时序4： Verilog 状态机设计技术

10.1 Verilog 状态机的一般形式

一、状态机的特点及优势

- 实现控制电路的方法

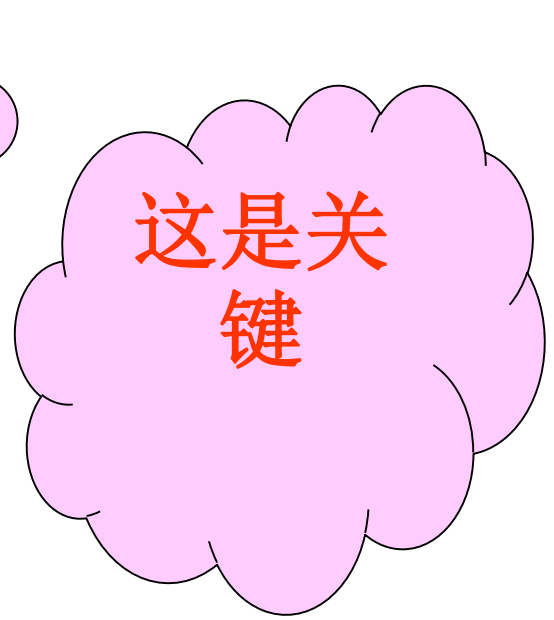
- 高性能，高可靠性
- P 166

什么是状态机？

- 复习时序逻辑电路

复习时序逻辑电路

- （一）同步时序逻辑电路的设计是以状态图为基础的
- 语句描述 → 状态图 → 同步时序逻辑电路



数字电子技术基础
第五版 阎石主编 P261

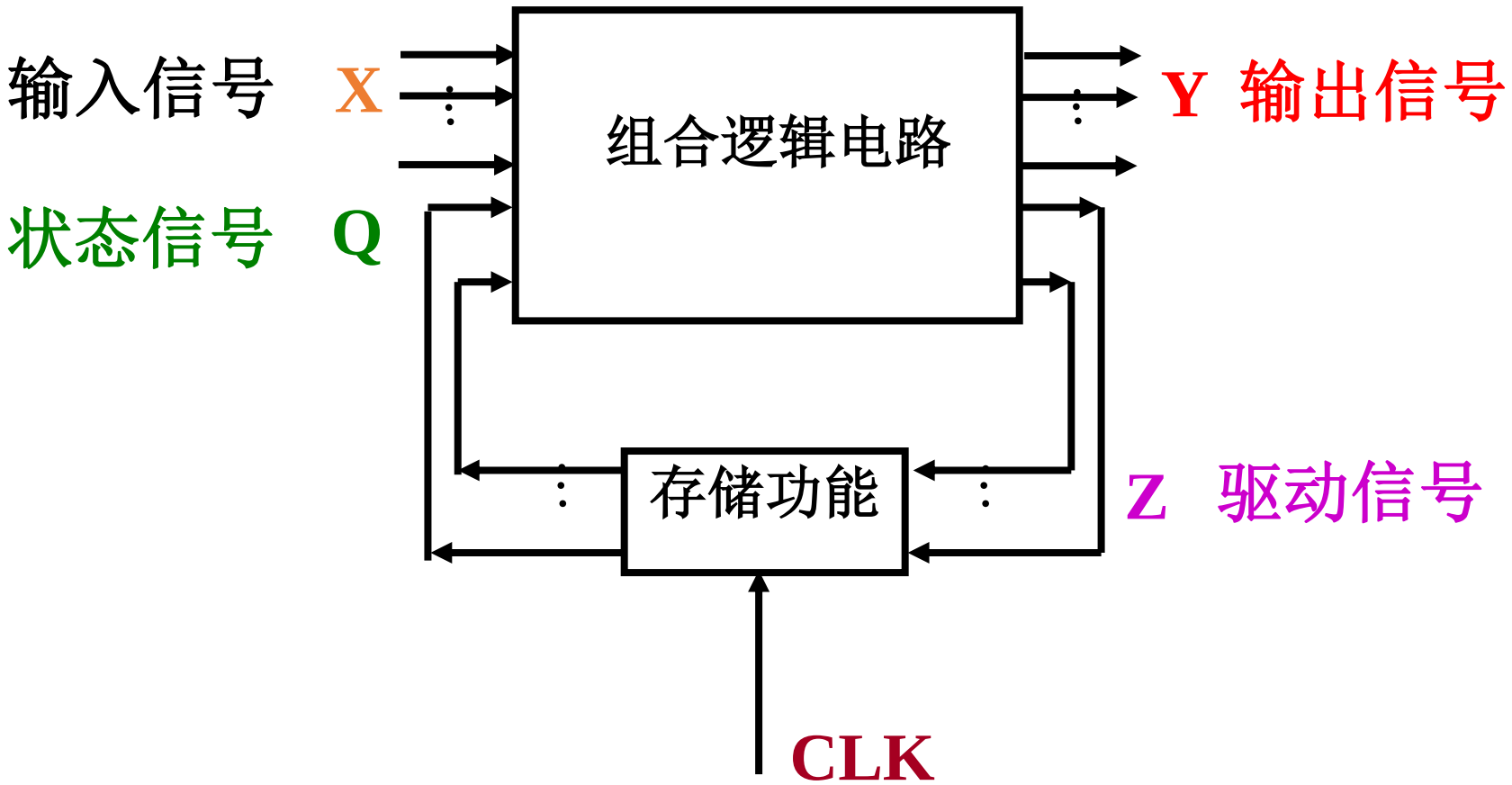
- 鉴于时序逻辑电路在工作时是在电路的有限个状态间按一定的规律转换的，所以又将时序逻辑电路称为状态机（state machine)

- 状态机分为 ？
- Moore状态机， Mealy状态机

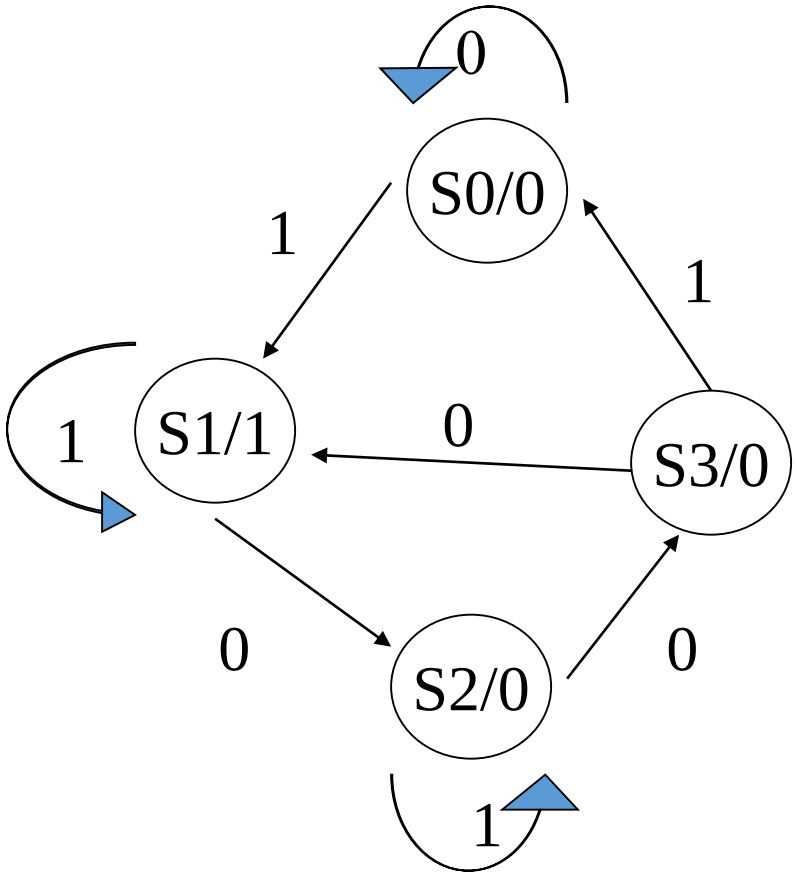
复习时序逻辑电路
（二）时序逻辑电路重要的基本概念

- 1、什么是时序逻辑电路？

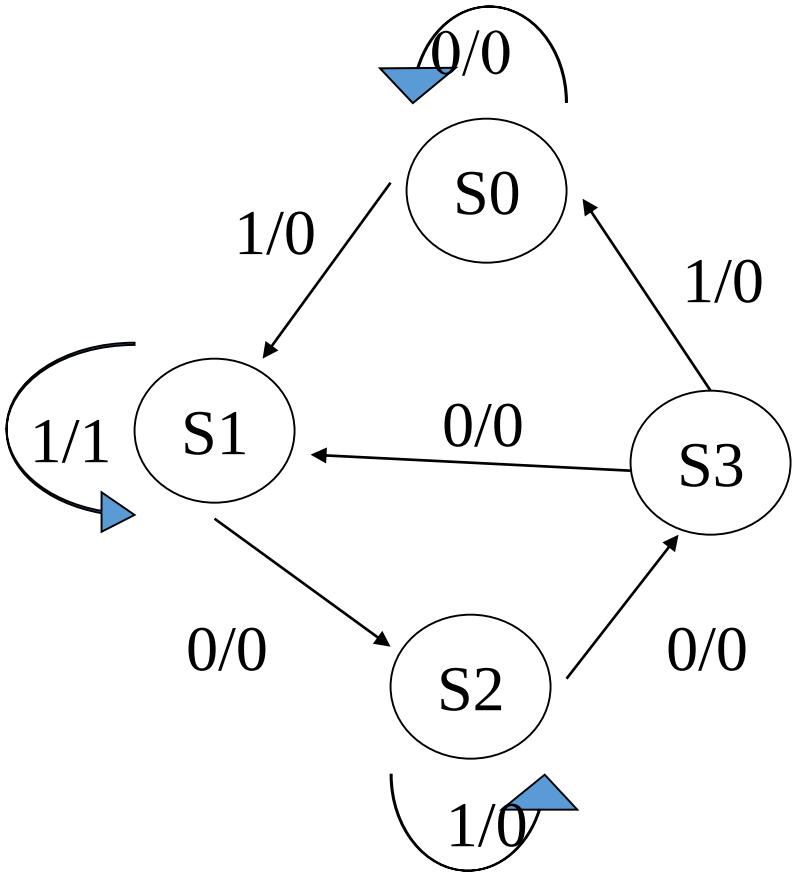
2、时序逻辑
电路结构框图



3、 Moore状态机

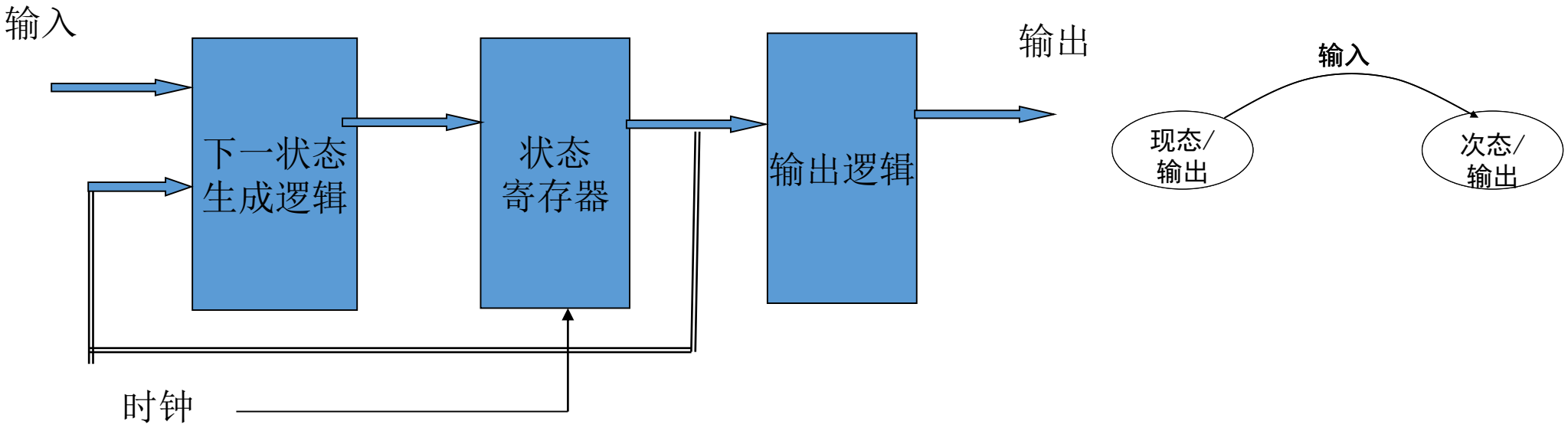


Mealy状态机

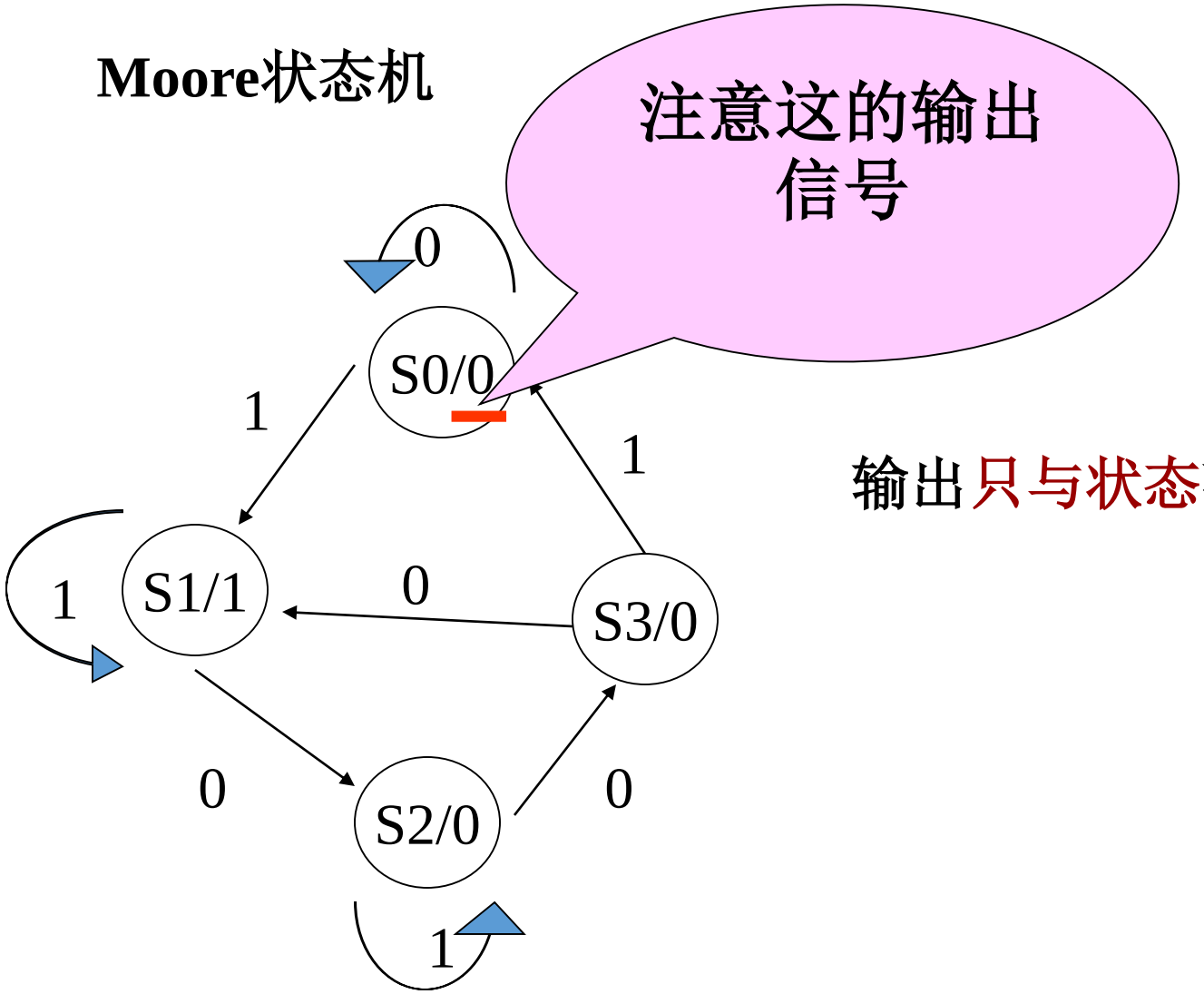


（三）、 Moore状态机

- **Moore型状态机输出为当前状态的函数。**
- **状态转换图示意图如下图所示**
- **由于状态的变化与时钟同步，因此输出的变化也与时钟同步，属于同步输出电路模型。**

