第 2-17 讲：PCA 实现数模转换（DAC）

1. 学习目的

1. 了解 DAC 数模转换原理及 RC 积分电路原理。
2. 掌握 STC8A8K64D4 系列单片机实现 DAC 功能的硬件和软件设计。

2. DAC 简介

DAC (全称是 Digital to Analog Convertor)数模转换器是一种将数字信号转换为模拟信号（以电流、电压或）的设备或电路。在很多数字系统中（例如计算机、单片机），信号以数字方式（0 或者 1）存储和传输，而数模转换器 DAC 可以将这样的信号转换为模拟信号，从而使得他们能够被外界（人或其他非数字系统）识别。数模转换器 DAC 的常见用法是在音乐播放器中将数字形式存储的音频信号输出为模拟的声音。

T 型电阻网络方式是一种常见的 DAC 实现方法，由 T 型电阻网络和运算放大器组成，下图是 8 位 DAC 的原理示意图。输入数字量中的每位都按其权值分别转换为模拟量，之后通过运算放大器求和相加。

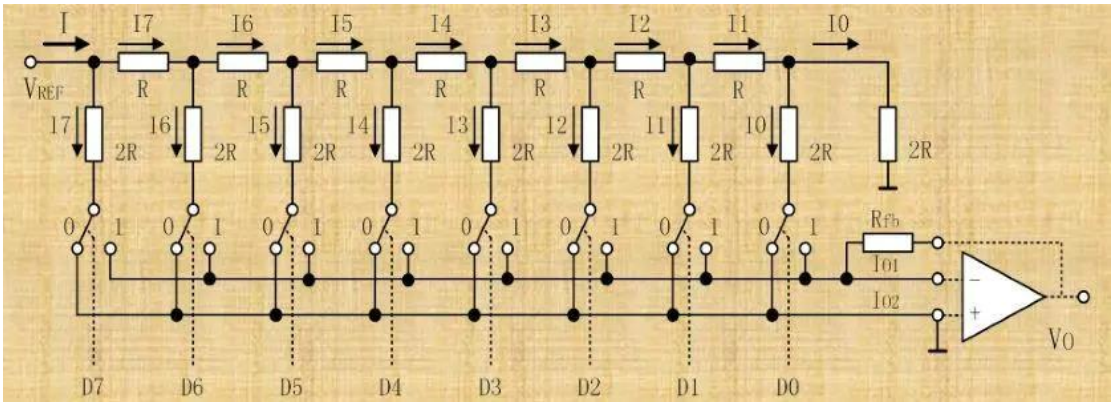

图 1：T 型电阻网络示意图

上图中，由于运算放大器的反相输入端为“虚地”，所以无论模拟开关连接到“0”还是 “1”，从 T 形电阻网络节点对“地”往右看的等效电阻均为 R，由此可计算出基准电流 $I=V_{REF}/R$。再根据电流可计算出流过各个分支的电流从右向左（I0~I7）依次是 I/2、I/4、I/8、I/16、I/32、I/64、I/128 和、I/256。

由此，每一位数字量都发挥了有效的位权，流向运算放大器反相输入端的总电流如下：

$$I_{\Sigma} = I_7 \times D7 + I_6 \times D6 + \cdots + I_0 \times D0$$

该电流经过运算放大器换成模拟电压输出，从而实现由数字信号到模拟信号的转换。这里以 8 位 DAC 示例，输出电压有 256 种变化，当然，这种 T 形电阻网络的转换原理可以推广到 n 位，实现 n 位 DAC。

市面上很多单片机片内集成了 DAC 外设，并且也有专用独立 DAC 芯片供用户选择使用，那为什么还需要使用 PWM 实现 DAC?

