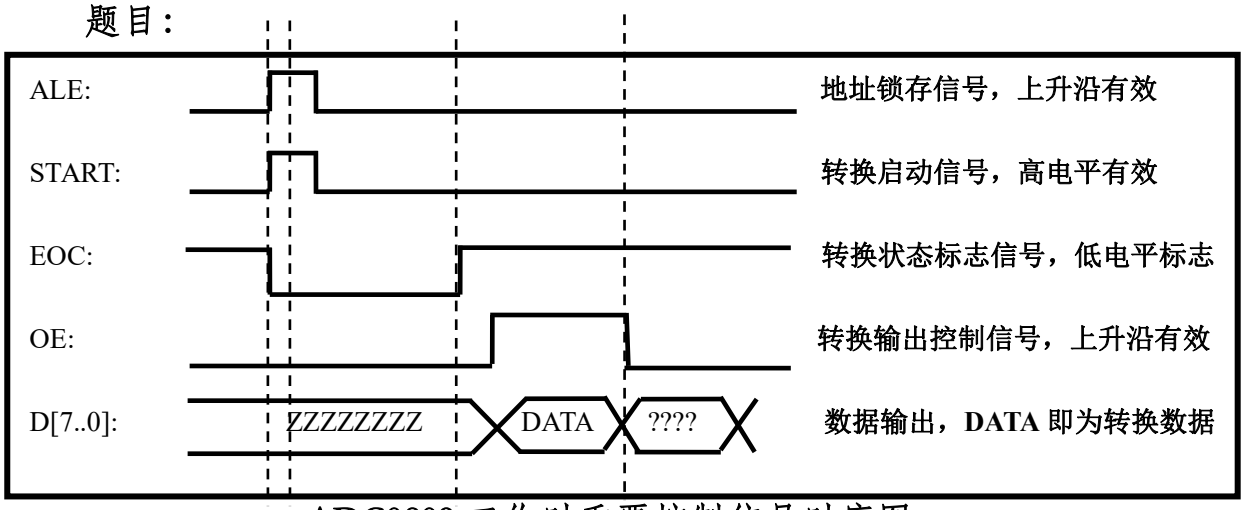
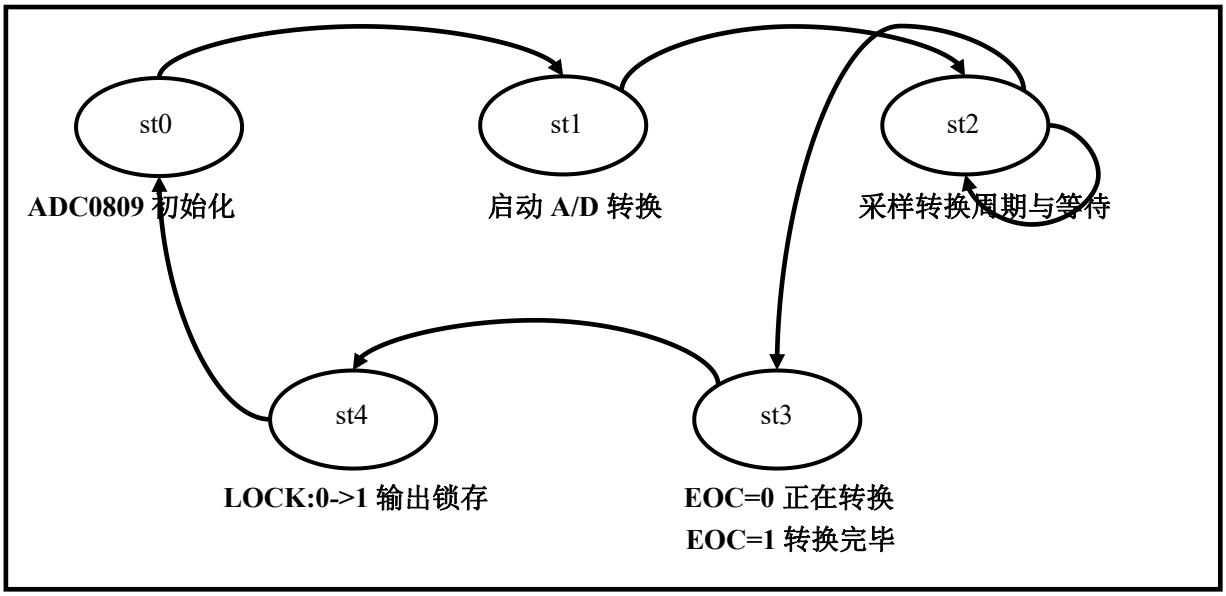
ADC0809 采样状态机设计大题分析：

