

关键词: 单总线; 温湿度传感器; 单片机; AM2302

收稿日期: 2020-07-06

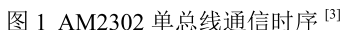
夏铁铮¹ 于虹博²

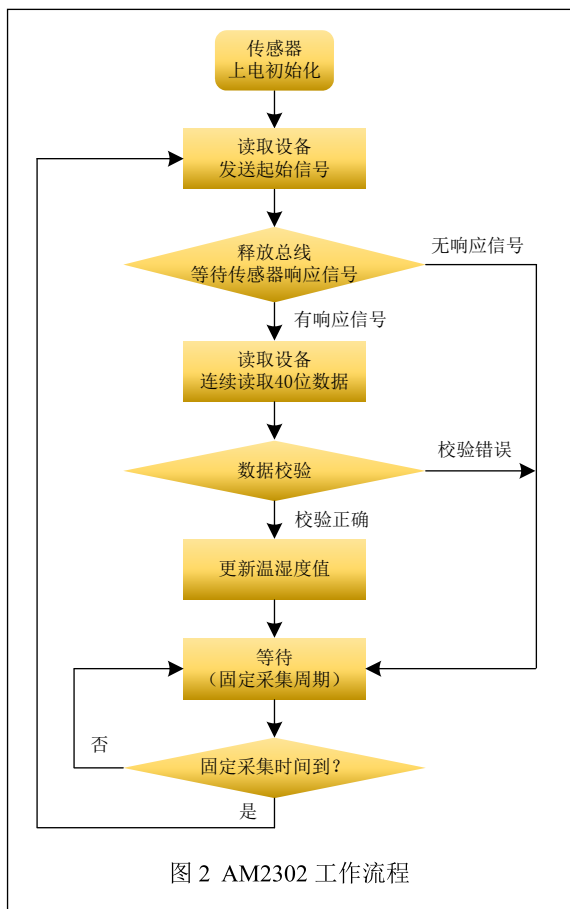
1. 哈尔滨芯明天科技有限公司, 黑龙江哈尔滨 150080; 2. 黑龙江外国语学院, 黑龙江哈尔滨, 150025

单

总线即 one-wire 总线，是美国 DALLAS 公司推出的外围串行扩展总线技术。它采用单根信号线，既传输时钟又传输数据，而且数据传输是双向的，具有节省 I/O 口线、资源结构简单、成本低廉、便于总线扩展和维护等诸多优点^[1]。市面常见的温度传感器如 18B20、温湿度传感器 DHT11 和 AM2302 等传感器均采用该总线端口。下面以常见的 AM2302 型温湿

AM2302 采用单总线模式, 基本指标如下: 全量程标定校准, 温度测量范围: $-40^{\circ}\text{C} \sim 80^{\circ}\text{C}$; 湿度测量精度为 $\pm 2.0\%\text{RH}$; 温度测量精度 $\pm 0.5^{\circ}\text{C}$; 响应时间 $< 5\text{s}$ [2]。通讯过程如图 1 所示。用户主机 (MCU) 发送一次起始信号 (把数据总线 SDA 拉低至少 $800\mu\text{s}$) 后, AM2302 从休眠模式转换到高速模式。待主机开始信号结束后, AM2302 发送响应信号, 从数





据总线 SDA 串行送出 40Bit 的数据，先发送字节的高位；发送的数据依次为湿度高位、湿度低位、温度高位、温度低位、校验位，发送数据结束触发一次信息采集，采集结束传感器自动转入休眠模式，直到下一次通信来临。

