

课程设计报告

课程名称 综合电子设计

题 目 简易数字示波器

指导教师

起止日期

系 别 自动化

专 业 自动控制

学生姓名

班级/学号

成 绩

摘要

本系统由CPLD 单片机控制模块，键盘，LED 幅度控制模块，低通滤波模块组成，采用当前主流DDS技术完成，能产生从1HZ-260KHZ正弦波，方波，三角波以及这三种同频率波的线性组合，失真度限制在6%之内。

一、功能介绍

1. 具有产生正弦波、方波、三角波三种周期性波形的性能。
2. 用键盘输入编辑生成上述三种波形（同周期）的线性组合波形。
3. 输出波形频率范围为1Hz~200kHz（非正弦波频率按10次谐波计算；重复频率可调，频率步进间隔1Hz。）
4. 输出波形幅度范围 0~5V（峰-峰值），可按步进为0.1V（峰-峰值）。
5. 具有显示输出波形种类、重复频率（周期）和幅度的功能。
6. 增加稳幅输出功能，当负载变化时，输出电压幅度变化不大于±3%（负载变化范围：100Ω~∞）。

二、方案论证与比较

常见信号源的制作方法有：

方案一：采用锁相式频率合成。将一个高稳定性和高精确度的标准频率经过加减乘除的运算产生同样稳定度的大量离散频率技术，它在一定程度上既要频率稳定精确，又要频率在很大范围内可变的矛盾。但频率受VCO可变频率范围的影响，高低频率比不可能做的很高，而且只能产生方波和正弦波。

方案二：采用模拟奋力元件或单片压控函数发生器MAX0832可产生正弦波，方波，三角波，通过调整外部元件可改变输出频率，但采用模拟器件由于元件分散性太大，即使使用单片函数发生器，参数也与外部元件有关，外接的电阻电容对参数影响很大，不能实现波形运算输出等智能化的功能。

方案三：采用DDFS即直接数字频率合成技术，以Nyquist 时域采样原理为基础，在时域中进行频率合成，它可以快速转换频率，频率，相位，幅度都可以实现程控，便于单片机控制，所以，本系统采用此方案。

