实验报告

实验名称：Lab2_数据通路与状态机

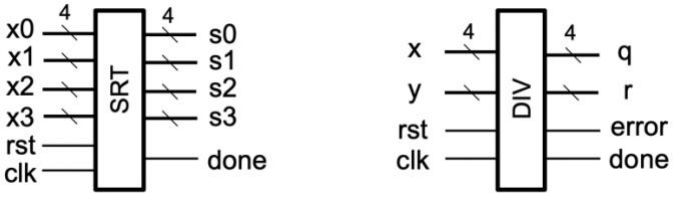
学生姓名：田宏宇 学号：PB17111573

实验日期：2019 年 3 月 29 日

一、实验内容

1. 排序：s0 ~ s3是x0 ~ x3的排序结果
2. 除法运算：x / y = q … r

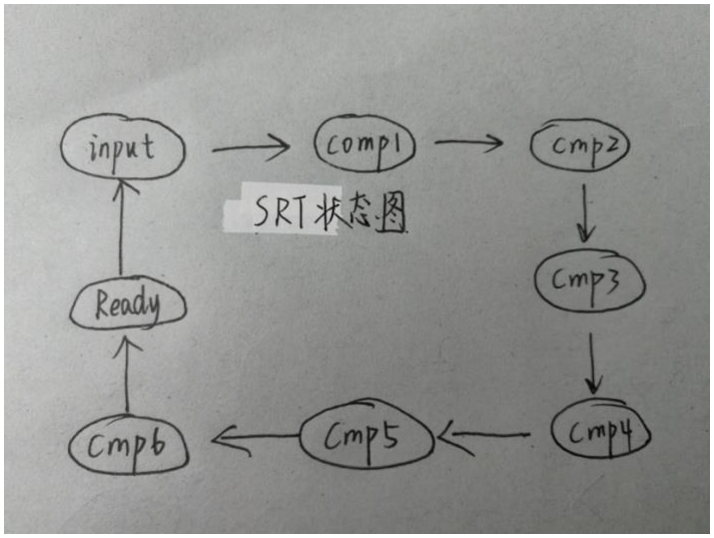
（以上均考虑无符号数）



二、实验设计

1.排序状态机

四组数据采用冒泡排序，至多需进行 6 次两两比较并交换



2.除法运算

(利用减法)

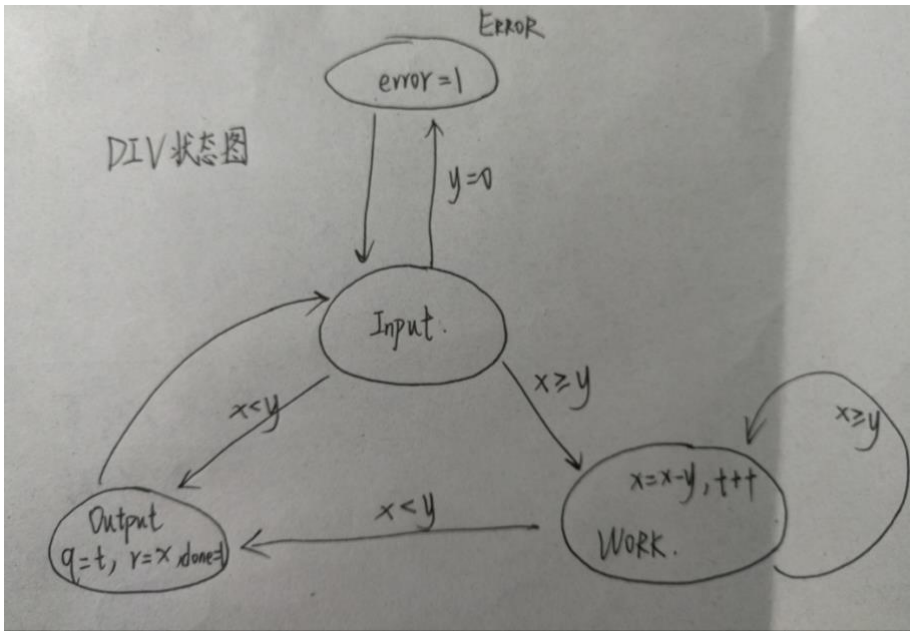
as for x and y:

if y=0, error=0;

else if (x<y) output: q=0, r=x;

