ADC0809,8端口8位AD芯片，有问题可见数据手册或者百度。





芯片工作频率，并非程序中的频率。



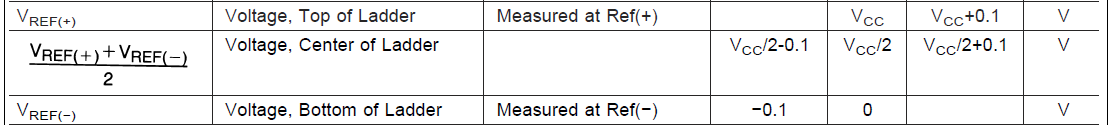


单次转换时间。ADC0809的处理数据的能力最快大约10KHz。多通道轮流使用，频率降低。

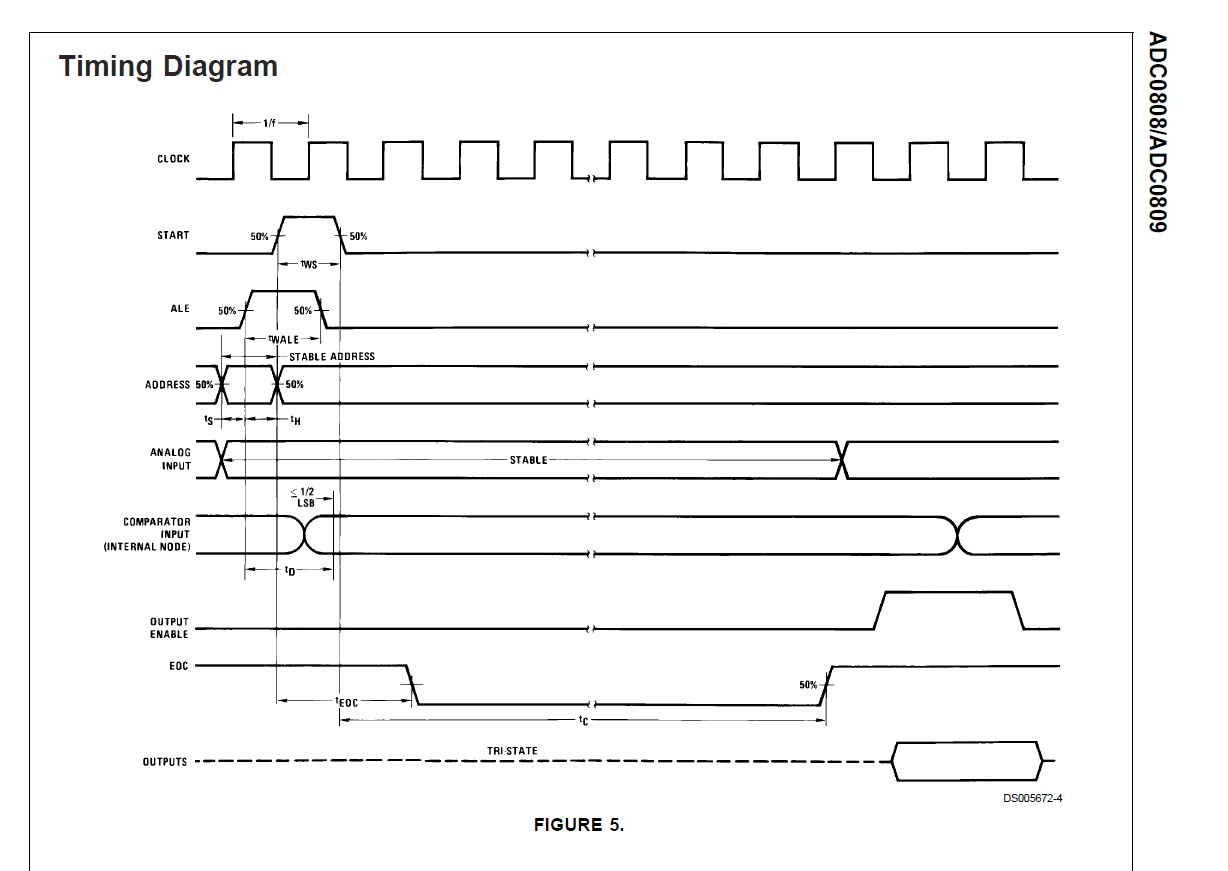
转换时间和精度够用就行（ADC0809便宜），不够就选择更高级的。

Vref（+）Vref（-）参考电位，输出电位的范围。





注意（Vref（+）+ Vref（-））/2的范围。

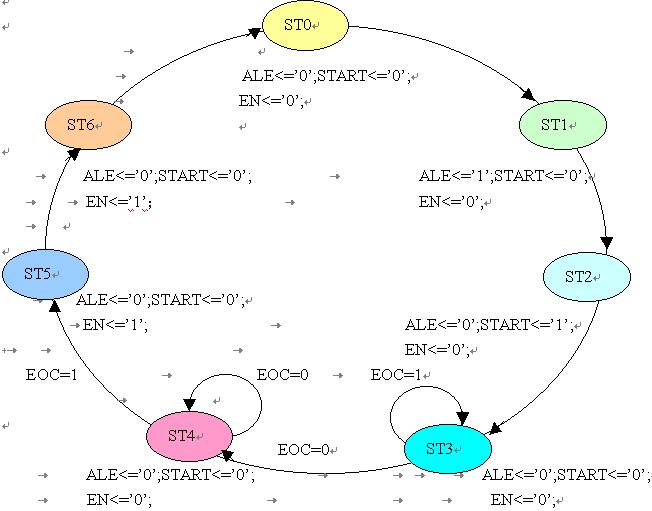


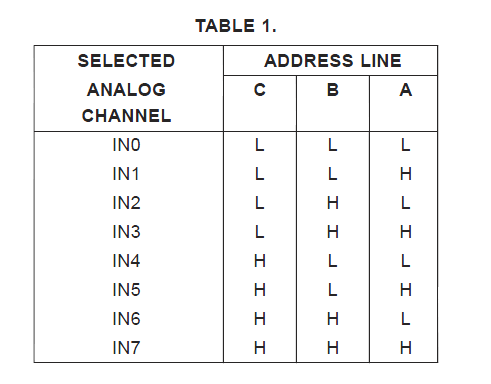
时序图。

根据时序图写就行了，下面仅供参考（延迟为后面程序服务）。

输出端具有三态输出锁存缓冲器，受输出允许信号的控制，当该信号为高电平时，打开输出缓冲器三态门，转换结果输出到数据总线上；当该信号为低电平时，输出数据线呈高阻态。

下图为网上的状态图，仅供参考





端口选择的逻辑表

**Verilog代码，仅供参考。**

module ADC0809(din,add,ale,start,eoc,oe,doutx,douty,clkadc,rst);

input [7:0] din; 8位输入

input clkadc; 时钟

input eoc; 转换结束标志

input rst; 复位

output [2:0] add; 端口选择

output start,ale,oe; 转换开始 地址输入允许 数据输出允许

output [7:0] doutx; X数据输出

output [7:0] douty; Y数据输出

reg [7:0] doutx;

reg [7:0] douty;

reg [2:0] add;

reg start,ale,oe;

reg [2:0] state;

reg flag; 标志位

reg END; 结束标志

reg [2:0] count=0; 延迟计数

parameter state0=3'b000,

state1=3'b001,

state2=3'b010,

