试验项目技术报告-热电偶温度测控系统

试验内容

- 热电偶测温原理，冷端补偿；
- DS18B20 单总线单片机数据传输；
- 仪表放大器的运用；
- AD 转换器的运用；
- VC 上位机虚拟仪表的设计，数据的显示；

试验结果

上位机基于 VC 软件的操作界面，对电加热炉的当前温度进行测量和显示。

试验所需元器件

热电偶（1 只），LM324 放大器（1 只），AD620 仪表放大器（1 只），AD0804 转换器（1 只），串口套件 1 套

1. 试验整体结构图

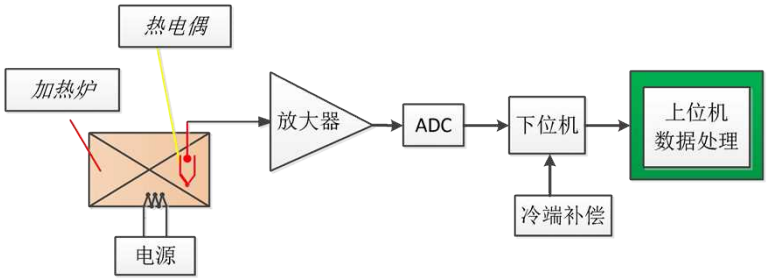


图 1 炉温测控系统整体结构图

试验中运用热电偶传感器将加热炉内温度与室温的温差转换为毫伏级电压，经仪表放大器（AD620）放大至标准电压输出信号（0~5V），送至 AD 模数转换器进行转换。同时，利用 DS18B20 测温芯片测量室温，对热电偶冷端进行补偿。

2. 单片机基础学问

本试验主要涉及运用单片机定时器定时串口写及波特率设置，DA 和 AD 转换器的限制，单片机串口读上位机的数据，C51 单片机的引脚功能，如图 2 所示。

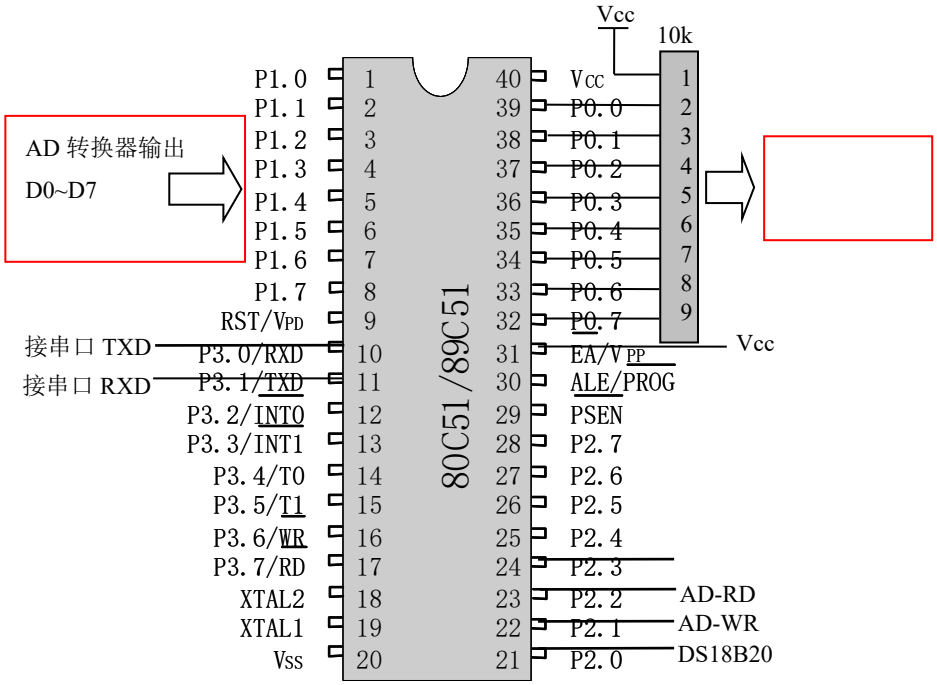


图 2 单片机管脚分布图

2.1 与试验有关的管脚

- 试验中 P3 口中两引脚其次功能：P3.0：RXD 串行口输入接串口模块发送端 TXD；P3.1：TXD 串行口输出接串口模块 RXD 端。P2.0 脚接温度传感器 DS18B20 数据传输总线，转换器写限制 P2.1 脚接 AD 转换器写限制，P2.2 脚接 AD 转换器读限制。
- 管脚 18，和管脚 19（XTAL1，XTAL2）：外部晶振的输入端；
- 管脚 20（V_{SS}）：接地端
- 管脚 31（EA/Vpp）：访问片内允许端，接 VCC；
- 管脚 40（电源）：+5V