else if (x>=y) do x=x-y until x<y,

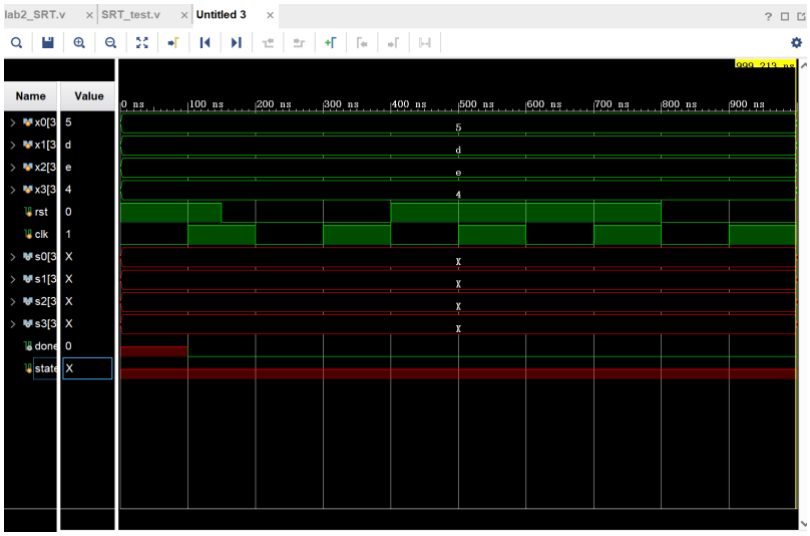
then output: q=t (times of subtraction), r=x;



