XGZP6895D PRESSURE SENSOR

FEATURES

- Wide Ranges: -100kPa···-0.5 ~ 0 ~ 0.5···700kPa(show in Pressure Range Example)
- 2.5V ~ 5.5V Power Supply
- Differential Pressure Type
- For Non-corrosive Gas or Air
- Calibrated Digital Signal(I2C Interface)(Refer to XGZP6899A for Analog signal)
- Temp. Compensated: 0℃ ~ +60℃(32℉ ~ +140℉)
- Temperature Measurable
- Affordable Cost, Easy-to-use

√ RoHS

APPLICATIONS

- Medical&Healthcare
- Industrial&Automation
- Domestic Appliance
- Consumer Electronic
- Automotive Electronic

INTRODUCTION

XGZP6895D is a prefect silicon pressure sensor offering a ratiometric digital data(I2C interface) for reading differential pressure over the specified full scale pressure span.

The XGZP6895D incorporates a silicon piezoresistive pressure sensor chip and an interior signal-conditional Application Specific Integrated Circuit(ASIC) in a SIP package with two air vents and long 6 PIN, which can be mounted freely from different direction .

The XGZP6895D is fully calibrated and temperature compensated for specified span, so XGZP6895D pressure sensor satisfy the perfect accuracy, which is designed for a wide range of application in medical&healthy, home appliances, consumer electronic, industry, automotive, IoT and other pneumatic devices etc by utilizing a microcontroller or microprocessor with I2C interface.

XGZP6895D pressure sensor is for high volume application at an affordable cost and perfect performance. Customized calibration parameter (e.g.pressure range etc.) are available.

PERFORMANCE SPECIFICATION

Unless otherwise specified, measurements were taken by 3.3Vdc with temperature of 25±1℃ and humidity from 25% ~ 85 % RH.

CHARACTERISTIC		MIN.	TYP.	MAX	UNIT
Available Pressure Range①		-100···-2.5 ~ 0 ~ 2.5···700			kPa
Power Supply②		2.5		5.5	Vdc
Current Consumption	Operating Current	-	1.7	-	mA
	Standby Current	-	100	-	nA
Output Resolution③	Pressure	24			Bit
	Temperature	16			Bit
Total Accuracy	10kPa < Pressure ≤200kPa	-	-	±2	%FSS
	Pressure ≤10kPa or > 200kPa	-	-	±2.5	%FSS
Temperature Accuracy		-1		1	℃
Long Term Stability(1000 hr, 25°C)		-	-	±0.5	%FSS
Over Pressure④	Pressure ≤5kPa	-	5X	-	FSS
	5kPa < Pressure ≤200kPa	-	2.5X	-	FSS
	200kPa < Pressure	-	-	1.5X	FSS
Burst Pressure④	Pressure ≤5kPa	-	10X	-	FSS
	5kPa < Pressure ≤200kPa	-	3X	-	FSS
	200kPa < Pressure	-	-	2X	FSS
Pressure on P2 Port(Common mode pressure)		-	-	1000	kPa
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)

Note: Port 1(P1) as High pressure cavity, and Port 2(P2) as Low pressure cavity..

PIN DEFINITION

PIN2	PIN3	PIN4	PIN6	PIN1/5/7/8
GND	VDD	SCL	SDA	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 ONLY

CONTACT CFSENSOR FOR ABOVE FILE IF REQUIRED

ORDER GUIDE (100kPa=0.1MPa=1bar≈14.5PSI)

XGZP6895 D 200 KP DG

- DG — Pressure Type(DG:Postive; DN: Negative; DPN:P+N)
- KP — Pressure Unit(KP: kPa HP:hPa)
- 200 — Pressure Range(000-999)
- D — Output Signal(A: Analog D:I2C)
- XGZP6895 — Series

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

PRESSURE RANGE EXAMPLE

Notes: 1. Unit conversion: 1000hPa=1000mbar≈750mmHg≈100kPa≈14.5PSI≈10mH_2O≈1bar=0.1MPa;
2. Available for more custom pressure range e.g. -7 ~ 7kPa, , 0-700kPa etc,.

Pressure Range (kPa)	Pressure Range (by other units)	Part Number
0 ~ 200	0 ~ 2bar	XGZP6895D200KPDG
★Above P/N is example only, consult CFSensor whether required pressure range is under normal production before place order.		