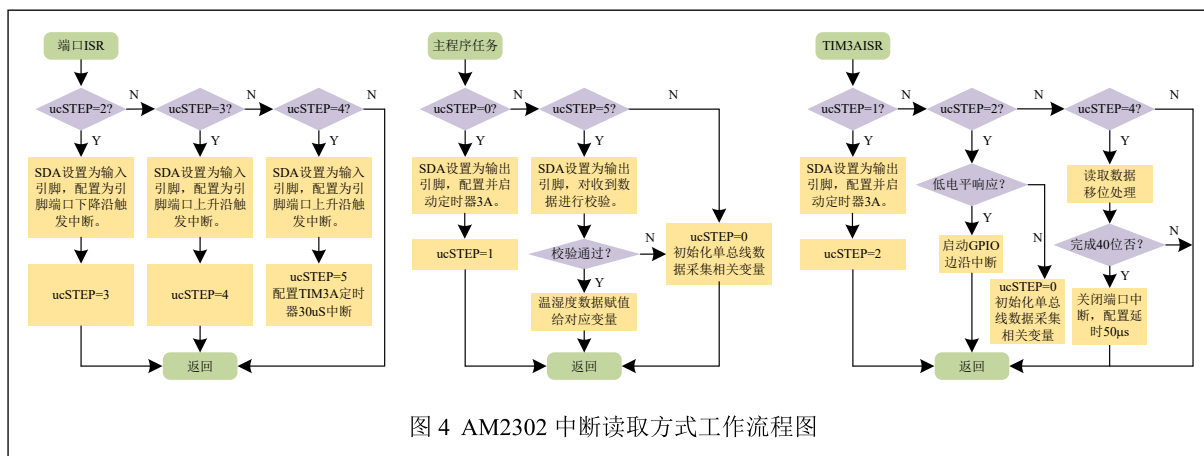
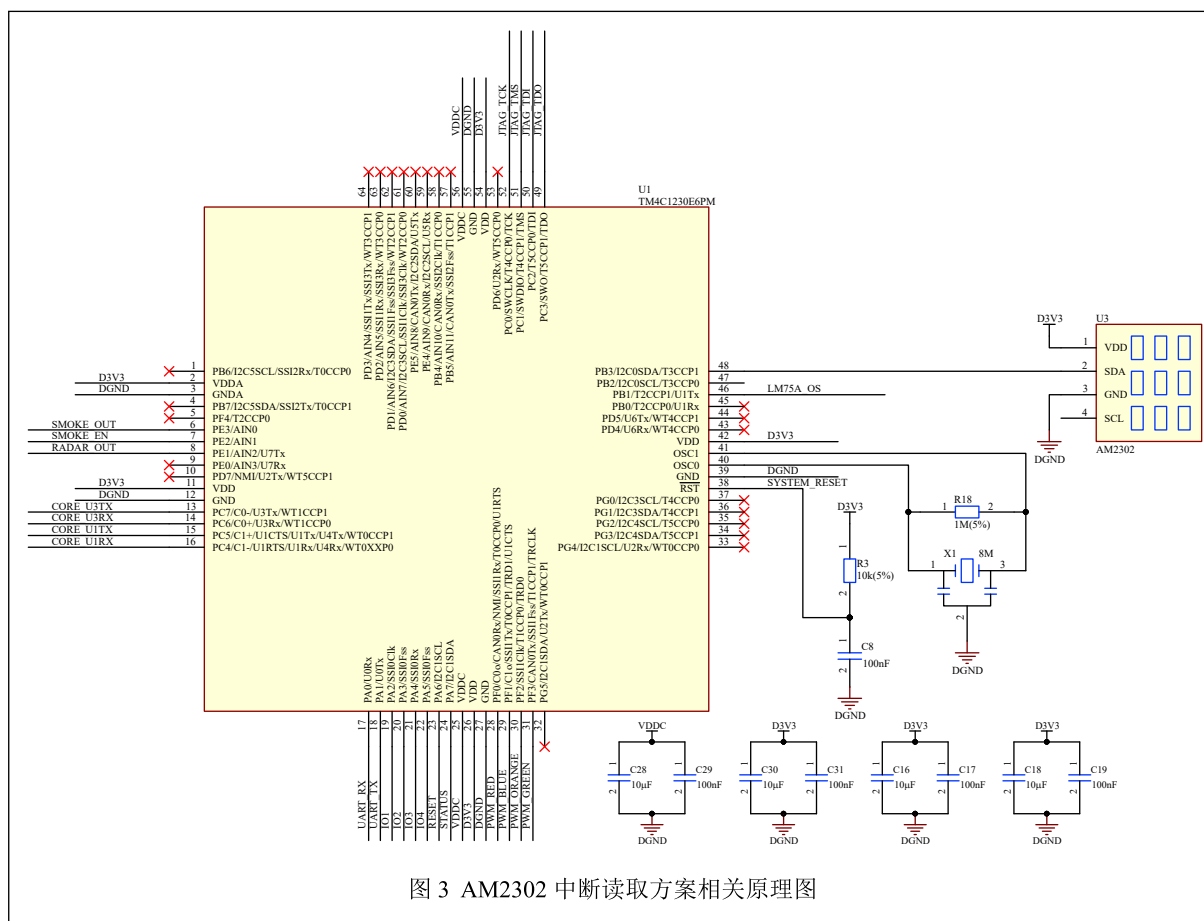
二、设计方案及代码

