

数字温度传感器DS1820(DS18B20)的应用

一 单线数字温度计DS1820介绍

DS1820数字温度计提供9位(二进制)温度读数指示器件的温度信息经过单线接口送入DS1820或从DS1820送出因此从主机CPU到DS1820仅需一条线(和地线)DS1820的电源可以由数据线本身提供而不需要外部电源因为每一个DS1820在出厂时已经给定了唯一的序号因此任意多个DS1820可以存放在同一条单线总线上这允许在许多不同的地方放置温度敏感器件DS1820的测量范围从-55到+125增量值为0.5可在1s(典型值)内把温度变换成数字

每一个DS1820包括一个唯一的64位长的序号该序号值存放在DS1820内部的ROM(只读存储器)中开始8位是产品类型编码(DS1820编码均为10H)接着的48位是每个器件唯一的序号最后8位是前面56位的CRC(循环冗余校验)码DS1820中还有用于贮

存测得的温度值的两个8位存储器RAM编号为0号和1号1号存储器存放温度值的符号如果温度为负()则1号存储器8位全为1否则全为00号存储器用于存放温度值的补码LSB(最低位)的1表示0.5将存储器中的二进制数求补再转换成十进制数并除以2就得到被测温度值(-55到125)DS1820的引脚如图2261所示每只DS1820都可以设置成两种供电方式即数据总线供电方式和外部供电方式采取数据总线供电方式可以节省一根导线但完成温度测量的时间较长采取外部供电方式则多用一根导线但测量速度较快温度计算

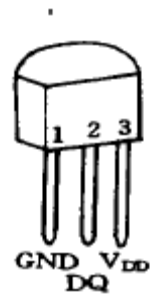


图 2. 26-1 DS1820 的引脚
 1. GND: 地;
 2. DQ: 数字输入/输出
 3. V_{DD}: 可选的 +5V 电源

1、 DS1820用9位存储温度值最高位为符号位下图为18b20的温度存储方式负温度S=1正温度S=0如 00AAH为+85,0032H为25, FF92H为-55

TEMPERATURE REGISTER FORMAT Figure 2

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
LS Byte	2 ⁶	2 ⁵	2 ⁴	2 ³	2 ²	2 ¹	2 ⁰	2 ⁻¹
	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
MS Byte	S	S	S	S	S	S	S	S

2、 DS18B20用12位存储温度值最高位为符号位下图为18b20的温度存储方式负温度S=1正温度S=0如

0550H为+850191H为25.0625,FC90H为-55

TEMPERATURE REGISTER FORMAT Figure 2

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
LS Byte	2 ³	2 ²	2 ¹	2 ⁰	2 ⁻¹	2 ⁻²	2 ⁻³	2 ⁻⁴
	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
MS Byte	S	S	S	S	S	2 ⁶	2 ⁵	2 ⁴

二DS1820工作过程及时序

DS1820工作过程中的协议如下

初始化RoM操作命令存储器操作命令处理数据

1初始化

单总线上的所有处理均从初始化开始

2ROM操作品令

总线主机检测到DS1820的存在便可以发出ROM操作命令之一这些命令如指令 代码

Read ROM(读ROM) [33H]

Match ROM(匹配ROM) [55H]

Skip ROM(跳过ROM) [CCH]

Search ROM(搜索ROM) [F0H]

Alarm search(告警搜索) [ECH]

3存储器操作命令

指令 代码

Write Scratchpad(写暂存存储器) [4EH]

Read Scratchpad(读暂存存储器) [BEH]

Copy Scratchpad(复制暂存存储器) [48H]

Convert Temperature(温度变换) [44H]

Recall EPROM(重新调出) [B8H]

Read Power supply(读电源) [B4H]

4时 序

主机使用时间隙(time slots)来读写DS1820的数据位和写命令字的位

(1)初始化

时序见图2.25-2主机总线 t_0 时刻发送一复位脉冲(最短为480us的低电平信号)接着在 t_1 时刻释放总线并进入接收状态DS1820在检测到总线的上升沿之后等待15-60us接着DS1820在 t_2 时刻发出存在脉冲(低电平持续60-240 us)如图中虚线所示

