内蒙古科技大學

INNER MONGOLIA UNIVERSITY OF SCIENCE & TECHNOLOGY

信 息 工 程 学 院

《物联网创新设计实践》课程实验报告

题　　目:	数字电压表的设计与调试
学生姓名:	王某某
学　　号:	2067165110
专　　业:	建筑电气与智能化
班　　级:	自动化 2020-3
指导教师:	张某某　教授

2022 年 10 月 17 日

目　　录

1 实验内容

1.1 实验目的

(1) 理解和掌握单片机 AD 扩展软硬件设计方法。

(2) 掌握 TLC549 模数转换扩展电路硬件设计及驱动程序设计方法。

(3) 掌握数字电压表软硬件设计及调试方法。

1.2 实验内容及要求

完成数字电压表软硬件设计与调试，实验箱通过电位器输出 0-5V 电压信号, 单片机通过 TLC549 模数转换芯片采集该电压, 并显示在 4 位 LED 数码管，要求写出实验报告，给出工程文件及调试视频。

实验报告要求绘制出主程序流程图、AD 转换子程序流程图、中断服务子程序流程图初始化子程序流程图。

2 硬件设计

2.1 TLC549 八位 AD 转换芯片介绍

AD 转换芯片的作用是将模拟信号转换为数字信号,TLC549 是 8 位串行 A/D 转换器芯片，可与通用微处理器、控制器通过 CLK、CS、DATAOUT 三条口线进行串行接口。具有 4MHz 片内系统时钟和软、硬件控制电路，转换时间最长 17μs，TLC549 为 40000 次/s。总失调误差最大为 ±0.5LSB，典型功耗值为 6mW。采用差分参考电压高阻输入，抗干扰，可按比例量程校准转换范围，VREF-接地，VREF+ - VREF-≥1V，可用于较小信号。

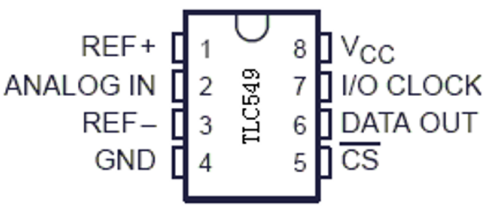

图 1　TLC549 八位 AD 转换芯片

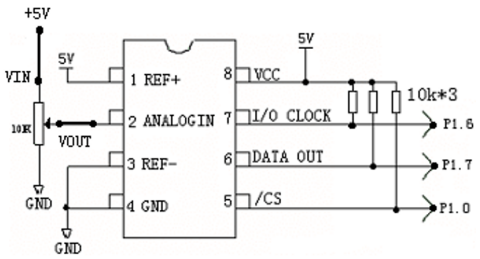

图 2　TLC549AD 转换电路原理图

2.2 TLC549 管脚功能及 AD 转换电路原理图

表 1　TCL549 芯片管脚定义

管脚名称	描述
REF+	正基准电压输入,2.5V≤REF+≤Vcc+0.1。
REF -	负基准电压输入端，-0.1V≤REF-≤2.5V。
VCC	系统电源 3V≤Vcc≤6V。
GND	接地端。
/CS	芯片选择输入端。
DATA OUT	转换结果数据串行输出端，高位在前，低位在后。
ANALOG IN	模拟信号输入端，0≤ANALOGIN≤Vcc。
I/O CLOCK	输入/输出时钟输入端，同于同步芯片的输入输出操作。

TLC549 的管脚功能定义如表 1 所示，其管脚分布如图 1 所示。

单片机扩展 TLC549AD 转换电路原理图如图 2 所示, 单片机 I/O 口分别与 I/O CLOCK、DATA OUT 及/CS 相连, 模拟量收入端接 ANALOG IN, REF+、REF-分别接 +5V 和 GND。

