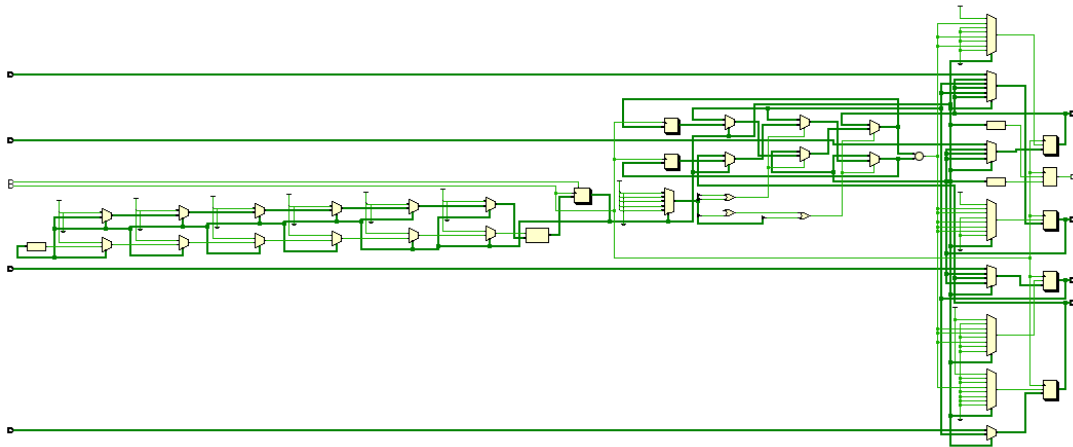


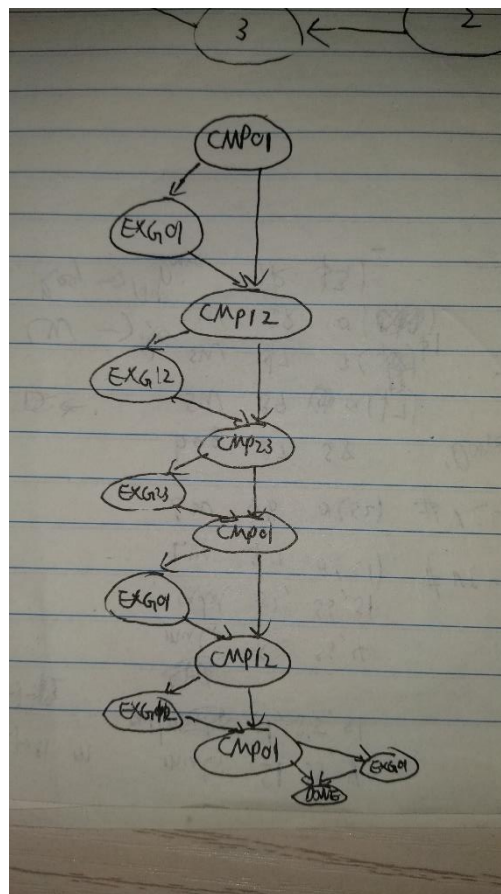
Lab1 运算器与寄存器

1. 设计逻辑

- (1) 四个四位数的排序器：利用三段式状态机通过 6 个状态来完成比较与换位排序，排序算法使用冒泡法。排序从大到小。



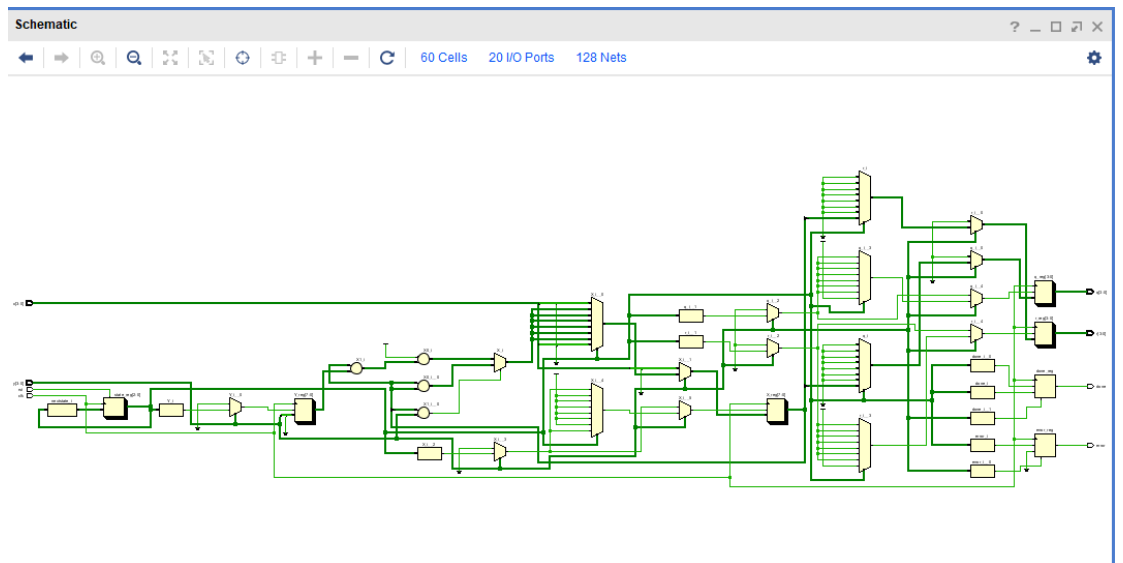
数据通路



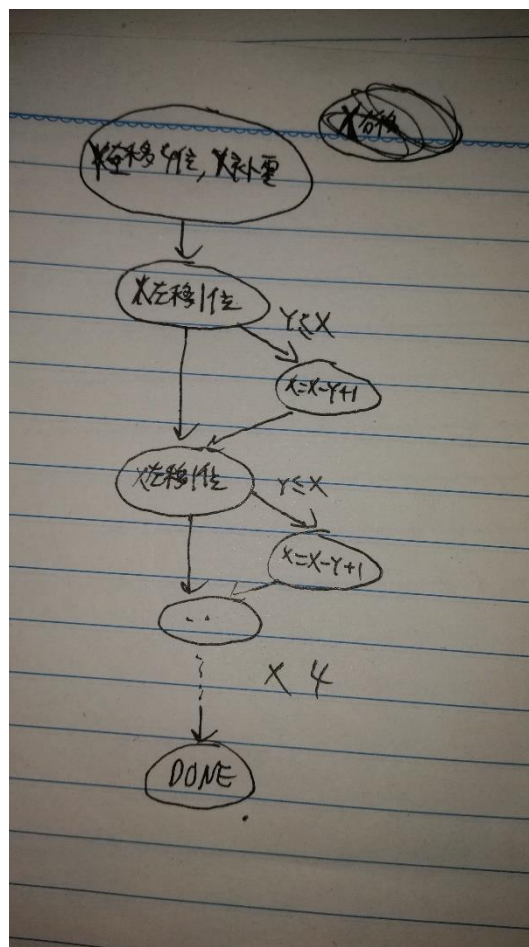
状态图

(2) 除法器：实现两个四位二进制数的除法运算，输出商和余数。ERROR

表示除数为 0，DONE 表示计算完成。



数据通路



状态图

2. 核心代码









































(1). 排序器

```
41  always @(posedge clk or negedge rst)
42  begin
43      if(rst)
44      begin
45          state<=3'b000;
46      end
47      else
48      begin
49          state<=nextstate;
50      end
51  end
52
53  always @(*)
54  begin
55      if(state==0)    nextstate=1;
56      else if(state==1) nextstate=2;
57      else if(state==2) nextstate=3;
58      else if(state==3) nextstate=4;
59      else if(state==4) nextstate=5;
60      else if(state==5) nextstate=6;
