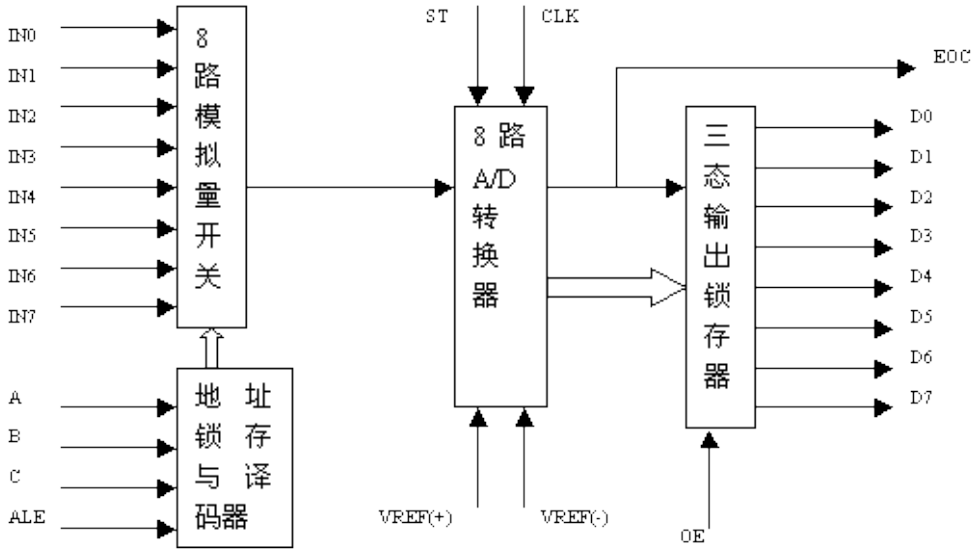
ADC0809 中文资料

ADC0809 是带有 8 位 A/D 转换器、8 路多路开关以及微处理机兼容的控制逻辑的 CMOS 组件。它是逐次逼近式 A/D 转换器，可以和单片机直接接口。

（1）**ADC0809 的内部逻辑结构**

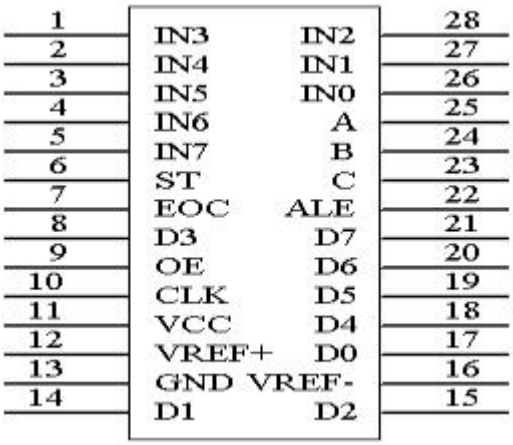
由下图可知，ADC0809 由一个 8 路模拟开关、一个地址锁存与译码器、一个 A/D 转换器和一个三态输出锁存器组成。多路开关可选通 8 个模拟通道，允许 8 路模拟量分时输入，共用 A/D 转换器进行转换。三态输出锁器用于锁存 A/D 转换完的数字量，当 OE 端为高电平时，才可以从三态输出锁存器取走转换完的数据。

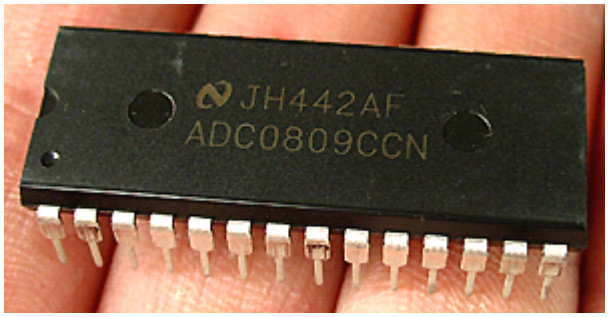


（2）． **ADC0809 引脚结构**

ADC0809 各脚功能如下：
D7-D0：8 位数字量输出引脚。
IN0-IN7：8 位模拟量输入引脚。
VCC：+5V 工作电压。
GND：地。
REF（+）：参考电压正端。
REF（-）：参考电压负端。
START：A/D 转换启动信号输入端。
ALE：地址锁存允许信号输入端。
（以上两种信号用于启动 A/D 转换）
EOC：转换结束信号输出引脚，开始转换时为低电平，当转换结束时为高电平。
OE：输出允许控制端，用以打开三态数据输出锁存器。
CLK：时钟信号输入端（一般为 500KHz）。

