GZP6827D

Pressure Sensor

Digital Output(IIC)

Datasheet

Version：V1.8

Issued Data：2023.11.17

Table of Contents

1. Product Description 4
1.1 Features 4
1.2. Application Areas 4
2. Functional Description 5
2.1 Pin Definition 5
2.2 Block Diagram 6
2.3 Accuracy 6
3. Technical Indicators 7
3.1 Maximum Rated Parameters 7
3.2 Performance Indicators 8
4. Application Circuit 8
5. I2C Communication Protocol 9
6. Value Conversion and Calculation 12
7. Structure Specification (unit: mm) 13
8. Order Guide 14
9. Model Example 14
10. Instructions for Use 15
10.1 Soldering 15
10.2 Cleaning Requirements 16
10.3 Storage and Transportation 16
10.4 Other Precautions 16
11. Packing Information 17
Safety Precautions 18
IIC Example Code (Attachment: IIC Code Example) ..... 错误！未定义书签。

Document Revision History

Revision	Description	Date
V1.0	Initial version	2021.01.22
V1.1	Add electrical performance and pressure reading	2021.05.18
V1.2	Add cover, table of contents	2021.10.20
V1,3	Add working modes and instruction descriptions, adjust product classification	2022.03.14
V1.4	Modify packaging specifications	2022.08.15
V1.5	Modify I2C device address	2022.08.29
V1.6	Template modification	2023.05.12
V1.7	Update history figure library images	2023.09.27
V1.8	Update history figure library images	2023.11.07
V1.9	Add the IC version	2025.01.07

The company reserves the right to make changes in the specifications contained herein without further notice.

The copyright of the product specification and the final right of interpretation of the product belong to Sencoch.

1. Product Description

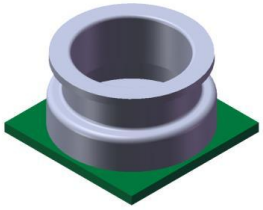
The GZP6827D type pressure sensor is packaged in a PCB board format, containing a sealed pressure sensor and a signal conditioning chip. It digitally compensates for the offset, sensitivity, temperature drift and non-linearity of the sensor. It uses a 24-bit ADC and the conditioning chip is equipped with a built-in temperature sensor to output high-precision pressure and temperature values. It also provides an IIC communication protocol interface, with strong anti-interference capability. All measurement data has been fully calibrated and temperature compensated.

The GZP6827D pressure sensor features fast response, high accuracy, excellent linearity, and long-term stability.

With its compact size and grooved pressure interface design, the GZP6827D pressure sensor facilitates easy assembly and sealing for users. It is particularly suitable for wearable devices with limited space.

1.1 Features

- Pressure range: 0 ~ 60PSI
- Gauge pressure type
- Suitable for non-corrosive gases
- Interior gel for moisture protection
- 1.8~3.6V power supply, low consumption
- Temperature compensated
- FIFO Function



1.2. Application Areas

- Electronic blood pressure monitors,electric breast pump.
- Wearable devices such as blood pressure watches and wristbands

2. Functional Description

This product is made with advanced micro-electromechanical principles. The crucial technology is the MEMS pressure sensor chip based on the silicon piezoresistive effect and the high-performance signal conditioning AISC chip. The silicon micro-piezoresistive MEMS pressure sensor chip is connected to a Wheatstone bridge composed of four strain-sensitive resistors. The output signal is amplified, temperature compensated and linearized by the ASIC chip. The linearity and temperature compensation of the transfer function are realized by the digital processing circuit in the ASIC. The high-precision pressure measurement in the full operating temperature range is achieved through the polynomial compensation algorithm and multi-point pressure calibration technology at multiple temperatures.

2.1 Pin Definition

The pin configuration of the pressure sensor is shown in Figure 1.

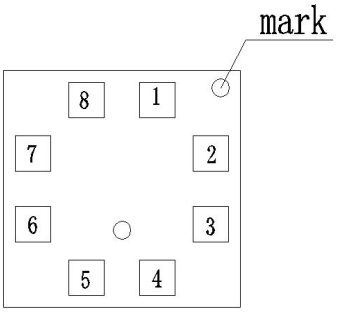


Fig.1 Pin Diagram

Tab.1 Pin Definition

Pin No.	Description	Remark
1	SCL	Signal output pin
2	SDA	Data output pin
3	INT	Interrupt output
4	NC	Floating pin
5	VDD	Power input positive
6	VDDIO	I/O Power input positive
7	GND	Power negative
8	GND	Power negative

2.2 Block Diagram

The sensor functional block diagram is shown in Figure 2.

