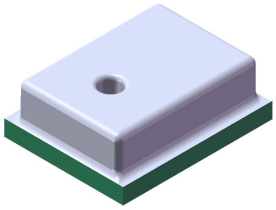
XGZP6818D PRESSURE SENSOR

FEATURES

- Wide Ranges: 0kPa ~ 100kPa···2500kPa(show in Pressure Range Example)
- Optional 3.3V ~ 5.5V Power Supply
- Absolute Pressure Type
- For Non-corrosive Gas or Air
- Calibrated Digital Signal(I2C Interface)(Refer to XGZP6818A for Analog signal)
- Multiple Working Methods
- High Resolution
- Temperature Accuracy: ±1°C



APPLICATIONS

- Medical device,Massage device, Hospital beds etc,.
- Consumer electronics,Inflator pump,Pneumatic etc,.
- Vacuum measurement, Industrial controls etc,.

INTRODUCTION

XGZP6818D is a perfect silicon pressure sensor offering a ratiometric I2C interface for reading pressure over the specified full scale pressure span.

The XGZP6818D incorporates a silicon piezoresistive pressure sensor die and an interior Application Specific Integrated Circuit(ASIC) in a small COB package.

The XGZP6818D is fully calibrated and temperature compensated for offset, sensitivity, temperature and non-linearity, so XGZP6818D pressure sensor satisfy the prefect repeatability, linearity, stability and sensibility, which can be applied directly in medical care&health, home appliances, consumer electronic, industry, automotive and other pneumatic devices etc.

XGZP6818D pressure sensor is for high volume application at an affordable cost but perfect performance.

Customized calibrations (e.g.pressure range etc.) are available

PERFORMANCE PARAMETER

Unless otherwise specified, measurements were taken with a a temperature of 25±1℃ and humidity ranging from 25％ ~ 85%RH.

CHARACTERISTIC		MIN.	TYP.	MAX	UNIT
Available Pressure Range①			0 ~ 100···2500		kPa
Power Supply②		3.3		5.5	Vdc
Current Consumption	Operating Current	-	1.5	-	mA
	Idle Current	-	1	-	uA
Output Resolution③	Pressure		21		Bit
	Temperature		16		Bit
Total Accuracy	Pressure ≤350kPa	-	-	±2	%FSS
	Pressure > 350kPa	-	-	±2.5	%FSS
Temperature Accuracy		-1		1	℃
Long Term Stability(1000 hr, 25°C)		-	-	±0.5	%FSS
Over Pressure④	Pressure ≤350kPa	-	2X	-	FSS
	Pressure > 350kPa	-	1.5X	-	FSS
Burst Pressure④	Pressure ≤350kPa	-	3X	-	FSS
	Pressure > 350kPa	-	2X	-	FSS
Compensation Temperature		0	-	60	℃
Operating Temperature		-30	-	100	℃
Storage Temperature		-40	-	125	℃
ESD Protection(Human Body Mode)		-	±2000	-	V
Response Time(combined conversion mode)		-	5	-	mS

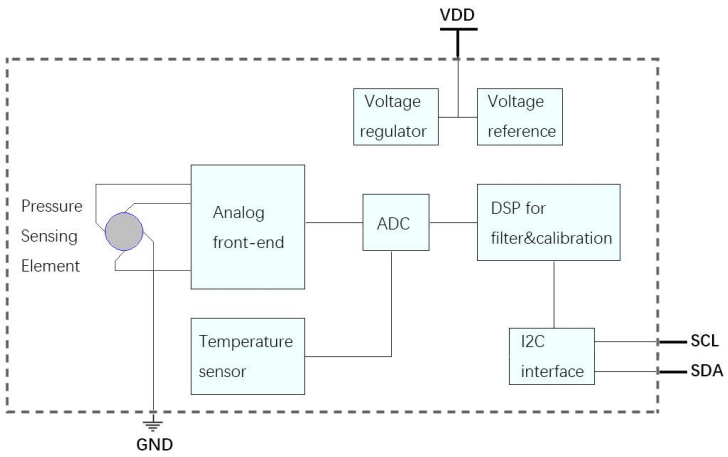
① The range cover all pressure ranges as shown as “PRESSURE RANGE EXAMPLE” list.

② Overload voltage(6.5Vdc above) or current(5mA above) may burn the IC and cause the sensor failure thoroughly.

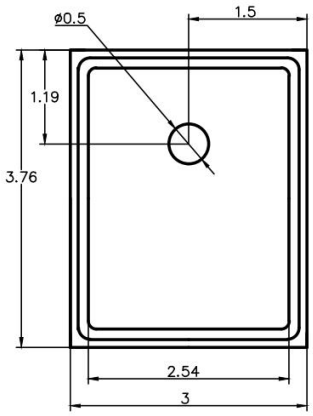
③ The highest data bit as the signed number.

