PVC4000 Series

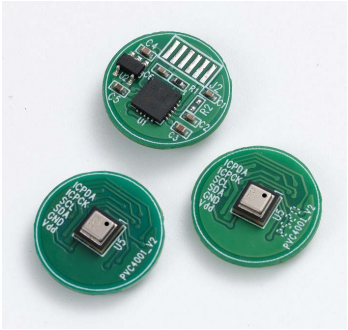
MEMS PIRANI VACUUM TRANSDUCER

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

DESCRIPTION

The PVC4000 Series is a family of vacuum transducers designed for cost-effective OEM integration. Devices in the series combine a MEMS thermal conduction sensor (Pirani sensor), measurement electronics, and a microprocessor in an ultra-compact PCB assembly. The sensing element is based on Posifa's second-generation MEMS thermal conduction chip, which operates on the principle that the thermal conductivity of gases is proportional to vacuum pressure. The electronics and microprocessor amplify and digitize the sensor signal and provide output via an I^2C interface. Because thermal conductivity varies with ambient temperature, the microprocessor applies a temperature compensation algorithm using input from a built-in temperature sensor. To minimize drift due to sensor self-heating, the microprocessor uses a pulsed excitation scheme in which the sensor is heated for about 100 ms and then turned off for one second. Output from the PVC4000 Series is uncalibrated; as an option, users can enter up to ten pairs of calibration points via I^2C, and a built-in piecewise linearization algorithm uses this data to provide calibrated output. To further facilitate integration, devices in the series include a connector-terminated wire harness soldered onto the PCB assembly.

The PVC4000 Series is offered in two device options. PVC4001 provides Pirani-based vacuum measurement over 10^{-3} to 760 Torr in an SMD-sensor configuration. PVC4001-C is the PVC4001 configuration with an onboard barometric pressure sensor added; because Pirani vacuum sensors typically lose resolution above 10 Torr, the barometric sensor supports measurement from 10 Torr to 760 Torr with 5% accuracy across that extended range.



PVC4001

PVC4001-C

FEATURES

- Range: 0.001 to 900 Torr (0.13 to 120K Pa)
- Fast Response Time < 200 ms
- Low power consumption for battery-powered instruments
- Temperature compensation
- Pulsed sensor excitation to prevent signal drift in high vacuum
- Piecewise linearization algorithm and I^2C interface for storing calibration data (optional)
- Resistant to contamination

APPLICATIONS

- Leak detection in any closed system maintained under primary vacuum, such as in Vacuum Insulated Panels
- Portable digital vacuum gauges

ABSOLUTE MAXIMUM RATINGS

- Operating Temperature: -25 to 85 °C
- Storage Temperature: -40 to 90 °C
- Shock: 100 g peak (5 drops, 3 axis)
- Overpressure: 27.5 bar

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

ELECTRICAL CHARACTERISTICS

Test Conditions: Vdd = 3.3 Vdc, Ta=21°C					
SPECIFICATIONS	**MIN**	**TYP**	**MAX**	**UNIT**	**CONDITIONS**
Range	1		900000	Micron	PVC4001-C
Output[1]	13500	17500	20000	Count	At 760000 Micron
	37000	40800	44000	Count	At 30 Micron
Sensitivity (Pirani)		8		Count/Micron	1 to 1000 Micron
		1.6		Count/Micron	1000 to 5000 Micron
		0.5		Count/Micron	5000 to 10000 Micron
		0.15		Count/Micron	10000 to 25000 Micron
Accuracy (Pressure)		5%			
Temperature Drift		5		Count/ °C	0 to 40 °C
Response Time		1.2		s	
Supply Voltage	2.7		5.5	Vdc	
Operating Current		11		mA	When heater is turned on
Operating Temperature Range	-25		85	°C	
Storage Temperature	-40		90	°C	

Note:
1. Pirani sensor output is not calibrated.

