EDA技术高级应用

哈尔滨工业大学（威海）



目　录

CONTENTS

哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

第4章

有限状态机

群名称:EDA技术高级应用
群　号:1044047663

实现控制功能，可采用有限状态机实现和CPU实现；

- CPU：需要许多操作（例如取数和执行）和部件（例如数据通路和ALU寄存器）；
- 有限状态机：存储在多个触发器中，表示行为的代码存储在门级网络中；

```
1  if a>37 and c<7 then
2       state <= alarm;
3       out_a <= '0';
4       out_b <= '0';
5       out_analog <= a+b;
6  else
7       state <= running;
8   end if;
```

CPU执行：10~20条机器指令，执行时间在最值间波动；

状态机执行：由门和触发器执行，执行时间为一个机器周期；

4.1 状态机设计相关语句

4.1.1 类型定义语句

```
type  数据类型名 is  数据类型定义 of 基本数据类型;
或
type  数据类型名 is  数据类型定义;
type st1 is array ( 15 downto 0 ) of std_logic ;
type week is (sun，mon，tue，wed，thu，fri，sat);
type m_state is （st0，st1，st2，st3，st4，st5）;
signal present_state，next_state ： m_state ;
```

4.1.2 状态机的优势

1、状态机克服了纯硬件数字系统顺序方式控制不灵活的缺点；

2、由于状态机的结构相对简单，设计方案相对固定；

3、状态机容易构成性能良好的同步时序逻辑模块；

4、状态机的VHDL表述丰富多样、有其独到的好处；

5、在高速运算和控制方面，状态机更有其巨大的优势；

6、高可靠性。

4.1.3 状态机结构

- 说明部分
- 主控时序进程
- 主控组合进程
- 辅助进程

1. 说明部分　　定义枚举型数据类型，定义状态变量。

说明部分一般放在architecture 和begin之间，例如：

```
architecture ……is……
type FSM_ST is (s0，s1，s2，s3);
signal current_state, next_state : FSM_ST;

…
```

