[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G01R 11/185 (2006.01)

[12] 发明专利申请公布说明书

[21] 申请号 200810209779.2

[43] 公开日 2009 年 8 月 5 日

[11] 公开号 CN 101498741A

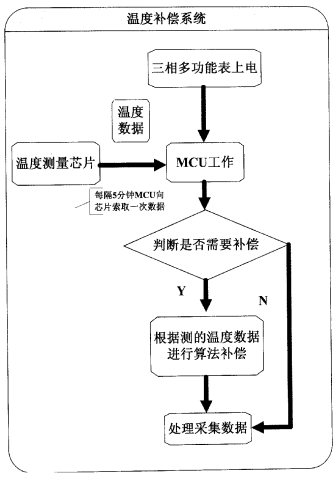
[22] 申请日 2008.12.25
[21] 申请号 200810209779.2
[71] 申请人 哈尔滨电工仪表研究所
地址 150081 黑龙江省哈尔滨市哈平路 128 号
[72] 发明人 于高波
权利要求书 1 页 说明书 9 页 附图 4 页

[54] 发明名称

一种多功能电能表的温度补偿方法

[57] 摘要

一种多功能电能表的温度补偿方法，本发明涉及一种电能信息监测装置的温度补偿方法，它克服了现有技术中只对多功能电能表局部进行温度补偿的缺陷，全面解决了多功能电能表温度补偿的问题，提高了电能表的精度。 所述的温度数据采集电路由数字式温度测量芯片、处理器 MCU 和外围辅助电路组成。 温度测量芯片将采集到的温度数据，根据 1 – Wire 总线协议把数字化的温度数据通过 I/O 口传给处理器 MCU，处理器 MCU 应用温度补偿算法对多功能电能表测量的电力数据进行针对性的补偿。 本发明能够保证多功能电能表在恶劣温度条件下的测量精度，具有电路设计简洁、温度补偿全面的特点。



1、一种多功能电能表的温度补偿方法，其特征在于所述的温度数据采集电路由数字式温度测量芯片、处理器 MCU 和外围辅助电路组成；温度测量芯片将采集到的温度数据，根据 1-Wire 总线协议把数字化的温度数据通过 I/O 口传给处理器 MCU，处理器 MCU 应用温度补偿算法对多功能电能表测量的电力数据进行针对性的补偿；

2、根据权利要求 1 所述的一种多功能电能表的温度补偿方法，其特征在于所述的温度测量芯片采用数字温度测量芯片 DS18B20；

3、根据权利要求 1 所述的一种多功能电能表的温度补偿方法，其特征在于处理器 MCU 采用 PIC18F6621；

4、根据权利要求 1 所述的一种多功能电能表的温度补偿方法，其特征在于所述温度补偿程序采用 MPLAB IDE v8.15 编译。

一种多功能电能表的温度补偿方法

技术领域

本发明涉及一种电能信息监测装置，特别涉及一种多功能电能表的温度补偿方法。

背景技术

电能表精度的高低直接关系到用户的切身利益,尤其是用电量大的关键部门。现有的高精度多功能电度表，在恶劣的温度条件下，尤其是在北方寒冷的天气环境下，组成电能表的各个功能块，甚至元器件都受温度变化的影响，对电能表的测量精度产生不同程度的影响。专利 92203323.4 提出的互感器钕铁硼阻尼体温度补偿法仅是对互感器进行了温度补偿，而对电能表的其他部件没有提出温度补偿的解决方法，不能保证电能表整体的测量精度；采用热敏电阻对晶振进行温度补偿的方法只是保证了晶振的精度，对于电能表的信号采集、信号传输、运算处理等其他功能块的温度补偿没有进行考虑。所以上述方法都不能全面解决温度变化对多功能电能表测量精度的影响，本发明为了克服上述问题，为了从多功能电能表的整体角度解决多功能电能表的温度补偿问题，提供了一种多功能电能表的温度补偿方法，从而提高了多功能电能表在温度影响下的测量精度，满足了用户的需求。

发明内容

本发明为了克服现有技术的不足之处，为了全面保证多功能电能表的测量精度，提供了一种多功能电能表的温度补偿方法。所述的温度数据采集电路由数字式温度测量芯片、处理器 MCU 和外围辅助电路组成。温度测量芯片将采集到的温度数据，根据 1-Wire 总线协议把数字化的温度数据通过 I/O 口传给处理器 MCU,处理器 MCU 应用温度补偿算法对多功能电能表测量的电力数据进行针对性的补偿。

