双面打印

组号 43

数字逻辑实验报告

实验 4

学　期	2022-2023 学年第 1 学期		实验日期	2023/2/21	
学　院	信息学部		专　业	计算机科学与技术（实验班）	
班　级	210710	学　号	21071003	姓　名	高立扬
组　号	43	学　号	21071004	姓　名	石昊阳

评　阅　内　容				
任务一	任务二	总结	格式	成　绩

题　目	实验 4：状态机电路设计

一、实验目的

1.通过本实验掌握典型状态机电路的功能和特点;掌握摩尔型和米利型状态机的基本分析方法和设计方法;掌握使用硬件描述语言设计状态机电路的方法;巩固和加深对课程基本理论知识的理解。

2.通过交通灯、流水灯、序列检测器等电路的设计与测试，掌握状态机电路的分析方法和设计方法;学会使用 Verilog HDL 设计状态机电路

二、任务一设计与实现

1.　要求

(1) 输出用 LED 显示，显示模式为 LED 灯从左至右或从右至左轮流点亮，也可自定义显示模式。根据实现模式，画出状态图。

(2) 用 Verilog 编写状态机程序

2.　设计思路

任务一因为分频器只需要分出 1hz 信号，所以对原先的分频器代码进行简化，只输出 1hz 信号。流水灯根据书上的提示进行书写，我们流水灯设置的是，初始状态全灭，然后由右向左流水。

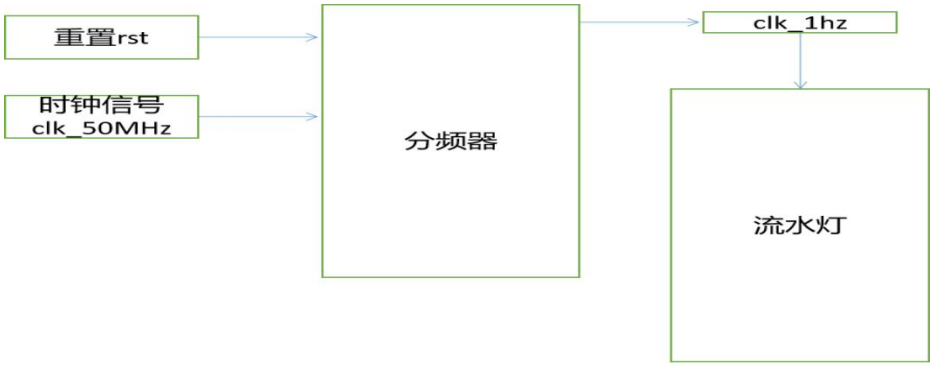

图 1.设计思路

3. 详细设计

表 1.详细设计

输入	输出
reset	LED 灯
0	1111_1111_1111_1111
1	1111_1111_1111_1111
1	1111_1111_1111_1110
1	1111_1111_1111_1101
1	1111_1111_1111_1011
1	1111_1111_1111_0111
1	1111_1111_1110_1111
1	1111_1111_1101_1111
1	1111_1111_1011_1111
1	1111_1111_0111_1111
1	1111_1110_1111_1111
1	1111_1101_1111_1111
1	1111_1011_1111_1111
1	1111_0111_1111_1111
1	1110_1111_1111_1111
1	1101_1111_1111_1111
1	1011_1111_1111_1111
1	0111_1111_1111_1111

```
module flowing_water_light(reset,out,clk);
input reset;
input clk;
output reg[15:0]out;
reg [4:0]state;
parameter st0=0,st1=1,st2=2,st3=3,st4=4,st5=5,st6=6,st7=7,st8=8 ,st9=9, st10=10, st11=11,st12=12,st13=13,st14=14,st15=15,st16=16
always @(state) begin
case (state)
    st0:out=16'b1111_1111_1111_1111;
    st1:out=16'b1111_1111_1111_1110;
    st2:out=16'b1111_1111_1111_1101;
    st3:out=16'b1111_1111_1111_1011;
    st4:out=16'b1111_1111_1111_0111;
    st5:out=16'b1111_1111_1110_1111;
    st6:out=16'b1111_1111_1101_1111;
    st7:out=16'b1111_1111_1011_1111;
    st8:out=16'b1111_1111_0111_1111;
    st9:out=16'b1111_1110_1111_1111;
    st10:out=16'b1111_1101_1111_1111;
    st11:out=16'b1111_1011_1111_1111;
    st12:out=16'b1111_0111_1111_1111;
    st13:out=16'b1110_1111_1111_1111;
    st14:out=16'b1101_1111_1111_1111;
    st15:out=16'b1011_1111_1111_1111;
    st16:out=16'b0111_1111_1111_1111;
    default: out=16'b1111_1111_1111_1111;
  endcase
end

reg clk_1hz;
reg [31:0]cnt;
always @ (posedge clk)
  if(cnt==32'd2/*5_000_000*/)
    begin
```

```
    clk_1hz<=~clk_1hz;
    cnt<='b0;
    end
  else
    begin
    cnt<=cnt+'b1;
    end
 always @(posedge clk_1hz)
begin
  if(!reset)
    state=st0;
  else
    case (state)
      st0:state<=st1;
      st1:state<=st2;
      st2:state<=st3;
      st3:state<=st4;
      st4:state<=st5;
      st5:state<=st6;
      st6:state<=st7;
      st7:state<=st8;
      st8:state<=st9;
      st9:state<=st10;
      st10:state<=st11;
      st11:state<=st12;
      st12:state<=st13;
      st13:state<=st14;
      st14:state<=st15;
      st15:state<=st16;
      st16:state<=st0;
      default state<=st0;
    endcase
end
endmodule
```