三、实验过程

1.排序

SRT 第一次仿真失败，s0-s3 未被赋值：

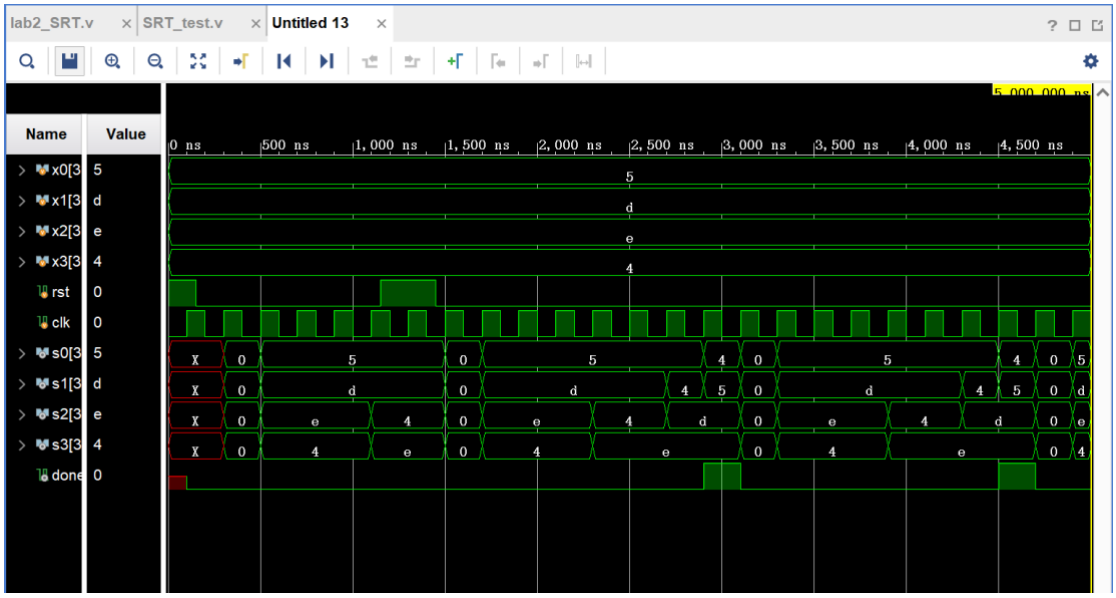


经检查发现错误原因：

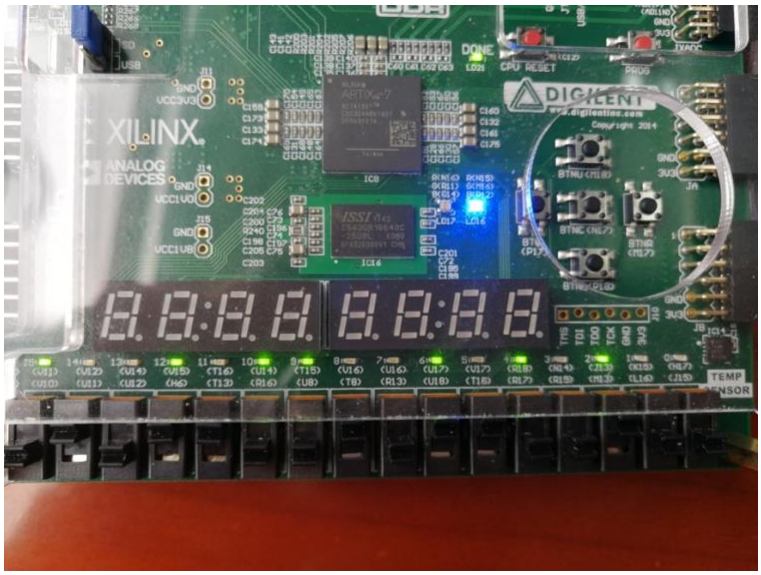
给 state 的选择值赋初值时用 IN=2'h01 种方式，忽略了 state 是四位二进制数，改为 4'h01 后解决了该问题：

```
parameter READY=4'h00;
parameter IN=4'h01;
parameter CMP1=4'h02;
parameter CMP2=4'h03;
parameter CMP3=4'h04;
parameter CMP4=4'h05;
parameter CMP5=4'h06;
parameter CMP6=4'h07;
```