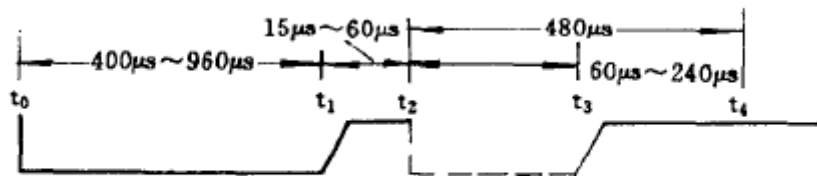


图 2. 25-2 初始化时序

以下子程序在MCS51仿真机上通过其晶振为12M.初始化子程序
RESET

PUSH B ;保存B寄存器

PUSH A 保存A寄存器

MOV A,#4 ;设置循环次数

CLR P1.0 ;发出复位脉冲

```

MOV B,#250 ;计数250次
DJNZ B,$ ;保持低电平500us
SETB P1.0 ;释放总线
MOV B,#6 ;设置时间常数
CLR C ;清存在信号标志
WAITL: JB P1.0,WH ;若总线释放跳出循环
      DJNZ B,WAITL ;总线低等待
      DJNZ ACC,WAITL;释放总线等待一段时间
      SJMP SHORT
WH: MOV B,#111
WH1: ORL C,P1.0
     DJNZ B,WH1 ;存在时间等待
SHORT: POP A
      POP B
      RET

```

(2)写时间隙

当主机总线 t_0 时刻从高拉至低电平时就产生写时间隙见图2253图2254从 t_0 时刻开始 $15\mu\text{s}$ 之内应将所需写的位送到总线上DS1820在 t_0 后 $15\sim 60\mu\text{s}$ 间对总线采样若低电平写入的位是0见图2253若高电平写入的位是1见图2254连续写2位间的间隙应大于 $1\mu\text{s}$

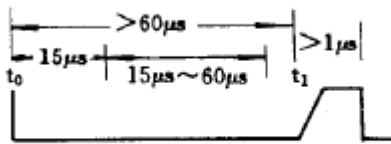


图 2.25-3 写 0 时序

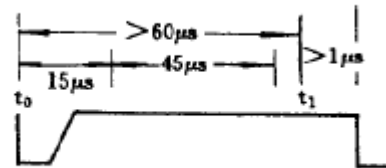


图 2.25-4 写 1 时序

写位子程序(待写位的内容在C中)

WRBIT:

```

      PUSH B ;保存B
      MOV B,#28 ;设置时间常数
      CLR P1.0 ;写开始
      NOP ;1us
      NOP ;1us
      NOP ;1us
      NOP ;1us
      MOV P1.0,C ;C内容到总线
WDLT: DJNZ B,WDLT;等待56Us
      POP B

      SETB P1.0 ;释放总线
      RET ;返回

```

写字节子程序(待写内容在A中):

WRBYTB:

PUSH B ;保存B

MOV B#8H ;设置写位个数

WLOP: RRC A ;把写的位放到C

ACALL WRBIT ;调写 1位子程序

DJNZ BWLOP ;8位全写完?

POP B

RET

(3)读时间隙

见图2255主机总线 t_0 时刻从高拉至低电平时总线只须保持低电平 $17t_s$ 之后在 t_1 时刻将总线拉高产生读时间隙读时间隙在 t_1 时刻后 t_2 时刻前有效 t_z 距 t_0 为 $15\mu s$ 也就是说 t_z 时刻前主机必须完成读位并在 t_0 后的 $60\mu s$ — $120\mu s$ 内释放总线读位子程序(读得的位到C中)

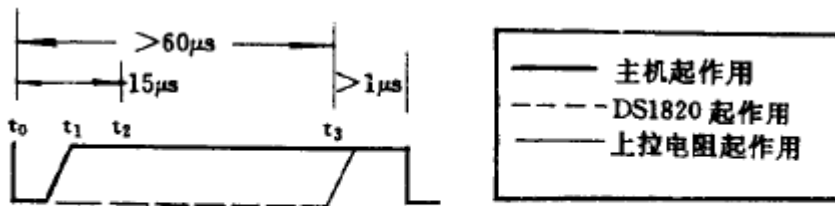


图 2.25-5 读时序

RDBIT:

PUSH B ;保存B

PUSH A ;保存A

MOV B,#23 ;设置时间常数

CLR P1.0 ;读开始图2255的 t_0 时刻

NOP ;1us

NOP ;1us

NOP ;1us

NOP ;1us

SETB P1.0 ;释放总线

MOV A,P1 ;P1口读到A

MOV C,EOH ;P1.0内容C

NOP ;1us

NOP ;1us

NOP ;1us

NOP ;1us

RDDLTL: DJNZ B,RDDLTL ;等待46us

SETB P1.0

POP A

POP B

RET

读字节子程序(读到内容放到A中)

RDBYTE:

PUSH B ;保存B

RLOP MOV B,#8H ;设置读位数

ACALL RDBIT ;调读1位子程序

RRC A ;把读到位在C中并依次送给A

DJNZ B,RLOP ;8位读完?

POP B ;恢复B

RET

三多路测量

每一片DS1820在其ROM中都存有其唯一的48位序列号在出厂前已写入片内ROM

中主机在进入操作程序前必须逐一接入1820用读ROM(33H)命令将该1820的序列号读出并登录

当主机需要对众多在线1820的某一个进行操作时首先要发出匹配ROM命令(55H)紧接着主机提供64位序列(包括该1820的48位序列号)之后的操作就是针对该1820的而所谓跳过ROM命令即为之后的操作是对所有1820的框图中先有跳过ROM即是启动所有1820进行温度变换之后通过匹配ROM再逐一地读回每个1820的温度数据

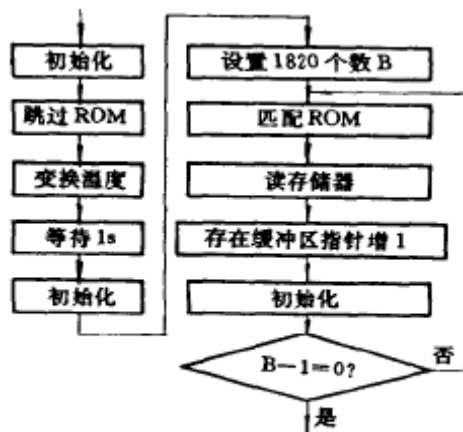


图 2.25-6 多路测温程序框图

在1820组成的测温系统中主机在发出跳过ROM命令之后再发出统一的温度转换启动码44H就可以实现所有1820的统一转换再经过1s后就可以用很少的时间去逐一读取这种方式使其T值往往小于传统方式由于采取公用的放大电路和AD转换器只能逐一转换显然通道数越多这种省时效应就越明显

四实际应用

1ds1820序列号获得

```

;-----|
; 读出ds1820序列号应用程序,P1.6接ds1820 |
;-----|
    
```

```

ORG 0000H
AJMP MAIN
    
```

```
ORG 0020H
MAIN:
MOV SP,#60H
CLR EA ;使用ds1820一定要禁止任何中断产生
LCALL INT ;初始化ds1820
MOV A,#33H

LCALL WRITE ;送入读ds1820的ROM命令
LCALL READ ;开始读出当前ds1820序列号
MOV 40H,A
LCALL READ
MOV 41H,A
LCALL READ
MOV 42H,A
LCALL READ
MOV 43H,A
LCALL READ
MOV 44H,A
LCALL READ
MOV 45H,A
LCALL READ
MOV 46H,A
LCALL READ
MOV 47H,A
SETB EA
SJMP $
INT: ;初始化ds1820子程序
CLR EA
L0:CLR P1.6 ;ds1820总线为低复位电平
MOV R2,#200
L1:CLR P1.6
DJNZ R2,L1 ;总线复位电平保持400us
SETB P1.6 ;释放ds1820总线
MOV R2,#30
L4:DJNZ R2,L4 ;释放ds1820总线保持60us
CLR C ;清存在信号
ORL C,P1.6
JC L0 ;存在吗?不存在则重新来
MOV R6,#80
L5:ORL C,P1.6
JC L3
DJNZ R6,L5
SJMP L0
L3:MOV R2,#240
L2:DJNZ R2,L2
```

```

RET
WRITE: ;向ds1820写操作命令子程序
CLR EA
MOV R3,#8 ;写入ds1820的bit数,一个字节8个bit

WR1:SETB P1.6
MOV R4,#8
RRC A ;把一个字节data(A)分成8个bit环移给 C
CLR P1.6 ;开始写入ds1820总线要处于复位(低)状态
WR2:DJNZ R4,WR2 ;ds1820总线复位保持16us
MOV P1.6,C ;写入一个bit
MOV R4,#20
WR3:DJNZ R4,WR3 ;等待40us
DJNZ R3,WR1 ;写入下一个bit
SETB P1.6 ;重新释放ds1820总线
RET
READ:
CLR EA
MOV R6,#8 ;连续读8个bit
RE1:CLR P1.6 ;读前总线保持为低
MOV R4,#4
NOP
SETB P1.6 ;开始读总线释放
RE2:DJNZ R4,RE2 ;持续8us
MOV C,P1.6 ;从ds1820总线读得一个bit
RRC A ;把读得的位值环移给 A
MOV R5,#30
RE3:DJNZ R5,RE3 ;持续60us
DJNZ R6,RE1 ;读下一个bit
SETB P1.6 ;重新释放ds1820总线
RET
END