所述的温度测量芯片采用数字温度测量芯片 DS18B20，所述的处理器 MCU 采用 PIC18F6621，所述的温度补偿算法采用 MPLAB8.0 的 C 语言环境编译。

与现有技术相比，本发明的有益效果是采用 1-Wire 总线式芯片 DS18B20 更节省 MCU 资源；仅用一个芯片就实现了数字化的温度采集，电子线路更简洁可靠，算法从整体性考虑更加全面完善。

附图说明

图 1 是系统整体原理框图；图 2 是计量功能块原理图；图 3 是前端信号调

理功能块原理图；图 4 是 MCU 和 DS18B20 原理图。

具体实施方式

参照附图，本具体实施方式如下：

所述的温度数据采集电路由数字式温度测量芯片、处理器 MCU 和外围辅助电路组成。温度测量芯片将采集到的温度数据，根据 1-Wire 总线协议把数字化的温度数据通过 I/O 口传给处理器 MCU，处理器 MCU 应用温度补偿算法对多功能电能表测量的电力数据进行针对性的补偿。

所述的温度测量芯片采用数字温度测量芯片 DS18B20，所述的处理器 MCU 采用 PIC18F6621，所述的温度补偿算法采用 MPLAB8.0 的 C 语言环境编译。DS18B20 是单总线数字温度计，只需占用单片机的一个 I/O 口位，其内部自带 A/D 转换器，通过内部的温度采集、A/D 转换等一系列过程，将温度数据以 1-Wire 总线协议的规定格式转换并输出，处理器 MCU 可将该数据还原为温度值，其分辨率可以达到 12 位；数字温度测量芯片 DS18B20 的测温范围为－55℃～＋125℃，固有测温分辨率为 0.5℃。

多功能电能表工作时，因为有表壳保护，温度变化波动不是太快，所以为了节省 MCU 资源，每隔五分钟向温度芯片索取一次温度数据即可。

如图 1 所示，多功能电能表温度补偿的工作流程如下：

多功能电能表上电工作后，处理器 MCU 每隔五分钟向温度芯片索取一次温度数据，数字温度测量芯片 DS18B20 通过 I/O 口根据 1-Wire 总线协议把数字化的温度数据传给处理器 MCU，MCU 判断温度数据，进行补偿数据的选择。

如图 2、图 3 所示，电力信号经前端信号处理采集电路的调制后，传递给以 ATT7022B 为核心的电力计量电路，经计量芯片处理，转换为数字信号，等待 MCU 索取数据。

如图 4 所示，数字温度测量芯片 DS18B20 采用了外部供电模式，外部供电模式就是在数字温度测量芯片 DS18B20 的电源引脚 VDD 上外接一个 5V 电源，且 I/O 口线上不再需要接一个上拉的 MOSFET，进行温度转换时 MCU 不再需要将 DS18B20 的 DQ 引脚上拉到高电平，MCU 通过第 64 脚 I/O 口 RE2 向温度测量芯片 DS18B20 发送请求，DS18B20 接收到请求，向 MCU 发送温度数据。

MCU 根据接收到的系统温度数据，进行判断，是否进行温度补偿，如果需要就根据温度的具体数值，用温度补偿算法来补偿采集的电量数据；如不需要，就按原有的计量算法，该计算是由处理器 MCU 进行计算的。

数字温度测量芯片 DS18B20 内部有 8 个字节的暂存存储器，其中头两个字

节表示测得的温度读数，数据格式如下表所示：

s	s	s	s	s	2^6	2^5	2^4

a 温度寄存器高 8 位

2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}

b 温度寄存器低 8 位

当测得温度为正时，s=0；当测得温度为负时，s=1；其余低位以二进制补码形式表示。

当 DS18B20 配置为 12 位分辨率时，温度寄存器中的所有位都是有效数据，能分辨的最小温度值为 0.0625℃；当 DS18B20 配置为 n 位分辨率时，温度寄存器中的位 0 无效，能分辨的最小温度值为 0.125℃；当 DS18B20 配置为 10 位分辨率时，温度寄存器中的位 0 和位 1 无效，能分辨的最小温度值为 0.25℃；当 Ds18B20 配置为 9 位分辨率时，温度寄存器中的位 0、位 1 和位 2 无效，能分辨的最小温度值为 0.5℃。表 1 给出了 DS18B20 在 12 位分辨率时温度/数据的对应关系。

