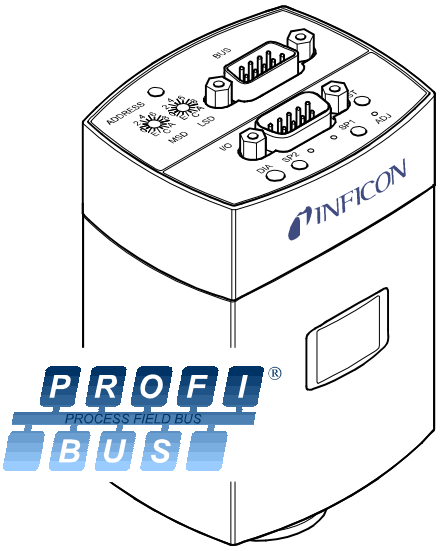
# Profibus

DP/V1 Interface for Pirani Capacitance Diaphragm and Pirani Standard Gauges

## PCG550 PCG552 PCG554
## PSG550 PSG552 PSG554



## General Information

**Caution**

Caution: data transmission errors

Any attempt to simultaneously operate the gauge via the RS232C Serial Interface and a Fieldbus interface (DeviceNet or Profibus) or the diagnostic port may result in incorrect data and data transmission errors.

Therefore, it is inadmissible to simultaneously operate the gauge via the RS232C Serial Interface and DeviceNet, Profibus, or the diagnostic port.

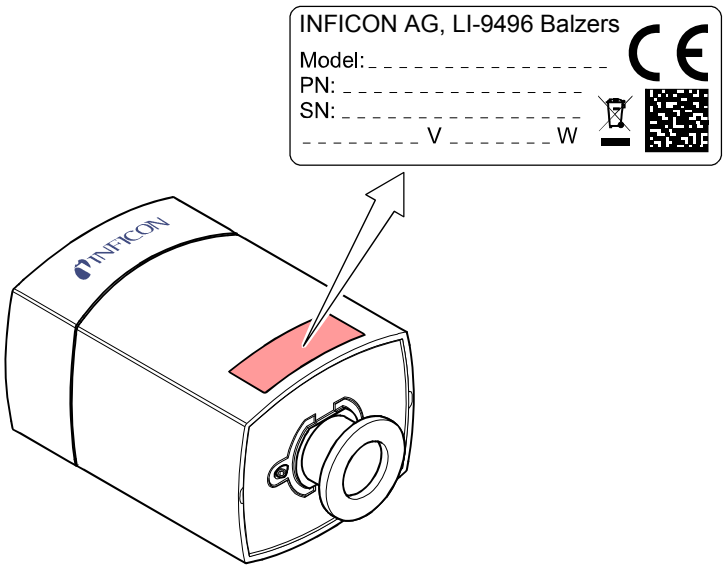
## About this Document

This document describes the functionality and programming of the Profibus interface of the Pirani Capacitance Diaphragm Gauges (PCG550, PCG552, PCG554) and the Pirani Standard Gauges (PSG550, PSG552, PSG554).

For safety information on and further technical data of the gauges, please refer to the respective operating manuals (→ 🕮 [1], [2]).

## Product Identification

In all communications with INFICON, please specify the information on the product nameplate. For convenient reference copy that information into the space provided below.



## Validity

This document applies to products of the Pirani Capacitance Diaphragm (PCG550, PCG552, PCG554) and Pirani Standard Gauges (PSG550, PSG552, PSG554) with Profibus interface.

Part numbers of standard products are indicated below. OEM products have other part numbers and different parameter settings (e.g. factory setting of setpoint) as defined in the corresponding ordering information.

**3Pxx-0xx-xx2x**

- Measuring signal
  - 0 ⇒ 0.61 ... 10.23 V
  - 1 ⇒ 1.2 ... 8.68 V
  - 2 ⇒ 0.375 ... 5.659 V
  - 3 ⇒ 1.57 ... 9.05 V
- Receptacle
  - 0 ⇒ FCC
  - 1 ⇒ D-Sub, 9-pin
  - 2 ⇒ D-Sub, 15-pin HD
  - 4 ⇒ D-Sub, 15-pin HD, RS485 INF
  - 5 ⇒ Hirschmann, Type GO 6
- Display Switching function
  - 0 ⇒ None
  - 1 ⇒ Display
  - 2 ⇒ 2 switching functions *)
  - 3 ⇒ ATM sensor & 2 switching functions *)
  - 4 ⇒ Display & 2 switching functions *)
  - 5 ⇒ Display & ATM sensor & 2 switching functions *)
  - 6 ⇒ 2 switching functions **)
- Flange
  - 1 ⇒ DN 16 ISO-KF
  - 2 ⇒ DN 16 ISO-KF long tube
  - 4 ⇒ DN 16 CF-F
  - 5 ⇒ DN 16 CF-R long tube
  - 6 ⇒ DN 25 ISO-KF
  - D ⇒ 4 VCR female
  - E ⇒ 8 VCR female
  - F ⇒ 1/8" NPT
  - K ⇒ Flange 29×29 mm
  - M ⇒ 4 VCR female 90°
  - N ⇒ 7/16-20 UNF male
- Unit
  - 0 ⇒ mbar
  - 1 ⇒ Torr
  - 2 ⇒ Pa
  - 3 ⇒ micron
- Filament
  - 1 ⇒ Tungsten
  - 2 ⇒ Nickel
  - 3 ⇒ Ceramic coated
  - 6 ⇒ Tungsten, galvanically isolated
  - 7 ⇒ Nickel, galvanically isolated
  - 8 ⇒ Ceramic coated, galvanically isolated
- Product family
  - C ⇒ Pirani Capacitance Diaphragm Gauge
  - I ⇒ Pirani Standard Gauge

*) Solid state relay
**) Electromechanical relay

The part number (PN) can be taken from the product nameplate.

If not indicated otherwise in the legends, the illustrations in this document correspond to PCG550 gauges with the DN 16 ISO-KF vacuum connection. They apply to other vacuum connections and to the other gauges by analogy.

We reserve the right to make technical changes without prior notice.

## Trademark

SEMI® Semiconductor Equipment and Materials International, California

# Contents


General Information 2
About this Document 2
Product Identification 2
Validity 3
Trademark 3

**1 Technical Data 7**

**2 Interface Connection 9**

**3 General Data 10**
3.1 Introduction 10
3.2 Operating Software 10
3.3 Node Address 10
3.4 "NET" LED 11
3.5 Data Rate 11
3.6 Ident Number 11
3.7 Configuration Data 12
3.8 User Parameter Data 12
3.9 Types of Communication 13

**4 Data Exchange Mode 14**
4.1 Acyclic Data Transmission with Profibus DPV1 Functionality 14
4.2 Cyclic Data Telegrams in Data Exchange Mode 15
4.3 Parameter Channel 16
4.3.1 PKE (parameter signature value) 16
4.3.2 PWE (process value) 17
4.3.3 Error code 17
4.3.4 Sequence Diagram 18
4.4 Process Data 19
4.4.1 Standard Telegrams 19
4.4.2 Configuration Data 20

**5 I&M Identification & Maintenance Functions 21**
5.1 I&M0 21

**6 Block Model 22**
6.1 Block types 22

**7 Device Block 23**
7.1 Information on the individual Indices 24
7.1.1 Block Type ID (ID 16) 24
7.1.2 Device Type (ID 17) 24
7.1.3 Standard Revision Level (ID 18) 24
7.1.4 Device Manufacturer Identifier (ID 19) 24
7.1.5 Manufacturer Model Number (ID 20) 24
7.1.6 Software or Firmware Revision Level (ID 21) 24
7.1.7 Hardware Revision Level (ID 22) 24
7.1.8 Serial Number (ID 23) 24
7.1.9 Device Configuration (ID 24) 25
7.1.10 Device State (ID 25) 25
7.1.11 Exception Status (ID 26) 26
7.1.12 Exception Detail Alarm (ID 27) 27
7.1.13 Exception Detail Warning (ID 28) 29
7.1.14 Run hours (ID 36) 30
7.1.15 Basic Device Firmware Revision Level (ID 202) 30
7.1.16 Common Exception Detail Alarm (ID 204) 30
7.1.17 Device Exception Detail Alarm (ID 205) 30
7.1.18 Manufacturer Exception Detail Alarm (ID 206) 30
7.1.19 Common Exception Detail Warning (ID 207) 30
7.1.20 Device Exception Detail Warning (ID 208 / ID 209) 30
7.1.21 Manufacturer Exception Detail Warning (ID 210) 31
7.1.22 Address Switch (ID 211) 31
7.1.23 Run hours x4 (ID 212) 31
7.2 Device Block Status diagram 31
7.2.1 General Device Block State descriptions 32
7.2.2 Device Block State Command (ID 15) 33

