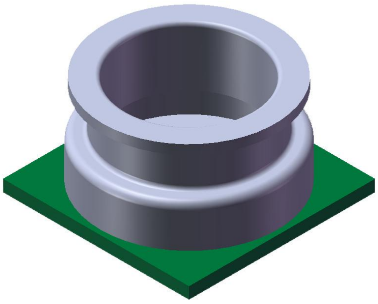
XGZP6827D PRESSURE SENSOR MODULE

FEATURES



- Wide Ranges: -100kPa···0kPa···200kPa(show in Pressure Range)
- Optional 1.8V ~ 3.6V Power Supply, Low Consumption.
- Gage(Positive&Vacuum) Type
- For dry Non-corrosive Gas or Air
- Calibrated Digital Signal(I2C Interface)
- Temp. Compensated: 0℃ ~ +60℃(32℉ ~ +140℉)
- Multiple work mode and FIFO available

APPLICATIONS

- For medical and healthy equipment, e.g. Blood pressure test and monitor, Patient Monitoring, Infusion and Syringe Pumps, Anesthesia machines, Respirators and Ventilators, NPWT, DVT, Catheter, COPD Treatment, Kidney dialysis, Cupping& Cosmetology, Hospital beds, Massage device etc.
- For white goods and consumer electronics, e.g. Refrigerator, Printer, Humidifier, Washer/Dryer, Coffee machine, Dust collector or Cleaner, Robotic, Watch, Sport equipment etc.
- For more fields, e.g. Air pump, Process control, Pressure switching, HVAC and pneumatic device, Air flow monitor, Automation, Automotive application etc.

INTRODUCTION

XGZP6827D is a perfect silicon pressure sensor module offering a digital I2C interface for reading pressure over the specified full scale pressure span, as well as ambient temperature measurement.

The XGZP6827D incorporates a silicon piezoresistive pressure sensor die and an interior Application Specific Integrated Circuit(ASIC) in a small COB package.

The XGZP6827D is fully calibrated and temperature compensated for offset, sensitivity, temperature and non-linearity, so XGZP6827D pressure sensor module 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.

XGZP6827D pressure sensor module is for high volume application at an affordable cost but perfect performance.

Customized calibrations(working voltage, output voltage, 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.

Item		Data	Unit
Available Pressure Range[1]		-100···0···200	kPa
Power Supply[2]		1.8V	Vdc
Current Consumption		6	uA(@OSR_P=1024X)
ADC Resolution[3]		24	Bit
SDA/SCL pull up resistor		4.7	Kohm
ESD HBM		4000	V
Accuracy[4]	10kPa < Pressure≤200kPa	±0.5	%Span
	Pressure≤10kPa	±2.0	
Long Term Stability[5]		±0.5	%Span
Over Pressure[6]		2X	Rated
Burst Pressure[7]		3X	Rated
Compensation Temp.[8]		0 ~ 60/32 ~ 140	℃/℉
Operating Temp.[9]		-40 ~ 100/-40 ~ 212	℃/℉
Storage Temp.		-40 ~ 125/-40 ~ 257	℃/℉
Response Time[10]		2.5	mS

1 Pressure Range(Operating pressure): The available pressure range including various span, not a specific pressure range.

2 Power supply: The default test voltage value: 3.3V, optional power supply voltage range:1.8 ~ 3.6V; For better accuracy, please specify the working voltage as order guide.

3 ADC Resolution:The ADC resolution is defined as the smallest incremental voltage that can be recognized and thus causes a change in the digital output.

4 Accuracy: The max. deviation in output from ideal transfer function at any pressure or temperature over the specified ranges, units are in percent of full scale span (%FSS), which mainly consists of: Offset and Span Shift;Linearity(Non-linearity); Repeatability; Pressure Hsteresis ; TcOffset and TcSpan.

4.1. The accuracy in table is the typical output accuracy during specified pressure range. Contact factory for higher accuracy requirement(e.g ±0.5%Span) if need.