I2C INTERFACE

I2C bus uses SCL and SDA as signal lines. Both lines are connected to VDD externally via pull-up resistors(Typ value:4.7kΩ) so that they are pulled high when the bus is free. I2C device factory setting slave address: 0X6D.
The master device can communicate with the product using commands in the following format:

S	SlaveAddr	0	A	RegisterAddr	A	ByteToWrite	A	P

Write One Byte To One Register

S	SlaveAddr	0	A	RegisterAddr	A	S	SlaveAddr	1	A	ByteFromSlave	P

Read One Byte From One Register

ELECTRICAL SPEC. OF I2C INTERFACE PIN

Symbol	Parameter	Min	Max	Unit
fscl	Clock frequency		400	KHz
tLOW	SCL low pulse	1.3		us
tHIGH	SCL high pulse	0.6		us
tSUDAT	SDA setup time	0.1		us
tHDDAT	SDA hold time	0.0		us
tSUSTA	Setup Time for a repeated start	0.6		us
tHDSTA	Hold time for a start condition	0.6		us
tSUSTO	Setup Time for a stop condition	0.6		us
tBUF	Time before a new transmission	1.3		us

I2C TIME DIAGRAM

The I2C interface p e shown above.

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

REGISTER DESC.

Add.	Desc.	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Default
0x06	DATA_MSB	R	Pressure Data out<23:16>								0x00
0x07	DATA_CSB	R	Pressure Data out<15:8>								0x00
0x08	DATA_LSB	R	Pressure Data out<7:0>								0x00
0x09	TEMP_MSB	R	Temp Data out<15:8>								0x00
0x0A	TEMP_LSB	R	Temp Data out<7:0>								0x00
0x30	CMD	RW	Sleep_time<7:4>				Sco	Measurement_ctrl<2:0>			0x00
0xA5	Sys_config	RW	Aout_config<7:4>				LDO_config	Unipolar	Data_out_control	Diag_on	OTP
0xA6	P_config	RW		Input Swap	Gain_P<5:3>			OSR_P<2:0>			OTP

Reg0x06-Reg0x08 :

Pressure ADC data Register

Reg0x09 & Reg0x0A :

Temperature ADC data Register

Reg0x30 : Measurement Command Register

Sleep_time<7:4>: 0000:0ms; 0001:62.5ms; 0010:125ms ;... 1111: 1s, only active during sleep mode conversion.

Sco: 1: Start of conversion, automatically come back to 0 after conversion ends (except sleep mode conversion).

Measurement_control<2:0>:

010: indicate the combined conversion (namely a temperature conversion immediately followed by a pressure-signal conversion).

011: indicate a sleep mode conversion (periodically perform a combined conversion with an interval time of 'sleep_time'),

Reg0xA5(configured at factory)

Aout_config<7:4>: Analog output setting(recommending reserve default value)

LDO_config: 0: set with 1.8V;
1: set with 3.6V.

Polarity: 0: ADC output in bipolar format(signed binary),
1: ADC output in unipolar format. (Unsigned binary, Only take effect when 'raw_data_on' = 1)

Raw_data_on: 0: output calibrated data(as default value),
1: output ADC raw data.

Diag_on: 1, Enable diagnosis function(default).

Reg0xA6(configured at factory)

Input Swap: Swap VINP and VINN inside the ASIC

Gain_<5:3>: set the gain of the sensor signal conversion channel. 000: gain=1, 001: gain=2, 010: gain=4, 011: gain=8, 100: gain=16, 101: gain=32, 110: gain=64, 111: gain=128.

OSR_P<2:0>: set the over sampling ratio of the sensor signal conversion channel. 000:1024X, 001:2048X, 010:4096X, 011:8192X, 100:256X, 101:512X, 110:16384X, 111:32768X.

READ OPERATION

As the following instruction sequences for reading data:
(Only for combined conversion (namely a temperature conversion immediately followed by a pressure-signal conversion))

1. Send instructions 0x0A(combined conversion mode command) to 0x30 register to start conversion.
2. Read the 0x30 register address. If Sco bit is 0, signify the acquisition end, the data can be read.
   (Or, skip above step and delay 20mS and to read data directly)
3. Read 0x06, 0x07, 0x08 register address data to form a 24-bit AD value (pressure data AD value); Read 0x09, 0x0A register address data to form a 16-bit AD value (temperature data AD value)

Read Pressure

The total pressure output value which include 0x06, 0x07 and 0x08 registers are 24 bits. The highest position is the signed bit, and the value is "0", it represents "positive" pressure; The symbol digit value is "1", it represents "negative" pressure.
Pressure_ADC value: = (Pressure 3rd Byte [23:16] x 65536+Pressure 2nd Byte [15:8] x 256 + Pressure1st Byte [7:0])

Note: 1 Pressure 3rd Byte [23:16] is the hexadecimal value read out by REG0x06 and need convert into decimal value;
2 Pressure 2nd Byte [15:8] is the hexadecimal value read out by REG0x07 and need convert into decimal value;
3 Pressure 1st Byte [7:0] is the hexadecimal value read out by REG0x08 and need convert into decimal value.