**8 Sensor Analog Input Function Blocks 34**
8.1 Sensor Analog Input Function Block (CDG) 34
8.1.1 Analog Input Block Adjust Command (ID 15) 35
8.1.2 Block Type ID (ID 16) 35
8.1.3 Process Value (ID 19) 35
8.1.4 Status (ID 20) 35
8.1.5 Data Type (ID 21) 35
8.1.6 Data Unit (ID 22) 36
8.1.7 Reading Valid (ID 23) 36
8.1.8 Full Scale (ID 24) 36
8.1.9 Safe State (ID 39) 36
8.1.10 Safe Value (ID 40) 36
8.1.11 Auto Zero Enable (ID 41) 36
8.1.12 Overrange (ID 44) 36
8.1.13 Underrange (ID 45) 36
8.2 Sensor Analog Input Function Block (Pirani) 37
8.2.1 Analog Input Block Adjust Command (ID 15) 37
8.2.2 Block Type ID (ID 16) 38
8.2.3 Process Value (ID 19) 38
8.2.4 Status (ID 20) 38
8.2.5 Data Type (ID 21) 38
8.2.6 Data Unit (ID 22) 38
8.2.7 Reading Valid (ID 23) 39
8.2.8 Full Scale (ID 24) 39
8.2.9 Safe State (ID 39) 39
8.2.10 Safe Value (ID 40) 39
8.2.11 Overrange (ID 44) 39
8.2.12 Underrange (ID 45) 39
8.3 Sensor Analog Input Function Block (ATM) 40
8.3.1 Analog Input Block Adjust Command (ID 15) 40
8.3.2 Block Type ID (ID 16) 41
8.3.3 Process Value (ID 19) 41
8.3.4 Status (ID 20) 41
8.3.5 Data Type (ID 21) 41
8.3.6 Data Unit (ID 22) 41
8.3.7 Reading Valid (ID 23) 42
8.3.8 Full Scale (ID 24) 42
8.3.9 Safe State (ID 39) 42
8.3.10 Safe Value (ID 40) 42
8.3.11 Overrange (ID 44) 42
8.3.12 Underrange (ID 45) 42

**9 OneOfN Analog Input Function Block 43**
9.1 Information on the Individual Indices 43
9.1.1 Channel Instance Selector (ID 46) 43
9.1.2 OneOfN PV Selector (ID 47) 43

**10 Discrete Output Function Block (Relay 1 and 2) 44**
10.1 Information on the Individual Indices 44
10.1.1 Block Type ID (ID 16) 44
10.1.2 Setpoint Value (ID 19) 44
10.1.3 Status (ID 20) 44
10.2 Discrete Output Function Block Behavior 45

**11 Trip Point Function Block (Relay 1 and 2) 46**
11.1 Trip Point Function Block behaviors 46
11.1.1 Setpoint function 46
11.1.2 Atmosphere detection 46
11.2 Trip Point Function Block (Relay 1 and Relay 2) 47
11.2.1 Block Type ID (ID 16) 47
11.2.2 High Trip Point (ID 17) 47
11.2.3 High Trip Enable (ID 18) 47
11.2.4 Low Trip Point (ID 19) 47
11.2.5 Low Trip Enable (ID 20) 47
11.2.6 Status (ID 21) 48
11.2.7 Discrete Output Function Block Instance (ID 26) 48
11.2.8 Data Units (ID 28) 48
11.2.9 Data Type (ID 29) 48
11.2.10 High Trip Point Hysteresis (ID 31) 48
11.2.11 Low Trip Point Hysteresis (ID 32) 49
11.2.12 Factor of ATM (ID 201) 49
11.2.13 Setpoint Mode (ID 202) 49

**12 Transducer Blocks 50**
12.1 CDG Transducer Block 50
12.1.1 Block Type ID (ID 101) 50
12.1.2 StatusExtension (ID 102) 50
12.1.3 Sensor Alarm (ID 103) 50
12.1.4 Sensor Warning (ID 104) 50
12.2 Pirani Transducer Block 51
12.2.1 Block Type ID (ID 101) 51
12.2.2 StatusExtension (ID 102) 51
12.2.3 Sensor Alarm (ID 103) 51
12.2.4 Sensor Warning (ID 104) 51
12.2.5 Pirani OFF (ID 201) 51
12.3 ATM Transducer Block 52
12.3.1 Block Type ID (ID 101) 52
12.3.2 StatusExtension (ID 102) 52
12.3.3 Sensor Alarm (ID 103) 52
12.3.4 Sensor Warning (ID 104) 52

**13 OneOfN Vacuum Gauge Transducer Block 53**
13.1 Information on the Individual Indices 53
13.1.1 OneOfN Status Extension (ID 120) 53

**Appendix A: Definitions 54**

**Appendix B: Block Type 57**

**Appendix C: Literature 58**

For cross-references within this document, the symbol (→ 🗎 XY) is used, for cross-references to further documents, listed under literature, the symbol (→ 🕮 [Z]).

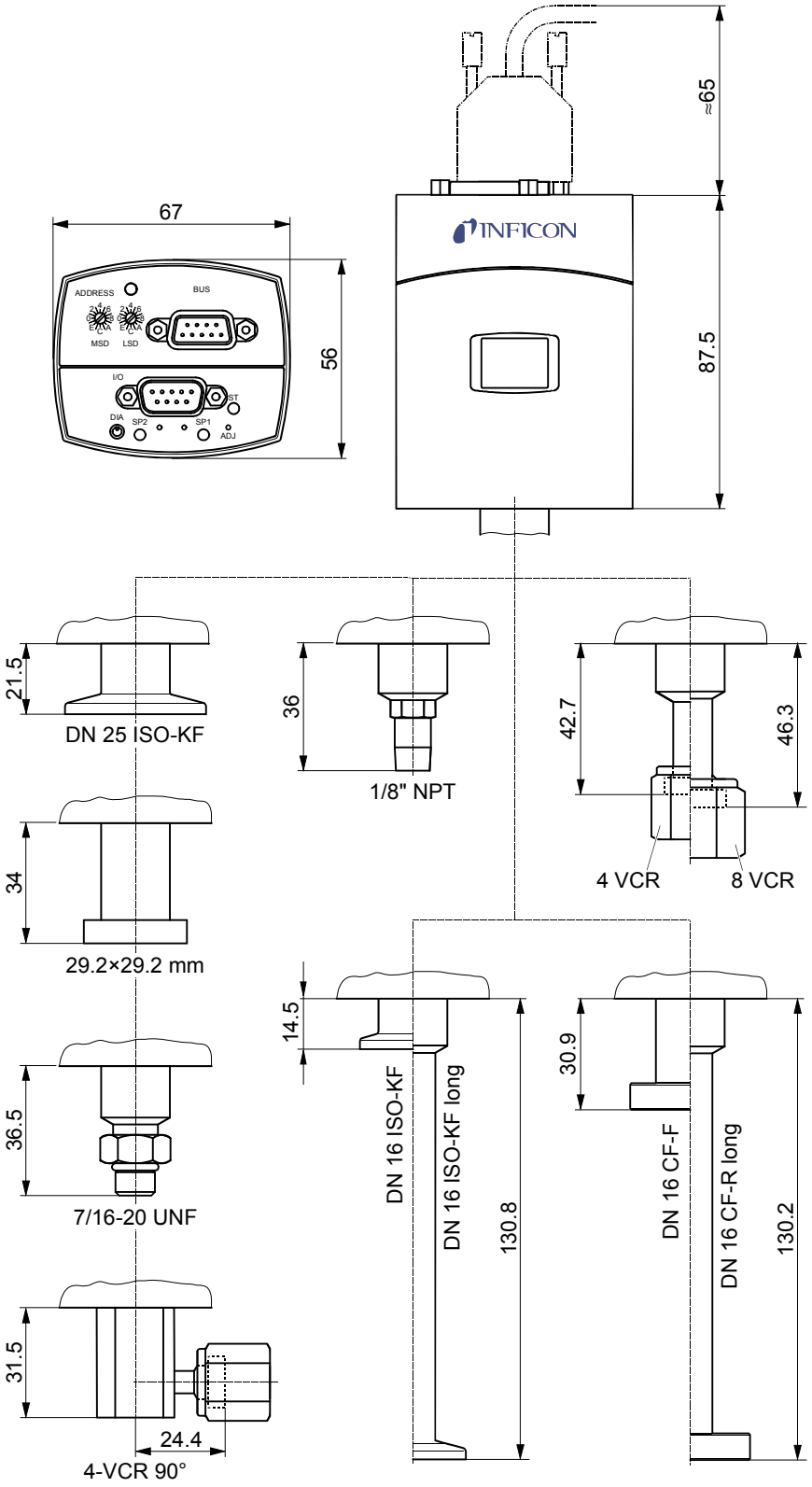
# 1 Technical Data

Further technical data → 🕮 [1], [2]

## Profibus interface

| | |
|---|---|
| Fieldbus name | Profibus |
| Standard applied | → 🕮 [5] |
| Communication protocol data format | → 🕮 [5] |
| Interface, physical | RS485 |
| Data rate | ≤12 MBaud<br>→ 🕮 [5] |
| Node address | |
| Local (Adjustable via hexadecimal "ADDRESS", "MSD", "LSD" switches) | 00 … $7D_{hex}$ (0 … $125_{dec}$) |
| Default setting | $01_{hex}$ |
| Via Profibus (hexadecimal "ADDRESS" switches set to >$7d_{hex}$ (>$125_{dec}$)) | 00 … $7D_{hex}$ (0 … $125_{dec}$) |
| Profibus connection | D-Sub, 9-pin, female |
| Cable | shielded, special Profibus cable<br>→ 🗎 9 and 🕮 [5] |
| Cable length, system wiring | according to Profibus specifications<br>→ 🕮 [5] |

Dimensions [mm]



# 2 Interface Connection

## Making a Profibus Interface Cable

For operating the Pirani Standard (PSG55x) or Pirani Capacitance Diaphragm (PCG55x) Gauge via Profibus, an interface cable conforming to the Profibus standard is required.

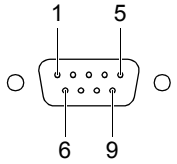
If no such cable is available, make one according to the following indications.