得到正确仿真如下

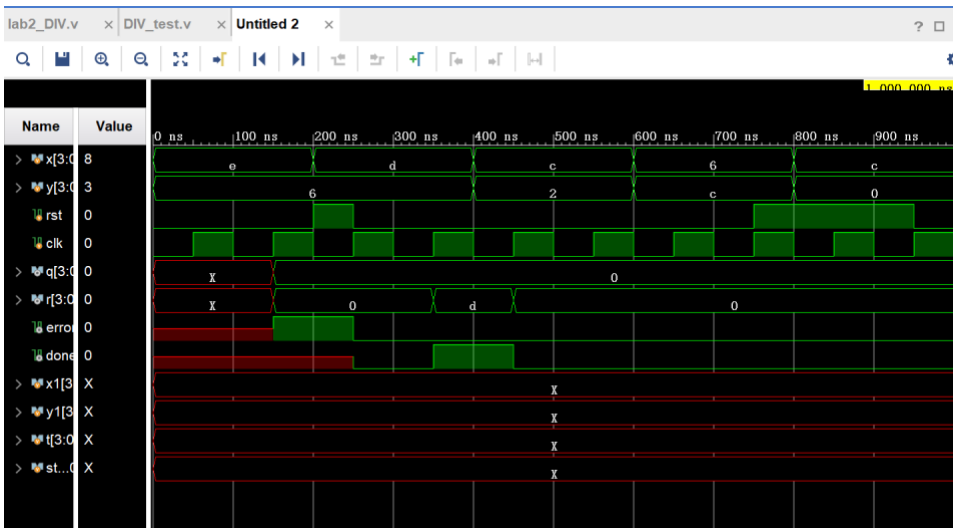


下载结果如图，BTNC 控制 clk，蓝灯亮起代表 done



2.除法运算

第一次仿真失败，输出结果很乱

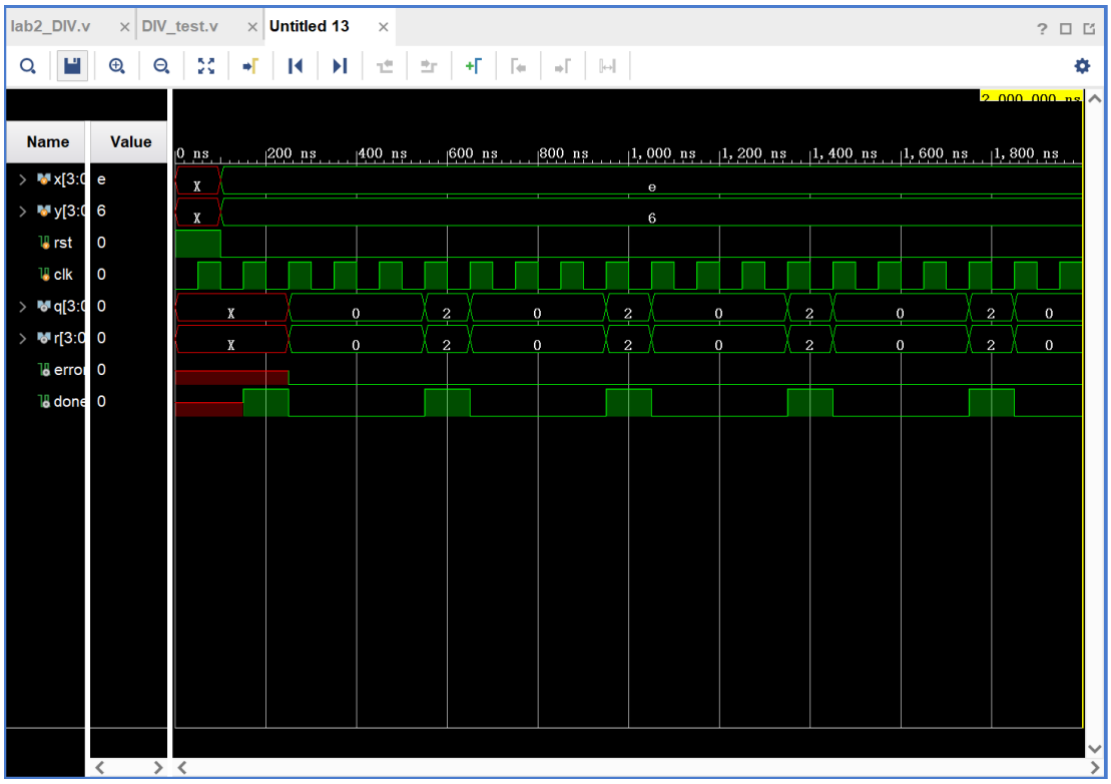


经过研究，发现问题：

1.某组数据运算至 done 需要周期数较多，所以在一组还未完成运算我仿真设置的下一组数据提前到来了，导致输出出错。

2.在仿真时非输入输出变量不会在仿真中表现出数据变化，之前没有认识到这一点，导致出现问题时第一时间以为是 state 等值未赋进去，找了半天也没有解决。

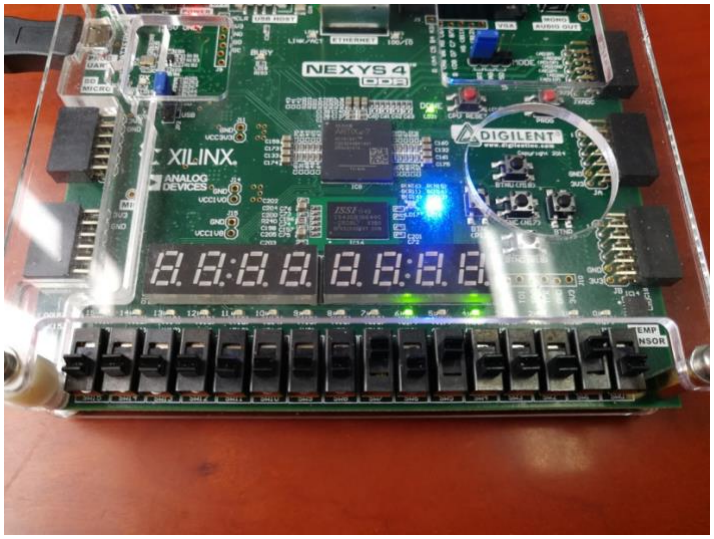
解决后：在 test 文件中改为只有一组除数与被除数，仿真如下：



下载结果如下：

红灯表示 y=0 时 ERROR

蓝灯为 done



四、实验总结

在这次实验中，我学会了四组数据排序和除法器的设计，对 verilog 的数据表示、case 语句和时序逻辑等有了进一步的理解，在实验中解决仿真时遇到的问题提高了我的 debug 能力。