三、系统设计

系统总体设计方框图：

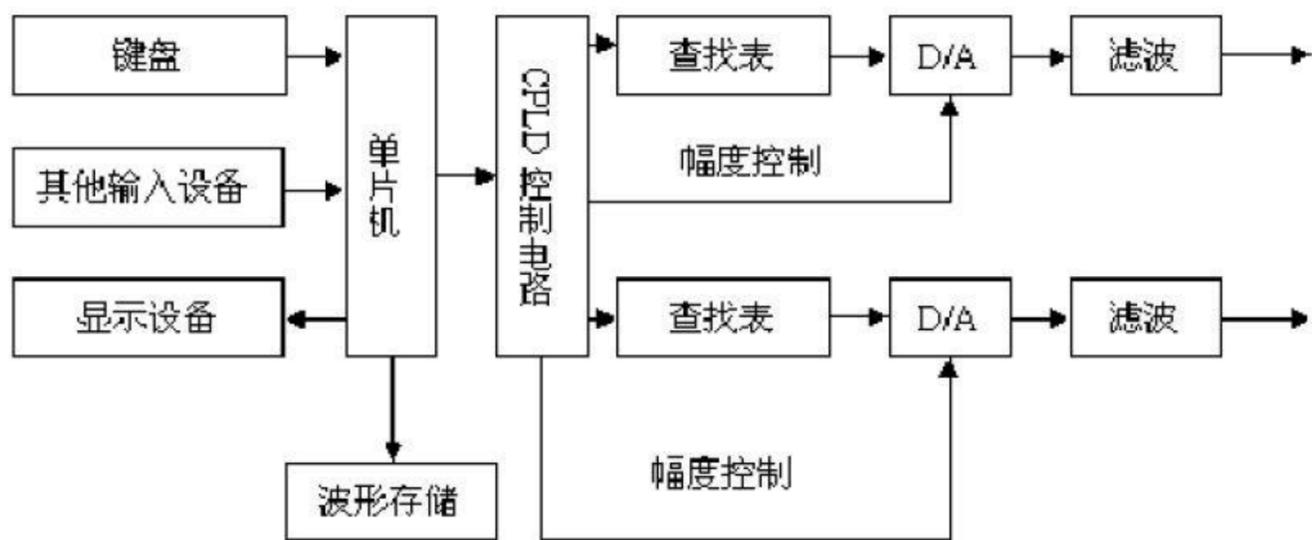


图 1. 系统总体设计方框图

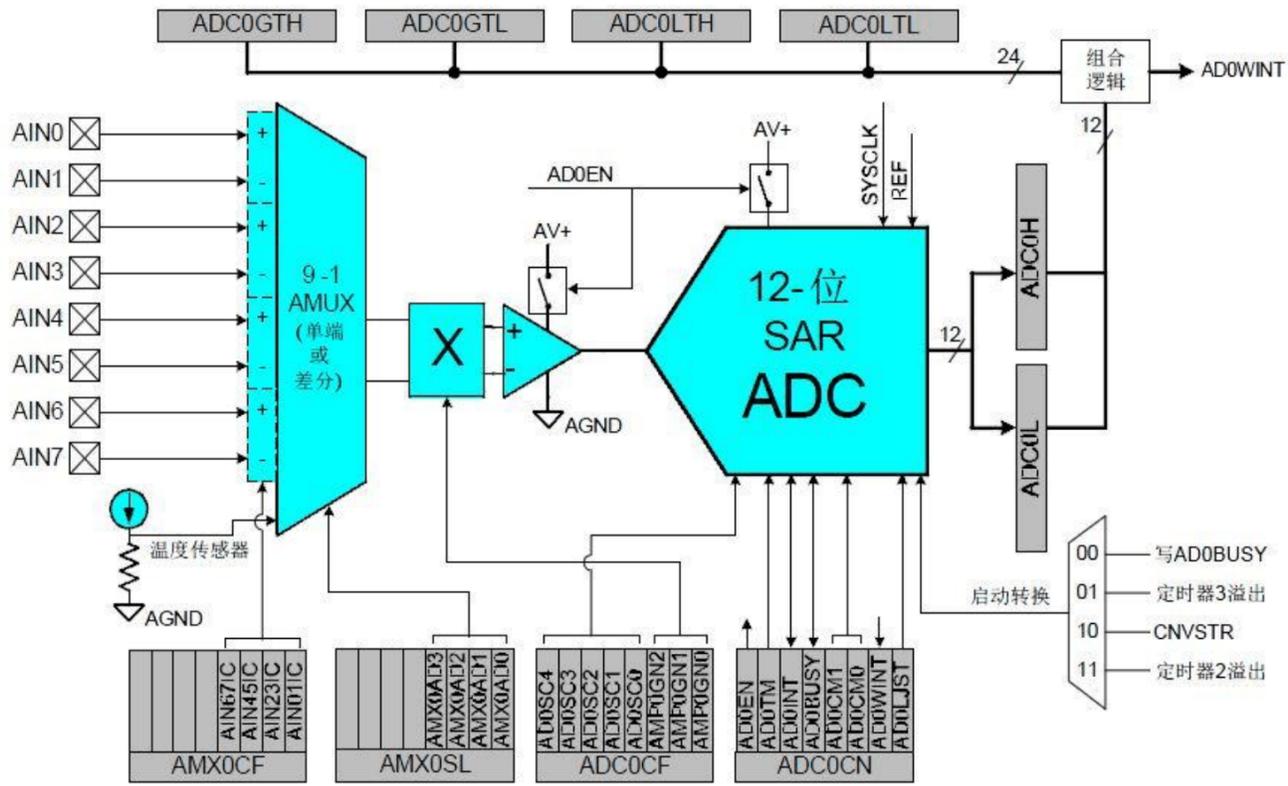
系统设计方案:

- 1、实现 A/D 芯片的模数转换功能，通过 keil 的 watch 窗口观察 ADC0 读取的数据的变化。
- 2、设置合适的采样频率和采样时间，对输入信号进行连续采样，对规定时间内的采样结果进行存储。
- 3、对已有数据进行 D/A 转换，实现数字量到模拟量的变化，并在示波器上显示结果。
- 4、添加单次按键触发等功能，实现在满足触发条件后，对一个采样周期内的输入进行存储和连续显示。
- 5、增加 1 通道输入，实现双踪示波。
- 6、综合上述情况实现完整的数字双踪示波器。

