NOVOSENSE
纳芯微电子

NSPGD1 Series
带气嘴集成式表压传感器

Datasheet (CN) 1.3

产品概述

NSPGD1系列集成压力传感器是纳芯微针对家电市场，推出的经过校准的表压传感器产品。该系列产品采用高性能信号调理芯片对先进的MEMS压阻芯体进行温度和压力校准和补偿，保证产品优异性能和可靠性的同时将两颗芯片进行集成封装，带气嘴的DIP8封装形式方便客户焊接和使用。调理过的产品可在0~70°C范围内提供精度范围内的标准输出，免去了客户对传感器进行校准的门槛，加速产品研发和量产的进程。NSPGD1系列集成压力传感器可选量程-10kPa至10kPa，适合于压力敏感元件结构材料相兼容的非腐蚀性气体的差压检测，特别适用于家电非接触式液位检测、小家电、医疗保健等领域，同时也适用于工业及物联网等领域。该系列支持模拟输出/数字I²C输出，以及特有的频率输出方式，应用更加灵活。

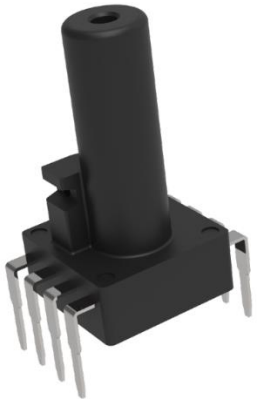
主要特性

- 可定制量程-10kPa~10kPa
- 温度范围 0~70°C
- 模拟/数字 I²C 输出工作电压范围 3V~5.5V
  频率输出工作电压范围 5±0.2V
- 可选模拟电压输出
- 可选 24bit I²C 输出
- 可选频率输出
- 高稳定性
  25°C 精度优于±1%F.S.
  0~70°C 模拟/数字 I²C 输出精度优于±1.5%F.S.，频率输出型±2.5% F.S.
- 正面进气方式，不易堵塞
- 内部防潮处理

应用

- 洗衣机
- 洗碗机、净水器
- 气垫床、按摩椅
- 智能血压计 、制氧机
- 工业控制
- 物联网压力检测

外形图



目录

1. 封装及引脚定义

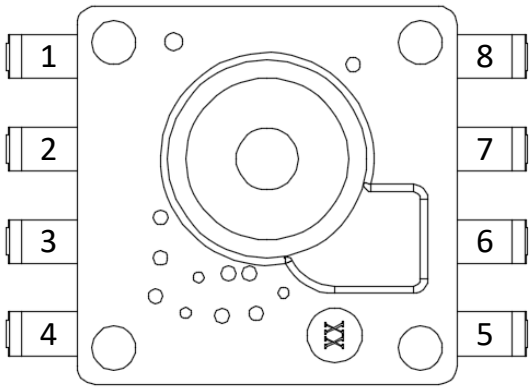


图 1.1 NSPGD1 Series 封装

表 1.1 模拟/频率输出引脚定义及功能描述

引脚编号	*符号*	*功能*
1	NC	悬空
2	VDD	电源输入端
3	GND	地
4	VOUT/FREQ	模拟电压输出/频率输出端
5	NC	悬空
6	NC	悬空
7	NC	悬空
8	NC	悬空

表 1.2 数字 I²C 输出引脚定义及功能描述

引脚编号	*符号*	*功能*
1	NC	悬空
2	NC	悬空
3	SDA	I²C 通讯数据线
4	SCL	I²C 时钟数据线
5	VDD	电源输入端
6	GND	地
7	NC	悬空
8	NC	悬空

2. 最大额定参数

参数	标示	最小值	典型值	最大值	单位	备注
VDD电压	VDD_{max}	-0.3		6.5	V	
模拟引脚电压		-0.3		VDD+0.3	V	
模拟引脚电流				25	mA	
过载压力	P_{proof}		3X		kPa	3倍最大工作压力
爆破压力	P_{burst}		5X		kPa	5倍最大工作压力
存储温度	T_{stg}	-30		85	°C	
ESD防护	HBM		2		kV	

3. 推荐工作范围

参数项	标示	最小值	典型值	最大值	单位	备注
VDD电压	VDD_{max}	3	5	5.5	V	模拟电压、数字I²C输出
		4.8	5	5.2		频率输出
工作压力	P_{amb}	-10		10	kPa	
工作温度	T_{opr}	0		70	°C	