2.3 TLC549 八位 AD 转换芯片的工作时序

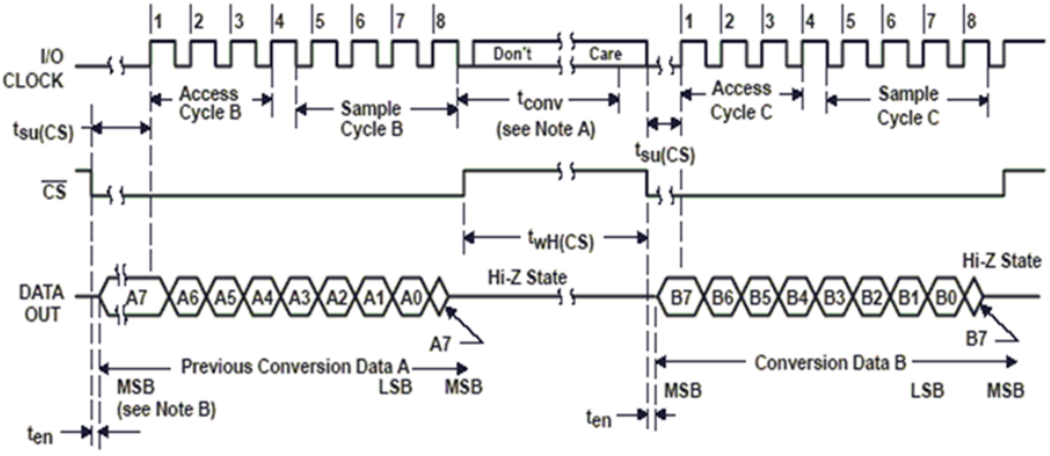

图 3　TLC549 工作时序图

TLC549 八位 AD 转换芯片的工作时序如图 3 所示。

当/CS 变为低电平后，TLC549 芯片被选中，同时前次转换结果的最高有效位 MSB（A7）自 DATA OUT 端输出，接着要求自 I/O CLOCK 端输入 8 个外部时钟信号，前 7 个 I/O CLOCK 信号的作用，是配合 TLC549 输出前次转换结果的 A6-A0 位，并为本次转换做准备：在第 4 个 I/O CLOCK 信号由高至低的跳变之后，片内采样/保持电路对输入模拟量采样开始，第 8 个 I/O CLOCK 信号的下降沿使片内采样/保持电路进入保持状态并启动 A/D 开始转换。转换时间为 36 个系统时钟周期，最大为 17us。直到 A/D 转换完成前的这段时间内，TLC549 的控制逻辑要求：或者/CS 保持高电平，或者 I/O CLOCK 时钟端保持 36 个系统时钟周期的低电平。由此可见，在自 TLC549 的 I/O CLOCK 端输入 8 个外部时钟信号期间需要完成以下工作：读入前次 A/D 转换结果；对本次转换的输入模拟信号采样并保持；启动本次 A/D 转换开始。

