2021 年全国大学生电子设计竞赛陕西赛区
设计报告封面

作品编号：_____ （由组委会填写）

作品编号：_____ （由组委会填写）

参赛队编号 （参赛学校填写）	学校编号		组（队）编号		选题编号
	0	5	2	6	H

说　明

1. 为保证本次竞赛评选的公平、公正，将对竞赛设计报告采用二次编码。
2. 本页作为竞赛设计报告的封面和设计报告一同装订。
3. “作品编号”由组委会统一编制，参赛学校请勿填写。
4. “参赛队编号”由参赛学校编写，其中“学校编号”应按照巡视员提供的组委会印制编号填写，“组（队）编号”由参赛学校根据本校参赛队数按顺序编排，“选题编号”由参赛队员根据所选试题编号填写，例如：“0105B”或“3367F”。
5. 本页允许各参赛学校复印。

2021 年全国大学生电子设计竞赛

用电器分析识别装置（H 题）

2021 年 11 月 7 日

摘　　要

本系统以电能 IC 监测电路为核心，以 STM32F103 为主控制器，使用互感器对用电器负载的电流和电压信号进行转换，并采样得到安全且精确的检测电流和电压，通过电能 IC 芯片 HLW8032 的转换、计算和通讯，将用电器的电气特性传输至单片机控制中心进行逻辑运算处理，从而实现在市电和控制电路的电气隔离下，对运行中用电设备的电气参量的测量。

同时系统具有插拔用电设备后自动监测并识别设备的功能，将实时运行用电器显示在单片机的限显示屏上。

关键词：HLW8032 微型精确互感器 用电器监测

目　　录

用电器分析识别装置（H 题）

【本科组】

1 方案的选择与设计原理

1.1 负载端电压电流采样方案论证

方案一：互感器隔离采样方式

隔离采样时，互感器可以实现电气隔离，将市电与数字电路完全分离，负载端即使存在较大的变动，不会引发数字电路出现剧烈振荡，并且数字地和市电的地线相分离，不会造成电荷的悬浮，以致于残余的电荷将单片机击穿，对于整个电路有很好的保护作用，而且可以抵抗电网的干扰。

优点：设备的使用较为安全

缺点:需要采用额外的互感元器件,随之带来一定的相移差异,影响计量的精度,对监测电路的功率因数的测量带来困难。

方案二：铜锰采样电阻非隔离采样方式

非隔离采样是通过在主电路中串联一个阻值非常低的锰铜取样电阻,将负载电流转换为差分电压信号，然后送入电能计量芯片的模拟输入端口进行采样。由于电阻本身是一种线性器件。

优点：采样电路的结构较为简单，成本较为低廉，且不会引入相位偏移和增益变化，取样效果稳定。

缺点：采样电路对于后一级电路拓扑没有保护措施，强电和弱点在同一线路上共存，可能对试验人员的人身安全造成伤害，同时有可能在用电端负载急剧变化时击穿后级的监测控制电路。

综合比较,基于对试验人员人生安全和监测系统稳定性的考虑,选择方案一。

1.2 监测计量方案论证

方案一：采用外置 ADC 模块的测量

分立式 ADC 通过测量较低的采样电阻两端电压,将其输出量利用电压比较器进行采集并转换分析，得到电压值的数字信号。

优点：可以使用单片机的外设电路进行搭建，对于硬件的要求较低。

缺点：集成度以及精度不够高，外设成本较高，并且程序设计复杂。

方案二：采用电力计量 IC 芯片 HLW8032

HLW8032 是一款包含 Σ-Δ 型模-数转换(ADC)、功率计算功能、电能到频率转换器和一个串行接口的完整的功率测量芯片。它可以精确测量线电压和电流的瞬

时值和有效值，并能计算有功功率，视在功率和功率因素，功能齐全。

优点：精度高、功耗小、可靠性高、适用环境能力强等，适用于单相两线电力用户的电能计量。

缺点：对于特种芯片的依赖程度较高，测量的拓展性较低

综合比较，基于准确性以及系统集成度的考虑，选择方案二。

1.3 控制系统方案论证

方案一：采用 51 单片机进行控制和运算

51 单片机的发展较早，其程序的编写主要依赖于 C 语言。

优点：对于 51 单片机的技术支持较多，程序编写相对简单。

缺点：运行速度慢，内存容量偏小，已经不太适用于现代控制系统。

方案二：采用基于 STM32F103 的 ARM 单片机

STM32F103 系列单片机基于 ARM Cortex 内核，具有相对强大的运算处理能力和充裕的内存空间，能够胜任较为复杂的现代控制系统的操作控制任务，故适用于本题中电能计量芯片的控制。