附

1.SRT 代码如下：

```
23 module SRT(
24     input [3:0] x0,
25     input [3:0] x1,
26     input [3:0] x2,
27     input [3:0] x3,
28     input rst,
29     input clk,
30     output reg [3:0] s0,
31     output reg [3:0] s1,
32     output reg [3:0] s2,
33     output reg [3:0] s3,
34     output reg done
35 );
36 /*reg [3:0] t0;
37 reg [3:0] t1;
38 reg [3:0] t2;
39 reg [3:0] t3;*/
40 reg [3:0] state;
41 parameter READY=4'h00;
42 parameter IN=4'h01;
43 parameter CMP1=4'h02;
44 parameter CMP2=4'h03;
45 parameter CMP3=4'h04;
46 parameter CMP4=4'h05;
47 parameter CMP5=4'h06;
48 parameter CMP6=4'h07;
```

```
always@(posedge clk)
begin
if(rst==1)
    begin
    state<=4'b0;
    done<=0;
    end
else
case(state)
IN:
    begin
    s0<=x0;
    s1<=x1;
    s2<=x2;
    s3<=x3;
    state<=CMP1;
    end
CMP1:
    begin
    sort(s0,s1);
    state<=CMP2;
    end
CMP2:
    begin
    sort(s1,s2);
    state<=CMP3;
    end
```

```
78  CMP3:
79      begin
80      sort(s2,s3);
81      state<=CMP4;
82      end
83  CMP4:
84      begin
85      sort(s0,s1);
86      state<=CMP5;
87      end
88  CMP5:
89      begin
90      sort(s1,s2);
91      state<=CMP6;
92      end
93  CMP6:
94      begin
95      sort(s0,s1);
96      state<=READY;
97      done<=1;
98      end
99  READY:
100     begin
101     s0<=4'b0;
102     s1<=4'b0;
103     s2<=4'b0;
104     s3<=4'b0;
105     state<=IN;
106     done<=0;
107     end
108 default:
```

```
108 default:
109     state<=READY;
110 endcase
111 end
112
113 task sort;
114  inout  [3:0]a;
115  inout  [3:0]b;
116  reg [3:0]temp;
117 if(a>b)
118 begin
119 temp=a;
120 a=b;
121 b=temp;
122 end
123 endtask
124
125 endmodule
```

SRT 仿真代码

```
module SRT_test();
    reg [3:0] x0,x1,x2,x3;
    reg rst;
    reg clk;
    wire [3:0] s0,s1,s2,s3;
    wire done;
    SRT LU (
        .x0(x0),
        .x1(x1),
        .x2(x2),
        .x3(x3),
        .rst(rst),
        .clk(clk),
        .s0(s0),
        .s1(s1),
        .s2(s2),
        .s3(s3),
        .done(done)
    );
    initial clk=0;
    initial rst=1;
    always #100 clk=~clk;
    initial
    begin
    x0=4'b0101;
    x1=4'b1101;
    x2=4'b1110;
    x3=4'b0100;
    end
    initial
    begin
    #150 rst=0;
    #1000 rst=1;
    #300 rst=0;
    end
    endmodule
```

2.DIV 代码如下

```
module DIV(
input [3:0]x,
input [3:0]y,
input  rst,
input  clk,
output reg [3:0]q,
output reg [3:0]r,
output reg error,
output reg done
);
reg [3:0] x1;
reg [3:0] y1;
reg [3:0] t;
reg [3:0] state;

parameter IN=4'b0000;
parameter WO=4'b0001;
parameter ERR=4'b0010;
parameter OUT=4'b0011;

always@(posedge clk)
begin
if(rst==1)
begin
state<=OUT;
end
else
case(state)
IN:
    begin
    t<=4'b0000;
    q<=4'b0000;
    r<=4'b0000;
    done<=0;
    error<=0;
```

```
    if(y==4'b0000)
        state<=ERR;
    else
    if(x>=y)
    begin
        state<=WO;
        x1<=x;
        y1<=y;
    end
    else
    begin
       x1<=x;
       state<=OUT;
    end
    end
ERR:
    begin
    q<=4'b0000;
    r<=4'b0000;
    error<=1;
    state<=IN;
    end
WO:
    begin
    //x1=x1-y1;
    //t=t+4'b0001;
    wo(x1,y1,t);
    if(x1>=y1)
        state<=WO;
    else
        state<=OUT;
    end
OUT:
    begin
    q<=t;
    r<=x1;
    done<=1;
    state<=IN;
    end
default:state<=ERR;
endcase
```

```
default:state<=ERR;
endcase
end

task wo;
inout [3:0]a;
inout [3:0]b;
inout [3:0]c;
reg [3:0] temp;
begin
    temp=a-b;
    a=temp;
    c=c+4'b0001;
end
endtask
endmodule
```

DIV 仿真代码

```
module DIV_test();
reg [3:0]x;
reg [3:0]y;
reg rst;
reg clk;
wire [3:0]q;
wire [3:0]r;
wire error;
wire done;
DIV LO (
    .x(x),
    .y(y),
    .rst(rst),
    .clk(clk),
    .q(q),
    .r(r),
    .error(error),
    .done(done)
);
initial clk=0;
initial rst=0;
initial begin
rst=1;
#100 rst=0;
x=4'h0e;
y=4'h06;/*
#600
rst=1;
#100 rst=0;
x=4'h0d;
y=4'h06;
#600
rst=1;
#100 rst=0;
x=4'h08;
y=4'h03;*/
end
always #50 clk=~clk;
endmodule
```