图 2.流水灯

top.v

```
1  module top(clk_50mhz,rst,out);
2     input clk_50mhz, rst;
3     wire clk_1hz;
4     output [15:0] out;
5
6     frequency_divider(.clk_50mhz(clk_50mhz), .rst(rst), .clk_1hz(clk_1hz));
7     flowing_water_light(.clk(clk_1hz), .reset(rst), .out(out));
8  endmodule
```

图 3.顶层文件

4. 仿真验证

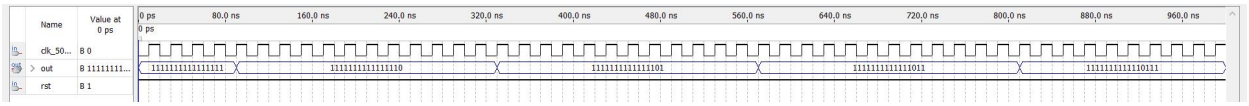

图 4.波形图

显而易见，流水灯正常工作。

5. 引脚分配

表 2.引脚分配

端口名称	输入端	
	时钟信号 clk	reset
引脚编号	T1	SW9
平台端口	T1	AB17

表 2.续表

输出端							
LED							
out[15]	out[14]	out[13]	out[12]	out[11]	out[10]	out[9]	out[8]
B9	B8	B7	E14	C15	F11	C13	E11
LED16	LED15	LED14	LED13	LED12	LED11	LED10	LED9

表 2.续表

输出端							
LED							
out[7]	out[6]	out[5]	out[4]	out[3]	out[2]	out[1]	out[0]
T17	R16	Y17	W15	W13	V15	V12	U12
LED8	LED7	LED6	LED5	LED4	LED3	LED2	LED1

Node Name	Direction	Location	I/O Bank	REF Grou	ter Locati	O Standa	Reserved	ent Strer	Slew Rate	erential F
clk_50mhz	Input	PIN_T1	2	B2_N0	PIN_T1	2.5 ...ult)		8mA...lt)		
out[15]	Output	PIN_B9	8	B8_N0	PIN_B9	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[14]	Output	PIN_B8	8	B8_N0	PIN_B8	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[13]	Output	PIN_B7	8	B8_N0	PIN_B7	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[12]	Output	PIN_E14	7	B7_N1	PIN_E14	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[11]	Output	PIN_C15	7	B7_N1	PIN_C15	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[10]	Output	PIN_F11	7	B7_N1	PIN_F11	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[9]	Output	PIN_C13	7	B7_N1	PIN_C13	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[8]	Output	PIN_E11	7	B7_N1	PIN_E11	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[7]	Output	PIN_T17	5	B5_N1	PIN_T17	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[6]	Output	PIN_R16	4	B4_N0	PIN_R16	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[5]	Output	PIN_Y17	4	B4_N0	PIN_Y17	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[4]	Output	PIN_W15	4	B4_N0	PIN_W15	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[3]	Output	PIN_W13	4	B4_N1	PIN_W13	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[2]	Output	PIN_V15	4	B4_N0	PIN_V15	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[1]	Output	PIN_V12	4	B4_N1	PIN_V12	2.5 ...ult)		8mA...lt)	2 (d...ult)	
out[0]	Output	PIN_U12	4	B4_N1	PIN_U12	2.5 ...ult)		8mA...lt)	2 (d...ult)	
rst	Input	PIN_AB17	4	B4_N0	PIN_AB17	2.5 ...ult)		8mA...lt)		

