DS18B20 数字温度计使用

1. DS18B20 基本知识

DS18B20 数字温度计是 DALLAS 公司生产的 1—Wire，即单总线器件，具有线路简单，体积小的特点。因此用它来组成一个测温系统，具有线路简单，在一根通信线，可以挂很多这样的数字温度计，十分方便。

1、DS18B20 产品的特点

（1）、只要求一个端口即可实现通信。
（2）、在 DS18B20 中的每个器件上都有独一无二的序列号。
（3）、实际应用中不需要外部任何元器件即可实现测温。
（4）、测量温度范围在－55℃到＋125℃之间。
（5）、数字温度计的分辨率用户可以从 9 位到 12 位选择。
（6）、内部有温度上、下限告警设置。

2、DS18B20 的引脚介绍

TO—92 封装的 DS18B20 的引脚排列见图 1，其引脚功能描述见表 1。

（底视图）图 1

表 1　DS18B20 详细引脚功能描述

序号	名称	引脚功能描述
1	GND	地信号
2	DQ	数据输入/输出引脚。开漏单总线接口引脚。当被用着在寄生电源下，也可以向器件提供电源。
3	VDD	可选择的 VDD 引脚。当工作于寄生电源时，此引脚必须接地。

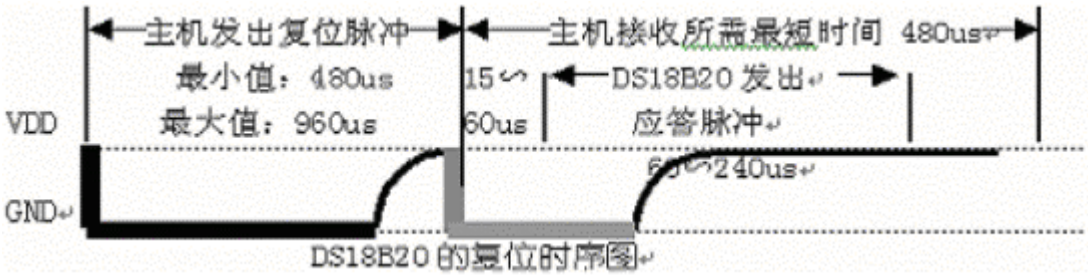
3. DS18B20 的使用方法

由于 DS18B20 采用的是 1—Wire 总线协议方式，即在一根数据线实现数据的双向传输，而对 AT89S51 单片机来说，硬件上并不支持单总线协议，因此，我们必须采用软件的方法来模拟单总线的协议时序来完成对 DS18B20 芯片的访问。

由于 DS18B20 是在一根 I/O 线上读写数据，因此，对读写的数据位有着严格的时序要求。DS18B20 有严格的通信协议来保证各位数据传输的正确性和完整性。该协议定义了几种信号的时序：初始化时序、读时序、写时序。所有时序都是将主机作为主设备，单总线器件作为

从设备。而每一次命令和数据的传输都是从主机主动启动写时序开始，如果要求单总线器件回送数据，在进行写命令后，主机需启动读时序完成数据接收。数据和命令的传输都是低位在先。

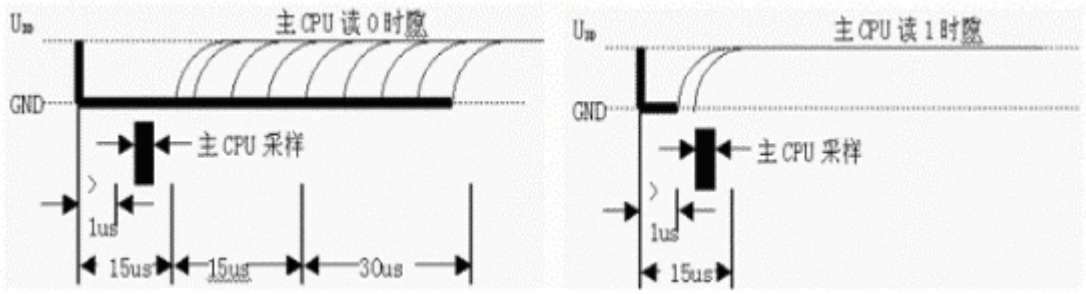
DS18B20 的复位时序



DS18B20 的读时序

对于 DS18B20 的读时序分为读 0 时序和读 1 时序两个过程。

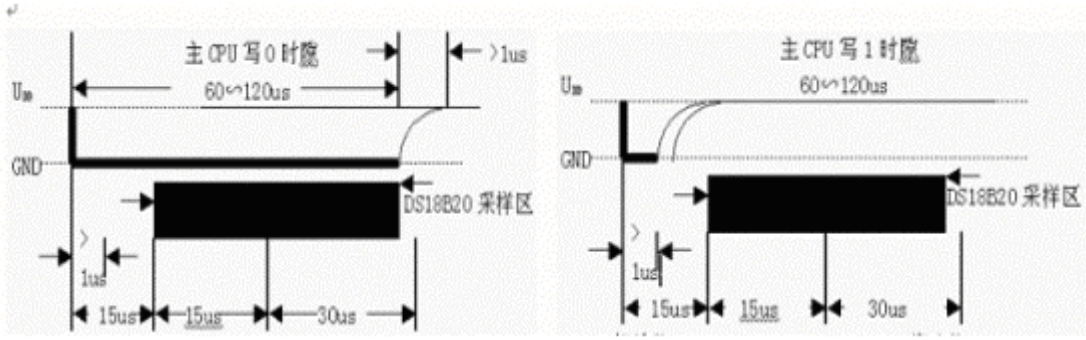
对于 DS18B20 的读时隙是从主机把单总线拉低之后，在 15 秒之内就得释放单总线，以让 DS18B20 把数据传输到单总线上。DS18B20 在完成一个读时序过程，至少需要 60us 才能完成。