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

Pressure Sensor I²C Specification

The PVC4001-C has two different sensors, each covering a specific measurement range:

- Pressure sensor: 900 Torr to 10 Torr
- Pirani sensor: 10 Torr to 1 mTorr

Both sensors are accessed via the same I²C interface. The pressure sensor has an I²C address of 0x6D, while the Pirani sensor has an address of 0x50.

The I²C communication protocol for the pressure sensor is outlined below.

1 I²C COMMANDS

TYPE	DESCRIPTION	SUPPORT
Measurement Request (MR)	Wakes up the sensor, performs a sensor measurement, stores the sensor measurement data in internal registers, and returns to sleep	I²C
Get Data (GD)	Retrieves the sensor measurement data from the internal pressure sensor registers*	I²C

*Note: GD does not initiate a new measurement. Repeated GD commands will return the same (or stale) sensor measurement data. An MR is required to perform a full sensor measurement cycle to refresh the sensor register data.

The GD command is used to read out data from the pressure sensor. With the start of communication, the entire sensor measurement output packet will be loaded in a serial output register. The register will be updated after the communication is finished. The output is always scaled to 24 bits. The ordering of the bits is “big-endian.”

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

2 I²C COMMAND FORMAT

2.1 PC GET DATA (GD)

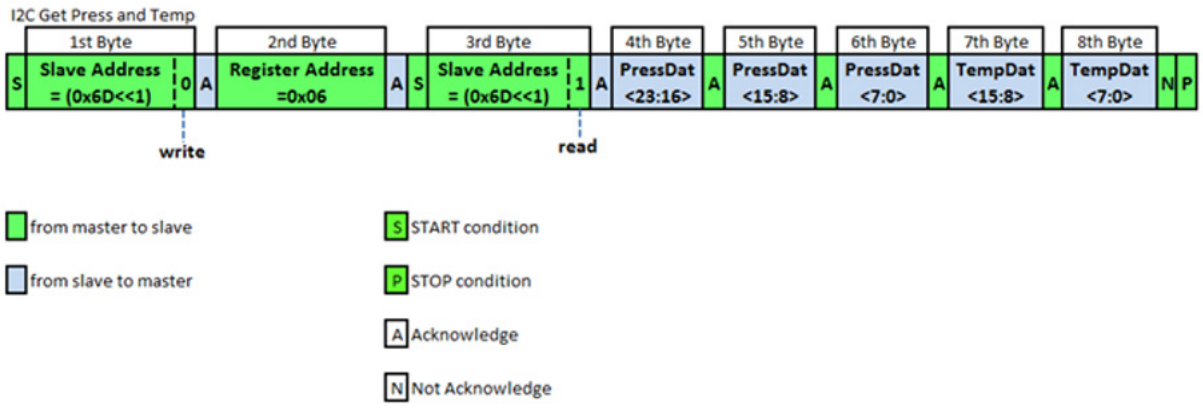
2.1.1 An I²C GD command starts with the 7-bit slave address and the 8th bit = 1 (READ). The device then sends acknowledge (ACK), indicating I²C communication success. The number of data bytes returned by the device is determined by the master, which controls NACK and stop conditions.

2.1.2 Figure 1 displays an example for sending three bytes followed by reading five bytes. The first byte contains the I²C address, followed by the internal register address (0x06). Then the I²C address is repeated, followed by the slave sending out three pressure bytes and two temperature bytes.