6. 实验现象

程序开始执行后，工作台上的 LED 灯从左到右间隔一秒依次闪烁。在拨动 rst 开关后，程序重新执行，从第一个 LED 灯开始闪烁。本次实验符合所有输出预期。

三、任务二设计与实现

1. 要求

(1) 设计一个 1010 序列检测器(不考虑序列重叠)或一个 8 位序列的序列检测器，序列为同组两位同学学号末两位相加，如相加后低于 16，需在和的基础上加 30(检测序列为结果对应的 8421BCD 码，如结果为 34，检测序列为 00110100)。

(2) 采用 Verilog 实现状态机。

2. 设计思路

任务二采用任务一的分频器，时钟信号输入到数据发生器，发生器产生的单个编码传入到检测器检测序列。检测器采用 Moore 型。

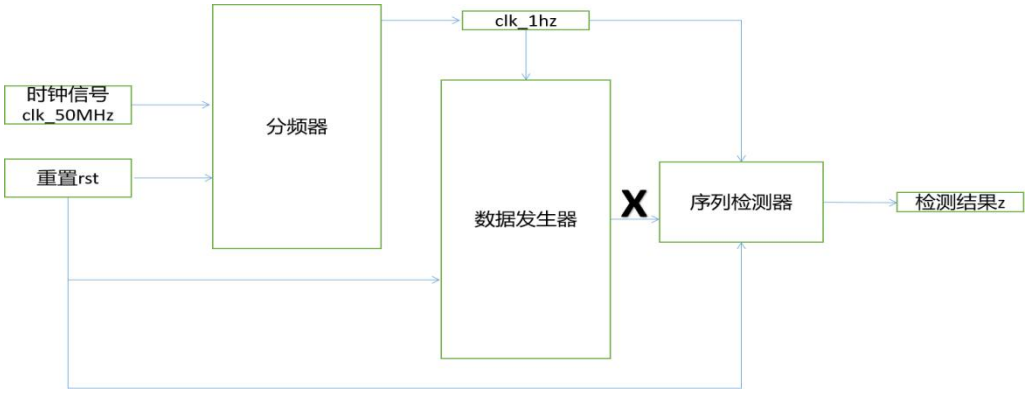

图 5.设计思路

3. 详细设计

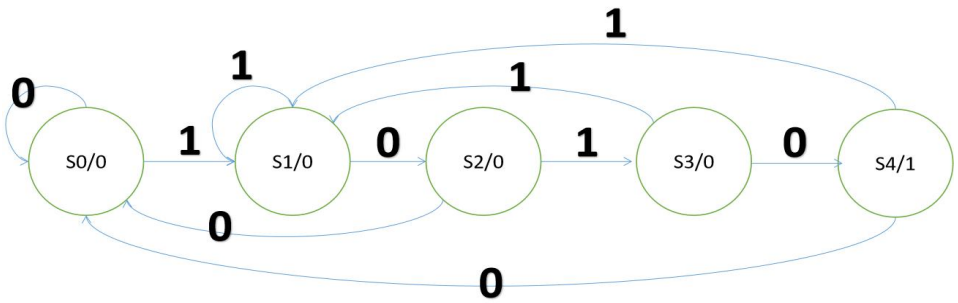

图 6.状态图

```
1  module data_generate (clk,clr,dout);
2  input clk,clr;
3  output dout;
4  reg [7:0] data;
5  reg dout;
6  always@(posedge clk)
7    begin
8    if(!clr)
9      begin
10     dout<=0;
11     data<=8'b00010100;
12     end
13   else
14     begin
15     dout<=data[7];
16     data<={data[6:0],data[7]};
17     end
18   end
19 endmodule
```