④ The indicated value is widespread value, contact CFSensor for more information on specific pressure range.

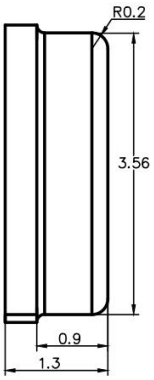
BLOCK DIAGRAM



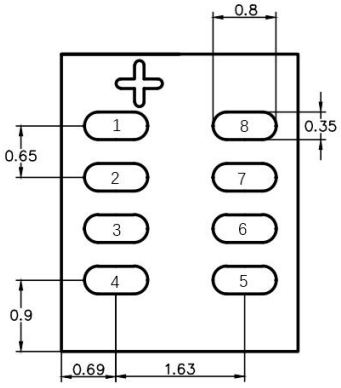
DIMENSION (Unit:mm Unspecified Tolerances:±0.1mm)



Front View



Side View

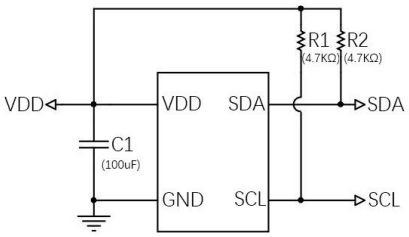


Bottom View

PIN DEFINITION

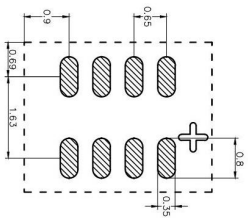
PIN1	PIN3	PIN4	PIN7	PIN8	PIN2/5/6
GND	SDA	SCL	GND	VDD	N/C

SCL	The clock signal
NC	Do not connect to external circuitry or ground
GND	Ground
VDD	Voltage supply
SDA	Data signal(Send& Receive)



Recommended Application Circuit

FOOTPRINT(Unit:mm)



NOTE: FOOTPRINT LAYOUT FOR REFERENCE

CONTACT CFSENSOR FOR ABOVE FILE IF REQUIRED

ORDER GUIDE

XGZP6818 D 00100 KP A

- A: Pressure Type(A:Absolute)
- KP: Pressure Unit(MP: MPa KP: kPa B:bar)
- 00100: Pressure Range(Min.P:00- Max.P:999)
- D: Output Signal(A: Analog D:I2C)
- XGZP6818: Series

Note: Custom requirement or parameter, please consult CFSensor and comment custom code herewith Part number.

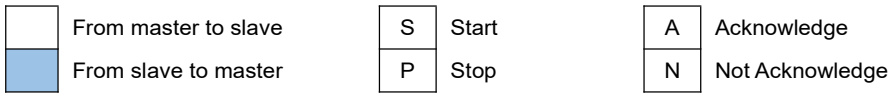
PRESSURE RANGE EXAMPLE

Notes: Unit conversion: 1000hPa=1000mbar≈750mmHg≈100kPa≈14.5PSI≈10mH_2O≈1bar=0.1MPa;

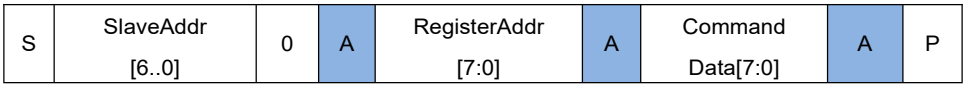
Pressure Range (kPa)	Pressure Range (by other units)	Part Number
0 ~ 100kPa	0 ~ 1bar /0 ~ 14.5PSIA	XGZP6818D00100KPA
30 ~ 110kPa	4 ~ 29PSIA	XGZP6818D30110KPA
0 ~ 700kPa	0 ~ 7bar /0 to 100 psi	XGZP6818D00700KPA
0 ~ 1000kPa	0 ~ 10bar /0 to 1MPa	XGZP6818D00010BA
0 ~ 1400kPa	0 ~ 14bar /0 to 1.4MPa	XGZP6818D00014BA
0 ~ 1500kPa	0 ~ 15bar /0 to 1.5MPa	XGZP6818D00015BA
★Above P/N is example only, consult CFSensor whether required pressure range is under normal production before place order.		