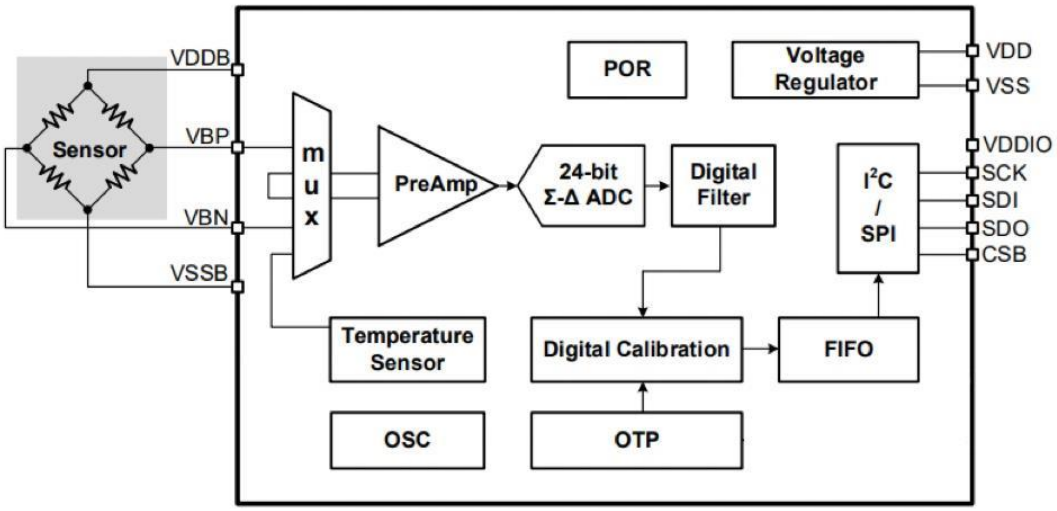


Fig.2 Block Diagram

2.3 Accuracy

The accuracy of the GZP6827D pressure sensor is composed of its linearity, repeatability, and hysteresis errors. The value calculated by the transfer function is the specified value of the sensor and also the theoretical value. The error of the sensor is equal to the difference between the actual output value of the sensor under the specified input pressure and the specified output value.

Overall Accuracy

The overall error includes more accuracy sources based on the product accuracy :

Pressure drift: The output deviation between the actual output voltage at zero point and full scale and the specified output voltage within the specified pressure range.

Temperature effect: The output deviation of zero point and full scale at different temperatures within the temperature range.

The overall accuracy is expressed by error band, and the data are shown in Figure 3 and Table 2 shown.

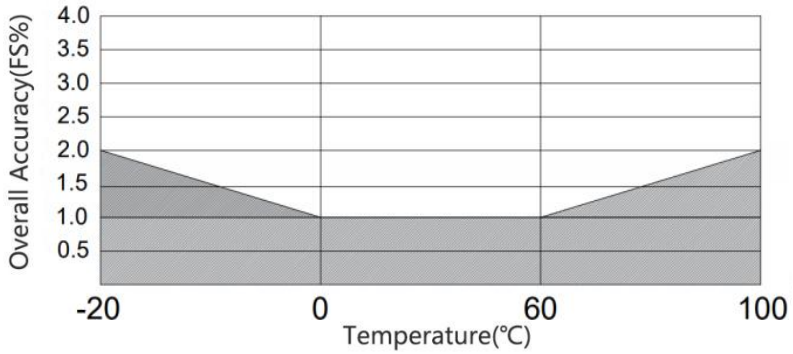


Fig.3 Relationship between overall accuracy and temperature

Tab.2 Overall Accuracy

Temperature (℃)	**Overall Accuracy(Fs%)**
-20~100	±2.0%
0~60	±1.0%

*Different pressure ranges have different overall errors. Please consult customer service for more details.

3. Technical Indicators

Measured at a power supply of 1.8~3.6V DC and a temperature of 25°C.

3.1 Maximum Rated Parameters

The maximum rated parameters of the sensor are shown in Table 3.

Tab.3 The maximum rated parameters

Parameter	**Min.**	**Typical Value**	**Max.**	**Unit**	**Remark**
Supply Voltage	-0.3		3.6	V	VDD/VDDIO
ESD Protection		±2		kV	HBM
Overload Pressure		1.5X		Rated	
Bursting Pressure		2X		Rated	
Working Temperature	-30		100		
Storage Temperature	-40		125		

1. Different pressure range may have different overload pressure and burst pressure, please consult Sencoch for more details.
2. Long exposure at the specified limits may cause degradation to the device.

3.2 Performance Indicators

The sensor performance indicators are shown in Table 4.

Tab.4 Sensor performance indicators

Parameter	Value	Unit	Remark
Pressure Range	-40...0 ~ 40...200	KPa	Customizable
Power Supply	1.8～3.6	V	
Standby Current	＜250	nA	25℃
Accuracy	±1	%Span	
Pressure Resolution	24	Bits	
Built-in Temperature Accuracy	±2	℃	@0~70℃
Compensation Temperature	0～60	℃	
Pull-up Resistors	4.7	K ohm	
Clock Frequency	400	KHz	Max.

* The different pressure range may have different accuracy, please consult Sencoch for more details.

4. Application Circuit

The recommended application circuit is shown in Figure 4.