### Cable type

Only a cable that is suited to Profibus operation may be used (→ 🕮 [5]).

### Procedure

❶ Make the Profibus interface cable according to the following indications:




D-Sub, 9-pin
male, soldering side

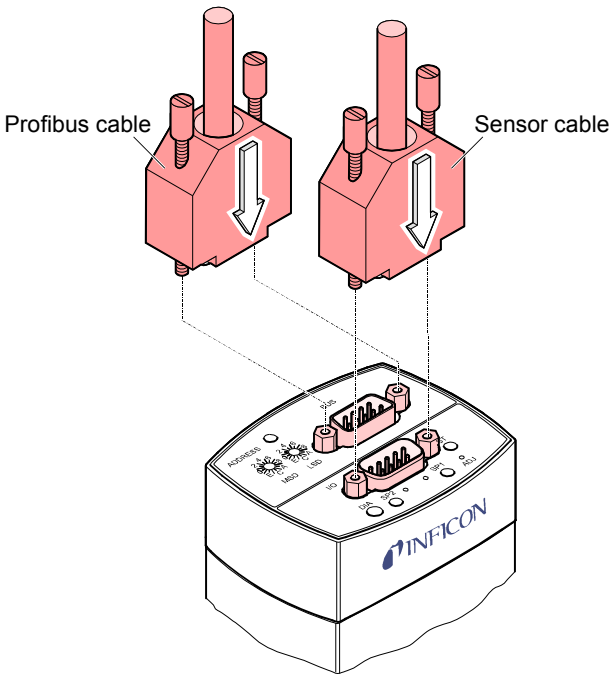
| | | | | |
|---|---|---|---|---|
| Pin 1 | Do not connect | | Pin 7 | Not connected internally |
| Pin 2 | Do not connect | | Pin 8 | RxD/TxD-N |
| Pin 3 | RxD/TxD-P | | Pin 9 | Not connected internally |
| Pin 4 | CNTR-P | [1] | | |
| Pin 5 | DGND | [2] | | |
| Pin 6 | VP | [2] | | |

[1] Only to be connected if an *optical link* module is used.

[2] Only required as line termination for devices at both ends of bus cable (→ 🕮 [5]).

Pin assignment of the D-Sub 15-pins sensor connector according to the respective operating manual (→ 🕮 [1], [2]).

❷ Plug the Profibus (and sensor) cable connector into the gauge.




❸ Lock the Profibus (and sensor) cable connector.

# 3 General Data

## 3.1 Introduction

Via the Profibus interface, the following and further data are exchanged in the standardized Profibus protocol:

- Pressure reading
- Pressure unit (Torr, mbar, Pa)
- Zero / Gain adjustment
- Status and error messages
- Status of the switching functions

**Caution**

Caution: data transmission errors

Any attempt to simultaneously operate the gauge via the RS232C Serial Interface and a Fieldbus interface (DeviceNet or Profibus) or the diagnostic port may result in incorrect data and data transmission errors.

Therefore, it is inadmissible to simultaneously operate the gauge via the RS232C Serial Interface and DeviceNet, Profibus, or the diagnostic port.

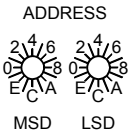
## 3.2 Operating Software

For operating the gauge via Profibus, prior installation of the PxG55x specific GSD file is required on the bus master side. This file can be downloaded from our website (www.inficon.com).

## 3.3 Node Address

The node address is a unique device address on a Profibus network. The gauge can only communicate with the network if its node address has been set properly. The valid address range is 0 … 125 in decimal form.

In the PxG55x, the node address must be set in hexadecimal form (00 … $7D_{hex}$). Two rotary switches at the back of the gauge are used for this:

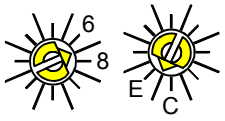



The MSD switch is used to set the high-order address nibble

The LSD switch is used to set the low-order address nibble

The default node address setting is $01_{hex}$.

Example: Node address = $7D_{hex}$:




The node address is polled by the firmware when the gauge is switched on. If the setting deviates from the stored value, the new value is taken over into the nonvolatile memory.

If a value >$7D_{hex}$ (>$125_{dec}$) is entered, the node address setting currently stored in the device remains valid. However, the address can now be set via the Profibus master with the "Set Slave Address" service.

## 3.4 "NET" LED

THE "NET" LED indicates the network status of the PxG55x gauge:

| LED | Status |
|---|---|
| Off | Device is off |
| Red | Error. The alarm bit in Exception Status is set (→ 🗎 26). |
| Green/flashing | Device is ok and uses acyclic data traffic (→ 🗎 14). |
| Green | Device is ok and uses cyclic data traffic (→ 🗎 15). |

## 3.5 Data Rate

The gauge supports all data rates defined in the IEC 61158 Type 3 / IEC 61784 standards (→ 🕮 [5]) up to 12 MBaud. Automatic data rate setting is supported. Alternatively, a fixed data rate can be selected.

## 3.6 Ident Number

The ident number assigned to the gauge by the PNO (→ 🕮 [4]) is:

| Gauge | Ident number (hexadecimal) |
|---|---|
| PxG55x | 0x0C85 |

## 3.7 Configuration Data

Depending on the standard telegrams used (→ section "Cyclic Message Telegrams"), the following configuration data have to be transmitted to the gauge during the configuration phase:

| Standard telegram M ⇑ S | Standard telegram S ⇑ M | M ⇑ S | S ⇑ M | | | | | | | | Configuration data |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ParamChannel | ParamChannel | Exception Status | OneOfN Status Extension | OneOfN PV Selector | Process Value UINT16 | Process Value float | Channel Status | Trip Relay 1-2 | |
| | 1 | | | x | | | x | | | | 0x42,0x82,0x05,0x03 |
| | 2 | | | x | | | | x | | | 0x42,0x84,0x05,0x08 |
| | 3 | | | x | x | x | x | | | | 0x44,0x84,0x05,0x05, 0x05, 0x03 |
| | 4 | | | x | x | x | | x | | | 0x44,0x86,0x05,0x05, 0x05, 0x08 |
| 2 | 7 | x | x | x | | | x | | | | 0xC4,0x87,0x8A,0x0A,0x0A,0x05,0x03 |
| 2 | 8 | x | x | x | | | | x | | | 0xC4,0x87,0x8C,0x0A,0x0A,0x05,0x08 |
| 2 | 9 | x | x | x | x | x | x | | | | 0xC6,0x87,0x8C,0x0A,0x0A,0x05,0x05, 0x05, 0x03 |
| 2 | 10 | x | x | x | x | x | | x | | | 0xC6,0x87,0x8E,0x0A,0x0A,0x05,0x05, 0x05, 0x08 |
| | 201 | | | x | | | x | | x | x | 0x44,0x85,0x05,0x03,0x05,0x0A |
| | 202 | | | x | | | | x | x | x | 0x44,0x87,0x05,0x08,0x05,0x0A |
| 2 | 203 | x | x | x | | | x | | x | x | 0xC6,0x87,0x8D,0x0A,0x0A,0x05,0x03,0x05,0x0A |
| 2 | 204 | x | x | x | | | | x | x | x | 0xC6,0x87,0x8F,0x0A,0x0A,0x05,0x08,0x05,0x0A |
| 206 | 205 | | | | | | | | | | 0xA1,0x91 |

## 3.8 User Parameter Data

Depending on the pressure unit setting (≙ data unit), the following configuration string has to be transmitted to the gauge (parameter data in hexadecimal format):

| Data string (hex) | Pressure unit |
|---|---|
| 00 00 00 10 01 | COUNTS |
| 00 00 00 13 01 | Torr |
| 00 00 00 13 08 | mbar |
| 00 00 00 13 09 | Pascal |
| 00 00 00 00 00 | TTR compatibility mode |

## 3.9 Types of Communication

PxG55x works according to the Profibus DPV1 specification and can be addressed in cyclic or acyclic data traffic (→ 🕮 [5]).

Acyclic data traffic should be used to make device or process specific settings such as definition of the Safe Values, Safe States etc. or for reading or writing of rarely used attributes.

Cyclic data traffic is used for continuous exchange of the required process parameter values, i.e. pressure value and status indications. A number of standard telegrams are available for cyclic data traffic. They can be selected according to requirements (→ section "Cyclic Message Telegrams").

# 4 Data Exchange Mode

## 4.1 Acyclic Data Transmission with Profibus DPV1 Functionality

The reading and writing operations defined in Profibus are based on a slot index address scheme. In PxG55x, all device functions are organized in the following blocks:

- Device block.
  Describes all organizational parameters of the gauge (serial number, manufacturer, software version, …)
- OneOfN Analog Input Function Block.
  Used to determine which function/transducer block parameter set is mapped into the corresponding block address space.
- Sensor Analog Input Function Block.
  Describes the function of the pressure presentation.
- OneOfN Vacuum Gauge Transducer Block
- Transducer Block.
  Describes the physical interface between the gauge and the process.
- Trip Point Function Block.
  Used to model the action of the trip point relays.
- Discrete Output Function Block.
  Used to control the digital outputs (Trip Function Relays).

### Block, slot and index assignment

Each block is assigned to a separate slot as shown in the table below.