I2C INTERFACE

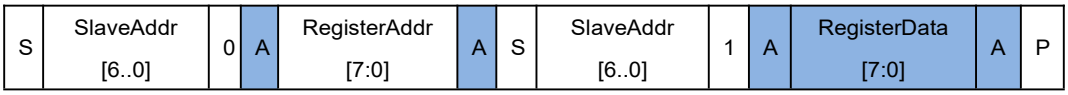
The I2C bus uses SCL and SDA as signal lines, both of which are connected to VDD through pull-up resistors (typ.value: 2.2K) and remain high level when not communicating. I2C device factory setting slave address: 0X58H



Write Operation



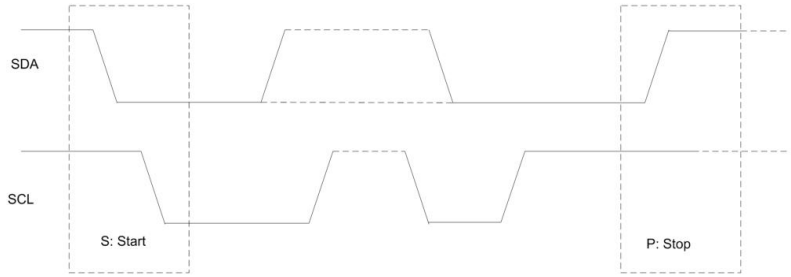
Read Operation



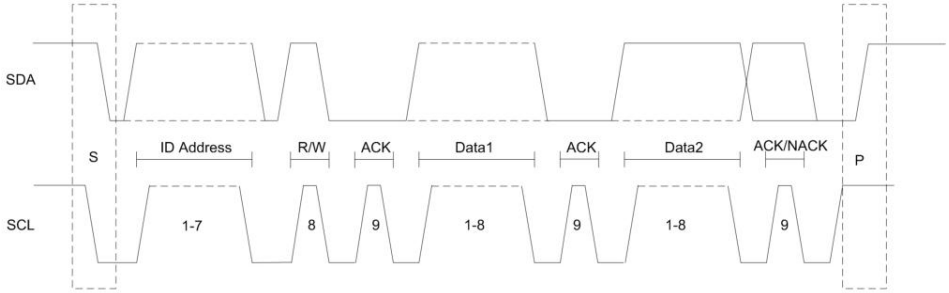
I2C TIME DIAGRAM

The I2C interface protocol has special bus signal conditions. Start (S), stop (P) and binary data conditions are shown below. At start condition, SCL is high and SDA has a falling edge. Then the slave address is sent. After the 7 address bits, the direction control bit R/W selects the read or write operation. When a slave device recognizes that it is being addressed, it should acknowledge by pulling SDA low in the ninth SCL (ACK) cycle.

At stop condition, SCL is also high, but SDA has a rising edge. Data must be held stable at SDA when SCL is high. Data can change value at SDA only when SCL is low.



I2C PROTOCAL



GENERAL REGISTER DESC.

Add.	Desc.	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Default
0x00	ID	R	ID<7:0>								0x58
0x01	Chip_Control	R/W	reserved<7:6>		data_Ready	reserved	data_out	measurement_ctrl	Active<1:0>		0x37
0x02	CFG_OSR	R/W	OSR_T<7:5>			OSR_P<4:2>			MODE[1:0]		OTP
0x03	CFG_MEAS	R/W	reserved<7:6>		T_SB[5:3]			PT_R[2:0]			OTP
0x04	P_data	R	Data out<23:16>								0x00
0x05	P_data	R	Data out<15:8>								0x00
0x06	P_data	R	Data out<7:0>								0x00
0x07	T_data	R	Temp out<15:8>								0x00
0x08	T_data	R	Temp out<7:0>								0x00
0x24	CFG_OPER	R/W	reserved<7:1>							DAC_EN	

Reg0x00: I2C device address, the default address is 0x58H.

Reg0x01: Chip launch control register, write "0x01" to launch the chip.

Reg0x02: The oversampling control register is described in detail as follows:

Bit #	Name	Description	
0x02.[1:0]	MODE[1:0]	00b: Sleep mode	
		01b: Normal mode	
		10b: One shot mode	
		others: Normal mode, cyclic measurement	
0x02.[4:2]	OSR_P[2:0]	Oversampling rate of pressure measurement	
		000b: over sampling x 256 001b: over sampling x 512 010b: over sampling x 1024 011b: over sampling x 2048	100b: over sampling x 4096 101b: over sampling x 8192 110b: over sampling x 16384 111b: over sampling x 32768
0x02.[7:5]	OSR_T[2:0]	Oversampling rate of pressure measurement	
		000b: over sampling x 256 001b: over sampling x 512 010b: over sampling x 1024 011b: over sampling x 2048	100b: over sampling x 4096 101b: over sampling x 8192 110b: over sampling x 16384 111b: over sampling x 32768