这是因为虽然市面上很多集成了 DAC 的单片机和独立 DAC 芯片，他们使用起来更方便，并且精度也高，但是很多时候，我们并不需要很高的精度，我们选择使用的单片机可能没有 DAC 外设，但是一般都有 PWM，因此，出于对成本的考虑，我们会使用 PWM 实现 DAC，从而节省成本。

3. 硬件电路设计

STC8A8K64D4 系列单片机片内没有集成 DAC 外设，因此，IK-64D4 开发板通过 PCA 工作于 PWM 模式输出高速 PWM 脉冲配合 RC 滤波电路实现 DAC 功能。

实现 DAC 转换是基于将高速 PWM 信号通过 RC 电路整合成比较平缓的电压信号作为模拟输出，通过改变高速 PWM 信号的占空比达到改变输出电平信号幅度的目的。为了达到比较理想的电压信号输出，P7.0 口输出的 PWM 信号经二阶 RC 滤波电路整合，如下图所示。

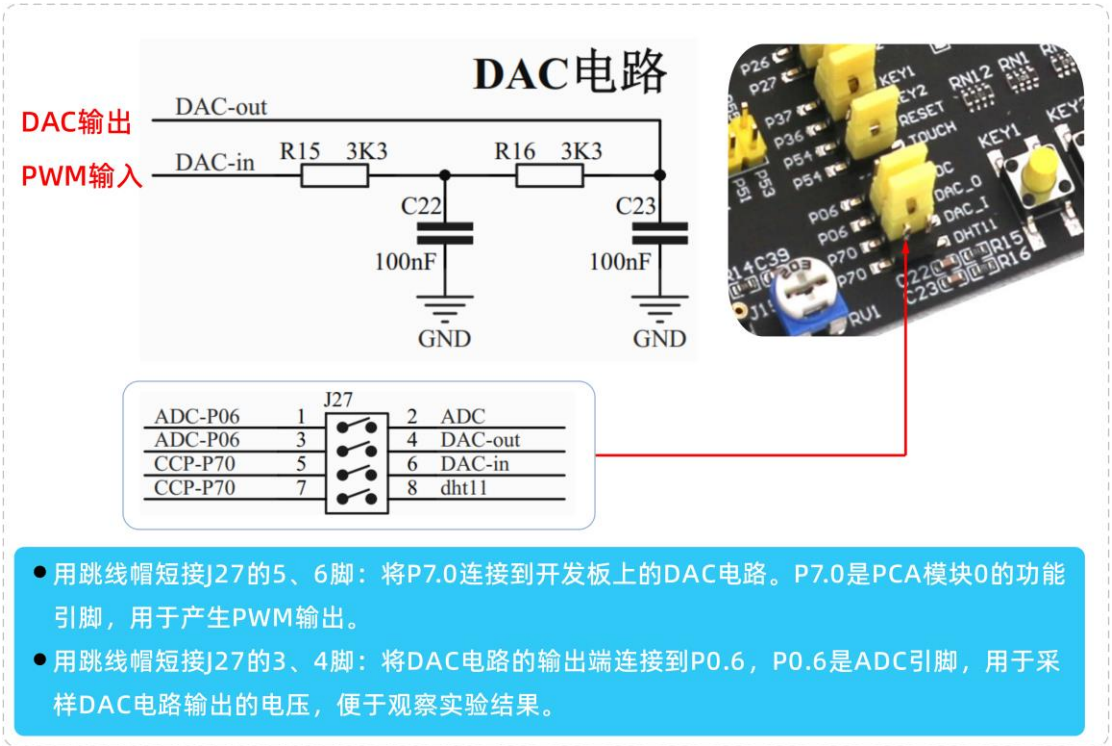

图 2：PWM 实现 DAC 电路

✧ 注：开发板 J27 端子需使用短路帽短接 P06 和 DAC_O 以及 P70 和 DAC_I，并且将 J27 端子的 P06 和 ADC 以及 P70 和 DHT11 的短路帽去掉。

4. 软件设计

4.1. PWM 实现 DAC 实验

✧ 注：本节的实验是在“实验 2-11-1：ADC 采样电位器电压（查询方式）”的基础上修改，本节对应的实验源码是：“实验 2-17-1： PWM 实现 DAC 实验”。