4.2 Non-linearity(Linearity): the deviation of measured output from "Best Straight Line" through three points (Offset pressure, FS pressure and ½ FS pressure)at constant temperature.

4.3 Repeatability: the deviation of measured output when the same pressure is applied continuously, with pressure approaching from the same direction within the specified operating pressure range,under the same operating conditions.

4.4 Pressure Hysteresis: the deviation of measured output at any pressure within the specified range, when this pressure is applied continuously, with pressure approaching from opposite directions within the specified operating pressure range, under the same operating conditions.

4.5 TcOffset (TCO:Temp. Coefficient of Offset): the deviation of measured output with minimum rated pressure applied, over the temperature range of 0° to 60°C, relative to 25°C.

4.6 TcSpan (TCS:Temp. Coefficient of Span): the deviation of measured output over the temperature range of 0° to 60°C, relative to 25°C.

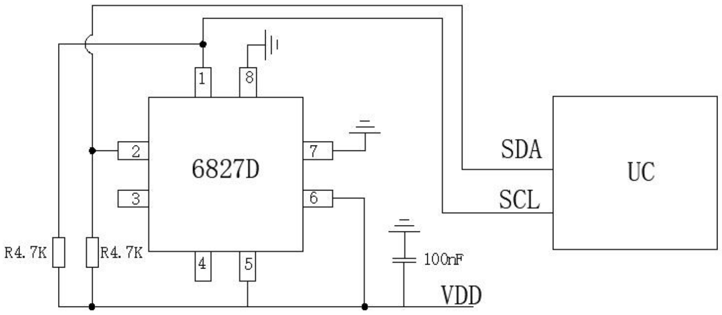
5. Long Term Stability: the sensor’s output deviation when subjected to 1000 hours pressure test.

6. Over Pressure (Proof pressure): the maximum pressure which may be applied without causing durable shifts of the electrical parameters of the sensing element and remain the specification once pressure is returned to the operating pressure range.
7. Burst Pressure: the maximum pressure which may be applied without causing damage to the sensing die or leaks; The sensor should not be expected to recover function after exposure to any pressure beyond the burst pressure.
8. Compensated Temperature: the temperature range over which the sensor have an output proportional to pressure within the specified performance limits.
9. Operating Temperature (or Ambient Temperature): the temperature range over which the sensor have an output proportional to pressure but may not remain within the specified performance limits.
10. Response Time: it is defined as the time for the incremental change in the output from 10% to 90% of of its final value when subjected to a specified step change in pressure.

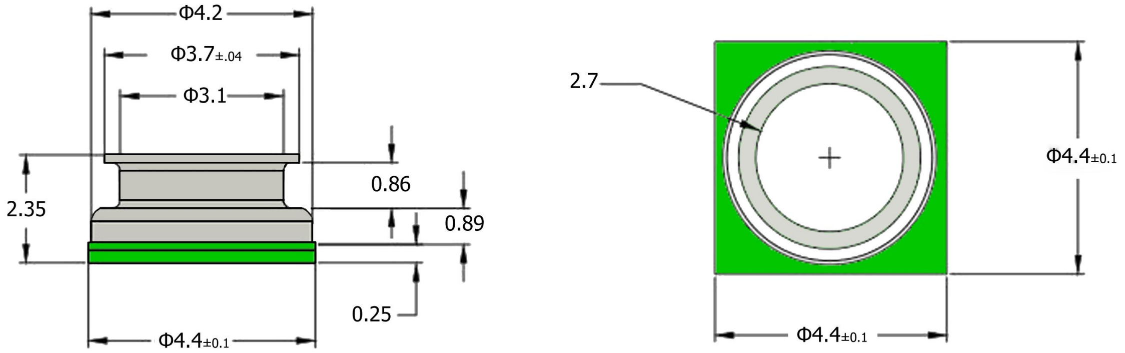
ELECTRICAL CHARACTERISTICS(ASIC)