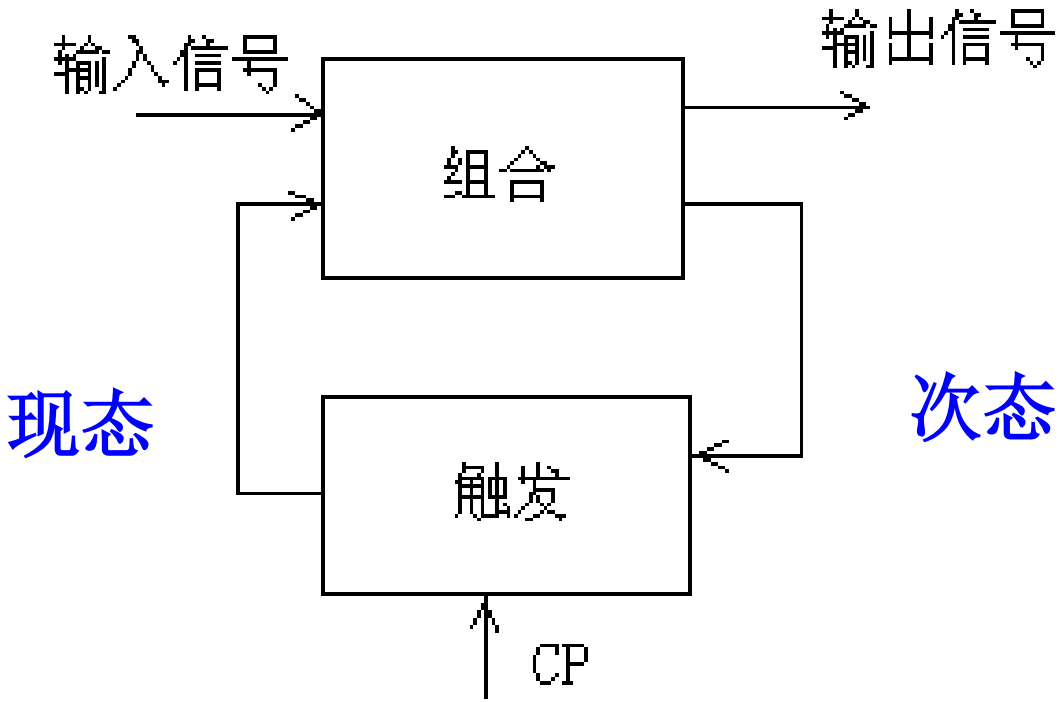


Moore状态机

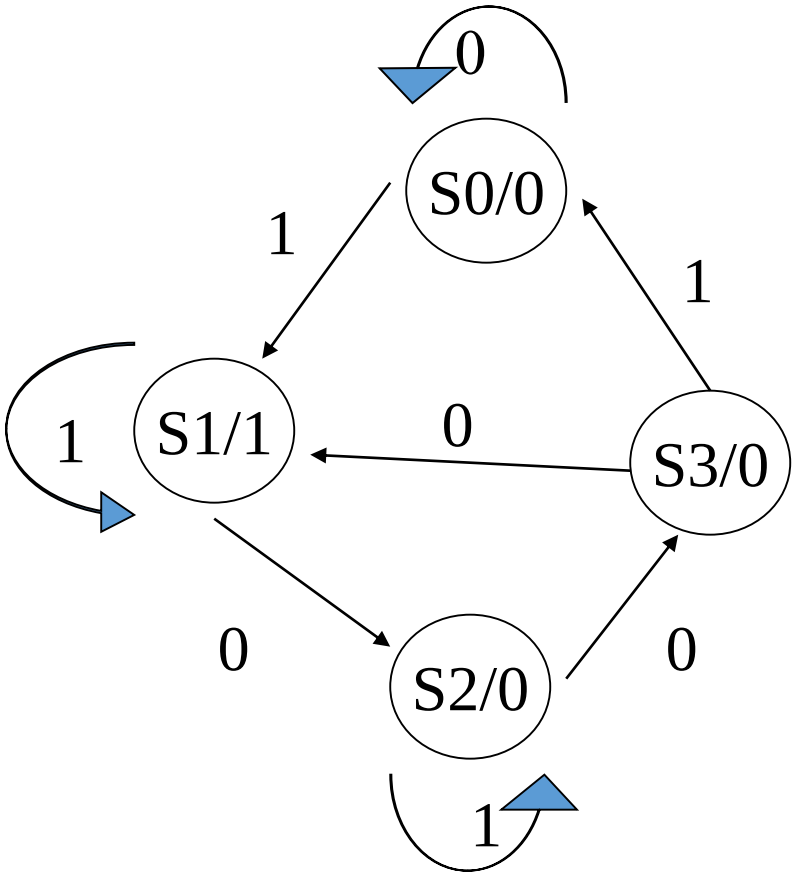


输出只与状态有关

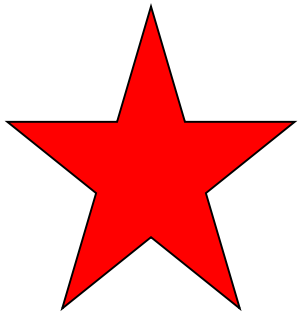
Moore状态机结构框图



结合结构框图　　读状态机

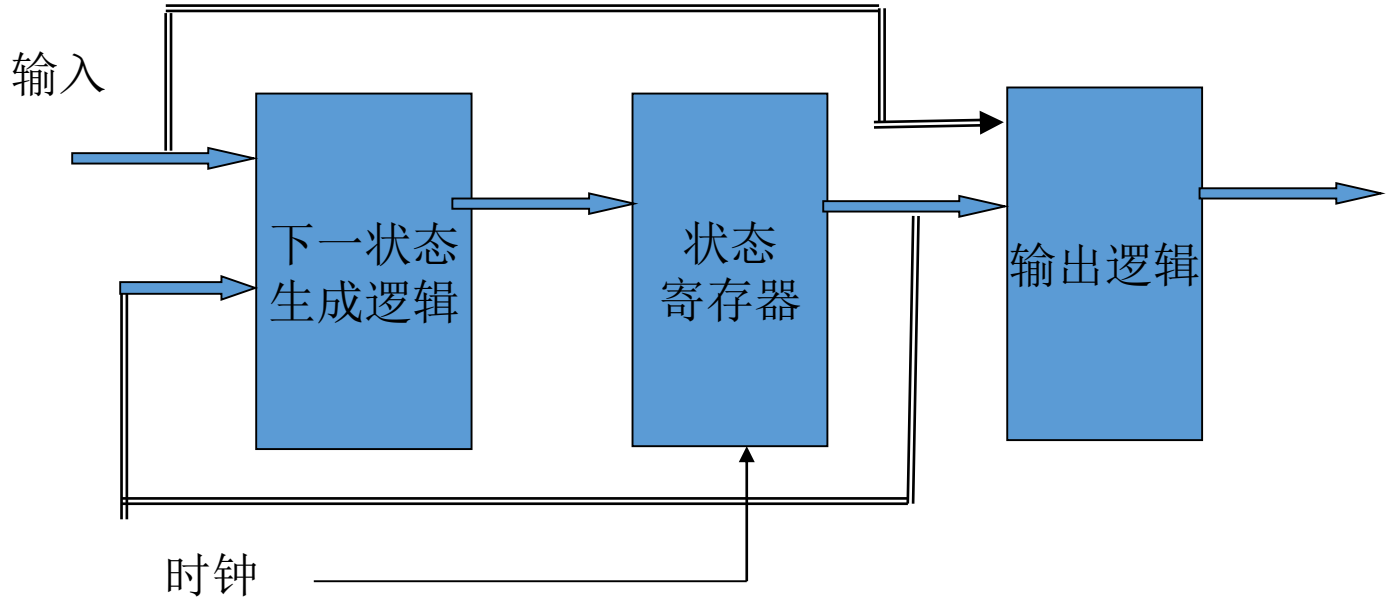


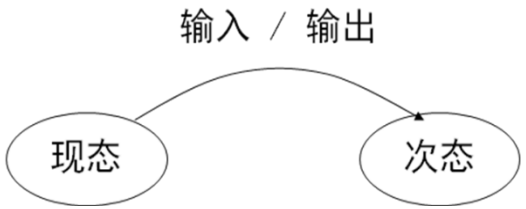
方法：画时序图



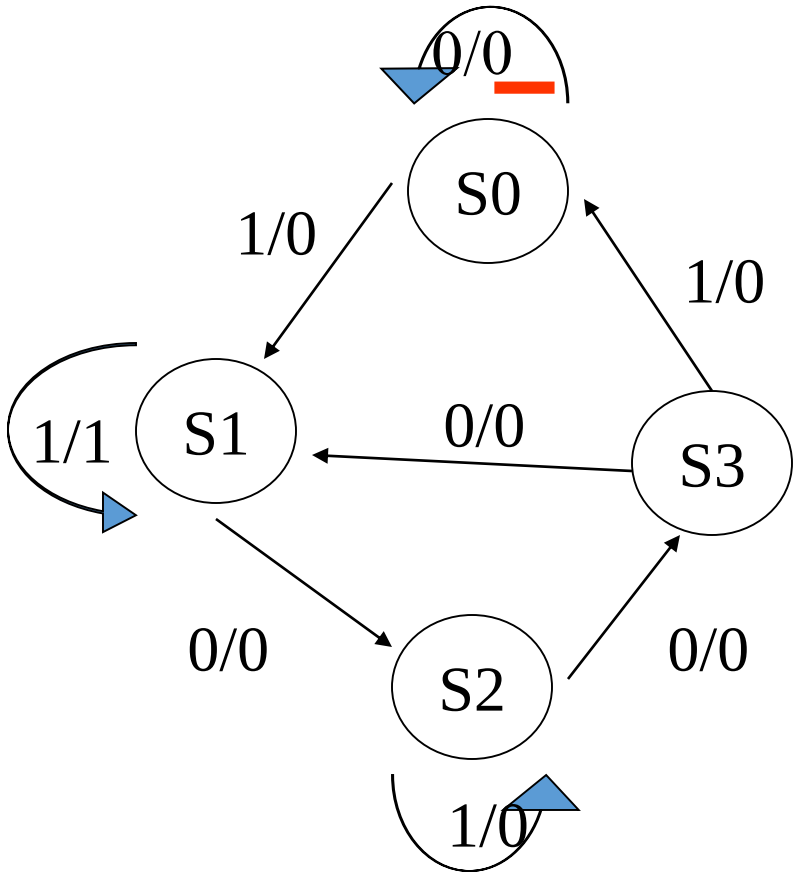
（四）、Mealy状态机

- **Mealy型状态机的输出为当前状态和所有输入信号的函数**
- **状态转换图示意图如下图所示**
- **由于输入信号与时钟信号无关，因此其输出在输入变化后将立即发生变化，不依赖时钟，因此Mealy型状态机属于异步输出电路模型。**



