PTE7300

I²C PRESSURE SENSOR

Installation & Communication Guide

Description

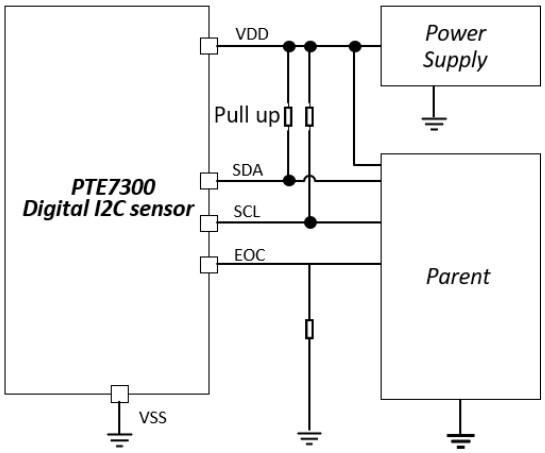
The PTE7300 pressure sensor is the ideal customer solution for challenging measuring requirements for general applications in the mid and high-pressure ranges. The PTE7300 features a wide range of ports and provides an I²C digital pressure output.

Electrical

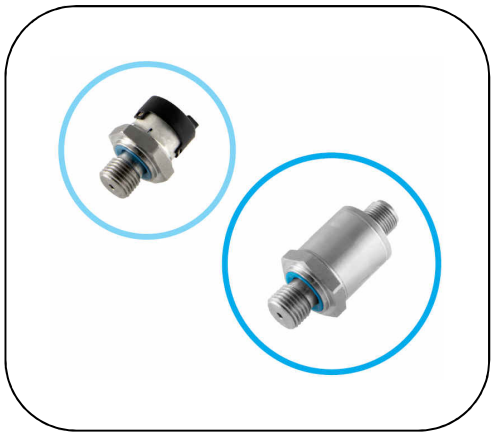
Pressure Ranges	0-16 bar to 0-600 bar (0-230 psi to 0-8700 psi)	
Pressure Reference	Gauge (Module) and Sealed Gauge (fully hermetic sensor)	
Supply Voltage	2.7VDC to 5.5VDC	
Digital Interface	I²C with CRC (memory integrity, and data transmission)	
Device Address	0xDA (including CRC) 0x6C (excluding CRC)	
Operating Current In Sleep Mode	6.5 uA (typical)	
Operating Current In Active Mode	3.7mA typical (4mA maximum)	
Available Data	Pressure	(int16, at address 0x30) min./max. of span mapped to ±16000 bit value
	Bridge temperature*	(int16, at address 0x2E) -40°C..+125°C mapped to ±16000 bit value
	Status synchronized	(uint16, at address 0x32)
	Device serial	(uint32, at address 0x50 (low word), and address 0x52 (high word))
	Note: Sensata offers a C++ library available on request that is written on Arduino platform, which allows for plug-n-play sensor control and is easy to port to other platforms.	
Resolution	15 bit	
Response Time (default)	< 1 ms	
Probe Configurations	Continuous or On-demand / single cycle	

Note: the bridge temperature is not part of the production quality control routines. Sensata will not guarantee any performance ratings for the sensor's temperature output. For accurate and reliable temperature measurement, it's advised to use a different sensor with qualified temperature output.

Typical Application Circuit



I²C output



PTE7300 SERIES

Note: the pull-up resistor resistance values and cable length highly depend on the application. Typical pull-up resistors are 4.7kOhm for a cable between 1-2 meters. Please note that, in general, a low internal resistance and low internal capacitance cable will increase the action radius since, in that case, the voltage is dropping less over distance. Also, lowering the resistance values increases the current which sharpens the edges of the digital communication signals, but at cost of power consumption when communicating with the device.

Contents

1 PIN ASSIGNMENTS

Figure 1. Pin Assignments – Top View

2 PIN DESCRIPTIONS

Table 1. Pin Descriptions

Pin	Pin Name	Type	Description
1	ALARM	Output	End of conversion or status output
2	VSUPPLY	S	Supply voltage
3	GND	S	Ground
4	SDA	Input/output	I²C serial data in/output as fixed function.
5	SDC	Input/output	I²C clock input as fixed function.