表 1 DS18B20 温度/数据的对应关系

温度(℃)	输出的二进制码	对应的十六进制码
+125.00	0000 0111 1101 0000	07D0H
+85.00	0000 0101 0101 0000	0550H
+25.06	0000 0001 1001 0001	0191H
+10.13	0000 0000 1010 0010	00A2H
+0.50	0000 0000 0000 1000	0008H
0.00	0000 0000 0000 0000	0000H
-0.50	1111 1111 1111 1000	FFF8H
-10.13	1111 1111 0101 1110	FF5EH
-25.06	1111 1110 0110 1111	FF6FH
-55.00	1111 1100 1001 0000	FC90H

从表 1 中可以看出，根据 DS18B20 输出的二进制码及其配置的分辨率，经过简单的计算就能得到测量的温度值。

程序部分：

```
/* Copyright (c) 2008,Harbin Research Institute of Electrical Instruments
* All rights Reserved.
* File Name: temperature_compensation.c
```

```
* Dependencies: define.h/math.h/pic18.h
* Processor: PIC18F6621
* File ID: HBSsoftware081101
* Date:   01\11\2008
* Abstract: temperature compensation of the Multi_Fuction Meter
* Current Version: v1.0
* Author: YuGaobo*/
      #include <pic18.h>
      #include <math.h>
      #include <define.h>
      #include <absacc.h>
      #define uchar unsigned char
      #define uint    unsigned int
      #define SETTEMP   P1                    //定义 P1 口为设定温度
      #define SELECT    P2                    //定义 P2 口为选择信号
      sbit RE2 = P3^4;                        //定义 ds18b20 通信端口
      int temp1,temp2,ID=0,set=0,blink;
      int temp1=27,xs=5;                      //定义整数和小数
      unsigned int max,mid,min,flag;
      /* -50° C~50° C，4° C/each time*/
      /*D_Ua,D_Ub,D_Uc,D_Ia,D_Ib,D_Ic,D_Uab,D_Ubc,D_Uca,D_In,D_F,D_PHa,D_PHb,D_PHc,D_PH,D_
        Pa,D_Pb,D_Pc,D_P,D_Qa,D_Qb,D_Qc,D_Q*/
const unsigned char Comp_tab[25][23]={
0x25,0x24,0x25,0x23,0x23,0x24,0x22,0x22,0x21,0x23,0x23,0x22,0x22,0x21,0x23,0x23,0x22,0x22,0x21,0x23,0x
23,0x22,0x22,//-50° C
0x20,0x1f,0x1f,0x1e,0x20,0x22,0x21,0x22,0x22,0x21,0x23,0x1f,0x1e,0x20,0x22,0x1f,0x1e,0x20,0x22,0x21,0x2
2,0x21,0x23,//-46° C
.....
0x08,0x09,0x09,0x08,0x09,0x10,0x08,0x08,0x09,0x09,0x08,0x09,0x08,0x09,0x09,0x08,0x09,0x08,0x09,0x09,0x
08,0x09,0x10,//46° C
0x11,0x12,0x11,0x13,0x13,0x14,0x11,0x12,0x11,0x13,0x11,0x12,0x11,0x13,0x13,0x14,0x11,0x12,0x11,0x13,0x
13,0x14,0x11,//50° C
    }; //根据该类多功能电能表试验所得补偿数据
```

```
    void ReadTemperature(void);                    //读取 ds18b20 当前温度
    int temperature_compensation(unsigned long Temper);    //温度数据处理子函数
    void Init_DS18B20(void);                       //ds18b20 初始化函数
    void delay_18B20(unsigned int i);              //ds18b20 延迟子函数
    unsigned char ReadOneChar(void);               //ds18b20 读一个字节
    void WriteOneChar(uchar dat);                  //ds18b20 写一个字节
    void delay_18B20(unsigned int i);
    Ua=Ua*Comp_tab[TC][0];
     ……
     Q=Q*Comp_tab[TC][22];                         //各项测量数据的温度补偿
    int temperature_compensation(unsigned long Temper)
                                                   // 温度补偿的子函数，需要输入无符号长整型的温度数据
int main(void)
    {
    unsigned long Temper = 0;
    max=30;                                        //上下限初始化
    min=20;
    while(1)
        {    mid=(max+min)/2;
             motor();
             key();
             if(ID==0)
             displaytemp();
         }
    temperature_compensation（）;
    Ua=Ua*Comp_tab[TC][0];
    Ub=Ub*Comp_tab[TC][1];
    Uc=Uc*Comp_tab[TC][2];
    Ia=Ia*Comp_tab[TC][3];
    Ib=Ib*Comp_tab[TC][4];
    Ic=Ic*Comp_tab[TC][5];
    Uab=Uab*Comp_tab[TC][6];
    Ubc=Ubc*Comp_tab[TC][7];