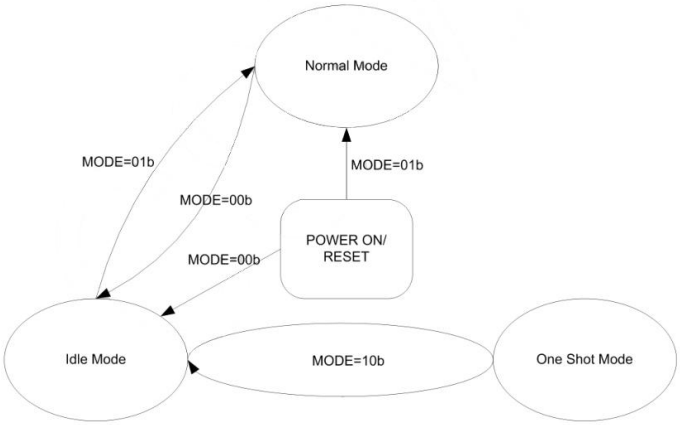
Reg0x03: CFG_MEAS (Measurement Command Register), the detailed description is as follows:

Bit #	Name	Description	
0x03.[2:0]	PT_R[2:0]	Pressure/Temperature measurement ratio in normal mode	
		000b: 64/1 001b: 32/1 010b: 16/1 011b: 8/1	100b: 4/1 101b: 1/1 Others: 128/1
0x03.[5:3]	T_SB[2:0]	Standby period setting in normal mode	
		000b: 0ms 001b: 62.5ms 010b: 125ms 011b: 250ms	100b: 500ms 101b: 750ms 110b: 1000ms 111b: 2000ms
0x03.[7:6]	reserved	reserve	

Reg0x04-Reg0x06: Pressure data register

Reg0x07-Reg0x08: Temperature data register

Working Methods Conversion



READ OPERATION

Read the data in accordance with the following instruction sequence:

1. VDD is powered on

2. First write 0x01 to the 0x01 address to start the chip

3. After a delay of 20ms, 5 bytes can be read continuously from 0x04 (ASIC will automatically refresh the data)

4. The first 3 bytes are air pressure data, and the calculation steps of the actual pressure value P are:
   Sum = (0x04 value * 2^16 + 0x05 value * 2^8 + 0x06 value),
   If sum < 8388608, P=sum / 2^21*(PMAX - PMIN) + PMIN(Unit: Pa)
   If sum ⩾8388608, P=(sum-2^24)/2^21*(PMAX - PMIN) + PMIN(Unit: Pa)
   Note: Corresponding relation between Pressure range and Calibration Parameter

Pressure Range	Calibration Parameter	
	PMIN	PMAX
30~110kPa	30000	110000
0~200kPa	20000	200000
0~700kPa	100000	700000
0~1000kPa	100000	700000
0~1400kPa	100000	700000

5. The last 2 bytes are the temperature data. The calculation steps of the actual temperature value Final_T are:
   RAW_T = 0x07 value * 2^8 + 0x08 value;
   If RAW_T > 32768, Inter_T = RAW_T - 65536; otherwise, Inter_T = RAW_T.
   The MCU continuously reads 2 byte values from the 0x20 address of the sensor, which is used for subsequent conversion of the real temperature. Set: the value of address 0x20 is read as byte1, and the value of address 0x21 is read as byte2.

Among them, bit[6:0] of byte1 is the absolute value of EOFF, bit[7] is the sign bit, when bit[7]=1, byte1 = -EOFF, and when bit[7]=0, Byte = EOFF. After the conversion of the absolute value and the sign of byte1, the value of byte1 is one of +/- 4096, +/- 8192, and +/- 16384.

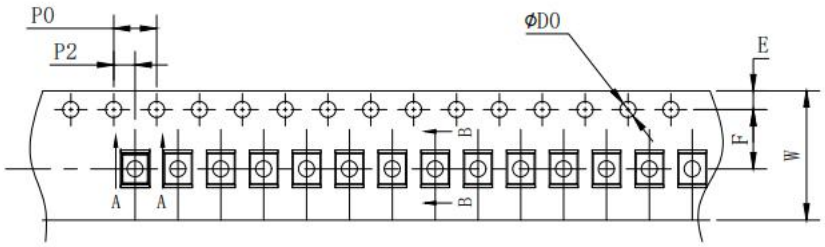
After the value of byte2 / 10, the Shift_N value is obtained, and the Shift_N value is one of 7, 6, and 5.5.

The final actual temperature value Final_T = (Inter_T - byte1) / 2 ^ Shift_N + 25.