A、B、C：地址输入线。





ADC0809 对输入模拟量要求：信号单极性，电压范围是 0－5V，若信号太小，必须进行放大；输入的模拟量在转换过程中应该保持不变，如若模拟量变化太快，则需在输入前增加采样保持电路。

地址输入和控制线：4 条

ALE 为地址锁存允许输入线，高电平有效。当 ALE 线为高电平时，地址锁存与译码器将 A，B，C 三条地址线的地址信号进行锁存，经译码后被选中的通道的模拟量进转换器进行转换。

A，B 和 C 为地址输入线，用于选通 IN0－IN7 上的一路模拟量输入。通道选择表如下表所示。

C	B	A	选择的通道
0	0	0	IN0
0	0	1	IN1
0	1	0	IN2
0	1	1	IN3
1	0	0	IN4
1	0	1	IN5
1	1	0	IN6
1	1	1	IN7

数字量输出及控制线：11 条

ST 为转换启动信号。当 ST 上跳沿时，所有内部寄存器清零；下跳沿时，开始进行 A/D 转换；

在转换期间，ST 应保持低电平。EOC 为转换结束信号。当 EOC 为高电平时，表明转换结束；否则，表明正在进行 A/D 转换。OE 为输出允许信号，用于控制三条输出锁存器向单片机输出转换得到的数据。OE＝1，输出转换得到的数据；OE＝0，输出数据线呈高阻状态。D7－D0 为数字量输出线。

