# 延时程序

## 含参数

void delay1(unsigned int t)

{

for(;t>0;t--);

}

### 12M t=1ms

void delay(unsigned long int t)

{

unsigned char i,j;

for(i=25;i>0;i--)

for(j=50;j>0;j--)

for(;t>0;t--);

}

### 33M t=1ms

void delay(unsigned long int t)

{

unsigned char i,j;

for(i=6;i>0;i--)

for(j=50;j>0;j--)

for(;t>0;t--);

}

## 不含参数 若需要特定秒，只需将延时函数循环

12M 延时1ms

void delay(void)

{

unsigned char i,j,k;

for(i=4;i>0;i--)

for(j=50;j>0;j--)

for(k=1;k>0;k--);

}

延时10ms

void delay(void)

{

unsigned char i,j,k;

for(i=40;i>0;i--)

for(j=50;j>0;j--)

for(k=1;k>0;k--);

}

延时100ms

void delay(void)

{

unsigned char i,j,k;

for(i=200;i>0;i--)

for(j=100;j>0;j--)

for(k=1;k>0;k--);

}

延时1s

void delay(void)

{

unsigned char i,j,k;

for(i=200;i>0;i--)

for(j=100;j>0;j--)

for(k=25;k>0;k--);

}

延时50us

void delay(void)

{

unsigned char i;

for(i=22;i>0;i--);

}

延时us 一条指令2us

# 矩阵式键盘

## 反转线法

unsigned char keyscan(void)

{ //P3为键盘接入端口

unsigned char cord\_h,cord\_l; //行列值中间变量

P3=0x0f; //行线输出全为0

cord\_h=P3&0x0f; //读入列线值

if(cord\_h!=0x0f) //先检测有无按键按下

{

delay(10); //去抖

if((P3&0x0f)!=0x0f)

{

cord\_h=P3&0x0f; //读入列线值

P3=cord\_h|0xf0; //输出当前列线值

cord\_l=P3&0xf0; //读入行线值

// while((P3&0xf0)!=0xf0); //等待松开并输出

//长按识别，若有则只识别一次，若无则一直识别

return(cord\_h+cord\_l); //键盘最后组合码值

}

}return(0xff); //返回该值

}

## 扫描法

unsigned char keyscan(void) //键盘扫描函数，使用行列逐级扫描法

{

unsigned char Val;

P3=0xf0; //高四位置高，低四位拉低

if(P3!=0xf0) //表示有按键按下

{

delay(10); //去抖

if(P3!=0xf0)

{ //表示有按键按下

P3=0xfe; //检测第一行

if(P3!=0xfe)

{

Val=P3&0xf0;

Val+=0x0e;

// while(P3!=0xfe); //长按识别，若有则只识别一次，若无一直识别

// delay(10); //去抖

// while(P3!=0xfe); //以下识别也相同

return Val;

}

P3=0xfd; //检测第二行

if(P3!=0xfd)

{

Val=P&0xf0;

Val+=0x0d;

while(P3!=0xfd);

delay(10);

while(P3!=0xfd);

return Val;

}

P3=0xfb; //检测第三行

if(P3!=0xfb)

{

Val=P3&0xf0;

Val+=0x0b;

while(P3!=0xfb);

delay(10);

while(P3!=0xfb);

return Val;

}

P3=0xf7; //检测第四行

if(P3!=0xf7)

{

Val=P3&0xf0;

Val+=0x07;

while(P3!=0xf7);

delay(10);

while(P3!=0xf7);

return Val;

}

}

}

return 0xff;

}

## 识别函数

unsigned char keypro(void)

{

switch(keyscan())

{

case 0xee:return 1;break;//0 按下相应的键显示相对应的码值

case 0xde:return 2;break;//1

case 0xbe:return 3;break;//2

case 0x7e:return 4;break;//3

case 0xed:return 5;break;//4

case 0xdd:return 6;break;//5

case 0xbd:return 7;break;//6

case 0x7d:return 8;break;//7

case 0xeb:return 9;break;//8

case 0xdb:return 10;break;//9

case 0xbb:return 11;break;//a

case 0x7b:return 12;break;//b

case 0xe7:return 13;break;//c

case 0xd7:return 14;break;//d

case 0xb7:return 15;break;//e

case 0x77:return 16;break;//f

default:return 0;break;

}

}

# 数码管动态显示

unsigned char code num[]={0x3f,0x06,0x5b,0x4f,0x66,0x6d,0x7d,0x07,0x7f,0x6f};

unsigned char code cho[]={0xfe,0xfd,0xfb,0xf7,0xef,0xdf,0xbf,0x7f};

sbit x26=P2^6; //位选择ld

sbit x27=P2^7; //字型控制ld

display(unsigned char pp[8]) //需要循环显示

{

unsigned char i,j=0,k;

for(i=8;i>0;i--)

{

x26=0;

P0=cho[j]; //P0控制口

x26=1;

x26=0;

k=pp[j];

x27=0;

P0=num[k];

x27=1;

x27=0;

j++;

delay(t); //延时10ms

}

}

# 蜂鸣器，可变频率方波

sbit x10=P1^0;

#define time 10

void init\_sys()

{

TMOD=0x02;

TL0=0xd2;

TH0=0xd2;

TR0=1;

ET0=1;

EA=1;

}

void time0() interrupt 1

{

flag++;

if(flag==time)

