催化式燃气探测数字传感器

BM22S3031-1
Arduino Library V1.0.1 说明

版本：V1.01　日期：2023-09-11

www.bestmodulescorp.com

目录

简介

BM22S3031-1 是倍创推出的一款催化式燃气探测数字传感器，使用 UART 通信方式。本文档对 BM22S3031-1 的 Arduino Lib 函数、Arduino Lib 安装方式进行说明。范例使用了 BMA36M303 模块，演示了报警信号输出、读取燃气浓度的功能。适用型号：

型号	说明
BM22S3031-1	催化式燃气探测数字传感器
BMA36M303	板载 BM22S3031-1 传感器

Arduino Lib 函数

Arduino Lib 名称：BM22S3031-1		Lib 版本：V1.0.1
构造函数 & 初始化		
1	BM22S3031_1(uint8_t statusPin, HardwareSerial *theSerial = &Serial)	
	描述	构造函数 (硬件 UART)
	参数	statusPin：STATUS 引脚，连接 BM22S3031-1 的 STATUS 引脚 或 BMA36M303 的 STA 引脚 *theSerial：可用的硬件 Serial 接口
	返回值	—
	备注	—
2	BM22S3031_1(uint8_t statusPin, uint8_t rxPin, uint8_t txPin)	
	描述	构造函数 (软件 UART)
	参数	statusPin：STATUS 引脚，连接 BM22S3031-1 的 STATUS 引脚 或 BMA36M303 的 STA 引脚 rxPin：RX 引脚，连接 BM22S3031-1 或 BMA36M303 的 TX 引脚 txPin：TX 引脚，连接 BM22S3031-1 或 BMA36M303 的 RX 引脚
	返回值	—
	备注	—
3	void begin()	
	描述	模块初始化
	参数	—
	返回值	void
	备注	—
4	void preheatCountdown ()	
	描述	等待完成模块预热
	参数	—
	返回值	void
	备注	1. 此函数运行结束，代表预热完成； 2. 当模块上电时，此函数判断其自动输出功能是否使能 a. 使能：将一直读取自动输出的信息包中抓取预热剩余时间，当该时间为零时退出函数； b. 除能：延时约 180s 后退出函数

功能函数		
5	uint8_t getSTATUS()	
	描述	获取 STATUS 引脚电平
	参数	—
	返回值	STATUS 引脚电平： 0：低电平 1：高电平
	备注	—
6	uint8_t getWorkStatus()	
	描述	获取模块当前的工作状态
	参数	—
	返回值	设备状态 (1 字节)： Bit 0 为 1：表示标定中 Bit1~2：保留 Bit 3 为 1：表示标定动作完成 Bit 4 为 1：表示零点标定完成 Bit 5 为 1：表示带气标定完成 Bit 6 为 1：表示传感器故障状态 Bit 7 为 1：表示报警状态
	备注	—
7	uint16_t readGasValue()	
	描述	读取燃气浓度值
	参数	—
	返回值	燃气浓度值，单位：ppm
	备注	最大有效值 10000ppm
8	uint16_t readADValue()	
	描述	读取当前燃气浓度对应的 A/D 值
	参数	—
	返回值	12-bit A/D 值
	备注	—
9	uint16_t readRefValue()	
	描述	读取上电基准值
	参数	—
	返回值	12-bit A/D 值
	备注	—
10	uint8_t requestInfoPackage(uint8_t array[])	
	描述	获取模块的所有信息
	参数	array[]：用于存储模块信息 (34 字节)
	返回值	执行情况 (1)
	备注	1. 建议除能串口自动输出功能后使用 2. 每个字节的含义请见 Datasheet