2.2 单片机能够运行起来的最小系统

1. 电源 2. 晶振 3. 复位电路

2.3 关于电平特性

单片机为 TTL 电平：高 +5V，低 0V。

计算机的串口为 RS232 电平：高 -12V，低+12V。因此，计算机与单片机之间通讯时须要加电平转换芯片 max232，连接图如图 5 所示，试验中我们运用 232 电平转 TTL 电平串口通讯模块来实现。

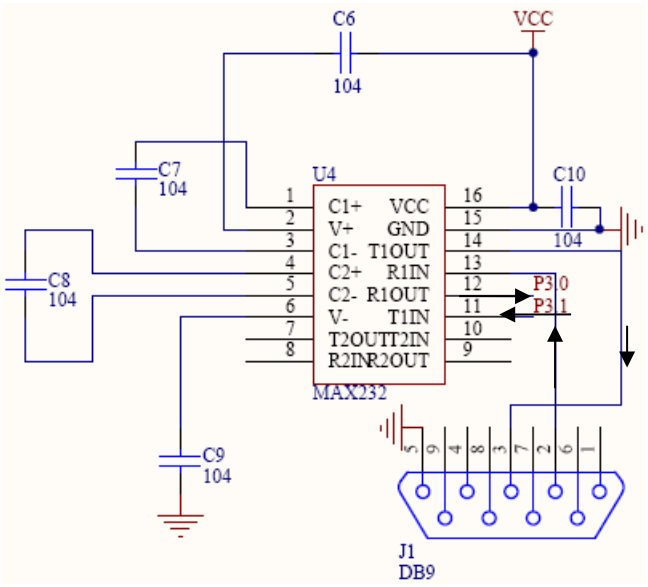


图 5. MAX232 串口连接图

2.3 定时器

- 与定时器有关的寄存器是 8 位 TCON 寄存器，地址是 0x88（注：地址可以被 8 整除，可以进行位操作），其各位的功能如下：

  sfr TCON = 0x88;

  sbit TR1 = TCON^6; // 启动定时器 1；

  sbit TR0 = TCON^4; //启动定时器 0；

  本试验我们只运用 TR1=1 对定时器进行启动，对波特率进行设置。

- 定时/计数器的工作方式寄存器 TMOD，地址为 0x89。低四位用于定时器/计数器 T0，高四位用于 T1。其格式如下。

位	7	6	5	4	3	2	1	0	
0x89	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	TMOD

分别为：门控位 GATE，低电平有效；C/T= 0/1：定时/计数模式选择位；M1M0：工作方式设置位。

M1M0	工作方式	说明
00	方式 0	13 位定时/计数器
01	**方式 1**	**16 位定时/计数器**
10	**方式 2**	**8 位自动重装定时/计数器**
11	方式 3	T0 分成两个独立的 8 位定时/计数器；T1 此方式停止计数

试验中，我们运用定时器 1 的自动重装模式设置数据传输波特率，即 TMOD=0x20。

T1 计数器寄存器：

sfr **TL1** = 0x8B; //计数器 1 低 8 位

sfr **TH1**　　= 0x8D; //计数器 1 高 8 位

2.4 串口读写

80C51 单片机是全双工的串行异步传输，80C51 串行口有两个物理上独立的接收、发送缓冲器 SBUF，它们占用同一地址 99H；接收器是双缓冲结构，只需读写缓冲器 SBUF 便可完成数据的接收和发送，由于发送时 CPU 是主动的，因此不会产生重叠错误。与串口相关的限制寄存器是 SCON，地址为 98H，其各位的功能如下：