4.1.3 状态机结构

2. 主控时序进程

主控时序进程主要负责状态机运转和在时钟驱动下负责状态的转换。

状态机是随外部时钟信号，以同步时序方式工作的。

3. 主控组合进程

也称为状态译码进程，其任务是根据外部输入的控制信号，或和当前状态的状态值确定下一状态的取向，以及确定相应的输出。

4. 辅助进程

用于配合状态机工作的组合或时序进程。

4.1.4 状态机的设计

- 在产生输出的过程中，由**是否使用输入信号**可以决定状态机的类型；
- 两种类型

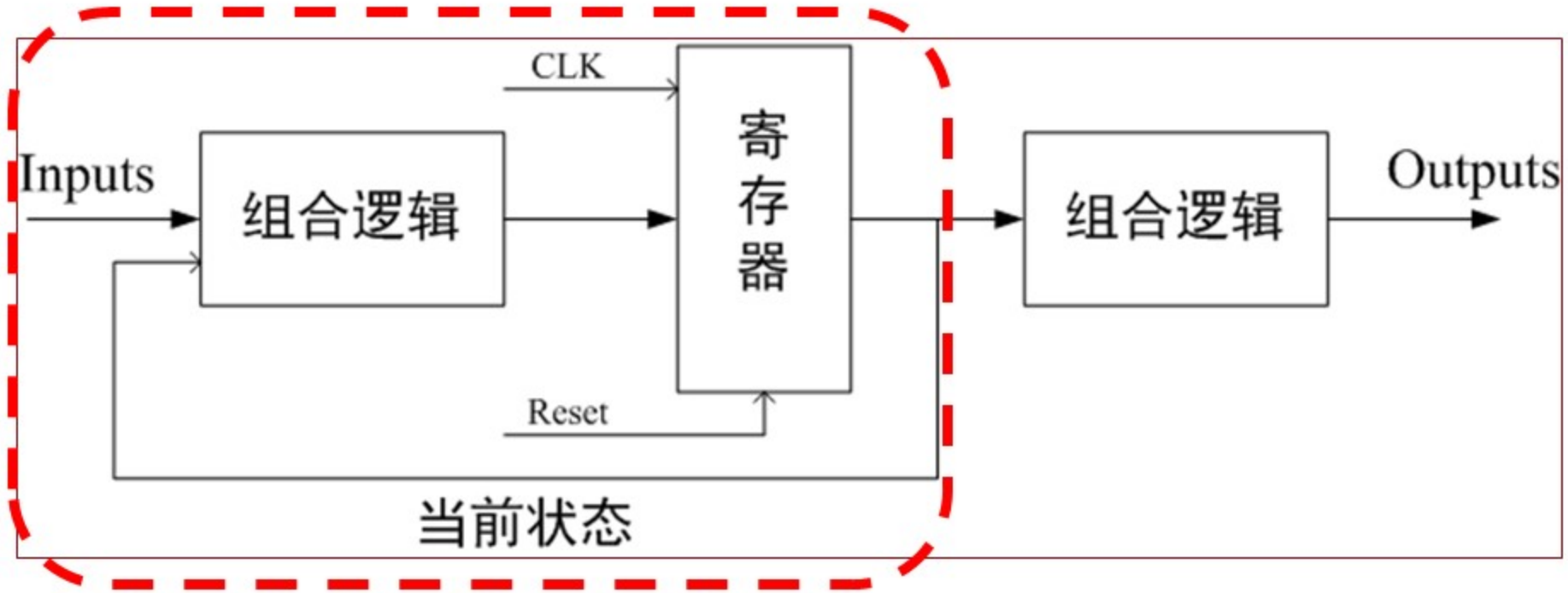
1. 莫尔（Moore）状态机：