11	bool isInfoAvailable()	
	描述	查询是否接收到模块自动输出的信息
	参数	—
	返回值	接收情况： true：已收到 false：未收到
	备注	在使能串口自动输出功能后使用
12	void readInfoPackage(uint8_t array[])	
	描述	读取模块自动输出的信息
	参数	array[]：用于存储模块信息 (34 字节)
	返回值	void
	备注	在“isInfoAvailable()==true”后使用
13	uint8_t resetModule()	
	描述	复位模块
	参数	—
	返回值	执行情况 (1)
	备注	此函数执行后模块进入预热状态
14	uint8_t restoreDefault ()	
	描述	将模块的参数恢复至出厂设置
	参数	—
	返回值	执行情况 (1)
	备注	此函数执行后模块进入预热状态
参数查询		
15	uint16_t getFWVer()	
	描述	查询固件版本
	参数	—
	返回值	固件版本号
	备注	0x0105 表示 V1.5
16	uint8_t getProDate(uint8_t array[])	
	描述	查询生产日期
	参数	array[]：用于存储生产日期，array[0]/array[1]/array[2]：年 / 月 / 日
	返回值	执行情况 (1)
	备注	如：0x21,0x06,0x2 表示 2021/6/2
17	bool isAutoTx()	
	描述	查询串口自动输出功能是否使能
	参数	—
	返回值	串口自动输出功能状态： true：使能自动输出 false：除能自动输出
	备注	—

18	uint8_t getStatusPinActiveMode()	
	描述	查询模块报警时，STATUS 引脚输出高电平或低电平
	参数	—
	返回值	报警电平： 0x08：输出高电平 0x00：输出低电平
	备注	—
19	uint16_t getAlarmThreshold()	
	描述	查询报警的燃气浓度阈值
	参数	—
	返回值	燃气报警阈值，单位：ppm
	备注	—
20	uint16_t getExitAlarmThreshold()	
	描述	查询退出报警的燃气浓度值
	参数	—
	返回值	退出燃气报警的阈值，单位：ppm
	备注	—
21	uint8_t getCALTime()	
	描述	查询标定时间
	参数	—
	返回值	标定时间，单位：秒
	备注	—
参数设置		
22	uint8_t setAutoTx(uint8_t autoTx = AUTO)	
	描述	设置模块是否通过 TX 引脚自动输出所有的模块数据
	参数	autoTx：自动输出状态 0x08(AUTO)：自动输出 (默认) 0x00(PASSIVE)：不自动输出
	返回值	执行情况 (1)
	备注	1. 信息输出周期为 1 秒 2. 在信息发送期间 (约 40ms)，模块无法接收任何指令
23	uint8_t setStatusPinActiveMode(uint8_t statusMode = HIGH_LEVEL)	
	描述	设置模块报警时，STATUS 引脚的输出电平
	参数	statusMode：报警时，STATUS 引脚的输出状态 0x08(HIGH_LEVEL)：输出高电平 (默认) 0x00(LOW_LEVEL)：输出低电平
	返回值	执行情况 (1)
	备注	—
24	uint8_t setAlarmThreshold(uint16_t alarmThreshold = 4500)	
	描述	设置燃气报警的浓度值
	参数	alarmThreshold：燃气报警的浓度值 (默认 4500)，单位：ppm
	返回值	执行情况 (1)
	备注	—

25	uint8_t setExitAlarmThreshold(uint8_t exitAlarmThreshold = 1500)	
	描述	设置退出燃气报警的浓度值
	参数	exitAlarmThreshold：退出燃气报警的浓度值 (默认 1500)，单位：ppm
	返回值	执行情况 (1)
	备注	该值需小于燃气报警的浓度值
26	uint8_t setCALTime(uint8_t CALtime = 120)	
	描述	设置标定时间
	参数	CALtime：标定时间 (默认 120)，范围：10≤CALtime<255，单位：秒
	返回值	执行情况 (1)
	备注	默认 120 秒
27	uint8_t calibrateModule()	
	描述	启动带气标定功能
	参数	—
	返回值	执行情况 (1)
	备注	1. 标定期间不可向模块发送指令 2. 标定需在标准气体下进行，详见 Datasheet 3. 启动后需等待模块标定完成，可使用 calibrateCountdown()
28	void calibrateCoundown()	
	描述	等待模块标定完成
	参数	—
	返回值	void
	备注	1. 此函数运行结束，代表标定完成； 2. 当模块上电时，此函数判断其自动输出功能是否使能 a. 使能：将一直读取自动输出的信息包中抓取标定剩余时间，当该时间为零时退出函数； b. 除能：延时设定的标定时间 (预设约 120s) 后退出函数