For Pressure conversion formula are as follows:
The highest bit is "0", which means positive pressure, then Pressure = Pressure_ ADC/k;
The highest bit is "1", which means negative pressure, then Pressure=(Pressure_ADC-2^24)/k;

Pressure Range(kPa)	K value	Sensor Pressure Range Example
130<Pressure Range≤260	32	0~200kPa; -100~200kPa...
65<Pressure Range≤130	64	0~100kPa; -100~100kPa; -100~0kPa...
32<Pressure Range≤65	128	0~40kPa; -40~40kPa; -40~0kPa...
16<Pressure Range≤32	256	0~20kPa; -20~20kPa; -20~0kPa; -30~0kPa...
8<Pressure Range≤16	512	0~10kPa; -10~10kPa; -10~0kPa...
4<Pressure Range≤8	1024	0~5kPa; -5~5kPa; -1~1PSI...
2≤Pressure Range≤4	2048	0~2.5kPa; -2.5~2.5kPa...
1≤Pressure Range<2	4096	0~1kPa; -1~1kPa; -0.5~0.5kPa...

Note:
1. the unit is Pa (default value). If need to display other units, fill the corresponding factor in the conversion formula;
2. the K value is selected according to the absolute value span from 0kPa to the Max. pressure point. eg for -100 ~ 100kPa or -100 ~ 0kPa or 0 ~ 100kPa, the absolute value span from 0kPa to the Max. Pressure value. is 100kPa, so the K value is 64.

Read Temperature

The bits of temperature output values in the 0x09 and 0x0A registers are 16 bits, the highest is the sign bit.

The sign digit value is "0" when it represents "positive" temperature, and the sign digit value is "1" when it represents "negative" temperature.

Supposing if the decimal values of REG0x09 and REG0x0A readout are X, Y,

For Temperature ADC value and conversion formula are as:

Temperature AD value: N=X*256+Y

If n<2^15, Temperature is positive value, actual temperature T =N/256; (℃).

If n>2^15, Temperature is negative value, actual temperature value = (N-2^16) /256; (℃)

PACKING INFORMATION

Packing	Tray	Inner Box	Outer Box	Note
Quantity	45PCS per Tray	135pcs(3pcs Tray)	1080pcs(6pcs Inner Box)	Anti-static bag

Note: 1,The sensor should be stored in an ESD protective container before using them.

2, The packing information may be not quite same with above for other different quantity and samples.

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 printed circuit board 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