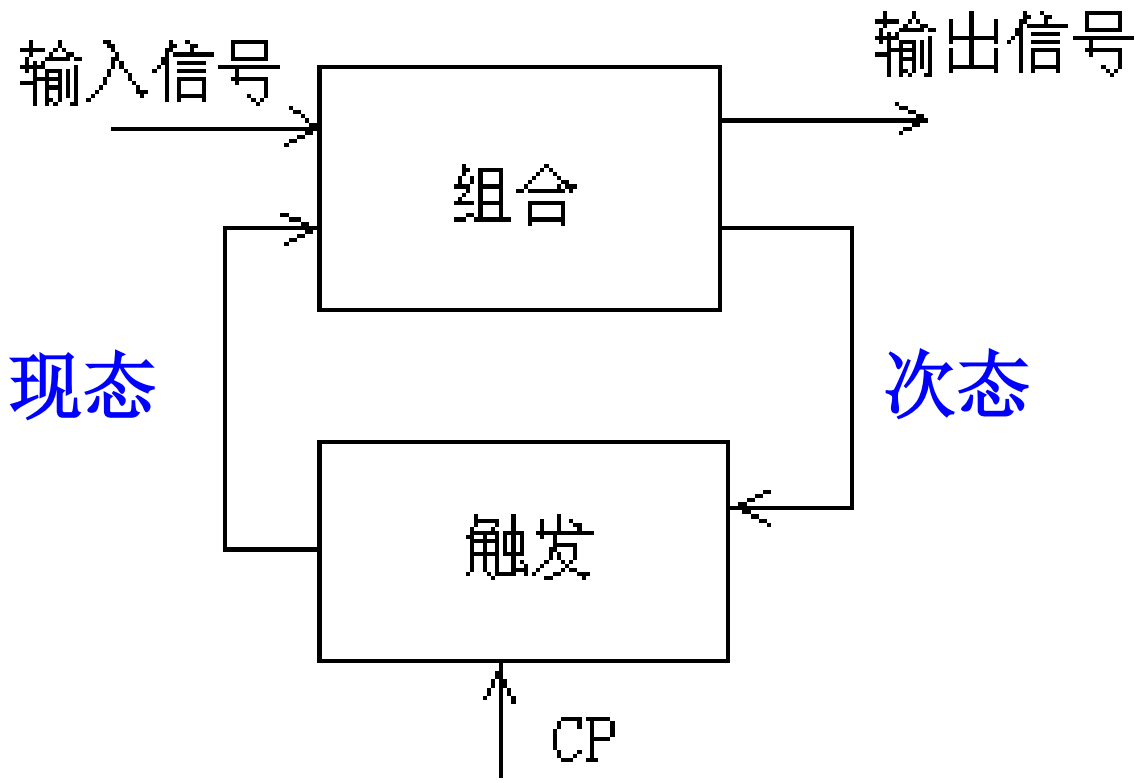


Mealy状态机



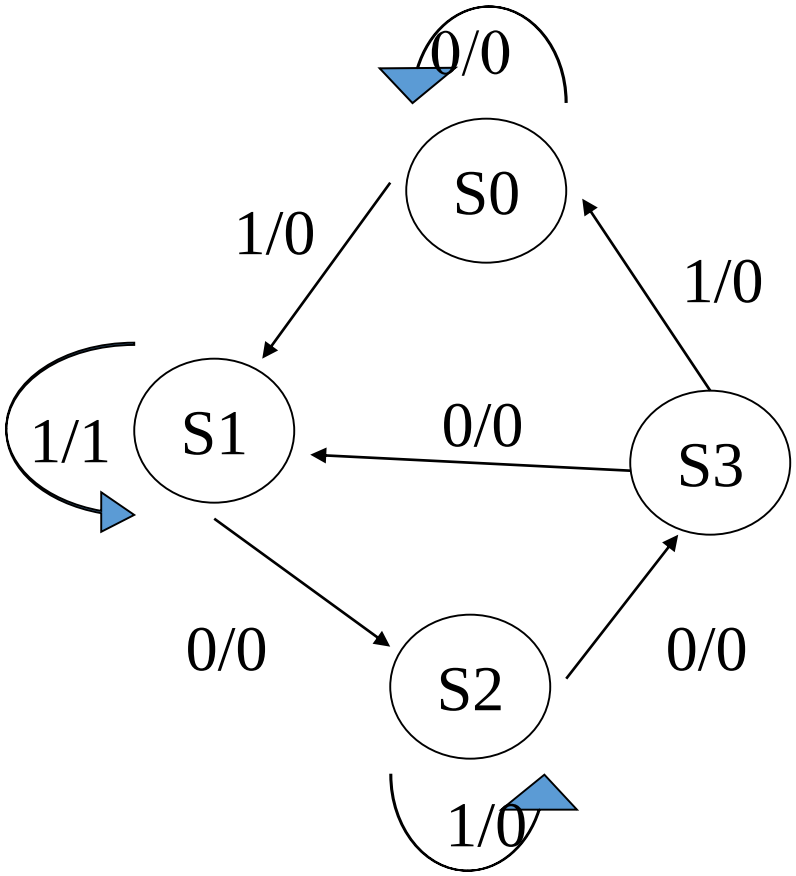
输出与状态及输入都有关系

Mealy状态机结构框图

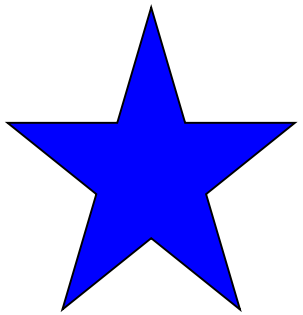


结合结构框图

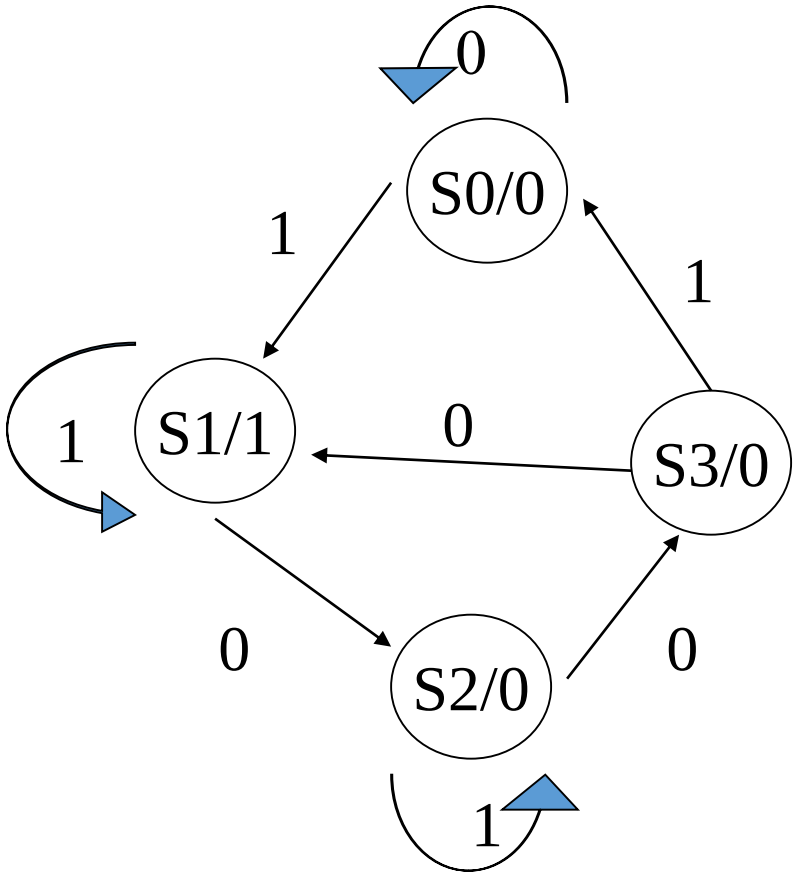
读状态机



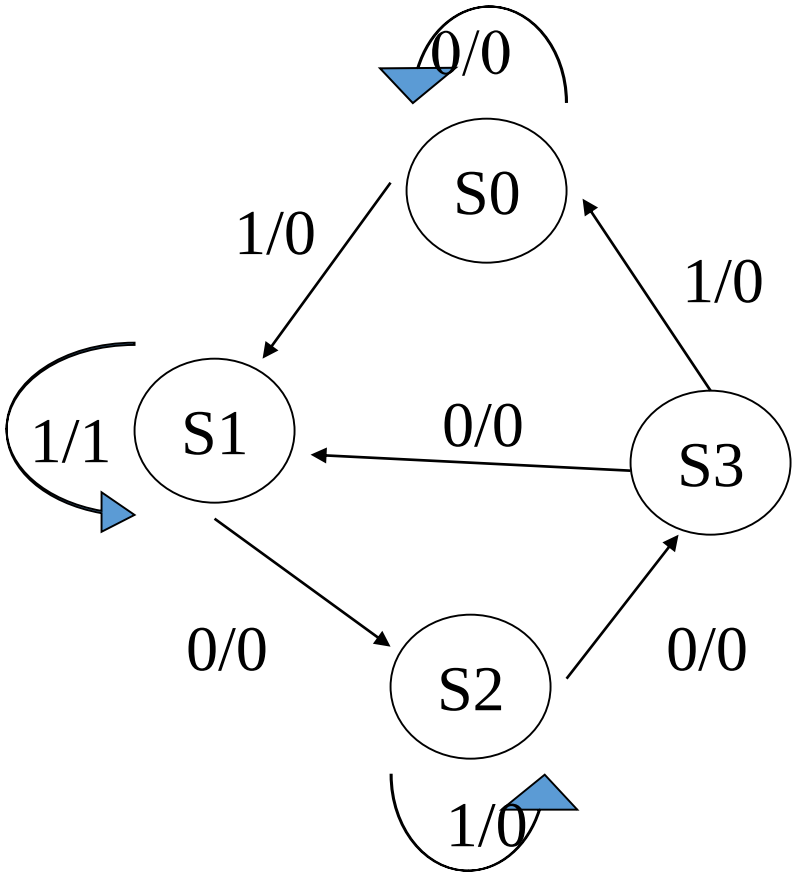
方法：画时序图



Moore状态机



Mealy状态机



10.1 Verilog 状态机的一般形式

- 二、状态机的一般结构
- 1、状态机说明部分
- 2、主控时序过程
- 3、主控组合过程
- 4、辅助过程（可以有，可以没有）

10.1 Verilog 状态机的一般形式

- 二、状态机的一般结构
- 1、状态机说明部分

parameter(P82)

- 定义常量的关键词
- 例如：
- **parameter sel=8,code=8’ha3；**

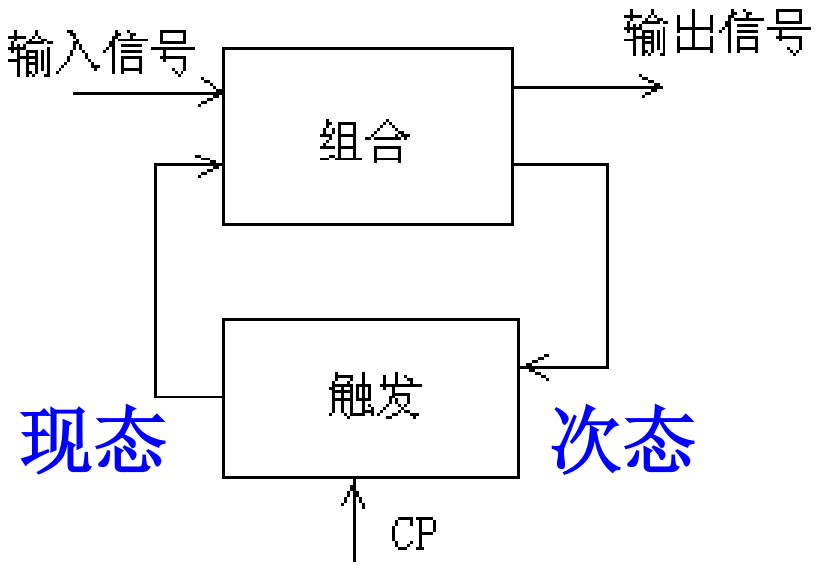
- 分别定义参数**sel**代表**8(**十进制**)**
- 参数**code**代表常量**a3(**十六进制**)**

10.1 Verilog 状态机的一般形式

- 二、状态机的一般结构
- **1、状态机说明部分**
- **parameter s0=0,s1=1,s2=2,s3=3;**

10.1 Verilog 状态机的一般形式

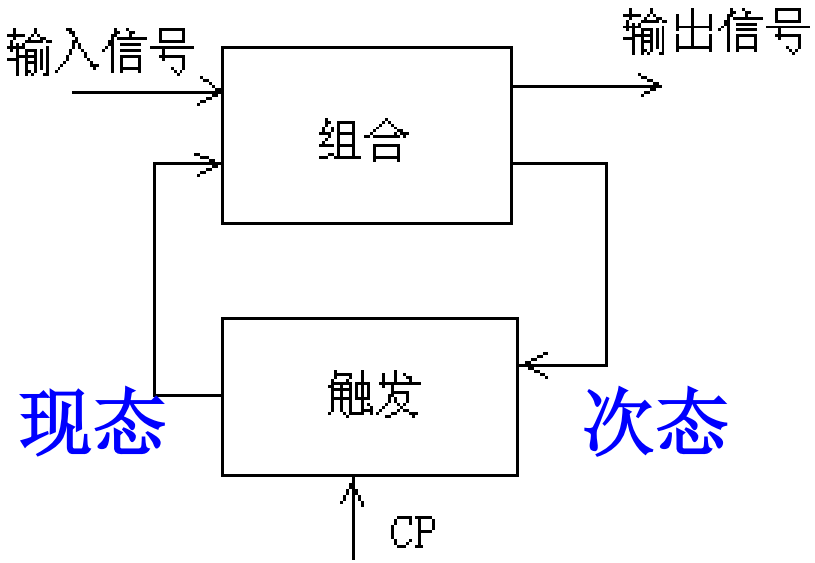
- 二、状态机的一般结构
- 1、状态机说明部分
- parameter s0=0,s1=1,s2=2,s3=3;

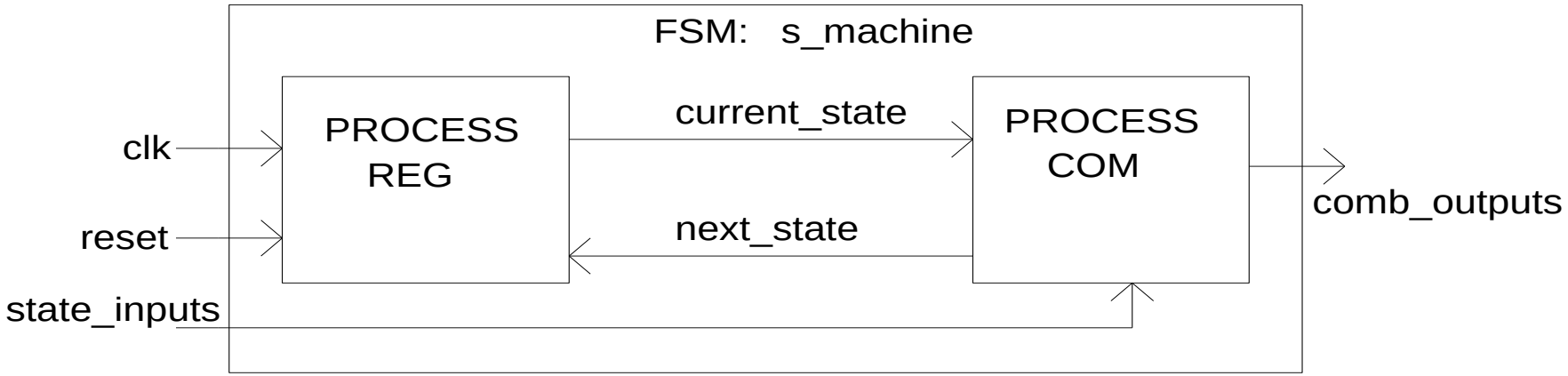


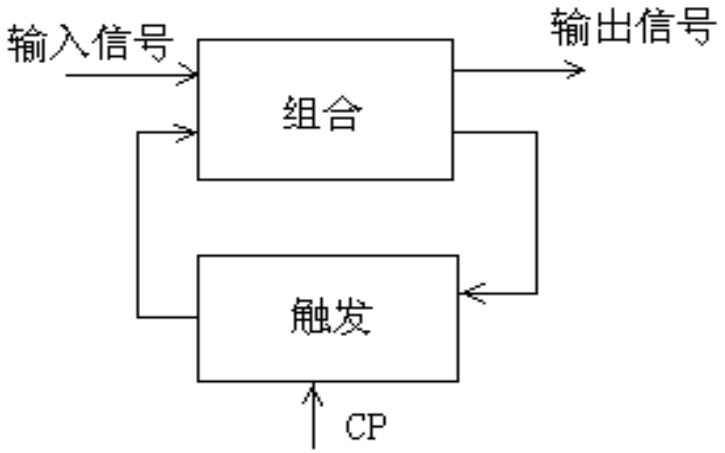
- 2、主控时序过程
- 在时钟驱动下负责状态转换
- 现态<=次态;