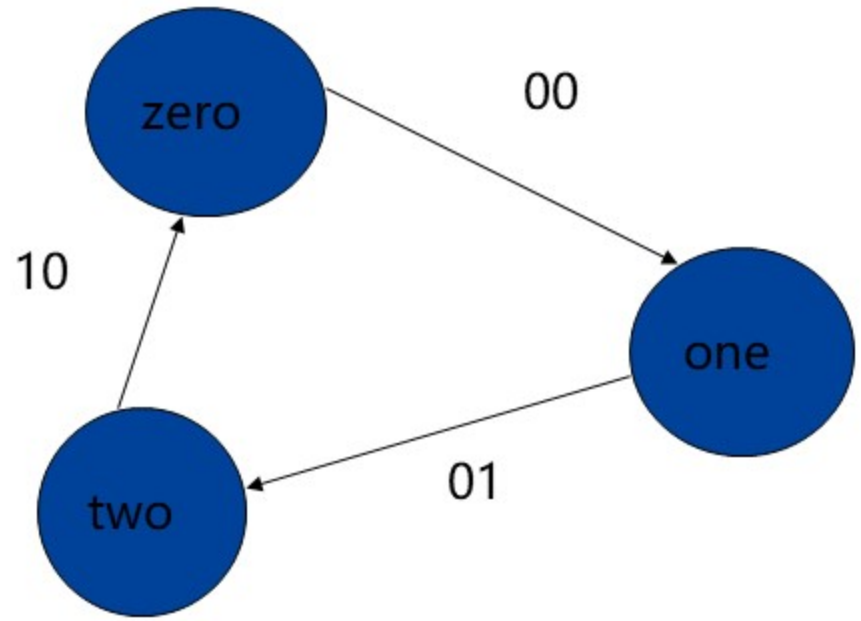
   输出仅是**当前状态**的函数；

2. **米里（Mealy）状态机**：

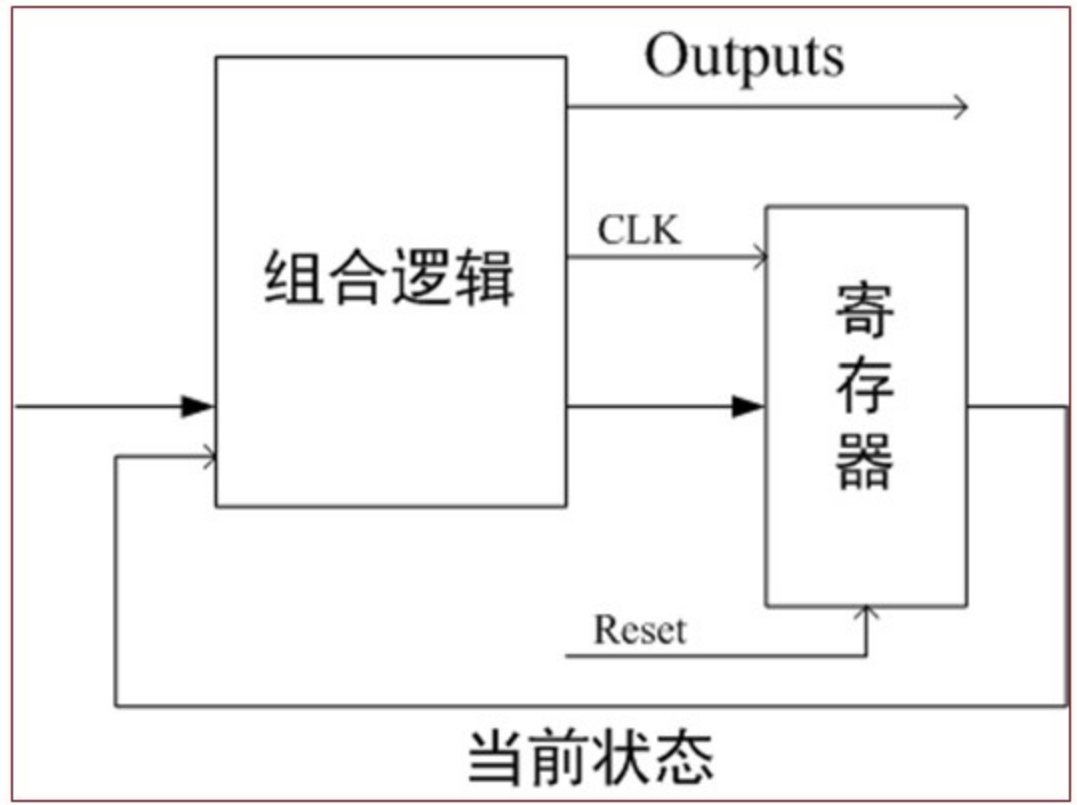
   输出是**当前状态**和**所有输入信号**的函数；

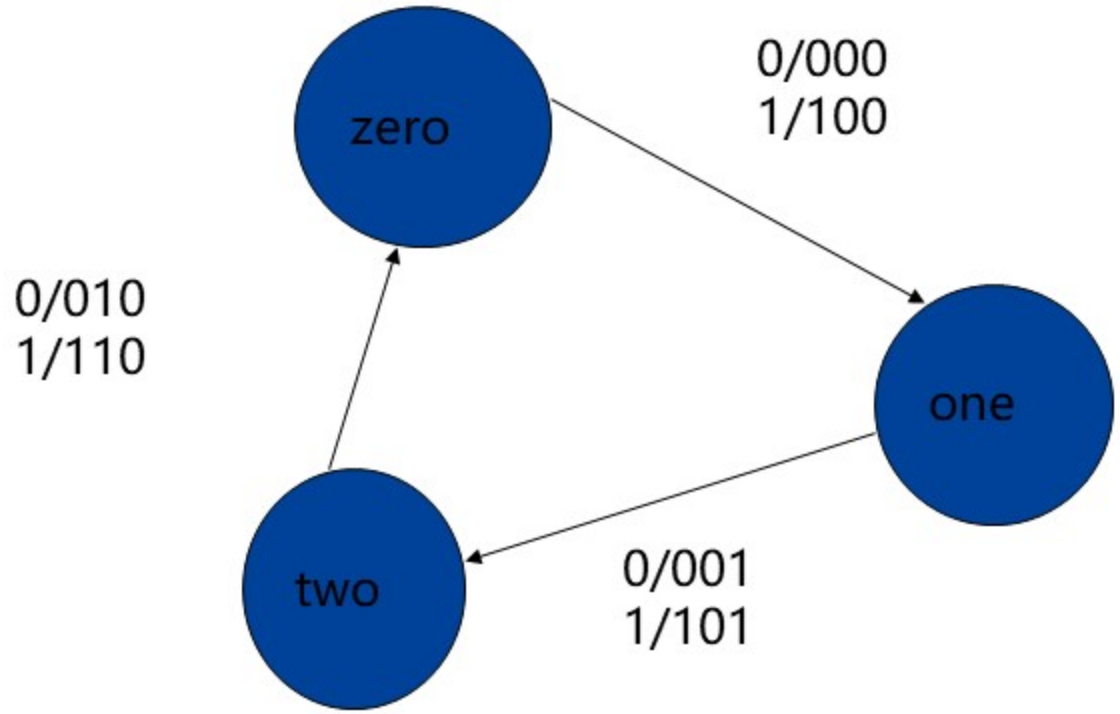
4.2 Moore状态机




zero
00
one
10
two
01

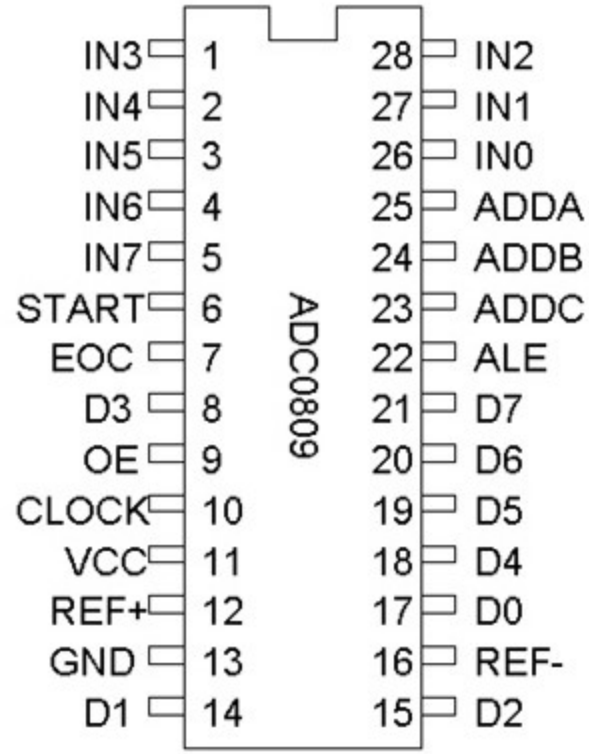
4.3 Mealy状态机



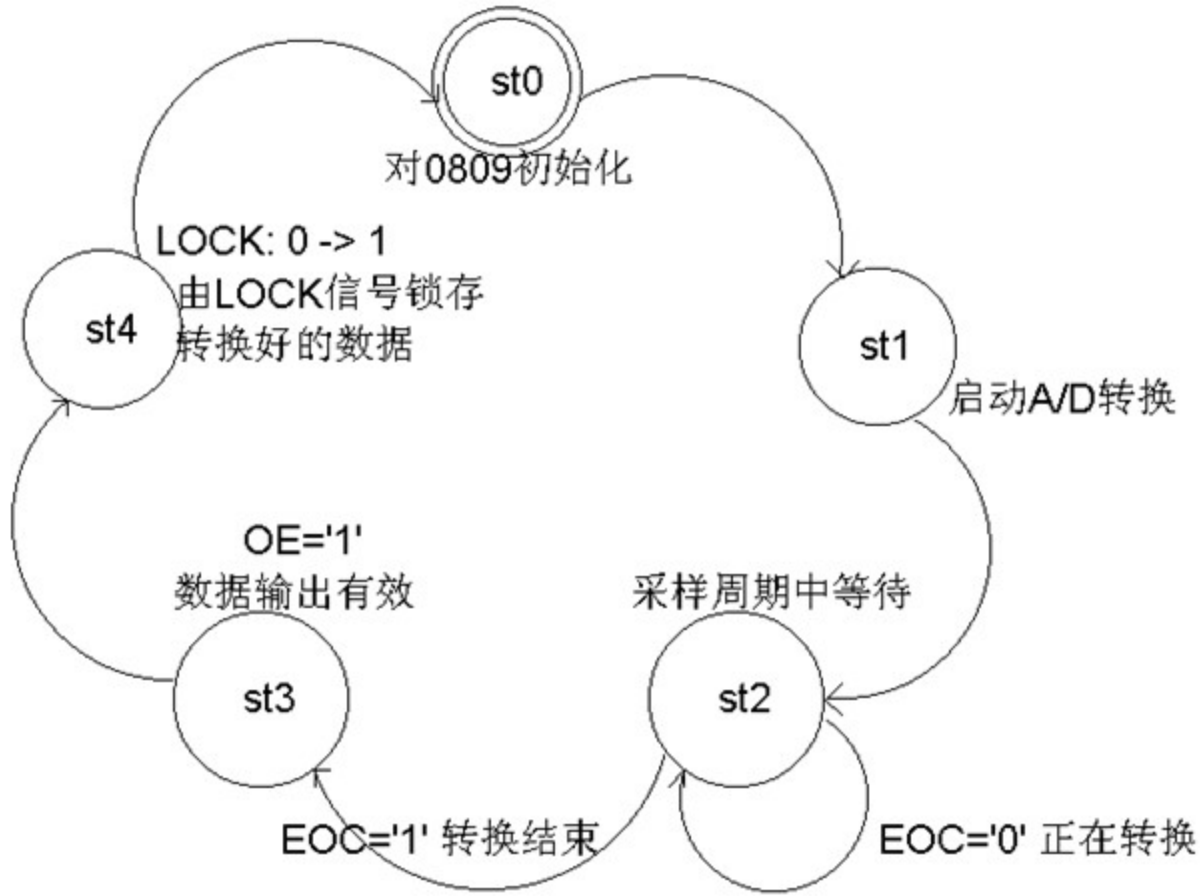