✧ 关于 PWM 的内容，读者可以参阅《第 2-16 讲：可编程计数器阵列 PCA》中的 PCA 实现 PWM 部分的内容。

4.1.1. 实验内容

配置 PCA 模块 0 工作于 PWM 模式， PCA 配置如下：

- 系统时钟：24MHz。
- PCA 时钟源：系统时钟= 24Mhz。
- 功能引脚：P7.0。
- 中断：不开启中断。
- PWM 位数：8 位。

程序运行后，在主循环中不断改变 PWM 的占空比，从而改变 P7.0 输出的电压值。实验中为了方便观察实验现象，配置 P0.6 为 ADC 功能引脚，并通过跳线帽将其连接到 DAC 电路的输出端，这样，就可以实时获取 DAC 输出的电压值并通过串口输出，以便于我们观察实验现象（DAC 输出电压值的改变）。

4.1.2. 代码编写

1. 新建一个名称为“pca.c”的文件及其头文件“pca.h”并保存到工程的“Source”文件夹，并将“pca.c”加入到 Keil 工程中的“SOURCE”组。
2. 引用头文件

因为在“pca.c”文件中使用了“pca.c”文件中的函数，所以需要引用下面的头文件“pca.h”。

代码清单：引用头文件

```
1.  //引用 pca 的头文件
2.  #include   " pca.h"
```

3. PCA 初始化

本例中，我们配置 PCA 模块 0 工作于 PWM 模式，PCA 时钟源使用系统时钟（24Mhz），功能引脚为 P7.0，代码清单如下。

代码清单：PCA 初始化

```
1.   /*******************************************************************************
2.   * 描   述 : PCA 初始化
3.   * 入   参 : 无
4.   * 返回值 : 无
5.   *******************************************************************************/
6.  void pca_init(void)
7.  {
8.     CCON = 0x00;        //CF、CR、CCF1、CCF0 位均清零
9.
10.    P_SW1 &= 0xEF;      //PCA 模块 0 功能引脚选择 P7.0
11.    P_SW1 |= 0x20;
12.    /*--------------------PCA 模式寄存器 CMOD 配置---------------------------
13.      位 7   位 6   位 5   位 4    位 3~位 1     位 0
```

```
14.      CIDL    x     x     x     CPS[2~0]     BCF
15.       0      x     x     x       100        0
16.
17.     CIDL=0：空闲模式下仍然计数
18.     CPS[2~0]=100：PCA 时钟源选择:系统时钟
19.     BCF=0：禁止 PCA 计数器溢出中断
20.    ----------------------------END-----------------------------------*/
21.    CMOD = 0x08;
22.    CL = 0x00;           //PCA 计数器赋初值
23.    CH = 0x00;           //PCA 计数器赋初值
24.
25.    IP &= ~0x02;         //中断优先级配置为 2（较高优先级）
26.    IPH |= 0x02;
27.     /*-----------------PCA 模块 0 模式控制寄存器 CCAPM0 配置-----------------
28.     位 7   位 6    位 5    位 4     位 3    位 2   位 1    位 0
29.      x   ECOM0  CCAPP0  CCAPN0   MAT0   TOG0  PWM0   ECCF0
30.      x     0      0       0       0      0     1       0
31.
32.     ECOM0=0：关闭 PCA 模块 0 的比较功能
33.     CCAPP0=0：关闭 PCA 模块 0 的上升沿捕获
34.     CCAPN0=0：关闭 PCA 模块 0 的下降沿捕获
35.     MAT0=0：关闭 PCA 模块 0 的匹配功能
36.     TOG0=0：关闭 PCA 模块 0 的高速脉冲输出功能
37.     PWM0=1：开启 PCA 模块 0 的脉宽调制输出功能
38.     ECCF0=0：禁止 PCA 模块 0 的匹配/捕获中断
39.    ----------------------------END-----------------------------------*/
40.    CCAPM0 = 0x02;
41.
42.    PCA_PWM0 &= 0x3F;              //PCA 模块 0 工作于 8 位 PWM 功能
43.    PCA_PWM0 &= 0xFC;              //EPC0H 位和 EPC0L 位置 0
44.    CCAP0L = 0x00;            //PCA 比较值寄存器赋初值
45.    CCAP0H = 0x00;            //PCA 比较值寄存器赋初值
46.    CR = 1;                   //启动 PCA 计数器阵列计数
47. }
```