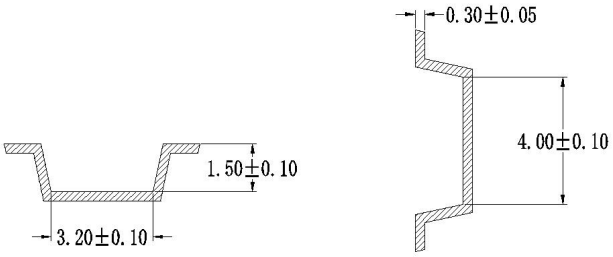
PACKING INFORMATION

Tape&Reel(unit:mm)

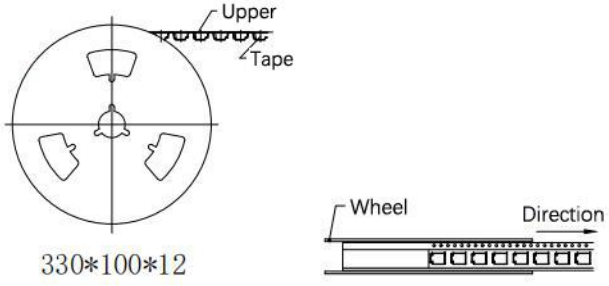
Standard Quantity/Reel: 3,000 pcs



Tape



Side view



Reel

Note: The packing method for less quantity than Standard Quantity/Reel may be not quite same with above.

OVERALL NOTES

Unless otherwise specified, following notes are general attention or presentation for all products from CFSensor.

Mounting

The following steps is for transmitting the air pressure to sensor after sensor soldering on PCB.

▼ For some sensors that come with inlet tube, select the flexible pipe to suit the pressure inlet that is firm enough to prevent the pressure leaks.

▼ Atmosphere hole (for Gauge type sensors) and Inlet pipe/hole can't be blocked with gel or glue etc,..

▼ Avoiding excessive external force operation

Soldering

Due to its small size, the thermal capacity of the pressure sensor is low. Therefore, take steps to minimize the effects of external heat. Damage and changes to characteristics may occur due to heat deformation. Use a non-corrosive resin type of flux. Since the pressure sensor is exposed to the atmosphere, do not allow flux to enter inside.

▼ Manual soldering

⊙Raise the temperature of the soldering tip between 260 and 300°C/500 and 572°F (30 W) and solder within 5 seconds.

⊙The sensor output may vary if the load is applied on the terminal during soldering.

⊙Keep the soldering tip clean.

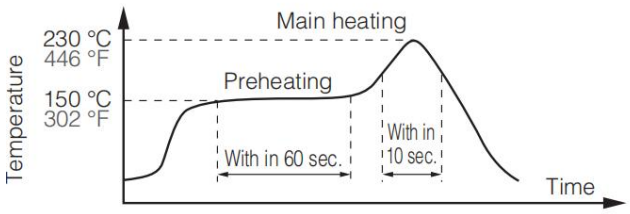
▼DIP soldering (DIP Terminal)

⊙Keep the temperature of the DIP solder tank below 260°C/500 and solder within 5 seconds.

⊙To avoid heat deformation, do not perform DIP soldering when mounting on the PCB which has a small thermal capacity.

▼Reflow soldering (SMD Terminal)

⊙The recommended reflow temperature profile conditions are given below.



⊙Self alignment may not always work as expected, therefore, please carefully note the position of the terminals and pattern.

⊙The temperature of the profile is assumed to be a value measured with the PCB of the terminal neighborhood.

⊙Please evaluate solderability under the actual mounting conditions since welding and deformation of the pressure inlet port may occur due to heat stress depending on equipments or conditions.

▼ Rework soldering

⊙Complete rework at a time.

⊙Use a flattened soldering tip when performing rework on the solder bridge. Do not add the flux.

⊙Keep the soldering tip below the temperature described in the specifications.

▼ Avoid drop and rough handling as excessive force may deform the terminal and damage soldering characteristics.

▼ Keep the circuit board warpage within 0.05 mm of the full width of the sensor.

▼ After soldering, do not apply stress on the soldered part when cutting or bending the circuit board.

▼ Prevent human hands or metal pieces from contacting with the sensor terminal. Such contact may cause anomalous outlets as the terminal is exposed to the atmosphere.

▼ After soldering, prevent chemical agents from adhering to the sensor when applying coating to avoid insulation deterioration of the circuit board.

Connecting

▼ Correctly wire as in the connection diagram. Reverse connection may damage the product and degrade the performance.

▼ Do not use idle terminals(N/C) to prevent damages to the sensor.