zero
one
two
0/000
1/100
0/010
1/110
0/001
1/101

4.3.1 多进程状态机

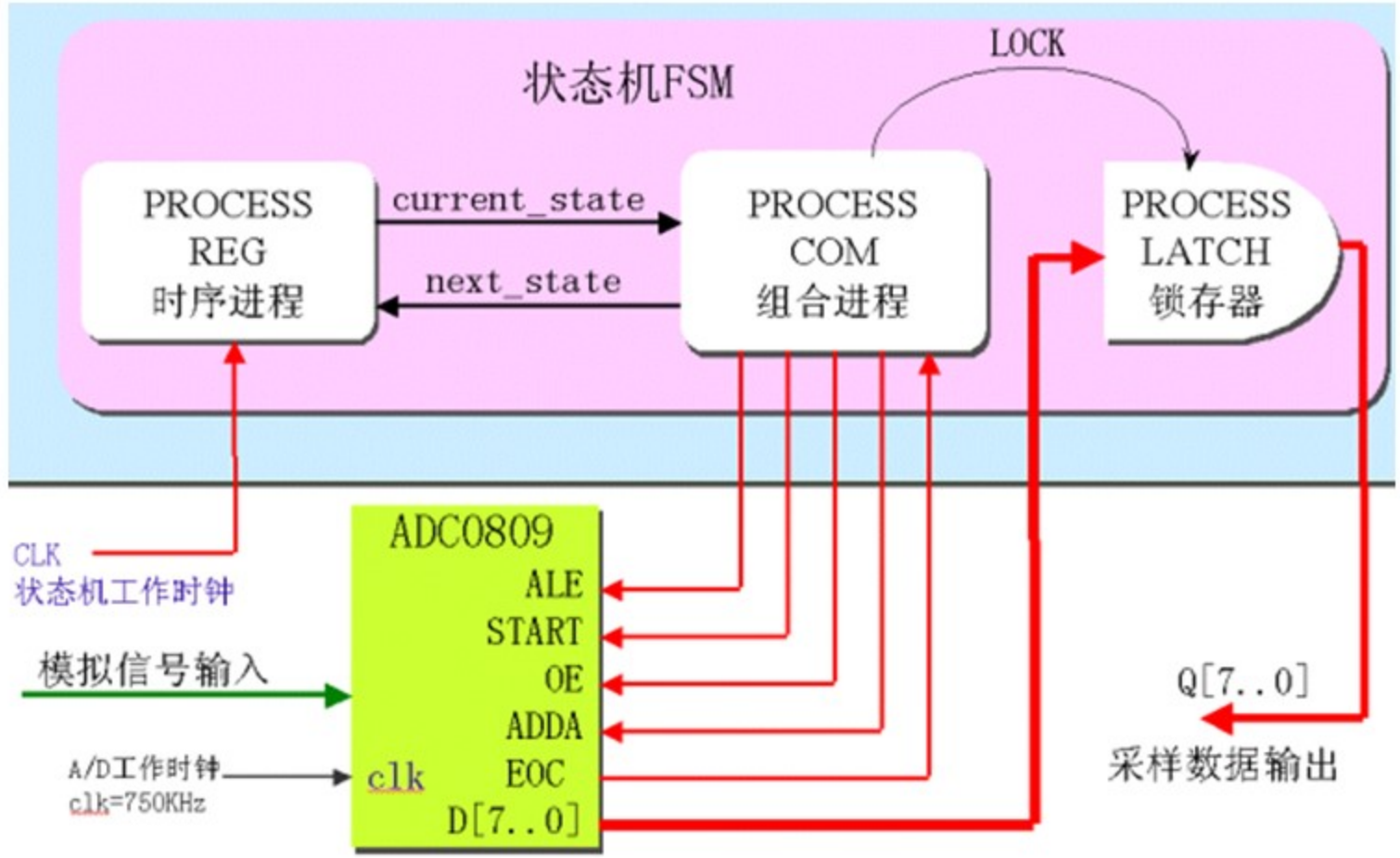
用状态机设计一个A/D采样控制器为例



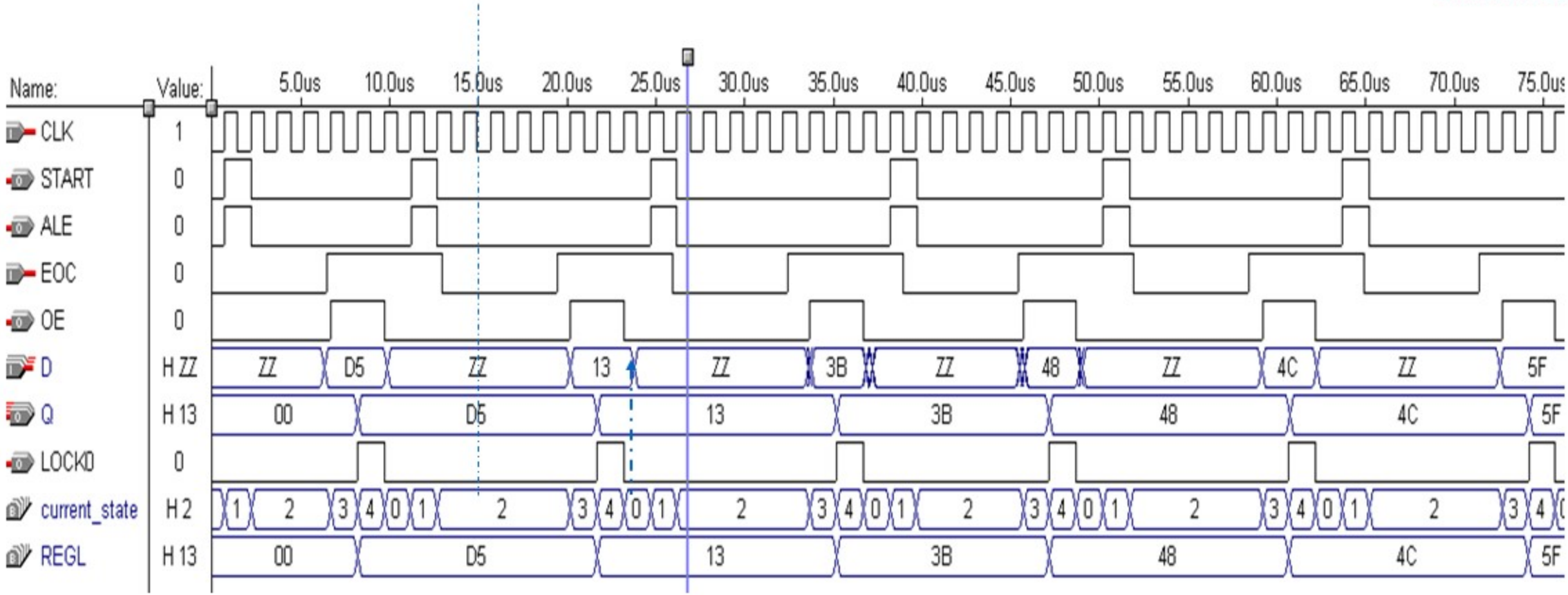
ADC0809工作时序



控制ADC0809采样状态图


状态机FSM
LOCK
PROCESS
REG
时序进程
current_state
next_state
PROCESS
COM
组合进程
PROCESS
LATCH
锁存器
CLK
状态机工作时钟