READ CODE(C51 Example)

```
#include <reg52.h>
#include <math.h>
#define DELAY_TIME 600
#define TRUE 1
#define FALSE 0
#define uchar unsigned char
#define uint unsigned int

//----define IIC SCL, SDA port----
sbit SCL = P1 ^ 7;
sbit SDA = P1 ^ 6;

//----delay time_us----
void DELAY(uint t)
{
	while (t != 0)
		t--;
}

//----IIC START CONDITION----
void I2C_Start(void)
{
	SDA = 1;	//SDA output high
	 DELAY(DELAY_TIME);
	SCL = 1;
	DELAY(DELAY_TIME);	//SCL output high
	SDA = 0;
	DELAY(DELAY_TIME);
	SCL = 0;
	DELAY(DELAY_TIME);
}

//----IIC STOP CONDITION----
void I2C_Stop(void)
{
	SDA = 0;	//SDA OUTPUT LOW
	DELAY(DELAY_TIME);
	 SCL = 1;
	DELAY(DELAY_TIME);
	SDA = 1;
	DELAY(DELAY_TIME);
```

```
    SCL = 0;    //SCL OUTPUT LOW
    DELAY(DELAY_TIME);
}

//----IIC SEND DATA "0"----
void SEND_0(void)
{
    SDA = 0;
     DELAY(DELAY_TIME);
    SCL = 1;
    DELAY(DELAY_TIME);
    SCL = 0;
    DELAY(DELAY_TIME);
}

//----IIC SEND DATA "1"----
void SEND_1(void)
{
    SDA = 1;
    DELAY(DELAY_TIME);
    SCL = 1;
    DELAY(DELAY_TIME);
    SCL = 0;
    DELAY(DELAY_TIME);
}

//----Check SLAVE's Acknowledge----
bit Check_Acknowledge(void)
{
    SDA = 1;
    DELAY(DELAY_TIME);
    SCL = 1;
    DELAY(DELAY_TIME / 2);
    F0 = SDA;
    DELAY(DELAY_TIME / 2);
    SCL = 0;
    DELAY(DELAY_TIME);
    if (F0 == 1)
        return FALSE;
    return TRUE;
}
```

```
//----Write One Byte of Data----
void WriteI2CByte(uchar b) reentrant
{
    char i;
    for (i = 0; i < 8; i++)
        if ((b << i) & 0x80)
            SEND_1();
        else
            SEND_0();
}

//----Read One Byte of Data----
uchar ReadI2CByte(void) reentrant
{
    char b = 0, i;
    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 == 1)
        {
            b = b << 1;
            b = b | 0x01;
        }
        else
            b = b << 1;
    }
    return b;
}

//----write One Byte of Data,Data from MASTER to the SLAVER----
void Write_One_Byte(uchar addr, uchar thedata)   //Write "thedata" to the SLAVER's address of "addr"
{
    bit acktemp = 1;
    I2C_Start();  //IIC START
    WriteI2CByte(0x6D << 1 + 0);          //IIC WRITE operation, SLAVER address bit: 0x6D
    acktemp = Check_Acknowledge();        //check the SLAVER
```

```
    WriteI2CByte(addr);          //address
    acktemp = Check_Acknowledge();
    WriteI2CByte(thedata);       //thedata
    acktemp = Check_Acknowledge();
    I2C_Stop();  //IIC STOP
}

//----Read One Byte of Data,Data from SLAVER to the MASTER----
uchar Read_One_Byte(uchar addr)
{
    bit acktemp = 1;
    uchar mydata;

    I2C_Start();
    WriteI2CByte(0x6D << 1 + 0);      //IIC WRITE operation, SLAVER address bit: 0x6D
    acktemp = Check_Acknowledge();
    WriteI2CByte(addr);
    acktemp = Check_Acknowledge();
    I2C_Start();
    WriteI2CByte(0x6D << 1 + 1);            //IIC READ operation, SLAVER address bit: 0x6D
    acktemp = Check_Acknowledge();
    mydata = ReadI2CByte();
    I2C_Stop();
    return mydata;
}

//----Delay_ms----
void Delay_xms(uint x)
{
    uint i, j;
    for (i = 0; i < x; i++)
        for (j = 0; j < 112; j++)
            ;
}

//----The main function----
void main(void)
{
    uchar pressure_H, pressure_M, pressure_L, temperature_H, temperature_L;
    //temporary variables of pressure and temperature
    long int pressure_adc, temperature_adc;
    //The value of pressure and temperature converted by the sensor's ADC
```

```
 long float pressure, temperature;
 //The calibrated value of pressure and temperature
 Delay_xms(1000);        //delay 1000ms

while (1)
{
    Write_One_Byte(0x30, 0x0A);
      //indicate a combined conversion (once temperature conversion immediately followed by once sensor signal
      conversion)
      //more measurement method, check Register 0x30
    while ((Read_One_Byte(0x30) & 0x08) > 0); //Judge whether Data collection is over, if above action has no
response, delete that command and execute this command as:  Delay_xms(20);

    pressure_H = Read_One_Byte(0x06);
    pressure_M = Read_One_Byte(0x07);
    pressure_L = Read_One_Byte(0x08);
      // Read ADC output Data of Pressure
    pressure_adc = pressure_H * 65536 + pressure_M * 256 + pressure_L;
      //Compute the value of pressure converted by ADC

      if (pressure_adc > 8388608)
          pressure = (pressure_adc - 16777216) / K; //unit is Pa, select appropriate K value according to pressure range.
    else
          pressure = pressure_adc / K; //unit is Pa, select appropriate K value according to pressure range.
      //The conversion formula of calibrated pressure, its unit is Pa

      temperature_H = Read_One_Byte(0x09);
    temperature_L = Read_One_Byte(0x0A);
    //Read ADC output data of temperature
      temperature_adc = temperature_H * 256 + temperature_L;
      //Compute the value of temperature converted by ADC
            temperature = (temperature_adc – 65536) / 256;        //unit is ℃
      else
            temperature = temperature_adc / 256;        //unit is ℃
      //The conversion formula of calibrated temperature, its unit is Centigrade

    Delay_xms(100);   //delay 100ms
}
}
```