目前所有应用案例中均采用厂家推荐的通讯例程，即采用 GPIO 端口模拟单总线的方式进行通讯，通讯流程图如图 2 所示。从例程中可以看出，读取设备发送起始信号、等待总线响应和读取数据的操作中都是通过 delay() 函数来实现的延时功能，在单任务或者实时性要求不高的系统中，该例程的实现对整个系统功能的影响可以忽略，但是往往实际的嵌入式产品实现方案中对实时性要求很高，整个数据读取过程最少耗时（数据全 0） $1\text{ms}+20\mu\text{s}+160\mu\text{s}+40\times(50+28)$

$\mu\text{s}=4.3\text{ms}$ ，在数据全 1 情况下达到了 $1\text{ms}+20\mu\text{s}+160\mu\text{s}+32\times(50+70)\mu\text{s}+8\times(50+28)\mu\text{s}=5.64\text{ms}$ ，在实时性要求较高的系统中耗时巨大，严重降低了系统的响应时间。如果系统中存在 ISR 函数，ISR 函数的运行会使原有的延时函数出现偏差，其他任务不确定的 ISR 次数使得 delay 函数的校准无法实现。因此，原有厂家例程不适用于多任务、实时性要求高的产品应用。

本文中，根据实际传感器信号特点，采用 GPIO 操作结合定时器及 GPIO 边沿触发中断的方式进行检测，其原理图如图 3 所示。

下面以 TI 公司推出的 TM4C1230 型号单片机为例进行说明。首先，在传感器上电后，单总线 SDA 引脚初始化为带有上拉的 OUT 引脚，初始化变量 ucSTEP=0。传感器读取函数为分支结构，在 ucSTEP=0 分支中引脚置低，配置定时器中断时间为 1mS，ucSTEP++，退出函数，运行主循环其他任务，当定时时间达到后进入定时器 ISR，在 ISR 当中，条件选择进入 ucSTEP=1 分支，SDA 引脚设置为输入端口，定时器中断时间设置为 $32\mu\text{s}$ ，退出中断，ucSTEP++。中断时间达到后，再次进入定时器 ISR，在 ucSTEP=2 分支中读取端口状态，当端口状态为低电平时，说明端口正常工作，使能端口输入上升沿中断，退出 ISR。当端口上升沿到来后，进入端口 ISR，修改中断条件为下降沿中断，ucSTEP++。当下降沿到来后，再次进入端口 ISR，在 ucSTEP=3 分支中，配置中断条件为上升沿中断，ucSTEP++。当上升沿再次进入后，进入端口 ISR，确认响应信号正常，进入数据读取模式，在 ucSTEP=4 分支中，配置中断为上升沿中断，用于每个数据位起始边沿的检测，设定定时器中断时间为 $30\mu\text{s}$ ，在 $50\mu\text{s}$ 电平结束进入上升沿后 $30\mu\text{s}$ 进入定时器中断，读取 SDA 状态，读取相应数据位并退出定时器中断，当数据再次到来后，采用相同方式读取并移位，直至所有数据接收完成为止，接收完成后 ucSTEP=5，退出中断。在主循环中再次进入任务函数，校验结果吻合后，读取成功，传感器参数赋值后，读取标志和 ucSTEP 变量清零，其工作流程如图 4 所示。该实例部分代码如下：



*main.c

```
#define AM2320_SDA_BASE GPIO_PORTB_BASE //
```

```
#define AM2320_SDA_PIN GPIO_PIN_3
```

```
int main(void)
```

$$\{ \bullet \bullet \bullet \bullet \bullet \bullet \}$$

```
// 初始化 AM2302 温湿度传感器操控引脚
```