图 7.数据发生器

```
1  module seqdet(clk,x,reset,z,c_state,n_state);
2  input clk,x,reset;
3  output reg z;
4  output reg [3:0] c_state,n_state;
5  parameter S0=0,S1=1,S2=2,S3=3,S4=4;
6
7   always @(posedge clk)
8     begin
9        if(c_state==S4) z=1'b1;
10          else z=1'b0;
11
12    end
13
14
15  always @(c_state,x)
16    begin
17     case(c_state)
18       S0:if(x) n_state<=S1; else n_state<=S0;
19       S1:if(x) n_state<=S1; else n_state<=S2;
20       S2:if(x) n_state<=S3; else n_state<=S0;
21       S3:if(!x) n_state<=S4; else n_state<=S1;
22       S4:if(x) n_state<=S1; else n_state<=S0;
23       default: n_state<=S0;
24  endcase
25  end
26   always @(posedge clk)
27  if(!reset) c_state<=S0;
28  else c_state<=n_state;
29   endmodule
```

图 8.序列检测器

```
module frequency_divider(clk_50mhz,rst,clk_1hz);
input clk_50mhz,rst;
output clk_1hz;
reg clk_1hz;
reg [31:0]cnt1;
//parameter A=50000000;
parameter A = 2;
always@(posedge clk_50mhz)
begin
if(!rst)
  begin
    cnt1<=1'b0;
    clk_1hz<=1'b0;
  end
else
  if(cnt1<A/2-1)
    cnt1<=cnt1+1'b1;
  else
    begin
      cnt1<=1'b0;
      clk_1hz<=~clk_1hz;
    end
end
endmodule
```

图 9.分频器

top.v

```
module top(clk,clr,reset,rst,z,c_state,n_state);
    input clk,clr,reset,rst;
    output z;
    output [3:0] c_state,n_state;
    wire clk_1hz,dout;

    frequency_divider(.clk_50mhz(clk),.rst(rst),.clk_1hz(clk_1hz));
    data_generate(.clk(clk_1hz),.clr(clr),.dout(dout));
    seqdet(.clk(clk_1hz),.x(dout),.reset(reset),.z(z),.c_state(c_state),.n_state(n_state));

endmodule
```

图 10.顶层文件

4. 仿真验证

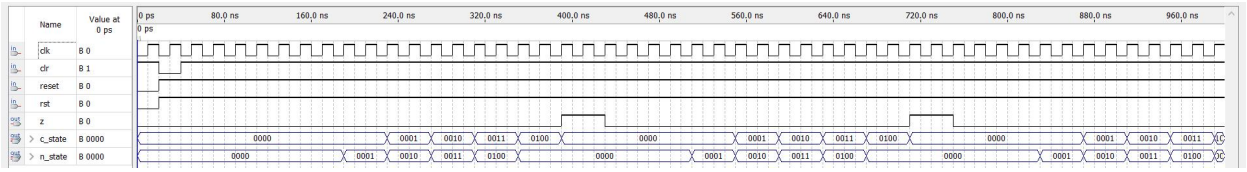

图 11.波形图

可见 z 在状态机进入最后状态的时候变为 1 了，此时序列也正是检测到了 1010.

5. 引脚分配

表 3.引脚分配

端口名称	输入端				输出端
	rst	clr	clk	reset	z
引脚编号	C8	E7	T1	F8	U12
平台端口	SW6	SW7	T1	SW8	LED1

Node Name	Direction	Location	I/O Bank	REF Grou	ter Locati	O Standa	Reserved	ent Strer	Slew Rate	erential I
clk	Input	PIN_T1	2	B2_N0	PIN_T1	2.5 ...ult)		8mA...lt)		
clr	Input	PIN_C8	8	B8_N0	PIN_E7	2.5 ...ult)		8mA...lt)		
reset	Input	PIN_F8	8	B8_N1	PIN_R18	2.5 ...ult)		8mA...lt)		
rst	Input	PIN_E7	8	B8_N1	PIN_V21	2.5 ...ult)		8mA...lt)		
z	Output	PIN_U12	4	B4_N1	PIN_U12	2.5 ...ult)		8mA...lt)	2 (d...ult)	

6. 实验现象

先将 Reset 开关与 rst 开关置 1，将 clr 开关先置 0，再置 1，发现 LED 灯每隔 8 秒闪烁一次。将 Reset 开关或 rst 开关置 0 再置 1 后，LED 灯闪烁间隔时间重置。本次实验符合所有输出预期。

