt<=0;
            flag<=0;
        end
        else
        begin
            cnt<=cnt+1;
```

```
        begin
            cnt<=cnt+1;
            if(cnt==25000)
            begin
                cnt<=0;
                flag<=1;
            end
            else flag<=0;
        end
    end
endmodule
```

- **7 段数码管模块**

```
module seg7(
input logic sys_clk,
input logic sys_rst_n,
input logic en_clk,
input logic[7:0]change,
input logic[7:0]price,
output logic[7:0]a_to_g,
output logic[3:0]an
    );
logic [3:0] tmp = 0;
logic [2:0] cnt = 0;
always_ff@(posedge sys_clk,posedge sys_rst_n)   begin
        if(!sys_rst_n)
        begin
            an <= 4'b0000;
            cnt <= 2'b00;
            tmp <= 4'b0000;
        end
        else if(en_clk)
        begin
           cnt <= cnt+1;
           if(cnt%4 == 0) begin
               an <= 4'b0001;
               tmp <= price[3:0];
           end
           if(cnt%4 == 1) begin
               an <= 4'b0010;
           tmp <= price[7:4];
       end
       if(cnt%4 == 2) begin
           an <= 4'b0100;
           tmp <= change[3:0];
       end
           if(cnt%4 == 3) begin
               an <= 4'b1000;
               tmp <= change[7:4];
       end
           end
    end
always_comb
begin
case(tmp)
               4'h0:a_to_g = 8'b11000000;
               4'h1:a_to_g = 8'b11111001;
               4'h2:a_to_g = 8'b10100100;
               4'h3:a_to_g = 8'b10110000;
               4'h4:a_to_g = 8'b10011001;
               4'h5:a_to_g = 8'b10010010;
               4'h6:a_to_g = 8'b10000010;
               4'h7:a_to_g = 8'b11111000;
               4'h8:a_to_g = 8'b10000000;
               4'h9:a_to_g = 8'b10010000;
           default: a_to_g = 8'b00000000;
           endcase
end
endmodule
```

- **有限状态机模块**

```
module stateMachine(
input sys_clk, sys_rst_n,
input press_5,
input press_10,
output logic [7:0] change = 0,
output logic [7:0] price = 0,
output logic open = 0
    );
    logic [4:0] current_state = 0;
    logic [4:0] next_state = 0;
    logic [35:0] cnt = 0;

always_ff@ (posedge sys_clk, posedge sys_rst_n) begin
    if(!sys_rst_n)
        current_state <= 5'b00000;
    else
        current_state <= next_state;
    end