Parameter	Min.	Typ.	Max.	Unit	Notes
Power Supply	1.8	3.3	3.6	V	
Working Temp.	-20		85	℃	
Standby Current		100	250	nA	Sleep statue in 25℃
Max.Average.Current			0.6	mA	Constant measnurement
LDO Output	1.58	1.67	1.85	V	
PSRR	17			dB	VDD=1.8V
	32			dB	VDD=2.0V
Start Time			1	ms	power-on to start of communication
			2.5	ms	power-on to start of measurement
Response Time			0.5	ms	Sleep state to start communication
			2	ms	Sleep state to start measurement
Temp. Meas. Range	-40		150	℃	Interior temp. sensor
IIC Clock Frequency			400	KHz	

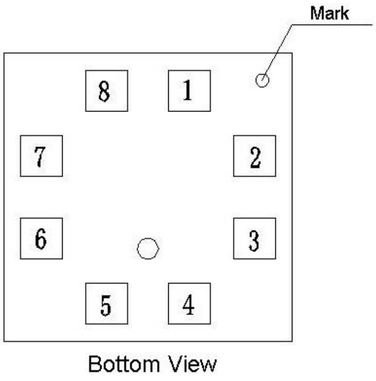
CRICUIT DIAGRAM



DIMENSION (Unit:mm)

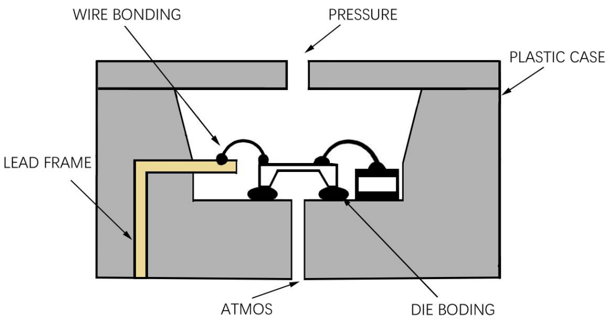


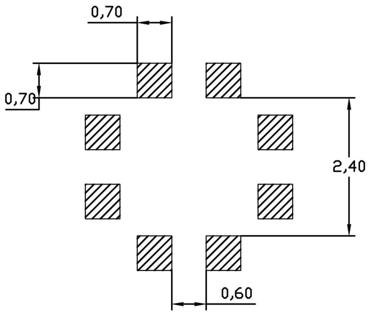
PIN	Name	I/O	Function
1	SCL	I	IIC serial clock input pin
2	SDA	I/O	IIC serial bi-directional input pin
3	INT	O	Interrupt output (data ready)
4	NC	-	NO Connect
5	VDD	I	Power supply
6	VDDIO	I	Power Supply for I/O Circuit
7	GND	I	Ground
8	GND	I	Ground



CROSS SECTION

FOOTPRINT(REFERENCE)





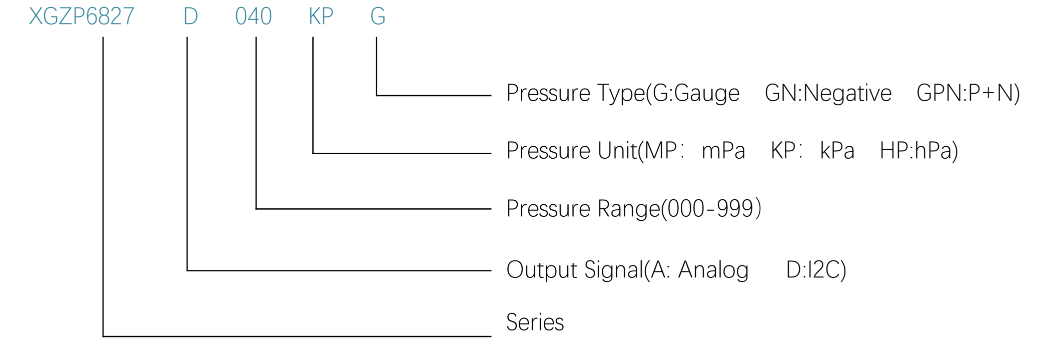
Notes:

1.Implement ESD protection during whole soldering and assembly process.