{

x10=~x10;

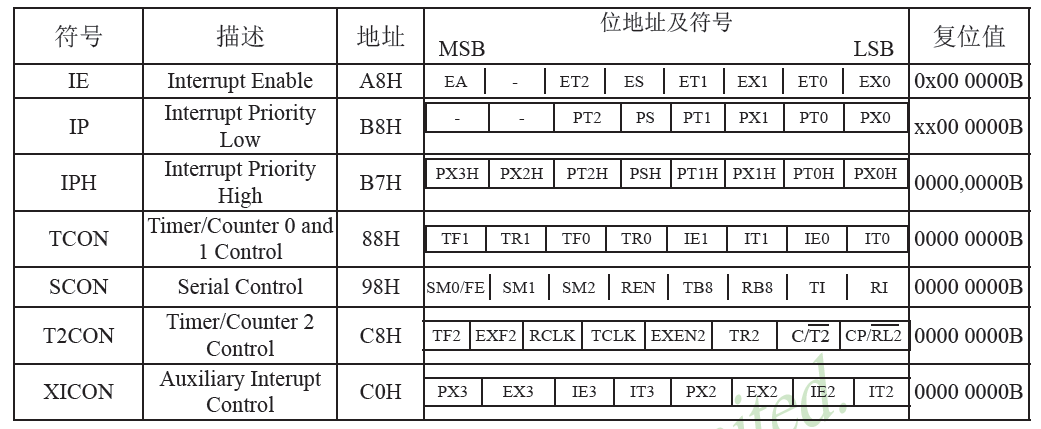
flag=0;

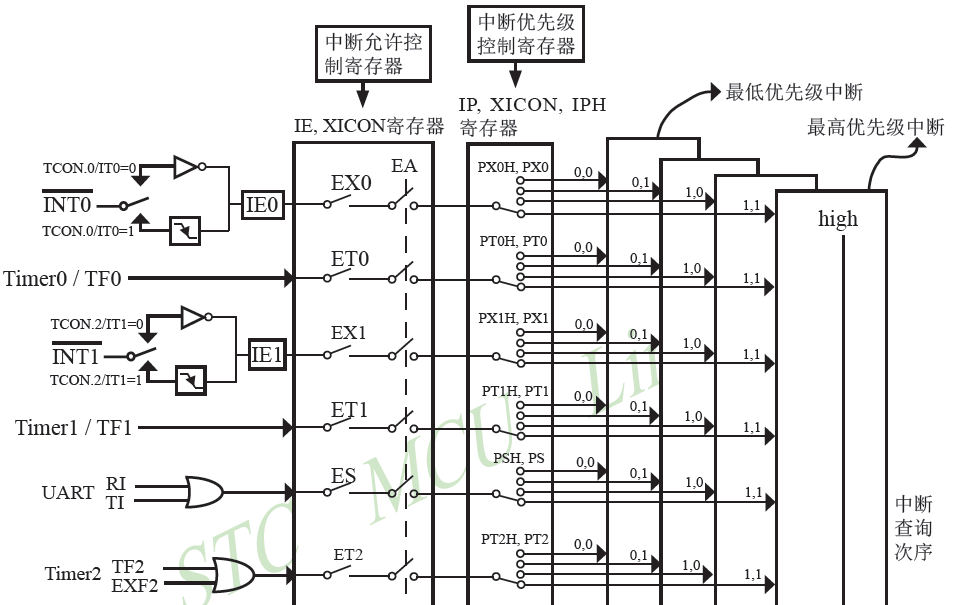
}

}

# 中断

## 寄存器



IE interrupt enable EA - ET2 ES ET1 EX1 ET0 EX0

中断允许 EA总中断 ET为c/t中断允许 ES串行口中断允许 EX外部中断允许

IP interrupt priority low - - PT2 PS PT1 PX1 PT0 PX0

中断优先级

TCON TF1 TF0 TR0 IE1 IT1 IE0 IT0

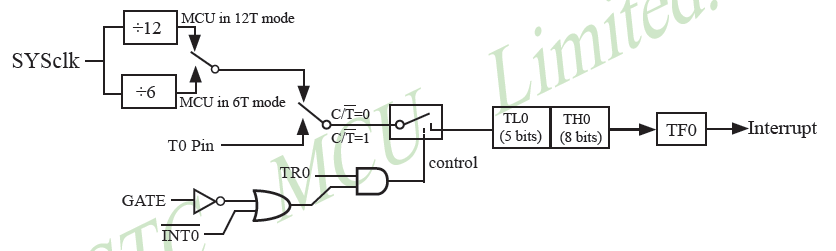
c/t控制 TF溢出标志溢出为一 TR运行控制开启控制置一 IE外部中断请求中断置一 IT外部中断触发方式低电平为零下降沿为一