10.1 Verilog 状态机的一般形式

- 二、状态机的一般结构
- 1、状态机说明部分
- parameter s0=0,s1=1,s2=2,s3=3;
- 2、主控时序过程
- 在时钟驱动下负责状态转换
- 3、主控组合过程
- 由现态和输入信号决定输出信号和次态
- 4、辅助过程（可以有，可以没有）
- 用于配合状态机工作的组合过程或时序过程




FSM: s_machine
PROCESS
REG
PROCESS
COM
clk
reset
state_inputs
current_state
next_state
comb_outputs


输入信号
输出信号
组合
触发
CP

读程序（分析程序 P250）

- module fsm_exp(clk,reset,state_inputs,comb_outputs);
- input clk,reset;
- input[0:1] state_inputs;
- output[3:0] comb_outputs;
- reg[3:0] comb_outputs;

- parameter s0=0,s1=1,s2=2,s3=3,s4=4;
- reg[4:0] c_st,next_state;

```
always@(posedge clk or negedge reset)
    begin
      if(!reset) c_st<=s0;
      else c_st<=next_state;
    end
always@(c_st or state_inputs)
  case(c_st)
  s0:begin comb_outputs<=5;
      if(state_inputs==2'b00) next_state<=s0;
      else next_state<=s1;    end
```

```
        s1:begin comb_outputs<=8;
           if(state_inputs==2'b01) next_state<=s1;
           else next_state<=s2;    end
        s2:begin comb_outputs<=12;
           if(state_inputs==2'b10) next_state<=s0;
           else next_state<=s3;    end
        s3:begin comb_outputs<=14;
           if(state_inputs==2'b11) next_state<=s3;
           else next_state<=s4;    end
        s4:begin comb_outputs<=9;
           next_state<=s0;    end
        default:next_state<=s0;
endcase        endmodule
```

掌握程序分析方法

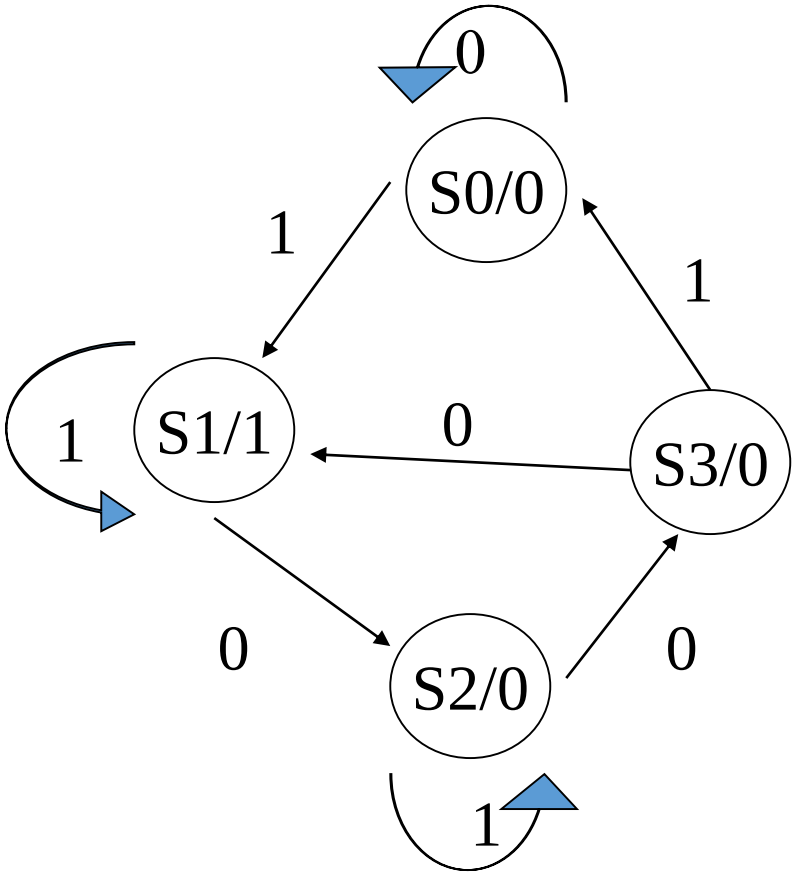
- 读程序
- 1、画端口信息
- 2、用画图的方式来表示逻辑功能
- 例如：根据程序画出状态转换图
- 根据程序列出真值表
- 根据程序画出时序图
- 等等

10.2 状态机的Verilog HDL设计步骤

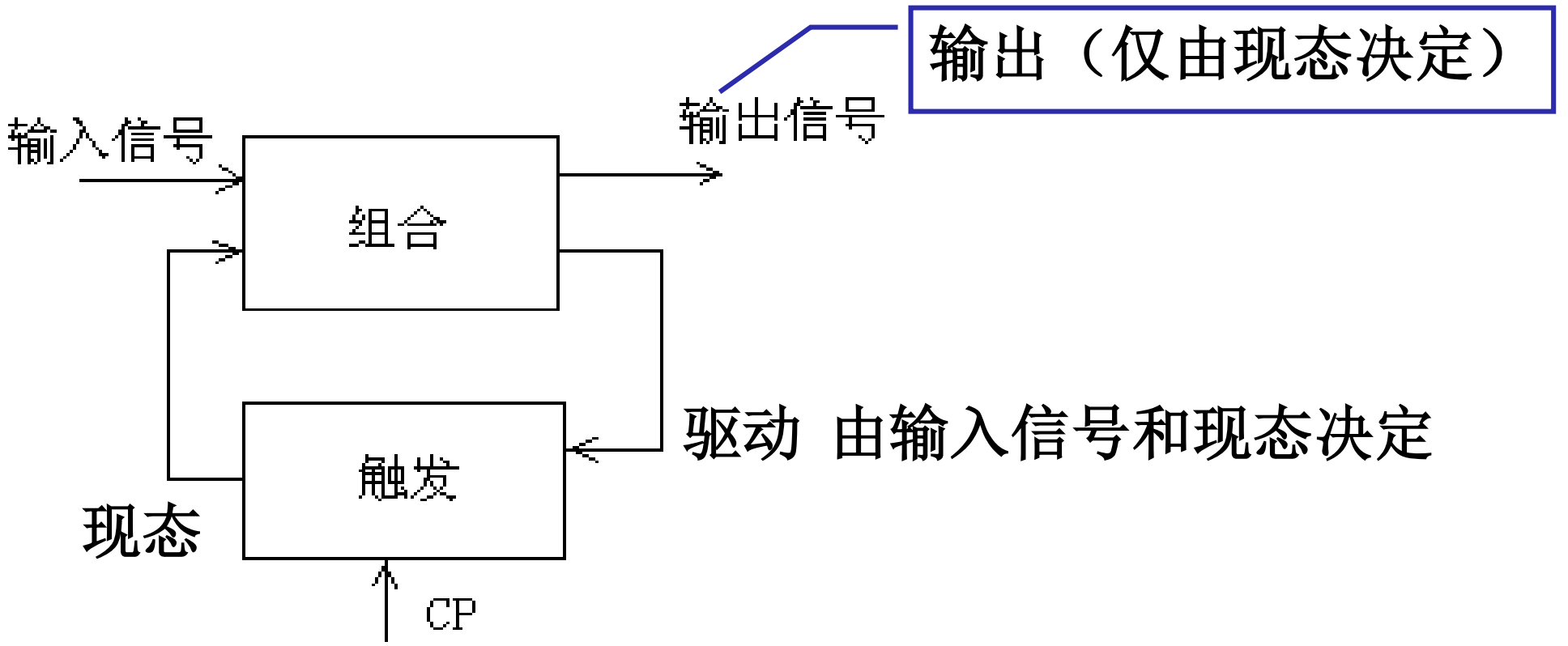
- 1、已知状态机
- 2、首先画出时序逻辑电路的结构框图
- 3、画出电路端口信息
- 4、画时序图（详细了解状态的转换过程）
- 5、编程