四、扩展实验

1.设计思路

我们希望设计一个用户可以自行输入希望序列检测器检测的数据，于是我们在任务二的基础上增加了一个模块以实现该功能。

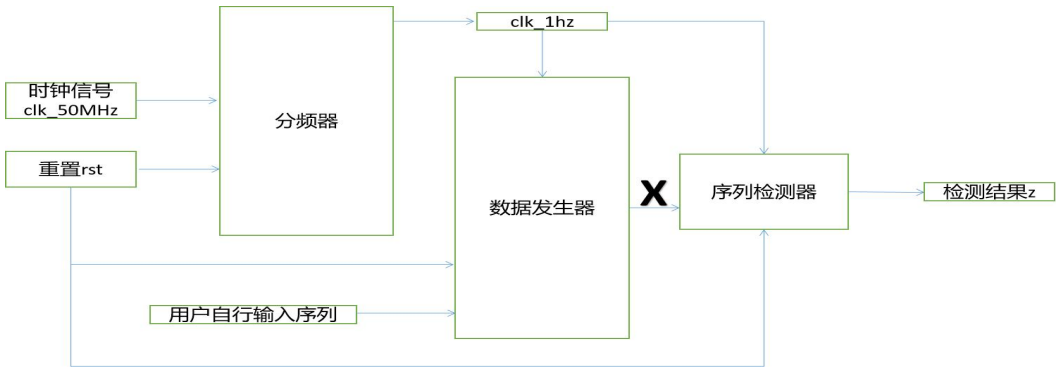

图 12.设计思路

2.详细设计

```
module data_generate (clk,clr,dout,indata);
input clk,clr;
input [7:0] indata;
output dout;
reg [7:0] data;
reg dout;
always@(posedge clk)
  begin
  if(!clr)
    begin
    dout<=0;
    data<=indata;
    end
  else
    begin
    dout<=data[7];
    data<={data[6:0],data[7]};
    end
  end
endmodule
```

图 13.数据发生器

```
module seqdet(clk,x,reset,z,c_state,n_state);
input clk,x,reset;
output reg z;
output reg [3:0] c_state,n_state;
parameter S0=0,S1=1,S2=2,S3=3,S4=4;

 always @(posedge clk)
   begin
      if(c_state==S4) z=1'b1;
         else z=1'b0;

   end


 always @(c_state,x)
   begin
    case(c_state)
      S0:if(x) n_state<=S1; else n_state<=S0;
      S1:if(x) n_state<=S1; else n_state<=S2;
      S2:if(x) n_state<=S3; else n_state<=S0;
      S3:if(!x) n_state<=S4; else n_state<=S1;
      S4:if(x) n_state<=S1; else n_state<=S0;
      default: n_state<=S0;
 endcase
 end
 always @(posedge clk)
if(!reset) c_state<=S0;
else c_state<=n_state;
 endmodule
```

图 14.序列检测器

```
module expand_task(clk,clr,reset,rst,z,c_state,n_state,indata);
   input clk,clr,reset,rst;
   input [7:0] indata;
   output z;
   output [3:0] c_state,n_state;
   wire clk_1hz,dout;

   frequency_divider(.clk_50mhz(clk),.rst(rst),.clk_1hz(clk_1hz));
   data_generate(.clk(clk_1hz),.clr(clr),.dout(dout),.indata(indata));
   seqdet(.clk(clk_1hz),.x(dout),.reset(reset),.z(z),.c_state(c_state),.n_state(n_state));

endmodule
```

图 15.顶层文件

3. 仿真验证

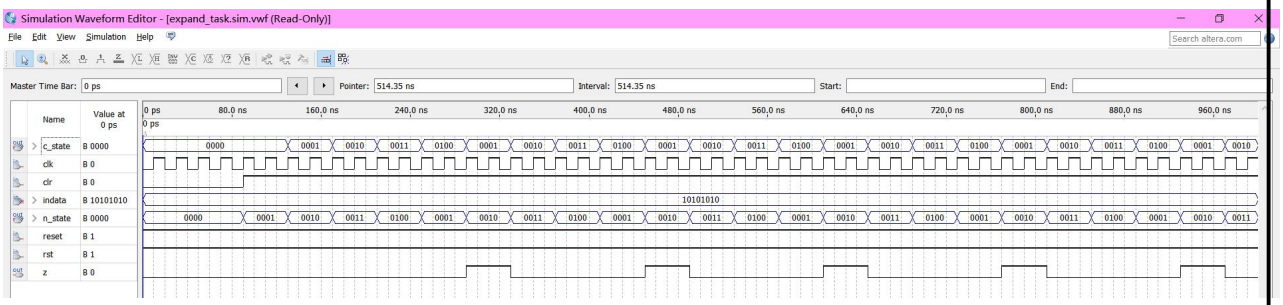

图 16.仿真验证

z 的频率变高，因为序列被设置为了 10101010。这个思路其实从另一种角度来说，构造出了新的分频器，如果控制 1010 出现的频率，分频器出的频率也可以随之改变。