2.1.3 The GD command is used to retrieve the pressure and temperature sensor data after an MR command has been executed.

2.1.4 Note that the two temperature byte codes are formatted in 2's complement.

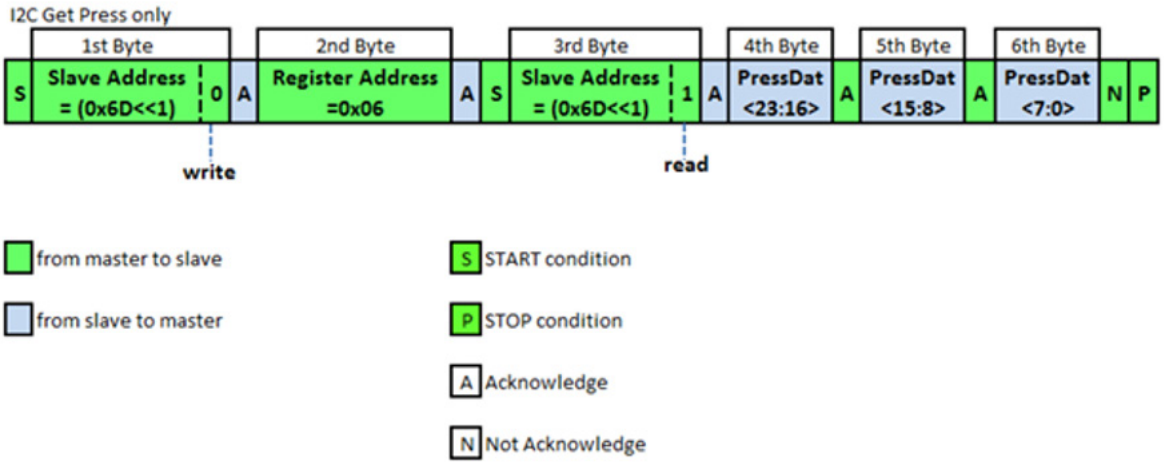
FIGURE 1: SLAVE ADDRESS FOLLOWED BY THREE PRESSURE AND TWO TEMPERATURE BYTES



For Pressure data only, the data stream can be terminated after thesixthpressure byte. See Figure 2below.

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

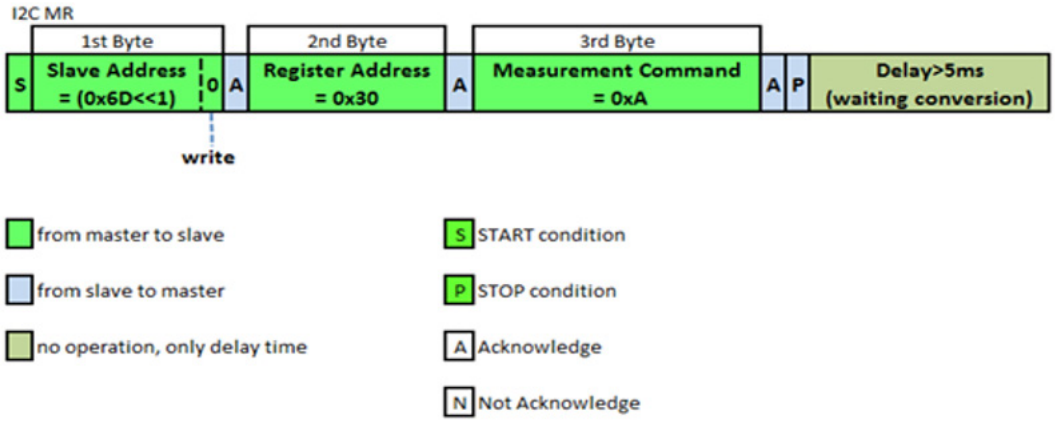
FIGURE 2: 7-BIT SLAVE ADDRESS FOLLOWED BY THREE PRESSURE BYTES



2.2 I²C MEASUREMENT REQUEST (MR)

2.2.1 The I²C MR is used to wake up the device from Sleep Mode and start a complete sensor measurement cycle, before the device returns to Sleep Mode again. The measurement cycles start with a temperature measurement, followed by a pressure measurement. The sensor measurements are digitized and run through an on-board compensation algorithm before the final measurement values are written to the digital output register. As shown in Figure 3, the communication requires the slave address (0x6D) and a WRITE bit (0) to initiate the MR. This is followed by two bytes: register address (0x30) and measurement (0xA). After the pressure responds with the slave ACK, the master terminates the communication with a stop condition.
Sensor measurement conversion time takes approximately 5 ms, so MRs should not be sent faster than every 5 ms.