例1

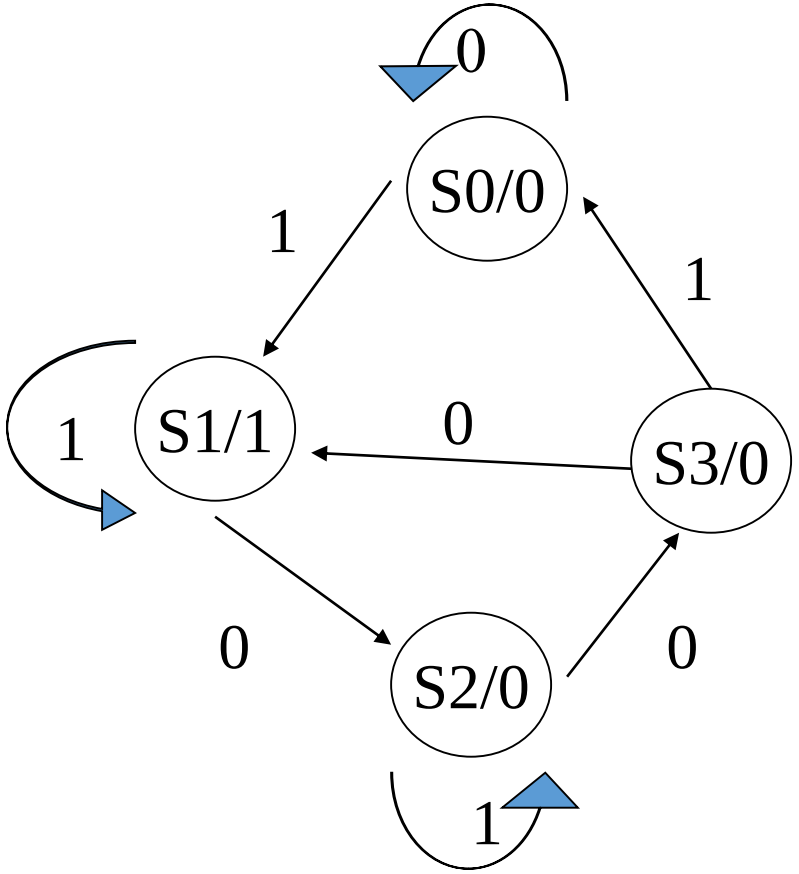
1、已知状态机



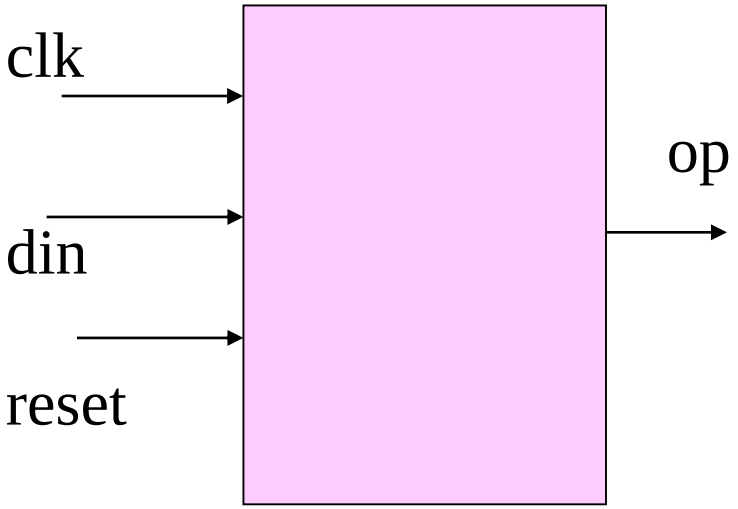
2、首先画出
时序逻辑电路的结构框图

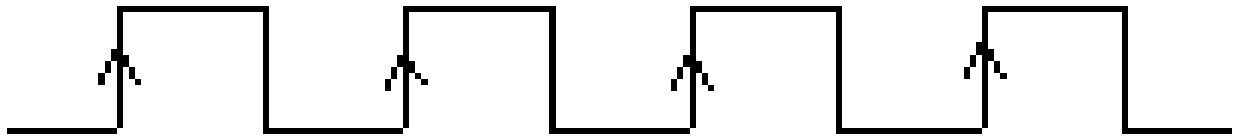


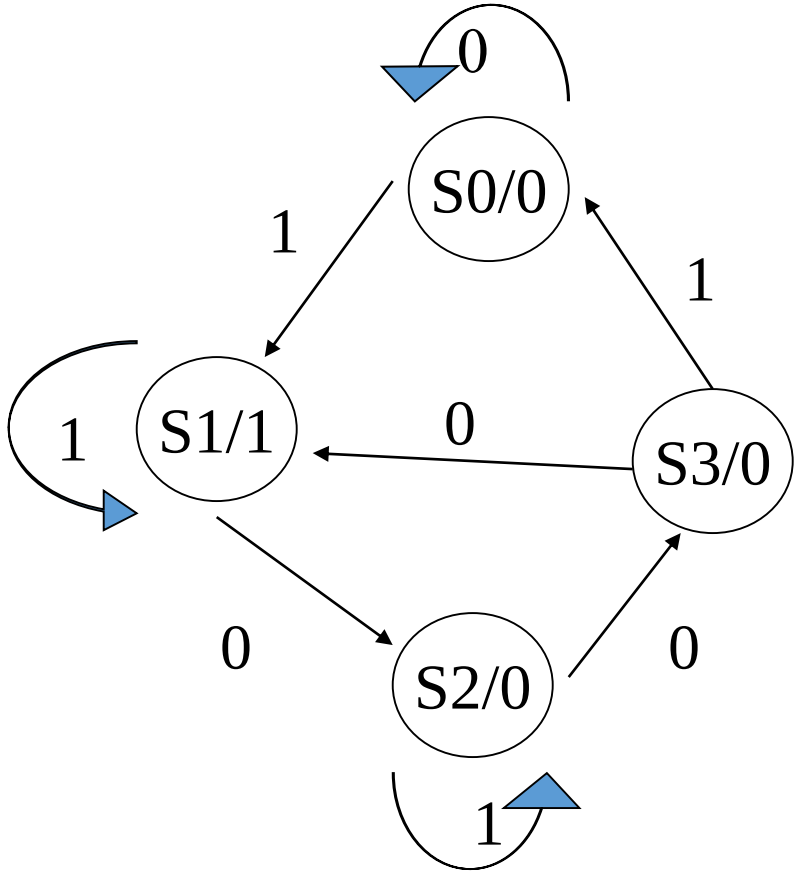
Moore状态机：输出仅由状态决定



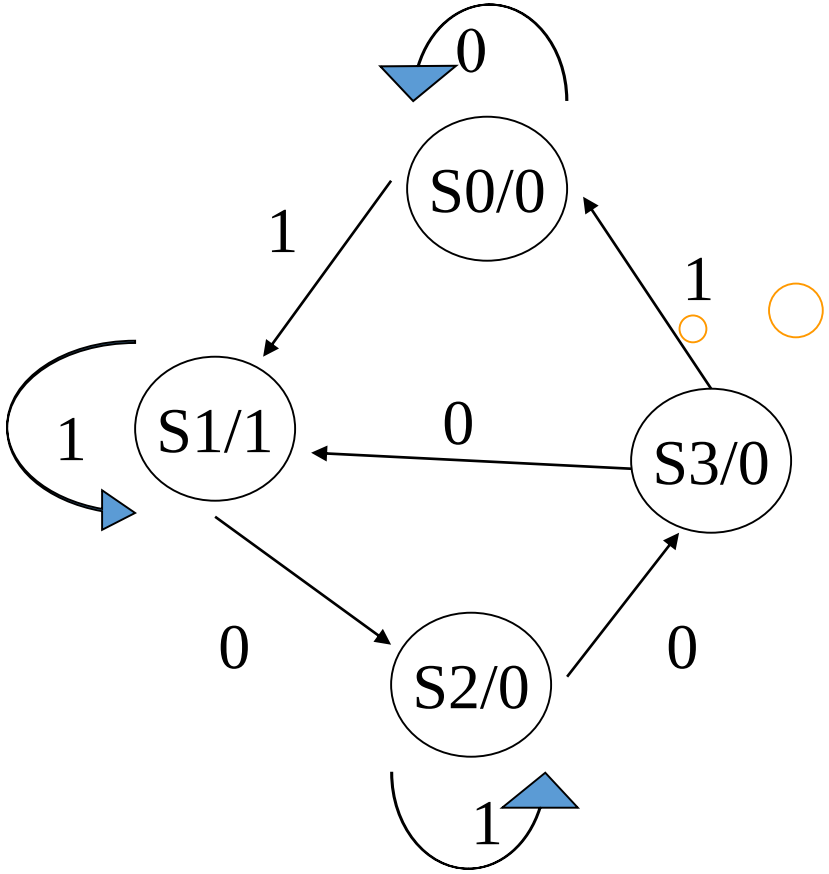
3、电路端口信息



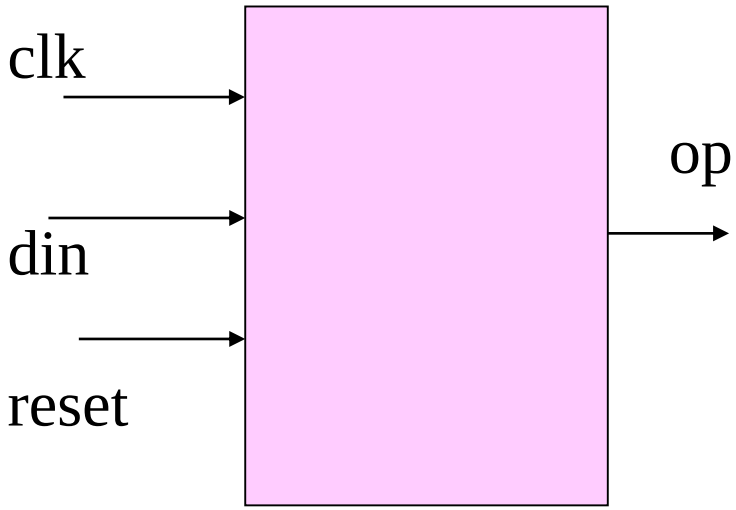




4、画时序图（详细了解状态的转换过程）



5、由状态机写出Verilog HDL语言的描述



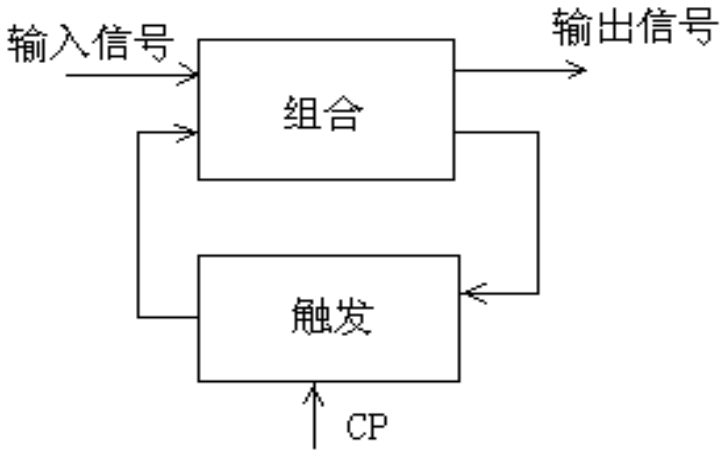
这是本课程的重点：

用Verilog HDL语言设计此状态图所表示的电路

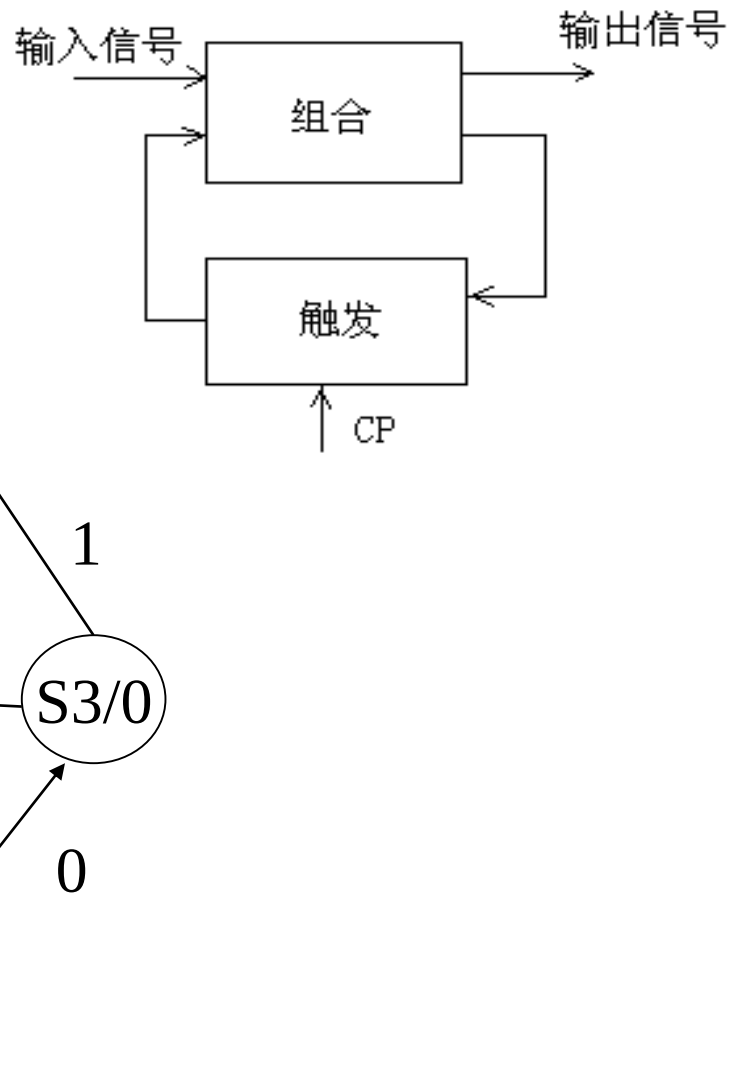
```
module li_1(clk,reset,din,op);
    input clk,reset,din;
   output op;
    reg op;

   parameter s0=0,s1=1,s2=2,s3=3;
   reg[2:0] pstate,nstate;

 always@(posedge clk or negedge reset)
       begin
         if(!reset) pstate<=s0;
         else pstate<=nstate;
      end
```



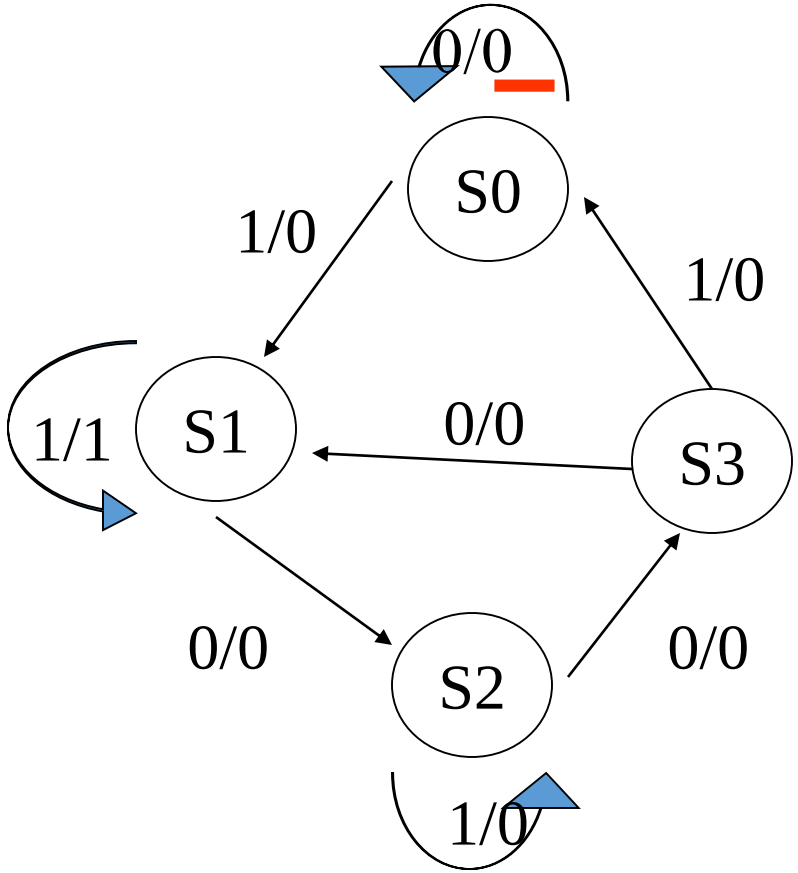
```
always@(pstate or din)
    case(pstate)
    s0: begin op<=0;
        if(din==1'b0) state<=s0;
        else nstate<=s1;    end
   s1: begin op<=1;
        if(din==1'b1) nstate<=s1;
        else nstate<=s2;    end
    s2: begin op<=0;
        if(din==1'b1) nstate<=s2;
        else nstate<=s3;    end
    s3: begin op<=0;
        if(din==1'b1) nstate<=s0;
        else nstate<=s1;    end
    default:nstate<=s0;
  endcase
```



掌握编写程序的方法

- 读题目要求，抓住已知条件　　状态图
- 根据已知条件，准备些材料
- 　　为了描述逻辑功能　　结构框图，时序图
- 画端口信息
- 编写程序　　**parameter**