PCG55x

| Slot ID | Block, if OneOfN Channel Instance … (→ 🗎 43) | | |
|---|---|---|---|
| | … Selector = 1 | … Selector = 2 | … Selector = 3 |
| 0 | Device Block | | |
| 1 | OneOfN Analog Input Function Block | | |
| | Analog Input Function Block 1 (CDG) | Analog Input Function Block 2 (Pirani) | Analog Input Function Block 3 (ATM) |
| | OneOfN Vacuum Gauge Transducer Block | | |
| | CDG Transducer Block | Pirani Transducer Block | ATM Transducer Block |
| 2 | Trip Point Function Block 1 | | |
| 3 | Trip Point Function Block 2 | | |
| 4 | Discrete Output Function Block 1 | | |
| 5 | Discrete Output Function Block 2 | | |

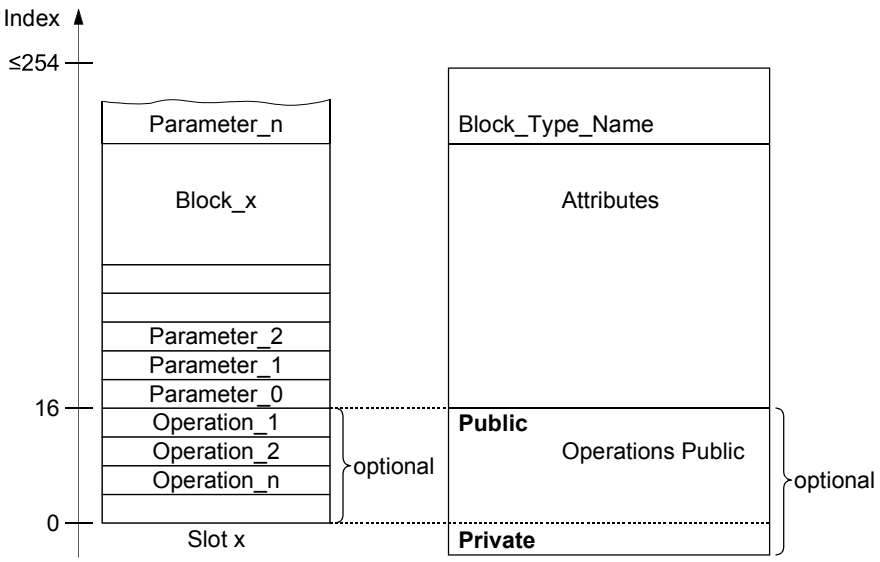
PSG55x

| Slot ID | Block, if OneOfN Channel Instance … (→ 🗎 43) |
|---|---|
| | … Selector = 1 |
| 0 | Device Block |
| 1 | OneOfN Analog Input Function Block |
| | Analog Input Function Block (Pirani) |
| | OneOfN Vacuum Gauge Transducer Block |
| | Pirani Transducer Block |
| 2 | Trip Point Function Block 1 |
| 3 | Trip Point Function Block 2 |
| 4 | Discrete Output Function Block 1 |
| 5 | Discrete Output Function Block 2 |

There are 254 indices per slot. The indices can have a width of 255 bytes.
All values that can be accessed via Profibus have to be mirrored to one of these slots/indices.

The parameters are generally numbered in ascending order, starting with index 16. Services such as "Zero Adjust" are numbered in descending order, starting with index 15.

Assignment of the block elements to the slot indices




## 4.2 Cyclic Data Telegrams in Data Exchange Mode

Within the DataExchange mode the DP-master class 1 cyclically transmits and requests data from all configured slaves in the network. This data transfer aims at the fast propagation of measurement data and command data within the system.

The next two figures contain the input and output data of the Profibus device. The data are divided between a parameter and a process data part.

The parameter data area contains a dedicated channel called parameter channel. It allows the transmission of "single" cyclic requests by specifying an additional protocol within the input and output data area. A concise description can be found on page 16 "Parameter channel".

The process data contains the measurement and command data as described on page 19 "Process data".

Input data

The Input Area (DxIn) as sent in the DataExchange telegram (transmitted by the PxG55x) consists of the 8 byte parameter channel (if there is a parameter channel in the telegram) and up to 8 byte of process data. The actual amount of process data is determined by the board configuration of the basic device.

| Octets 0…7 | Octets 8…15 |
|---|---|
| Parameter channel | Process data |

Output data

The Output Area (DxOut) as received in the DataExchange telegram (transmitted by the master) contains the 8 byte parameter channel and 8 control bytes in the process part of the telegram.

| Octets 0…7 | Octets 8…15 |
|---|---|
| Parameter channel | Process data |

## 4.3 Parameter Channel

The parameter channel was created as a means to transmit acyclic requests within a cyclic data stream. Within this section a brief introduction into the matter is provided. The following table exhibits the principle structure of the parameter channel.

| Octets | | | | | | | | Octets | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | … |
| PKE | | IND | res. | PWE | | | | Standard telegram | | | | | | |
| Parameter channel (PKW) | | | | | | | | | | | | | | |

| | | |
|---|---|---|
| PKE | = | parameter signature value |
| IND | = | index within a slot |
| res. | = | reserved |
| PWE | = | process value |
| Standard telegram | = | cyclic telegram |
| PKW | = | parameter channel (PKE + IND + PWE) |

The PKW allows a read and write access to the parameter space of the slave. Within the PKW mechanism the master formulates an instruction. The slave processes the instruction and formulates the response.

Instructions and responses can't be blocked: Exactly one instruction is transmitted in one Output Telegram and exactly one response is transmitted in one Input Telegram. Therefore exactly 4 bytes of user data may be transmitted within one instruction or one response.

The instructions / responses are coded in the parameter signature word (PKE):

### 4.3.1 PKE (parameter signature value)

| Bit position | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| AK | | | | res. | | | | PNU (Parameter Number, Slot) | | | | | | | |

| Where: | Bits | Meaning |
|---|---|---|
| | 15 … 12 | AK ≙ Instruction / response signature |
| | 11 … 8 | reserved |
| | 7 … 0 | define the slot from which data are read or onto which a value is to be written |

Instruction signature

In Master ⇒ Slave communication, the AK field contains the instruction signature of the master.

In Slave ⇒ Master communication, the AK field contains the instruction signature of the slave.

| AK | Function Master ⇒ Slave (Instruction signature) | AK normal | Function Slave ⇒ Master (Response signature) | AK error |
|---|---|---|---|---|
| 0 | No instruction | 0 | No response | |
| 1 | Read parameter value | 1 | Transmit parameter value (word) | 7 [1)] |
| | | 2 | Transmit parameter value (double word) | |
| | | 11 | Transmit parameter value (byte) | |
| 2 | Write parameter value (data type: word) | 1 | Transmit parameter value (word) | 7 [1)] |
| 3 | Write parameter value (data type: double word) | 2 | Transmit parameter value (double word) | 7 [1)] |
| 10 | Write parameter value (data type: byte) | 11 | Transmit parameter value (byte) | 7 [1)] |

[1)] Instruction cannot be executed (error code)

On the left of the table, the instruction signatures of the master are listed according to their function. On the right of the table, the corresponding normal responses (AK Normal) and error codes (AK Error) transmitted by the slave are listed.

Instruction – response sequence

1) The master transmits an instruction to the slave and repeats that instruction until it receives a response from the slave.
2) The slave keeps transmitting the response to the instruction until the master transmits a new instruction.
3) The master marks the end of the first instruction cycle by setting AK to zero. Only after that, a new instruction/response cycle may be started.

### 4.3.2 PWE (process value)

The PWE represents the data element to be transmitted. If a byte is to be transmitted, that byte has to be in position 8 of the parameter channel. Integers are transmitted with bytes 7 and 8. Double integer and float values are transmitted with bytes 5 … 8.

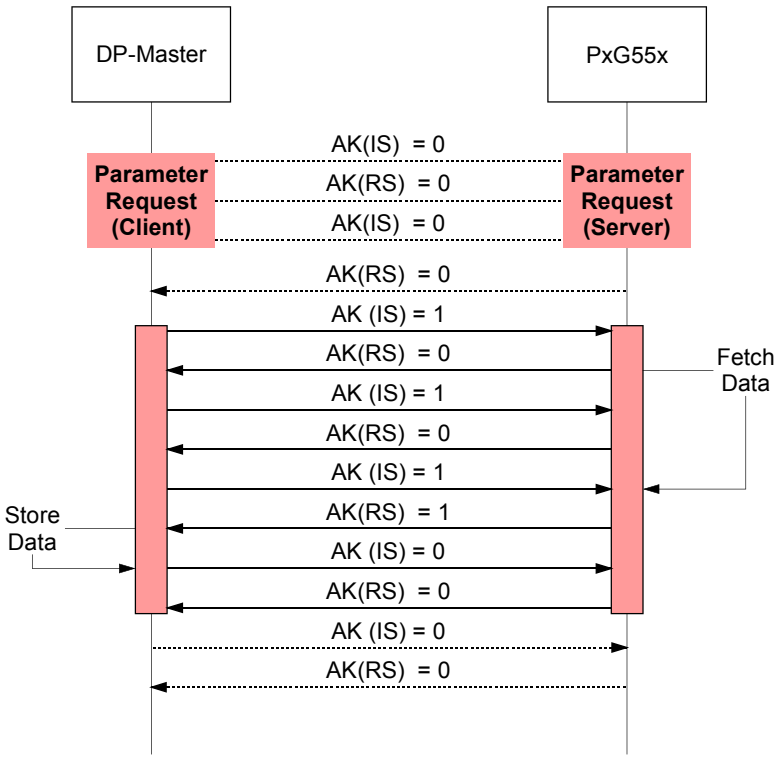
### 4.3.3 Error code

The slave sends an error message on octet 7 and octet 8 (interpreted as INT16) in case a request cannot be fulfilled.

| Error code | Meaning |
|---|---|
| 0 | Undefined slot |
| 1 | Parameter not changeable |
| 2 | Lower or upper value range limit overflow |
| 3 | Subindex error |
| 4 | No array |
| 5 | Data type error |
| 6 | Setting not allowed (only resettable) |
| 7 | Description element not changeable |
| 8 | Reserved |
| 9 | Reserved |
| 10 | Access group error |
| 11 | No operation sovereignty |
| 12 | Password error |
| 13 | Text not readable in cyclic data transfer |
| 14 | Name not readable in cyclic data transfer |
| 15 | No text array existent |
| 16 | Reserved |
| 17 | Instruction not processable due to bad behavior state |
| 18 | Other errors |
| 19 | Data not readable in cyclic error |
| 20…100 | Reserved |
| 201 | Already in requested state |
| 202 | Object state conflict |

### 4.3.4 Sequence Diagram

The following sequence diagram contains an example parameter request by the cyclic DP master in order to illustrate the application of parameter channel.