4. 引脚分配

表 4.引脚分配

端口名称	输入端							
	indata[7]	indata[6]	indata[5]	indata[4]	indata[3]	indata[2]	indata[1]	indata[0]
引脚编号	F8	E7	C8	D6	V13	AA15	M20	N18
平台端口	SW8	SW7	SW6	SW5	SW4	SW3	SW2	SW1

端口名称	输入端				输出端
	rst	clr	clk	reset	z
引脚编号	E6	F7	T1	A3	U12

平台端口	SW14	SW15	T1	SW16	LED1

Node Name	Direction	Location	I/O Bank	REF Grou	ter Locati	O Standa	Reserved	ent Strer	Slew Rate	erential I
c_state[3]	Output				PIN_T16	2.5 ...ult)		8mA...lt)	2 (d...ult)	
c_state[2]	Output				PIN_G10	2.5 ...ult)		8mA...lt)	2 (d...ult)	
c_state[1]	Output				PIN_F10	2.5 ...ult)		8mA...lt)	2 (d...ult)	
c_state[0]	Output				PIN_D8	2.5 ...ult)		8mA...lt)	2 (d...ult)	
clk	Input	PIN_T1	2	B2_N0	PIN_T1	2.5 ...ult)		8mA...lt)		
clr	Input	PIN_F7	8	B8_N1	PIN_F7	2.5 ...ult)		8mA...lt)		
indata[7]	Input	PIN_F8	8	B8_N1	PIN_F8	2.5 ...ult)		8mA...lt)		
indata[6]	Input	PIN_E7	8	B8_N1	PIN_E7	2.5 ...ult)		8mA...lt)		
indata[5]	Input	PIN_C8	8	B8_N0	PIN_C8	2.5 ...ult)		8mA...lt)		
indata[4]	Input	PIN_D6	8	B8_N1	PIN_D6	2.5 ...ult)		8mA...lt)		
indata[3]	Input	PIN_V13	4	B4_N1	PIN_V13	2.5 ...ult)		8mA...lt)		
indata[2]	Input	PIN_AA15	4	B4_N1	PIN_AA15	2.5 ...ult)		8mA...lt)		
indata[1]	Input	PIN_M20	5	B5_N0	PIN_M20	2.5 ...ult)		8mA...lt)		
indata[0]	Input	PIN_N18	5	B5_N0	PIN_N18	2.5 ...ult)		8mA...lt)		
n_state[3]	Output				PIN_W7	2.5 ...ult)		8mA...lt)	2 (d...ult)	
n_state[2]	Output				PIN_E8	2.5 ...ult)		8mA...lt)	2 (d...ult)	
n_state[1]	Output				PIN_A5	2.5 ...ult)		8mA...lt)	2 (d...ult)	
n_state[0]	Output				PIN_B5	2.5 ...ult)		8mA...lt)	2 (d...ult)	
reset	Input	PIN_A3	8	B8_N1	PIN_A3	2.5 ...ult)		8mA...lt)		
rst	Input	PIN_E6	8	B8_N1	PIN_E6	2.5 ...ult)		8mA...lt)		
z	Output	PIN_U12	4	B4_N1	PIN_U12	2.5 ...ult)		8mA...lt)	2 (d...ult)	

5. 实验现象

我们将序列设置为 10101010，先将 Reset 开关与 rst 开关置 1，将 clr 开关先置 0，再置 1，发现 LED 灯每隔 4 秒闪烁一次。将 Reset 开关或 rst 开关置 0 再置 1 后，LED 灯闪烁间隔时间重置。本次实验符合所有输出预期。

五、总结

我们小组由于疫情原因，本实验没能在上学期做完，而是在本学期去实验台上完成的实验。因为时间比较紧张，我们小组也出现了较多的低级错误。

①实验一中，流水灯模块是由石昊阳完成，而顶层文件则是由高立扬完成。石昊阳已经在流水灯中写入了分频器代码，而高立扬在写顶层文件的时候又加入了分屏器模块，因此是流水灯闪烁的时间间隔缩短了一倍。我们在之后的合作中会更加积极的向对方讲解自己的想法与代码，这大大的提高了我们的团队合作能力。