编写下列状态机



Mealy状态机

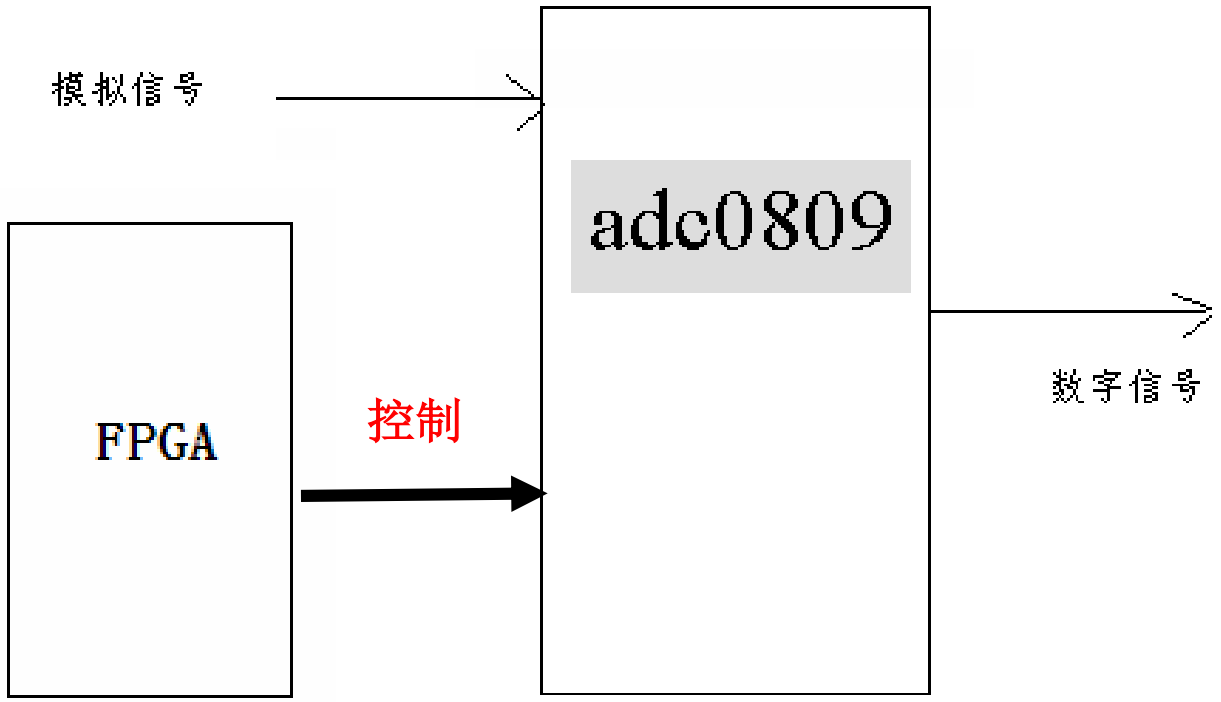
输出与状态及输入都有关系

10.2 状态机的Verilog HDL设计步骤

- 1、已知状态机
- 2、首先画出时序逻辑电路的结构框图
- 3、画出电路端口信息
- 4、画时序图（详细了解状态的转换过程）
- 5、编程

状态机应用实例：
ADC0809采样控制状态机的实现

- 第一步：构思电路框架



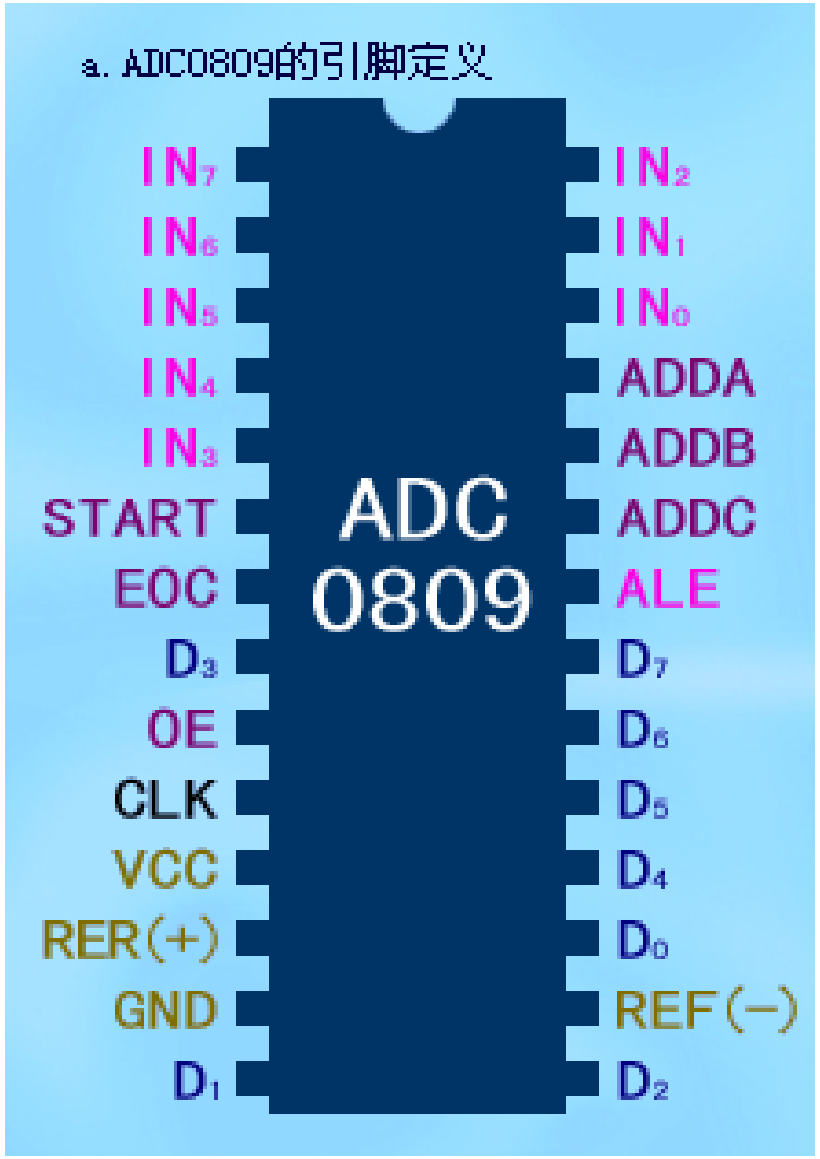
状态机应用实例：
ADC0809采样控制状态机的实现

- 第二步：研究电路的元件
- 1、首先了解ADC0809
- 2、 ADC0809的符号图
- 3、ADC0809的工作原理

1、首先了解ADC0809

ADC0809芯片是按典型的逐位比较工作原理集成的，但需外接参考电源和时钟（10KHZ－1.2MHZ），在时钟为640KHZ时，一次变换时间为100μs

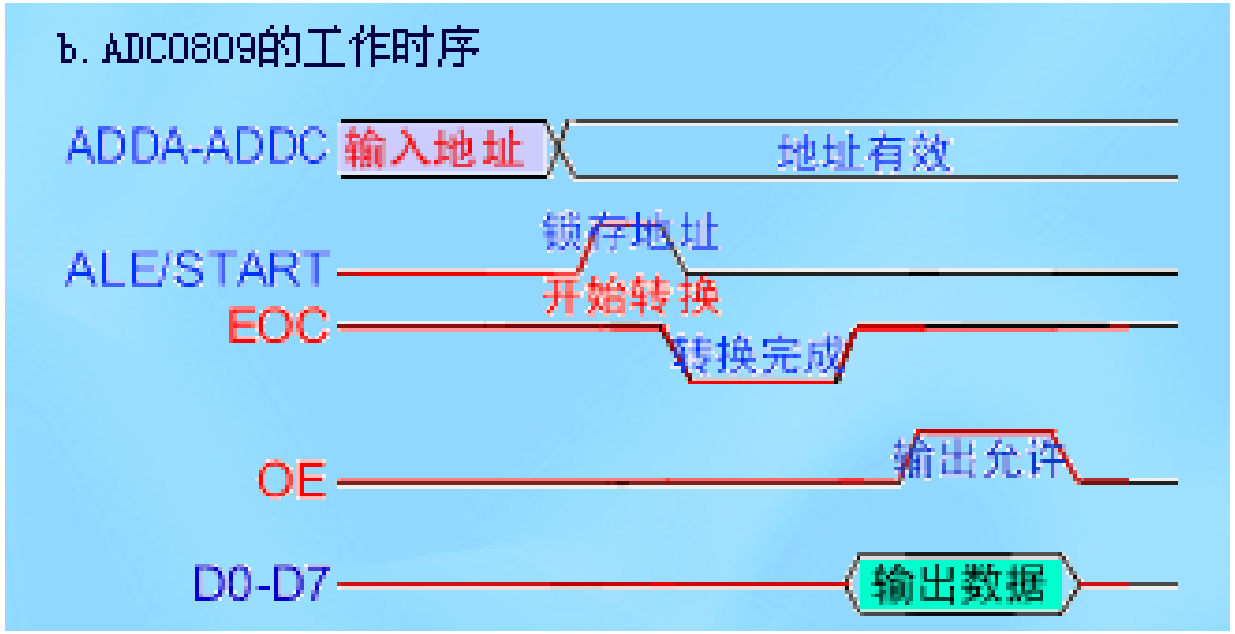
芯片内有一个八选一的模拟开关，利用ADDA－ADDC三个信号编码选择相应的输入。



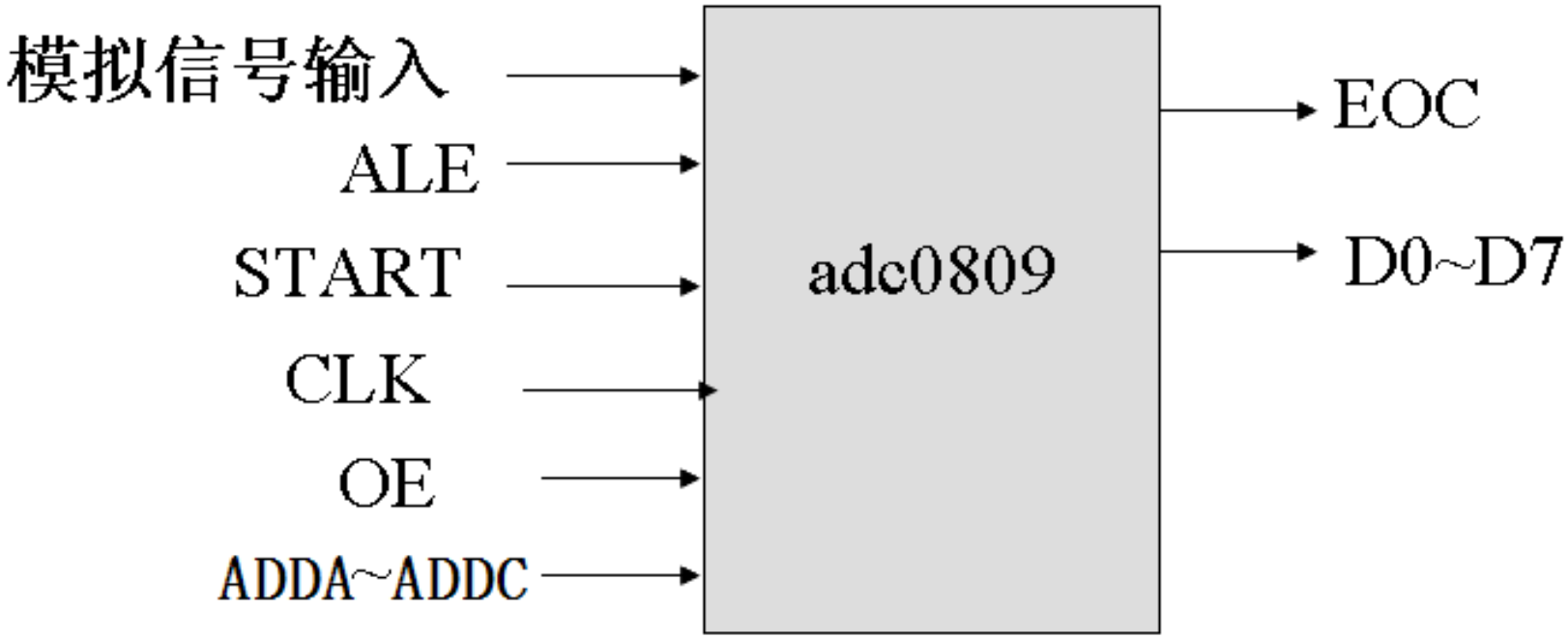
ALE：地址锁存信号，上升沿有效

START：启动输入信号，上升沿有效

b. ADC0809的工作时序



2、ADC0809的符号图



- 这里的CLK是将模拟量转换成数字量逐次渐近比较的时钟信号
- 根据芯片工艺，时钟范围为（10KHZ－1.2MHZ）

3、工作原理

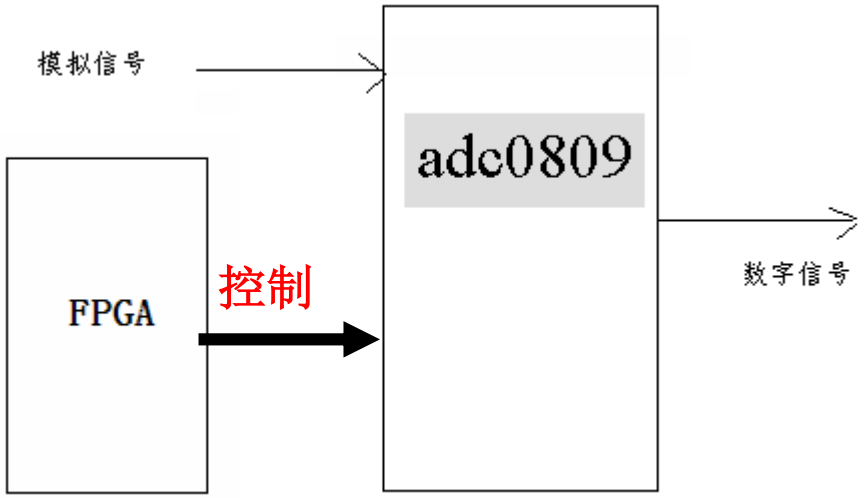
- 地址信号加入后，利用ALE加一个正跳变脉冲，将端口上的地址信号锁存于内部地址寄存器中，对应着的模拟电压输入就和内部变换电路接通。

3、工作原理

- 为了启动变换，必须在START端加一个正跳变信号，此后变换就开始了。
- 当EOC为低电平时，表示正在变换中，当EOC由低变高时，表示变换结束。
- 此时，只要在OE端加一个高电平，即可开启三态缓冲器，从数据线读数据