```

```
    Uca=Uca*Comp_tab[TC][8];
    In=In*Comp_tab[TC][9];
    F=F*Comp_tab[TC][10];
    PHa=PHa*Comp_tab[TC][11];
    PHb=PHb*Comp_tab[TC][12];
    PHc=PHc*Comp_tab[TC][13];
    PH=PH*Comp_tab[TC][14];
    Pa=Pa*Comp_tab[TC][15];
    Pb=Pb*Comp_tab[TC][16];
    Pc=Pc*Comp_tab[TC][17];
    P=P*Comp_tab[TC][18];
    Qa=Qa*Comp_tab[TC][19];
    Qb=Qb*Comp_tab[TC][20];
    Qc=Qc*Comp_tab[TC][21];
    Q=Q*Comp_tab[TC][22];
    .....
    }
int temperature_compensation(unsigned long Temper)
      {
        unsigned int TC=0;                  //定义整数型参数 TC=0
        Temper=Temper+50;                   //温度加 50，因为温度测量范围为-50° C 至 50° C，而温度
                                            补偿数据二维数组的组标号最低是从零开始，所以为了对
                                            照补偿数组，把温度提为正值。
        Temper=Temper/4;                    //温度补偿每 4° C 一组补偿数据
        TC=ceil(Temper);                    //作为标号进行取整运算
        return TC;                          //TC 数值返回主函数
      }
    /*********延时>K*1ms,*，*12.000mhz>11.0596 有误差********/
void delayms(int ms)
      {
        uchar i;
        while(ms--)
        {
```

```
            for(i=250;i>0;i--);
        }
    }
    /**********ds18b20 延迟子函数（晶振 11.0596MHz ）*******/
void delay_18B20(unsigned int i)
    {
      while(i--);
    }
    /**********ds18b20 初始化函数**********************/
void Init_DS18B20(void)
    {
      unsigned char x=0;
      DQ = 1;                          //DQ 复位 ds18b20 通信端口
      delay_18B20(8);                  //稍做延时
      DQ = 0;                          //MCU 将 DQ 拉低
      delay_18B20(80);                 //精确延时 大于 480us
      DQ = 1;                          //拉高总线
      delay_18B20(4);
      x=DQ;                            //稍做延时后 如果 x=0 则初始化成功 x=1 则初始化失败
      delay_18B20(20);
    }
    /**********ds18b20 读一个字节**************/
unsigned char ReadOneChar(void)
    {
      uchar i=0;
      uchar dat = 0;
      for (i=8;i>0;i--)
          {
            DQ = 0;                    // 高电平拉成低电平时读周期开始
            dat>>=1;
            DQ = 1;                    // 给脉冲信号
            if(DQ)
            dat|=0x80;
```

```
            delay_18B20(4);
          }
    return(dat);
}
/*************ds18b20 写一个字节****************/
void WriteOneChar(uchar dat)
{
    unsigned char i=0;
    for (i=8; i>0; i--)
      {
        DQ = 0;                    //从高电平拉至低电平时,写周期的开始
        DQ = dat&0x01;             //数据的最低位先写入
        delay_18B20(5);            //60us 到 120us 延时
        DQ = 1;
        dat>>=1;                   //从最低位到最高位传入
      }
}
/**************读取 ds18b20 当前温度************/
void ReadTemperature(void)
{
unsigned char a=0;
unsigned char b=0;
unsigned char t=0;
Init_DS18B20();
WriteOneChar(0xCC);                // 跳过读序号列号的操作
WriteOneChar(0x44);                // 启动温度转换
delay_18B20(100);                  // this message is wery important
Init_DS18B20();
WriteOneChar(0xCC);                //跳过读序号列号的操作
WriteOneChar(0xBE);                //读取温度寄存器等（共可读 9 个寄存器） 前两个就是温度
delay_18B20(100);
a=ReadOneChar();                   //读取温度值低位
b=ReadOneChar();                   //读取温度值高位
```

```
temp1=b<<4;                    //高 8 位中后三位数的值
temp1+=(a&0xf0)>>4;            //低 8 位中的高 4 位值加上高 8 位中后三位数的值  temp1
                               是温整数值
temp2=a&0x0f;                  //小数的值
}
```