4. 技术规格

4.1 电气参数

参数项	标示	最小值	典型值	最大值	单位	备注
上电复位	VDD_{POR}		2		V	
典型工作电流	I_{avdd}		2.5		mA	工作电流
				200	nA	数字输出模式休眠电流
模拟输出综合精度	ACC		±1		%F.S.	@25°C
			±1.5		%F.S.	0~70°C 模拟电压、数字 I^2C 输出
			±2.5		%F.S.	0~70°C 频率输出
ADC分辨率	RES_{RAW}		24		Bits	
电源抑制比	PSRR	90	120		dB	
DAC分辨率			12		Bits	
输出驱动负载电阻	R_{load}	1			kΩ	电压输出模式
输出驱动负载电容	C_{load}			15	nF	电压输出模式
阶跃响应时间	T_{RESP}		10		ms	
电压输出噪声	V_{rms}			0.5	mV	
EEPROM数据保持	T_{live}	10			a	@125°C

4.2 I^2C 通讯电气特性

参数项	标示	最小值	典型值	最大值	单位	备注
时钟频率	f_{scl}			400	kHz	
时钟低脉冲维持时间	t_{LOW}	1.3			μs	
时钟高脉冲维持时间	t_{HIGH}	0.6			μs	
SDA建立时间	t_{SUDAT}	0.1			μs	
SDA保持时间	t_{HDDAT}	0.0			μs	
每次开始时的建立时间	t_{SUSTA}	0.6			μs	
开始条件保持时间	t_{HDSTA}	0.6			μs	
停止时间建立时间	t_{SUSTO}	0.6			μs	
两次通讯之间间隔时间	t_{BUF}	1.3			μs	

5. 功能描述

5.1 概述

NSPGD1 系列压力传感器通过 MEMS 压阻表压压力芯体作为压力敏感元件，该元件会输出一个与环境压力呈正比例关系的一个原始信号输出。内置 24 位 ADC 的调理芯片驱动该敏感元件，并对其原始信号进行放大、温度补偿、线性度补偿后输出一个与施加压力呈线性关系的输出信号。

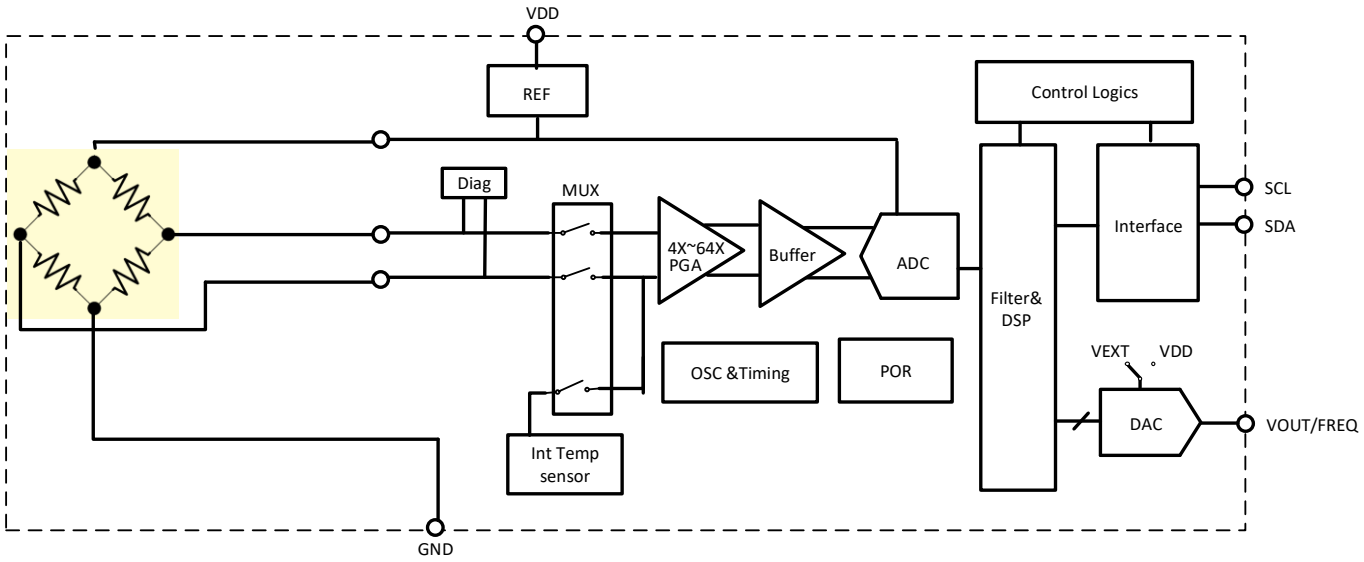


图 5.1 功能框图

5.2 模拟输出传递函数

$$VOUT = (A \times P + B) \times 5$$ @绝对电压输出，VDD=5V

$$VOUT = (A \times P + B) \times 3.3$$ @绝对电压输出，VDD=3.3V

$$VOUT = (A \times P + B) \times VDD$$ @比例电压输出

其中：1） VOUT 为输出电压，单位 V；

2）P 为待测表压压力，单位 kPa；

表 5.1 模拟输出典型对照表

示例料号	压力量程 kPa		输出电压 V		传递函数系数	
	P_L	P_H	O_L	O_H	A	B
NSPGD1F002RT02	0kPa	2kPa	0.1*VDD	0.9*VDD	0.4	0.1