缺点：编程难度较高，对于使用者的掌握能力要求较高。

综合比较，由于本题需要存储较多的数据，并进行一定次数的循环，考虑到这些要求对于单片机能力的挑战，故选择方案二。

2 系统理论分析与计算

2.1 用电器监测模块的分析

2.1.1 互感器的选择

考虑到负载端电流范围为 5mA-10A,因此电流互感器选择 ZMCT120，其额定电流为 10A，在 0-100A（采样电阻为 10 欧姆）均能线性工作，电压互感器选型为 ZMPT107-1，二者均体积小，精度高，一致性好。

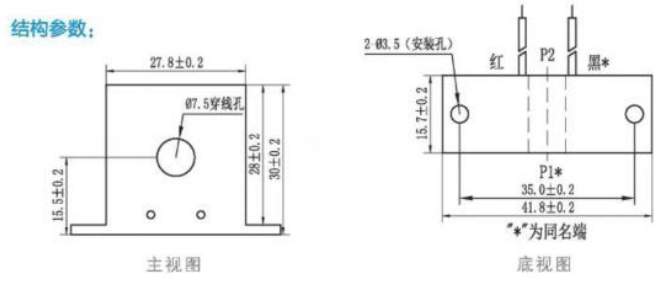

图1 ZMPT120 电流互感器结构示意图

2.1.2 处理模块分析

HLW8032 芯片是一种包含两个Σ-Δ型模-数转换(ADC)，高能电能计算电能功能和一个串行接口，它可以精确测量和计算有功电能、瞬时功率、IRMS、VRMS，采用低成本的互感器实现电流和电压的检测。

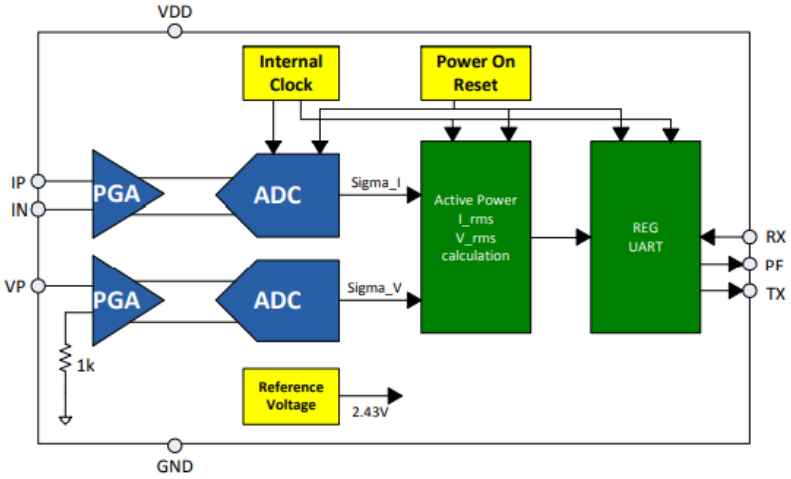

图2 HLW8032 内部功能框图

各指标参量分析计算：

（1） 电压有效值

电压有效值的计算公式如下：

$$U_{RMS}=\sqrt{\frac{1}{N}\sum_{n=1}^{N}u^2(n)}$$

$$u(n)=\frac{\text{电压参数寄存器}}{\text{电压寄存器}}\times\text{电压系数}\quad\left(\text{电压系数}=\frac{150\text{k}}{150k\cdot1000}=1\right)$$

利用电压通道 ADC 的输出结果连续计算有效值。电压有效值的计算方法是在 HLW8032 内部的 ADC 输出电压信号的平方进行低通滤波，即求平均值，然后将求取结果的平方根储存到 24 位寄存器中，与过零信号同步读取 VRMSx 寄存器，稳定电压电压有效值读数的变化。

（2） 电流有效值

因而同理利用电流通道 ADC 的输出结果得到电流有效值：

$$I_{RMS}=\sqrt{\frac{1}{N}\sum_{n=1}^{N}i^2(n)}$$

$$i(n)=\frac{\text{电流参数寄存器}}{\text{电流寄存器}}\times\text{电流系数}\quad\left(\text{电压系数}=\frac{2000}{6\times1000}=\frac{1}{3}\right)$$