2.4 数字电压表电路原理图

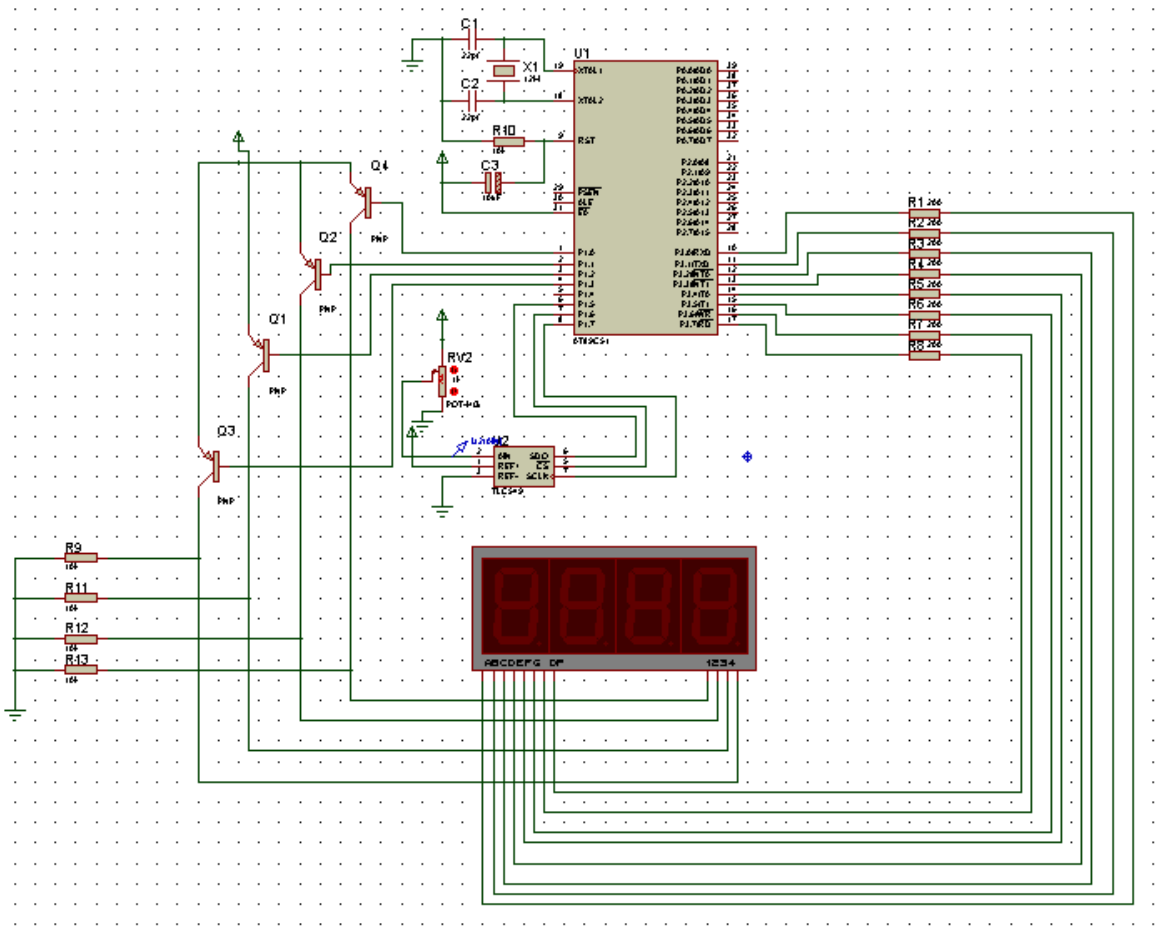

图 4　数字电压表电路原理图

数字电压表的电路原理图如图 4 所示, 包括由电源电路、时钟电路及复位电路组成的单片机最小系统电路, AD 转换接口电路及 4 位数码管驱动电路。P1.5-P1.7 分别与 TLC549 的 CLK、/CS 及 DOUT 相连,4 位数码管驱动电路采用动态刷新工作模式, P1.0-P1.3 输出 LED 数码管驱动电路的字位码, P3 口输出字型码。

3 软件设计

3.1 主程序设计

主程序流程图如图 5 所示, 首先调用系统初始化子程序, 这里主要是对定时器 T0 初始化, T0 中断产生两个时标,1 个 2ms 的时标 flag 用于刷新 4 位 LED 数码管,1 个 80ms 时标 flag1 用于更新电压显示值。80ms 时标到时, 读取当前 AD 值, 进行标度变换并更新数据显示缓冲区。2ms 时间到, 通过 P3 口和 P1 口分别输出字型码和字位码, 实现 4 位 LED 数码管的动态刷新。