回到起点

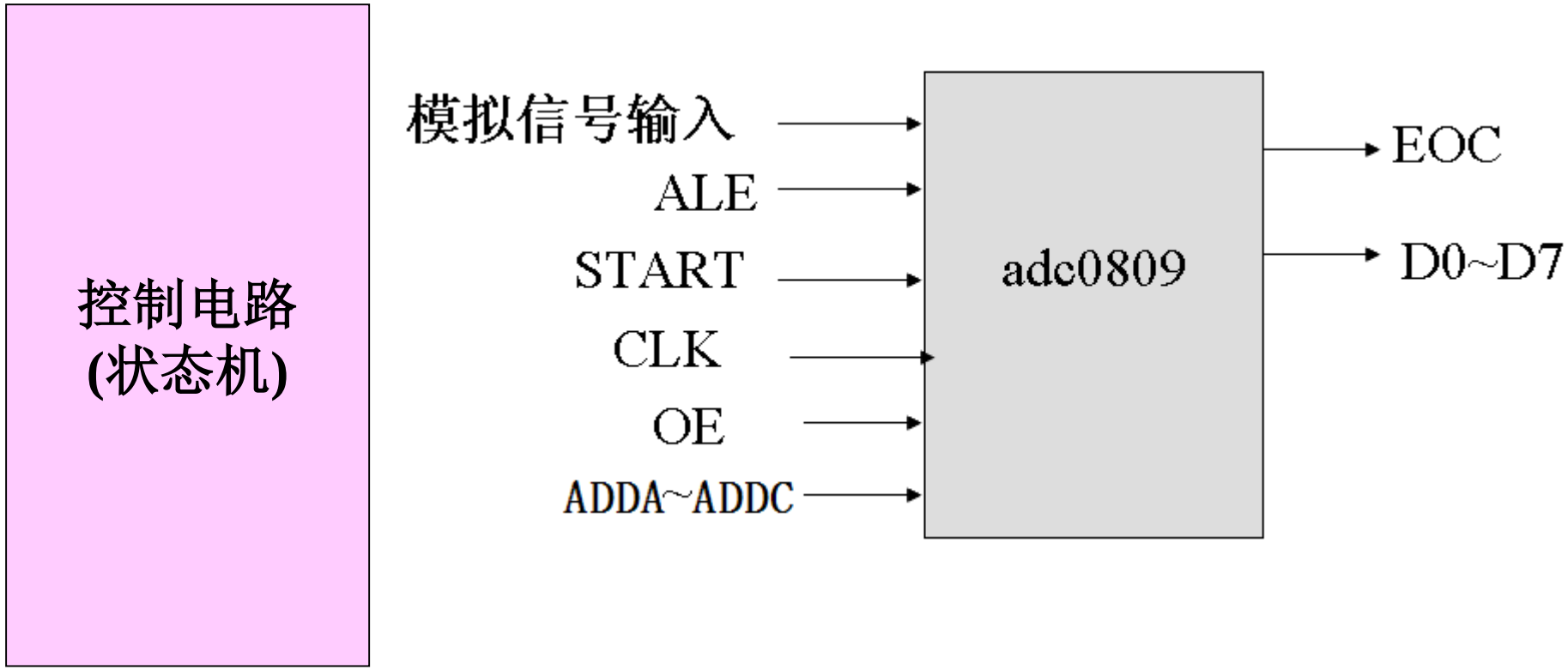
- 第一步：构思电路框架



第三步：FPGA编程？

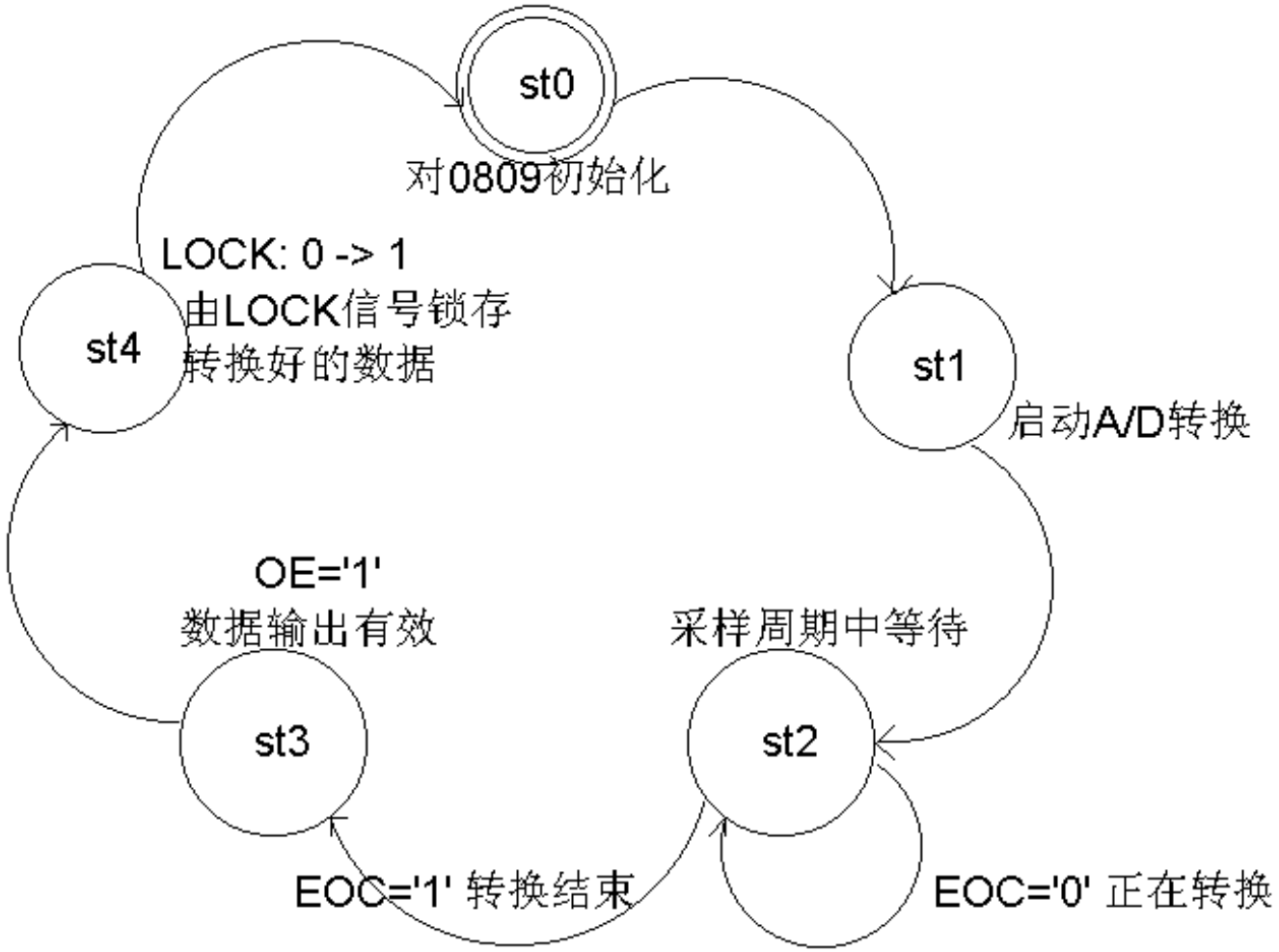
- 1、画出电路的端口信息？

第三步、 两个模块之间的连接以及两者之间工作的协调关系



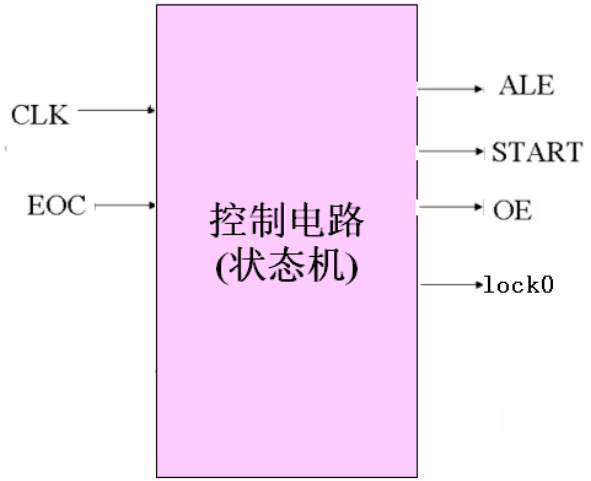
画出FPGA控制电路的状态转换图

控制电路简化状态机

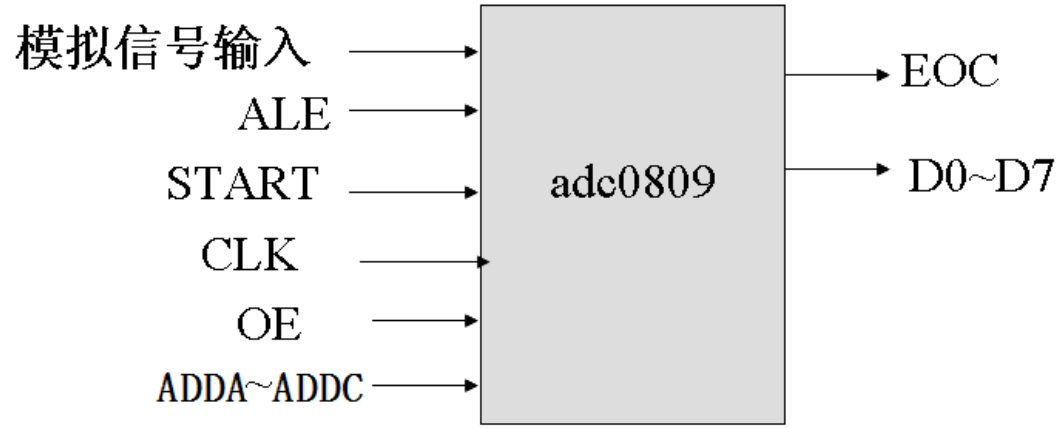

st0
对0809初始化
st1
启动A/D转换
st2
采样周期中等待
EOC='0' 正在转换
st3
EOC='1' 转换结束
st4
OE='1'
数据输出有效
LOCK: 0 -> 1
由LOCK信号锁存
转换好的数据

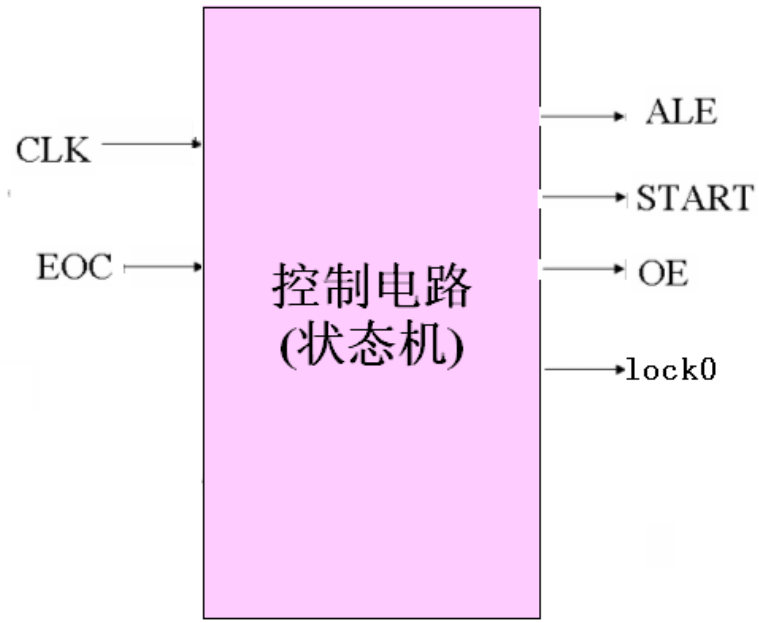
第四步：FPGA编程

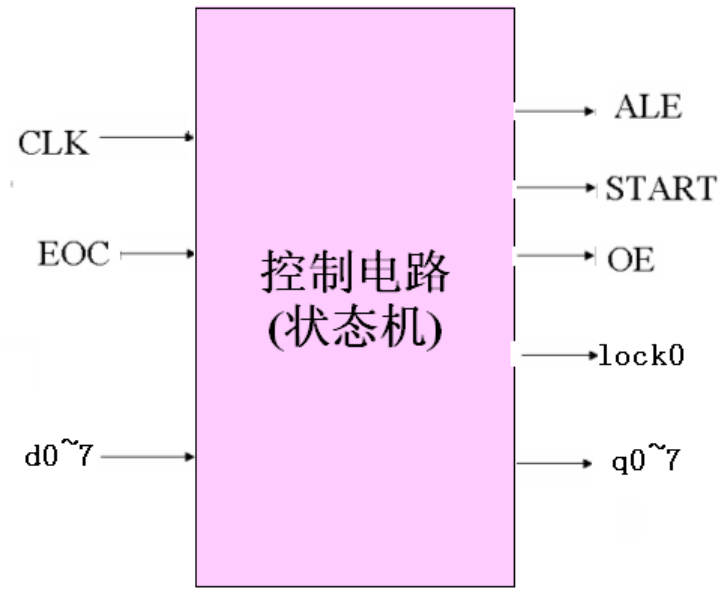
- 1、画出电路的端口信息



- 这里的CLK是状态机工作频率
- 越高越好


模拟信号输入
ALE
START
CLK
OE
ADDA~ADDC
adc0809
EOC
D0~D7


CLK
EOC
控制电路
(状态机)
ALE
START
OE
lock0


CLK
EOC
d0~7
控制电路
(状态机)
ALE
START
OE
lock0
q0~7

状态机的Verilog HDL设计步骤

- 1、已知状态机
- 2、首先画出时序逻辑电路的结构框图
- 3、画出电路端口信息
- 4、画时序图（详细了解状态的转换过程）
- 5、编程

第四步：FPGA编程

- 2、电路工作情况

- 对状态机的进一步了解或完善状态机

第四步：FPGA编程

- 3、状态机的结构框图
-
- 4、较详细的电路连接图

10.2.1 ADC采样控制设计及多过程结构状态机

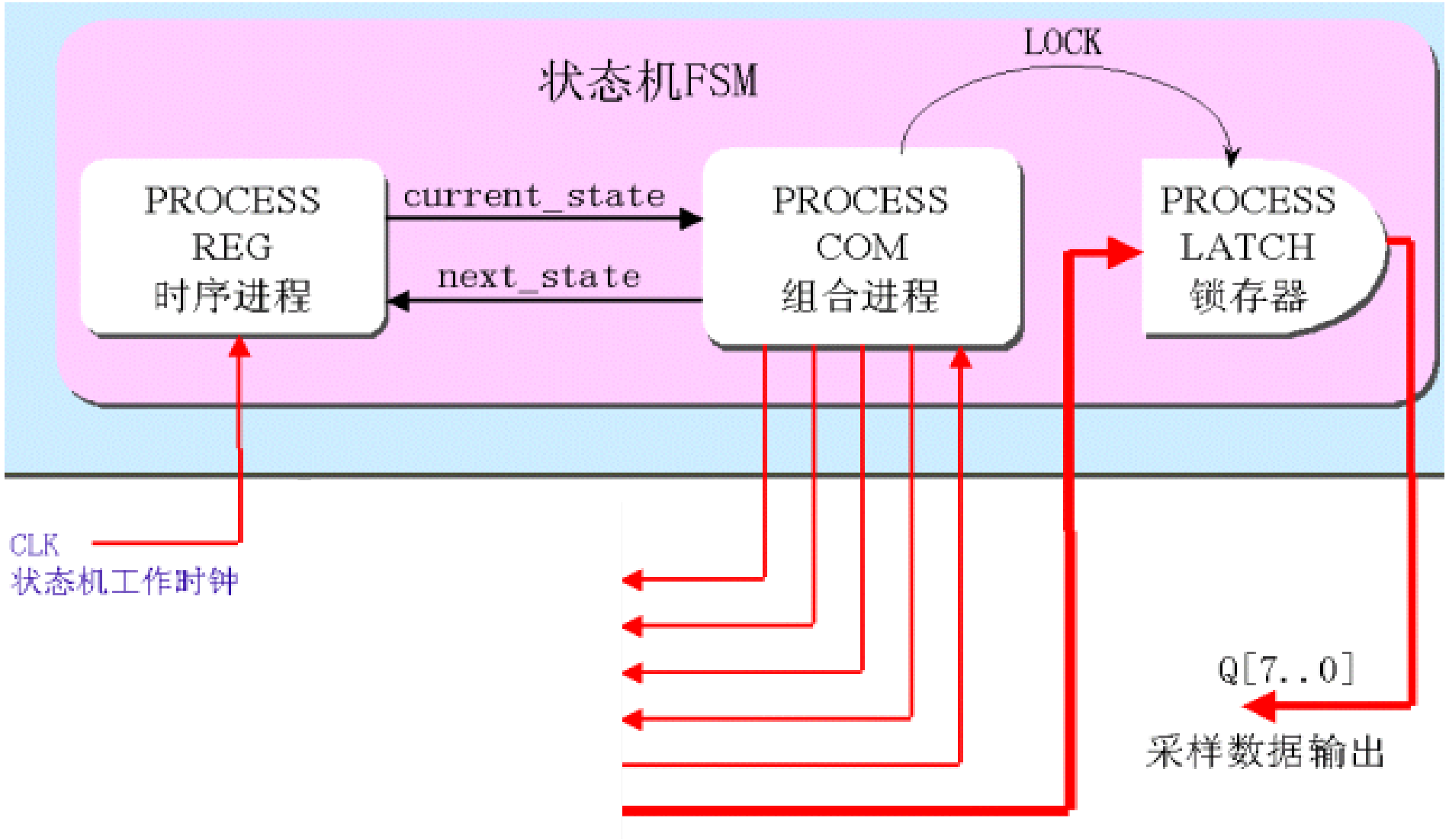


图7-6 采样状态机结构框图

准备工作

- 电路图
- 状态图
- 结构框图
- 电路端口信息
- 详细了解状态的转换过程

5、编程

- **P254**

【例 8-2】为了仿真方便观察，输出口增加了内部锁存信号 LOCK_T

```
module ADC0809 (D, CLK, EOC, RST, ALE, START, OE, ADDA, Q,LOCK_T);
  input[7:0] D;              //来自0809转换好的8位数据
  input CLK,RST;             //状态机工作时钟,和系统复位控制
  input EOC;                 //转换状态指示，低电平表示正在转换
  output ALE;                //8个模拟信号通道地址锁存信号
  output START,OE ;          //转换启动信号，和数据输出三态控制信号
  output ADDA,LOCK_T ;       //信号通道控制信号和锁存测试信号
  output[7:0] Q;
  reg ALE, START, OE;
  parameter s0=0,s1=1,s2=2,s3=3,s4=4;  //定义各状态子类型
  reg[4:0] cs , next_state ;           //为了便于仿真显示，现态名简为cs
  reg[7:0] REGL;  reg LOCK;            // 转换后数据输出锁存时钟信号
  always @(cs or EOC)  begin           // 组合过程，规定各状态转换方式
     case (cs)
       s0 : begin  ALE=0 ; START=0 ; OE=0 ; LOCK=0 ;
```