（3）功率计算

有功功率计算：

$$有效功率\ \mathrm{P}=\frac{1}{nT}\int_{0}^{nT}u(t)i'(t)dt$$

$$其中：\ u(t)i'(t)=\frac{功率参数寄存器}{功率寄存器}\times 电压系数\times 电流系数$$

视在功率计算：

$$视在功率\mathrm{S}=有效电压\mathrm{U}_{RMS}\times 有效电流\mathrm{I}_{RMS}$$

（4）功率因数计算：

$$功率因数=\frac{有功功率P}{视在功率\mathrm{S}}$$

2.2 取样电阻的计算

2.2.1 负载电流取样电阻的计算

由于负载端用电器中电流的变化范围为$5\mathrm{mA}-10\mathrm{A}$,通过 2000：1 电流互感器后电流取样电阻流过的电流从$2.5\mu\mathrm{A}-10\mathrm{mA}$,而电能计量芯片的电流测量端口间能承受的最大差分电压为$30\mathrm{mV}$，在满足芯片耐压的前提下，为提高电流和功率测量的精确度，负载端电流取样电阻选取 3 欧姆。

$$U_{\min}=I_{\min}R=15\mu A$$

$$U_{\max}=I_{\max}R=30mA$$

$$P_{\max}=I^{2}{}_{\max}R=30mV\cdot 10\mathrm{m}A=3\times 10^{-4}W$$

通过上述计算可得，其两端电压范围满足后续计量芯片接口的要求，且取样电阻端功耗很小，几乎可以忽略不计，对后续电气功率测量不会造成干扰。

2.2.2 负载电压取样电阻的计算

交流侧两端 220V 电压通过 1000：1000 的电压互感器传送至数据端，为使 HLW8032 电压有效值输出端为 1：1 输出，且端口差分电压值控制在 490mV 以内，因此电压互感器后级串联 50 欧姆电阻。

3 电路与程序设计

3.1 电路的设计

3.1.1 系统总体框图

运用电流互感器将负载电路中的电流转变为几毫伏的交流电压信号，220V 交流信号经过电压互感器处理后变成几百毫伏的交流信号，然后利用电路将采样得到的信号送入电力 IC 计量芯片 HLW8032，先进行 ADC 转换，再在内置的电能计量内核实现各参数的计算，并将数据存储到寄存器中。数字信号从寄存器中输出时，利用 uart 接口与单片机进行通讯，在单片机的显示屏上显示具体的电流值以及各电气参数。