M12x15-POLE

Pin Number	Description
1	(ALARM)
2	VSUPPLY
3	GND
4	SDA
5	SDC

MODULE

Pin Number	Description
1	(ALARM)
2	VSUPPLY
3	GND
4	SDA
5	SDC

3 ABSOLUTE MAXIMUM RATINGS

The absolute maximum ratings are stress ratings only. The sensor might not function or be operable below and above the recommended operating conditions given in Table 2. Stresses exceeding the absolute maximum ratings will change the sensor accuracy; lead to imprecision, and eventually cause irreversible damage to the device. In addition, extended exposure to stresses above the recommended operating conditions might affect device reliability. Sensata does not recommend designing to the “Absolute Maximum Ratings.”

Table 2. Absolute Maximum Ratings

Symbol	Parameter	Minimum	Maximum	Units
VSUPPLY	Maximum Analog Supply Voltage	-0.3	6	V
VD_IO	Maximum Voltage at all Digital I/O Pins	-0.3	VSUPPLY+0.3SUPPL But ≤ 6	V
TOPERATION	Operation Temperature	-40	100	°C
TSTOR	Storage Temperature	-40	125	°C

4 I²C INTERFACE

Table 3. Operating Conditions

No.	Description	Symbol	Min	Typ	Max	Unit
1	I²C clock frequency	I²C _F_SCK			400	kHz
2	I²C clock low time (tLO)	I²C_T_LO	1300			ns
3	I²C clock high time (tHI)	I²C_T_HI	600			ns
4	I²C (repeated) start condition hold time (tSH)	I²C_T_SH	600			ns
5	²C data setup time (tSU)	I²C_T_SU	100			ns
6	I²C data hold time (tH)	I²C_T_H	0			ns
7	I²C repeated start setup time	I²C_T_RSH	600			ns
8	I²C stop condition setup time (tLO)	I²C_T_PSU	600			ns
9	I²C rise time (tR)	I²C_T_R			300	ns
10	I²C fall time (tF)	I²C_T_F			300	ns
11	I²C bus free time between STOP and START conditions (tBUF)	I²C_T_BUF	600			ns
12	Pulse width of spikes suppressed by the I²C input filter*)	I²C_T_SSP	0		500	ns

The I²C child interface provides direct access to the entire memory map of the IC. An external I²C parent (e.g. a µC) can read and write all memory addresses (registers) of the device using the following commands:

• **Random write:** Sets a memory address and writes data to consecutive memory addresses of the device starting at the set memory address.

• **Random read:** Sets a memory address and reads data from consecutive memory addresses of the device starting at the set memory address.

• **Read last:** Reads data from the device starting at the last memory address set by the parent. This facilitate s repeated reading of the same memory addresses without transmitting a memory address first.

All read/writes must start at **word aligned addresses** (i.e. LSB of memory address equals 0) and read/write an **even number of bytes**. Maximum length for CRC protected read/writes is 4 bytes.