②实验一中，我们发现后八个 LED 灯不能闪烁。通过查找资料得知后八个 LED 灯需要在模式二下才能正常工作。

③实验二中，我们发现实验台上的现象与波形图不一致。老师告诉我们可以将顶端中的一些变量输出到实验台上，以便查找错误。我们用这个方法顺利的找到并修改了错误。

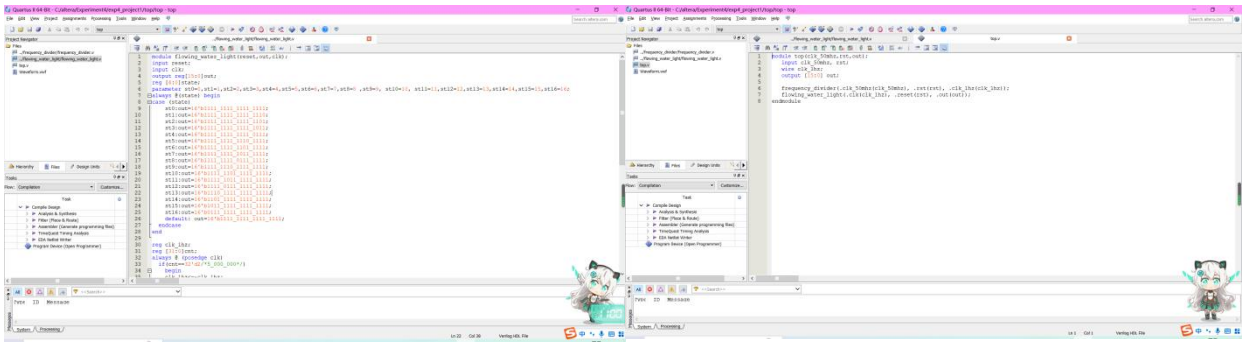

图 1-2.流水灯代码和顶层文件代码

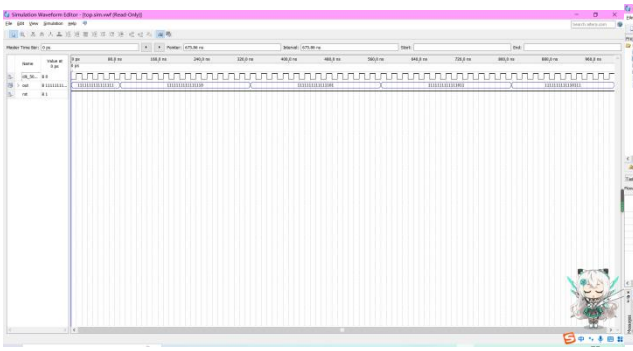

图 3.流水灯波形图

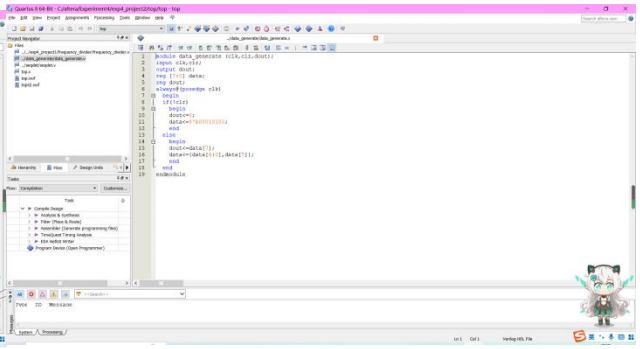

图 4.状态机序列生成器

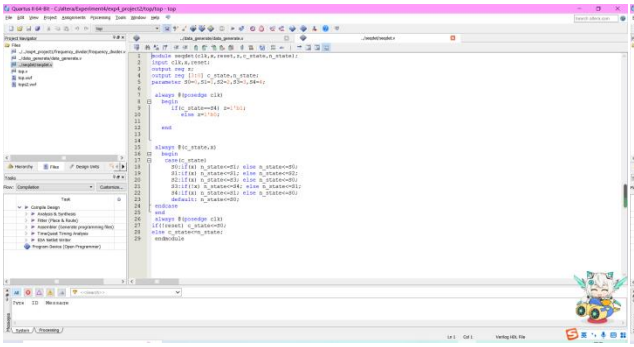