工作流程图：

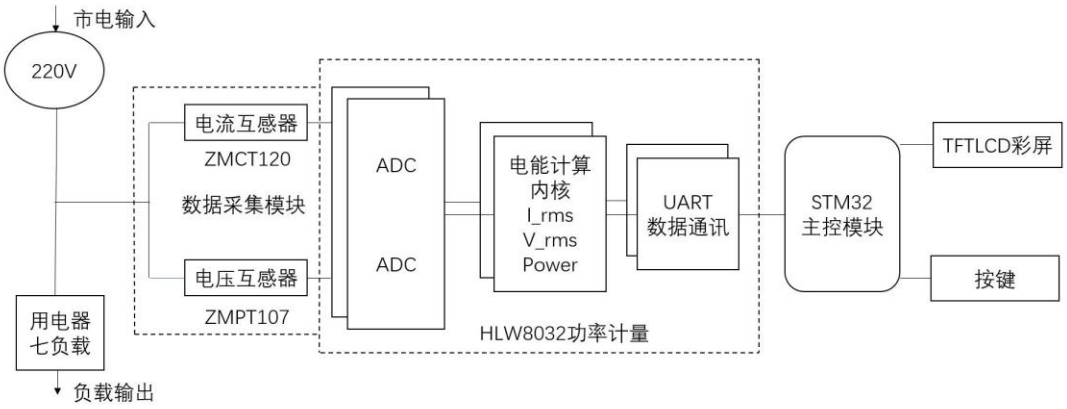

图3 系统框图

3.1.2 隔离采样及处理电路设计

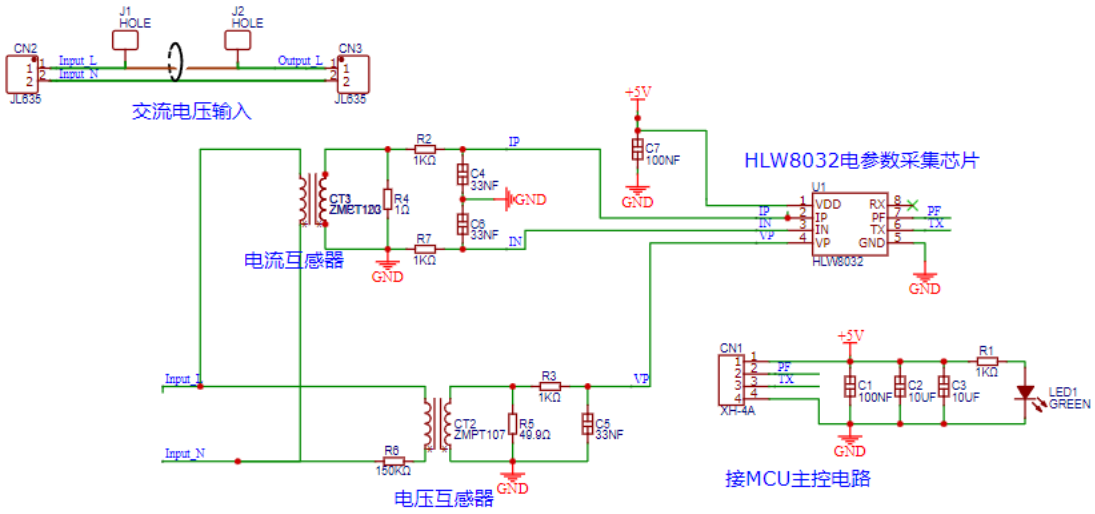

图4 采样和处理电路

电参数采集及处理电路通过互感器实现负载端电流和电压采集，并使用 HLW8032 电能计量模块实现数模(A/D)转换，电流电压有效值，有功功率计算，再通过 MCU 主控电路实现各指标的运算和计量。

3.1.3 电源模块电路设计

市电为 220V 交流电压，而 STM32 微处理器和计量芯片 HLW8032 均为 5V 供电，因此需要设计电源供电电路将 220V 交流电压转换成低压直流供电电压。

选用 HIECUBE 的电源转换模块 AP05N07，将 220V 交流电压转换成 5V 直流电源。HIECUBE 电源转换模块具有非常小的体积，高功率密度，且内置 EMC 电路，具有卓越的纹波性能，满载纹波低至 20mV 以下，内有噪声谐波抑制电路，抗外界干扰能力强。

开关电源模块电路原理图中，R2 为压敏电阻，对输入端的浪涌电压进行防护，R3 为 NTC 热敏电阻，可以减少电源模块在启动过程中的冲击电流，输出端并联；一个大电容以减小输出电压波动。

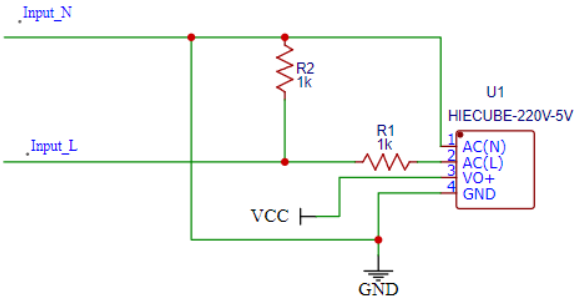

图5 220V 转 5V 电路原理图

3.1.4 主控 STM32 单片机 MCU 电路设计

主控电路包括了 STMF103ZET6 最小系统：复位电路、晶振电路、BOOT 选择电路，通过 STM32 最小系统对信号进行采集和数模转换。FSMC 对外接 TFT 彩屏进行驱动，实时显示电参数值，并在通过算法设计完成自主学习电器特性，主控电路中还加入了独立按键电路，用于数据的存储和显示界面模式的切换。

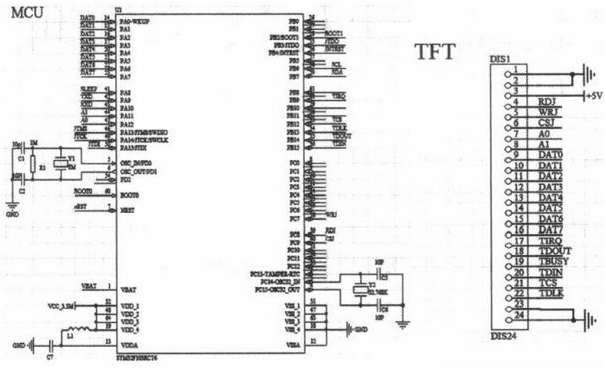

图6 主控单片机 MCU 电路

3.2 程序的设计

3.2.1 程序功能描述与设计思路

软件工作思路：

单片机在通电后，首先进入 1 号状态——“分析”状态，按下按键 1 后进入 2 号状态——“学习”状态，学习和分析状态都要对每一个时刻的采样电流和采样电压进行提取、存储和分析；分析状态只需要对于任意插拔用电器进行判断，所以单片机进行分析的过程较为简单，只需要对用电器进行相关信息的读取并分析其编号；但是，学习环节需要擦除原先保存的产品信息，对每个用电器件的电气参数进行单独的测量并存储，这里需要更加精确的参数处理过程。

按照环节划分，系统程序的编写主要包括：主程序、分析程序、学习程序、特性存储程序、显示程序等。

主程序：设置单片机的全局变量、调用按键功能，通过 uart 通信协议对 HLW8032 模块的数据进行调用，并通过输入显示程序来实现显示的功能。

分析程序：对 HLW8032 数据读取，并将其存储到 STM32 单片机的 Flash 闪存中，利用功率、电流的加权组合的信息进行比较，得到概率最大的用电器件的组合方式。