接下页

接上页

```
                next_state <= s1 ;   end          //0809初始化
        s1 : begin  ALE=1 ; START=1 ; OE=0 ; LOCK=0 ;
                next_state <= s2 ;    end         //启动采样信号START
        s2 : begin  ALE=0 ; START=0 ; OE=0 ; LOCK=0 ;
           if (EOC==1'b1)  next_state = s3 ;      //EOC=0表明转换结束
             else  next_state = s2 ;  end          //转换未结束，继续等待
        s3 : begin ALE=0 ; START=0 ; OE=1;  LOCK=0; //开启OE，打开AD数据口。
                next_state = s4 ;     end         //下一状态无条件转向s4
        s4 : begin ALE=0 ;  START=0 ; OE=1;  LOCK=1; //开启数据锁存信号
                next_state <= s0 ;     end
        default : begin  ALE=0 ; START=0 ; OE=0 ; LOCK=0 ;
                next_state = s0 ;     end
     endcase    end
   always @(posedge CLK or  posedge RST)  begin  //时序过程
     if (RST) cs <= s0 ;
       else  cs <= next_state ; end // 由现态变量cs将当前状态值带出过程
   always @(posedge LOCK)                        //寄存器过程
     if (LOCK)   REGL <= D ; // 此过程中，在LOCK的上升沿将转换好的数据锁入
   assign ADDA =0 ; assign Q = REGL ; //选择模拟信号进入通道IN0
   assign LOCK_T = LOCK ;  //将测试信号输出
endmodule
```

10.2.2　序列信号检测器

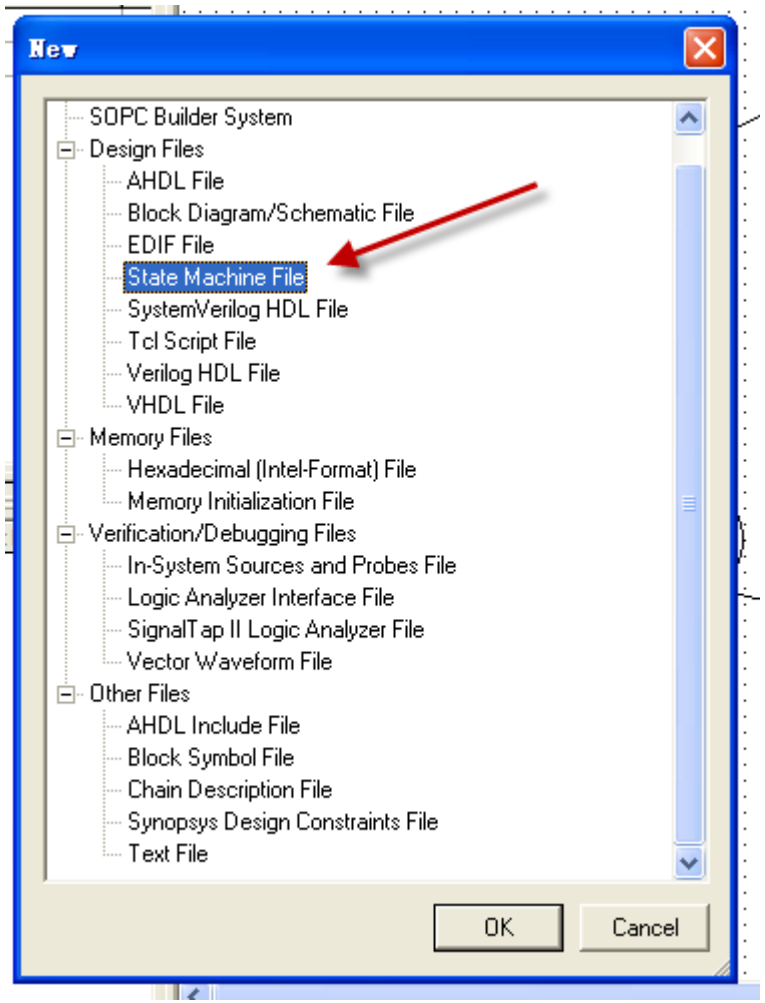
- 检测脉冲序列信号
- 例：检测11010011

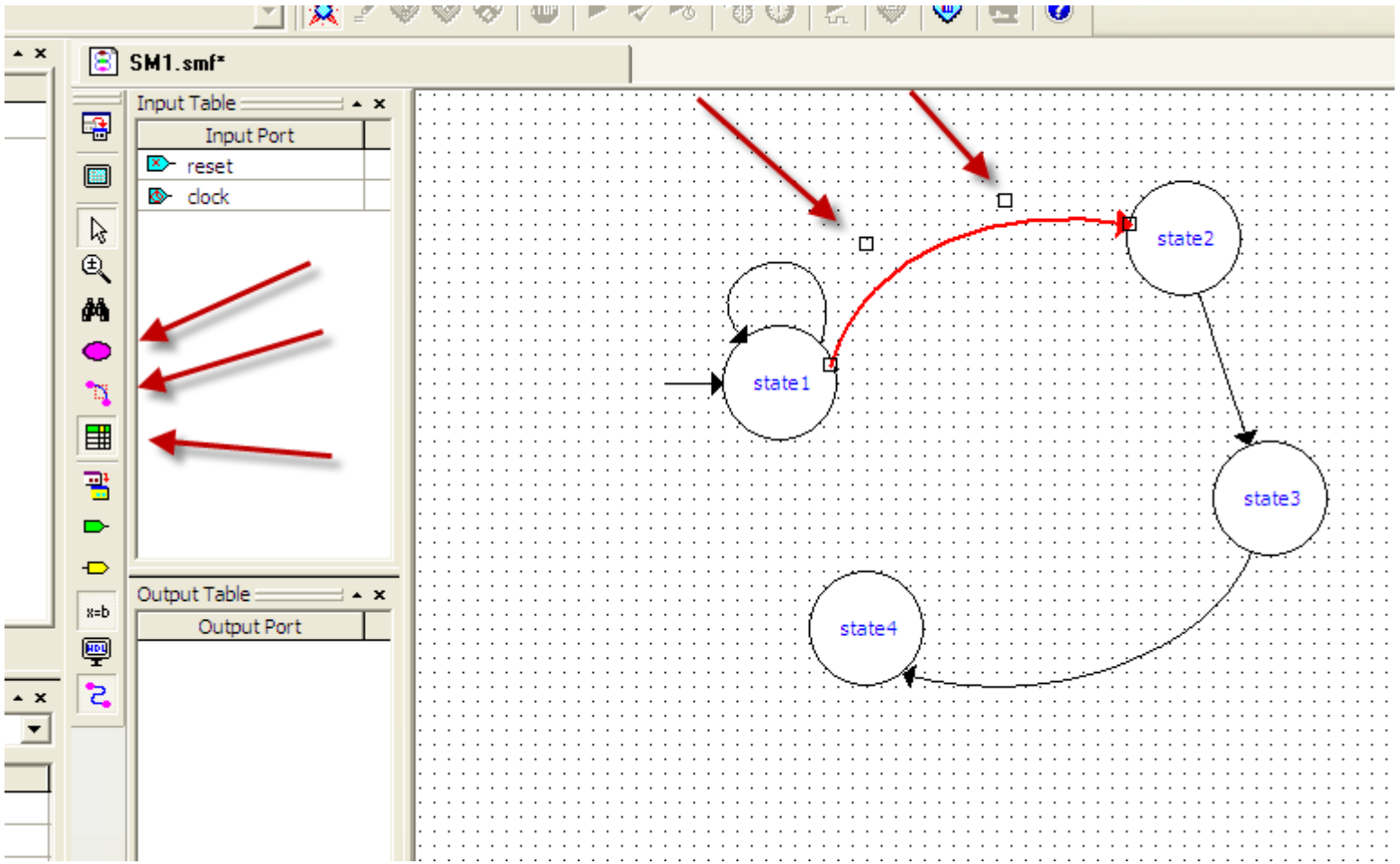
- RST　0->1->0
- CLK
- DIN　　1->1->0->1->0->0->1->1->
- SOUT　　0->0->0->0->0->0->0->1
- State　　0->1->2->3->4->5->6->7

10.3 Mealy型状态机

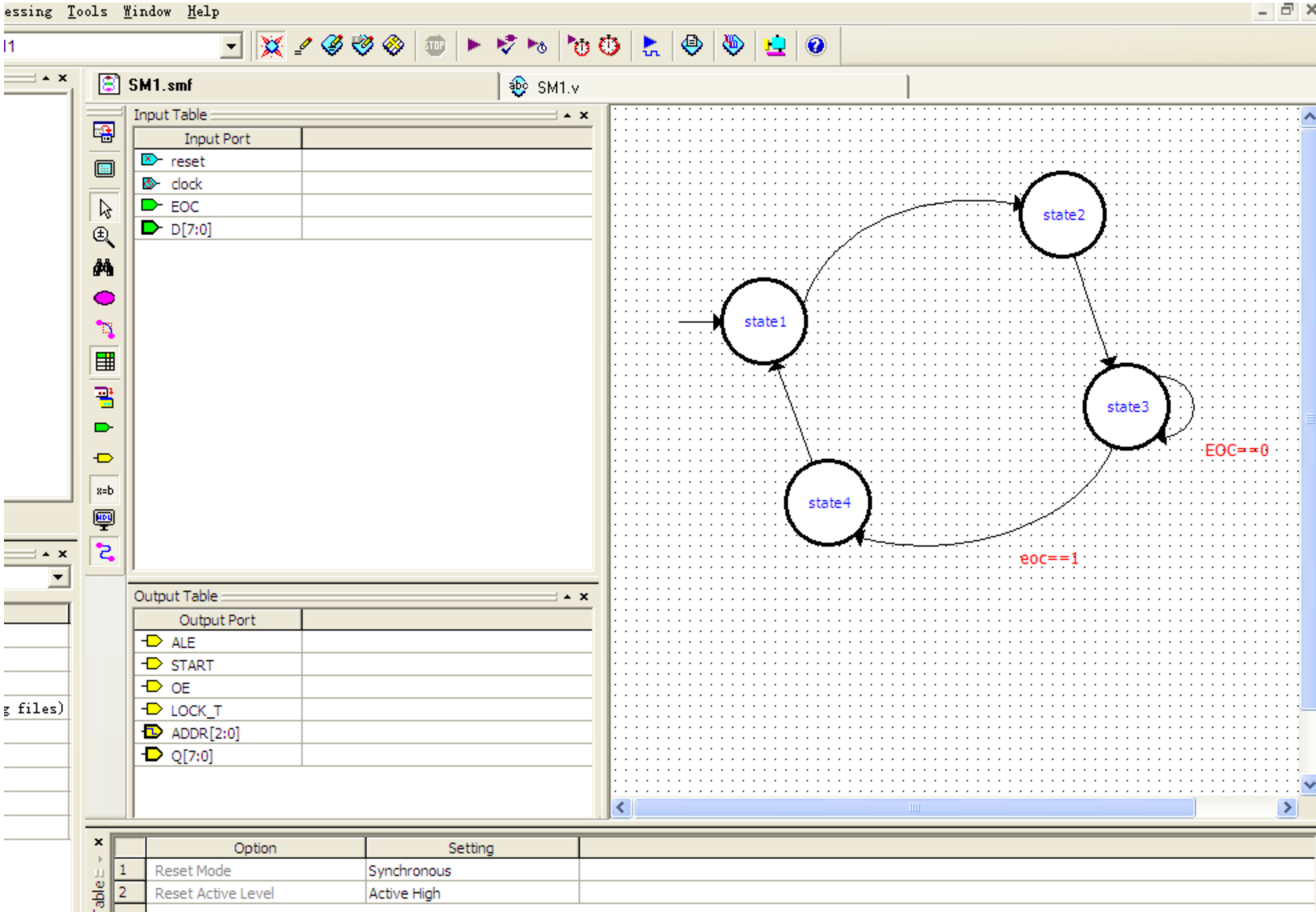
- 输出变化快一个时钟
- 输出是当前状态与当前输入的函数
- 读教材P259例10—5，画状态转移图

10.4 状态机图形编辑设计

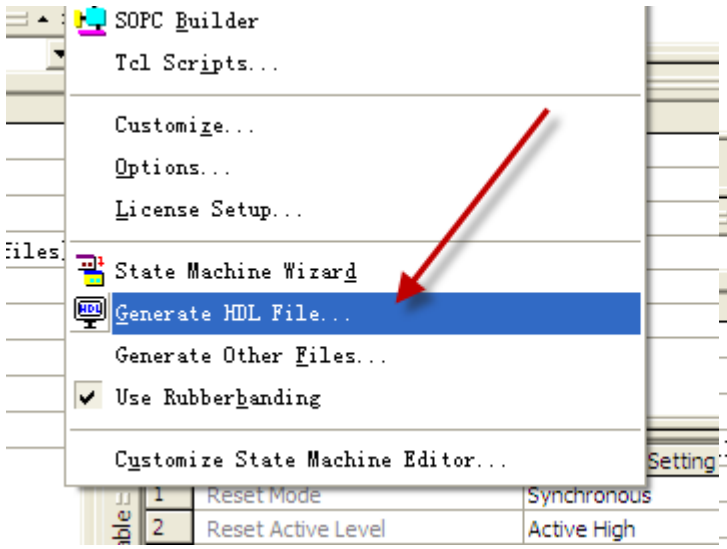




例：ADC0809状态机



将状态转移图转换为代码



SM1.smf | SM1.v

```
19
20  `timescale 1ns/1ns
21
22 module SM1 (
23      reset,clock,EOC,
24      ALE,START,OE,LOCK_T,ADDR[2:0]);
25
26      input reset;
27      input clock;
28      input EOC;
29      tri0 reset;
30      tri0 EOC;
31      output ALE;
32      output START;
33      output OE;
34      output LOCK_T;
35      output [2:0] ADDR;
36      reg ALE;
37      reg START;
38      reg OE;
39      reg LOCK_T;
40      reg [2:0] ADDR;
41      reg [2:0] reg_ADDR;
42      reg [3:0] fstate;
43      reg [3:0] reg_fstate;
44      parameter state1=0,state2=1,state3=2,state4=3;
```

10.5 不同编码类型状态机

- 1.直接输出型
- 2.宏定义编码
- 3.顺序编码
- 4.一位热码
- 状态编码设置

10.6 异步有限状态机设计

- 见P270例

10.7 安全状态机设计

- 对非法状态的处理

- 状态导引法
- 编码监测法

- EDA可部分帮助

10.8 硬件数字技术排除毛刺

- 毛刺的产生

- 延时方式去除毛刺
- 逻辑方式去除毛刺

end