FIGURE 3: I²C MEASUREMENT REQUEST COMMAND



131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

3 CALCULATING OUTPUT

After retrieving the data, the compensated output can be scaled to real-world values by following the equations below.

3.1 PRESSURE OUTPUT

An example of the 24-bit compensated pressure with a full-scale range of 30 kPa to 120 kPa can be calculated as follows:

Pressure [kPa] = (Pressure 3rd Byte [23:16] x 65536 + Pressure 2nd Byte [15:8] x 256 + Pressure 1st Byte [7:0]) / 2^6/1000

3.2 TEMPERATURE OUTPUT

The 16-bit compensated temperature can be calculated as follows:

Positive Temperature [°C] = (Temperature High Byte [15:8] x 256 + Temperature Low Byte [7:0]) / 2^8

Negative Temperature [°C] = (Temperature High Byte [15:8] x 256 + Temperature Low Byte [7:0]-65536) / 2^8

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

4 SAMPLE CODE FOR PRESSURE SENSOR:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using System.Windows;
using Dln;

static void Main()
{
        byte[] sendBytes = new byte[2];
        sendBytes[0] = 0x0a;
        maf.i2c_Write((byte)maf.i2c_ch_1, 0x6d, 0x30, sendBytes, 1);
        Thread.Sleep(100);

        byte[] recdata2 = maf.i2c_Read((byte)maf.i2c_ch_1, 0x6d, 0x06, 5);

        if (recdata2[0] != 0 || recdata2[1] != 0 || recdata2[2] != 0)
        {

                int Posfia_Pressure = (int)((recdata2[0] * 65536 + recdata2[1] * 256 + recdata2[2]) / 64);

        }
}


public void i2c_Write(ushort i2c_no, ushort address, byte memAddr, byte[] buffer, int numBytes)
{
   /        / int memAddr = 0;
        int memLen = 1;
        byte[] buf = new byte[numBytes];
         if (i2c == null)
         {
                return;
        }
        for (int i = 0; i < numBytes; i++)
         {
        buf[i] = Convert.ToByte(buffer[i]);
        }
         try
        {
                i2c.Write(address, memLen, memAddr, buf, numBytes);
        }
        catch
         {
                //i2c.Write(address, memLen, memAddr, buf, numBytes);
        }
}
```

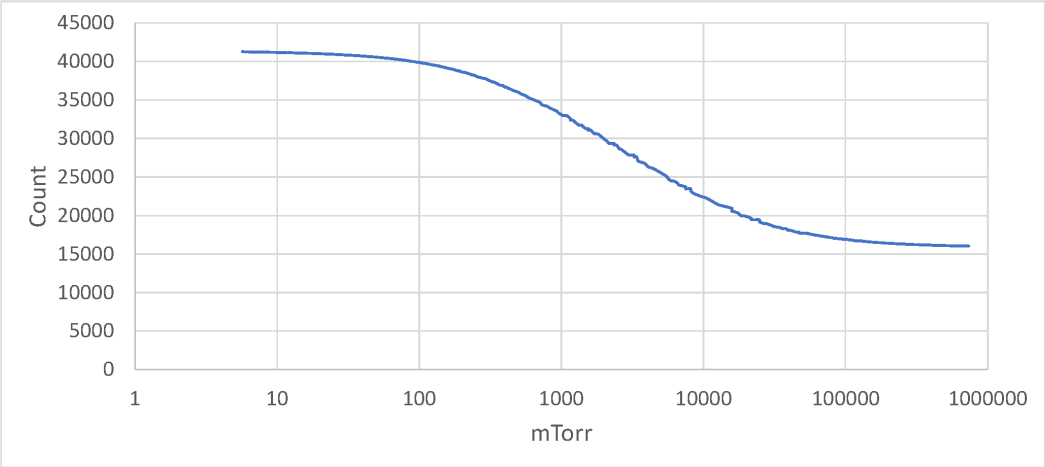
131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