2.Overload voltage(max.3.6Vdc) may burn the ASIC and cause the sensor fail throughly.

3.More detalis about soldering and storage etc., refer to Overall notes.

ORDER GUIDE



XGZP6827 D 040 KP G

Pressure Type(G:Gauge GN:Negative GPN:P+N)

Pressure Unit(MP： mPa KP： kPa HP:hPa)

Pressure Range(000-999）

Output Signal(A: Analog D:I2C)

Series

Note: Any custom requirement, please remark herewith Part number(e.g custom pressure range etc,.)

ROUTINE PRESSURE RANGE

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 ~ 20	0 ~ 200mbar / 0 ~ 150mmHg	XGZP6827D 020KPG
0 ~ 40	0 ~ 400mbar / 0 ~ 300mmHg	XGZP6827D 040KPG
0 ~ 100	0 ~ 1bar /0 ~ 14.5PSI	XGZP6827D 100KPG
-100 ~ 0	-1 ~ 0bar / -14.5 ~ 0PSI	XGZP6827D 100KPGN
-40 ~ 40	-400 ~ 400mbar / -300 ~ 300mmHg	XGZP6827D 040KPGPN
Other higher pressure range or custom pressure span, consult CFSensor		

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: 4.7K) and remain high level when not communicating. I2C device factory setting slave address: 0X38

□ From master to slave	S Start	A Acknowledge
🟦 From slave to master	P Stop	N Not Acknowledge

Write Operation

S	SlaveAddr	0	A	Command	A	P

S	SlaveAddr	0	A	CmdDat[23:16]	A	CmdDat[15:8]	A	CmdDat[7:0]	A	P

Write 0xE1 0x08 0x00 to sensor address 0x70 to set XGZP6827D to enable the calibration calculation function of pressure value and temperature value; write 0xAC 0x77 0x00 to sensor address 0x70 to notify XGZP6827D to start a pressure value and temperature value measurement .

0x70 means the default 7bits I2C sensor slave device address is 0x38, and the last 1bit is 0 means the master device MCU writes the slave device.

The write address is 0X38 << 1 + 0 = 0X70, the read address is 0X38 << 1 + 1 = 0X71

Read Operation

(a) 8bit status word can be read at any time

S	SlaveAddr	1	A	Status	N	P

The status of the sensor is judged by the Status byte read from the sensor. For details, please refer to Table 3. Bit description of status word.

(b) When the FIFO is disabled, after sending the 0xAC 0x77 0x00 command, wait for the end of the sensor measurement, and then read the 20bit pressure and 20bit temperature measurement data.

S	Slave Addr	1	A	Status	A	Pressure Dat [19:12]	A	Pressure Dat [11:4]	A	Pressure Dat [3:0]	Temp Dat [19:16]	A	Temp Dat [15:8]	A	Temp Dat [7:0]	N	P

PressureDat[19:0] is the calibrated pressure value output by the sensor;

TempDat[19:0] is the calibrated temperature value.

Table： Status of Bits

Use the I2C interface for read operation, and the valid data read always starts from the status byte (Status). The status byte reflects the state of the chip after the last command was executed.

Bit	Significancy	Description
Bit7	(Busy indication)	1--device busy, under measurement Status 0--device idle, under sleep Status
Bit6:5	(Mode Status)	00--present status is NOR mode； 01--present status is CYC mode； 1x--present status is CMD mode。
Bit4	CRC_flag (Memory integrity/ error flag)	0--Indicates that the OTP memory data integrity test (CRC) passed； 1--Indicates that the integrity test has failed, indicating that there is an error in the OTP data, and the retention of the OTP data may be invalid.
Bit3	(Calibration Enable)	0--The calibration calculation function is disabled, and the output data is the original data output by the ADC； 1--The calibration calculation function is enabled, and the output data is the calibrated data.
Bit2	FIFO_Enable	0--FIFO function is disabled； 1--FIFOfunction is enabled
Bit1	FIFO_FULL	0--FIFO data is not full； 1--FIFO data is full
Bit0	FIFO_EMPTY	0--FIFO data is not empty； 1--FIFO data is empty.