题目：

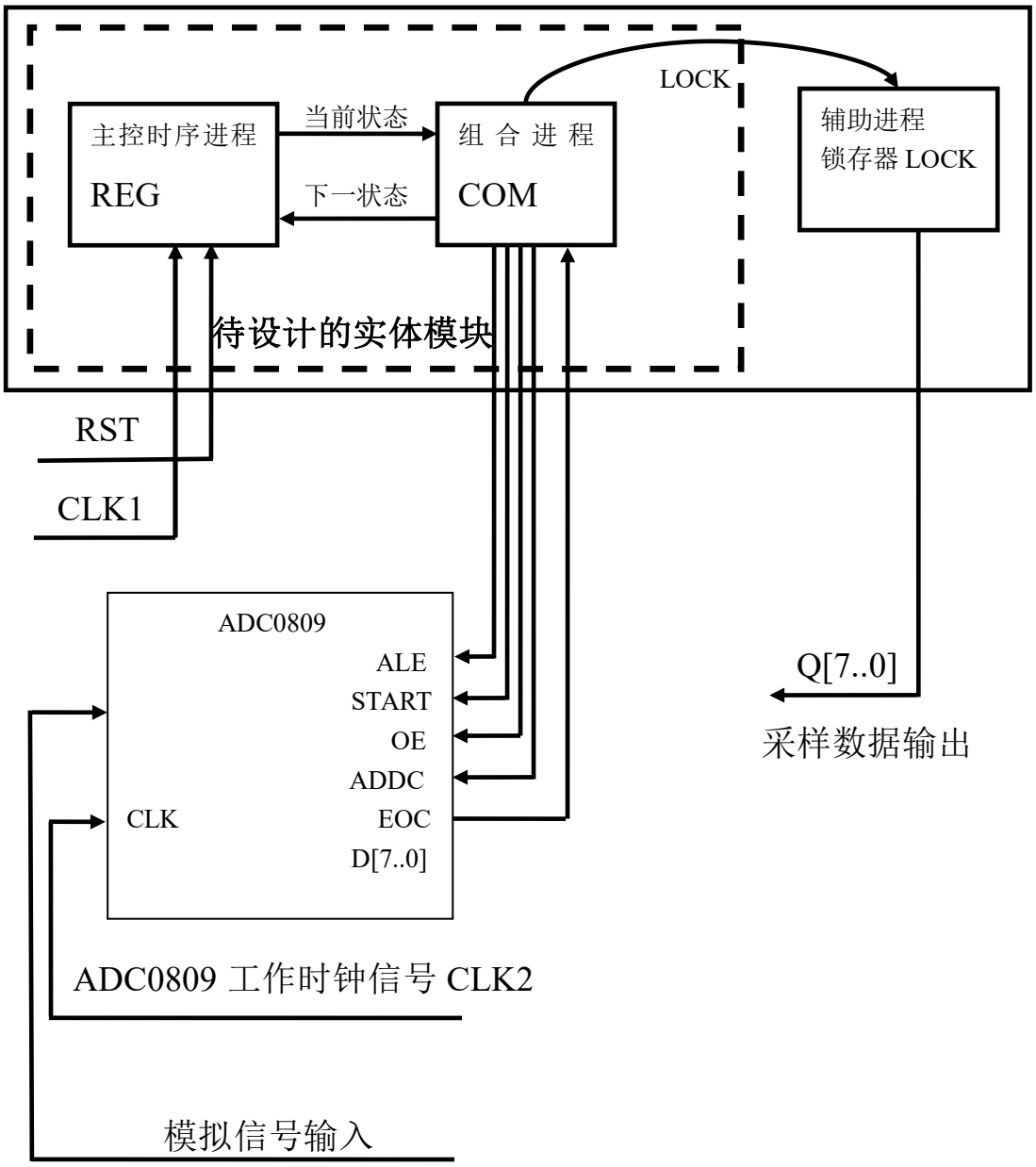


ADC0809 工作时重要控制信号时序图



控制 ADC0809 采样状态图

对照时序图分析：st0、st1、st2、st3、st4 状态时 ALE、START、EOC、OE、D[7..0]的信号值。



程序设计思路：

①主控时序进程 REG：检测复位信号 RST 是否为 1，如果为 1，当前状态变为 S0，否则，来一个 CLK1 上升沿，就把下一状态信号 NS 送给当前状态信号 CS，这就是主控时序进程的功能：负责状态机的状态变换，给出状态更换信号，这个状态更换信号为主控组合进程的敏感信号，也就是说，主控时序信号负责简介驱动主控组合进程。

②主控组合进程 COM：检测敏感信号 CS，和来自 ADC0809 转换完成标志信号 EOC，如果 E0C='1'，说明转换完成（请注意，这里指经过转换过程后的 EOC 电平），状态转换为 S3，否则停留在 S2。这个进程是这样执行的，最开始时当前状态为 S0，S0 在主控时序进程 REG 中由于 CLK1 的作用要变为"NS",S0 要变为"NS"，主控组合进程 COM 敏感信号 CS 有效，驱动 COM 进行，于是到 CASE 语句，为 S0 时，将有：

ALE<='0';START<='0';OE<='0';LOCK<='0';NS<='S1'; 这样 CS<=S1 了

这样 ALE、START、OE、LOCK 就会在 COM 进程结束时被赋值，且 ALE、START、OE 这几个信号可以传到 ADC0809 转换器中，LOCK 可以传到 LATCH 锁存器中，如果信号

有效，那么相应的硬件（转换器、锁存器）将执行对应的功能。了解到这一点后，就好分析了。紧接着下一个 CLK1 上升沿到来，CS=”S1”又要变为 CS<=”NS”，COM 被启动，CASE 语句 S1 满足，ALE<=’1’;START<=’0’;OE<=’0’;LOCK<=’0’;NS<=’S2’;这样 CS<=S2 了，而 ALE 信号从之前的’0’变为现在的’1’，产生了上升沿，地址锁存有效，START 也变为高电平’1’了，ADC0809 被启动，数据转换在硬件里自动开始，OE、LOCK 均无变化且无效；紧接着下一个 CLK1 