CLK 为时钟输入信号线。因 ADC0809 的内部没有时钟电路，所需时钟信号必须由外界提供，通常使用频率为 500KHZ，

VREF（＋），VREF（－）为参考电压输入。

2. ADC0809 应用说明

（1）. ADC0809 内部带有输出锁存器，可以与 AT89S51 单片机直接相连。

（2）. 初始化时，使 ST 和 OE 信号全为低电平。

（3）. 送要转换的哪一通道的地址到 A，B，C 端口上。

（4）. 在 ST 端给出一个至少有 100ns 宽的正脉冲信号。

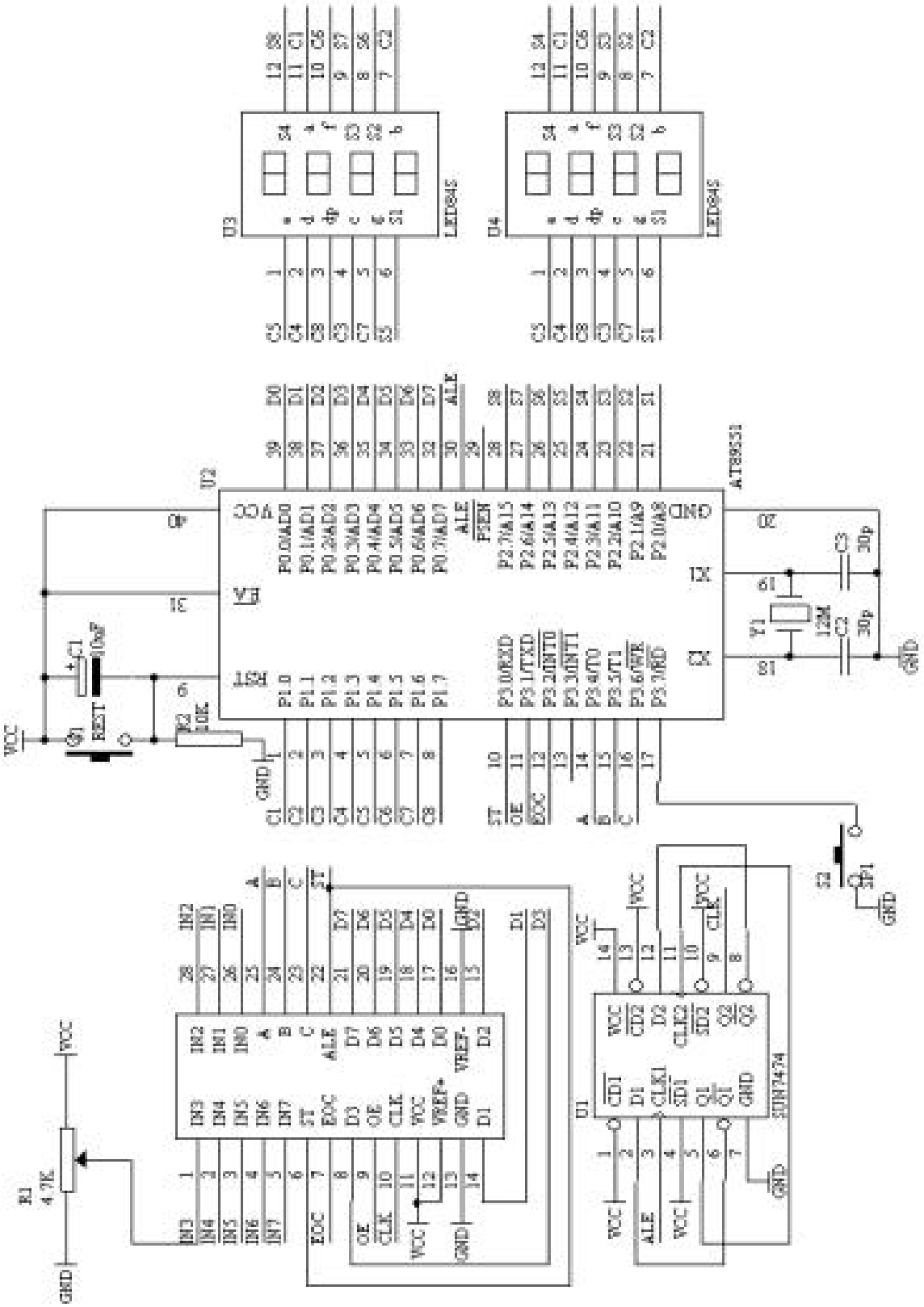
（5）. 是否转换完毕，我们根据 EOC 信号来判断。

（6）. 当 EOC 变为高电平时，这时给 OE 为高电平，转换的数据就输出给单片机了。

3. 实验任务

如下图所示，从 ADC0809 的通道 IN3 输入 0－5V 之间的模拟量，通过 ADC0809 转换成数字量在数码管上以十进制形成显示出来。ADC0809 的 VREF 接＋5V 电压。

4. **ADC0809 应用电路原理图**



6. 程序设计内容

（1）． 进行 A/D 转换时，采用查询 EOC 的标志信号来检测 A/D 转换是否完毕，若完毕则把数据通过 P0 端口读入，经过数据处理之后在数码管上显示。

（2）． 进行 A/D 转换之前，要启动转换的方法：

ABC＝110 选择第三通道

ST＝0，ST＝1，ST＝0 产生启动转换的正脉冲信号 .

C 语言源程序

```
#include
unsigned char code dispbitcode[]={0xfe,0xfd,0xfb,0xf7,
  0xef,0xdf,0xbf,0x7f};
unsigned char code dispcode[]={0x3f,0x06,0x5b,0x4f,0x66,
0x6d,0x7d,0x07,0x7f,0x6f,0x00};
unsigned char dispbuf[8]={10,10,10,10,10,0,0,0};
unsigned char dispcount;

sbit ST="P3"^0;
sbit OE="P3"^1;
sbit EOC="P3"^2;
unsigned char channel="0xbc";//IN3
unsigned char getdata;

void main(void)
{
TMOD=0x01;
TH0=(65536-4000)/256;
TL0=(65536-4000)%256;
TR0=1;
ET0=1;
EA=1;

P3=channel;

while(1)
{
ST=0;
ST=1;
ST=0;
while(EOC==0);
OE=1;
getdata=P0;
OE=0;
```

```
dispbuf[2]=getdata/100;
getdata=getdata%10;
dispbuf[1]=getdata/10;
dispbuf[0]=getdata%10;
}
}

void t0(void) interrupt 1 using 0
{
TH0=(65536-4000)/256;
TL0=(65536-4000)%256;
P1=dispcode[dispbuf[dispcount]];
P2=dispbitcode[dispcount];
dispcount++;
if(dispcount==8)
{
dispcount=0;
}
```