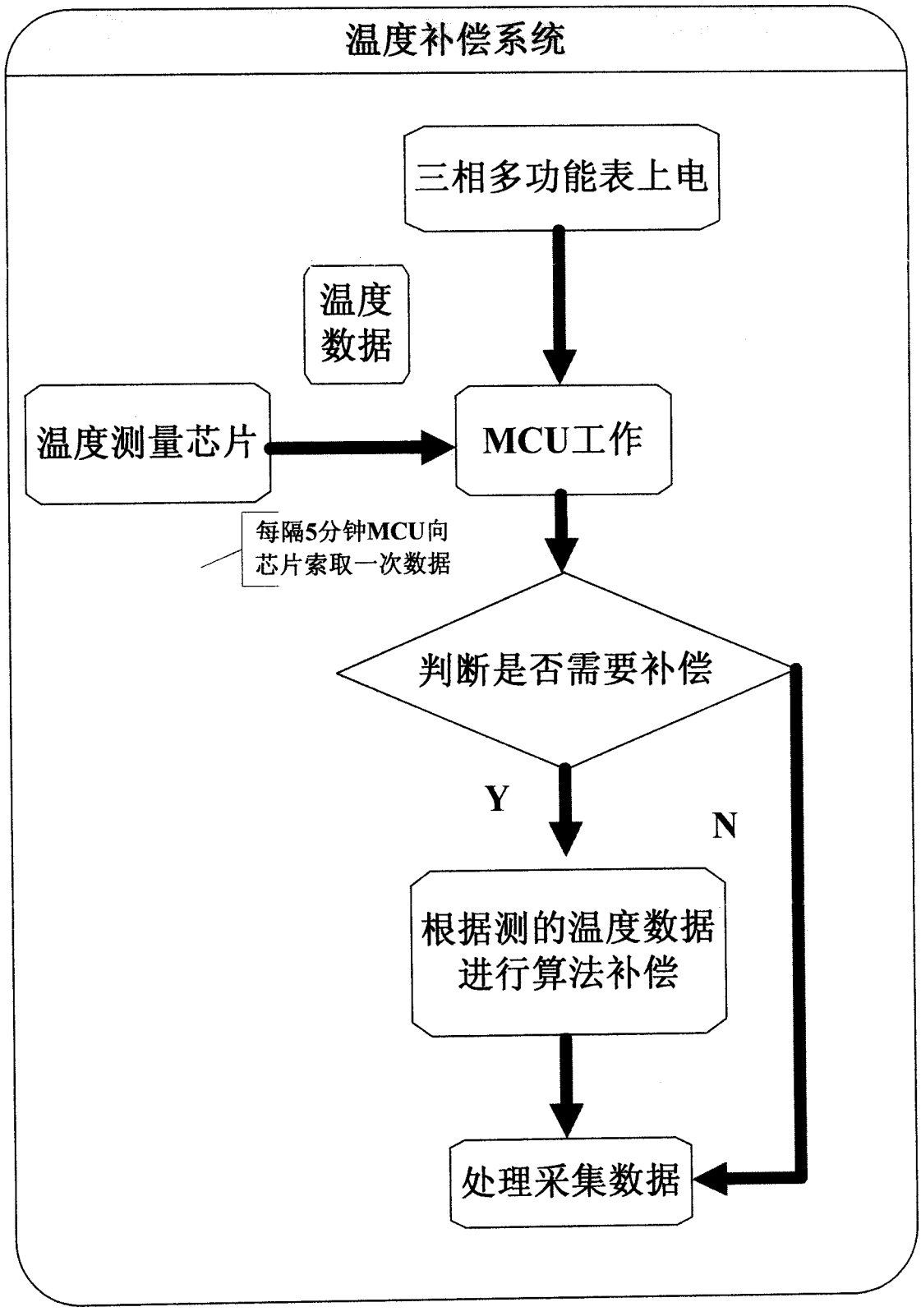

温度补偿系统
三相多功能表上电
温度
数据
温度测量芯片
MCU工作
每隔5分钟MCU向
芯片索取一次数据
判断是否需要补偿
Y
N
根据测的温度数据
进行算法补偿
处理采集数据