3.2.2 程序流程图

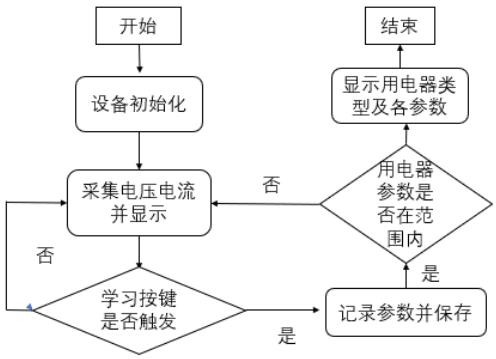

图7 主程序流程图

4 测试结果及分析

4.1 各用电器性能参数测试结果（数据）

图8 测试使用的仪器设备

用电器编号	电器名称	电流有效值	电压有效值	有功功率	功率因数
1	台灯	5mA	227.73V	0.5W	0.46
2	自制电器	5mA	228.11V	1.1W	1.00
3	灯条	61mA	228.02V	7.0W	0.50
4	风扇	28mA	227.98V	2.8W	0.44
5	剃须刀	32mA	231.17V	3.4W	0.50
6	开关	45mA	227.89V	3.7W	0.37
7	热水	8165mA	225.67V	1842W	1.00

4.2 测试分析与结论

测量误差分析：

在测试过程中，单片机显示用电器参数与实际值有细微差别，可能是由于使用了互感器对信号进行隔离采样，会给用电器监测模块的功率因数角引入一定的相角误差；而且不同用电器内阻不同，使得出来的电压电流值也存在细微偏移。

实测监测识别结果：

监测模块在实测中能够对单独运作的七种用电器实现较为精准的识别，并能在显示屏上显示对应电器的电流有效值，电压有效值，有功功率和功率因数等参量，且实际识别时间均小于 2s;

对于多负载同时工作，随意增减用电器，显示屏基本能够准确显示对应的用电器及其工作情况。当最大功率电器热水壶工作时，最小功率电器的增减和工作情况识别准确度相对低一些，但其他情况下均能准确识别并显示，响应时间基本满足响应时间小于 2s.