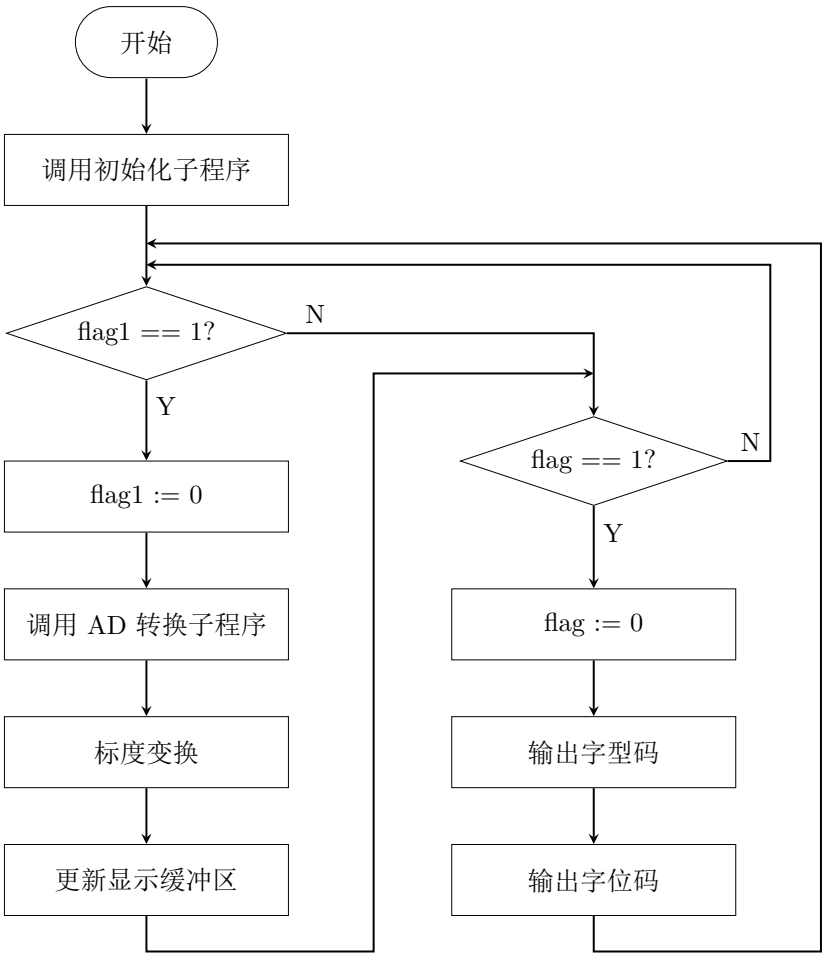

图 5　主程序流程图

3.2 系统初始化子程序设计

初始子程序流程图如图 6 所示[1]。

首先，修改 TMOD 的值, 将 T0 设置为非门控定时器方式 1(16 位定时器方式), 之后将 EA、ET0 置为 1 开 T0 中断，设置 T0 计数器 TH0、TL0 的初值，最后将 TR0 置为 1 启动定时器 T0。

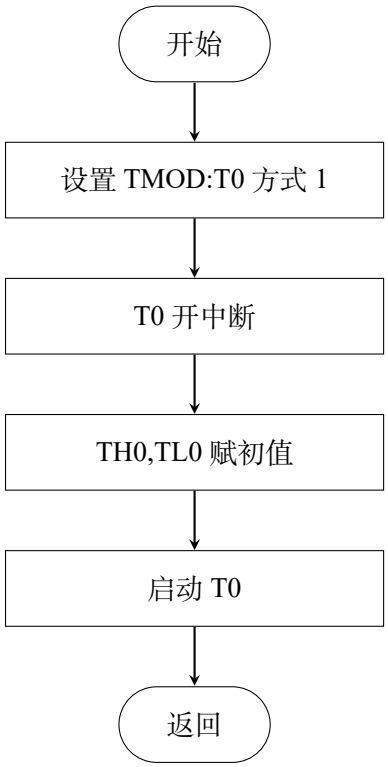

图 6　系统初始化子程序流程图

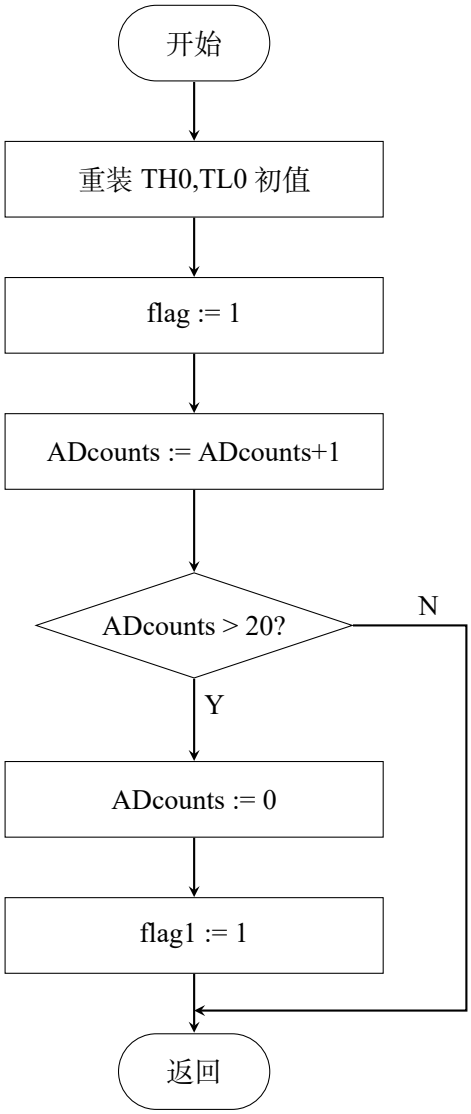

图 7　中断服务子程序流程图

3.3 中断服务子程序设计

中断服务子程序流程图如图 7 所示, 每 2ms 进入中断, 进入中断服务子程序后首先重装 TH0,TL0 初值, 将 2ms 时标 flag 置为 1, 将计数器 ADcounts 增 1, 增到 40 时将