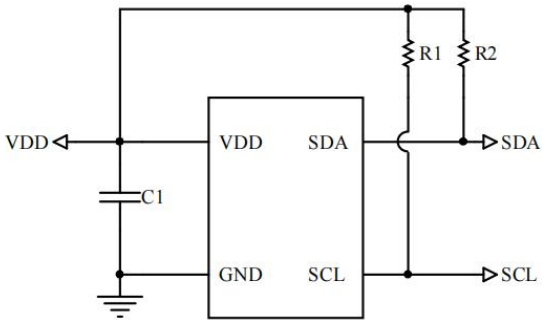


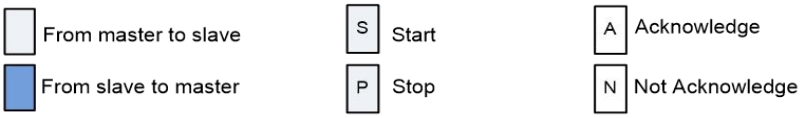
Fig.4 Application circuit

Notice :

1. Please confirm the electrical definition before soldering and assembly
2. Do not have any electrical connection with NC pin, otherwise it may cause sensor failure
3. Provide anti-static protection during soldering
4. Overload voltage (6.5Vdc) may burn out the circuit chip
5. Please add a 0.1uf capacitor between VDD and GND
6. Please pay attention to the power polarity during assembly

5. I²C Communication Protocol

The I2C bus uses SCL and SDA as signal lines, both of which are connected to VDD through pull-up resistors (typically 4.7K) and remain high level when not communicating.
The sensor is calibrated at the factory. Sending the 0xAC command to get the calibrated data..



Write Command



0xF0 means that the default 7bits I2C sensor slave device address is 0x78, and the last 1bit is 0 means that the master device MCU writes to the slave device. 0xAC is the command word to start the slave device sensor to perform a measurement. (The write address is 0X78<<1+0=0XF0, and the read address is 0X78<<1+1=0XF1)

Read Command



After sending the write command, need to wait for a while till measurement finish from the slave device sensor, and then send the read command to read the measurement data. Then sending the 0XF1 command to determine whether the sensor data acquisition has been completed

The waiting time depends on the settings of [13:11] Pressure Oversampling Rate of OTP (Address: 0x14) and [15:14] Temperature Oversampling Rate of OTP (Address: 0x14). The waiting time is =Tp+Tt.

Pressure Oversampling Rate and Measurement Time Comparison Table

OSR_Pressure[13:11] （Binary）	OSR	Measurement Time Tp(ms)
000	32768	203
001	16384	105
010	8192	56
011	4096	31
100	2048	19
101	1024	13
110	512	10

Temperature Oversampling Rate and Measurement Time Comparison Table

OSR Temperature[15:14] （Binary）	OSR	Measurement Time Tt(ms)
00	2048	19
01	4096	31
10	8192	56
11	16384	105

Read Pressure Value

The read calibration data consists of 6 bytes, which are 1-byte status word, 3-byte pressure calibration value, and 2-byte temperature calibration value.

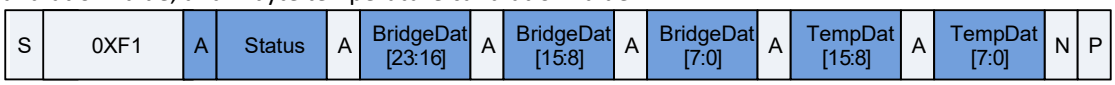


Table5: Status of Bits

Bit	Significancy	Description
Bit7	Reserved	Absolute value 0
Bit6	Power indication	1: Power on; 0: Power off
Bit5	Busy indication	1: Data collection incomplete 0: Data collection complete, data for reading.
Bit4	Reserved	Absolute value 0
Bit3	Reserved	Absolute value 0
Bit2	Reserved	Absolute value 0
Bit1	Reserved	Absolute value 0
Bit0	Reserved	Absolute value 0

Table6: I2C Command

Command(byte)	Return	Description	NOR Mode	CMD Mode
0x00~0x1F	16-bit data	Read data in the OTP that address matching command	Support	Support
0x40~0x5F Followed command byte: 0x0000 ~0xFFFF	—	Write data to OTP; Address is Command value subtract 0x40 (Address is 0x00 to 0x1F)	Support	Support
0xA0~0xA7 Followed command byte: 0xXXXX	24-bit raw data Get_Raw	Get_Raw Conduct one measurement, and write the raw ADC data to registers. See table 6-3 for further interpretation	Support	Support
0xA8	24-bit raw data Get_Raw	Start_NOM Quit CMD mode, enter NOR mode	No-Support	Support
0xA9	—	Start_CM Quit NOR mode, enter CMD mode	Support	No-Support
0xAA	—	Write_ChecksumC If CRC values are not wrote to OTP, the command check data in OTP register and writes CRC values to OTP	Support	Support
0xAC	24-bit compensated bridge data and 16-bit compensated temperature data	Get_Cal Measure based on OTP settings(AZBM, BM,AZTM and TM), write conpensated bridge and temperature data to I2C interface	Support	Support
0xB0~0xBF	24-bit compensated bridge data and 16-bit compensated temperature data	Get_Cal_S and Get_Cal are the same except that Get_Cal measures based on OTP defined OSR and Get_Cal_S measures based on command defined OSR, see following table	Support	Support