The master must mark the end of the instruction cycle by setting AK to zero. Only after that, a new instruction/response cycle may be started.

## 4.4 Process Data

The structure of the process data part of the input and output data area depends on the chosen message telegram. Applicable configurations are described below.

When selecting a message telegram, ascertain what output format of the measured value (integer/float) is required and whether a parameter channel is needed or not. The gauge can also be operated in such a way that the master does not transmit any output data to the slave.

### 4.4.1 Standard Telegrams

| Standard telegram | Master ⇔ Slave | Bytes | Meaning |
|---|---|---|---|
| 2 | M ⇒ S | 0…7 | Parameter Channel |
| | | | |
| 1 | S ⇒ M | 0<br>1…2 | Exception Status<br>Process Value UINT16 |
| 2 | S ⇒ M | 0<br>1…4 | Exception Status<br>Process Value float |
| 3 | S ⇒ M | 0<br>1<br>2<br>3…4 | Exception Status<br>OneOfN Status Extension<br>OneOfN PV Selector<br>Process Value UINT16 |
| 4 | S ⇒ M | 0<br>1<br>2<br>3…6 | Exception Status<br>OneOfN Status Extension<br>OneOfN PV Selector<br>Process Value float |
| 7 | S ⇒ M | 0…7<br>8<br>9…10 | Parameter Channel<br>Exception Status<br>Process Value UINT16 |
| 8 | S ⇒ M | 0…7<br>8<br>9…12 | Parameter Channel<br>Exception Status<br>Process Value float |
| 9 | S ⇒ M | 0…7<br>8<br>9<br>10<br>11…12 | Parameter Channel<br>Exception Status<br>OneOfN Status Extension<br>OneOfN PV Selector<br>Process Value UINT16 |
| 10 | S ⇒ M | 0…7<br>8<br>9<br>10<br>11…14 | Parameter Channel<br>Exception Status<br>OneOfN Status Extension<br>OneOfN PV Selector<br>Process Value float |
| 201 | S ⇒ M | 0<br>1…2<br>3<br>4…5 | Exception Status<br>Process Value UINT16<br>OneOfN Status Extension<br>Trip Relay 1-2 |
| 202 | S ⇒ M | 0<br>1…4<br>5<br>6…7 | Exception Status<br>Process Value float<br>OneOfN Status Extension<br>Trip Relay 1-2 |
| 203 | S ⇒ M | 0…7<br>8<br>9…10<br>11<br>12…13 | Parameter Channel<br>Exception Status<br>Process Value UINT16<br>OneOfN Status Extension<br>Trip Relay 1-2 |

(continued)

**Standard Telegrams** (concluded)

| Standard telegram | Master ⇔ Slave | Bytes | Meaning |
|---|---|---|---|
| 204 | S ⇒ M | 0…7 | Parameter Channel |
| | | 8 | Exception Status |
| | | 9…12 | Process Value float |
| | | 13 | OneOfN Status Extension |
| | | 14…15 | Trip Relay 1-2 |
| 205 [1] | S ⇒ M | 1 | Status (Bit 4…7) [2] and Measurement Value [2] |
| | | 2 | Measurement Value [2] |
| 206 [1] | M ⇒ S | 1…2 | Two bytes are sent to the slave but are not processed by the slave |

[1] Available in TTR compatibility mode only.

[2] Bit 7: 0 = no error / 1 = error
Bit 6: 0 = Trip Point 1 not activated / 1 = Trip Point 1 activated
Bit 5: 0
Bit 4: 0

The measurement value is given as a 12-bit value. Byte 1 / Bit 3 represents the MSB, Byte 2 / Bit 0 the LSB

$p\,[mbar] = 10^{Measurement\_Value / 482 - 4.778}$
$p\,[Torr] = 10^{Measurement\_Value / 482 - 4.9029}$
$p\,[Pa] = 10^{Measurement\_Value / 482 - 2.778}$

### 4.4.2 Configuration Data

The devices support the following standard telegrams:

| Standard telegram M ⇒ S | Standard telegram S ⇒ M | Configuration data |
|---|---|---|
| - | 1 | 0x42,0x82,0x05,0x03 |
| - | 2 | 0x42,0x84,0x05,0x08 |
| - | 3 | 0x44,0x84,0x05,0x05, 0x05, 0x03 |
| - | 4 | 0x44,0x86,0x05,0x05, 0x05, 0x08 |
| 2 | 7 | 0xC4,0x87,0x8A,0x0A,0x0A,0x05,0x03 |
| 2 | 8 | 0xC4,0x87,0x8C,0x0A,0x0A,0x05,0x08 |
| 2 | 9 | 0xC6,0x87,0x8C,0x0A,0x0A,0x05,0x05, 0x05, 0x03 |
| 2 | 10 | 0xC6,0x87,0x8E,0x0A,0x0A,0x05,0x05, 0x05, 0x08 |
| - | 201 | 0x44,0x85,0x05,0x03,0x05,0x0A |
| - | 202 | 0x44,0x87,0x05,0x08,0x05,0x0A |
| 2 | 203 | 0xC6,0x87,0x8D,0x0A,0x0A,0x05,0x03,0x05,0x0A |
| 2 | 204 | 0xC6,0x87,0x8F,0x0A,0x0A,0x05,0x08,0x05,0x0A |
| 206 | 205 | 0xA1,0x91 |

# 5 I&M Identification & Maintenance Functions

## 5.1 I&M0

I&M is a concept for manufacturer- and sector-independent standardized identification of field devices. The field device provides specific information in an electronic nameplate, which can be primarily accessed online with initial operation and maintenance.

The reading and writing is made on an I&M parameter block consisting of 64 octets. (→ 🕮 [4])

| Content | Octets | Coding (H) |
|---|---|---|
| **Header** | | |
| Manufacturer specific | 10 | $00_H$ |
| **I&M Block** | | |
| MANUFACTURER_ID | 2 | $0x017A_H$ = INFICON |
| ORDER_ID | 20 | Ordering no. |
| SERIAL_NUMBER | 16 | Serial no. |
| HARDWARE_REVISION | 2 | Hardware revision [1)] |
| SOFTWARE_REVISION | 4 | Software revision [1)] |
| REVISION_COUNTER | 2 | Revision counter |
| PROFILE_ID | 2 | $5A00_H$ |
| PROFILE_SPECIFIC_TYPE | 2 | $0000_H$ |
| IM_VERSION | 2 | $0101_H$ |
| IM_SUPPORTED | 2 | $0000_H$ |

1) of the Profibus board

# 6 Block Model

Data to the PxG55x can be transmitted by means of a number of communication protocols and corresponding masters. Profibus defines a master class 1 as normal control unit of the slave (typically a PLC) and a master class 2 as configuration and service unit. The following communication protocols are defined according to the Profibus DPV1 standard.

MS0 Cyclic data traffic between master class 1 and slave

MS1 Acyclic data traffic between master class 1 and slave

MS2 Acyclic data traffic between master class 2 and slave

In the PxG55x, all functions that are made available by the gauge via Profibus are organized in blocks. Access to the individual parameters of the blocks is possible via acyclic services or, for byte, integer and float values, also in cyclic data traffic via the parameter channel.

## 6.1 Block types

The following block types are defined in the PxG55x gauge:

Device Block

The Device Block contains all data that are required for describing the device and handling its state (status of Device State Machine). See chapter 7.

Function Block

Application specific values such as pressure values that result from or can be calculated from the values of the transducer block are represented in the function blocks.

- Analog Input Function Block (CDG).
  → 34
- Analog Input Function Block (Pirani).
  → 37
- Analog Input Function Block (ATM).
  → 40
- OneOfN Analog Input Function Block.
  → 43
- Discrete Output Function Block (Relay 1 and Relay 2).
  → 44
- Trip Point Function Block (Relay 1 and Relay 2).
  → 46

Transducer Block

The physical, process specific functions or interfaces between the PxG55x and the process are represented in transducer blocks.