Cleaning

▼ Since the pressure sensor is exposed to the atmosphere, do not allow cleaning fluid to enter inside from atmosphere hole (for Gauge type sensors) and inlet pipe.

▼ Avoid ultrasonic cleaning since this may cause breaks or disconnections in the wiring.

Environment

▼Please avoid using or storing the pressure sensor in a place exposed to corrosive gases (such as the gases given off by organic solvents, sulfurous acid gas, hydrogen sulfides, etc.) which will adversely affect the performance of the pressure sensor chip.

▼Since this pressure sensor itself does not have a water-proof construction(even available media can be liquid), please do not use the sensor in a location where it may be sprayed with water, etc.

▼Avoid using the pressure sensors in an environment where condensation may form. Furthermore, its output may fluctuate if any moisture adhering to it freezes.

▼The pressure sensor is constructed in such a way that its output will fluctuate when it is exposed to light. Especially when pressure is to be applied by means of a transparent tube, take steps to prevent the pressure sensor chip from being exposed to light.

▼Avoid using pressure sensor where it will be susceptible to ultrasonic or other high-frequency vibration.

▼Keeping the sensors sealed in static shielding bags with an oxygen-free condition and use the sensor as soon as possible once unfold the package, because the sensors' PINs may be oxidated a bit under atmosphere environment(slight oxidation wouldn't affect soldering and performance)

More Precautions

▼That using the wrong pressure range or mounting method may result in accidents.

▼The only direct pressure medium you can use is non-corrosive gas or air as illuminated above(Note: some sensors are compatible with liquid media). The use of other media, in particular, corrosive gases and liquid (organic solvent based, sulfurous acid based, and hydrogen sulfide based, etc.) or contains foreign substances will cause malfunction and damage. Please do not use them and check with CFSensor.

▼The pressure sensor is positioned inside the pressure inlet. Never poke wires or other foreign matter through the pressure inlet since they may damage the sensor or block the inlet. Avoid use when the atmospheric pressure inlet(only for Gauge type pressure sensor) is blocked.

▼Use an operating pressure which is within the rated pressure range. Using a pressure beyond this range may cause damage.

▼Since static charge can damage the pressure sensor, bear in mind the following handling precautions.

⊙When storing the pressure sensor, use a conductive material to short the pins or wrap the entire sensor in aluminum foil. Common plastic containers should not be used to store or transport the sensor since they readily become charged.

⊙When using the pressure sensor, all the charged articles on the bench surface and the work personnel should be grounded so that any ambient static will be safely discharged.

▼Based on the pressure involved, give due consideration to the securing of the pressure sensor.

【 SAFETY NOTES 】

Using these sensors products may malfunction due to external interference and surges, therefore, please confirm the performance and quality in actual use. Just in case, please make a safety design on the device (fuse, circuit breaker, such as the installation of protection circuits, multiple devices, etc.), so it would not harm life, body, property, etc even a malfunction occurs. To prevent injuries and accidents, please be sure to observe the following items:

- The driving current and voltage should be used below the rated value.
- Please follow the terminal connection diagram for wiring. Especially for the reverse connection of the power supply, it will cause an accident due to circuit damage such as heat, smoke, fire, etc.
- In order to ensure safety, especially for important uses, please be sure to consider double safety circuit configuration.
- Do not apply pressure above the maximum applied pressure. In addition, please be careful not to mix foreign matter into the pressure medium. Otherwise, the sensor will be discarded, or the media will blew out and cause an accident.
- Be careful when fixing the product and connecting the pressure inlet. Otherwise, accidents may occur due to sensor scattering and the blowing out of the media.
- If the sensor come with sharp PIN, please be careful not to hurt your body when using it.

【 WARRANTY 】

The information in this sheet has been carefully reviewed and is believed to be accurate; however, no responsibility is assumed for inaccuracies. Furthermore, this information does not convey to the purchaser of such devices any license under the patent rights to the manufacturer. CFSensor reserves the right to make changes without further notice to any product herein. CFSensor makes no warranty, representation or guarantee regarding the suitability of its product for any particular purpose, nor does CFSensor assume any liability arising out of the application or use of any product or circuit and specifically disclaims any and all liability, including without limitation consequential or incidental damages. Typical parameters can and do vary in different applications. All operating parameters must be validated for each customer application by customer's technical experts. CFSensor does not convey any license under its patent rights nor the rights of others.