Table7： Get_Cal_S Command

Command 0xBX(HEX)	Function	Detail
X [3] Bit	OSR_T, ADC OSR of temperature measurement	0: 4x OSR 1: 8x OSR
X [2:0] Bit	OSR_P, ADC OSR of pressure measurement	000: 128x OSR 100: 8x OSR 001: 64x OSR 101:4x OSR 010: 32x OSR 110: 2x OSR 011: 16x OSR 000: 1x OSR

For example, to set the temperature ADC to 4x oversampling and the piezoelectric panel ADC to 1x oversampling, the command format is 0xB7,Just replace 0xAC with 0xB7

Table8： OPT Register

Addr	Bit Range	Description	Notes/Explanations
0x00~0x13		Calibration Coefficient	
0x14	15:14	Temperature_OSR	00:8X 01:16 X 10: 32X 11:64X
	13:11	Pressure_OSR	000: 128X 001: 64X 010: 32X 011: 16X 100: 8X 101: 4X 110: 2X 111: 1X
	10:8	ADC offset	000 : 1/16 —> [-1/16, 15/16] 001 : 2/16 —> [-2/16, 14/16] 010 : 3/16 —> [-3/16, 13/16] 011 : 4/16 —> [-4/16, 12/16] 100 : 5/16 —> [-5/16, 11/16] 101 : 6/16 —> [-6/16, 10/16] 110 : 7/16 —> [-7/16, 9/16] 111 : 8/16 —> [-8/16, 8/16]
	7:6	Reserved	
	5	Signal Polarity	1： No Inversion，0： Inversion
	4:0	Reserved	
0x15~0x16		Internal Test	
0x17	4	Interrupt Enable	0： disable, 1: enable
	3:2	Interrupt 0 Configuration Bits	00 : Invalid 01 : Calibration value exceeds preset upper limit (TH_H) 10 : Calibration value falls below preset lower limit (TH_L) 11 : Calibration value exceeds preset upper limit (TH_H) or falls below preset lower limit (TH_L)
	1:0	Interrupt 1 Configuration Bits	00 : Invalid 01 : Calibration value exceeds preset upper limit (TH_H) 10 : Calibration value falls below preset lower limit (TH_L) 11 : Calibration value exceeds preset upper limit (TH_H) or falls below preset lower limit (TH_L)

6 Value Conversion and Calculation

Calculation Formula:

Pressure = (PMAX-PMIN)/(DMAX-DMIN)(Dtest-DMIN)+PMIN

Temperature = Temp_ADC/65536190-40

Where:

PMAX: Calibration upper limit pressure value for pressure range; PMIN: Calibration starting pressure value for pressure range

DMAX: AD value corresponding to calibration upper limit pressure; DMIN: AD value corresponding to calibration starting pressure

Dtest: Current pressure reading AD value

After reading calibration data, a simple conversion is needed for the unsigned number represented in AD value form.

Example Illustration: Assuming PMAX: 40KPA, calibration starting pressure 0KPA, corresponding AD output is DMAX: 14260633 and DMIN: 2516582

The read calibration data is: 0x00 0x9B 0xB0 0xC5 0x5F 0x30

0x00 is the status word: Bit5 = 1 indicates that the last I2C was busy, and a wait is required. If Bit5 = 0, the device is not busy, and data can be read. For detailed descriptions of each bit in the status word, please refer to Table 4. Bit Description.

0x9B 0xB0 0xC5 three bytes are the pressure calibration value, converted to decimal is 10203333;

0x5F 0x30 two bytes are the temperature calibration value, converted to decimal is 24368;

According to the above formula:

Actual pressure value = ((40-0) / (14260633-2516582)) * (10203333-2516582) +0 = 26.18 KPA.

Actual temperature value = (24368 / 65536 * 190 – 40) = 30.65 ℃.

Note: Corresponding relation between Pressure range and AD value

Pressure Range(kPa)	Lower Range Value	Upper Range Value	Opposite AD value(Pmin)	Opposite AD value(Pmax)
	Pmin	Pmax	Dmin_P	Dmax_P
0~40	0	40	2516582	14260633

7.Structure Specification (unit: mm)

Refer to Figure 6 for sensor dimensions. (Tolerance ±0.5mm)

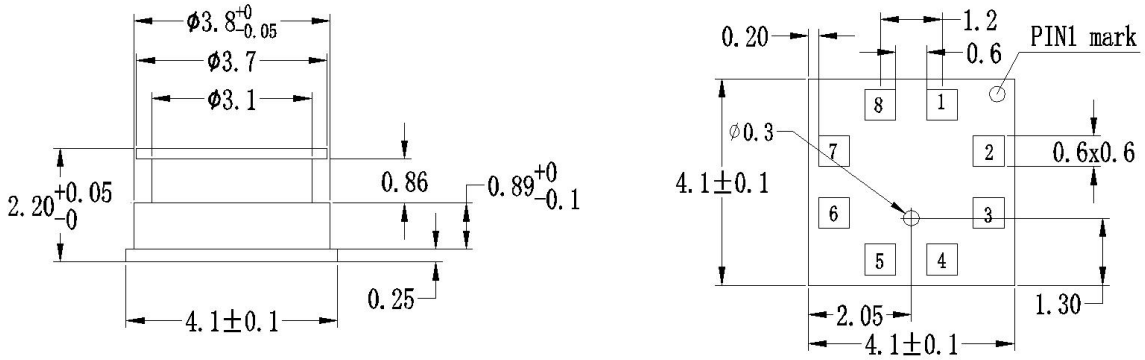


Fig.6 Sensor Dimensions

Recommended Footprint Layout refer to Figure 7.