位	7	6	5	4	3	2	1	0	
0x98	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	TCON

其中，

- SM0 和 SM1 为工作方式选择位，可选择四种工作方式；
- SM2：多机通信限制位，方式 1（10 位）置 0；
- **REN，允许串行接收位**。由软件置 REN=1，则启动串行口接收数据；若软件置 REN=0，则禁止接收；
- TB8，在方式 2 或方式 3 中，是发送数据的第九位，可以用软件规定其作用；
- RB8，在方式 1 时，若 SM2=0，则 RB8 是**接收到的停止位**；
- TI，发送中断标记位。在方式 0 时，当串行发送第 8 位数据结束时，串行**发送停止位的起先**时，由**内部硬件使 TI 置 1**，向 CPU 发中断申请（**中断号 4，要开相应的中断即 ES=1**）。在中断服务程序中，必需用软件将其清 0，取消此中断申请；
- **RI，接收中断标记位**。在方式 0 时，当串行接收第 8 位数据结束时，串行接收停止位的中间时，由内部硬件使 RI 置 1，向 CPU 发中断申请（中断号 4）。也必需在中断服务程序中，用软件将其清 0，取消此中断申请。

串行口的工作方式

SM0	SM1	方　式	说　　明	波特率
0	0	0	移位寄存器	$f_{osc}/12$
0	1	1	10 位异步收发器（8 位数据）	可变
1	0	2	11 位异步收发器（9 位数据）	$f_{osc}/64$ 或 $f_{osc}/32$
1	1	3	11 位异步收发器（9 位数据）	可变

a．本试验中我们运用方式 1，10 位异步收发器（8 位数据，1 位起始位，1 位停止位），发送中断标记位 TI 用以检测数据是否发送完毕，接收中断标记位 RI 用以检测数据是否接收完毕。

b．波特率的设置

在串行通信中，收发双方对发送或接收数据的速率要有约定（一般设置收发的波特率相同）。通过软件可对单片机串行口编程为四种工作方式，其中方式 0 和方式 2 的波特率是固定的，而方式 1 和方式 3 的波特率是可变的，由定时器 T1 的溢出（频）率来确定。

方式 1 的波特率 =（$2^{SMOD}/32$）•（T1 溢出率）；//SMOD 电源寄存器 PCON 的第 7 位；0x80

当 T1 作为波特率发生器时，其工作在自动重装 8 位定时器方式（即方式 2），溢出率取决于 TH1 中的计数值。

T1 溢出率 = fosc /{12×[256 －（TH1）]}

fosc 为单片机的时钟频率。计数器 1 的低 8 位 TL1 计数满时，发送中断 TR1 后，将自

动将 TH1 的数据装入到 TL0 中，重新进行计数，实现高精度定时。单片机可以通过 T1 的软件编程来适应发送数据的波特率的要求，例如本试验中设置波特率为 9600，**SMOD=1，**时钟频率 **fosc = 11.0592MHZ，则 TH1=0xfd，TL1=0xfd。**

2.5 IE 中断允许限制寄存器

试验中，须要处理数据串行通讯中断信息，因此要了解中断允许限制寄存器的设置，中断允许限制寄存器 IE 的各位的作用如下：

sfr IE = 0xA8; //可以被 8 整除，可以进行位操作

sbit EA = IE^7;// 中断允许总限制器 = 1（最高级权限允许）

sbit ET2 = IE^5;

sbit ES = IE^4;// 串行口中断允许位（本级权限允许）

sbit ET1 = IE^3;//开放定时器 1 中断（本级权限允许）

sbit EX1 = IE^2; //开外部中断 1（本级权限允许）

sbit ET0 = IE^1;//开放定时器 0 中断（本级权限允许）

sbit EX0 = IE^0;//开外部中断 0（本级权限允许）

在本试验中，

设置 EA=1;