DS18B20 的写时序

对于 DS18B20 的写时序仍然分为写 0 时序和写 1 时序两个过程。

对于 DS18B20 写 0 时序和写 1 时序的要求不同，当要写 0 时序时，单总线要被拉低至少 60us，保证 DS18B20 能够在 15us 到 45us 之间能够正确地采样 I0 总线上的“0”电平，当要写 1 时序时，单总线被拉低之后，在 15us 之内就得释放单总线。



4. 实验任务

用一片 DS18B20 构成测温系统，测量的温度精度达到 0.1 度，测量的温度的范围在－20 度到＋100 度之间，用 8 位数码管显示出来。

5. C 语言源程序

```
#include <AT89X52.H>
#include <INTRINS.h>
unsigned char code displaybit[]={0xfe,0xfd,0xfb,0xf7,
0xef,0xdf,0xbf,0x7f};
unsigned char code displaycode[]={0x3f,0x06,0x5b,0x4f,
0x66,0x6d,0x7d,0x07,
0x7f,0x6f,0x77,0x7c,
0x39,0x5e,0x79,0x71,0x00,0x40};
unsigned char code dotcode[32]={0,3,6,9,12,16,19,22,
25,28,31,34,38,41,44,48,
50,53,56,59,63,66,69,72,
75,78,81,84,88,91,94,97};
unsigned char displaycount;
unsigned char displaybuf[8]={16,16,16,16,16,16,16,16};
unsigned char timecount;
unsigned char readdata[8];
sbit DQ=P3^7;
bit sflag;
bit resetpulse(void)
{
unsigned char i;
DQ=0;
for(i=255;i>0;i--);
DQ=1;
for(i=60;i>0;i--);
return(DQ);
for(i=200;i>0;i--);
}
void writecommandtods18b20(unsigned char command)
{
unsigned char i;
unsigned char j;
for(i=0;i<8;i++)
{
if((command & 0x01)==0)
{
DQ=0;
```

```
for(j=35;j>0;j--);
DQ=1;
}
else
{
DQ=0;
for(j=2;j>0;j--);
DQ=1;
for(j=33;j>0;j--);
}
command=_cror_(command,1);
}
}
unsigned char readdatafromds18b20(void)
{
unsigned char i;
unsigned char j;
unsigned char temp;
temp=0;
for(i=0;i<8;i++)
{
temp=_cror_(temp,1);
DQ=0;
_nop_();
_nop_();
DQ=1;
for(j=10;j>0;j--);
if(DQ==1)
{
temp=temp | 0x80;
}
else
{
temp=temp | 0x00;
}
for(j=200;j>0;j--);
}
return(temp);
}
void main(void)
{
TMOD=0x01;
TH0=(65536-4000)/256;
```

```
TL0=(65536-4000)%256;
ET0=1;
EA=1;
while(resetpulse());
writecommandtods18b20(0xcc);
writecommandtods18b20(0x44);
TR0=1;
while(1)
{;}
}
void t0(void) interrupt 1 using 0
{
unsigned char x;
unsigned int result;
TH0=(65536-4000)/256;
TL0=(65536-4000)%256;
if(displaycount==2)
{
P0=displaycode[displaybuf[displaycount]] | 0x80;
}
else
{
P0=displaycode[displaybuf[displaycount]];
}
P2=displaybit[displaycount];
displaycount++;
if(displaycount==8)
{
displaycount=0;
}
timecount++;
if(timecount==150)
{
timecount=0;
while(resetpulse());
writecommandtods18b20(0xcc);
writecommandtods18b20(0xbe);
readdata[0]=readdatafromds18b20();
readdata[1]=readdatafromds18b20();
for(x=0;x<8;x++)
{
displaybuf[x]=16;
}
```

```
sflag=0;
if((readdata[1] & 0xf8)!=0x00)
{
sflag=1;
readdata[1]=~readdata[1];
readdata[0]=~readdata[0];
result=readdata[0]+1;
readdata[0]=result;
if(result>255)
{
readdata[1]++;
}
}
readdata[1]=readdata[1]<<4;
readdata[1]=readdata[1] & 0x70;
x=readdata[0];
x=x>>4;
x=x & 0x0f;
readdata[1]=readdata[1] | x;
x=2;
result=readdata[1];
while(result/10)
{
displaybuf[x]=result%10;
result=result/10;
x++;
}
displaybuf[x]=result;
if(sflag==1)
{
displaybuf[x+1]=17;
}
x=readdata[0] & 0x0f;
x=x<<1;
displaybuf[0]=(dotcode[x])%10;
displaybuf[1]=(dotcode[x])/10;
while(resetpulse());
writecommandtods18b20(0xcc);
writecommandtods18b20(0x44);
}
}
```