图 5.状态机本体

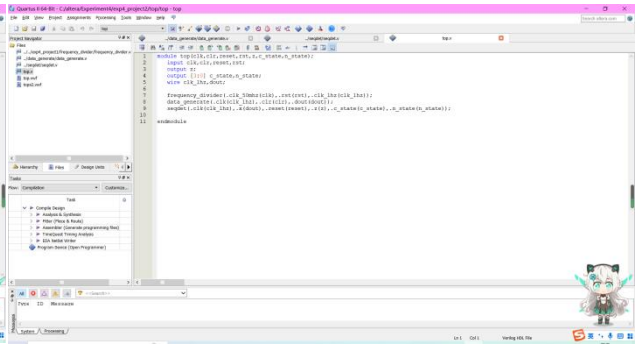

图 6.任务二顶层文件

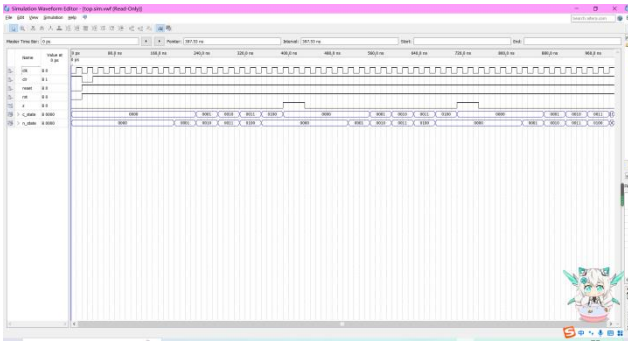

图 7.任务二波形图

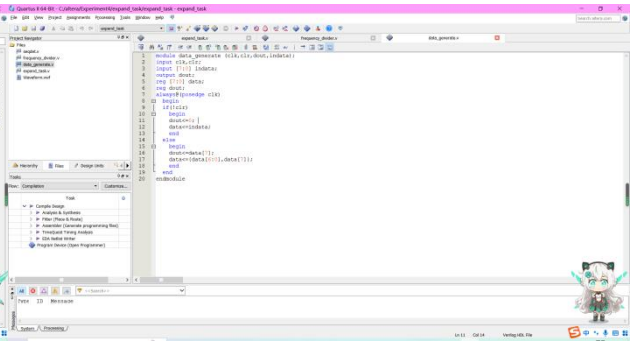

图 8.扩展任务序列生成器改版

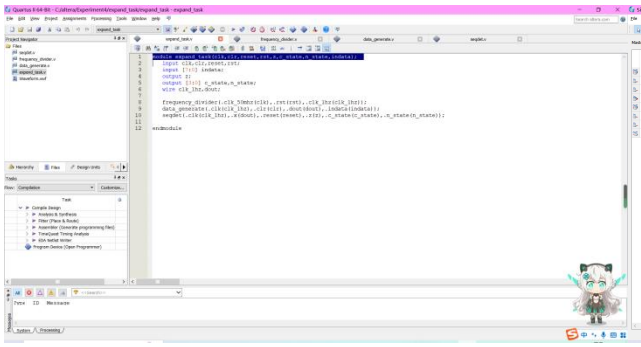

图 9.任务二波形图

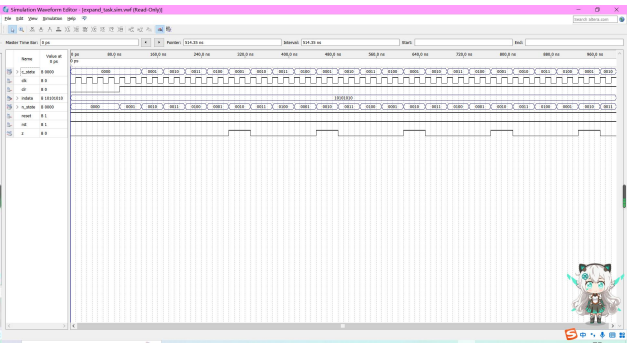

图 10.扩展任务序列生成器改版

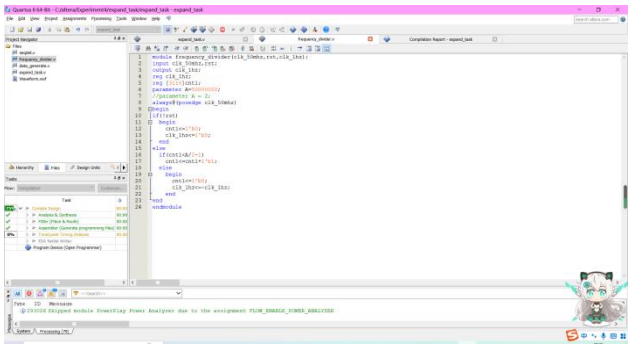

图 11.公用分频器