图 1

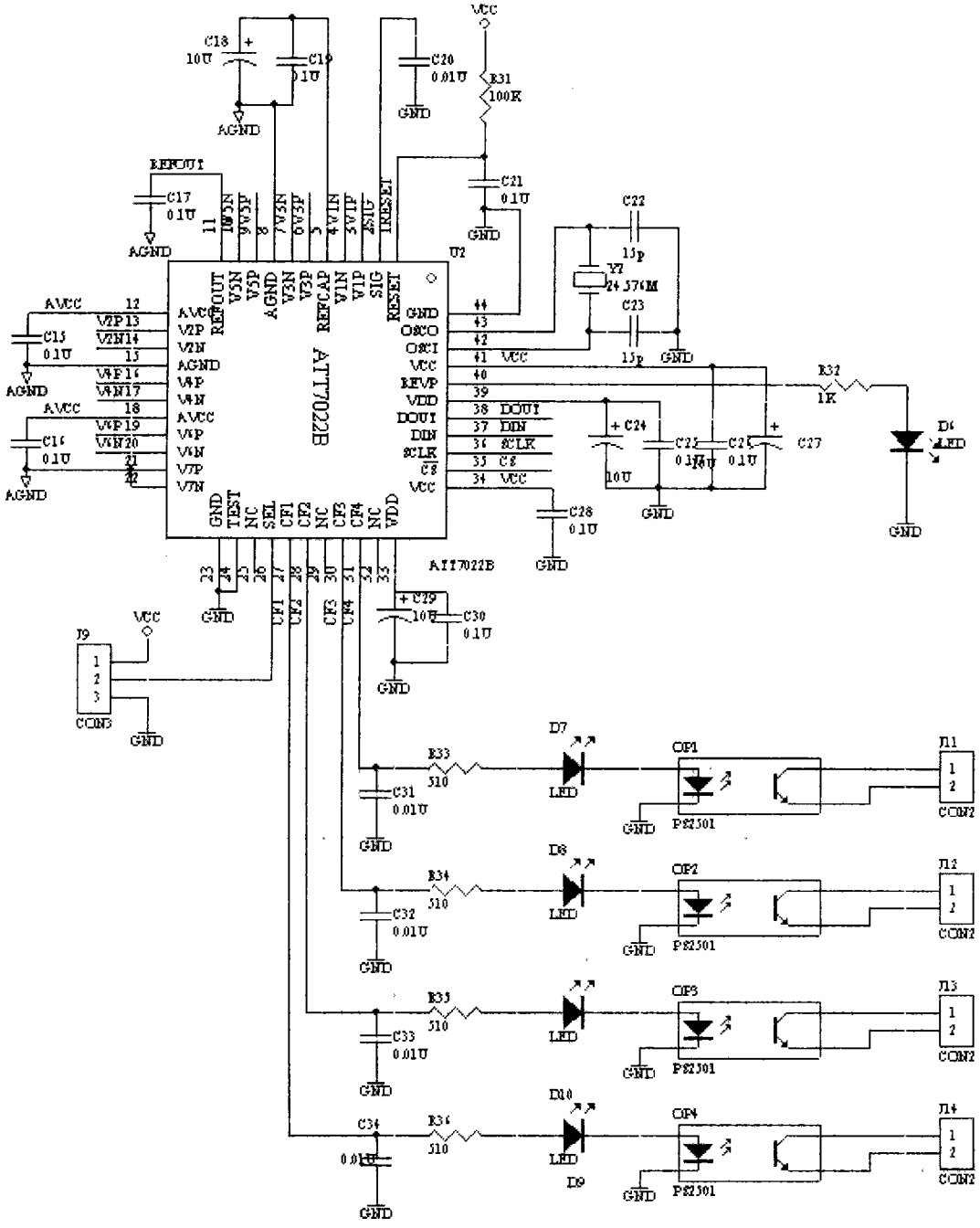

VCC
C18 10U
C19 0.1U
C20 0.01U
R31 100K
AGND
GND
REFOUT
C17 0.1U
C21 0.1U
C22 15p
Y1 24.576M
C23 15p
U2
AIT7022B
AVCC
C15 0.1U
C16 0.1U
R32 1K
D6 LED
C24 10U
C25 0.1U
C26 0.1U
C27
C28 0.1U
C29 10U
C30 0.1U
J9
CON3
D7
D8
D9
D10
R33 510
R34 510
R35 510
R36 510
C31 0.01U
C32 0.01U
C33 0.01U
C34 0.01U
OP1
OP2
OP3
OP4
PS2501
J11
J12
J13
J14
CON2