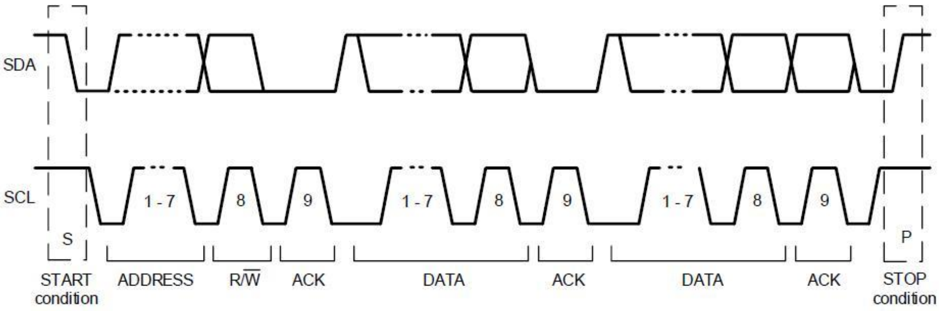
Sleep Standby

After the sensor completes the measurement, it enters the sleep standby state, and the standby power consumption is 0.1uA。

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.



- Conversion example：

After reading the calibration data, it is necessary to perform simple conversion on the unsigned number in the form of AD value.

For ease of understanding, assume that the calibration data read is: 0x04; 0x9B; 0xB0; 0xC5; 0x56; 0xAA. Among them, 0x04 is the status byte, if its Bit7 is 1, it means that the latest I2C is busy, and it needs to wait for a period of time; if its Bit7 is 0, it means that the device is not busy, and data can be read. For a detailed description of each bit of the status word, see Table(Status of Bits) above.

The two bytes of 0x9B, 0xB0, and the upper 4 bits of 0xC5 are the pressure calibration values, and the lower 4 bits of 0xC5, 0x56, and 0xAA are the temperature calibration values.

The pressure calibration value is converted as follows:

Convert the two bytes of 0x9B, 0xB0 and the upper 4 bits of 0xC5 into decimal numbers as 637708. This calculation assumes that the calibration range is 0Kpa~40Kpa, and the corresponding AD output is 157286.4 ~ 891289.6 (15% AD ~ 85% AD), the actual pressure value = (40-0) / (891289.6 - 157286.4) * (637708 - 157286.4) = 26.1809 KPa.

The temperature calibration value is converted as follows:

The lower 4 bits of 0xC5, 0x56, and 0xAA two bytes are converted into decimal numbers as 349866, and the actual temperature value = 349866 / 1048576.0 * 200.0 - 50.0 = 16.7316 ℃.

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 flexiable 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.

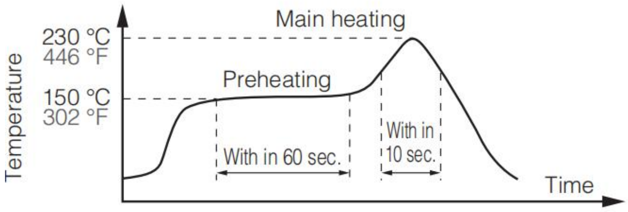
▼DIP soldering (DIP Terminal)

⊙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 solderbility 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.