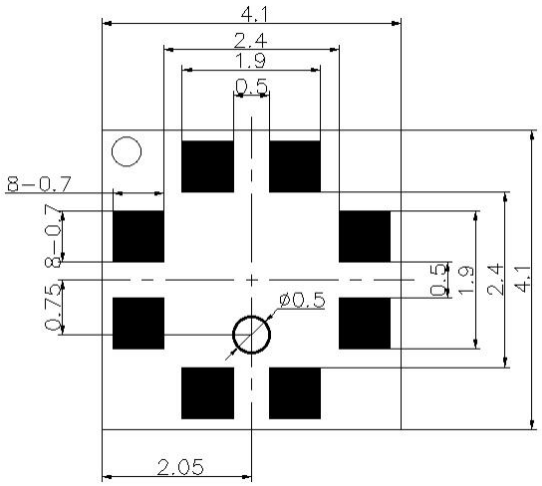


Fig.7 Recommended Footprint

8. Order Guide

GZP 6827 D D 040KPP B01 WX

Tab.9 Order Guide

GZP	Pressure Sensor Series
6829	Product Series
D	Output type A: Analog output D: IIC output
D	IIC Address 0x78
040KPP	Pressure range: 060 means the measured pressure value is 40 (including 0~40, -40~0, -40~40) Pressure unit: KP: KPa MP: MPa PS: PSI BA: Bar Pressure type: P: positive pressure (such as 0~40) N: negative pressure (such as -40~0) W: negative pressure to positive pressure (such as -40~40) Therefore, 040KPA means the measured pressure from 0 to 40KPA
B01	Packaging Method: B01: Tape
WX	Company interior code

9.Model Example

Tab.10 Model Example

Pressure Range	**Model**
0 ~ 40KPP	GZP6827DD040KPP B01 WX

*For more customized ranges and special parameter part numbers, please consult the manufacturer.

10. Instructions for Use

10.1 Soldering

Since this product has a small structure with low heat capacity, please minimize the influence of heat from the outside. Otherwise, it may be damaged due to thermal deformation and cause changes in characteristics. Please use non-corrosive rosin type flux. In addition, since the product is exposed to the outside, please be careful not to allow flux to penetrate into the inside.

1) Manual soldering

· Please use a soldering iron with a head temperature of 260 to 300°C (30 W) and perform the work within 5 seconds.

· When soldering with a load applied to the terminals, the output may change, so be careful.

Please keep the soldering iron tip clean.

2) Reflow soldering (SMD terminal type)

The recommended reflow oven temperature setting conditions are show:

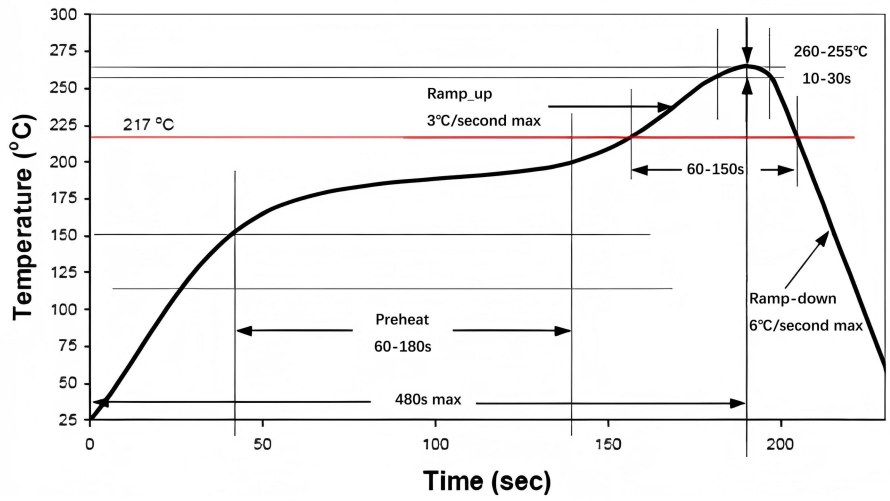


Fig.8 Remelting temperature setting conditions

3) The warping of the printed circuit board relative to the entire sensor should be kept below 0.05mm. Please manage this.

4) After installing the sensor, when cutting and bending the substrate, be careful not to cause stress on the soldered parts.