80ms 时标 flag1 置为 1。

3.4 AD 转换子程序设计

TLC549AD 转换的关键算法如图 8 所示。

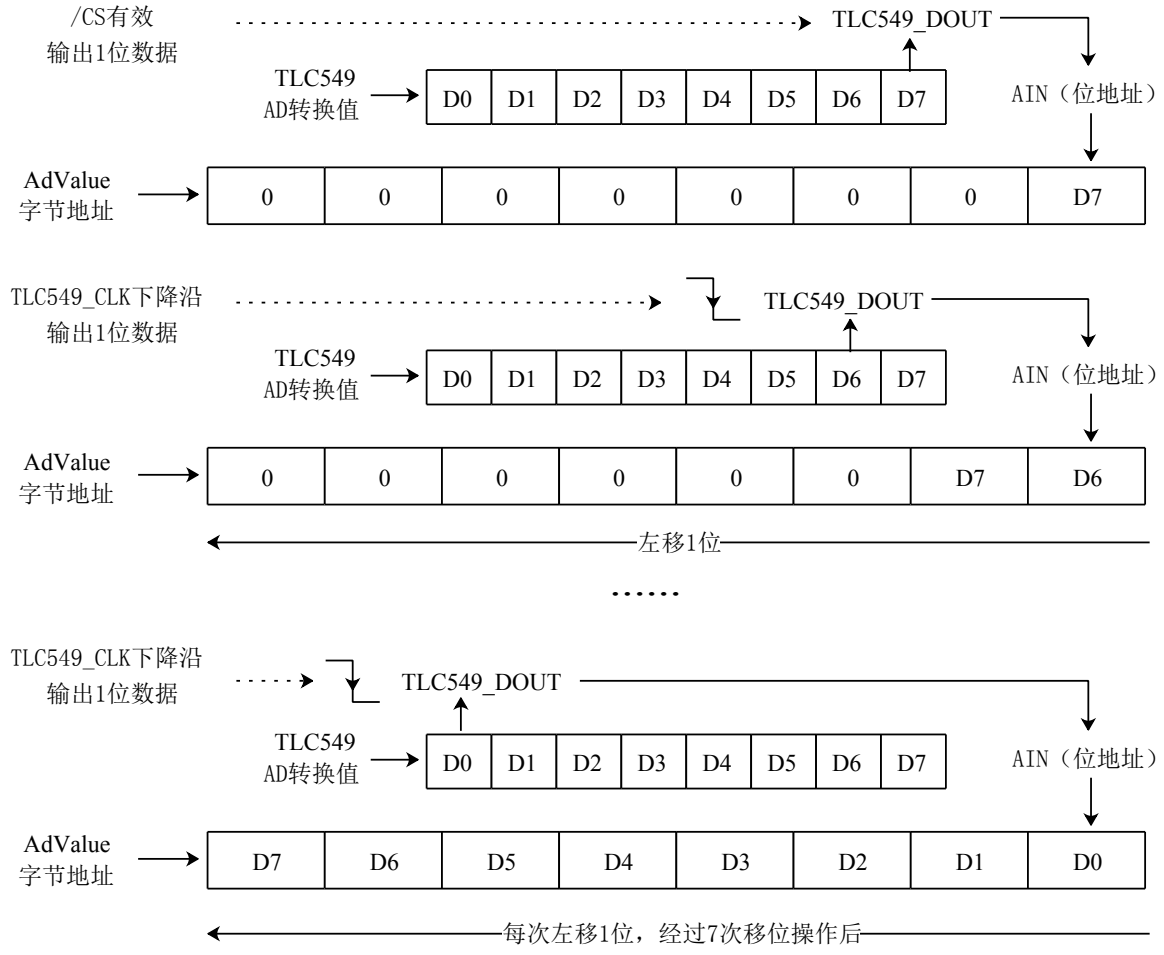

图 8　TLC549 移位算法图

AD 转换子程序设计如图 9 所示, 根据 TLC549 的时序图, 先将/CS 置为低电平。循环体执行 8 次, 每次先将 AdValue 左移 1 位, 再从 CLK 输出下降沿信号并从 DOUT 读出 1 位数据到 AdValue 最低位, 由于 TLC549 的数据输出高位在前低位在后，经过 8 次移位后正好读出 8 位 AD 值。最后将 AD 转换结果返回。