结论：

说明我们制作的用电器监测装置可以实时准确指示用电器的类别和工作状态，识别精确度和总响应时间能够基本达到设计要求，具有良好的学习识别功能。

5 参考文献

[1] 谭浩强. C 语言程序设计[M]. 北京：清华大学出版社，2012

[2] 黄志伟. 全国大学生电子设计竞赛制作实训（第二版）[M]. 北京：北京航空航天大学出版社，2011

[3] 刘月武，李杏春. 新型单相双向功率/电能集成电路 CS5463 的原理与应用[J]. 疫情仪表用户，2010

[4] 侯殿有，单片机 C 语言程序设计[M]. 北京：人民邮电出版社，2010-11

6 附录

代码 1：

```
#include "sys.h"
#include "delay.h"
#include "usart.h"
#include "led.h"
#include "lcd.h"
#include "usmart.h"
#include "key.h"
#include "exti.h"
#include "adc.h"
#include "bsp_ads1256.h"
#include "ad9850.h"
#include "timer.h"
#include "math.h"
#include "judge.h"
#include "stmflash.h"

u16 USART_RX_STA=0;
  u8 cmd=0;
  u32 t=0;
  u32 Buf[1024]={0};
  u32 Buf_1[1024]={0};
  u32 adcx;
  u32 Mode_use=0;
  u32 Temp=0;
  u32 situat=0;
  uint8_t Tx_Buf_phase[32];
  uint8_t Tx_Buf_magn[32];
  uint8_t Tx_Buf_power[32];
  uint8_t Tx_Buf_Icurr[32];
  uint8_t Tx_Buf_Volt[32];
u8 aRxBuffer[RXBUFFERSIZE];
u8 nam[7];
  u16 nam_dian[7];
u16 datatemp[14];
  u8 i=0;
  float Phase_1=0;
  float Phase_2=0;
  float Phase_dev=0;
```

```
  float magn_1=0;
  float magn_2=0;
  float magn_dev=0;

  float VoltValue = 0.0;
  double IcurrentValue = 0.0;
  float PowerValue = 0.0;
    float Power_1 = 0.0;
       float Power_2 = 0.0;
       float Power = 0.0;

           float Power1=0;
  float Power2=0;
  uint32_t VpR    = 0;
  uint32_t VR     = 0;
  uint32_t CpR    = 0;
  uint32_t CR     = 0;
  uint32_t PpR    = 0;
  uint32_t PR     = 0;
  u32 Meg[1024];
  float p_sta[7];
u8 timeout;

    struct features
{
         u8       sign[8];
      float    Phase[8];
      float    Magn[8];
      float    Effect_pow[8];
}feature;

  u16 TEXT_Buffer[14];
float Power_canshu[7]={0};
#define SIZE sizeof(TEXT_Buffer)
#define FLASH_SAVE_ADDR    0X08070000

int main(void)
{
    u32      dat_sum=0;
       u16 cnt=0;
    float p0=0;
       u16    sigh=0;
```

```
    u8      x=0;
    u8      s=0;
    u8              fak=1;
       float temp;
          u32 adc;
       u8 ch_num;
    u16 i=0;
       u8 f=0;
       HAL_Init();
    Stm32_Clock_Init(RCC_PLL_MUL9);
    delay_init(72);
    uart_init(4800);
uart3_init(4800);
    usmart_dev.init(84);
    LED_Init();
    LCD_Init();
    MY_ADC_Init();
    EXTI_Init();
    KEY_Init();
    TIM3_Init(7200-1,1-1);//72000000/7200/1=10000Hz
      while(1)
    {
          if(USART_RX_BUF[1] == 0x5A)
          {
          memcpy(USART_RX_BUF_3,USART_RX_BUF,24);
          }
          USART_RX_BUF_3[0]=0x55;
          i=0;
          STMFLASH_Read(FLASH_SAVE_ADDR,(u16*)datatemp,SIZE);

          for(i=0;i<7;i++)
          {

          Power_canshu[i]=((float)((u32)datatemp[i]<<16)+(u32)datatemp[i+7])/10000;

          }

          LCD_ShowString(30,390,200,16,24,"Readong learned number:");
          LCD_ShowxNum(200,390,datatemp[0],1,24,0);
          LCD_ShowxNum(230,400,datatemp[1],1,24,0);
          LCD_ShowxNum(200,410,datatemp[2],1,24,0);
          LCD_ShowxNum(230,420,datatemp[3],1,24,0);
          LCD_ShowxNum(200,430,datatemp[4],1,24,0);
          LCD_ShowxNum(230,440,datatemp[5],1,24,0);
```

```
            LCD_ShowxNum(200,450,datatemp[6],1,24,0);

            if(cmd==0)LCD_ShowString(200,190,200,16,24,"learnoff!");
            else LCD_ShowString(200,190,200,16,24,"learnon!");

if(Mode_use==0)
{
        LCD_ShowString(30,10,200,16,24,"Mode 1 Recognize");
        LCD_ShowString(30,190,200,16,24,"NUMBER 1:");
        LCD_ShowString(30,215,200,16,24,"NUMBER 2:");
        LCD_ShowString(30,240,200,16,24,"NUMBER 3:");
        LCD_ShowString(30,265,200,16,24,"NUMBER 4:");
        LCD_ShowString(30,290,200,16,24,"NUMBER 5:");
        LCD_ShowString(30,315,200,16,24,"NUMBER 6:");
        LCD_ShowString(30,340,200,16,24,"NUMBER 7:");
        LCD_ShowxNum(200,195,USART_RX_BUF_3[2],4,16,0);

            if(USART_RX_BUF_3[1] == 0x5A)
            {
                dat_sum=0;
            for(i=2;i<23;i++)
            {
                dat_sum=dat_sum+USART_RX_BUF_3[i];
            }
            {


                    VpR    =    (USART_RX_BUF_3[2]<<16)    +    (USART_RX_BUF_3[3]<<8)    +
USART_RX_BUF_3[4];
                VR     =    (USART_RX_BUF_3[5]<<16)    +    (USART_RX_BUF_3[6]<<8)    +
USART_RX_BUF_3[7];
                VoltValue = (VpR/VR)*1.88 ;


                    CpR    =(USART_RX_BUF_3[8]<<16)    +    (USART_RX_BUF_3[9]<<8)    +
USART_RX_BUF_3[10];
                    CR     =    (USART_RX_BUF_3[11]<<16)    +    (USART_RX_BUF_3[12]<<8)    +
USART_RX_BUF_3[13];
                IcurrentValue = ((float)CpR/(float)CR)*0.25;

                    PpR    =    (USART_RX_BUF_3[14]<<16)    +    (USART_RX_BUF_3[15]<<8)    +
USART_RX_BUF_3[16];
                    PR     =    (USART_RX_BUF_3[17]<<16)    +    (USART_RX_BUF_3[18]<<8)    +
USART_RX_BUF_3[19];
```

```
            PowerValue = ((float)PpR/(float)PR)*1.88*0.25;

      }
  }
Phase_2=Phase_1;
Phase_1=acos(PowerValue/(IcurrentValue*VoltValue));
Phase_dev=fabs(Phase_2-Phase_1);
  if(Phase_dev<=0.01)
  {
if(cnt==65000)cnt=0;
