**中国科学技术大学计算机学院**

**《数字电路实验》报告**



实验题目：信号处理及有限状态机

学生姓名：陈鸿绪

学生学号：PB21000224

完成日期：11.26

计算机实验教学中心制

2020年09月

【实验题目】

信号处理及有限状态机

【实验目的】

进一步熟悉 FPGA 开发的整体流程;

掌握几种常见的信号处理技巧;

掌握有限状态机的设计方法;

能够使用有限状态机设计功能电路;

【实验环境】

VLAB：vlab.ustc.eud.cn

FPGAOL:fpgaol.ustc.edu.cn

Logisim

Vivado

【实验练习】

T1. 在不改变电路功能和行为的前提下，将前面 Step5 中的代码 改写成三段式有限状态机的形式，写出完整的 Verilog 代码。

[解]：Verilog代码如下：

**module test(**

**input clk,rst,**

**output led);**

**reg [1:0] cur\_state;**

**reg [1:0] next\_state;**

**parameter S0=2’b00;**

**parameter S1=2’b01;**

**parameter S2=2’b10;**

**parameter S3=2’b11;**

**always@(\*)**

**begin**

**case(cur\_state)**

**S0: next\_state = S1;**

**S1: next\_state = S2;**

**S2: next\_state = S3;**

**S3: next\_state = S0;**

**endcase**

**end**

**always@(posedge clk or posedge rst)**

**begin**

**if(rst) cur\_state <= 2’b0;**

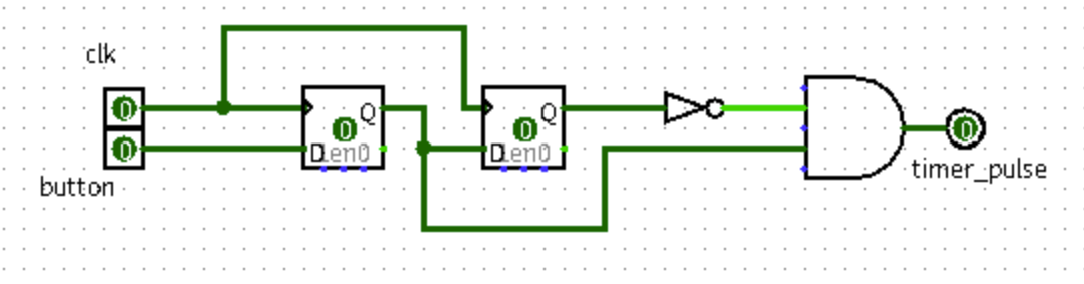
**else cur\_state <= next\_state;**

**end**

**assign led = (cur\_state==2'b11) ? 1'b1 : 1'b0;**

**endmodule**

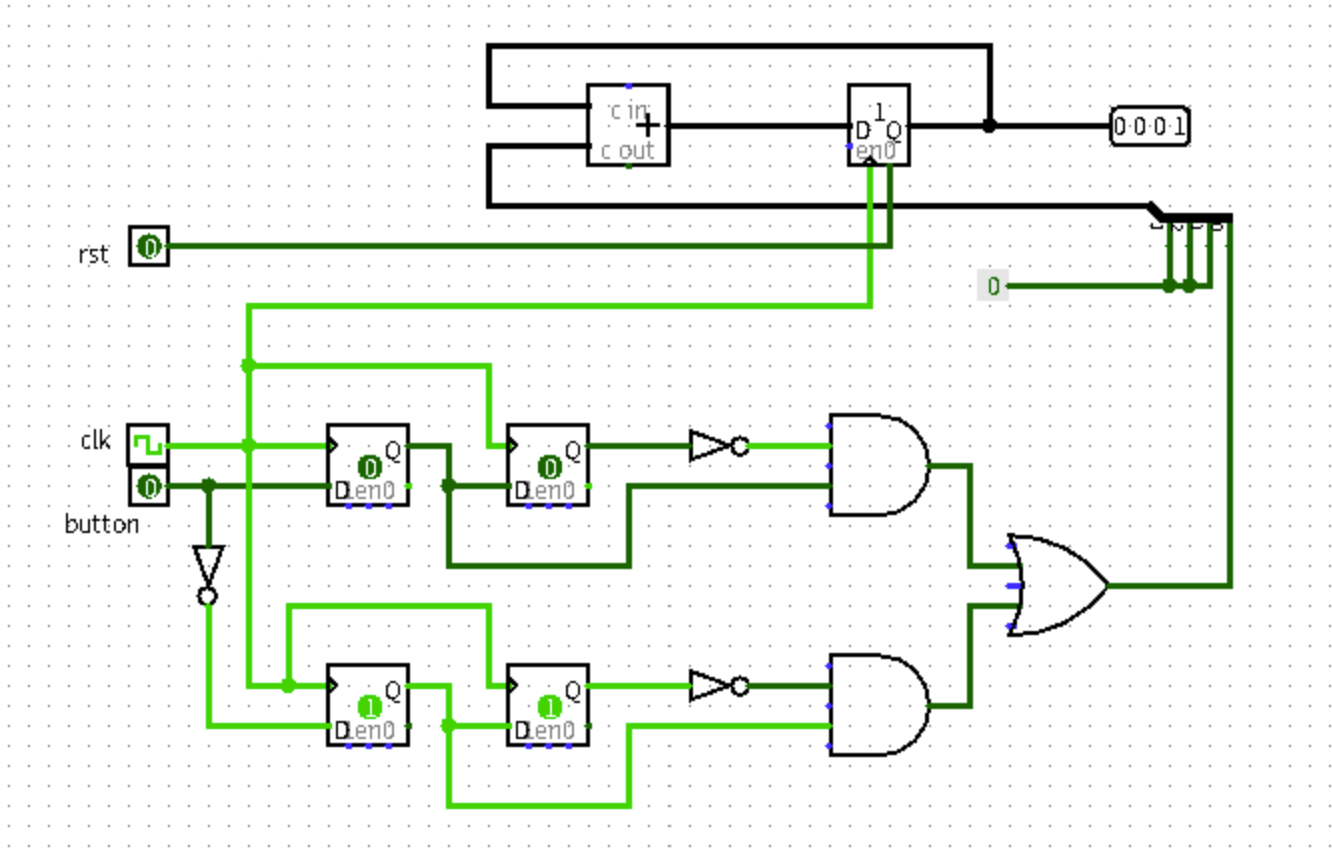
T2.请在 Logisim 中设计一个 4bit 位宽的计数器电路，如下图所示，clk 信号为计数器时钟，复位时（rst==1）计数值为 0，在输入信号 sw 电平发生变化时，计数值 cnt 加 1，即在 sw 信号上升沿时刻和下降沿时刻各触发一次计数操作，其余时刻计数器保持不变。

[解]：由实验文档的消除毛刺的逻辑电路：

该电路实现了取信号边缘得到一个周期脉冲（上升沿）

再将button取反，同理实现下降沿信号边缘的一个周期的脉冲

将两个逻辑电路结合可以得到sw只要变化就触发计数的操作

下图为设计的逻辑电路：

T3. 设计一个 8 位的十六进制计数器，时钟采用板载的 100MHz 时钟，通过 sw[0]控制计数模式，开关为 1 时为累加模式，为 0 时为 递减模式，按键控制计数，按下的瞬间根据开关的状态进行累加或递 减计数。计数值用数码管显示，其复位值为“1F”。