5) Since the sensor terminals are exposed, if metal pieces touch the terminals, abnormal output will occur. Be careful not to touch them with metal pieces or hands.

6) After soldering, when coating is applied to prevent insulation degradation of the substrate, be careful not to allow chemicals to adhere to the sensor.

10.2 Cleaning Requirements

1) Since the product is an open type, be careful not to allow cleaning fluid to enter the interior.

2) Please avoid using ultrasonic cleaning as it may cause product failure.

10.3 Storage and Transportation

1) This product is not drip-proof, so do not use it in a location where it may be splashed with water.

2) Do not use in an environment where condensation occurs. In addition, if moisture attached to the sensor chip freezes, it may cause changes in sensor output or damage.

3) The chip of the pressure sensor is structurally exposed to light, and the output will change. Especially when applying pressure through a transparent cover, please avoid light from reaching the chip of the sensor.

4) Normally packaged pressure sensors can be transported by ordinary transportation tools. Please note: The product should be protected from moisture, impact, sunburn and pressure during transportation.

10.4 Other Precautions

1) If the installation method is incorrect, it may cause an accident, so please be careful.

2) Avoid using the product in a manner that applies high-frequency vibrations, such as ultrasonic waves.

3) The only pressure media that can be used directly are air and non-corrosive gases. Other media, especially when used in corrosive media or media containing foreign matter, may cause malfunctions and damage, so please avoid using it in the above environment .

4) There is a pressure sensor chip inside the pressure inlet. If a needle or other foreign object is inserted into the pressure inlet, the chip will be damaged and the inlet will be blocked, so please avoid such operations. In addition, please avoid blocking the atmosphere inlet when using it.

5) Please use the pressure within the rated pressure range. Using outside the range may cause damage.

6) Since static electricity may cause damage, please pay attention to the following matters when using.

Please ground the charged objects and workers on the table to discharge the static electricity around safely.

7) If you have any questions, please feel free to ask.

11. Packing Information

Tape Packing:

2500PCS/Tape

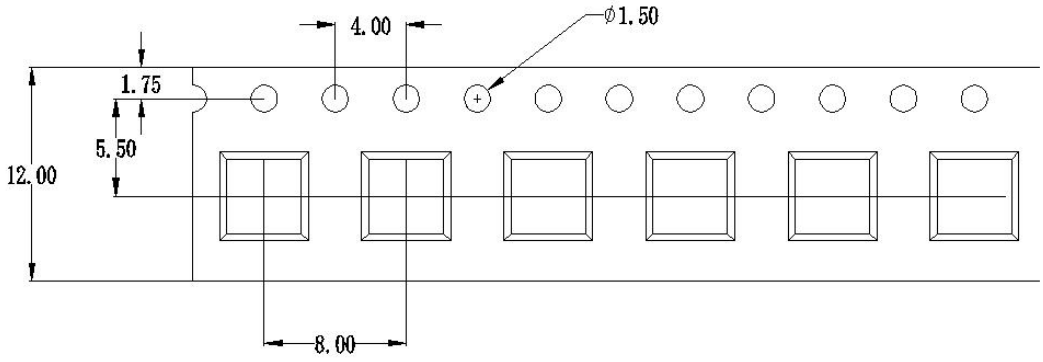


Fig.9 Tape

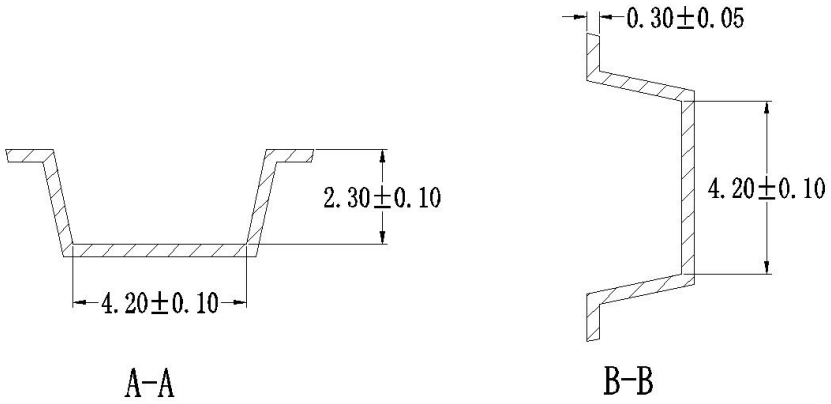


Fig.10 Detail of the tape

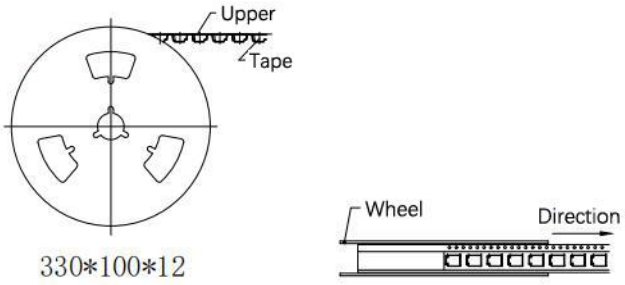


Fig.11 Tape Carrier

Safety Precautions

This product is made of semiconductor components for general electronic equipment (communication equipment, measuring equipment, working machinery, etc.). Products using these semiconductor components may malfunction and fail due to external interference and surges, so please confirm the performance and quality under actual use. To be on the safe side, please perform safety design on the device (setting of protection circuits such as fuses and circuit breakers, multiple devices, etc.) so that life, body, property, etc. will not be harmed in the event of a malfunction. To prevent injuries and accidents, please be sure to comply with the following matters:

·The driving current and voltage should be used below the rated values.