四、 单元电路设计及其初始化

1. ADC 电路的设计

图 5.1 12 位 ADC0 功能框图



初始化程序为：

```
void ADC0_Init (void)
```

```
{
    ADC0CN = 0x05;           // ADC0 T3           定时采样,左对齐
    REF0CN = 0x03;           //           启用内部基准源
    AMX0CF=0x00;             //           选择采样输入源
    ADC0CF = ((SYSCLK/2500000)-1) << 3; // ADConversion clock = 2.5MHz
    ADC0CF &= 0xf8;
    ADC0GTH=0Xff;
    ADC0GTL=0Xff;
    ADC0LTH=0X00;
    ADC0LTL=0X00;           // PGA gain = 2
    EIE2 |= 0x02;           //           启用 ADC 中断
}
```

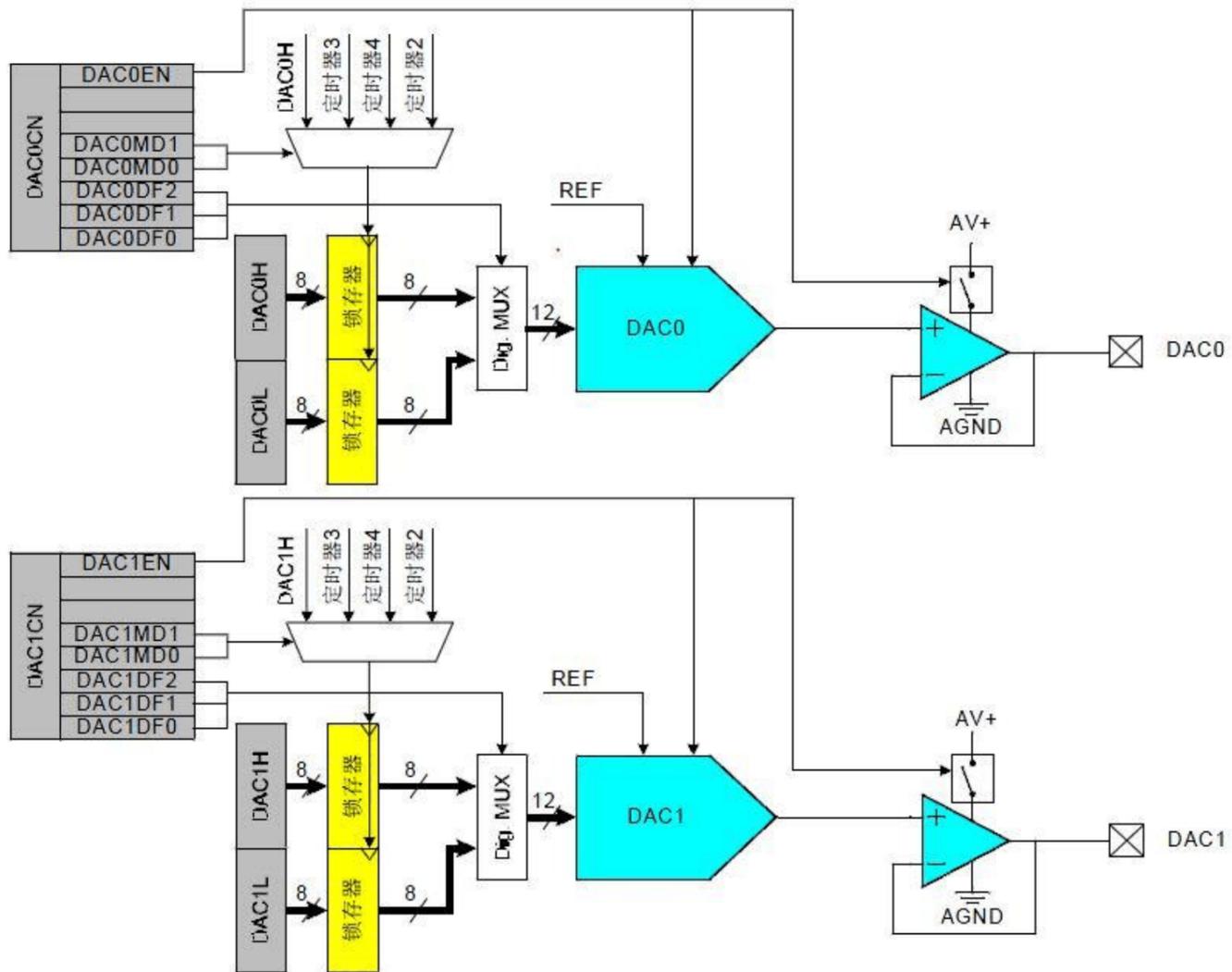
图 5.8 ADC0CN: ADC0 控制寄存器(C8051F020/1)