```
public byte[] i2c_Read(ushort i2c_no, byte address, byte memAddr, int numBytes)
{
        // int  = 0;
        int memLen = 1;

        byte[] buffer = new byte[Rx_bytes];

        if (i2c == null)
        {
                return buffer;
        }
        try
        {
                i2c.Read(address, memLen, memAddr, buffer);
        }
        catch
        {
                Console.WriteLine("I2C Read Error.");
                buffer[0] = 255;
        }

        return buffer;
}
```

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

TYPICAL OUTPUT CURVE

Vdd = 3.3, Ta = 21°C



Pressure	Output		Pressure	Output		Pressure	Output
760000	17135		5493	24650		83	38630
300000	17280		3017	27350		46	39090
100000	17765		1657	30150		25	39360
24560	19670		910	32750		10	39560
10000	22300		500	35050		8	39595
7411	23450		204	37530		5	39630

LOW-POWER OPERATION

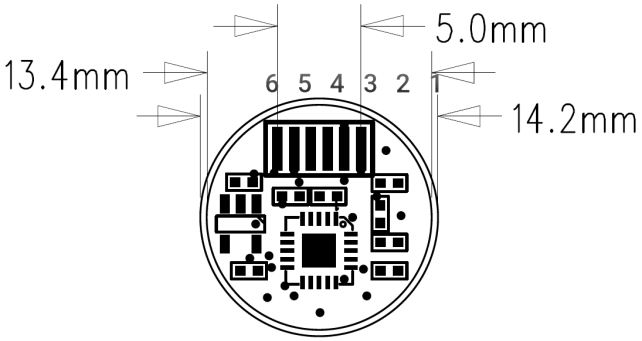
For battery-powered instruments that require further reduction in power consumption we recommend powering PVC4000 in an intermittent mode:

1. Turn on PVC4000
2. Wait for 150 milliseconds
3. Read from the I²C interface
4. Turn off PVC4000
5. Wait for 1 second, and repeat from Step 1

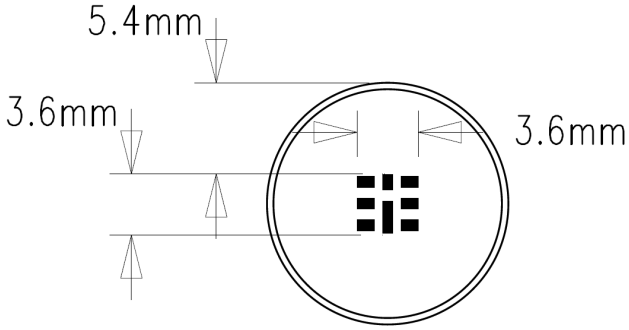