Please wire according to the electrical definition . In particular, reverse connection of the power supply may cause accidents due to circuit damage such as heat, smoke, and fire, so please be careful.

·Be careful when fixing the product and connecting the pressure inlet .

Warranty and Disclaimer

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. Sencoch Technology reserves the right to make changes without further notice to any product herein. Sencoch Technology makes no warranty, representation or guarantee regarding the suitability of its product for any particular purpose, nor does Sencoch Technology 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. Sencoch Technology does not convey any license under its patent rights nor the rights of others.

IIC Example Code (Attachment: IIC Code Example)

```
/*********************************************************************************
********
//*****************************************Digital tube displays pressure and temperature
//*****************************************STC12+MAX7219*************************
***
//***************************CLK=P2^2 CS=P2^1
DIN=P2^0*******************************************************************************
*****
//*****************************************SCL=P1^7
SDA=P1^6*******************************************************************************
****
//*********************************************************************************
********
#include <STC12C5A60S2.H>
#include <stdio.h>
#include <math.h>
#include "MAX7219.h"
#include "GZP6827D.h"
#include "IIC.h"

extern float pressure_kpa ;//, temp = 0.0;//float 4byte
extern unsigned long pressure_pa ;
extern unsigned long temp ;
void Delay300ms()          //@11.0592MHz
{
    unsigned char i, j, k;

    i = 13;
    j = 156;
    k = 83;
    do
    {
        do
        {
```

```
            while (--k);
        } while (--j);
    } while (--i);
}

void main()
{
        unsigned char dis[8] = {0,0,0,0,0,0,0,0};
        Delay_Ms(DELAY_TIME);
        Init_MAX7219();
    while(1)
    {
    GZP6827D_get_cal();
        dis[0] = (unsigned char)(pressure_pa / 10000000);
            dis[1] = (unsigned char)(pressure_pa % 10000000 / 1000000);
                dis[2] = (unsigned char)(pressure_pa % 1000000 / 100000);
                    dis[3] = (unsigned char)(pressure_pa % 100000 / 10000);
                        dis[4] = (unsigned char)(pressure_pa % 10000 / 1000);
                            dis[5] = (unsigned char)(pressure_pa % 1000 / 100);
                                dis[6] = (unsigned char)(pressure_pa % 100 / 10);
                                    dis[7] = (unsigned char)(pressure_pa % 10);

            Write_Max7219(8, dis[0]);
                Write_Max7219(7, dis[1]);
                    Write_Max7219(6, dis[2]);
                        Write_Max7219(5, dis[3]);
                            Write_Max7219(4, dis[4]|0x80);    //Display decimal point
                                Write_Max7219(3, dis[5]);
                                    Write_Max7219(2, dis[6]);
                                        Write_Max7219(1, dis[7]);
        Delay300ms();

            GZP6827D_get_cal();
            temp=temp*10;
            dis[0] = (unsigned char)(temp / 10000000);
```

```
                dis[1] = (unsigned char)(temp % 10000000 / 1000000);
                    dis[2] = (unsigned char)(temp % 1000000 / 100000);
                        dis[3] = (unsigned char)(temp % 100000 / 10000);
                            dis[4] = (unsigned char)(temp % 10000 / 1000);
                                dis[5] = (unsigned char)(temp % 1000 / 100);
                                    dis[6] = (unsigned char)(temp % 100 / 10);
                                        dis[7] = (unsigned char)(temp % 10);


            Write_Max7219(8, dis[0]);
                Write_Max7219(7, dis[1]);
                    Write_Max7219(6, dis[2]);
                        Write_Max7219(5, dis[3]);
                            Write_Max7219(4, dis[4]);         //Display decimal point
                                Write_Max7219(3, dis[5]);
                                    Write_Max7219(2, dis[6]|0x80);
                                        Write_Max7219(1, dis[7]);
            Delay300ms();
        }
}

#include "GZP6827D.h"
#include <math.h>
// Define the upper and lower limits of the calibration pressure
#define PMIN 0    //Zero range pressure for example 0Kpa
#define PMAX 40    //Full Point Pressure Value, for example 40Kpa
#define DMIN 2516582    //AD value corresponding to pressure zero, for example 15%AD=2^24*0.1
#define DMAX 14260633 //AD Value Corresponding to Full Pressure Range, for example 85%AD=2^24*0.9