▼ Please keep the sensors sealed using static shielding bags on storage. The PINs of sensor(if have) may be black by oxidation under atmosphere environment if the materials include the oxidizable material, althought it wouldn't affect the sensor 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
14-15F/4Bldg 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)

```
#define DELAY_TIME 20     //Time-Delay Parameter, Can be adjusted as needed

sbit SCL = P4^6;    //IIC clock line
sbit SDA = P4^7;    //IIC data line

unsigned char Device_Address = 0x38 << 1;    //The IIC Address of XGZP6827D

volatile bit XGZP6827D_IsCalibrated = 0;
   //variable, used to mark whether XGZP6827D has activated the function of calibration
float pressure = 0.0, temperature = 0.0;
   //variable, used to save the actual pressure (its unit is Pa) and actual temperature (its unit is ℃)

//Ms Time-Delay Function    Ms
void I2C_Delay_Ms(unsigned char n)
{
        unsigned char i, j;
        for(i=0; i<n; i++)
                for(j=0; j<123; j++);
}

void Start(void)    //IIC Start signal
{
        SDA = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SDA = 0;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 0;
        I2C_Delay_Ms(DELAY_TIME);
}


void Stop(void)    //IIC Stop signal
{
        SDA = 0;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SDA = 1;
        I2C_Delay_Ms(DELAY_TIME);
}

unsigned char Check_ACK(void)    //Read ACK signal
{
        unsigned char ack;
        SDA = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME / 2);
        ack = SDA;
        I2C_Delay_Ms(DELAY_TIME / 2);
        SCL = 0;
        I2C_Delay_Ms(DELAY_TIME);
        return ack;
}

void Send_ACK(void)    //Send ACK signal
{
        SDA = 0;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 0;
        I2C_Delay_Ms(DELAY_TIME);
        SDA = 1;
        I2C_Delay_Ms(DELAY_TIME);
}

void Send_NACK(void)    //Send NACK signal
{
        SCL = 0;
        SDA = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 0;
        I2C_Delay_Ms(DELAY_TIME);
}

void SendByte(unsigned char byte1)    //Send one byte
{
    unsigned char i = 0;
    do
    {
        if (byte1 & 0x80)
        {   SDA = 1;   }
        else
        {   SDA = 0;   }
        I2C_Delay_Ms(DELAY_TIME);
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME);
        byte1 <<= 1;
        i++;
        SCL = 0;
    } while (i < 8);
    SCL = 0;
    I2C_Delay_Ms(DELAY_TIME);
    SDA = 1;   //Release SDA   Makes it possible for the slave to send an ACK.
    I2C_Delay_Ms(DELAY_TIME);
}

unsigned char ReceiveByte(void)   //Receive one byte
{
    unsigned char i = 0, tmp = 0;
    do
    {
        tmp <<= 1;
        SCL = 1;
        I2C_Delay_Ms(DELAY_TIME);
        if (SDA)
        {   tmp |= 1;   }
        SCL = 0;
        I2C_Delay_Ms(DELAY_TIME);
        i++;
    } while (i < 8);
    return tmp;
}

//Write the sensor's address and several command bytes
unsigned char XGZP6827D_IIC_Write(unsigned char address, unsigned char *buf, unsigned char count)
{
    unsigned char timeout, ack;
    address &= 0xFE;   // The lowest bit of address is 0 means writing
    Start( );
    I2C_Delay_Ms(DELAY_TIME);
    SendByte(address);      //Send the address to the IIC
    I2C_Delay_Ms(DELAY_TIME);
    timeout = 0;
    do   //Getting the ACK response from the sensor means success, otherwise means failure
    {
        ack = Check_ACK( );
        timeout++;
        if (timeout == 10)
        {
            Stop( );
            return 1;
        }
    } while (ack);
    while (count)
    {
        SendByte(*buf);   //Write bytes to the sensor
        I2C_Delay_Ms(DELAY_TIME);
        timeout = 0;
        do
        {
            I2C_Delay_Ms(DELAY_TIME);
            ack = Check_ACK( );
            timeout++;
            if (timeout == 10)
            {   return 2;   }
        } while (ack);
        buf++;
        count--;
    }
    Stop( );
    return 0;
}

//Read several bytes of data from the sensor
unsigned char XGZP6827D_IIC_Read(unsigned char address, unsigned char *buf, unsigned char count)