SCON SM0 SM1 SM2 REN TB8 RB8 TI RI

串行口控制

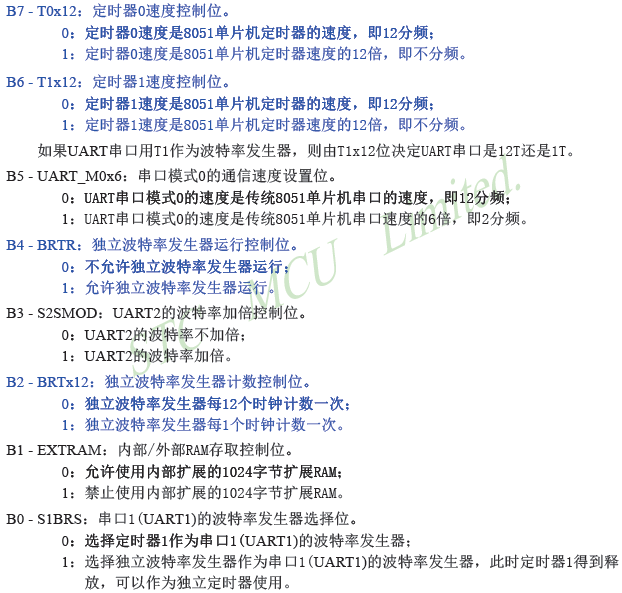
TMOD GATE1 C/T M1 M0 GATE0 C/T M1 M0

c/t工作方式 gate为零或INT0为一时开c/t c=1/t=0 M0M1工作模式

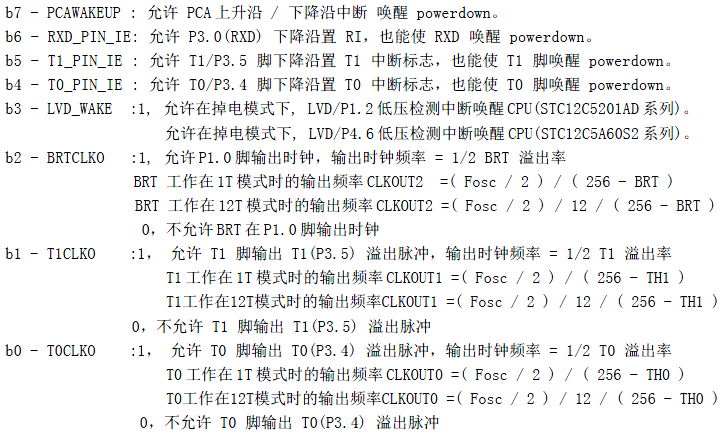


### STC12C5A60s2特有

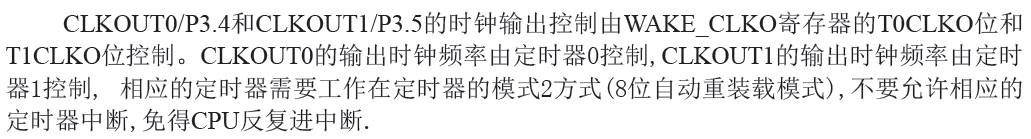
AUXR 8EH T0x12 T1x12 UART\_m0x6 BRTR S2SMOD BRTx12 EXTRAM S1BRS

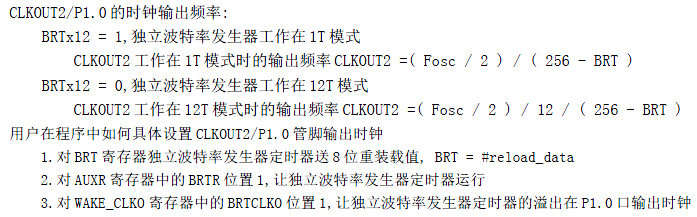


WAKE\_CLK0 8FH PCAWAKEUP RXD\_PIN\_IE T1\_PIN\_IE T0\_PIN\_IE LVD\_WAKE BRTCLK0 T1CLK0 T0CLK0



BRT 9CH





#### 程序输出三时钟

//sfr AUXR=0x8E;

sfr WAKE\_CLK0=0x8F;

//sfr BRT=0x9C;

void timerinit(void)

{

TMOD=0X22; //初始化定时器0，工作方式1

TL0=0xfe; //2.75M

TH0=0xfe;

TR0=1;

TL1=0x1a; //2K

TH1=0x1a;

TR1=1;

BRT=0x76; //40K

AUXR=0x94;

WAKE\_CLK0=0x07;

}

### 外部中断初始化

Void intrinit(void)

{

EX1=1;

IT1=0/1; //触发方式

EA=1;

}

### 定时器初始化

Void timerinit（void）

{

TMOD=0X01; //初始化定时器0，工作方式1

TL0=; //定时

THO=;

ET0=1;

EA=1;

TR0=1;

}

### T2输出时钟(52)

sfr T2MOD=0XC9;

void timerinit(void)

{

T2MOD=0X02;

RCAP2L=0xf7; //f=3M/T2 12时钟/机器周期

RCAP2H=0xff;

TR2=1;

C\_T2=0;