4. 主函数

主函数中初始化 PCA 和 ADC，PCA 计数启动后，在主循环中不断改变 PWM 的占空比，使得 DAC 输出不同的电压。

ADC 采样 DAC 输出的电压值后，通过串口输出采样值及其对应的电压值，代码清单如下。

代码清单：主函数

```
1.  /*************************************************************************
2.   * 描  述 : 主函数
```

```
3.    * 入  参 : 无
4.    * 返回值 : 无
5.    ******************************************************************************/
6.   int main(void)
7.   {
8.      u16 adc_value;  //存放 ADC 采样值
9.      float voltage;  //存放 ADC 采样值计算后的电压值
10.     u16 TempData;
11.
12.     P2M1 &= 0xBF;   P2M0 &= 0xBF;     //设置 P2.6 为准双向口（LED1）
13.     P3M1 &= 0xFE;   P3M0 &= 0xFE;     //设置 P3.0 为准双向口（串口 1 的 RxD）
14.     P3M1 &= 0xFD;   P3M0 |= 0x02;     //设置 P3.1 为推挽输出（串口 1 的 TxD）
15.     P7M1 &= 0xFE;   P7M0 |= 0x01;     //设置 P7.0 为推挽输出
16.
17.     pca_init();
18.     uart1_init();                 //串口 1 初始化
19.     adc_config();                 //初始化 ADC
20.     EA = 1;                       //使能总中断
21.
22.     while(1)
23.     {
24.        TempData++;
25.        CCAP0H = (u8)(256 - TempData);      //P7.0 引脚输出频率不变但占空比不断变化的脉冲信号
26.        if(TempData>138)TempData=1;          //占空比达到很大时重新设定占空比
27.        delay_ms(5);
28.        adc_value = get_adc_value();                  //读取 ADC 采样值
29.        voltage   = (2.5*adc_value)/4096;               //将 ADC 采样值转换为电压（单位 V）
30.        printf("ADC 采样值:  %d\r\n",adc_value);       //串口打印上传的采集的原始值
31.        printf("采样值对应的电压：%.2fV\r\n",voltage);//串口打印 ADC 采样电压
32.        delay_ms(200);                                //延时 200ms，方便观察数据
33.        led_toggle(LED_1);  //翻转指示灯 D1 状态，指示一次 ADC 采样完成
34.     }
35.  }
```