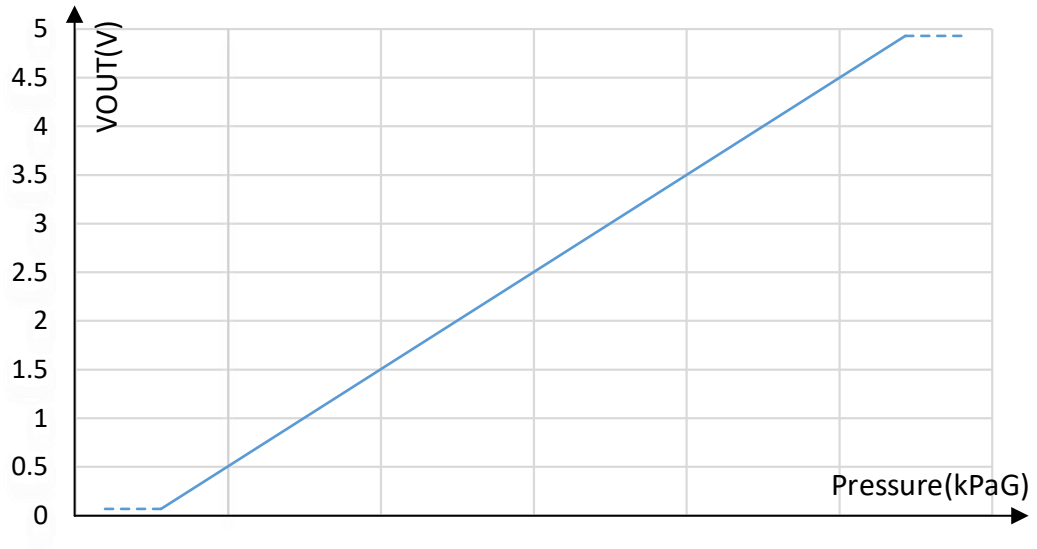


图 5.2 模拟输出示例

5.3 数字 I²C 输出传递函数

使用如下等式，可将压力值转换为数字输出寄存器值：

$$Code = (A \times P + B) * 8388607$$

其中 Code 为芯片寄存器值；P 为实际压力值，单位 kPa；

表 5.2 数字 I²C 输出典型对照表

示例料号	压力量程 kPa		数字输出 code		传递函数系数	
	P_L	P_H	O_L	O_H	A	B
NSPGD1F006DT04	0kPa	6kPa	838861	7549746	0.13333	0.1

5.3.1 寄存器定义

地址	位地址	寄存器名称	默认值	描述
0x30	7 – 4	Reserve	4'b0000	写入0x0A开始单次压力采集模式，当寄存器值变为0x02时，单次采集完成；
	3	Sco	1'b0	
	2 – 0	Measurement_ctrl<2:0>	3'b000	
0x06	7 – 0	PDATA<23:16>	0x00	有符号数，2的补码：存储经过校准的压力传感器数据； Code = Data0x06*2^{16}+ Data0x07*2^8+ Data0x08；
0x07	7 – 0	PDATA<15:8>	0x00	
0x08	7 – 0	PDATA<7:0>	0x00	

结合 5.3 中 Code 至压力转换公式，可计算测量压力值；

举例：当 0x06、0x07、0x08 寄存器的值分别为 0x3F、0xFF、0xFF，代入 NSPGD1F006DT04 参数计算，Code = 4194303， P(kPa) = (4194303/8388607-B)/A，最终得到压力值约为 3kPa。

5.4 频率输出传递函数

$$\text{FREQ} = (A \times P + B) \times \text{F.S.}$$

注：1）FREQ 是输出频率，单位 kHz；

2）P 为压力值，表压，范围：P_L~P_H mmH2O；上述传递函数仅在压力范围内成立；

3）F.S.是频率输出满量程值，芯片可配置四种频率满量程，分别是 250kHz、125kHz、61.5kHz、31.25kHz；标定时会选择和最大输出频率值最接近且大于最大输出频率的满量程挡位；

4）传递函数为实际应用中的传递函数，涉及到的各参数适用于第 6 部分应用指南中的图 6.1。

表 5.3 频率输出典型对照表

示例料号	压力量程P		输出频率 FREQ		传递函数系数		输出满量程
	P_L	P_H	F_L	F_H	A	B	F.S.
NSPGD1F004FT12	0mmH2O	350mmH2O	3.125kHz	28.125kHz	0.002285714	0.1	31.25kHz