ES=1;（开串口中断允许，允许接收串口数据接收/发送完毕产生的中断 RI 或 TI）。

ET0=1；定时器 0 中断，定时采样 AD 转换器转换的数据。

3. 硬件电路

3.1 8 位 A/D 模/数转换器 ADC0804

ADC0804 连接电路如图 6 所示。其中， 为片选信号，低电平有效，试验中接地处理；为数据转换完成标记位（悬空不用）；

读数字输出限制，低电平有效，试验中写模拟量输入限制，上升沿有效，接单片机 P2.1，AD 读限制接单片机 P2.2。输入模拟电压有 VIN(+)和 VIN(-)输入。读写时序如图 7、8 所示。

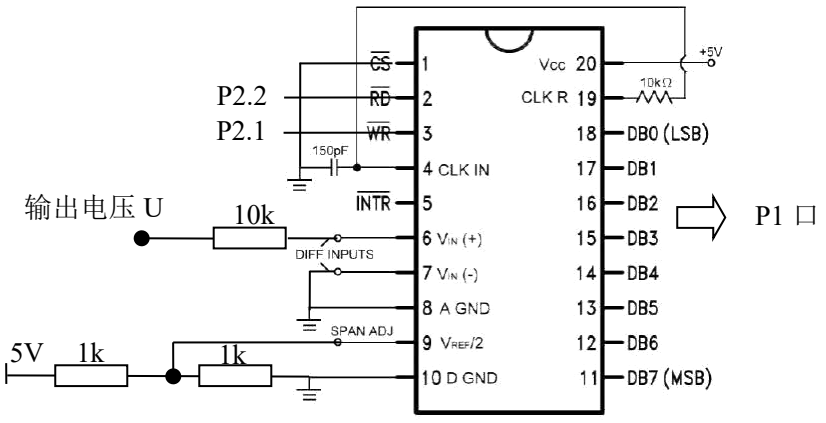


图 6. ADC0804 连接

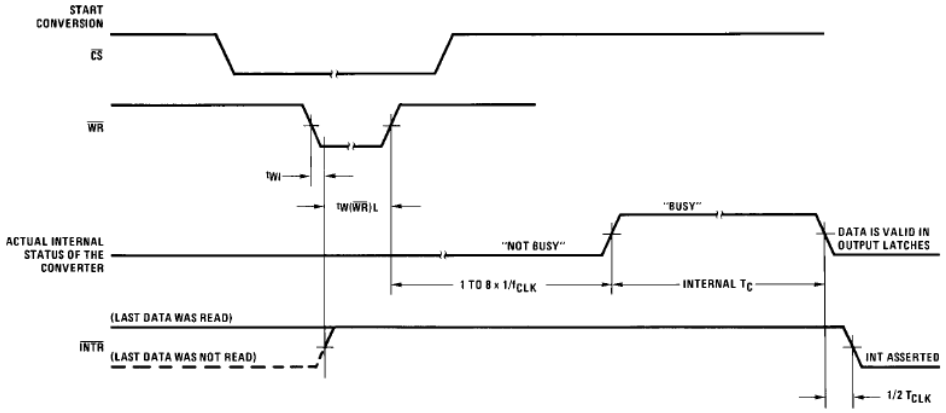


图 7. ADC0804 写入模拟电压时序图

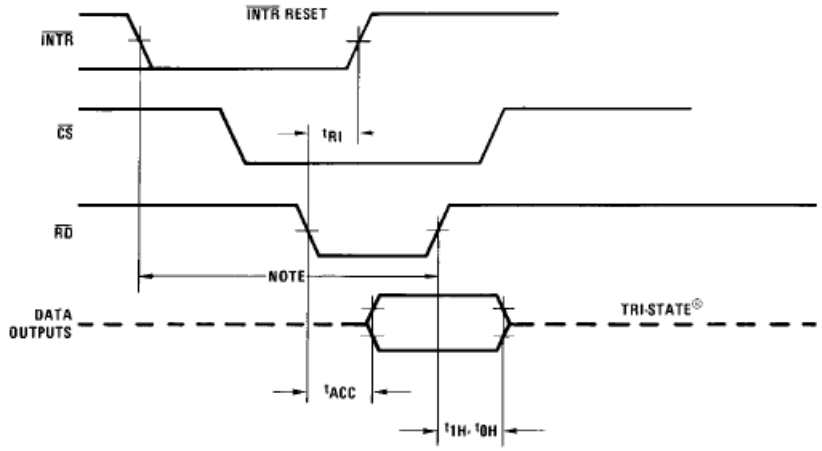


图 8. ADC0804 数字量读 时序图

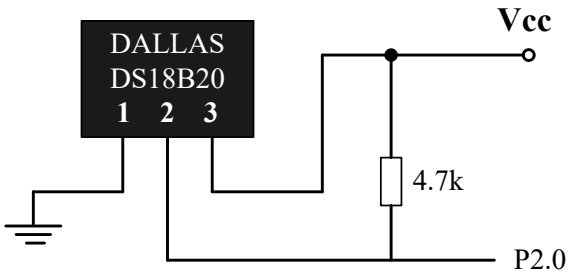
3. 2 温度传感器

3.2.1 DS18B20

采集环境的温度可采纳 IC 化的温度传感器。常用的此类温度传感器有 AD590 和 DS18B20。比较二者，AD590 只能将采集来的温度转化为电流输出，在实际应用中，须要先将 AD590 输出的电流转化为电压，再利用 A/D 转换元件进行模/数转换，将模拟量转化为数字量，最终送入单片机中，而 DS18B20 数字温度传感器能干脆将被测温度转化成串行数字信号，以供单片机处理。

DS18B20 温度传感器如图 12 所示，它是美国 DALLS[1]公司推出的一线数字式智能温度传感器，测量温度范围为- 55～125 ℃，通过简洁的编程可实现 9～12 位的数字值读取方式,分别在 93. 75ms 和 750ms 内将温度值转化为 9 位和 12 位的数量级,对应的辨别率为 0. 5 ℃、0. 25 ℃、0. 125 ℃、0. 0625 ℃，CPU 只需一根端口线就可以与 DS18B20 通信，每个 DS18B20 都有一个全球唯一的 64 位的二进制 ROM 代码标记着器件的 ID 号，整个系统要严格依据该器件单总线协议规定的时序进行工作，详细的内容是初始化器件、识别器件和进行数据交换等。

