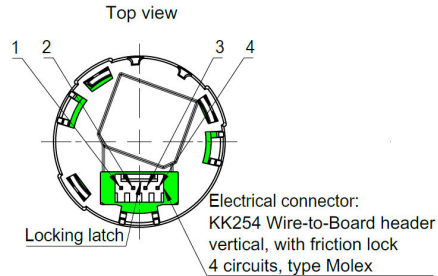


:: ELECTRICAL INTERFACE ::

- Pin 1: 0 V**
- Pin 2: SDA**
- Pin 3: SCL**
- Pin 4: Between +3.3V to +5.0V**



:: GENERAL INFORMATION ::

I2C_Start, I2C_Stop, I2C_Sendbyte and I2C_Readbyte are functions defined by the programming language. For more information see datasheet of I2C-device and programming library of your I2C-Master device.

:: READ OUT SENSOR DATA ::

Address (write to device): 0xDC
Address (read from device): 0xDD
ADC type (see datasheet): MCP3427

No.	Step	Description	Example (pseudo code)	Result
#1	Configure	Send configuration byte (0x98) after bus connection of sensor or power on reset	I2C_Start() I2C_Sendbyte(0xDC) I2C_Sendbyte(0x98) I2C_Stop()	ADC is configured
#2	Read Out	Read sensor signal as 3 Byte repetition	I2C_Start() I2C_Sendbyte(0xDD) I2C_Readbyte(ACK) I2C_Readbyte(ACK) I2C_Readbyte(NACK) I2C_Stop()	3 Byte repetition until NACK send by master: Upper Data Byte (8 Bit); Lower Data Byte (8 Bit); Configuration Byte (8 Bit)
#3	Control Option	Compare (Bit 0-6) of configuration byte from read out (#2) to configuration byte 0x98 (#1)	-	Read out is performed correct

Important note: The read out value (#3, Upper Data Byte and Lower Data Byte) is an INT(16) number and may not be treated as INT(32). Calculation example for INT(16): FF1A = -230.

.: READ OUT TEMPERATURE .:

Address (write to device): 0x36
 Address (read from device): 0x37
 Temperature sensor type (see datasheet): MCP9808

No.	Step	Description	Example (pseudo code)	Result
#1	Select and Read Out	Read out 2 Bytes from temperature device [Bit 4 – 7 (Upper Data Byte) = flag bits]	<pre> I2C_Start() I2C_Sendbyte(0x36) I2C_Sendbyte(0x05) I2C_Start() I2C_Sendbyte(0x37) I2C_Readbyte(ACK) I2C_Readbyte(NACK) I2C_Stop() </pre>	<p>Temperature device is activated and 2 Bytes received from temperature register</p> <p>Bit 7 (Upper Data Byte): Critical temperature (not used) Bit 6 (Upper Data Byte): Upper limit (not used) Bit 5 (Upper Data Byte): Lower limit (not used) Bit 4 (Upper Data Byte): Algebraic sign Bit 0 - 3 (Upper Data byte) + Bit 0 - 7 (Lower Data Byte): ambient temperature bits</p> <p>Clear flag bits: Upper Data Byte = Upper Data Byte & 0x1F</p> <p>Temperature $T_A \geq 0^\circ\text{C}$ [Bit 4 (Upper Data Byte) = 0] $T_A = (\text{Upper Data Byte} \times 2^4 + \text{Lower Data Byte} \times 2^{-4})$</p> <p>Temperature $T_A < 0^\circ\text{C}$ [Bit 4 (Upper Data Byte) = 1] $T_A = 256 - (\text{Upper Data Byte} \times 2^4 + \text{Lower Data Byte} \times 2^{-4})$</p>

.: READ OUT EEPROM DATA .:

Address (write to device): 0xA6
 Address (read from device): 0xA7
 EEPROM type (see datasheet): 24LC32A

Example for read out the first two Bytes (address: 0x00)

No.	Step	Description	Example (pseudo code)	Result
#1	Select and Read Out	Send selected EEPROM register address as High Byte and Low Byte to EEPROM device and read out EEPROM content until NACK send by master	<pre> I2C_Start() I2C_Sendbyte(0xA6) I2C_Sendbyte(0x00) I2C_Sendbyte(0x00) I2C_Start() I2C_Sendbyte(0xA7) I2C_Readbyte(ACK) I2C_Readbyte(NACK) I2C_Stop() </pre>	<p>EEPROM register address is selected and get back EEPROM content (Bytes) until NACK send by master</p>

The EEPROM content (e. g. serial number, manufacturing data, sensitivity, ambient measurement conditions, temperature and humidity coefficients and **data types**) depends on individual agreements with customer (see EEPROM document).

.: READ OUT BATTERY VOLTAGE :.

Address (write to device): 0xDC
Address (read from device): 0xDD
ADC type (see datasheet): MCP3427

No.	Step	Description	Example (pseudo code)	Result
#1	Configure	Send configuration byte (0xA8) after bus connection of sensor or power on reset	<pre>I2C_Start() I2C_Sendbyte(0xDC) I2C_Sendbyte(0xA8) I2C_Stop()</pre>	ADC is configured
#2	Read Out	Read sensor signal as 3 Byte repetition until (Byte 3) bitwise AND 0x80 == 0	<pre>I2C_Start() I2C_Sendbyte(0xDD) I2C_Readbyte(ACK) I2C_Readbyte(ACK) I2C_Readbyte(NACK) I2C_Stop()</pre>	3 Byte repetition until NACK send by master: Upper Data Byte (8 Bit); Lower Data Byte (8 Bit); Configuration Byte (8 Bit) verify (Configuration Byte) bitwise AND 0x80 == 0
#3	Calculate	Use Upper Data Byte (HByte) and Lower Data Byte (LByte) to calculate battery voltage	-	Battery voltage [µV]

.: CALIBRATION AND MEASUREMENT :.

- Supply sensor with zero gas (contains no NO₂, e.g. N₂ or scrubbed air) and record sensor signal as baseline (temperature, humidity, pressure and flow constant). Convert Upper Data Byte and Lower Data Byte of sensor signal to decimal number (0 to 32768).
- Supply sensor with calibration gas (e. g. 1ppm NO₂ bal. N₂) and record sensor signal as calibration signal (no change in temperature, humidity, pressure and flow). Convert Upper Data Byte and Lower Data Byte of sensor signal to decimal number (0 to 32768).
- Calibration signal minus baseline (=span) as decimal number equals calibration gas concentration (e. g. 1ppm NO₂ bal. N₂) The sensitivity of the sensor can be calculated.
- Measure other test gases and Read Out sensor output. Change of decimal number is linear to change of NO₂ concentration over full scale if measurement and calibration conditions are the same (repeat calibration if measurement conditions differ from calibration conditions).

Example:

1-3. Calibration	Baseline (100 Vol.%N ₂):	Sensor output: 200 Digits	
	Calibration gas (1ppm NO bal. N ₂):	Sensor output: 5200 Digits	
	Span and sensitivity (20ppm NO):	Span = 5200 -200 Digits = 5000 Digits	Sensitivity = 5000 Digits / 1ppm
4. Measurement	Test gas (3ppm NO bal. N ₂):	Span: 15000 Digits	5000 Digits / 1ppm * 3ppm = 15000 Digits
Measurement	Test gas (0.1ppm NO bal. N ₂):	Span: 500 Digits	5000 Digits / 1ppm * 0.1 ppm= 500 Digits