图 2

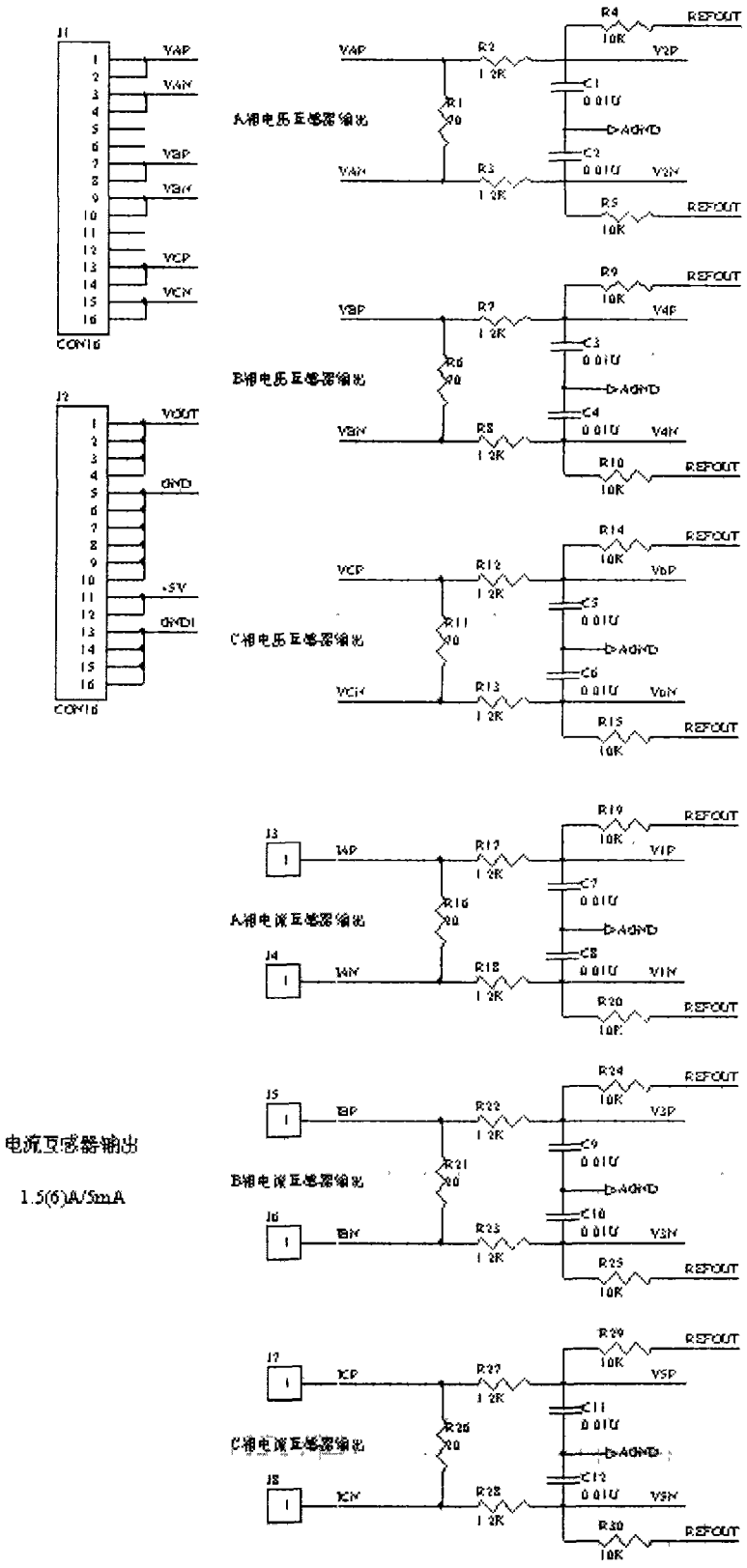

J1
CON16
VAP
VAN
VBP
VBN
VCP
VCN
J2
CON16
VOUT
GND
+5V
GND1
A相电压互感器输出
B相电压互感器输出
C相电压互感器输出
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14
R15
10R
1.2R
20
C1
C2
C3
C4
C5
C6
0.01U
AGND
REFOUT
V2P
V2N
V4P
V4N
VbP
VbN
J3
J4
J5
J6
J7
J8
IAP
IAN
IBP
IBN
ICP
ICN
A相电流互感器输出
B相电流互感器输出
C相电流互感器输出
R16
R17
R18
R19
R20
R21
R22
R23
R24
R25
R26
R27
R28
R29
R30
C7
C8
C9
C10
C11
C12
VIP
VIN
V3P
V3N
VSP
VSN
电流互感器输出
1.5(6)A/5mA