}

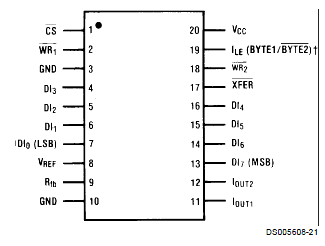
# 数模转换

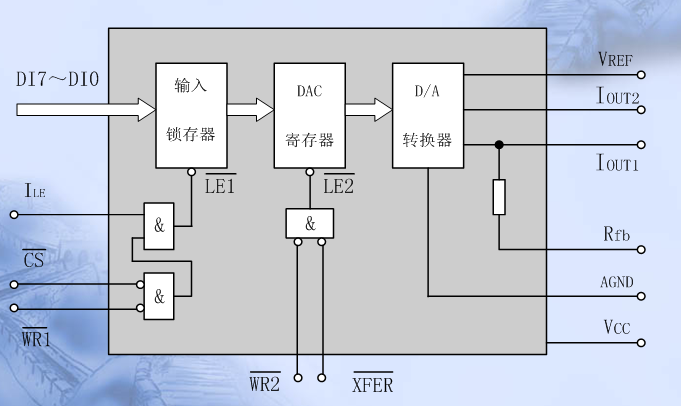
## DAC0832

VREF基准电压

IOUT12 差分输出

RFB反馈电阻





unsigned char daread（void）

{

CS=0; //片选

ILE=1; //输入锁存器控制

WR1=0; //WR1输入负脉冲，输入锁存器工作

XFER=0; //数据传输控制

WR2=0; //WR2输入负脉冲，DAC寄存器工作

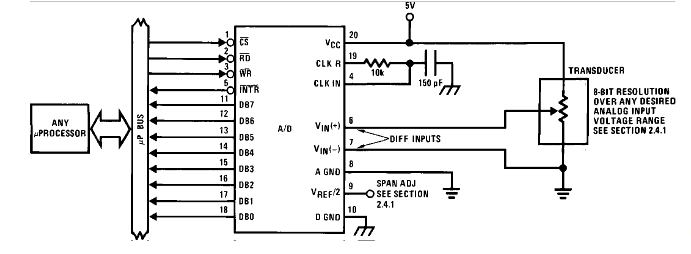
}

## ADC0804

CLK R/IN 输入震荡时钟 f=1/1.1rc常用c=150pf r=10k

VIN差分输入

VREF/2接VCC/2



Unsigned char adread（void）

{

CS=0;

WR=0; //或写作WR=1;WR=0;WR=1;//WR输入低电平，输入锁存器工作

RD=1;

RD=0; //查询工作方式

RD=1;

INTR; //中断工作方式

}

# 串行通信

异步通信 同步通信

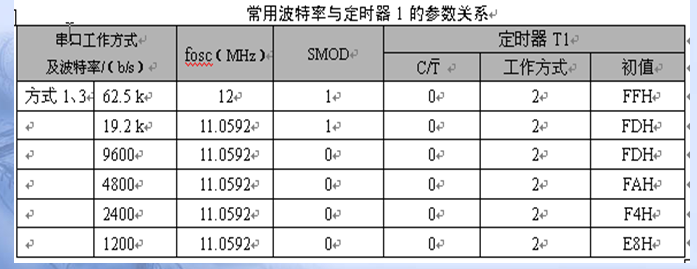
单工 半双工 全双工

奇偶校验 代码和校验 循环冗余校验

波特率

工作方式1 波特率=（2SMOD/32）**·**（T1溢出率）

Timer1 溢出率 = fosc /{12×[256 －（TH1）]}



传输距离

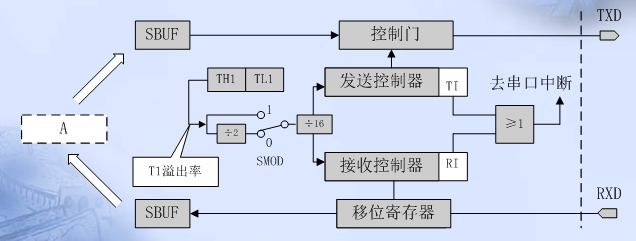
RS-232C 阳头 阴头 9针 25针

MAX232CPE 为RS-232电平与TTL电平转换驱动电路



RS-422A

RS-485



SBUF 物理地址不同 机器寻址相同

## 寄存器



SCON SM0 SM1 SM2 REN TB8 RB8 TI TRI

串行口控制 SM0 SM1工作方式选择REN 允许串行接收允许接收为一 TI/RI发送/接收中断标志位触发中断为一由软件清零 SM2多机通信控制 TB8/RB8在方式2或3中发送/接收数据第九位作用

PCON SMOD - - - - - - -

SMOD波特率倍增倍增为一

### 初始化

unsigned char re,se; //定义全局变量

void vartinit（void）

{

TMOD=0x20; //timer1工作方式2

TH1=0xfd; //SMOD=0 fosc=11.0592M 波特率=9600

TL1=0xfd;

EA=1;

ES=1; //开串行口中断

TR1=1; //开timer1中断

REN=1; //或SCON=0x50

SM0=0; //串行口工作方式1

SM1=1;

}

/\*

void vartinit(void) //用独立波特率发生器作为波特率发生器

{

SCON=0x50;

BRT=0xfd; //设定时钟

AUXR=0x11;

// PCON=0x80; //波特率倍增

// AUXR1=0x80;

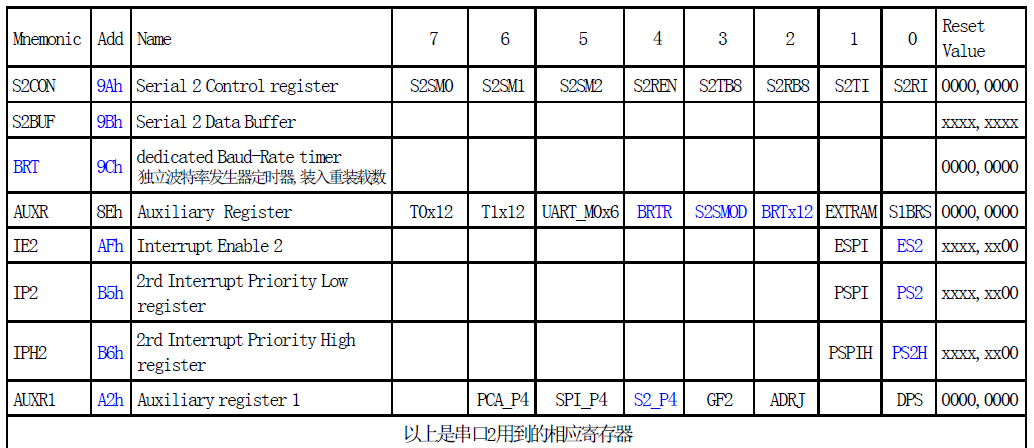
ES=1;