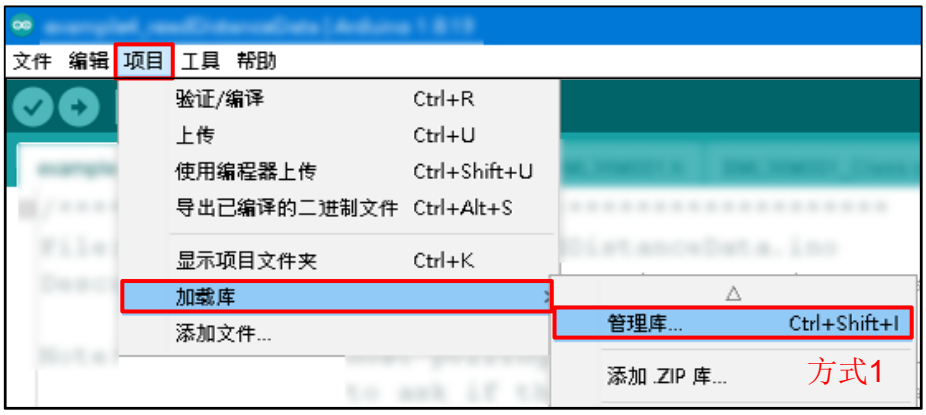
注 1：0 – 指令执行成功；1 – 模块应答的数据校验错误；2 – 通信超时

Arduino Lib 下载及安装

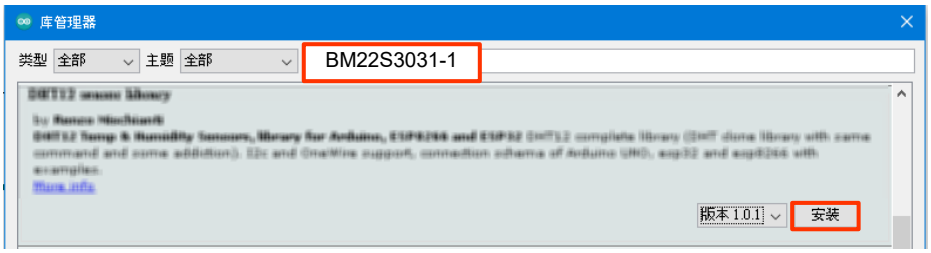
BM22S3031-1 Library：可参考下面两种方法安装 BM22S3031-1 的 Arduino Library。