ROM GPIODirModeSet (AM2320 SDA BASE,

```
AM2320_SDA_PIN, GPIO_DIR_MODE_OUT);ROM
```

GPIOPadConfigSet (AM2320 SDA BASE, AM2320

```

SDA_PIN, GPIO_STRENGTH_8MA, GPIO_PIN_TYPE_
OD);
ROM_GPIOPinWrite (AM2320_SDA_BASE, AM2320_
SDA_PIN, AM2320_SDA_PIN & 0xFF);
ROM_GPIODirModeSet (AM2320_SDA_BASE,
AM2320_SCL_PIN, GPIO_DIR_MODE_OUT); ROM_
GPIOPadConfigSet (AM2320_SDA_BASE, AM2320_
SCL_PIN, GPIO_STRENGTH_12MA, GPIO_PIN_
TYPE_OD);
ROM_GPIOPinWrite (AM2320_SDA_BASE, AM2320_
SCL_PIN, AM2320_SCL_PIN & 0x00); .....
while(1){.....
if (ucTandHSampleFlag == 0xFF)
{
if ((EnableCabinTemperature == ENABLE)
|| (EnableCabinHumidity == ENABLE))
{// 此处采集温湿度
if(ucSTEP==0)
{
SDA_Mode_Out_OD(); // 输出模式
SDA_Dout_0;
ucSTEP++;
ROM_TimerLoadSet(TIMER3_BASE,TIMER_A,
ROM_SysCtlClockGet()/(1000));
ROM_TimerEnable(TIMER3_BASE, TIMER_A);
if(ucSTEP==5)
SDA_Mode_Out_OD();
// 读取结束, 引脚改为输出模式
.....
//SDA_Dout_1// 主机拉高
// 检查读取的数据是否正确
AM2302_Data.byte.data5=AM2302_Data.byte.data4+
AM2302_Data.byte.data3+AM2302_Data.byte.data2+
AM2302_Data.byte.data1;
if(AM2302_Data.byte.data0==(uint8_t) (AM2302_Data.
byte.data4+AM2302_Data.byte.data3+AM2302_Data.
byte.data2+AM2302_Data.byte.data1))
{

```

```

NewSendData.ucCabinHumidity = (uint16_t)(AM2302_
Data.byte.data4<<8|AM2302_Data.byte.data3)/10; //
(uint16_t)AM2302_Data.byte.data0;
NewSendData.scCabinTemperature = (uint16_t)
(AM2302_Data.byte.data2<<8|AM2302_Data.byte.
data1)/10;//(int16_t)AM2302_Data.byte.data2;
}
AM2302_Data.uint64 = 0;
ucSTEP = 0;
ucTandHSampleFlag = 0;
}
// 定时器 ISR
void Timer3AIntHandler(void)
{
static uint64_t temp = 0;
static uint8_t ucCounter =0;
uint8_t a;
ROM_TimerIntClear(TIMER3_BASE, TIMER_TIMA_
TIMEOUT);
ROM_TimerDisable(TIMER3_BASE,TIMER_TIMA_
TIMEOUT);
if(ucSTEP==1)
{
ucSTEP++;
ucCounter =0;
SDA_Mode_IN();
ROM_TimerLoadSet(TIMER3_BASE, TIMER_
A, ROM_SysCtlClockGet()/(1000000/32))//
30us
ROM_TimerEnable(TIMER3_BASE,TIMER_
TIMA_TIMEOUT);
}
else if(ucSTEP==2){
a=ROM_GPIOPinRead(AM2320_SDA_
BASE,AM2320_SDA_PIN);
if(a==Bit_RESET) { // 判断从机是否有低电平
响应信号 如不响应则跳出, 响应则向下运行
GPIOIntEnable(AM2320_SDA_BASE, AM2320_SDA_

```

```

PIN);

    IntEnable(INT_GPIOB);}
    else{
        ucDHT11_STEP = 0;
        ucTandHSampleFlag = 0;}
    }
    else if(ucDHT11_STEP==4){
        if((uint8_t)ROM_GPIOPinRead(AM2320_
SDA_BASE,AM2320_SDA_PIN)==Bit_SET)    //x us 后
仍为高电平表示数据 “1”
        {
            AM2302_Data.uint64|=((uint64_t)0x01<<(39-
ucCounter));
        }
        else {
            AM2302_Data.uint64 &=~((uint64_
t)0x01<<(39-ucCounter));}
            if(++ucCounter==40) {
                ucSTEP++;
                GPIOIntDisable(AM2320_SDA_BASE,
AM2320_SDA_PIN);
                ROM_TimerLoadSet(TIMER3_BASE, TIMER_A, ROM_
SysCtlClockGet()/(1000000/50));// 50us
            }
        }
    }
    else
        return;
}

// 端口 ISR
void PorTBIntHandler(void)
{
    .....

    if(ucSTEP==2) {
        ROM_GPIOIntTypeSet(AM2320_SDA_BASE,
AM2320_SDA_PIN,GPIO_FALLING_EDGE);
        GPIOIntEnable(AM2320_SDA_BASE, AM2320_SDA_
PIN);

        ucSTEP++; }