if(cnt==0)Power1=0;
cnt=cnt+1;
Power1=PowerValue+Power1;
Power2=Power1/cnt;
  }
LCD_ShowString(30,85,200,16,24,"AveragePow:");
sprintf ((char *)Tx_Buf_magn,"%3.6f\r\n", Power2);
LCD_ShowString(200,85,200,16,24,(char *)Tx_Buf_magn);
JudgeSta();
  i=0;
  for(i=0;i<7;i++)
  {
  nam_dian[i]=nam[i];
  }
              sprintf ((char *)Tx_Buf_phase,"%3.3f¡ã\r\n", Phase_dev);
              LCD_ShowString(200,60,200,16,24,(char *)Tx_Buf_phase);
              LCD_ShowString(30,115,200,16,24,"PowerValue:");
              sprintf ((char *)Tx_Buf_power,"%3.6f\r\n", PowerValue);
              LCD_ShowString(200,115,200,16,24,(char *)Tx_Buf_power);

              LCD_ShowString(30,140,200,16,24,"IcurrentValue:");
              sprintf ((char *)Tx_Buf_Icurr,"%3.6f\r\n", IcurrentValue);
              LCD_ShowString(200,140,200,16,24,(char *)Tx_Buf_Icurr);

              LCD_ShowString(30,165,200,16,24,"Tx_Buf_Volt:");
              sprintf ((char *)Tx_Buf_Volt,"%3.6f\r\n", VoltValue);
              LCD_ShowString(200,165,200,16,24,(char *)Tx_Buf_Volt);

  LCD_ShowxNum(140,190,nam_dian[0],1,16,0);
              LCD_ShowxNum(140,215,nam_dian[1],1,16,0);
              LCD_ShowxNum(140,240,nam_dian[2],1,16,0);
              LCD_ShowxNum(140,265,nam_dian[3],1,16,0);
              LCD_ShowxNum(140,290,nam_dian[4],1,16,0);
              LCD_ShowxNum(140,315,nam_dian[5],1,16,0);
```

```
                        LCD_ShowxNum(140,340,nam_dian[6],1,16,0);
}

if(Mode_use==1)//Ñ§Ï°
{
        LCD_ShowString(30,10,200,16,24,"Mode 2 Learning ");
      dat_sum=0;
              if(USART_RX_BUF_3[1] == 0x5A)
           {

           for(i=2;i<23;i++)
           {
                dat_sum=dat_sum+USART_RX_BUF_3[i];
           }
           if(dat_sum%256==USART_RX_BUF_3[23]){

                    VpR    =    (USART_RX_BUF_3[2]<<16)    +    (USART_RX_BUF_3[3]<<8)    +
USART_RX_BUF_3[4];
                  VR     =     (USART_RX_BUF_3[5]<<16)     +     (USART_RX_BUF_3[6]<<8)     +
USART_RX_BUF_3[7];
                  VoltValue = (VpR/VR)*1.88 ;
                    CpR     =(USART_RX_BUF_3[8]<<16)     +     (USART_RX_BUF_3[9]<<8)     +
USART_RX_BUF_3[10];
                    CR      =    (USART_RX_BUF_3[11]<<16)    +    (USART_RX_BUF_3[12]<<8)    +
USART_RX_BUF_3[13];
                  IcurrentValue = ((float)CpR/(float)CR)*0.25;
                    PpR    =    (USART_RX_BUF_3[14]<<16)    +    (USART_RX_BUF_3[15]<<8)    +
USART_RX_BUF_3[16];
                    PR      =    (USART_RX_BUF_3[17]<<16)    +    (USART_RX_BUF_3[18]<<8)    +
USART_RX_BUF_3[19];
                    PowerValue = ((float)PpR/(float)PR)*1.88*0.25;
                }
            }

           Phase_2=Phase_1;
           Phase_1=acos(PowerValue/(IcurrentValue*VoltValue));
           Phase_dev=fabs(Phase_2-Phase_1);
            if(Phase_dev<=0.01)
            {
           if(cnt==65000)cnt=0;
           if(cnt==0)Power1=0;
           if(Phase_dev==0){
           cnt=cnt+1;
           Power1=PowerValue+Power1;
```

```
            Power2=Power1/cnt;
            }
             }

            if((situat>0)&&(cmd==1)){
                feature.Effect_pow[situat]=PowerValue;
                feature.Magn          [situat]=magn_dev;
                feature.Phase         [situat]=Phase_dev;
                feature.sign          [situat]=situat;
                memcpy(TEXT_Buffer,datatemp,14);
            TEXT_Buffer[situat-1]=((u32)((feature.Effect_pow[situat]-p0)*10000))>>16;
            TEXT_Buffer[situat-1+7]=((u32)((feature.Effect_pow[situat]-p0)*10000));
            STMFLASH_Write(FLASH_SAVE_ADDR,(u16*)TEXT_Buffer,SIZE);
            }


                LCD_ShowString(30,35,200,16,24,"Label Of Study:");
                LCD_ShowxNum(200,35,situat,3,24,0);

                LCD_ShowString(30,60,200,16,24,"Phase:");
                sprintf ((char *)Tx_Buf_phase,"%3.3f¡ã\r\n", Phase_dev);
                LCD_ShowString(200,60,200,16,24,(char *)Tx_Buf_phase);

                LCD_ShowString(30,85,200,16,24,"AveragePow:");
              sprintf ((char *)Tx_Buf_magn,"%3.6f\r\n", Power2);
              LCD_ShowString(200,85,200,16,24,(char *)Tx_Buf_magn);

                LCD_ShowString(30,115,200,16,24,"PowerValue:");
              sprintf ((char *)Tx_Buf_power,"%3.6f\r\n", PowerValue);
                LCD_ShowString(200,115,200,16,24,(char *)Tx_Buf_power);

              LCD_ShowString(30,140,200,16,24,"IcurrentValue:");
                sprintf ((char *)Tx_Buf_Icurr,"%3.6f\r\n", IcurrentValue);
                LCD_ShowString(200,140,200,16,24,(char *)Tx_Buf_Icurr);

                LCD_ShowString(30,165,200,16,24,"Tx_Buf_Volt:");
                sprintf ((char *)Tx_Buf_Volt,"%3.6f\r\n", VoltValue);
                LCD_ShowString(200,165,200,16,24,(char *)Tx_Buf_Volt);

                LCD_ShowString(30,365,200,16,24,"Be learned number:");
                LCD_ShowxNum(200,365,TEXT_Buffer[situat-1],1,16,0);
            }
        }
        }
```