The following transducer blocks are implemented:

- Capacitance Diaphragm Gauge (CDG) Transducer Block.
  → 50
- Heat Transfer Vacuum Gauge (Pirani) Transducer Block
  → 51
- Atmosphere Pressure Sensor (ATM) Transducer Block
  → 52
- OneOfN Vacuum Gauge Transducer Block
  → 53

# 7 Device Block

The following table lists the services and parameters integrated in the Device Block. → Appendix A for abbreviations

| ID | Name | Structure | Data type | Bytes | Access | Store | → |
|---|---|---|---|---|---|---|---|
| 15 | Device Block State | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 33 |
| 16 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 24 |
| 17 | Device Type | Simple | VSTRING(n) | 8 | 2_R | N | 24 |
| 18 | Standard Revision Level | Simple | VSTRING(n) | 9 | 2_R | N | 24 |
| 19 | Device Manufacturer Identifier | Simple | VSTRING(n) | 20 | 2_R | N | 24 |
| 20 | Manufacturer Model Number | Simple | VSTRING(n) | 20 | 2_R | N | 24 |
| 21 | Software or Firmware Revision Level (Profibus Adapter) | Simple | VSTRING(n) | 8 | 2_R | N | 24 |
| 22 | Hardware Revision Level (Profibus Adapter) | Simple | VSTRING(n) | 8 | 2_R | N | 24 |
| 23 | Serial Number | Simple | VSTRING(n) | 30 | 2_R | N | 24 |
| 24 | Device Configuration | Simple | VSTRING(n) | 50 | 2_R | N | 25 |
| 25 | Device State | Simple | UINT8 | 1 | 0_XI<br>1_R<br>2_R | V | 25 |
| 26 | Exception Status | Simple | UINT8 | 1 | 0_XI<br>1_R<br>2_R | V | 26 |
| 27 | Exception Detail Alarm | Record | → below | - | 1_R<br>2_R | V | 27 |
| 28 | Exception Detail Warning | Record | → below | - | 1_R<br>2_R | V | 29 |
| 36 | Run hours | Simple | UINT16 | 2 | 1_R<br>2_R | N | 30 |
| 202 | Basic Device Firmware Revision Level | Simple | VSTRING(n) | 8 | 2_R | N | 30 |
| 204 | Common Exception Detail Alarm | Array | UINT8 | 2 | 1_R<br>2_R | V | 30 |
| 205 | Device Exception Detail Alarm | Array | UINT8 | 4 (PCG)<br>2 (PSG) | 1_R<br>2_R | V | 30 |
| 206 | Manufacturer Exception Detail Alarm | Simple | UNIT8 | 1 | 1_R<br>2_R | V | 30 |
| 207 | Common Exception Detail Warning | Array | UINT8 | 2 | 1_R<br>2_R | V | 30 |
| 208 | Device Exception Detail Warning I | Array | UINT8 | 1 | 1_R<br>2_R | V | 30 |
| 209 | Device Exception Detail Warning II | Array | UINT8 | 4 (PCG)<br>2 (PSG) | 1_R<br>2_R | V | 30 |
| 210 | Manufacturer Exception Detail Warning | Simple | UINT8 | 1 | 1_R<br>2_R | V | 31 |
| 211 | Address Switch | Simple | UINT8 | 1 | 1_R<br>2_R | V | 31 |
| 212 | Run hours (*4) | Simple | UINT32 | 4 | 1_R<br>2_R | N | 31 |

## 7.1 Information on the individual Indices

### 7.1.1 Block Type ID (ID 16)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Device Block" is set to "1". The other defined block types are listed in Appendix B.

### 7.1.2 Device Type (ID 17)

The Device Type parameter identifies the device type which is connected to the field bus via Profibus.

The Device Type of the PSG55x gauge is "VPG" which is an acronym for "Vacuum Pressure Gauge".
The Device Type of the PCG55x gauge is "CVG" which is an acronym for "Combination Vacuum Gauge".

### 7.1.3 Standard Revision Level (ID 18)

This parameter describes the version of the "Sensor/Actuator Network Specific Device Model" published by the SEMI® (Semiconductor Equipment and Materials International, California), according to which the profile of this device has been developed.

The fixed setting of this parameter is "E54-0997".

### 7.1.4 Device Manufacturer Identifier (ID 19)

This parameter describes the manufacturer of the device. It is set to "INFICON AG".

### 7.1.5 Manufacturer Model Number (ID 20)

This parameter provides the part number of the gauge. → 🗎 3 "Validity"

### 7.1.6 Software or Firmware Revision Level (ID 21)

This describes the version of the Profibus firmware in the following format: xxyyzz (where 'xx' is the compatibility index, 'yy' is the actual version, 'zz' is the developing version).

### 7.1.7 Hardware Revision Level (ID 22)

This parameter describes the version of the Profibus adapter hardware in the following format: xxyyzz (where 'xx' is the generic version, 'yy' is the actual version, 'zz' is the developing version).

### 7.1.8 Serial Number (ID 23)

This parameter provides the serial number of the device.

### 7.1.9 Device Configuration (ID 24)

This parameter provides the current device configuration, e.g. " DS9 SP WO ATM PB CDG AO0 PIR".
The following abbreviations are used:

| Abbreviation | Description |
|---|---|
| DS9 | D-Sub connector 9-pin |
| FCC | FCC connector |
| HM | Hirschmann connector |
| DS15HD | D-Sub connector 15-pin HD |
| SP | Setpoints |
| PS | Potential separation |
| WO | Tungsten filament |
| NI | Nickel filament |
| ATM | Atmosphere sensor |
| DIS | Display |
| DN | DeviceNet |
| PB | Profibus |
| RS1 | RS485 protocol 1 |
| RS2 | RS485 protocol 2 |
| RS3 | RS485 protocol 3 |
| RS4 | RS485 protocol 4 |
| CDG | Capacitance Diaphragm sensor |
| AO0 | 8.5 V analog out |
| AO1 | 10 V analog out |
| AO3 | Special analog out |
| TTR21X | TTR21X compatibility |
| PIR | Heat Transfer (Pirani) vacuum sensor |
| ION | Hot- / Cold-Ionization vacuum sensor |

### 7.1.10 Device State (ID 25)

This parameter indicates the overall status of the gauge. Due to the structure of the Device State Machine, the following states are possible:

| Parameter value | Status |
|---|---|
| 0 | Undefined |
| 1 | Self testing |
| 2 | Idle |
| 3 | Self test exception |
| 4 | Executing |
| 5 | Abort |
| 6 | Critical fault |
| 7…50 | Reserved by PNO |
| 51…99 | Device-specific |
| 100…255 | Manufacturer-specific |

### 7.1.11 Exception Status (ID 26)

The Exception Status describes the alarm and warning states of the gauge in an "Extended error output format".

A difference is made between warnings and errors.

Alarms and errors are divided into three groups (→ 🗎 27 "Exception Detail Alarm" and 🗎 29 "Exception Detail Warning" for details):

- ALARM / Warning Device Common — For errors that occur independently of the type of device used, e.g. supply error, RAM, ROM, or EEPROM error.
- ALARM / Warning Device Specific — For device specific errors.
- ALARM / Warning Manufacturer Specific — For errors defined by the manufacturer that are not mentioned in the standard.

In each of the above groups, there are several error or warning conditions. The individual fields are presented in the "Exception Detail Alarm" and "Exception Detail Warning". If an error message occurs in "Exception Detail Alarm" or "Exception Detail Warning", the corresponding bit is set in the Exception Status. Therefore, if bits 0…6 of the Exception Status are on "0" there is no warning message pending.

If a bit is set, the actual error can be read in the corresponding group.

The Exception Status is output in cyclic data and informs on the current error status using only one byte. If an error occurs, the current error status can be read via acyclic services or in cyclic data exchange via the parameter channel. This ensures that while the current error status is always available in the cyclic data, no unnecessary data overhead is transmitted.

| Bit | Function | Meaning |
|---|---|---|
| 0 | ALARM, device common | The bit is set if an error of the Alarm Device Common group is detected. |
| 1 | ALARM, device specific | The bit is set if an error of the Alarm Device Specific group is detected. |
| 2 | ALARM, manufacturer specific | The bit is set if an error of the Alarm Manufacturer Specific group is detected. |
| 3 | – | – |
| 4 | WARNING, device common | The bit is set if an error of the Warning Device Common group is detected. |
| 5 | WARNING, device specific | The bit is set if an error of the Warning Device Specific group is detected. |
| 6 | WARNING, manufacturer specific | The bit is set if an error of the Warning Manufacturer Specific group is detected. |
| 7 | Expanded Format | Is constantly on "1" and marks the use of the expanded error output format. |

### 7.1.12 Exception Detail Alarm (ID 27)

If one of the bits 0…2 is set in the Exception Status, the current error can be read in the "Exception Detail Alarm" parameter. Depending on the device configuration specified by instrument-type the "Exception Detail Alarm" parameter consists of a total of 8 or 10 bytes that inform on the error status of the gauge.