AD 转换子程序代码如下所示:

```
uint8_t getAdValue() {
    uint8_t i;
    TLC549_CS = 0;
```

```
    TLC549_DOUT = 1;
    TLC549_CLK = 0;
    for (i = 0; i < 8; i++) {
        AdValue = AdValue << 1;
        TLC549_CLK = 1;
        _nop_();
        AIN = TLC549_DOUT;
        TLC549_CLK = 0;
        _nop_();
    }
    TLC549_CS = 1;
    return AdValue;
}
```

3.5 关键算法

3.5.1 标度变换算法

标度变换就是将 AD 转换芯片返回的数字量转换为对应物理量的过程。

Y_{max}: 物理量的最大值，Y_{min}: 物理量的最小值, X_{max}:AD 的最大值，X_{min}:AD 的最小值,z 为当前 AD 返回值。

则：$\frac{Y_{max}-Y_{min}}{X_{max}-X_{min}}$ 为 1 格 AD 值对应的物理量大小, 在本例中为 $\frac{5.0-0.0}{255.0-0.0}$，加小数点的原因是将其设为浮点类型，否则计算机会取整，丢失精度。

因此，标度变换后的物理量 $= z \times \frac{Y_{max}-Y_{min}}{X_{max}-X_{min}}$。

在程序实现中, 需要强制转换为 float 类型, 为了便于传输和显示，将其放大 1000 倍取整（保留三位小数）。

标度变换核心代码如下所示：

```
if (adFlag) {
    adFlag = 0;
    fAdValue = (float)(ucAdValue * (5.0 / 255.0));
    uiAdValue = fAdValue * 1000;
}
```