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
AD0EN	AD0TM	AD0INT	AD0BUSY	AD0CM1	AD0CM0	AD0WINT	AD0LJST	00000000
位7	位6	位5	位4	位3	位2	位1	位0 (可位寻址)	SFR地址: 0xE8
位 7:	AD0EN: ADC0 允许位 0: ADC0 禁止, ADC0 处于低功耗待机状态。 1: ADC0 允许, ADC0 处于活动状态, 并准备转换数据。							
位 6:	AD0TM: ADC 跟踪方式位 0: 当 ADC 被允许时, 除了转换期间之外一直处于跟踪方式。 1: 由 ADSTM1-0 定义跟踪方式。							
位 5:	AD0INT: ADC0 转换结束中断标志 该标志必须用软件清 '0'。 0: 从最后一次将该位清 0 后, ADC0 还没有完成一次数据转换。 1: ADC 完成了一次数据转换。							
位 4:	AD0BUSY: ADC0 忙标志位 读: 0: ADC0 转换结束或当前没有正在进行的数据转换, AD0INT 在 AD0BUSY 的下降沿被置 '1'。 1: ADC0 正在进行转换。 写: 0: 无作用 1: 若 ADSTM1-0=00b 则启动 ADC0 转换。							
位 3-2:	AD0CM1-0: ADC0 转换启动方式选择位。 如果 AD0TM = 0: 00: 向 AD0BUSY 写 1 启动 ADC0 转换。 01: 定时器 3 溢出启动 ADC0 转换。 10: CNVSTR 上升沿启动 ADC0 转换。 11: 定时器 2 溢出启动 ADC0 转换。 如果 AD0TM = 1: 00: 向 AD0BUSY 写 1 时启动跟踪, 持续 3 个 SAR 时钟, 然后进行转换。 01: 定时器 3 溢出启动跟踪, 持续 3 个 SAR 时钟, 然后进行转换。 10: 只有当 CNVSTR 输入为逻辑低电平时 ADC0 跟踪, 在 CNVSTR 的上升沿开始转换。 11: 定时器 2 溢出启动跟踪, 持续 3 个 SAR 时钟, 然后进行转换。							
位 1:	AD0WINT: ADC0 窗口比较中断标志。 0: 自该标志被清除后未发生过 ADC0 窗口比较匹配。 1: 发生了 ADC0 窗口比较匹配。							
位 0:	AD0LJST: ADC0 数据左对齐选择位。 0: ADC0H-ADC0L 寄存器数据右对齐。 1: ADC0H-ADC0L 寄存器数据左对齐。							

采用数据左对齐存储方式。由0端口输入。T3定时器启动, 当T3溢出时启动ADC0转换结束时产生15号中断。执行中断程序。

2. DA电路的设计

图 8.1 DAC 功能框图



初始化程序：

```

void DAC0_Init(void)
{
    DAC0CN = 0x82;
}

void DA0_Out (unsigned int con_volt)
{
    con_volt=con_volt<<2;
    DAC0L=con_volt%256;
    DAC0H=con_volt/256;
}
    
```

其中控制字DAC0CN说明如下：采用更新于写入DAC0H时，数据格式为高八位的低六位和低八为的高六位。

图 8.4 DAC0CN: DAC0 控制寄存器

R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	复位值
DAC0EN	-	-	DAC0MD1	DAC0MD0	DAC0DF2	DAC0DF1	DAC0DF0	00000000
位7	位6	位5	位4	位3	位2	位1	位0	SFR地址: 0xD4