61      else if(state==6) nextstate=6;
62  end
63
64  always @(posedge clk)
65  begin
66      if (state==1|state==4|state==6)
67      begin
68          a=s0;
69          b=s1;
70      end
71      else if (state==2|state==5)
72      begin
73          a=s1;
74          b=s2;
75      end
76      else if (state==3)
77      begin
78          a=s2;
79          b=s3;
80      end
81
82      case(state)
83          3'b000:
84          begin
85              done<=0;
86              s0<=x0;
87              s1<=x1;
88              s2<=x2;
89              s3<=x3;
90          end
```

| | | |
|-----|---|----------|
| 91 | ⬇ | 3' b001: |
| 92 | ⬇ | begin |
| 93 | ⬇ | if(a<b) |
| 94 | ⬇ | begin |
| 95 | ⋮ | s0<=s1; |
| 96 | ⋮ | s1<=s0; |
| 97 | ⬇ | end |
| 98 | ⬇ | end |
| 99 | ⬇ | 3' b010: |
| 100 | ⬇ | begin |
| 101 | ⬇ | if(a<b) |
| 102 | ⬇ | begin |
| 103 | ⋮ | s1<=s2; |
| 104 | ⋮ | s2<=s1; |
| 105 | ⬇ | end |
| 106 | ⬇ | end |
| 107 | ⬇ | 3' b011: |
| 108 | ⬇ | begin |
| 109 | ⬇ | if(a<b) |
| 110 | ⬇ | begin |
| 111 | ⋮ | s2<=s3; |
| 112 | ⋮ | s3<=s2; |
| 113 | ⬇ | end |
| 114 | ⬇ | end |
| 115 | ⬇ | 3' b100: |
| 116 | ⬇ | begin |
| 117 | ⬇ | if(a<b) |
| 118 | ⬇ | begin |
| 119 | ⋮ | s0<=s1; |
| 120 | ⋮ | s1<=s0; |
| 121 | ⬇ | end |
| 122 | ⬇ | end |
| 123 | ⬇ | 3' b101: |
| 124 | ⬇ | begin |
| 125 | ⬇ | if(a<b) |
| 126 | ⬇ | begin |
| 127 | ⋮ | s1<=s2; |
| 128 | ⋮ | s2<=s1; |
| 129 | ⬇ | end |
| 130 | ⬇ | end |
| 131 | ⬇ | 3' b110: |
| 132 | ⬇ | begin |
| 133 | ⬇ | if(a<b) |
| 134 | ⬇ | begin |
| 135 | ⋮ | s0<=s1; |
| 136 | ⋮ | s1<=s0; |
| 137 | ⬇ | end |
| 138 | ⋮ | done<=1; |
| 139 | ⬇ | end |
| 140 | ⬇ | endcase |
| 141 | ⬇ | end |
| 142 | ⋮ | |