试验中我们运用 DS18B20 温度传感器检测环境温度，用于对热电偶的冷端补偿，DS18B20 温度传感器温度检测精度 0.0625℃。基于 DS18B20 的温度测量装置硬件电路如图 2 所示，DS18B20 温度传感器数据单总线连接至单片机 P2.0 脚，作为数据的读入和写出口。



DS18B20 的内部结构如图 2.5 所示。主要由 4 部分组成：64 位 ROM、温度传感器、非挥发的温度报警触发器 TH 和 TL、配置寄存器。

配置寄存器为高速暂存存储器中的第 5 个字节。DS18B20 在工作时按此寄存器中的辨别率将温度转换成相应精度的数值，其各位定义如表所示。其中，TM 为测试模式标记位，出厂时被写入“0”，不能变更；R0、R1 是温度计辨别率设置位，如表所示。

表 1 DS18B20 配置寄存器结构表

BIT7	**BIT6**	**BIT5**	**BIT4**	**BIT3**	**BIT2**	**BIT1**	**BIT0**
TM	R1	R0	1	1	1	1	1

表 2 配置寄存器与辨别率关系表

R0	R1	温度计辨别率/bit	最大转换时间/ms
0	0	9	93.75
0	1	10	187.50
1	0	11	375
1	1	12	750

图所示是 12 位辨别率的状况，假如配置为低辨别率，则其中无意义位为“0”。

表 1 DS18B20 温度值格式表

低 8 位	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
高 8 位	S	S	S	S	**S**	2^6	2^5	2^4

3.2.2 DS18B20 温度采集时序

DS18B20 具有独特的单线接口方式，在与微处理器连接时仅采纳一条口线来实现微处理器与 DS18B20 的双向通讯，这就须要肯定的协议来对读写数据提出严格的时序要求，而 AT89C51 单片机并不支持单线传输。因此，必需采纳软件的方法来模拟单线的协议时序。主机操作单线器件 DS18B20 必需遵循下面的依次。