位 7: DAC0EN: DAC0 允许位
 0: DAC0 禁止: DAC0 输出引脚为高阻态; DAC0 处于节电停机方式
 1: DAC0 允许: DAC0 正常输出; DAC0 处于工作状态。

位 6-5: 未用。读 = 00b; 写 = 忽略。

位 4-3: DAC0MD1-0: DAC0 方式位。
 00: DAC 输出更新发生在写 DAC0H 时。
 01: DAC 输出更新发生在定时器 3 溢出时。
 10: DAC 输出更新发生在定时器 4 溢出时。
 11: DAC 输出更新发生在定时器 2 溢出时。

位 2-0: DAC0DF2-0: DAC0 数据格式位:
 000: DAC0 数据字的高 4 位在 DAC0H[3:0], 低字节在 DAC0L 中。

DAC0H				DAC0L			
			MSB				LSB

001: DAC0 数据字的高 5 位在 DAC0H[4:0], 低 7 位在 DAC0L[7:1]。

DAC0H				DAC0L			
			MSB				LSB

010: DAC0 数据字的高 6 位在 DAC0H[5:0], 低 6 位在 DAC0L[7:2]。

DAC0H				DAC0L			
			MSB				LSB

011: DAC0 数据字的高 7 位在 DAC0H[6:0], 低 5 位在 DAC0L[7:3]。

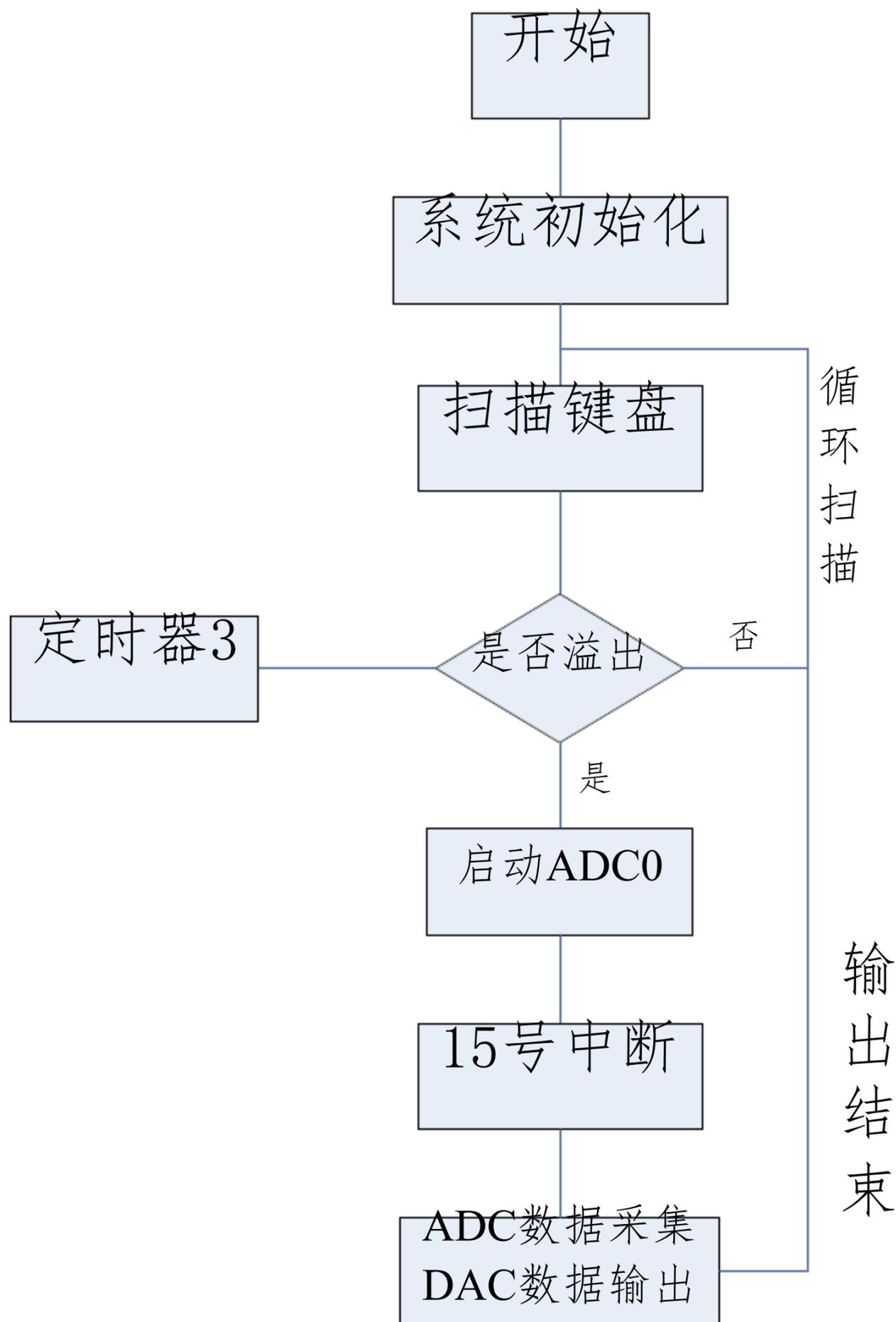
DAC0H				DAC0L			
			MSB				LSB

1xx: 高有效字节在 DAC0H 中, 低 4 位在 DAC0L[7:4]。

DAC0H				DAC0L			
			MSB				LSB

五、 软件设计

1.软件流程图:



3. 相关程序及其功能

AD DA转换

```

#include <c8051f020.h>
sfr16 ADC0    = 0xbe;
sfr16 ADC0GT  = 0xc4;
sfr16 ADC0LT  = 0xc6;
sfr16 TMR3RL  = 0x92;           //Timer3 reload value
sfr16 TMR3    = 0x94;
  
```

```

#define SYSCLK    22118400    //18432000
#define SAMPLE_RATE 2500000    // Sample frequency in Hz    采样频率
#define INT_DEC    1    // integrate and decimate ratio
#define uchar unsigned char
#define uint unsigned int
void SYSCLK_Init (void);
void ADC0_Init (void);
void Timer3_Init (int counts);
void ADC0_ISR (void);
void DA0_Out (unsigned int con_volt);
void DAC0_Init();
uchar int_dec=INT_DEC;
uchar tt=0;
void main (void)
{