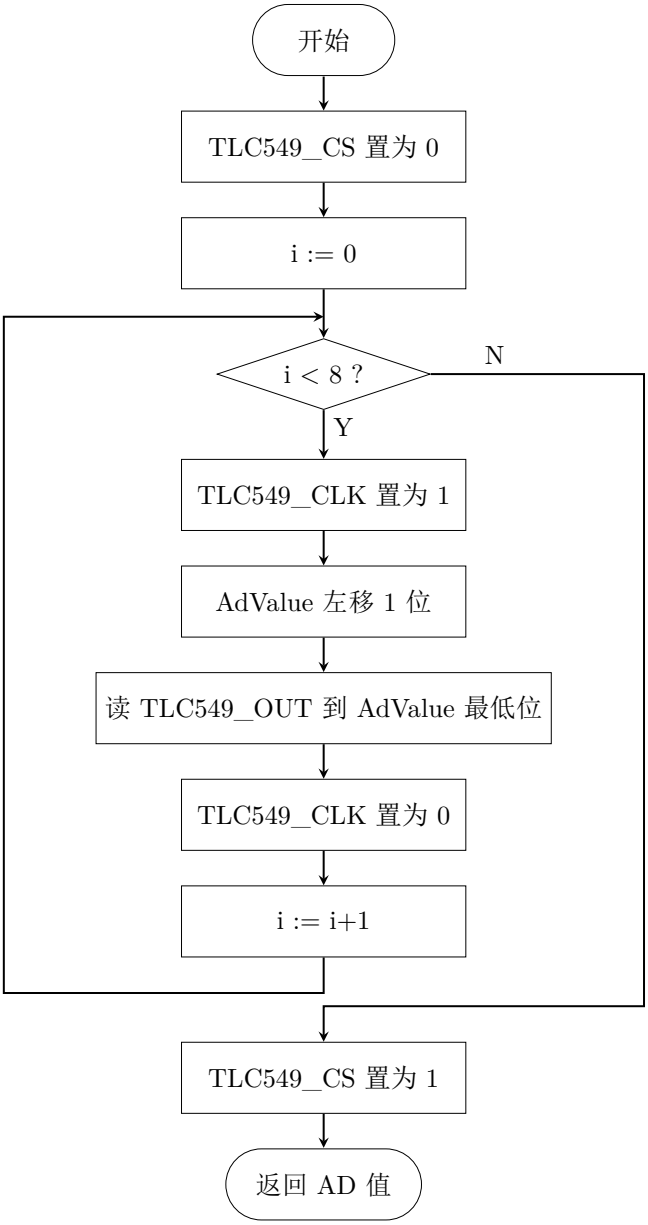

图 9　AD 转换子程序流程图

3.5.2 更新显示缓冲区算法

uiAdValue 返回的是放大 1000 倍的整型值, 需要将其千位, 百位, 十位, 个位拆分开以更新显示缓冲区。

通过除以 10 取余获得个位, 通过除以 10 取整经过 4 次循环分别取出千位, 百位, 十位, 个位并更新到缓冲区。

更新显示缓冲区核心代码如下所示:

```
1
2 for (j = 0; j < 4; j++) {
```

```
    dispBuf[3 - j] = uiAdValue % 10;
    uiAdValue /= 10;
}
```