上升沿到来，CS=”S2”又要变为 CS=”NS”，COM 又被启动，CASE 语句 S2 满足，则 ALE<=’0’;START<=’0’;OE<=’0’;LOCK<=’0’;在这条语句之后有一个关于 EOC 的判断语句，这是因为 EOC 是待设计模块的一个输入信号，当 ADC0809 开始转换数据时，EOC 变为低电平，此后一直保持为低电平，直到 ADC0809 转换完数据时，EOC 自动变为 1，因此检测 EOC 可以监测数据转换的进度情况，在关于 EOC 的 IF 语句判断中，如果 E0C=’1’，说明转换完成，那么 NS<=S3，这样 CS<=S3 了；否则，NS<=S2，则 CS 也停留在 S2，这样一直循环，直到转换完数据，EOC 变为 1 才结束这个循环。紧接着，CLK1 再来一个上升沿，CS=”S3”要变为 CS=”NS”，COM 启动，CASE 语句 S3 满足，则
ALE<=’0’;START<=’0’;OE<=’1’;LOCK<=’0’;NS<=’S4，这样 CS<=S4 了，并且，OE 信号由之前的’0’变为了现在的’1’，产生上升沿，看清楚了，OE 是待设计模块输出传给 ADC0809 的，所以 ADC0809 被控制，输出了转换的 8 位数据，这个地方在时序图上有点延迟的原因在于信号的传输延时，ADC0809 是在 E0C 为’1’之后才被控制要输出转换后的数据，那么这个过程是有时间差的，XYZ 明白不？？
再接着 CLK1 来一个上升沿，CS=”S4”要变为 CS=”NS”，COM 被启动，CASE 语句 S4 满足，则
ALE<=’0’;START<=’0’;OE<=’1’;LOCK<=’1’;NS<=’S0’;这个时候，OE 还是为’1’，没有上升沿，OE 信号无效，但是 LOCK 信号由之前的’0’变为’1’，LOCK 信号发生变化了，所以呢，主控组合进程还是间接驱动辅助进程 LATCH1 的。
③辅助进程 LATCH1：辅助进程一般就是为了消除毛刺，也就是把要输出的数据先锁存起来，稳定后再由锁存器输出。程序中设计的辅助进程 LATCH1 敏感信号是 LOCK，并且有一个关于 LOCK 的判断，如果 LOCK 信号产生了上升沿，那么锁存器里面的数据就被输出出去，也就是说，LOCK 信号是锁存器输出允许开关，上升沿有效。