注：NSPGD1F004FT12 为示例料号，具体应用压力量程及输出频率范围可根据客户要求进行定制，以兼容传统机械式液位测量模组参数；

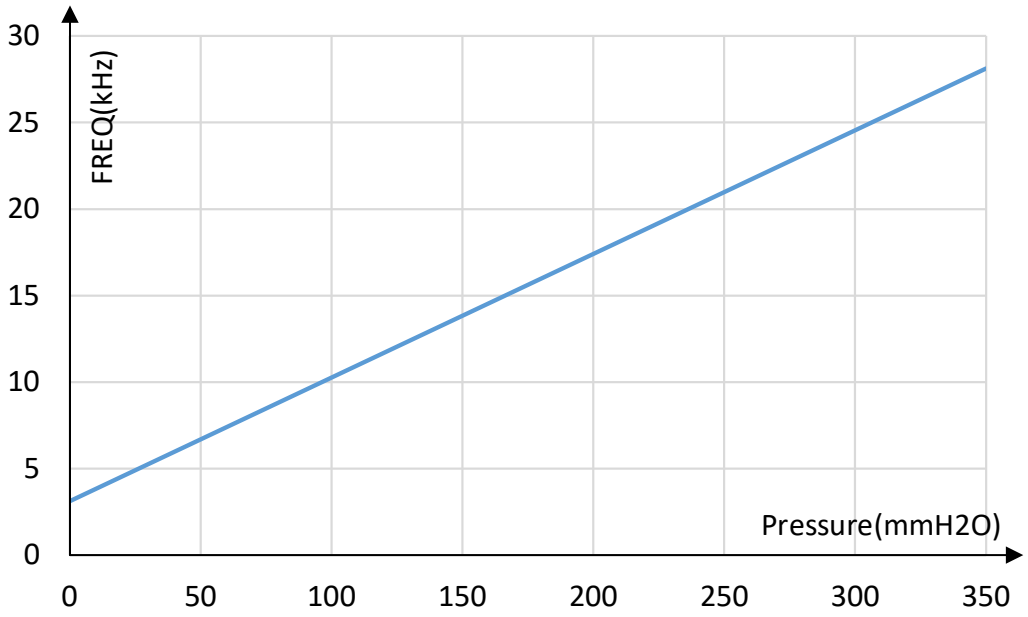


图 5.3 频率输出示例

以 NSPGD1F004FT12 在洗衣机液位测量应用为例，洗衣机液位量程 0~350mmH2O，对应输出频率 3.125kHz~28.125kHz，F.S.= 31.25kHz，以表中对应传递函数系数代入传递函数，可得到不同压力（水位）对应的频率输出。

5.5 I²C 通讯接口协议

I²C 总线使用 SCL 和 SDA 作为信号线。这两根线都通过上拉电阻连接到 VDD，不通信时都保持为高电平。该系列产品的 I²C 设备地址如下：

A7	A6	A5	A4	A3	A2	A1	W/R
1	1	1	1	1	1	1	0/1

表 5.3 I²C 地址

I²C 通讯协议有着特殊的开始(S)和终止 (P)条件。当 SCL 处于高电平同时，SDA 的下降沿标志数据传输开始。I²C 主设备依次发送从设备的地址（7 位）和读/写控制位。当从设备识别到这个地址后，产生一个应答信号并在第九个周期将 SDA 拉低。得到从设备应答后，主设备继续发送 8 位寄存器地址，得到应答后继续发送或读取数据。SCL 处于高电平，SDA 发生一个上升沿动作标志 I²C 通信结束。除了开始和结束标志之外，当 SCL 为高时 SDA 传输的数据必须保持稳定。当 SCL 为低时 SDA 传输的值可以改变。I²C 通信中的所有数据传输以 8 位为基本单位，每 8 位数据传输之后需要一位应答信号以保持继续传输。