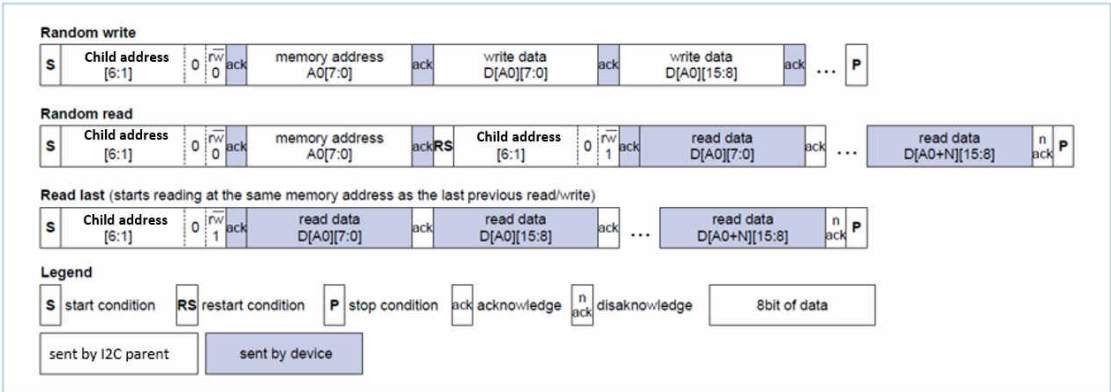
4.1 Device Address

Table 4. Address Definition

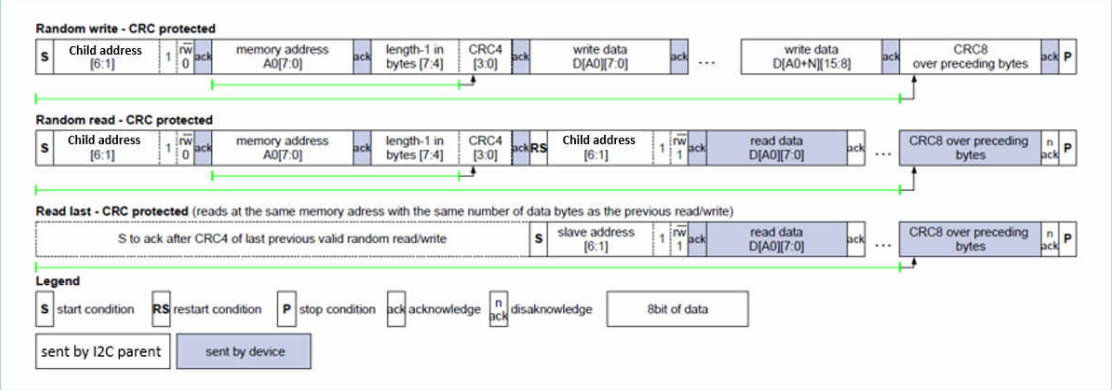
Device address	0XDA	Including CRC
	0X6C	Excluding CRC

4.2 Command Format

Commands of I²C EEPROM compatible protocol



Commands of CRC protected protocol



Example:

I²C random read example

Byte #	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*
Send By Parent (SBP)	0xD8	0x2E	0xD9						
SBP Comment	1101100 0 1101100: Child address [6:0] => 0x6C with LSB=0 => no CRC 0: r/$\overline{w}$ => write	memory address	1101100 1 1101100: Child address [6:0] => 0x6C with LSB=0 => no CRC 1: r/$\overline{w}$ => read						
Send By Device (SBD)				0xF2	0x7D	0xEA	0x82	0x1E	0x00
SBD Comment				0x2E: DSP_T [7:0]	0x2E: DSP_T [15:8]	0x30: DSP_S [7:0]	0x30: DSP_S [15:8]	0x32: STATUS_ SYNC [7:0]	0x32: STATUS_ SYNC [15:8]

I²C random write example

Byte #	0	1	2	3
Send By Parent (SBP)	0xD8	0x36	0x9E	0xCF
SBP Comment	1101100 0 1101100: Child address [6:0] => 0x6C with LSB=0 => no CRC 0: r/w => write	memory address	written to STATUS[7:0]	written to STATUS[15:8]
Send By Device (SBD)				
SBD Comment				

I²C random read with CRC example

Byte #	0	1	2	3	4	5	6	7	8
SBP (Send By Parent)	0xDA	0x2E	0x31	0xDB					
SBP comment	1101101 0 1101101: Child address [6:0] => 0x6D with LSB=1 => with CRC 0: r/w => write	memory address	3: length-1 = 4Byte y: CRC4	1101101 1 1101101: Child address [6:0] => 0x6D with LSB=1 => with CRC 1: r/w => read					

Byte #	0	1	2	3	4	5	6	7	8
SBS (sent by sensor)					0xF2	0x7D	0xEA	0x82	0xE4
SBS comment					0x2E: DSP_T [7:0]	0x2E: DSP_T [15:8]	0x30: DSP_S [7:0]	0x30: DSP_S [15:8]	CRC8