【 CONTACT 】

CFSensor
22F/14Bldg High-Tech Park High-Tech Area Wuhu P.R.C.241000
Tel/Fax:+86 18226771331 Email:INFO@CFSensor.com

North America II Europe II Southeast Asia II Middle East II Latin America

IIC Example Code (C51 Language)

```
#include <reg52.h>
#include <math.h>
#define DELAY_TIME 600   //Time-Delay Parameter  时延参数，可根据需要做适当调整
#define TRUE 1
#define FALSE 0
#define uchar unsigned char
#define uint unsigned int

float SPAN = 700;   //SPAN is the span of the sensor 传感器的量程 0~700KPa

sbit SCL = P1 ^ 7;   //IIC clock line   定义 IIC 总线时钟线
sbit SDA = P1 ^ 6;   //IIC clock line   定义 IIC 总线时钟线

//Time-Delay Function   时延函数
void DELAY(uint t)
{
    while (t != 0)
        t--;
}

void I2C_Start(void)   //IIC Start signal   发送 IIC 总线起始信号
{
    SDA = 1;   //SDA output high   SDA 输出高电平
     DELAY(DELAY_TIME);
    SCL = 1;   //SCL output high   SCL 输出高电平
    DELAY(DELAY_TIME);
    SDA = 0;   //SDA output low   SDA 输出低电平
    DELAY(DELAY_TIME);
    SCL = 0;   //SCL output low   SCL 输出低电平
    DELAY(DELAY_TIME);
}

void I2C_Stop(void)   //IIC Stop signal   发送 IIC 总线停止信号
{
    SDA = 0;
    DELAY(DELAY_TIME);
     SCL = 1;
    DELAY(DELAY_TIME);
    SDA = 1;
```

```
    DELAY(DELAY_TIME);
    SCL = 0;
    DELAY(DELAY_TIME);
}

void SEND_0(void)   //IIC send data "0"   向 IIC 总线发送"0"
{
    SDA = 0;
     DELAY(DELAY_TIME);
    SCL = 1;
    DELAY(DELAY_TIME);
    SCL = 0;
    DELAY(DELAY_TIME);
}

void SEND_1(void)   //IIC send data "1"   向 IIC 总线发送"1"
{
    SDA = 1;
     DELAY(DELAY_TIME);
    SCL = 1;
    DELAY(DELAY_TIME);
    SCL = 0;
    DELAY(DELAY_TIME);
}

bit Check_Acknowledge(void)   //Read ACK signal   读取 ACK 信号
{
     char F0 = 0;
    SDA = 1;
     DELAY(DELAY_TIME);
    SCL = 1;
    DELAY(DELAY_TIME / 2);
    F0 = SDA;
    DELAY(DELAY_TIME / 2);
    SCL = 0;
    DELAY(DELAY_TIME);
    if (F0)
       return FALSE;
    return TRUE;
}

void Writel2CByte(uchar b) reentrant   //Write One Byte of Data   发送一个字节
```

```
{
    char i;
    for (i = 0; i < 8; i++)
        if ((b << i) & 0x80)   //Send high bits first   先发送高位
            SEND_1();
        else
            SEND_0();
}

uchar ReadI2CByte(void) reentrant   //Receive one byte   读取一个字节
{
    char b = 0, i, F0 = 0;
    for (i = 0; i < 8; i++)
    {
        SDA = 1;
          DELAY(DELAY_TIME);
        SCL = 1;
          DELAY(DELAY_TIME);
        F0 = SDA;
          DELAY(DELAY_TIME);
        SCL = 0;
        if (F0)
        {
            b = b << 1;   //Receive high bits first   先读取高位
            b = b | 0x01;
        }
        else
            b = b << 1;
    }
    return b;
}

//Write a register's address and a command byte to the sensor
//向传感器写寄存器地址和一个命令字节
//"addr": register's address, "thedata": the command byte
void Write_One_Byte(uchar addr, uchar thedata)
{
    bit acktemp = 1;
     I2C_Start();   //IIC START Signal   发送 IIC 启动信号
     WriteI2CByte(0x58 << 1 + 0);   //The SLAVER address is 0x58
     //传感器的 IIC 总线地址为 0x58
     // The lowest bit of address is 0 means writing   地址值最低位为 0 表示写
```

```
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    WriteI2CByte(addr);    //Send the register's address  发送寄存器的地址值
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    WriteI2CByte(thedata);  //Write command to the sensor  向传感器写命令字节
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    I2C_Stop();  //IIC STOP Signal  发送 IIC 停止信号
}

//Read one byte of data from the sensor  从传感器读取一个字节
uchar Read_One_Byte(uchar addr)
{
    bit acktemp = 1;
    uchar mydata;
    I2C_Start();  //IIC START Signal  发送 IIC 启动信号
     WriteI2CByte(0x58 << 1 + 0);  //The SLAVER address is 0x58
     //传感器的 IIC 总线地址为 0x58