1. 初始化

与DS18B20的全部通讯都是由一个单片机的复位脉冲和一个DS18B20的应答脉冲起先的。单片机先发一个复位脉冲，保持低电平常间最少 480μs，最多不能超过 960μs。然后，

单片机释放总线，等待 DS18B20 的应答脉冲。DS18B20 在接受到复位脉冲后等待 15～60μs 才发出应答脉冲。应答脉冲能保持 60～240μs。单片机从发送完复位脉冲到再次限制总线至少要等待 480μs。

2. ROM 操作吩咐

一旦总线主机检测到应答脉冲，便可以发起 ROM 操作吩咐。共有 5 位 ROM 操作吩咐。如表 3。

3. 内存操作吩咐

在胜利执行了 ROM 操作吩咐之后，才可以运用内存操作吩咐。主机可以供应 6 种内存操作吩咐，如表 4。

表 3 DS18B20 的 ROM 操作吩咐

吩咐类型	吩咐字节	功能说明
Raed Rom	33H	此吩咐读取激光 ROM 中的 64 位，只能用于总线上单个 DS18B20 器件的状况，多挂则会发生数据冲突
Match Rom （匹配 ROM）	55H	此吩咐后跟 64 位 ROM 序列号，寻址多挂接总线上的 DS18B20。只有序列号完全匹配的 DS18B20 才能响应后面的内存操作吩咐，其他不匹配的将等待复位脉冲。此吩咐可用于单挂接或者多挂接总线。
Skip Rom （跳过 ROM）	**CCH**	**此吩咐用于单挂接总线系统时，可以无需供应 64 位 ROM 序列号皆可运行内存操作吩咐。假如总线上接多个 DS18B20，并且在此吩咐后执行读吩咐，将会发生数据冲突。**
Search Rom （搜寻 ROM）	F0H	主机调用此吩咐，通过一个解除法过程，可以识别出总线上全部器件的 ROM 序列号。
Alarm Search （告警搜寻）	ECH	此吩咐流程图和 Search Rom 吩咐相同，但是 DS18B20 只有在最近的一次温度测量时满意了告警触发条件，才会响应。

表 4　DS18B20 内存操作吩咐

吩咐类型	吩咐字节	功能说明
Write Scratchpad （写暂存器）	4EH	此吩咐写暂存器中地址 2～4 的 3 个字节（TH、TL 和配置寄存器）在发起复位脉冲之前，3 个字节都必须要写。
Rrad Scratchpad （读暂存器）	BEH	此吩咐读取暂存器内容，从字节 0 始终读取到字节 8。主机可以随时发起复位脉冲以停止此操作。
Copy Scratchpad （复制暂存器）	48H	此吩咐将暂存器中的内容复制进 E^2RAM，以便将温度告警触发字节存入非易失内存。假如在此吩咐后产生读时隙，那么只要器件在进行复制就会输出 0，复制完成后，再输出 1。
Convert T （温度转换）	44H	此吩咐起先温度转换操作。假如在此吩咐后主机产生读时隙，只要器件在进行温度转换就会输出 0，转换完成后再输出 1。

4. 数据处理

DS18B20 要求有严格的时序来保证数据的完整。在单线 DQ 上，存在复位脉冲、应答脉冲、写“0”、写“1”、读“0”和读“1”几种信号类型。其中，除了应答脉冲之外，均由主机产生。而数据位的读和写则是通过运用读、写时隙实现的。

当主机将数据线从高电平拉至低电平常，产生写时隙。有 2 种类型的写时隙：写“1”和写“0”。全部写时隙必需在 60μs 以上（即由高拉低后持续 60μs 以上），各个写时隙之间必需保证最短 1μs 的复原时间。DS18B20 在 DQ 线变低后的 15～60μs 的窗口对 DQ 进行采样，假如为高电平，就为写“1”；假如为低电平，就为写“0”。对于主机产生写“1”时隙的状况，数据线必需先被拉低，然后释放，在写时隙起先后的 15μs，允许 DQ 线拉至高电平。对于主机写“0”时隙的状况，DQ 线必需被拉至低电平且至少保持低电平 60μs。