ADC0809
ALE
START
OE
ADDA
EOC
D[7..0]
clk
模拟信号输入
A/D工作时钟
clk=750KHz
Q[7..0]
采样数据输出

采样状态机结构框图



ADC0809采样状态机工作时序

```
COM1: PROCESS(current_state,EOC)  --负责状态转换
  BEGIN
   CASE current_state IS
   WHEN st0=> next_state <= st1;
   WHEN st1=> next_state <= st2;
   WHEN st2=> IF (EOC='1') THEN next_state <= st3;
                ELSE next_state <= st2; END IF ;
   WHEN st3=> next_state <= st4;--开启OE
   WHEN st4=> next_state <= st0;
   WHEN OTHERS => next_state <= st0;
   END CASE ;
 END PROCESS COM1 ;
COM2: PROCESS(current_state)      --负责状态译码
  BEGIN
   CASE current_state IS
   WHEN st0=>ALE<='0';START<='0';LOCK<='0';OE<='0' ;
   WHEN st1=>ALE<='1';START<='1';LOCK<='0';OE<='0' ;
   WHEN st2=>ALE<='0';START<='0';LOCK<='0';OE<='0' ;
   WHEN st3=>ALE<='0';START<='0';LOCK<='0';OE<='1' ;
   WHEN st4=>ALE<='0';START<='0';LOCK<='1';OE<='1' ;
   WHEN OTHERS => ALE<='0';START<='0';LOCK<='0';
   END CASE ;
 END PROCESS COM2;
```