131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

PACKAGE DIMENSIONS

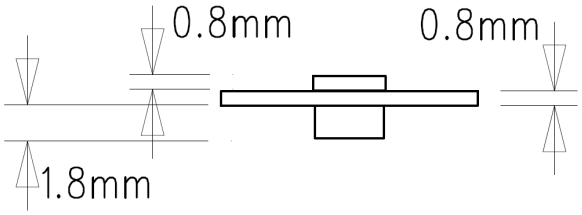
PVC4001



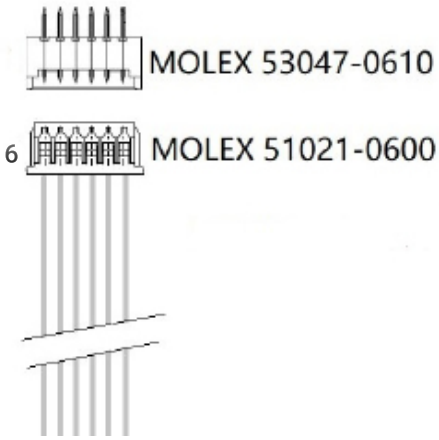
Top



Bottom



Side



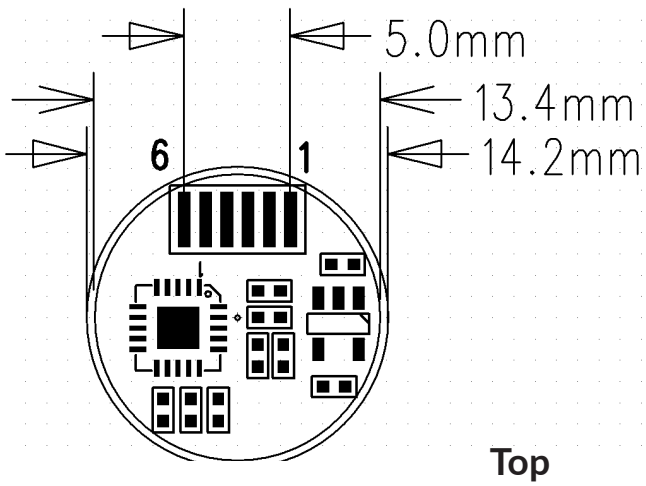
Pad/Wire#	Description (Wire Color)
1	Vdd (Red)
2	GND (Black)
3	SDA (Yellow)
4	SCL (Green)
5	ICPCK (for firmware update) (Blue)
6	ICPDA (for firmware update) (White)

Note: PVC4001 comes with a wire harness that is soldered onto the PCB. The length is 56mm (2.2 inch). The wire harness is terminated with a Molex connector P/N 51021-0600.

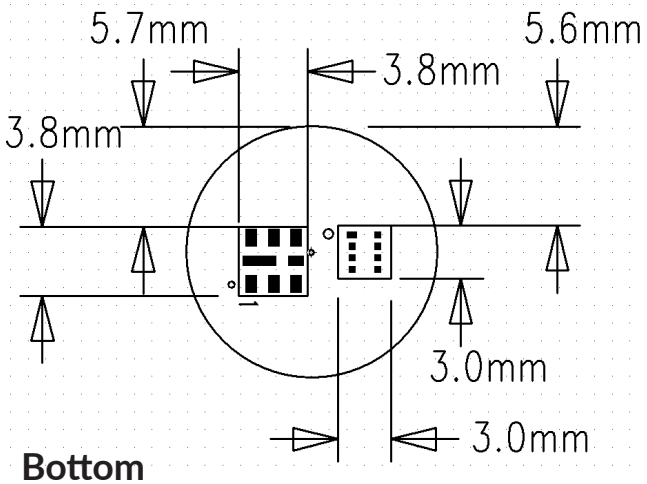
131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

PACKAGE DIMENSIONS

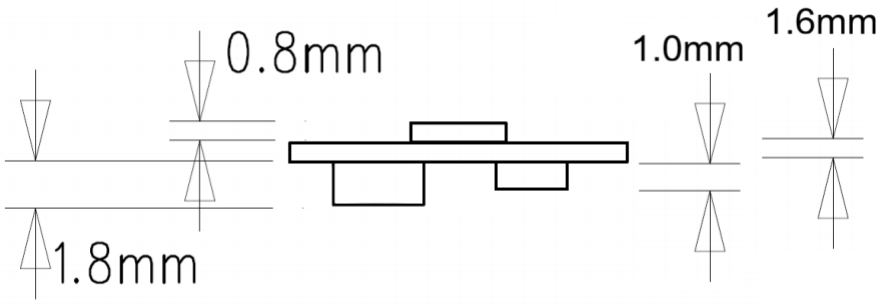
PVC4001-C



Top



Bottom



Side

Pad/Wire#	Description (Wire Color)
1	Vdd (Red)
2	GND (Black)
3	SDA (Yellow)
4	SCL (Green)
5	ICPCK (for firmware update) (Blue)
6	ICPDA (for firmware update) (White)

131 E. Brokaw Road | San Jose CA 95112 | +1 (408) 392-0989 | info@posifatech.com | www.posifatech.com

ORDERING INFORMATION

PART NUMBER	SPECIFICATIONS
PVC4001	10^{-3} to 760 Torr, SMD sensor
PVC4001-C	With on-board barometric pressure sensor

Please contact Posifa or your local distributor to place an order.