[解]：首先给出Verilog设计代码：

**module test(**

**input clk,sw,btn,rst,**

**output reg [2:0] sel,**

**output reg [3:0] dat**

**);**

**reg button\_r1,button\_r2;**

**reg [7:0] cout;**

**always@(posedge clk) button\_r1 <= btn;**

**always@(posedge clk) button\_r2 <= button\_r1;**

**assign button\_edge = button\_r1 & (~button\_r2);**

**always@(posedge clk)**

**begin**

**if(rst) cout <= 8'h1f;**

**else if(button\_edge)**

**begin**

**if(sw) cout<=cout+1;**

**else cout<=cout-1;**

**end**

**end**

**reg [2:0] temp;**

**always@(posedge clk)**

**begin**

**temp<=temp+1;**

**sel[2]<=0;**

**sel[1]<=0;**

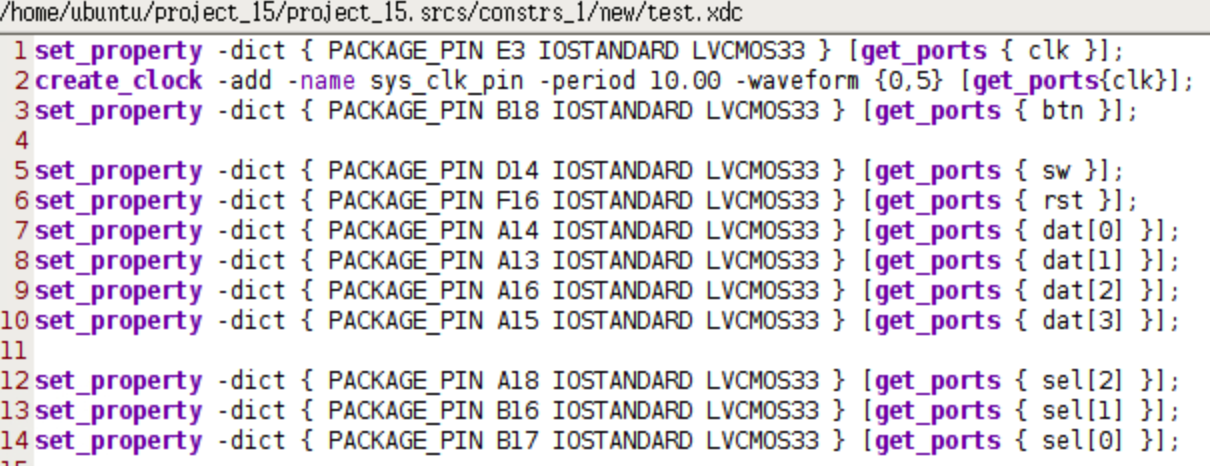
**sel[0]<=temp[2];**

**if(temp[2]==0) dat<=cout[3:0];**

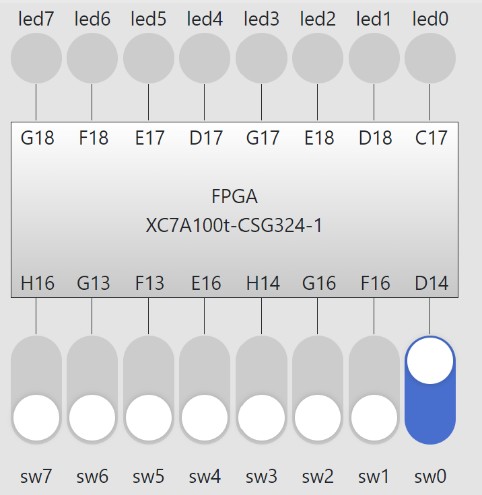
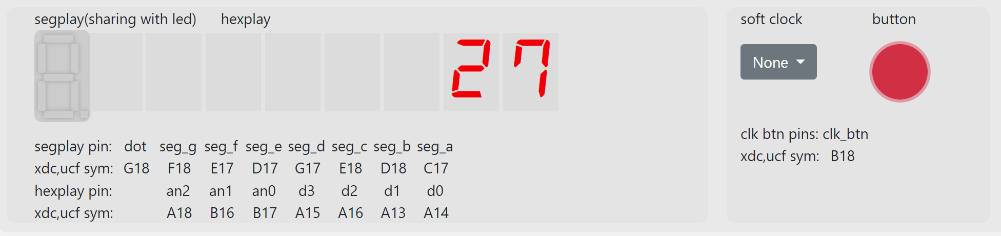
**else dat<=cout[7:4];**