4 实验调试过程

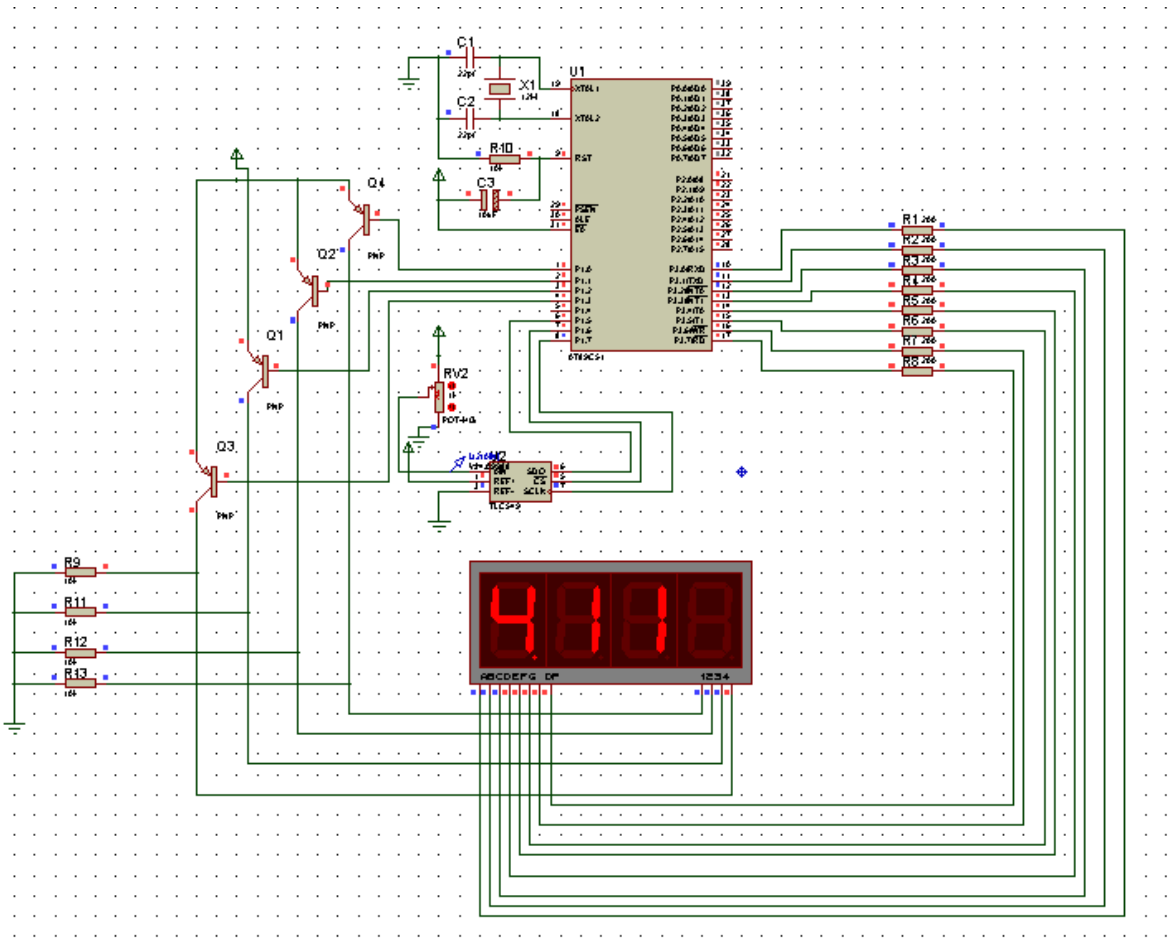

图 10　实验调试结果图

描述实验过程及取得的结果!

将在实验调试过程中遇到的问题及解决方法写在这里!

5 总结

将体会总结写在这里!

附录 A 源代码

```
#include <reg51.h>
#include <intrins.h>
#include "common.h"
#define FOSC 11059200ul
#define T0_H (65536-(2*FOSC)/(1000*12))/256
#define T0_L (65536-(2*FOSC)/(1000*12))%256

sbit COM0 = P1 ^ 0;
sbit COM1 = P1 ^ 1;
sbit COM2 = P1 ^ 2;
sbit COM3 = P1 ^ 3;
sbit TLC549_CS = P1 ^ 6;
sbit TLC549_DOUT = P1 ^ 5;
sbit TLC549_CLK = P1 ^ 7;
bit flag = 0;
bit flag1 = 0;
uint8_t ADcounts = 0;
uint8_t dispBuf[] = { 2, 0, 1, 8};
uint8_t code ledDeg[] = { 0xC0, 0xF9, 0xA4, 0xB0, 0x99, 0x92, 0x82, 0
   xF8, 0x80, 0x90 };
uint8_t bdata AdValue;
sbit AIN = AdValue ^ 0;

void initSys();

uint8_t getAdValue() {
    uint8_t i;
    TLC549_CS = 0;
    TLC549_DOUT = 1;
    TLC549_CLK = 0;
    for (i = 0; i < 8; i++) {
        AdValue = AdValue << 1;
        TLC549_CLK = 1;
```

```
33          _nop_();
34          _nop_();
35          AIN = TLC549_DOUT;
36          TLC549_CLK = 0;
37          _nop_();
38          _nop_();
39      }
40      TLC549_CS = 1;
41      return AdValue;
42  }
43
44  void main() {
45      uint8_t i = 0;
46      uint8_t j = 0;
47      uint8_t ucAdValue = 0;
48      uint16_t uiAdValue = 0;
49      float fAdValue = 0.0;
50      initSys();
51      while (1) {
52          if (flag1) {
53              flag1 = 0;
54              ucAdValue = getAdValue();
55              fAdValue = (float)(ucAdValue * (5.0 / 255.0));
56              uiAdValue = fAdValue * 1000;
57              for (j = 0; j < 4; j++) {
58                  dispBuf[3 - j] = uiAdValue % 10;
59                  uiAdValue /= 10;
60              }
61          }
62          if (flag) {
63              flag = 0;
64              switch (i) {
65              case 0:
66                  COM0 = 0; COM1 = 1; COM2 = 1; COM3 = 1;
67                  break;
```

```
            case 1:
                COM0 = 1; COM1 = 0; COM2 = 1; COM3 = 1;
                break;
            case 2:
                COM0 = 1; COM1 = 1; COM2 = 0; COM3 = 1;
                break;
            case 3:
                COM0 = 1; COM1 = 1; COM2 = 1; COM3 = 0;
                break;
            }
            P3 = ledDeg[dispBuf[i]];
            if (i == 0)
                P3 &= 0x7f;
            if (++i >= 4)
                i = 0;
        }
    }
}

void initSys() {
    TMOD &= 0xF0;
    TMOD |= 0x01;
    EA = 1;
    ET0 = 1;
    TH0 = T0_H;
    TL0 = T0_L;
    TR0 = 1;
}

void timer0ISR()interrupt 1 {
    TH0 = T0_H;
    TL0 = T0_L;
    flag = 1;
    if (ADcounts++ >= 40) {
        ADcounts = 0;
```

```
103             flag1 = 1;
104         }
105     }
```

附录 B 电路原理图