EA=1;

}

/\*

### 串行口2



sfr S2CON=0x9a;

sfr S2BUF=0x9b;

sfr BRT=0x9c;

sfr IE2=0xaf;

sfr IP2=0x5b;

sfr IPH2=0xb6;

sfr AUXR=0x8e;

sfr AUXR1=0xa2;

void vart2init(void)

{

S2CON=0x50;

BRT=0xfd;

AUXR=0x11;

IE2=0x01;

EA=1;

}

void receive2(void) interrupt 8

{

S2RI=S2CON&0x01;

if(S2RI)

{

re2=S2BUF;

S2CON=S2CON&0xfe;

}

S2TI=S2CON&0x02;

if(S2TI)

{

S2BUF=se2;

S2CON=S2CON&0xfd;

}

}

## 中断方式 //先接收两位，再发送四位

unsigned char read0,read1;

unsigned char send0,send1,send2,send3;

unsigned char flagre=0,flagwr=0;

void king(void) interrupt 4

{

if(RI) //判断接收中断还是发送中断

{

if(flagre==0)

{

read0=SBUF;

flagre=1;

}

else if(flagre==1)

{

read1=SBUF;

flagre=0;

if(flagwr==0)

{

SBUF=send0;

flagwr=1;

}

}

RI=0;

}

if(TI)

{

switch(flagwr)

{

// case 0:{SBUF=send0;flagwr=1;break;}

case 1:{SBUF=send1;flagwr=2;}break;

case 2:{SBUF=send2;flagwr=3;}break;

case 3:{SBUF=send3;flagwr=0;}break;

}

TI=0;

}

}

## 查询方式

### 发送两位数据

unsigned char flag=1; //flag标志位

unsigned char send0,send1;

//TI=0;

if(flag==1)

{

SBUF=send1;

flag=0;

while(!TI); //以空循环方式等待前一个数据发送完毕，返回中断

// if(TI==1){} //或以查询方式

TI=0; //发送中断标志位软件清零

}

else if(flag==0)

{

SBUF=send0;

flag=1;

while(!TI);

TI=0;

}

### 接收两位数据

unsigned char flag=1;

unsigned char read0,read1;

if(flag==1)

{

while(RI==0); //以空循环方式等待数据发送过来，返回中断标志位

read0=SBUF;

flag=0;

RI=0;

}

else if(flag==0)

{

while(RI==0);

read1=SBUF;

flag=1;

RI=0;

}

# LCD显示

## YM1602C

unsigned char dis1[]={0x4d,0x41,0x52,0x53};

unsigned char dis2[];

unsigned char code lcdnumber[]={0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39};

sbit x24=P2^4; //RS

sbit x25=P2^5; //EN

sbit x26=P2^6; //RW

void delay(char t)

{

unsigned char i,j;

for(i=25;i>0;i--)

for(j=50;j>0;j--)

for(;t>0;t--);

}

/\*

unsigned char busy(void) //询问是否忙，不忙为零

{

unsigned char bf;

x24=0;

x26=1;

x25=1;

delay(5);

bf=P0&0x80;

x25=0;

return(bf);

}

\*/

void wrcmd(unsigned char cmd)

{

// while(busy());

x24=0;

// x26=0;

x25=0;

P0=cmd;

x25=1;

delay(5); //EN最小脉冲宽度

x25=0;

}

void wrdoc(unsigned char doc)

{

// while(busy());

x24=1;

// x26=0;

x25=0;

P0=doc;

x25=1;

delay(5);

x25=0;

}

void position(unsigned char pos)

{

wrcmd(pos|0x80); //选择地址，并赋值

}

void lcdinit(void)

{

wrcmd(0x38); //功能设置，0x28八位总线，0x30单行显示，0x3c为5\*10点阵

wrcmd(0x0c); //开显示，0x0e有光标，0x0f光标闪烁

// wrcmd(0x06); //光标移动方式，0x04光标左移，0x07文字移动随光标一起

wrcmd(0x01); //清屏

// wrcmd(0x02); //光标复位至地址00H

// wrcmd(0x08); //关闭显示

}

void lcddisplay(void) //显示函数，可直接调用

{

unsigned char i;

lcdinit();

delay(10);

position(0x00); //写第一行显示值，0x0n从第n+1格开始写

for(i=0;/\*dis1[i]!='\0'\*/i<4;i++)

{

wrdoc(dis1[i]);

}

position(0x40); //写第二行显示值

for(i=0;/\*dis1[i]!='\0'\*/i<16;i++) //16为单行显示最多数目

{

wrdoc(dis2[i]);