always_comb begin
    case(current_state)
```

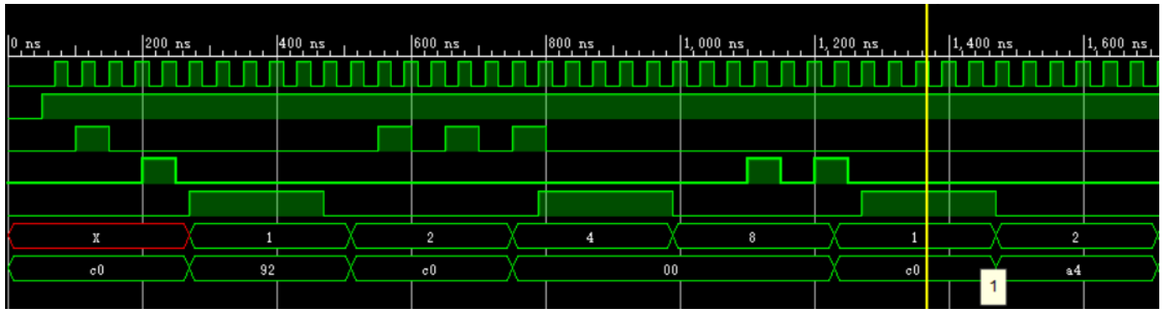
5'b00001:begin

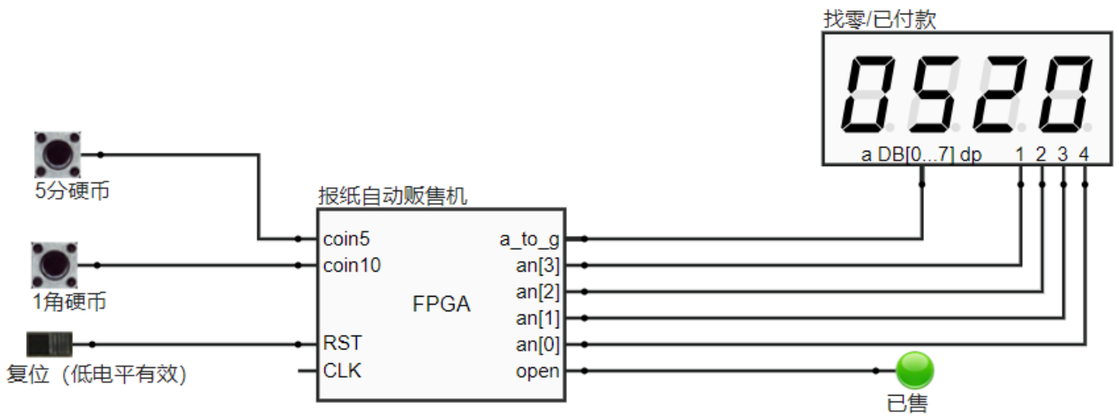
```
    5'b00000:begin
        if(press_5==0&&press_10==0) next_state = 5'b00000;
        else if(press_5==1) next_state = 5'b00001;
        else if(press_10==1) next_state = 5'b00100;
    end
    5'b00001:begin
        if(press_5==0&&press_10==0) next_state = 5'b00001;
        else if(press_5==1) next_state = 5'b00010;
        else if(press_10==1) next_state = 5'b01000;
    end
    5'b00010:begin
        if(press_5==0&&press_10==0) next_state = 5'b00010;
        else if(press_5==1) next_state = 5'b01000;
    end
    5'b00100:begin
        if(press_5==0&&press_10==0) next_state = 5'b00100;
        else if(press_5==1) next_state = 5'b01000;
        else if(press_10==1) next_state = 5'b10000;
    end
    5'b01000:begin
                if(cnt==250000000)begin
                   next_state = 5'b00000;
                end
                else next_state = 5'b01000;
            end

    5'b10000:begin
                if(cnt==250000000)begin
                    next_state = 5'b00000;
                end
                else next_state = 5'b10000;
            end
    default: next_state = 5'b00000;
    endcase
end

always_ff@ (posedge sys_clk) begin
    case(next_state)
     5'b00000:begin
        cnt <= 0;
        change <= 0;
        price <= 0;
        open <= 0;
    end
```

```
    5'b00001:begin
       change <= 0;
       price <= 8'b00000101;
       open <= 0;
    end
    5'b00010:begin
       change <= 0;
       price <= 8'b00001010;
       open <= 0;
    end
    5'b00100:begin
       change <= 0;
       price <= 8'b00001010;
       open <= 0;
    end
    5'b01000:begin
       change <= 0;
       price <= 8'b00001111;
       open <= 1;
       cnt<=cnt+1;
    end

    5'b10000:begin
       change <= 8'b00000101;
       price <= 8'b00010100;
       open <= 1;
       cnt<=cnt+1;
     end
    endcase
end
endmodule
```

四．仿真与实验结果（注：仿真需要给出波形图截图，截图要清晰，如果波形过长，可以分段截取；实验结果为远程 FPGA 硬件云平台的截图）

注：远程 FPGA 硬件云平台截图只需要一个测试激励即可





五．实验中遇到的问题和解决办法

1. 编写状态机模块的时候忘记考虑不存在两个五分和一个一角的情况。

2. 仿真模拟的时候一开始没有把 cnt5 和 cnt10 置为零，导致出现了错误。

3. 通过本实验学会了用三段式实现状态机。

教师签字：

年　月　日