I²C random write with CRC example

Byte #	0	1	2	3	4	5
Send By Parent (SBP)	0xDA	0x36	0x16	0x9E	0xCF	0xA1
SBP Comment	1101101 0 1101101: Child address [6:0] => 0x6D with LSB=1 => with CRC 0: r/w => write	memory address	1: length-1 => 2 bytes to write 6: CRC4	written to STATUS[7:0] after CRC8	written to STATUS[15:8] after CRC8	CRC8
Send By Device (SBD)						
SBD Comment						

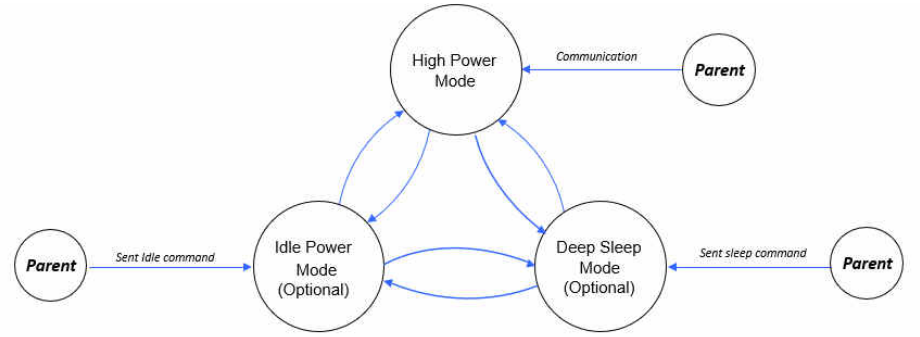
NOTE: the CRC4 and CRC8 fields are computed in the same bit and byte order as the transmission over the bus. Their polynomials are:

- CRC4 polynomial 0x03, initialization value 0x0F
- CRC8 polynomial 0xD5, initialization value 0xFF

If the CRC4 sent by the I²C parent does not match the CRC4 calculated by the I²C child, the I²C frame is aborted by not sending an acknowledge bit from the child after the CRC4. Additionally, the CRC4 error is reported as the event bit 11 at the STATUS register. If a command is aborted by a CRC4 error, no 'read last' command must follow.

4.3 System Operation Modes and State Diagram

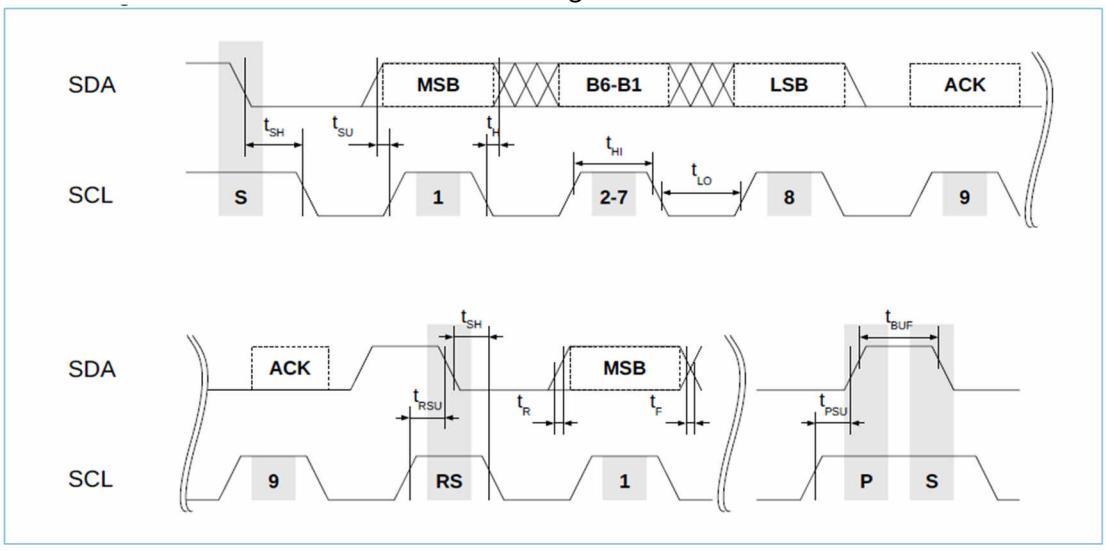
System balances performance and current consumption by different operation mode configurations. The Figure shows the Sensor state diagram.