**end**

**endmodule**

再给出XDC约束文件：

将生成的 bit文件烧写在fpga平台上，运行过程截图如下：

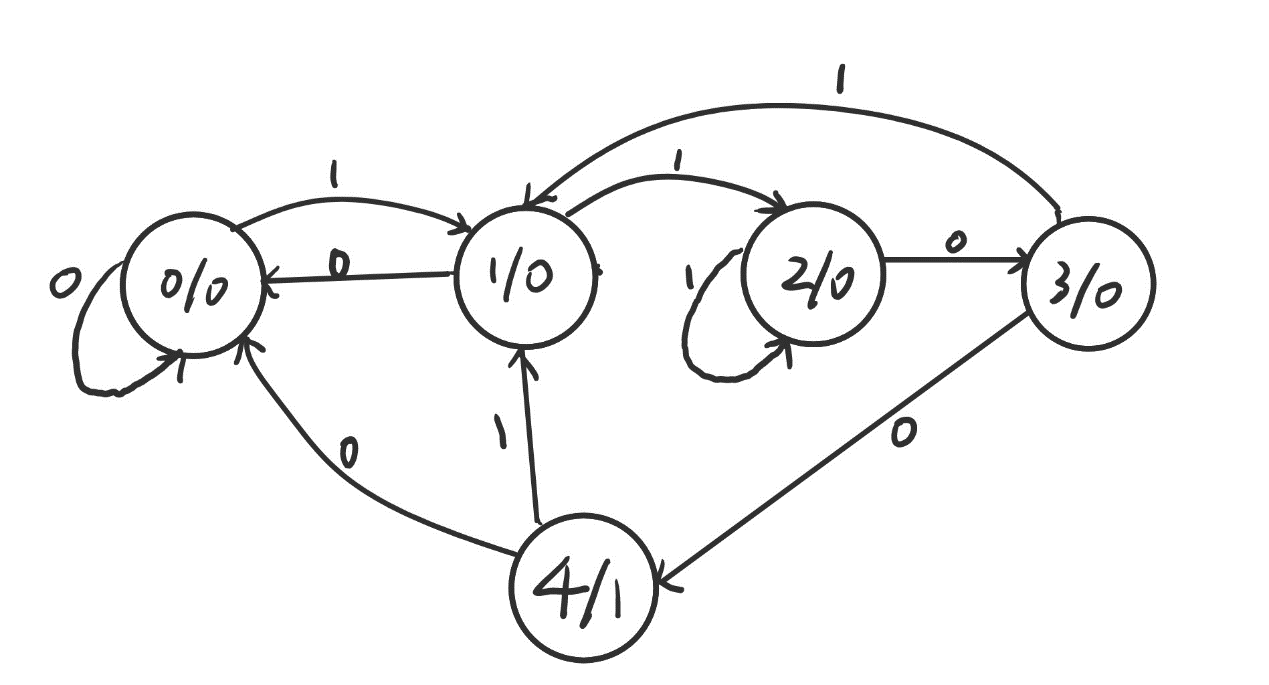


T4.使用有限状态机设计一个序列检测电路，并进行计数，当检测到输入序列为“1100”时，计数器加一，用一个数码管显示当前状态编码，一个数码管显示检测到目标序列的个数，用 4 个数码管显示最近输入的 4 个数值，用 sw[0]进行数据的串行输入，按键每按下一次将输入一次开关状态，时钟采用板载的 100MHz 时钟。 要求画出状态跳转图，并在 FPGA 开发板上实现电路，例如当输入 “0011001110011”时，目标序列个数应为2，最近输入数值显示“0011”， 状态机编码与具体实现有关。

[解]：

|  |  |  |  |
| --- | --- | --- | --- |
| 当前状态 | 输入0 | 输入1 | 输出 |
| 0 | 0 | 1 | 0 |
| 1 | 0 | 11 | 0 |
| 11 | 110 | 11 | 0 |
| 110 | 1100 | 1 | 0 |
| 1100 | 0 | 1 | 1 |