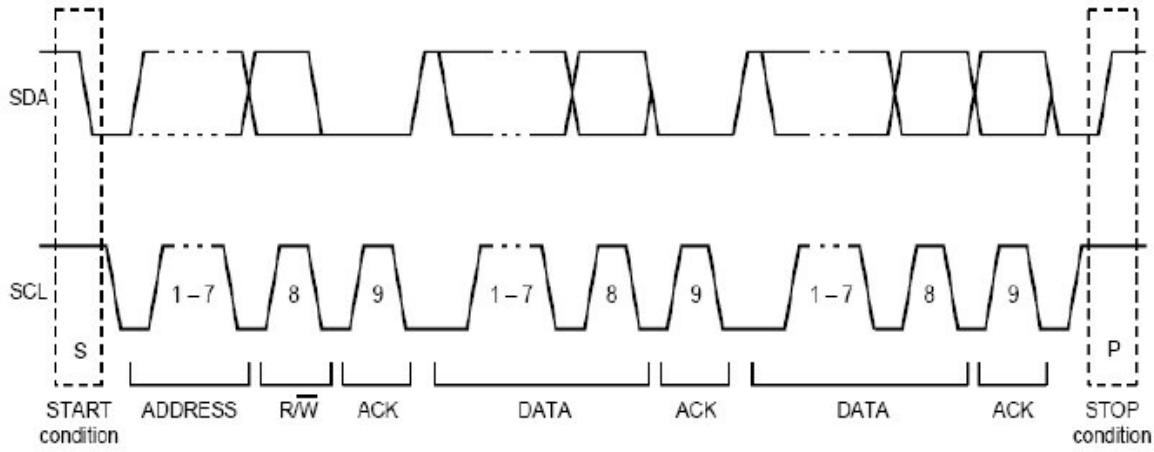


图 5.3 I²C 协议

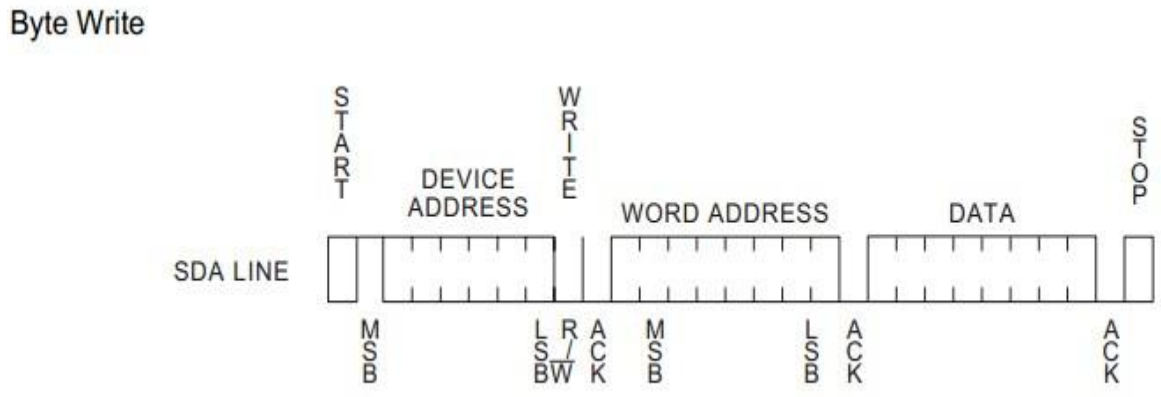


图 5.4 I²C 写时序

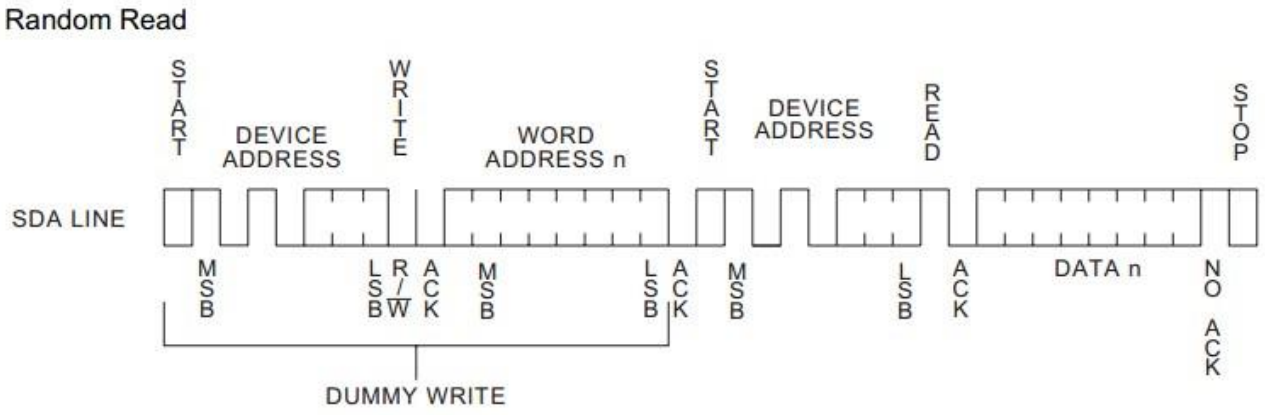


图 5.5 I²C 读时序

6. 应用指南

6.1 应用电路

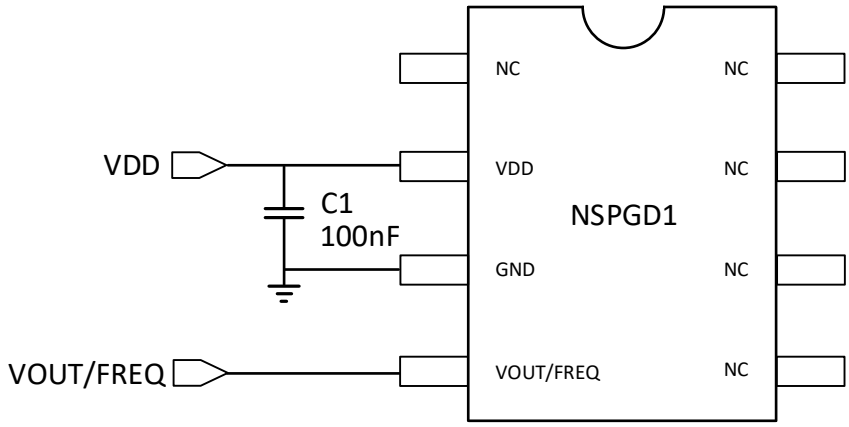


图 6.1 模拟/频率输出典型应用电路

注：对于ESD 更高要求的应用场合，推荐客户在VOUT 、GND 间以及VDD 、GND 间增加TVS 管，VOUT增加电阻；具体参见图6.2。

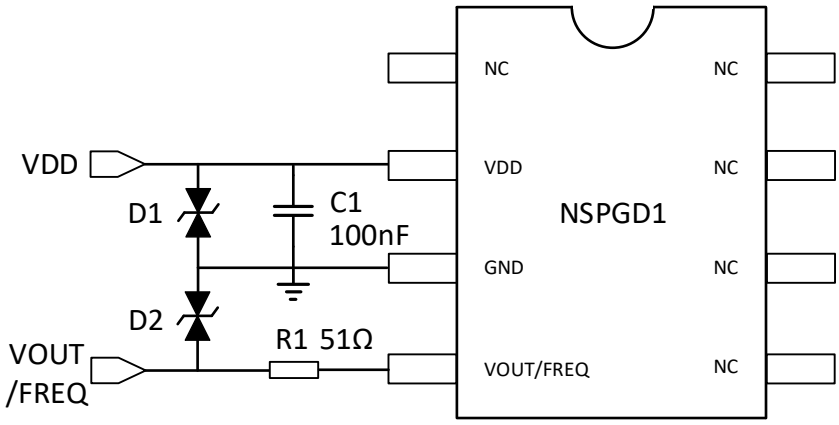


图 6.2 模拟/频率输出防护电路

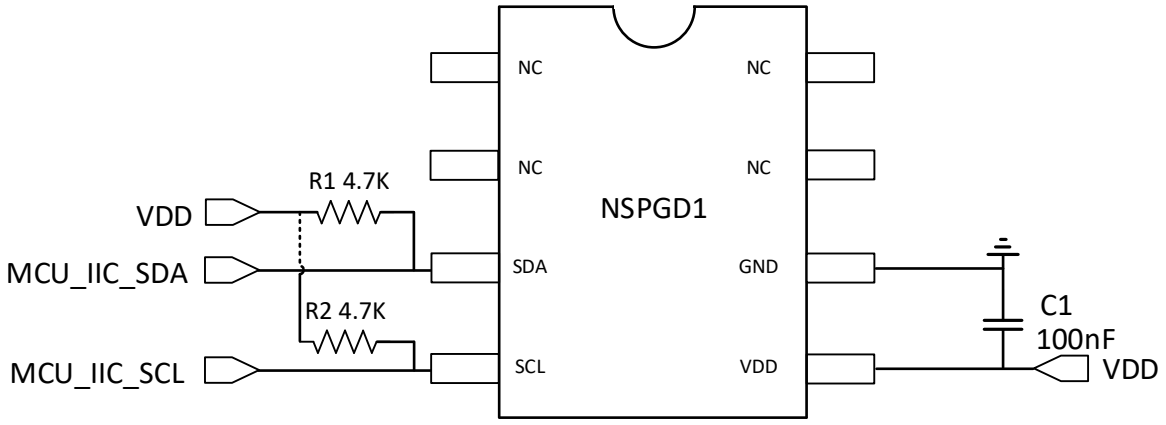