图 3

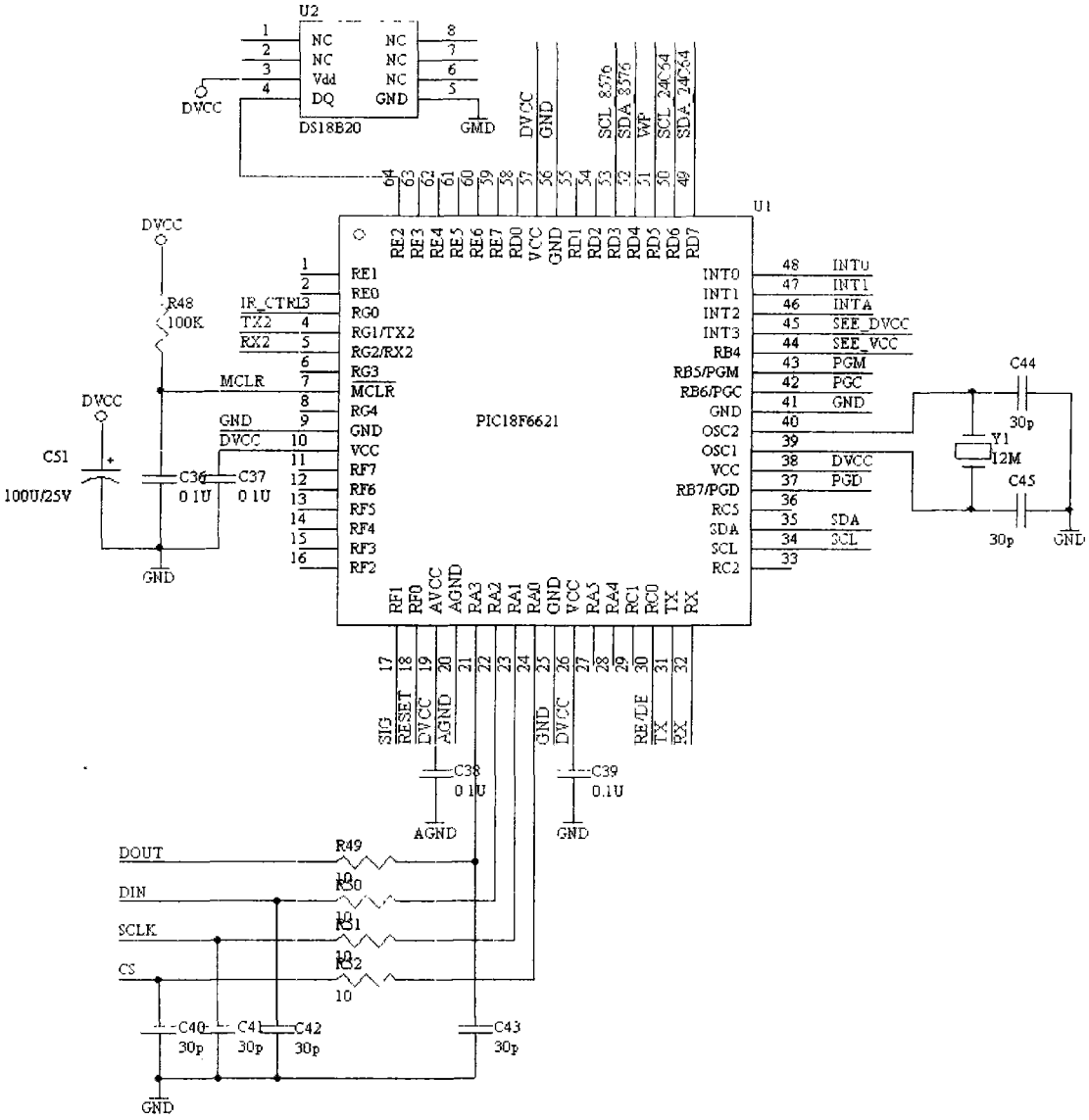

U2
NC
NC
Vdd
DQ
NC
NC
NC
GND
DS18B20
DVCC
GMD
U1
PIC18F6621
R48
100K
IR_CTRL
TX2
RX2
MCLR
C51
100U/25V
C36
0.1U
C37
0.1U
C44
30p
Y1
12M
C45
30p
C38
0.1U
C39
0.1U
AGND
GND
DOUT
DIN
SCLK
CS
R49
R50
R51
R52
10
C40
30p
C41
30p
C42
30p
C43
30p

图 4