```

2温度转换和读取

```

;|-----|
;| 获取单个ds1820转化的温度值的应用程序,P1.6接ds1820 |
;|-----|

```

```

ORG 0000H
AJMP MAIN
ORG 0020H
MAIN:
MOV SP,#60H
LCALL GET_TEMP
SJMP $
GET_TEMP:

```

CLR PSW.4

```
SETB PSW.3 ;设置工作寄存器当前所在的区域
CLR EA ;使用ds1820一定要禁止任何中断产生
LCALL INT ;调用初使化子程序
MOV A,#0CCH
LCALL WRITE ;送入跳过ROM命令
MOV A, #44H
LCALL WRITE ;送入温度转换命令
LCALL INT ;温度转换完全,再次初使化ds1820
MOV A,#0CCH
LCALL WRITE ;送入跳过ROM命令
MOV A,#0BEH
LCALL WRITE ;送入读温度暂存器命令
LCALL READ
MOV R7,A ;读出温度值低字节存入R7
LCALL READ
MOV R6,A ;读出温度值高字节存入R6
SETB EA
RET
INT: ;初始化ds1820子程序
CLR EA
L0:CLR P1.6 ;ds1820总线为低复位电平
MOV R2,#200
L1:CLR P1.6
DJNZ R2,L1 ;总线复位电平保持400us
SETB P1.6 ;释放ds1820总线
MOV R2,#30
L4:DJNZ R2,L4 ;释放ds1820总线保持60us
CLR C ;清存在信号
ORL C,P1.6
JC L0 ;存在吗?不存在则重新来
MOV R6,#80
L5:ORL C,P1.6
JC L3
DJNZ R6,L5
SJMP L0
L3:MOV R2,#240
L2:DJNZ R2,L2
RET
WRITE: ;向ds1820写操作命令子程序
CLR EA
MOV R3,#8 ;写入ds1820的bit数,一个字节8个bit
WR1:SETB P1.6
```

```
MOV R4,#8
RRC A ;把一个字节data(A)分成8个bit环移给 C
CLR P1.6 ;开始写入ds1820总线要处于复位(低)状态
WR2:DJNZ R4,WR2 ;ds1820总线复位保持16us
MOV P1.6,C ;写入一个bit
MOV R4,#20
WR3:DJNZ R4,WR3 ;等待40us
DJNZ R3,WR1 ;写入下一个bit
SETB P1.6 ;重新释放ds1820总线
RET
READ:
CLR EA
MOV R6,#8 ;连续读8个bit
RE1:CLR P1.6 ;读前总线保持为低
MOV R4,#4
NOP
SETB P1.6 ;开始读总线释放
RE2:DJNZ R4,RE2 ;持续8us
MOV C,P1.6 ;从ds1820总线读得一个bit
RRC A ;把读得的位值环移给 A
MOV R5,#30
RE3:DJNZ R5,RE3 ;持续60us
DJNZ R6,RE1 ;读下一个bit
SETB P1.6 ;重新释放ds1820总线
RET
END
```