对上表状态进行编码：自上而下状态分别为0、1、2、3、4

画出其对应的状态图：

再写出Verilog设计文件：

**module test(**

**input clk,sw,button,**

**output reg [3:0] dat,**

**output reg [2:0] sel**

**);**

**reg button\_r1,button\_r2;**

**always@(posedge clk) button\_r1 <= button;**

**always@(posedge clk) button\_r2 <= button\_r1;**

**assign button\_pulse = button\_r1 & (~button\_r2);**

**reg [2:0] cur\_state,next\_state;**

**always@(\*)**

**if(sw==1)**

**begin**

**case(cur\_state)**

**3'b000: next\_state=3'b001;**

**3'b001: next\_state=3'b010;**

**3'b010: next\_state=3'b010;**

**3'b011: next\_state=3'b001;**

**3'b100: next\_state=3'b001;**

**default: next\_state=3'b000;**

**endcase**

**end**

**else**

**begin**

**if(cur\_state==3'b011) next\_state=3'b100;**

**else if(cur\_state==3'b010) next\_state=3'b011;**

**else next\_state=3'b000;**

**end**

**reg [3:0] four\_num;**

**always@(posedge clk)**

**begin**

**if(button\_pulse)**

**begin**

**cur\_state<=next\_state;**

**four\_num<=(four\_num<<1);**

**four\_num[0]<=sw;**

**end**

**end**

**reg r1,r2;**

**always@(posedge clk) r1<=cur\_state[2];**

**always@(posedge clk) r2<=r1;**

**assign pulse=r1&(~r2);**

**reg [3:0] all\_num;**

**always@(posedge clk)**

**if(pulse) all\_num=all\_num+1;**

**reg [4:0] temp;**

**always@(posedge clk)**

**begin**

**if(temp==5'b10111) temp=0;**

**else temp=temp+1;**

**end**

**always@(posedge clk)**

**begin**

**sel=temp[4:2];**

**case(sel)**

**3'b000: dat<=cur\_state;**

**3'b001: dat<=all\_num;**

**3'b010: dat<=four\_num[0];**

**3'b011: dat<=four\_num[1];**

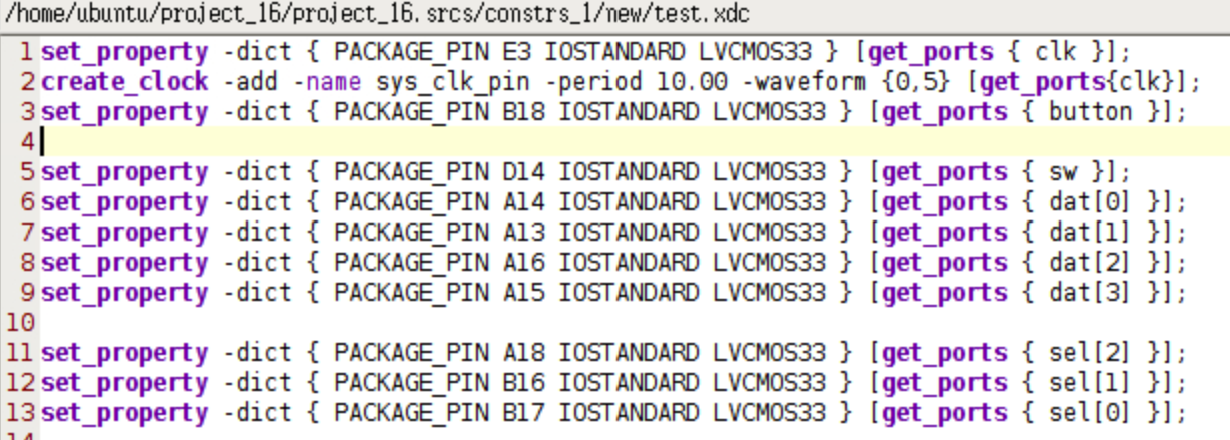
**3'b100: dat<=four\_num[2];**

**3'b101: dat<=four\_num[3];**

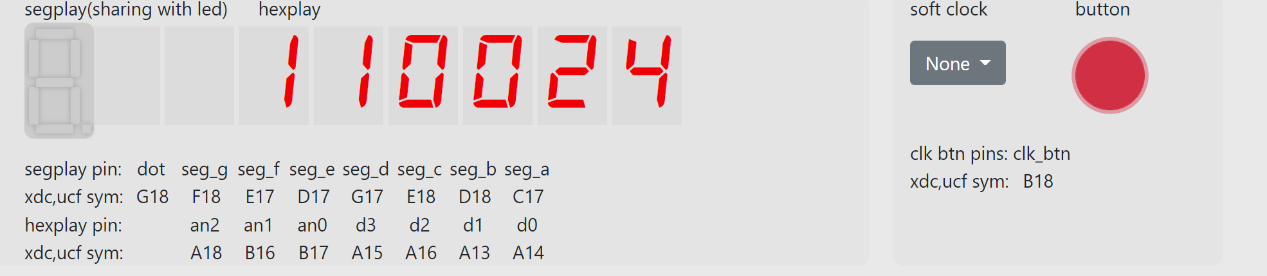
**endcase**

**end**

**endmodule**

其次再给出XDC约束文件：

将生成的bit文件烧写在fpga平台上，如下是运行过程中一个截图：



【总结与思考】

收获：我进一步熟悉 FPGA 开发的整体流程、掌握几种常见的信号处理技巧、并且掌握了有限状态机的设计方法、还能够使用有限状态机设计功能电路。

实验难度：适中

实验内容：稍多