    WDTCN = 0xde;
    WDTCN = 0xad;
    SYSCLK_Init ();
    Timer3_Init (SYSCLK/SAMPLE_RATE);
    ADC0_Init ();
    AD0EN = 1;
    DAC0_Init();
    EA = 1;
    while (1)
    {

    }
}
void SYSCLK_Init (void)    // 配置系统时钟使用外部晶振22.1184MHz
{
    int i;
    OSCXCN = 0x67;    //    晶体振荡器未用，晶体振荡器方式
    for (i=0; i < 256; i++);    // XTLVLD blanking interval (>1ms)
    while (!(OSCXCN & 0x80));    // Wait for crystal osc. to settle
    OSCICN = 0x88;    //    选择外部振荡器作为系统时钟，内部振荡器
    禁止
}
void ADC0_Init (void)
{
    ADC0CN = 0x06;    // ADC0 T3    定时采样,左对齐
    REF0CN = 0x03;    //    启用内部基准源
    AMX0SL = 0x00;    //    选择采样输入源0
    ADC0CF = (SYSCLK/2500000) << 3;    // ADC conversion clock = 2.5MHz
    ADC0CF|= 0x00;
}

```

```

ADC0LTL= 0x00;
ADC0GTH= 0xFF;
ADC0GTL= 0xFF;
EIE2 |= 0x02;           //          ADC 中断
}
void DAC0_Init(void)
{
    DAC0CN = 0x80;
}
void DA0_Out (unsigned int con_volt)
{
    unsigned char hbyte,lbyte;
    hbyte=con_volt/256;
    lbyte=con_volt%256;
    DAC0L=lbyte;
    DAC0H=hbyte;
}
void Timer3_Init (int counts)
{
    TMR3CN = 0x02;
    TMR3RL = -counts;
    TMR3   = 0xffff;
    EIE2  &= ~0x01;
    TMR3CN |= 0x04;
}
void ADC0_ISR (void) interrupt 15
{
    AD0INT = 0;
    int_dec--;
    if (int_dec == 0)
    {
        int_dec =INT_DEC;
        DA0_Out(ADC0);
    }
}

```

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：<https://d.book118.com/696145010004010101>