float pressure_kpa = 0.0;//, temp = 0.0;
unsigned long pressure_pa = 0;
unsigned long temp = 0.0;

//The 7-bit IIC address of the sensor is 0x78
unsigned char Device_Address = 0x78 << 1;
```

```
//Read the status of IIC and judge whether IIC is busy
unsigned char GZP6827D_IsBusy(void)
{
    unsigned char status;
    GZP6827D_IIC_Read(Device_Address, &status, 1);
    status = (status >> 5) & 0x01;
    return status;
}

void GZP6827D_get_cal(void)
{
    unsigned char buffer[6] = {0};
    unsigned long Dtest = 0;
    unsigned int temp_raw = 0;
    //Send 0xAC command and read the returned six-byte data
    buffer[0] = 0xAC;
    GZP6827D_IIC_Write(Device_Address, buffer, 1);
    Delay_Ms(DELAY_TIME);
    while (1)
    {
        if (GZP6827D_IsBusy())
        {
            Delay_Ms(DELAY_TIME);
        }
        else
            break;
    }
    GZP6827D_IIC_Read(Device_Address, buffer, 6);

    //The returned pressure and temperature values are converted into actual values according to the calibration range
    Dtest = (unsigned long)((((unsigned long)buffer[1]) << 16) | (((unsigned int)buffer[2]) << 8) | ((unsigned char)buffer[3]));
    temp_raw = ((unsigned int)buffer[4] << 8) | (buffer[5] << 0);
    if (Dtest != 0)
```

```
    {
        pressure_kpa = (float) ((PMAX-PMIN)/(DMAX-DMIN)*(Dtest-DMIN)+PMIN);     //单位：KPa
        pressure_pa = (unsigned long) (pressure_kpa * 1000.0);     //unit：Pa
    }
    else
    {
        pressure_kpa = 0.0;   //unit：KPa
        pressure_pa = 0;     //unit：Pa
    }
    temp = (double)temp_raw /65536    * 190 - 40;
}

//Write a byte of data through IIC
unsigned char GZP6827D_IIC_Write(unsigned char address, unsigned char *buf, unsigned char count)
{
    unsigned char timeout, ack;
    address &= 0xFE;
    Start();
    Delay_Ms(DELAY_TIME);
    SendByte(address);
    /*    Set_SDA_INPUT();    */
    Delay_Ms(DELAY_TIME);
    timeout = 0;
    do
    {
        ack = Check_ACK();
        timeout++;
        if (timeout == 10)////////////////
        {
            Stop();
            return 1;
        }
    } while (ack);
    while (count)
```

```
    {
        SendByte(*buf);
        /*    Set_SDA_INPUT();    */
        Delay_Ms(DELAY_TIME);
        timeout = 0;
        do
        {
            ack = Check_ACK();
            timeout++;
            if (timeout == 10)
            {
                return 2;
            }
        } while (0);
        buf++;
        count--;
    }
    Stop();
    return 0;
}

//Read a byte of data through IIC
unsigned char GZP6827D_IIC_Read(unsigned char address, unsigned char *buf, unsigned char count)
{
    unsigned char timeout, ack;
    address |= 0x01;
    Start();
    SendByte(address);
    /*    Set_SDA_INPUT();    */
    Delay_Ms(DELAY_TIME);
    timeout = 0;
    do
    {
        ack = Check_ACK();
        timeout++;
```

```
            if (timeout == 10)
            {
                Stop();
                return 1;
            }
        } while (ack);
        Delay_Ms(DELAY_TIME);
        while (count)
        {
            *buf = ReceiveByte();
            if (count != 1)
                Send_ACK();
            buf++;
            count--;
        }
        Stop();
        return 0;
}

#include "IIC.h"
//*****************************************************
//MS Delay Function (Tested with 12M Crystal Oscillator)
//*****************************************************
void Delay_Ms(unsigned char n)
{
        unsigned char i,j;          //Change "char" to "int"
            for(i=0;i<n;i++)
                for(j=0;j<123;j++);
}

//Start signal
void Start(void)
{
        /*    Set_SDA_OUTPUT(); */
        SDA = 1;
```

```
        Delay_Ms(DELAY_TIME);
            SCL = 1;
                Delay_Ms(DELAY_TIME);
                    SDA = 0;
                        Delay_Ms(DELAY_TIME);
                            SCL = 0;
                                Delay_Ms(DELAY_TIME);//**
}

//Stop signal
void Stop(void)
{
    /*  Set_SDA_OUTPUT(); */
    SDA = 0;
        Delay_Ms(DELAY_TIME);
            SCL = 1;
                Delay_Ms(DELAY_TIME);
                    SDA = 1;
                        Delay_Ms(DELAY_TIME);
                            SCL = 0;                    //**
                                Delay_Ms(DELAY_TIME); //**
}

//Read ACK signal
unsigned char Check_ACK(void)
{
    unsigned char ack;
    /*  Set_SDA_INPUT();    */
        SDA = 1;                        //**
            Delay_Ms(DELAY_TIME); //**
                SCL = 1;
                    Delay_Ms(DELAY_TIME / 2);
                        ack = SDA;
                            Delay_Ms(DELAY_TIME / 2);
```