PCG55x

The parameter contains an array of 10 bytes, which are assigned as follows:

| Byte no | Name | Description | Value |
|---|---|---|---|
| | **Common Exception Detail Alarm** | | |
| 0 | Size | Number of subsequent bytes used for description of the alarm (simple, UINT8, 1 byte) | 2 |
| 1 | Detail 0 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| 2 | Detail 1 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| | **Device Exception Detail Alarm** | | |
| 3 | Size | Number of subsequent bytes used for description of the alarm (simple, UINT8, 1 byte) | 4 |
| 4 | Detail 0 | Error message (array, UINT8, 2 bytes) | Sensor Alarm (Inst. 1, ID 103) of PCG55x Transducer Block → 🗎 50 |
| 5 | Detail 1 | | |
| 6 | Detail 2 | Error message (array, UINT8, 2 bytes) | Sensor Alarm (Inst. 2, ID 103) of PCG55x Transducer Block → 🗎 50 |
| 7 | Detail 3 | | |
| | **Manufacturer Exception Detail Alarm** | | |
| 8 | Size | Number of subsequent bytes used for description of the alarm (simple, UINT8, 1 byte) | 1 |
| 9 | Detail | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |

PSG55x

The parameter contains an array of 8 bytes, which are assigned as follows:

| Byte no | Name | Description | Value |
|---|---|---|---|
| | **Common Exception Detail Alarm** | | |
| 0 | Size | Number of subsequent bytes used for description of the alarm (simple, UINT8, 1 byte) | 2 |
| 1 | Detail 0 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| 2 | Detail 1 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| | **Device Exception Detail Alarm** | | |
| 3 | Size | Number of subsequent bytes used for description of the alarm | 2 |
| 4 | Detail 0 | Error message (array, UINT8, 2 bytes) | Sensor Alarm (Inst. 1, ID 103) of PSG55x Transducer Block → 🗎 50 |
| 5 | Detail 1 | | |
| | **Manufacturer Exception Detail Alarm** | | |
| 6 | Size | Number of subsequent bytes used for description of the alarm (simple, UINT8, 1 byte) | 1 |
| 7 | Detail | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |

Common Exception Detail Alarm / Warning

| Bit | Common Exception Detail 0 |
|---|---|
| 0 | Internal diagnostic exception |
| 1 | Microprocessor exception |
| 2 | EPROM exception |
| 3 | EEPROM exception |
| 4 | RAM exception |
| 5 | Communications exception |
| 6 | Internal real-time exception |
| 7 | 0 |

| Bit | Common Exception Detail 1 |
|---|---|
| 0 | 0 |
| 1 | 0 |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |

Manufacturer Exception Detail Alarm / Warning

| Bit | Manufacturer Exception Detail |
|---|---|
| 0 | Internal communication exception |
| 1 | Incompatible software |
| 2 | 0 |
| 3 | 0 |
| 4 | 0 |
| 5 | 0 |
| 6 | 0 |
| 7 | 0 |

### 7.1.13 Exception Detail Warning (ID 28)

If one of bits 4…6 is set in the Exception Status, the current warning can be read in the parameter "Exception Detail Warning". Depending on the device configuretion specified by instrument-type the "Exception Detail Warning" parameter consists of a total of 9 or 11 bytes that inform on the warning status of the gauge.

PCG55x

The parameter contains an array of 11 bytes, which are assigned as follows:

| Byte no | Name | Description | Value |
|---|---|---|---|
| | **Common Exception Detail Warning** | | |
| 0 | Size | Number of subsequent bytes used for description the warning (simple, UINT8, 1 byte) | 2 |
| 1 | Detail 0 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| 2 | Detail 1 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| | **Device Exception Detail Warning** | | |
| 3 | Size | Number of subsequent bytes used for description the warning | 5 |
| 4 | OneOfN Status Extension | Status extension (simple, UINT8, 1 byte) | Status extension (ID 102) of PCG55x Transducer Block |
| 5 | Detail 0 | Error message (array, UINT8, 2 bytes) | Sensor Warning (Inst. 1, ID 104) of PCG55x Transducer Block |
| 6 | Detail 1 | | |
| 7 | Detail 2 | Error message (array, UINT8, 2 bytes) | Sensor Warning (Inst. 2, ID 104) of PCG55x Transducer Block |
| 8 | Detail 3 | | |
| | **Manufacturer Exception Detail Warning** | | |
| 9 | Size | Number of subsequent bytes used for description of the warning (simple, UINT8, 1 byte) | 1 |
| 10 | Detail | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |

PSG55x

The parameter contains an array of 9 bytes, which are assigned as follows:

| Byte no | Name | Description | Value |
|---|---|---|---|
| | **Common Exception Detail Warning** | | |
| 0 | Size | Number of subsequent bytes used for description the warning (simple, UINT8, 1 byte) | 2 |
| 1 | Detail 0 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| 2 | Detail 1 | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |
| | **Device Exception Detail Warning** | | |
| 3 | Size | Number of subsequent bytes used for description the warning | 3 |
| 4 | OneOfN Status Extension | Status extension (simple, UINT8, 1 byte) | Status extension (ID 102) of PSG55x Transducer Block → 🗎 51 |
| 5 | Detail 0 | Error message (array, UINT8, 2 bytes) | Sensor Warning (Inst. 1, ID 104) of PSG55x Transducer Block → 🗎 50 |
| 6 | Detail 1 | | |
| | **Manufacturer Exception Detail Warning** | | |
| 7 | Size | Number of subsequent bytes used for description of the alarm (simple, UINT8, 1 byte) | 1 |
| 8 | Detail | Error message (simple, UINT8, 1 byte) | → table 🗎 28 |

### 7.1.14 Run hours (ID 36)

This parameter identifies the number of hours that the basic device has been powered ON. The parameter has a resolution of 1 hour.

### 7.1.15 Basic Device Firmware Revision Level (ID 202)

This proprietary parameter describes the firmware version of the basic device in the following format: xxyyzz (where 'xx' is the compatibility index, 'yy' is the actual version, 'zz' is the developing version).

### 7.1.16 Common Exception Detail Alarm (ID 204)

This proprietary parameter allows access of the corresponding part (Detail 0 and Detail 1) of the Common Exception Detail Alarm parameter (ID 27) (→ 🗎 27).

### 7.1.17 Device Exception Detail Alarm (ID 205)

This proprietary parameter allows access of the corresponding part (Detail) of the Device Exception Detail Alarm parameter (ID 27) (→ 🗎 27).

### 7.1.18 Manufacturer Exception Detail Alarm (ID 206)

This proprietary parameter allows access of the corresponding part (Detail) of the Manufacturer Exception Detail Alarm parameter (ID 27) (→ 🗎 27).

### 7.1.19 Common Exception Detail Warning (ID 207)

This proprietary parameter allows access of the corresponding part (Detail 0 and Detail 1) of the Common Exception Detail Warning parameter (ID 28) (→ 🗎 29).

### 7.1.20 Device Exception Detail Warning (ID 208 / ID 209)

This proprietary parameter allows access of the corresponding part (Status Extension and Detail) of the Device Exception Detail Warning parameter (ID 28) (→ 🗎 29).

### 7.1.21 Manufacturer Exception Detail Warning (ID 210)

This proprietary parameter allows access of the corresponding part (Detail) of the Manufacturer Exception Detail Warning parameter (ID 28) (→ 🗎 29).

### 7.1.22 Address Switch (ID 211)

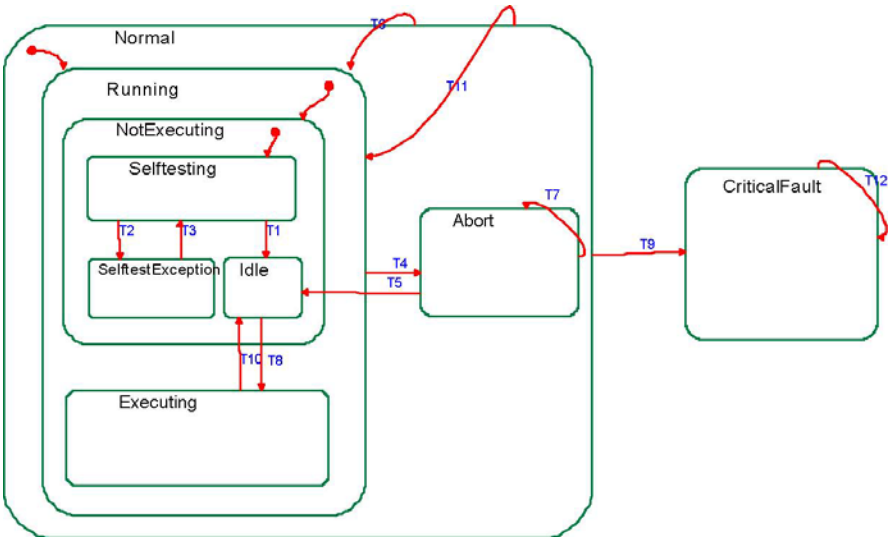
This parameter is for testing purposes only. It holds the value (0...255) of the switch position according to the labeling of the switch.

### 7.1.23 Run hours x4 (ID 212)

This parameter identifies the number of hours that the basic device has been powered ON. The parameter has a resolution of 0.25 hours.

## 7.2 Device Block Status diagram

The PxG55x behaves as described in the status diagram below.




After the start, the gauge independently runs through the INIT and SELFTESTING status and eventually changes to the IDLE status (if there is no error) or to the SELFTEST_EXCEPTION status (if there is a gauge error).

When data traffic is taken up, a difference has to be made between cyclic and acyclic data traffic.

Cyclic data traffic

As soon as cyclic data interchange is taken up, the gauge automatically changes to the EXECUTING status.

Acyclic data traffic

In acyclic data traffic, a START service has to be transmitted to bring the gauge to the EXECUTING status.