当主机从 DS18B20 读数据时，把数据线从高电平拉至低电平，产生读时隙。数据线 DQ 必需保持低电平至少 1μs，来自 DS18B20 的输出数据在读时隙下降沿之后 15μs 内有效。因此，在此 15μs 内，主机必需停止将 DQ 引脚置低。在读时隙结束时，DQ 引脚将通过外部上拉电阻拉回至高电平。全部的读时隙最短必需持续 60μs，各个读时隙之间必需保证最短 1μs 的复原时间。

全部的读写时隙至少须要 60μs，且每两个独立的时隙之间至少须要 1μs 的复原时间。在写时序中，主机将在拉低总线 15μs 内释放总线，并向 DS18B20 写“1”。若主机拉低总线后能保持 60μs 的低电平，则向单总线器件写“0”。DS18B20 仅在主机发出读时隙时才向主机传输数据，所以，当主机向 DS18B20 发出读数据吩咐后，必需立刻产生读时隙，以便 DS18B20 能传输数据。实现环境温度采集转换并读取数据的程序流程图如图所示。

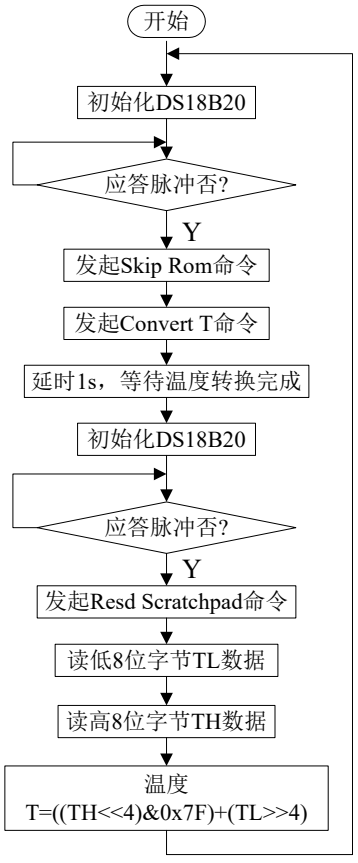


图　温度采集子程序流程图

4. 单片机主要程序

单片机主要程序如下：

```
#include <reg52.h>
#include <intrins.h>
#define uint unsigned int
#define uchar unsigned char
uchar ADOut,tt;
uchar tempHB,tempLB;
uint temp;//温度

sbit ds18b20=P2^0;
sbit adWR=P2^1;
sbit adRD=P2^2;

bit no;
bit tc;

void Delay(unsigned int nTimeDelay)
{
    //晶振为 11.0592 Mhz，1MS 延时
    unsigned int i;
    while (nTimeDelay--)
        for (i=0;i<125;i++)
          {}
}

void Reset()
{
    uint i;

    ds18b20=0;        //拉低 DQ 总线起先复位
    i=206;
    while(i>0)
        i--;          //保持 DQ 低大约 900uS
    ds18b20=1;        //释放 DQ 位
    i=8;
    while(i>0)
        i--;
}
//-----------------------------------------
bit WaitPresence()//名称: 等待芯片应答信号函数
{
    uchar i;
    bit k;
    i=16;
```

```
    while(i>0)
        i--;
     if(ds18b20==0)
        k=1;
     else
        k=0;
    i=72;
    while(i>0)
        i--;
     return k;
}
//------------------------------------
bit ReadBit()//名称:  读一位数据函数
{
    uint i;
    bit dat;
    ds18b20=0;
    i++;
    ds18b20=1;
    i+=2;
    dat=ds18b20;
    i=16;
    while(i>0)
        i--;
    return dat;
}
//------------------------------------
unsigned char ReadByte()//名称:  读一字节函数
{
    uchar i,j,dat;
    dat=0;
    for(i=1;i<=8;i++)
    {
        j=ReadBit();
        dat=(j<<7) | (dat>>1);
    }
    return dat;
}
//------------------------------------
void WriteByte(unsigned char dat)//名称:  写一字节函数
{
    uint i;
    uchar j;
    bit testb;
```

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/328026023050006115