代码 2：

```
#include "sys.h"
#include "judge.h"
#include "stdlib.h"
extern u8 nam[7];
extern float Power_canshu[7];
float absv[128]={0.0};
extern    float PowerValue;
extern float p_sta[7];
u8 a[30];
u8 h=0;
u8 zhongshu(void);
u8 sigh=0;
void JudgeSta(void)
{

    u8 t=0;
  float minv=10000;
    int ElectricalSta[7]={0};
    int len=7;
    int i=0,j=0;
    float p_sta[7];
    int times=0,timestmp=0;


    for(times=0;times<128;times++)
    {
        timestmp=times;
        for(i=0;i<7;i++){
            ElectricalSta[i]=0;
        }//×´Ì¬ÇåÁã
        i=0;
        while(timestmp)
        {
            ElectricalSta[i]=timestmp%2;
            timestmp=timestmp/2;
            i++;
        }
        for(i=6;i>=0;i--)
```

```
            printf("%d ",a[i]);
            printf("\n");
            */
            for(i = 0; i < len; i ++){
                p_sta[i] = ((float)ElectricalSta[i])*Power_canshu[i];
            }
            for(i=0;i<7;i++) {
                p_sum+=p_sta[i];
            }

            if(((float)p_sum-(float)PowerValue)>0.0){
                absv[times]=(float)p_sum-(float)PowerValue;
            }
            else{
                absv[times]=(float)PowerValue-(float)p_sum;
            }
            p_sum=0;
        }
    minv=10000;
        for(i=0;i<128;i++){
            if(absv[i]<minv)minv=absv[i];
        }

        for(i=0;i<128;i++){
            if(minv==absv[i]){
                t=i;
                a[h]=t;
                h=h+1;
                if(h>=30){
                    h=0;
                    sigh=1;
                }
                else sigh=0;
                break;
            }
        }
        if(sigh==1){
        u8 i,n,c,max=0,j,number;
        n=30;
  for(i=0;i<n;i++)
  {
number=1;
      for(j=0;j<n;j++)
      {
```

```
            if(a[j]==a[i])
            {
            number++;
            }
            if(max<number)
            {
              max=number;
              c=a[i];
            }
      }
  }
       t=c;
       i=0;

       while(t){
             nam[i]=t%2;
             t=t/2;
             i++;
       }
}

u8 zhongshu()
{
  u8 i,n,c,max=0,j,number;

       n=30;
for(i=0;i<n;i++)
  {
number=1;
     for(j=0;j<n;j++)
     {
          if(a[j]==a[i])
          {
          number++;
          }
          if(max<number)
          {
            max=number;
            c=a[i];
          }
     }
  }
  return c;
}
```