{
    unsigned char timeout, ack;
    address |= 0x01;   // The lowest bit of address is 1 means reading
    Start( );
    SendByte(address);
    I2C_Delay_Ms(DELAY_TIME);
    timeout = 0;
    do   //Getting the ACK response from the sensor means success, otherwise means failure
    {
        ack = Check_ACK( );
        timeout++;
        if (timeout == 10)
        {
            Stop( );
            return 1;
        }
    } while (ack);
    I2C_Delay_Ms(DELAY_TIME);
    while (count)
    {
        *buf = ReceiveByte( );   //Read bytes from the sensor
        if (count != 1)   //Except the last byte, send ACK to the sensor after receiving byte
            Send_ACK( );
        else
            Send_NACK( );
        buf++;
        count--;
    }
    Stop( );
    return 0;
}

//Read the status of the sensor and judge whether IIC is busy
unsigned char XGZP6827D_IsBusy(void)
{
    u8 status;
    XGZP6827D_IIC_Read(Device_Address, &status, 1);
    status = (status >> 7) & 0x01;
    return status;
}

//Judge whether XGZP6827D has activated the function of calibration
unsigned char XGZP6827D_IsEnable(void)
{
    u8 status;
    XGZP6827D_IIC_Read(Device_Address, &status, 1);
    status = (status >> 3) & 0x01;
```

```
        return status;
}

void XGZP6827D_get_cal(void)
    //Reading and compute pressure and temperature from the XGZP6812D
{
        unsigned int i = 0;
        unsigned char buffer[10] = {0};
        unsigned long press_raw = 0;    //variable, used to save the calibrated pressure
        unsigned long temp_raw = 0;     //variable, used to save the calibrated temperature

        if(!XGZP6827D_IsCalibrated)    //If XGZP6827D has not activated calibrating function
       {
                buffer[0] = 0xE1;
                buffer[1] = 0x08;
                buffer[2] = 0x00;
                XGZP6827D_IIC_Write(Device_Address, buffer, 3);
                 //Let XGZP6827D activate calibrating function
                I2C_Delay_Ms(5);
                for(i = 0; i<5; i++)
                {
                        if(XGZP6827D_IsEnable( ))    //If XGZP6827D has activated calibrating function
                        {
                                I2C_Delay_Ms(5);
                                XGZP6827D_IsCalibrated = 1;
                                printf("XGZP6827D is calibrated\r\n");
                                break;
                        }
                }
        }
        else
        {
                printf("XGZP6827D is calibrated\r\n");
        }
        buffer[0] = 0xAC;
        buffer[1] = 0x77;
        buffer[2] = 0x00;
        XGZP6827D_IIC_Write(Device_Address, buffer, 3);
          //Let XGZP6827D start one measurement
        I2C_Delay_Ms(5);
        while(1)
        {
```

```
        if(XGZP6827D_IsBusy( ))    //If XGZP6827D has not completed a measurement
        {
            I2C_Delay_Ms(5);
            printf("XGZP6827D is busy\r\n");
        }
        else    //If XGZP6827D has completed a measurement, data are ready to read
        {
            printf("XGZP6827D is ready\r\n");
            break;
        }
    }
    XGZP6827D_IIC_Read(Device_Address, buffer, 6);    //Read 6 bytes out from XGZP6827D

    press_raw = ((((unsigned long)buffer[1]) << 16) | (((unsigned int)buffer[2]) << 8) | (buffer[3] & 0xf0)) / 16;
    //compute the calibrated pressure of XGZP6827D
    temp_raw = (((((unsigned long)buffer[3] & 0x0f) << 16)) | ((unsigned int)buffer[4] << 8) | (buffer[5]));
    //compute the calibrated temperature of XGZP6827D

    pressure = ((float)press_raw - 157286.4) * 40.0 / (891289.6 - 157286.4);
    //compute the actual pressure of XGZP6827D
    temperature = (float)temp_raw / 1048576.0 * 200.0 - 50.0;
    //compute the actual temperature of XGZP6827D

    printf("pressure is %f\r\n",pressure);
    printf("temperature is %f\r\n\r\n",temperature);
}

int main(void)    //The main function
{
    while(1)
    {
        XGZP6827D_get_cal( );    //
            //Reading and compute pressure and temperature from the XGZP6812D
        I2C_Delay_Ms(1000);
    }
}
```