排序器核心代码截图，采用三段式状态机

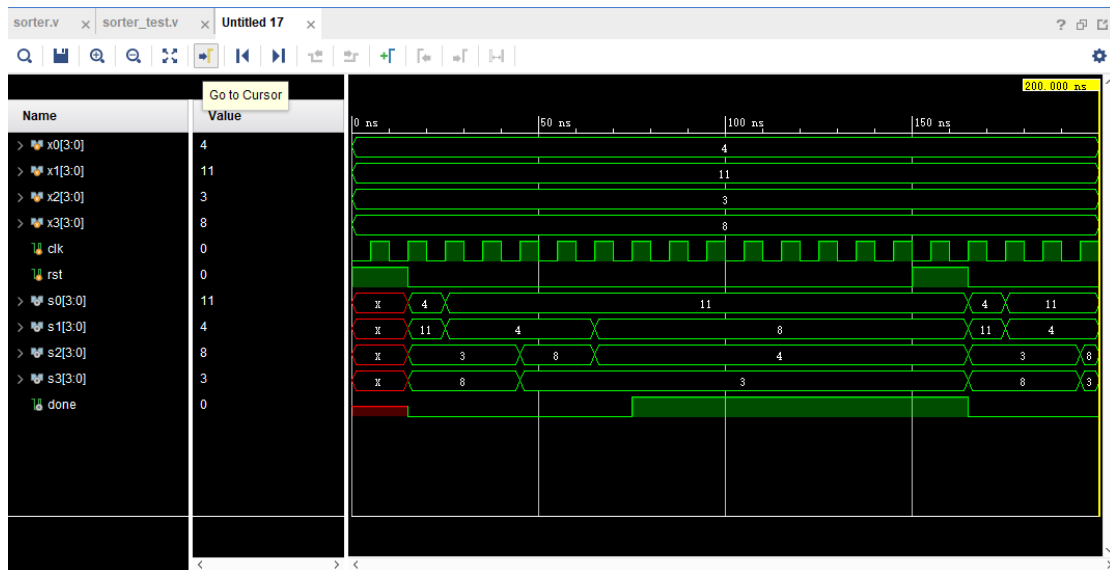
(2). 除法器

```
39   always @(posedge clk, negedge rst)
40  begin
41  if(rst)
42  state<=3'b000;
43  else
44  state<=nextstate;
45  end
47  always @(*)
48  begin
49  case(state)
50  3'b000: nextstate=3'b001;
51  3'b001: nextstate=3'b010;
52  3'b010: nextstate=3'b011;
53  3'b011: nextstate=3'b100;
54  3'b100: nextstate=3'b101;
55  3'b101: nextstate=3'b110;
56  3'b110: nextstate=3'b111;
57  3'b111: nextstate=3'b111;
58  endcase
59  end
60
61  always @(posedge clk)
62  begin
63  if(y==4'b0000)
64  begin
65  error=1;
66  done=1;
67  end
68  else
69  begin
70  case(state)
71  3'b000:
72  begin
73  X={4'b0000, x};
74  Y={y, 4'b0000};
75  q=4'b0000;
76  r=4'b0000;
77  done=0;
78  error=0;
79  end
```