程序代码：

```
LIBRARY  IEEE;
USE  IEEE.STD_LOGIC_1164.ALL;
USE  IEEE.STD_LOGIC_ARITH.ALL;
USE  IEEE.STD_LOGIC_UNSIGNED.ALL;

ENTITY  ADC0809  IS
    PORT(CLK,RST  :  IN  STD_LOGIC;
         D  :  IN  STD_LOGIC_VECTOR(7 DOWNTO 0); --数据输入信号
         EOC  :  IN  STD_LOGIC ;  --转换标志信号
         ALE  :  OUT  STD_LOGIC;  --转换地址信号
         START  :  OUT  STD_LOGIC;  --转换启动信号
         OE  :  OUT  STD_LOGIC;  --数据转换后输出允许信号
         ADDA  :  OUT  STD_LOGIC;  ---转换模式选择 ADDA
         LOCK_T  :  OUT  STD_LOGIC;  --锁存数允许输出开关信号
         Q  :  OUT  STD_LOGIC_VECTOR(7  DOWNTO 0) --锁存输出信号
        );
END ENTITY;

ARCHITECTURE  BHV  OF  ADC0809  IS
TYPE  STATES  IS  (S0，S1，S2，S3，S4); --定义了 5 个状态值
SIGNAL  CS,NS  :  STATES  :=S0;
--当前状态 CS、下一状态 NS 取值范围只能是 S0、S1、S2、S3、S4
SIGNAL  REGL  :  STD_LOGIC_VECTOR(7 DOWNTO 0);
SIGNAL  LOCK  :  STD_LOGIC;
BEGIN

ADDC <=  '0';  --在结构体里面单独一个赋值语句，要把它当做无条件启动的进程看待

LOCK_T <= LOCK;

--这两句话的意思是，ADDC 并行进程运行结束时被赋值'0'；
--  LOCK_T 端口输出信号并行进程结束时被赋值为"LOCK"

COM:  PROCESS(CS,EOC)
--主控组合进程，负责状态译码状态转换
--其敏感信号为当前状态信号 CS，转换状态标志信号 EOC
CASE  CS  IS  --根据当前状态确定下一状态
    WHEN  S0 => ALE<='0';START<='0';OE<='0';LOCK<='0';NS<='S1';
    --当前 S0 状态，下一状态 S1 代表启动装换
    WHEN  S1=>ALE<='1';START<='0';OE<='0';LOCK<='0';NS<='S2';
    --当前 S1 状态，已经开始转换了，是否转换完，需要进入 S3 状态判断
    WHEN  S2=>ALE<='0';START<='0';OE<='0';LOCK<='0';
        IF  EOC='1'  THEN  --转换完毕，进入下一状态 S3
```

```
                NS<=S3;
            ELSE
                NS<=S2;
            END IF;
        WHEN    S3=>ALE<='0';START<='0';OE<='1';LOCK<='0';NS<='S4';
        WHEN    S4=>ALE<='0';START<='0';OE<='1';LOCK<='1';NS<='S0';
        WHEN    OTHES=>ALE<='0';START<='0';OE<='0';LOCK<='0';NS<='S0'

END CASE;
END PROCESS;

REG:    PROCESS(CLK,RST)
        IF    RST='1'    THEN
            CS<=S0;
        ELSIF    RISING_EDGE(CLK)    THEN
            CS<=NS;
        END IF;
END PROCESS REG;

LATCH1：    PROCESS(LOCK)
        IF    RISING_EDGE(LOCK)    THEN
            REGL <= D;
        END IF;
        Q <= REGL;
END PROCESS;

END BHV;
```