方式 1：搜索安装

搜索安装：Arduino IDE → 项目 → 加载库 → 管理库 → 搜索 BM22S3031-1 → 安装



搜索安装流程 **1**

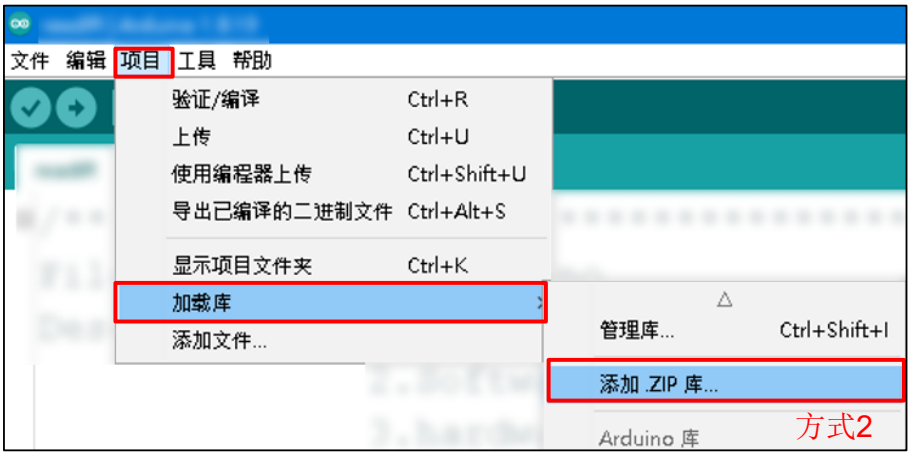


搜索安装流程 **2**

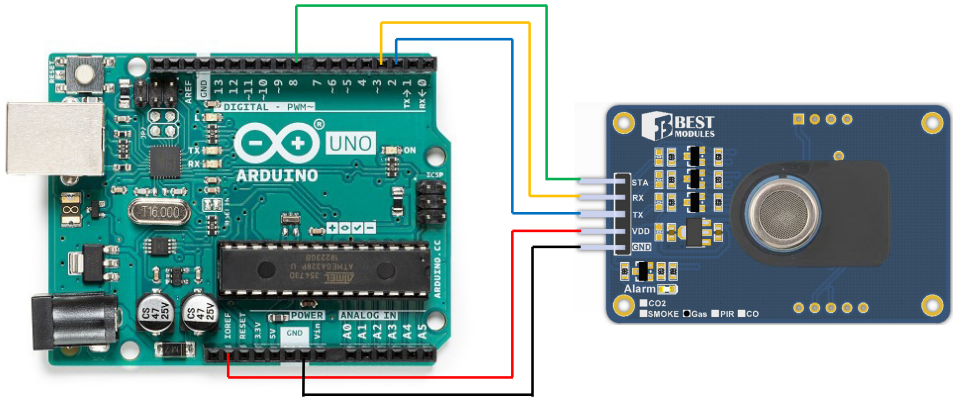
方式 2：添加 .ZIP 库，需提前下载 .ZIP 库

下载方法：打开倍创官方网站 (https://www.bestmodulescorp.com/bm22s3031-1.html) 文件目录下的 Arduino 范例程序 (BM22S3031-1 Library)。

添加 .ZIP 库：Arduino IDE → 项目 → 加载库 → 添加 .ZIP 库 ...



Arduino 范例



实物连接示意图

范例 1：readStatusPin

范例实现功能：通过查询 STA pin 状态来判断是否有报警

- 当模块无报警时，LED(D13) 闪烁周期 1 秒
- 当模块报警时，LED(D13) 闪烁周期 0.2 秒

1. 范例打开：Arduino IDE → 文件 → 示例 → Lib 选择 (BM22S3031-1) → 选择范例 (readStatusPin)
2. 示例说明：

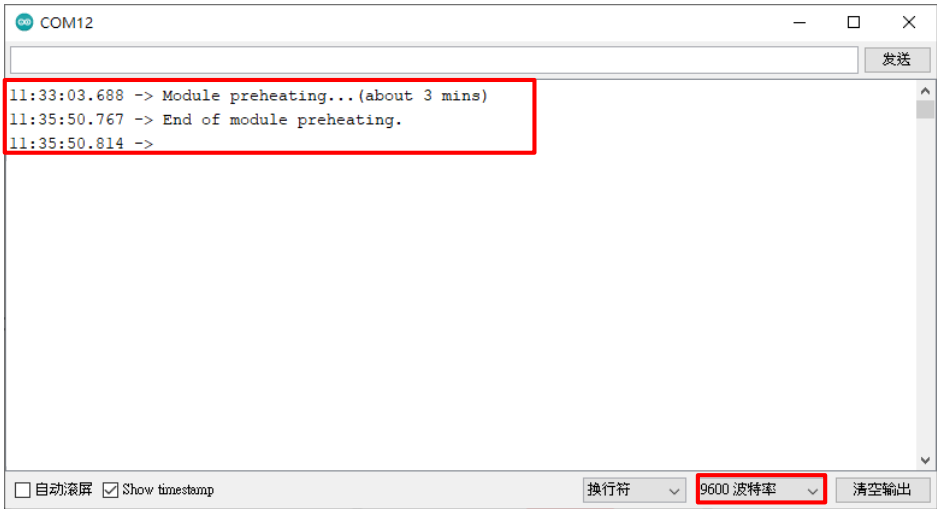
a. 构建对象

```
#include < BM22S3031-1.h> // 调用 BM22S3031-1 库

#define LED (13)    // LED 控制引脚：13

/* 创建对象 & 设置 Softeware serial 引脚 */
BM22S3031_1 gas(8, 2, 3); // Softeware serial: 8->STA, 2->RX, 3->TX
void setup()
{
/* 初始化模块 */
  gas.begin(); // Softeware serial 初始化（波特率：9600bps），设置 8 号引脚
               // 为输入模式

/* 配置串口监视器 */
  Serial.begin(9600); // Serial 初始化，波特率 9600bps

/*LED 控制引脚初始化 */
  pinMode(LED, OUTPUT);
  digitalWrite(LED, LOW);

/* 等待模块预热结束 */
  Serial.println("Module preheating...(about 3 mins)");
  gas.preheatCountdown(); // 等待模块预热结束
  Serial.println("End of module preheating.");
  Serial.println();
}
```