图 6.2 I²C 输出型典型应用电路图

7. 封装信息

7.1 产品封装尺寸

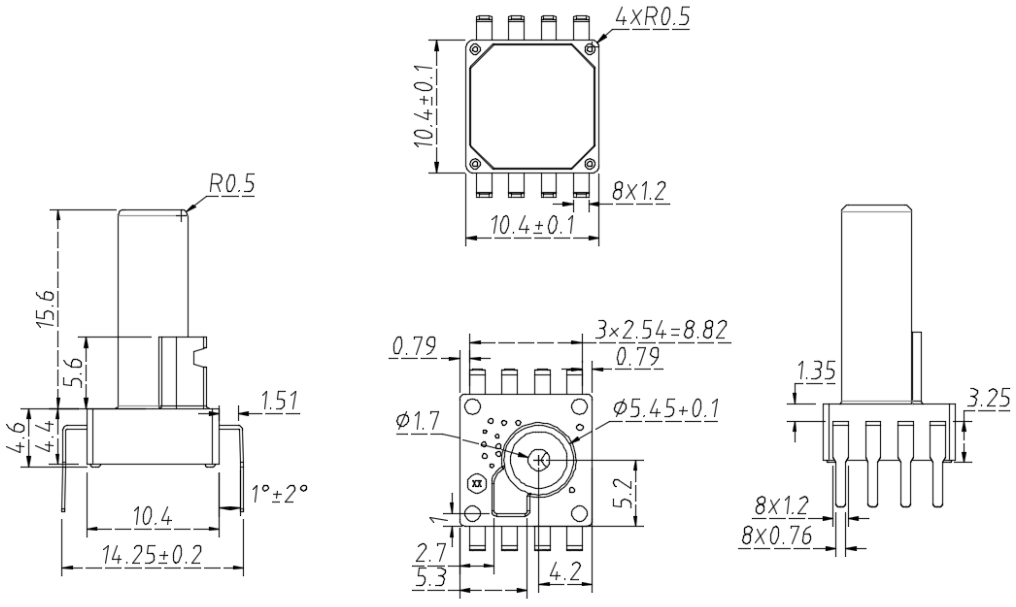


图 7.1 产品封装尺寸图 mm

7.2 焊盘建议尺寸

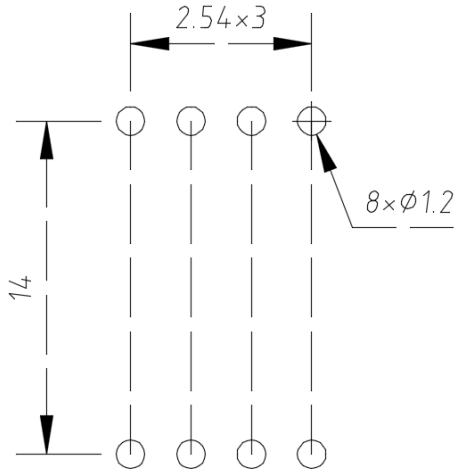
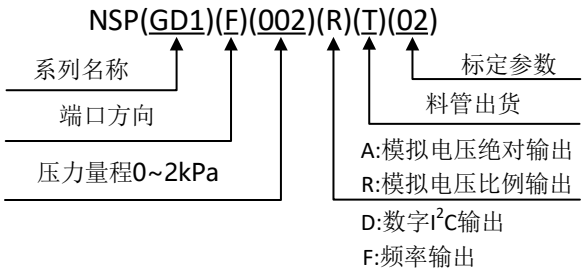


图 7.2 焊盘建议尺寸图 mm

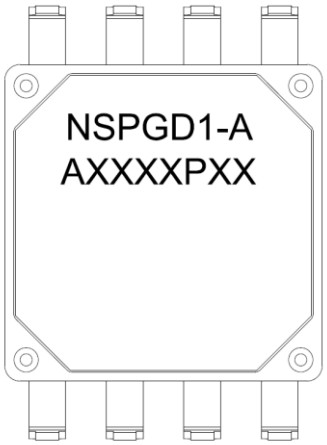
8. 订购信息

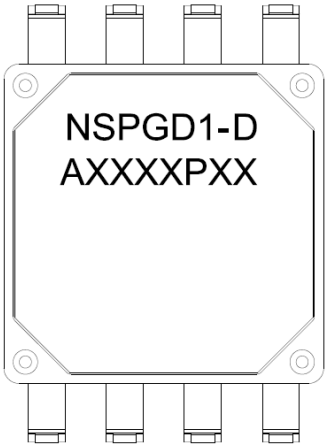
产品料号	输出类型	压力量程		输出范围		钳位信息		传递函数系数	
		P_L	P_H	O_L	O_H	V_{CL}	V_{CH}	A	B
NSPGD1F002RT02	比例电压	0kPa	2kPa	0.1*VDD	0.9*VDD	4.76%	94%	0.40000	0.10000
NSPGD1F004RT03	比例电压	0kPa	3.92kPa	0.2*VDD	0.984*VDD	0%	100%	0.20000	0.20000
NSPGD1F006DT04	I²C 数字	0kPa	6kPa	10%	90%	0%	100%	0.13333	0.10000
NSPGD1F003AT05	绝对电压	0mmH2O	325mmH2O	0.5V	4.5V	0%	100%	0.00246	0.10000
NSPGD1F005AT06	绝对电压	0mmH2O	450mmH2O	0.5V	4.5V	0%	100%	0.00178	0.10000
NSPGD1F010RT07	比例电压	0kPa	10kPa	0.1*VDD	0.9*VDD	0%	100%	0.08000	0.10000
NSPGD1F004FT12	频率输出	0mmH2O	350mmH2O	3.125kHz	28.125kHz	0%	100%	0.002285714	0.10000