Command sent to register 0x22	Description
0x7BBA (**Idle**)	Puts the device into power state idle. Aborts any conversion in progress. The timer counting down the period will be reset.
0x8B93 (**Start**)	Starts a measurement and put the chip into power state run.
0xB169 (**Reset**)	Performs a rest with complete power up sequence
0x6C32 (**Sleep**)	Enter the power state sleep.

4.4 I²C Bus Timing

I²C Bus Timing Behavior



5 PLUG-N-PLAY C++ CLASS LIBRARY

5.1 Installing the library (Arduino IDE)

The PTE7300_I2C library folder should be copied to the local user's \Documents\Arduino\libraries\ folder to be visible inside Arduino IDE. Example sketches are added for guidance.
The C++ library can also easily be ported to other platforms by rewriting only the constructor and the low-level readRegister and writeRegister methods source code, to modify the interface driver. The higher-level methods remain the same.

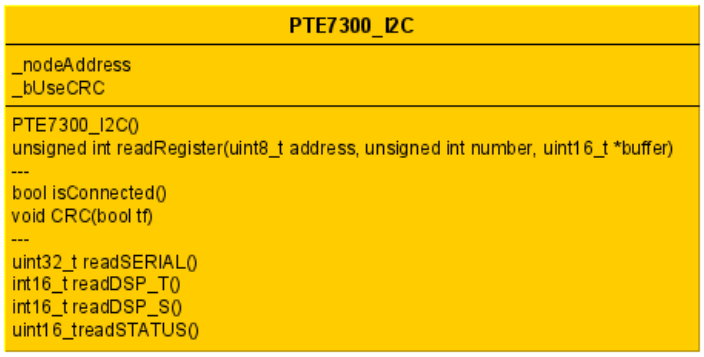
5.2 Example Arduino code – reading pressure output

readDSP_S §

```
#include <PTE7300_I2C.h>

PTE7300_I2C mySensor; // attach sensor
int16_t DSP_S; // allocate memory for pressure register value

void setup() {
  // initiate serial communication
  Serial.begin(9600);
}

void loop() {
  // first check is the sensor is connected
  if(mySensor.isConnected()){
      DSP_S = mySensor.readDSP_S(); // read pressure register value
      Serial.println(DSP_S); // print value to serial bus
  }
  else{
    Serial.println("Device not connected");
  }

  delay(100); // wait for 100ms
}
```

1. Create class instance 'mySensor'
2. Reserve signed 16 bit integer memory for the pressure value
3. Initialize serial interface to readout printed messages on the USB interface
4. Check if the sensor communication is ok, using the 'isConnected' method
5. Readout the pressure value ('mySensor.readDSP_S' method) and store the value in the pre-allocated memory of the microchip
6. Print the value on the USB interface to readout on the connected PC
7. If no device is connected, do nothing, and notify the PC that no device has been found

5.3 Class description (UML)



The constructor will automatically set the nodeAddress attribute to the default value. A 'bUseCRC' attribute is a boolean flag that is by default true, which controls the CRC checking state on the communication messages. The CRC checking feature can be disabled by means of the 'CRC' method if needed.
The 'isConnected' method is a simple and useful feature to check if the device is responding, prior to communicating with the device, to prevent bus failures or blocking drivers that wait forever.
The last set of methods is used to readout information from the device.

6 REVISION HISTORY

Revision Date	Description of Change
March 18, 2021	Preliminary version.

Americas
+1 (800) 350 2727
sensors@sensata.com
switches@sensata.com
Europe, Middle East & Africa
+359 (2) 809 1826
pressure-info.eu@sensata.com
Asia Pacific
sales.isasia@list.sensata.com
China +86 (21) 2306 1500
Japan +81 (45) 277 7117
Korea +82 (31) 601 2004
India +91 (80) 67920890
Rest of Asia +886 (2) 27602006 ext 2808