}

}

# 温度传感器

## 18b20 程序

sbit DQ=P0^0; //定义传感器的数据传输口

unsigned char temperature[4]; //定义全局变量

unsigned char tempflag;

unsigned long tempintr temp;

unsigned char init(void)

{

unsigned char init=1;

DQ=1;

;;;; //延时几个us

DQ=0;

delay(1); //延时500us到1ms

DQ=1;

delay1(); //延时100us内

init=DQ;

delay(20); //延时10ms

return(init); //0为初始化成功

}

unsigned char read(void) //读一个字节

{

unsigned char re,i;

for(i=0;i<8;i++)

{

DQ=0;

re>>=1;

DQ=1;

;;;; //延时几个us

if(DQ)

re=re|0x80;

delay1(); //延时100us内

}

return(re);

}

void write(unsigned char wr)

{

unsigned char i;

for(i=0;i<8;i++)

{

DQ=0;

DQ=wr&0x01;

delay1(); //延时100us内

DQ=1;

wr>>=1;

;;;; //延时几个us

}

}

unsigned char readtemp(void) //只需调用此函数即获得温度

{

unsigned char templ,temph;

float tempdata;

tempinit(); //未检查初始化成功的返回值

write(0xcc);

write(0x44);

delay(10); //延时10ms

tempinit();

write(0xcc);

write(0xbe);

templ=read();

temph=read();

temp=temph;

temp<<=8;

temp |=templ;

if(temp&0xf000!=0)

{

tempflag=1; //1表示温度为正

}

tempdata=temp\*0.0625;//处理温度数据

temp=tempdata\*10;

/\* temperature[0]=flag; //也可直接取每位温度

temperature[1]=temp/100;

temperature[2]=temp%100/10;

temperature[3]=temp%10;

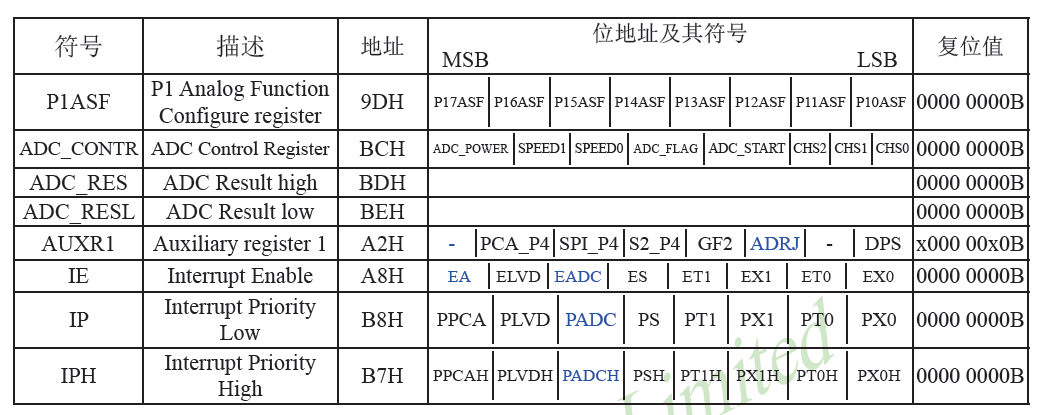
\*/

return(temp); //返回值为三位数温度，也可设为全局变量

}

# STC12C5A60S2的AD转换器

## 寄存器



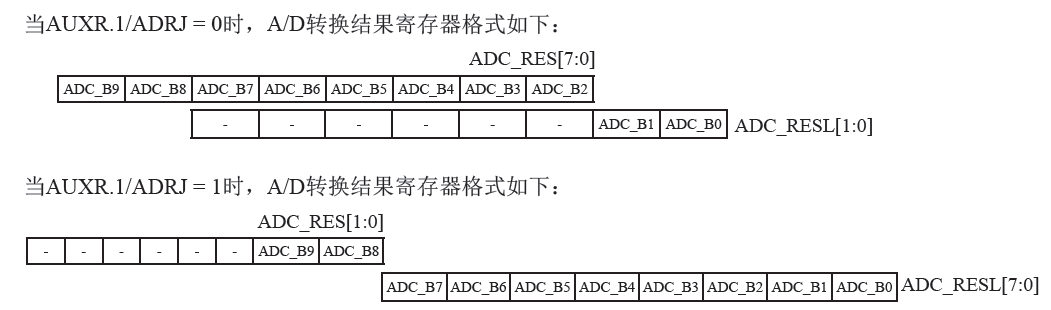
P1ASF P1口模拟功能使能控制

ADC\_RES/ADC\_RESL 转换结果寄存器

ADC\_CONTR ADC\_POWER为1打开AD转换器电源 SPEED1/SPEED0转换速度控制 ADC\_FLAG转换结束中断软件清零 ADC\_START为1启动转换结束自动为0 CHS2/CHS1/CHS0模拟通道选择

AUXR.ADRJ转换结果储存方式

IE.EA总中断 IE.EADC为1AD转换中断开启



## 程序//循环四路AD扫描转换

sfr ADC\_CONTR=0XBC;

sfr ADC\_RES=0XBD;

sfr ADC\_RESL=0XBE;

sfr P1ASF=0X9D;

sfr AUXR1=0XA2;

void initadc()

{

P1ASF=0xf0;

ADC\_RES=0;

ADC\_RESL=0；

ADC\_CONTR=0x88;

AUXR1=0x00； //若只用八位，则用此模式

delay(); //10us

IE=0xa0;