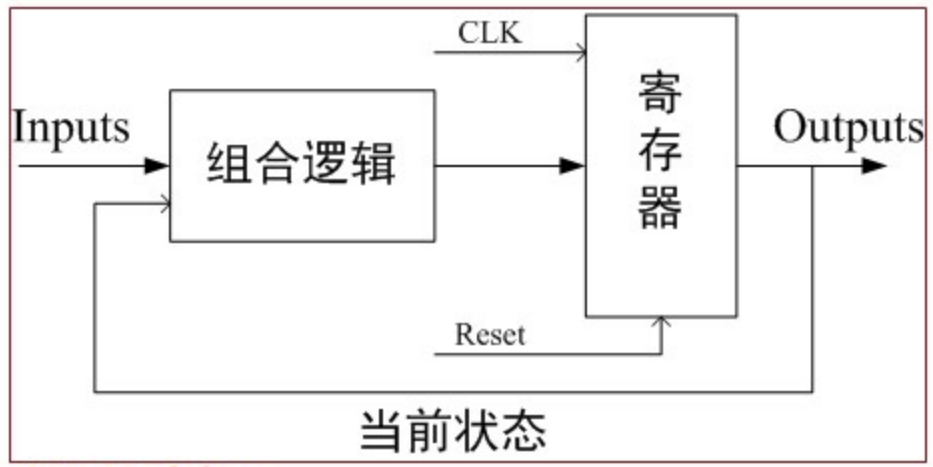
4.4 可以消除毛刺的状态机

无论是Mealy型还是Moore型状态机，其输出信号都可能有“毛刺”，因为它们的输出信号都来自组合逻辑。

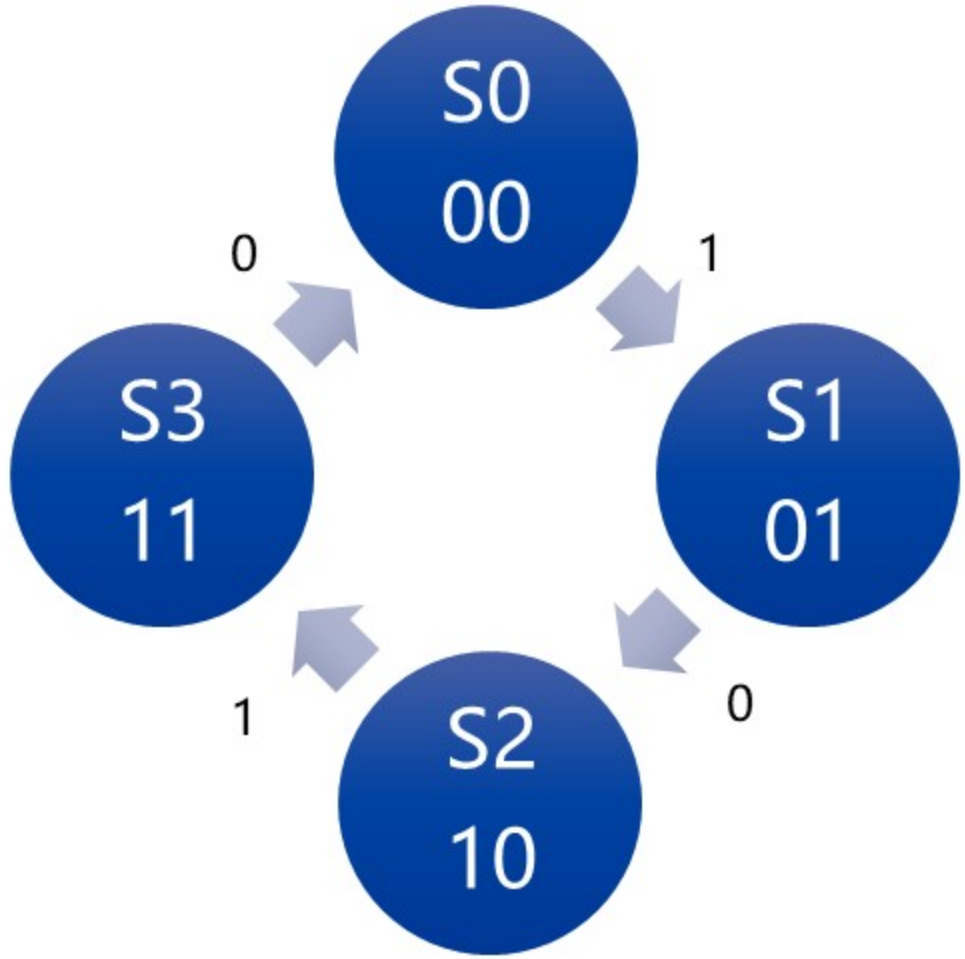
- 1.直接把状态作为输出信号；
- 2.在Moore型状态机基础上，使用时钟同步输出信号；
- 3.在Mealy型状态机基础上，使用时钟同步输出信号；

直接把状态作为输出信号

- 直接把状态作为输出信号(output=state)，是状态机的一种特殊类型。必须在VHDL源码中**对状态编码加以明确规定**，使状态和输出信号的取值一致。



- 输出译码电路已被优化掉！


S0
00
1
S1
01
0
S2
10
1
S3
11
0

小 结

- 状态机的结构：
  - A、组合逻辑部分（状态译码器和输出译码器）
  - B、寄存器部分
- 各部分的功能
  - 1、状态译码器
    - 确定状态机的下一个状态
  - 2、输出译码器
    - 确定状态机输出
  - 3、状态寄存器
    - 存储状态机的内部状态