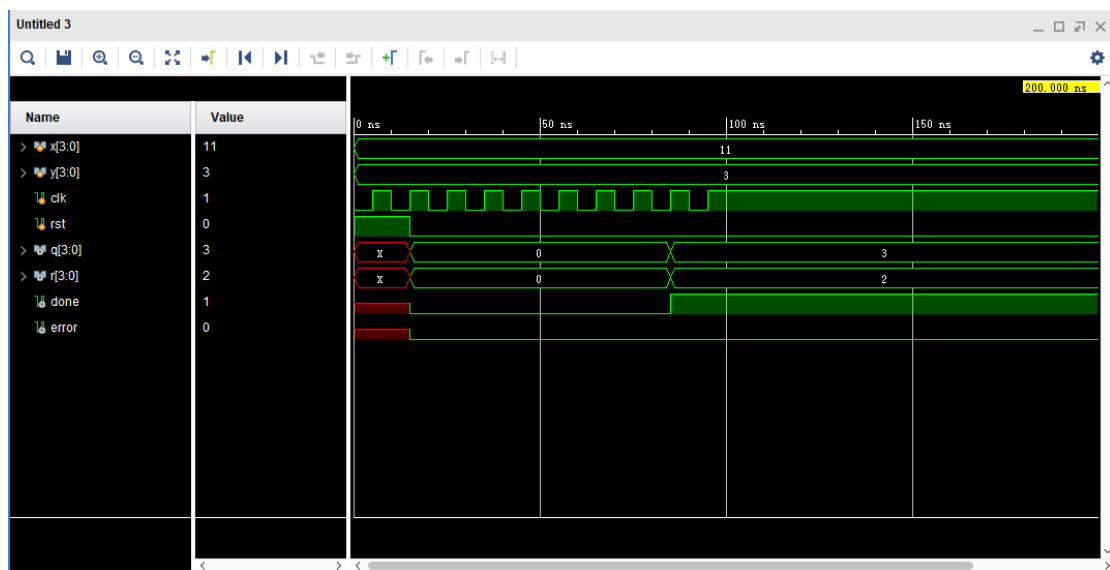
| | | |
|-------|---|---------------|
| 80 | ⬇ | 3' b001: |
| 81 | ⬇ | begin |
| 82 | ○ | X<=X+X; |
| 83 | ○ | Z=X[7:4]; |
| 84 | ○ | if(Z>=y) |
| 85 | ○ | X<=X-Y+1' b1; |
| 86 | ⬇ | end |
| 87 | ⬇ | 3' b010: |
| 88 | ⬇ | begin |
| 89 | ○ | X<=X+X; |
| 90 | ○ | Z=X[7:4]; |
| 91 | ○ | if(Z>=y) |
| 92 | ○ | X<=X-Y+1' b1; |
| 93 | ⬇ | end |
| 94 | ⬇ | 3' b011: |
| 95 | ⬇ | begin |
| 96 | ○ | X<=X+X; |
| 97 | ○ | Z=X[7:4]; |
| 98 | ○ | if(Z>=y) |
| 99 | ○ | X<=X-Y+1' b1; |
| 100 | ⬇ | end |
| <hr/> | | |
| 101 | ⬇ | 3' b100: |
| 102 | ⬇ | begin |
| 103 | ○ | X<=X+X; |
| 104 | ○ | Z=X[7:4]; |
| 105 | ○ | if(Z>=y) |
| 106 | ○ | X<=X-Y+1' b1; |
| 107 | ⬇ | end |
| 108 | ⬇ | 3' b101: |
| 109 | ⬇ | begin |
| 110 | ○ | X<=X+X; |
| 111 | ○ | Z=X[7:4]; |
| 112 | ○ | if(Z>=y) |
| 113 | ○ | X<=X-Y+1' b1; |
| 114 | ⬇ | end |
| 115 | ⬇ | 3' b110: |
| 116 | ⬇ | begin |
| 117 | ○ | X<=X+X; |
| 118 | ○ | Z=X[7:4]; |
| 119 | ○ | if(Z>=y) |
| 120 | ○ | X<=X-Y+1' b1; |
| 121 | ⬇ | end |
| 122 | ⬇ | 3' b111: |
| 123 | ⬇ | begin |
| 124 | ○ | q<=X[3:0]; |
| 125 | ○ | r<=X[7:4]; |
| 126 | ○ | done=1; |
| 127 | ⬇ | end |
| 128 | ⬇ | endcase |
| 129 | ⬇ | end |
| 130 | ⬇ | end |

除法器核心代码截图，同样采取三段式状态机

3. 仿真结果与下载结果：



排序器仿真，输入四个数为 4 11 3 8 输出排序结果为 11 8 4 3



除法器仿真，输入被除数 11 除数 3，输出商 3 余 2， error 为 0

下载照片：



排序器初始输入 4231



排序结果输出 4321，同时 DONE 指示灯亮起



输入 11/3，输出商 3 余 2，DONE 亮起

4. 结果分析

排序结果准确，除法运算结果准确

5. 实验总结

在一周前的 lab1 实验中重新温习了 verilog 的语法，本次试验相较上一次更加得心应手。本次试验主要学习了状态机的 verilog 表示方法，着重学习了两段式与三段式。三段式状态机可修改性和稳定性都更强，因此在写排序器和除法器时使用了三段式。写状态机的难点在于画正确的状态图，分清一共有多少个状态及状态间的转换关系还有状态中的具体操作。

6. 意见/建议：

希望具体实验要求能早一点明确地给出来，不要临时改要求导致之前做了些无用功。