}

### 中断方式

void adc interrupt 5

{

ADC\_CONTR&=!0x10 //中断标志位清零

//得到数据，可处理数据发送数据八位或十位，发送ch

if(++ch>4)ch=0; //ch为通道选择

ADC\_CONTR=0x88|ch；

}

### 查询方式

unsigned char adcch(unsigned char ch)

{

ADC\_CONTR=0x88|ch;

delay(); //10us

while(!ADC\_CONTR&0x10); //转换完成检测

ADC\_CONTR=~0x10;

}

void main()

{

while(1)

{

adcch(0);

adcch(1);

adcch(2);

adcch(3);

}

}

# STC12C5A60S2的PCA/PWM模块

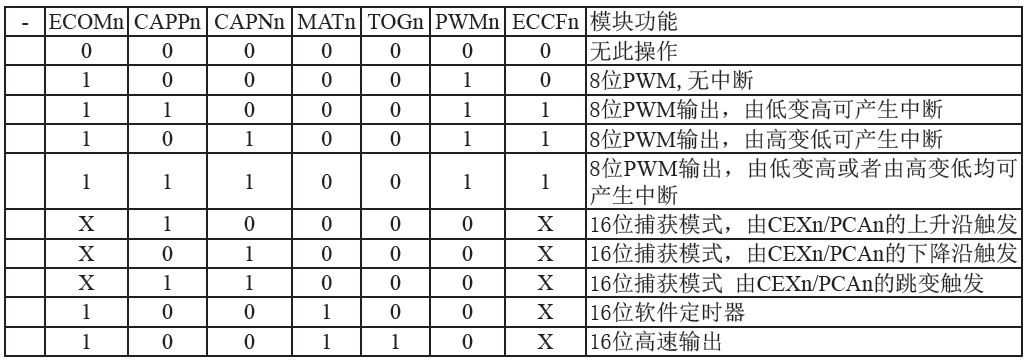
## 寄存器



CCON控制寄存器 CF计数器溢出标志位软件清零 CR计数器阵列运行 CCF1/CCF0模块1/0中断标志软件清零

CMOD工作模式寄存器 CIDL空闲模式计数器继续工作 CPS2/CPS1/CPS0计数脉冲源选择 ECF 溢出中断CF使能

CCAPM0模块0比较/捕获寄存器 ECOM0允许比较器功能位 CAPP0正捕获控制位允许上升沿捕获 CAPN0负捕获控制位允许下降沿捕获 TOG0翻转控制位 PWM0脉宽调节控制位 ECCF0使能CCF0中断



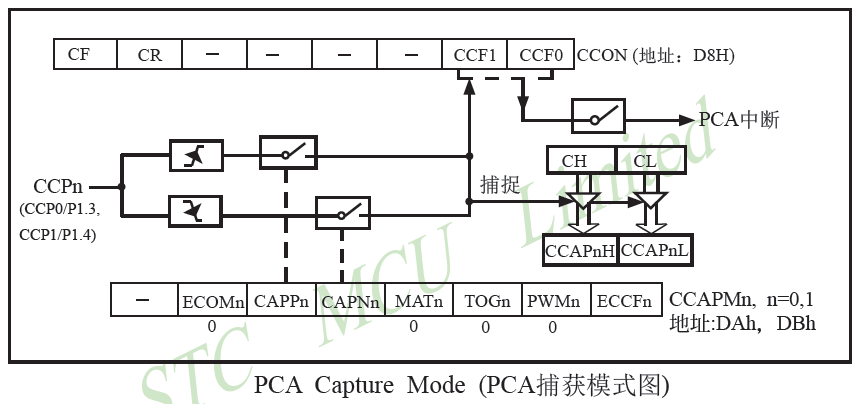
CL/CH 16位计数器低8位/高8位

CCAPnL/CCAPnH 捕捉/比较寄存器

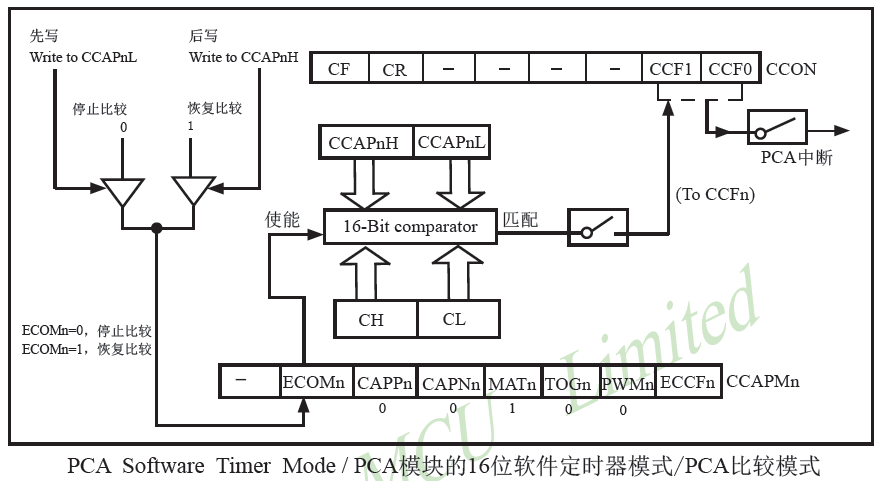
PCA\_PWMn EPCnH/EPCnL 在PWM模式下与CCAPnH/CCAPnL组成9位数