```

```

        else if(ucDHT11_STEP==3) {
            ROM_GPIOIntTypeSet(AM2320_SDA_
BASE, AM2320_SDA_PIN , GPIO_RISING_EDGE);
            GPIOIntEnable(AM2320_SDA_BASE, AM2320_SDA_
PIN);

            ucSTEP++; }
        else if(ucDHT11_STEP==4) {
            ROM_GPIOIntTypeSet(AM2320_SDA_
BASE, AM2320_SDA_PIN , GPIO_RISING_EDGE);
            GPIOIntEnable(AM2320_SDA_BASE, AM2320_SDA_
PIN);

            ROM_TimerLoadSet(TIMER3_BASE, TIMER_
A,ROM_SysCtlClockGet()/(1000000/30));// 30us
            ROM_TimerEnable(TIMER3_
BASE,TIMER_TIMA_TIMEOUT);

            ucSTEP++;
        }
        else;
    }
}

```

三、总结

通过实际调试，该程序可以稳定运行于基于 TM4C1230E6PM 的嵌入式系统中，经过连续测试，传感器数值可以保持正确采集，没有通讯错误出现，其它任务均正常运行，原有阻塞式读取方式，通过断点采集单次无温湿度采集要求 main() 主循环任务执行时间平均为 10.35ms，有温湿度采集要求平均执行时间为 15.05ms，采用中断方式，无采集要求 main() 主循环任务平均执行时间为 10.35ms，有温湿度采集要求平均执行时间为 10.98ms，单个循环时间降低 27%，在温湿度传感器单次采集周期内，处理器单线程时间的 95% 以上可穿插其他主程序，系统效率提高明显。

参考文献

- [1] 吴江, 陈尚松. 用单总线技术设计环境状态监控系统 [J]. 电子技术应用, 2000, 26(6): 31-33.
- [2] 朱嵘涛, 徐爱钧. 单总线传感器 AM2302 温湿度测控系统原理及应用 [J]. 单片机与嵌入式系统应用,

2016,16(4): 46-48.

[3] 广州奥松电子有限公司. 温湿度模块 AM2302 产品手册 [M]. Rev 1.1, 2016.

Research on Interrupt Reading Method of Temperature and Humidity Sensor Based on Single Bus Protocol

XIA Tie-zheng¹, YU Hong-bo²

(1. Harbin Core Tomorrow Science & Technology Co., Ltd., Harbin 150080, China; 2. Heilongjiang International University, Harbin 150025, China)

Abstract: A method of reading and writing one-wire bus sensor controlled by external interrupt combined with timer is introduced in this paper. The long-term occupancy of system resources in traditional blocking reading mode is avoided during this method. In applications requiring long-time parameter monitoring, system resources can be effectively utilized and processing lag or even omission after external IO and communication signals entered the blocking process can be avoided. The fluctuation of output waveform cycle and pulse width which is caused by uncertain interrupt service program (ISR), the communication failure which is caused by excessive pulse width fluctuation also can be avoided. With the advantages of high stability, low resource consumption, no blocking and accurate timing, effectively improves the utilization rate of new MCU, unnecessary time overhead is reduced, and the schemes was made more in line with the trend of MCU high speed and real-time performance. The schemes have been validated experimentally and applied in batch products, which has practical engineering significance.

Key words: one-wire bus; temperature and humidity sensor; MCU; AM2302

作者简介

夏铁铮: 哈尔滨芯明天科技有限公司, 工程师, 研究方向为嵌入式测控产品开发。

通信地址: 哈尔滨市南岗区学府路 191 号哈尔滨创业孵化产业园 I2 栋 104 室

邮编: 150086

邮箱: xiatiezheng@163.com

于虹博: 黑龙江外国语学院, 讲师, 研究方向为计算机软硬件技术类教学。