命名规则:



9. 丝印信息





NSPGD1：压力传感器系列；

A：模拟输出类型；

D：数字输出类型；

AXXXXPXX ：批次号；

10. 包装信息

产品采用料管包装形式，每根料管 50EA；每盒 10 根料管，最小起订量 1000EA。

11. 修订历史

Revision	*Description*	*Date*
0.1	Initial Version.	2021/3/1
1.0	正式版本；	2021/3/2
1.1	内容更新；	2021/3/5
1.2	增加频率输出描述；	2021/11/12

Notes：

1. I²C 例程

```
void IIC_Init(void)
{
    SCL_H;
    SDA_H;
    SCL_W;
    SDA_W;
}

void IIC_Start(void)
{
    SDA_W;
    SCL_H;
    SDA_H;
    delay10us();
    SDA_L;
    delay10us();
}

void IIC_Stop(void)
{
    SCL_L;
    delay10us();
    SCL_H;
    SDA_W;
    SDA_L;
    delay10us();
    SDA_H;
    delay10us();
}

void IIC_ACK(void)
{
    SDA_W;
    SDA_L;
    SCL_H;
    delay10us();
    SCL_L;
}

void IIC_NACK(void)
{
    SDA_W;
    SDA_H;
    SCL_H;
```

```
    delay10us();
    SCL_L;
}

uchar IIC_Wait_ACK(void)
{
    int ErrTime=0;
    SDA_R;
    SCL_H;
    delay10us();
    while(Read_SDA)
    {
        ErrTime++;
        if(ErrTime>200)
        {
            IIC_Stop();
            return 1;
        }
    }
    SCL_L;
    SDA_W;
    SDA_L;
    delay10us();
    return 0;
}

void IIC_Send(uchar IIC_Data)
{
    uchar i;
    SDA_W;
    SCL_L;
    delay10us();
    for(i=0;i<8;i++)
    {
        if((IIC_Data&0x80)>>7)
            SDA_H;
        else
            SDA_L;
        IIC_Data<<=1;
        SCL_H;
        delay10us();
        SCL_L;
        delay10us();
    }
}

uchar IIC_Receive(uchar ACK)
{
```

```
    uchar i,Receive_Data=0x00;
    SDA_R;
    for(i=0;i<8;i++)
    {
        SCL_L;
        delay10us();
        SCL_H;
        Receive_Data<<=1;
        if(Read_SDA==1)
            Receive_Data++;
        else
            ;
        delay10us();
    }
    SCL_L;
    delay10us();
    if(ACK==0x01)
        IIC_ACK();
    else
        IIC_NACK();
    return Receive_Data;
}

void NSPGD1_Write_Byte(uchar WriteAddr,uchar WriteData)
{
    uchar flag;
    IIC_Start();
    IIC_Send(0xFE|0x00);
    IIC_Wait_ACK();
    IIC_Send(WriteAddr);
    IIC_Wait_ACK();
    IIC_Send(WriteData);
    IIC_Wait_ACK();
    IIC_Stop();
}

void NSPGD1 _Read_Byte(uchar ReadAddr, uchar *pBuffer)
{
    IIC_Start();
    IIC_Send(0xFE|0x00);
    IIC_Wait_ACK();
    IIC_Send(ReadAddr);
    IIC_Wait_ACK();
    IIC_Start();
    IIC_Send(0xFE|0x01);
    IIC_Wait_ACK();
    pBuffer[0]=IIC_Receive(0);
    IIC_Stop();
```

```
}

void NSPGD1 _Read_3Byte(uchar ReadAddr,uchar *pBuffer)
{
    IIC_Start();
    IIC_Send(0xFE|0x00);
    IIC_Wait_ACK();
    IIC_Send(ReadAddr);
    IIC_Wait_ACK();
    IIC_Start();
    IIC_Send(0xFE|0x01);
    IIC_Wait_ACK();
    pBuffer[0]=IIC_Receive(1);
    pBuffer[1]=IIC_Receive(1);
    pBuffer[2]=IIC_Receive(0);
    IIC_Stop();
}

    Void Main()
    {
        uChar PData[3]={0,0,0};
         IIC_Init();
        NSPGD1_Write_Byte(0x30,0x0A);
        Delay_3ms();
        NSPGD1 _Read_3Byte(0x06,PData);
    }
```