b. 根据 STATUS 状态 (模块有无报警) 改变 LED 闪烁周期

```
void loop()
{
  if (gas.getSTATUS() == 0)
  {
/* 无报警，LED 闪烁周期为 1 秒 */
    digitalWrite(LED, HIGH);
    delay(500);
    digitalWrite(LED, LOW);
    delay(500);
  }
  if (gas.getSTATUS() == 1)
  {
/* 报警，LED 闪烁周期为 0.2 秒 */
    digitalWrite(LED, HIGH);
    delay(100);
    digitalWrite(LED, LOW);
    delay(100);
  }
}
```

3. 打开串口监视器，波特率选择 9600，串口监视器显示如下；提示预热结束后观察 LED 闪烁情况。



范例 2：readGasValue

范例实现功能：实现功能：接收模块每秒自动输出的信息，并将 Gas 的相关信息打印到串口监视器

1. 范例打开：Arduino IDE → 文件 → 示例 → Lib 选择 (BM22S3031-1) → 选择范例 (readGasValue)
2. 示例说明：

a. 构建对象

```
#include < BM22S3031-1.h> // 调用 BM22S3031-1 库

/* 创建数组、变量用于存放数据 */
uint8_t moduleInfo[34] = {0};
uint16_t ADValue, gasValue, gasAlarmThreshold;

/* 创建对象 & 设置 Softeware serial 引脚 */
BM22S3031_1 gas(8, 2, 3); // Softeware serial: 8->STA, 2->RX, 3->TX
void setup()
{
/* 初始化模块 */
  gas.begin(); // Softeware serial 初始化 ( 波特率 : 9600bps), 设置 8 号引
               // 脚为输入模式

/* 配置串口监视器 */
  Serial.begin(9600); // Serial 初始化 , 波特率 9600bps

/* 等待模块预热结束 */
  Serial.println("Module preheating...(about 3 mins)");
  gas.preheatCountdown(); // 等待模块预热结束
  Serial.println("End of module preheating.");
  Serial.println();
  delay(1200);
}
```

b. 接收模块自动发送的数据

```
void loop()
{
  if (gas.isInfoAvailable() == true) // 轮询是否接收到模块发出的数据
  {
    gas.readInfoPackage(moduleInfo); // 读取模块发送的数据到moduleInfo[]
    printInfo(); // 打印模块的部分信息
  }
}
```

c. 根据接收到的数据，打印部分信息

```
void printInfo()
{
  /* 打印燃气报警阈值 (PPM) */
  Serial.print("Gas alarm threshold: ");
  gasAlarmThreshold = (moduleInfo[23] << 8) + moduleInfo[24];
  Serial.print(gasAlarmThreshold);
  Serial.println(" PPM");

  /* 打印当前燃气浓度 (PPM) */
  Serial.print("Gas concentration: ");
  gasValue = (moduleInfo[9] << 8) + moduleInfo[10];
  Serial.print(gasValue);
  Serial.println(" PPM");

  /* 打印实时 A/D 值 */
  Serial.print("Gas AD Value: ");
  ADValue = (moduleInfo[5] << 8) + moduleInfo[6];
  Serial.println(ADValue);

}
```

3. 打开串口监视器，波特率选择 9600；串口监视器显示如下：