AUXR1 PCA功能从P1口切换到P4口 PCA\_P4 SPI\_P4 S2\_P4 GF2 ADRJ DSP

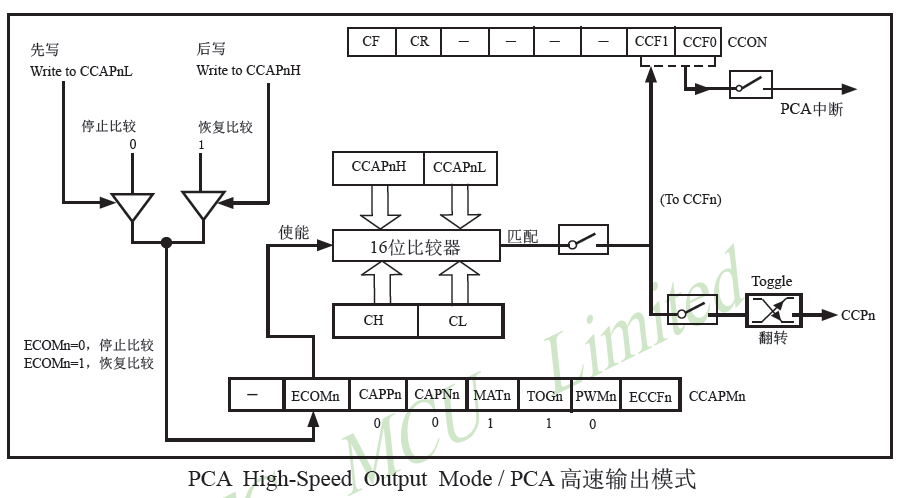
## 捕获模式



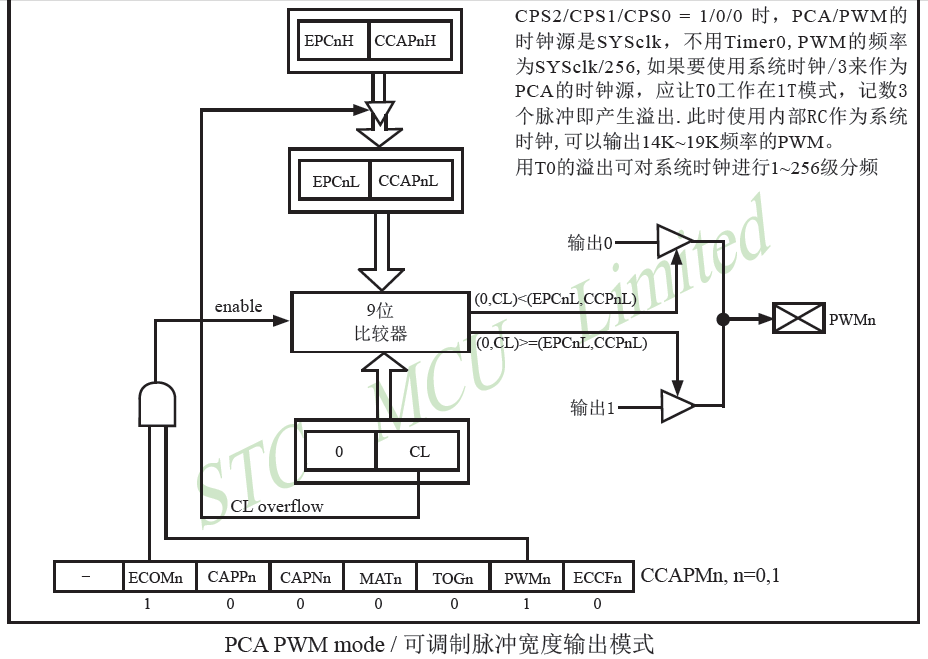
## 16位软件定时器模式



## 高速输出模式



## 脉宽调节模式//无中断模式



#### 程序

sfr CCON=0XD8;

sbit CCF0=CCON^0;

sbit CCF1=CCON^1;

sbit CR=CCON^6;

sbit CF=CCON^7;

sfr CMOD=OXD9;

sfr CL=0XE7;

sfr CH=0XF7;

sfr CCAPM0=0XDA;

sfr CCAP0L=0XEA;

sfr CCAP0H=0XFA;

sfr CCAPM1=0XDB;

sfr CCAP1L=0XEB;

sfr CCAP1H=0XFB;

sfr PCAPWM0=0XF2;

sfr PCAPWM1=0XF3;

void PWM(void)

{

CCON=0x00;

CL=0X00;

CH=0X00;

CMOD=0X02; //二分之一系统时钟关闭CF中断，0x03为开中断

CCAP0H=CCAP0L=0X80; //50%占空比,无需设置PCAPWM0

CCAPM0=0X42; //无中断八位PWM模式

CCAP1H=CCAP1L=0XFF;

PCAPWM1=0X03; //PWM固定输出低

/\*

CCAP1H=CAP1L=0X00;

PCAPWM1=0X00; //PWM固定输出高

\*/

CCAPM1=0X42;

CR=1; //计数器开始计数

}

# NOTICE

1 十进制数与十六进制数等价，只需在十六进制数前加ox，而在二进制数后加b

2给数组赋变量的值时，需要一个数一个数的赋值，分别给其赋变量

3获取某一位用按位或P0^1=P0&0x80