state3=3'b011,

state4=3'b100,

state5=3'b101;

always @(posedge clkadc or negedge rst)

begin

if(!rst) state<=state0;

else

begin

case(state)

state0: 初始化

begin

END<=0;

ale<=0;

oe<=0;

start<=0;

if(flag==1'b0) 端口选择

begin

add<=3'b000;

flag<=1'b1;

end

else if(flag==1'b1)

begin

add<=3'b001;

flag<=1'b0;

end

else

begin

add<=3'b000;

flag<=1'b1;

end

state<=state1;

end

state1:

begin

ale<=1; 地址输入允许

oe<=0;

start<=0;

state<=state2;

end

state2:

begin

ale<=0;

start<=1; 转换允许

oe<=0;

state<=state3;

end

state3:

begin

ale<=0;

start<=0;

if(eoc==1) 等待转换结束信号

begin

state<=state4;

oe<=1; 输出允许

end

else

state<=state3;

end

state4: 输出允许，等待数据稳定

begin

ale<=0;

start<=0;

oe<=1;

state<=state5;

end

state5:

begin

ale<=0;

start<=0;

oe<=1;

END<=1;

if(count==3'd4) 延迟（后面程序需要）

begin

state<=state0; 转换完成

count<=0;

end

else count<=count+3'd1;

end

endcase

end

end

always @(posedge END) 接受数据，锁存

begin

if(flag==1'b0)

douty<=din;

else

doutx<=din;

end

endmodule