4.1.3. 硬件连接

本实验通过 P7.0 输出 PWM 信号实现 DAC 输出，P0.6 配置为 ADC 用于采样 DAC 电路输出的电压，按照下图所示连接跳线帽。

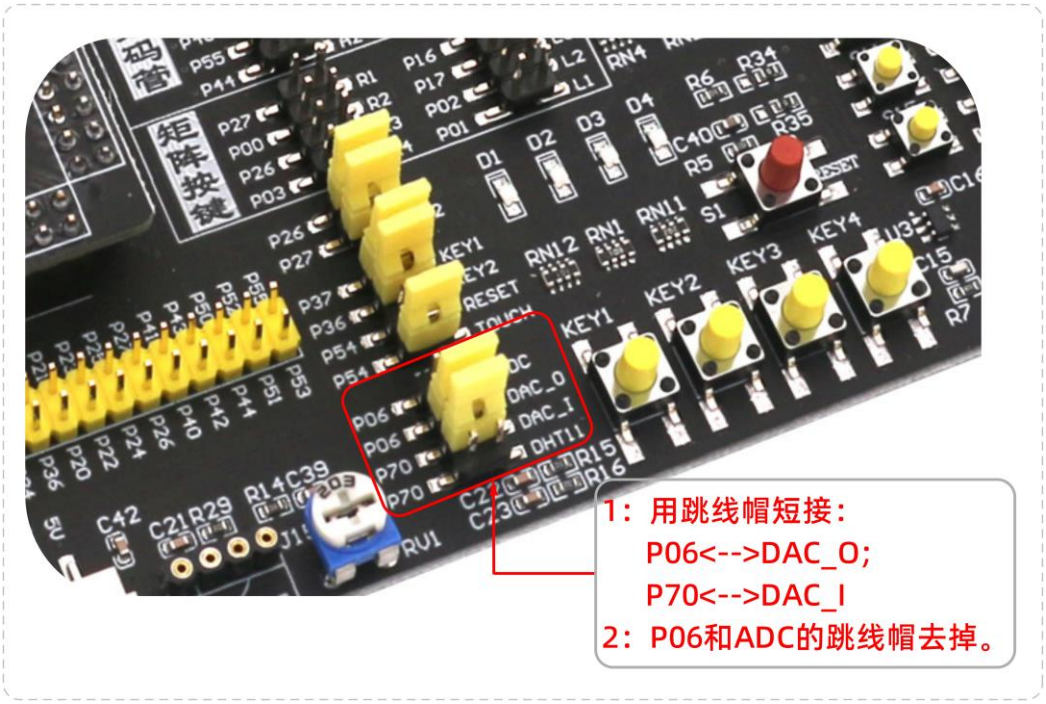

图 3：硬件连接

4.1.4. 实验步骤

1) 解压“…\第 3 部分：配套例程源码”目录下的压缩文件“实验 2-17-1： PWM 实现 DAC 实验”，将解压后得到的文件夹拷贝到合适的目录，如“D\STC8”（这样做的目的是为了防止中文路径或者工程存放的路径过深导致打开工程出现问题）。
2) 双击“…\pca_dac\project”目录下的工程文件“pca_dac.uvproj”。
3) 点击编译按钮编译工程，编译成功后生成的 HEX 文件“pca_dac.hex”位于工程的“…\pca_dac\Project\Object”目录下。
4) 打开 STC-ISP 软件下载程序，下载使用内部 IRC 时钟，IRC 频率选择：24MHz。
5) 电脑上打开串口调试助手，选择开发板对应的串口号，将波特率设置为 9600bps。程序运行后，在串口接收窗口可以看到开发板上报的 ADC 采样值及其对应的电压值，如下图所示。

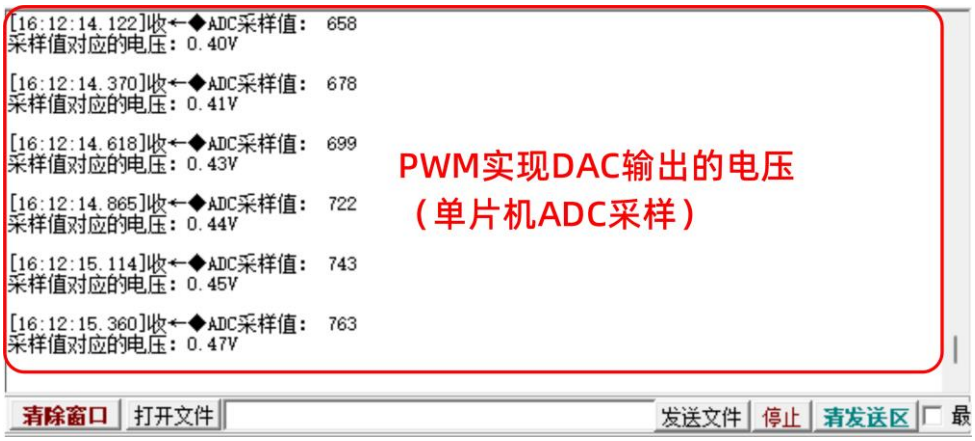

图 4：串口调试助手中观察电压值