     // The lowest bit of address is 0 means writing  地址值最低位为 0 表示写
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    WriteI2CByte(addr);    //Send the register's address  发送寄存器的地址值
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    I2C_Start();  //IIC START Signal  发送 IIC 启动信号
     WriteI2CByte(0x58 << 1 + 1);  //The SLAVER address is 0x58
     //传感器的 IIC 总线地址为 0x58
     // The lowest bit of address is 1 means Reading  地址值最低位为 1 表示读
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    mydata = ReadI2CByte();    //Read the above register's data  读取上述寄存器的数据值
    acktemp = Check_Acknowledge();  //check the SLAVER's ACK  检查传感器的 ACK
    I2C_Stop();  //IIC STOP Signal  发送 IIC 停止信号
    return mydata;
}

//Ms Time-Delay Function  Ms 延时函数
void Delay_xms(uint x)
{
    uint i, j;
    for (i = 0; i < x; i++)
        for (j = 0; j < 112; j++)
            ;
}

void main(void)  //The main function  主函数
{
```

```
uchar pressure_H, pressure_M, pressure_L,temperature_H,temperature_L;
//Temporary variables used to save bytes of pressure and temperature from the sensor
//临时变量，用于存放从传感器中读出的压力值和温度值的字节数据
long int pressure_ad,temperature_ad;
//Temporary variables used to save AD values of pressure and temperature from the sensor
//临时变量，用于存放从传感器中读出的压力和温度的 AD 值
float pressure,temperature,Shift_N;
//pressure: actual pressure   变量 pressure 用于存放实际压力值
//temperature: actual temperature   变量 temperature 用于存放实际温度值
uchar byte1,byte2;
int EOFF;

Delay_xms(1000);
while (1)
{
   Write_One_Byte(0x01, 0x01);
    //Send 0x01 to the register whose address is 0x01 to start a data collection
   //向传感器 0x01 寄存器发送 0x01 以启动采集数据
   Delay_xms(20);

   pressure_H = Read_One_Byte(0x04);   //Read bytes of pressure from the sensor
   pressure_M = Read_One_Byte(0x05);   //从传感器中读出压力值的字节数据
   pressure_L = Read_One_Byte(0x06);
   pressure_ad = pressure_H * 65536 + pressure_M * 256 + pressure_L;
   //compute the AD pressure of the sensor   计算传感器 AD 转换后的压力值
   temperature_H = Read_One_Byte(0x07);   //Read bytes of temperature from the sensor
   temperature_L = Read_One_Byte(0x08);   //从传感器中读出温度值的字节数据
   temperature_ad = temperature_H * 256 + temperature_L;
   //compute the AD temperature of the sensor   计算传感器 AD 转换后的温度值

   //compute the actual pressure of the sensor   计算传感器实际的压力值
   //pressure's unit is Pa   变量 pressure 的单位为 Pa
   if (pressure_ad >= 8388608)
      pressure = (float) (pressure_ad - 16777216) / 2^21 *(PMAX-PMIN) * 1000+Pmin;
   else
      pressure = (float) pressure_ad / 2^21 * (PMAX-PMIN) * 1000+Pmin;

   //compute the actual temperature of the sensor   计算传感器实际的温度值
   //temperature's unit is Centigrade   变量 temperature 的单位为℃
   if(temperature_ad > 32768)
      temperature_ad -= 65536;
   byte1 = Read_One_Byte(0x20);   //Read temperature parameter from the sensor
```

```
        byte2 = Read_One_Byte(0x21);    //Read temperature parameter from the sensor
        if (byte1 == 0x0C)    //According byte1 to evaluate the variable EOFF
           EOFF = 4096;
        else if (byte1 == 0x8C)    //根据 byte1 的值，求变量 EOFF 的值
           EOFF = -4096;
        else if(byte1 == 0x0D)
           EOFF = 8192;
        else if(byte1 == 0x8D)
           EOFF = -8192;
        else if(byte1 == 0x0E)
           EOFF = 16384;
        else if(byte1 == 0x8E)
           EOFF = -16384;
        Shift_N = byte2 / 10;    //compute the variable Shift_N  计算变量 Shift_N 的值
        temperature = (temperature_ad - EOFF) / 2 ^ Shift_N + 25;
        //the actual temperature of the sensor    传感器的实际温度值

         printf("Actual pressure is %f Pa\r\n",pressure);
         printf("Actual temperature is %f Centigrade\r\n\r\n",temperature);

        Delay_xms(1000);
      }
}
```