### 7.2.1 General Device Block State descriptions

| Status name | Description |
|---|---|
| NORMAL (State set) | The firmware of the device has started. All configured block instances exist. No device internal initialization and check tasks have been carried out yet. |
| RUNNING (State set) | This is the entry sub-state to NORMAL. All block instances are initialized. The parameters have appropriate initial or default values (as defined in this profile). Acyclic access to the device is possible (i.e. MS2 services and MS0 diagnosis service (get_diagnosis)) |
| NOTEXECUTING (State set) | This is the entry sub-state to RUNNING. Device is not executing (e.g., it is not performing its device-specific function). No self tests have been carried out after a new start of the device. |
| SELFTESTING | This is the entry sub-state to NORMAL, RUNNING and NOTEXECUTING. All block instances exist and have been initialized. Device is performing device-specific and device type-specific tests to determine if it is qualified to be running. |
| IDLE | All blocks and device hardware and software have been initialized and have successfully completed self testing. The device is ready for cyclic data transfer. |
| SELFTESTEXCEPTION | Object has detected an exception condition during self testing. The details of the exception are stored in the appropriate parameter values of the Device Block. |
| EXECUTING | Device is executing (e.g., it is performing its device-specific function) its functions according to the purpose of the device. The detailed purposes are described in the certain blocks of the profile. The device performs cyclic data transfer. |
| ABORT | Device Block instance is in an aborted state. The device-specific functions are not performed properly. If the reason of the abort is not active, the device state can switch automatically back to IDLE state. The cyclic data transfer is not active. |
| CRITICALFAULT | The Device Block (and device) is in a fault state from which there is no recovery. The result of the device-specific function is bad. The conditions required for exit from a critical fault are outside the scope of this document. The cyclic data transfer is not active. |

### 7.2.2 Device Block State Command (ID 15)

There are a number of special commands for bringing the gauge into a status it does not automatically go to.

The device block state method allows the master to force the behavior of the device block state machine (as shown in parameter device state) by setting it to one of the values shown in the table below. Depending of the actual state, some values may not be allowed at a time.

| ID value | Name | Description |
|---|---|---|
| 0 | Inactive | No action. |
| 1 | Reset | Used for reinitializing the device. |
| 2 | Abort | Brings the device to the ABORT status. |
| 3 | Recover | Used for bringing the device from the ABORT status into the Recovered State ≙ IDLE. |
| 4 | Execute | Brings the unit to the EXECUTING status, in which the gauge functions normally. As soon as cyclic data traffic is initialized, this status command is executed automatically. |
| 5 | Stop | Brings the gauge to the IDLE status. |
| 6 | Perform Diagnostic | Stops the running activity and starts SELFTEST. |
| 7…127 | Reserved by PNO | |
| 128 | Factory Reset (Manufacturer specific) | Used for reinitializing the device (Profibus interface and basic device) to the factory settings. |
| 129 | Reset (Manufacturer specific) | Used for reinitializing the device (Profibus interface and basic device). |
| 130…255 | Manufacturer specific | |

# 8 Sensor Analog Input Function Blocks

All gauge functions of the PxG55x are described in the Analog Input Function Block. Depending on the device configuration there are up to 3 measuring systems included.

PCG55x

Because the gauge includes three measuring systems, there are three Analog Input Function Block Instances:

- Instance 1 represents the Capacitance Diaphragm Gauge (CDG) measuring part of the gauge (→ 🗎 34).
- Instance 2 represents the Pirani measuring part of the gauge (→ 🗎 37).
- Instance 3 represents the Atmosphere Pressure Sensor (ATM) measuring part of the gauge (→ 🗎 40).

PSG55x

Because the gauge includes only one measuring system, there is only one Analog Input Function Block Instance:

- Instance 1 represents the Pirani measuring part of the gauge (→ 🗎 37).

## 8.1 Sensor Analog Input Function Block (CDG)

This function block describes the functionality of the Capacitance Diaphragm Gauge (CDG) measuring part of the gauge.

The PSG55x does not have a Capacitance Diaphragm Gauge (CDG) and this function block (CDG) is not supported.

The following attributes are supported:

| ID | Name | Structure | Data type | Bytes | Access | Store | → 🗎 |
|---|---|---|---|---|---|---|---|
| 15 | AIBlockAdjust Command | Record | UINT8 | 2 | 1_R/W<br>2_R/W | N | 35 |
| 16 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 35 |
| 19 | ProcessValue (PV) | Simple | *) | – | 0_XI<br>1_R<br>2_R | D | 35 |
| 20 | Status | Simple | UINT8 | 1 | 0_XI<br>1_R<br>2_R | D | 35 |
| 21 | Data Type | Simple | UINT8 | 1 | 2_R/W | N | 35 |
| 22 | Data Unit | Simple | UINT16 | 2 | 2_R/W | N | 36 |
| 23 | Reading Valid | Simple | Boolean | 1 | 1_R<br>2_R | D | 36 |
| 24 | Full Scale | Simple | *) | – | 1_R<br>2_R | N | 36 |
| 39 | Safe State | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 36 |
| 40 | Safe Value | Simple | *) | – | 1_R/W<br>2_R/W | N | 36 |
| 41 | Auto Zero Enable | Simple | Boolean | 1 | 1_R/W<br>2_R/W | N | 36 |
| 44 | Overrange | Simple | *) | – | 1_R<br>2_R | N | 36 |
| 45 | Underrange | Simple | *) | – | 1_R<br>2_R | N | 36 |

*) According to data type value (ID 21)

### 8.1.1 Analog Input Block Adjust Command (ID 15)

| Byte | Name | Structure | Data Type | Bytes | Access | Store |
|---|---|---|---|---|---|---|
| 0 | Adjust Command | Simple | UINT8 | 1 | 1_R/W 2_R/W | N |
| 1 | Target Value | Simple | *) | 1 | 1_R/W 2_R/W | N |

*) According to data type value (ID 21)

| Adjust Command | Name | Description |
|---|---|---|
| 0 | Zero Adjust | Used to calibrate diaphragm zero pressure value. A target value has not to been used. |
| 1…255 | Reserved | Reserved, no action |

| Target value | Description |
|---|---|
| – | Not available |

Zero adjust algorithm

To perform the Zero Adjust pump down to a value 2 decades below the minimum pressure range of the gauge and then start the Zero Adjust Service.

### 8.1.2 Block Type ID (ID 16)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Sensor Analog Input Function Block" is set to "2". The other defined block types are listed in Appendix B.

### 8.1.3 Process Value (ID 19)

The process value contains the measurement value of the Diaphragm Gauge transducer instance (CDG), transformed on the basis of the chosen data unit. If the device is not in state EXECUTING (ID 25, Device Block), the Process value is set to the value specified by the Safe State (ID 39). This could be the Safe Value, Last Value, Full Scale, Zero, device specific, or vendor specific.

Conversion between different Data Units:

p [mbar] = p [Torr] * 1.33322369
p [Torr] = p [mbar] / 1.33322369
p [Pa] = p [Torr] * 133.322369

Conversion between different Data Types:

$p\,[Counts] = 3511 * \log (p\,[mbar] * 2^{20}) + 1$
$p\,[mbar] = 10^{((p\,[Counts] - 1) / 3511)} / 2^{20}$

### 8.1.4 Status (ID 20)

This parameter provides the Alarm and Warning State of this block instance. Because Alarm and Warning Trip Points are not implemented, this value is always "0".

### 8.1.5 Data Type (ID 21)

This parameter determines the data type of ProcessValue and all related parameters. Two data types are supported: Float and Integer16.

This parameter can be changed via master class 2 acyclic data transfer only if the device is not in cyclic data transfer with master class 1. The data type is valid for all block instances of the device. This means that all parameters determining the data type will have the same value. A change of one data type parameter changes all others to the same value.

It is possible to adjust the data type during cyclic data transfer start up (setting of configuration data, CFG-String. In this case the data type value is updated with the value requested by the configuration data.

The data type can be changed from Float to Integer16. After startup it is set to the value stored in the EEPROM. The default setting is "Float".

| Coding | Data type |
|---|---|
| 3 | INT16 |
| 8 | FLOAT (default) |

### 8.1.6 Data Unit (ID 22)

The device supports the four data units described below:

| Coding (hex) | Unit |
|---|---|
| 1001 | COUNTS |
| 1301 | Torr |
| 1308 | mbar |
| 1309 | Pascal |

For safety reasons, it is not possible to change the pressure unit while the gauge is cyclically interchanging data with a DP/V0 master.

The data unit setting can only be modified when the gauge is in the IDLE status.

In cyclic data traffic, the data unit must be set in the User Parameter Data. All settings previously made in acyclic data traffic are overwritten.
→ 12 "User Parameter Data"

If the data unit is set in one instance, that data unit setting applies to all instances. Likewise, the data unit setting made in the User Parameter Data is valid for all instances.

### 8.1.7 Reading Valid (ID 23)

This parameter indicates that the pressure reading is within a valid range. This means that the following conditions are fulfilled:

- The gauge is in the EXECUTING status (Device Block, ID 25)
- The exception status contains no manufacturer warning or alarm
- The transducer block contains no sensor alarm (ID 103)

If this value is set to zero, the pressure reading is not valid. In such a case, either check Exception Status (ID 26, Device Block) to find out whether there is an error or check the Status Extension (ID 102, Transducer Block) to find out whether the measured value is out of the specified measuring range (overrange or underrange mode).

### 8.1.8 Full Scale (ID 24)

This parameter contains the valid maximum measurement scale of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

### 8.1.9 Safe State (ID 39)

When the gauge is not in the EXECUTING status (ID 25, Device Block) or if there is a device error, a value defined by Safe State is output as pressure value. You can select among the following Safe State values:

| Option | Coding | ProcessValue (PV) behavior |
|---|---|---|
| Zero | 0 | The PV is set to 0. |
| Full Scale | 1 | The PV is set to the full scale value (ID 24). |
| Hold Last Value | 2 | The PV is set to the last valid value obtained in the EXECUTING status. |
| Use Safe Value | 3 | The PV is set to the Safe Value (ID 40). |

### 8.1.10 Safe Value (ID 40)

The Safe Value is the value output with the Process Value Parameter (ID 19) when an error occurs or the gauge goes to the NOT EXECUTING status. If this value is set to zero, it will remain on zero when the data unit is changed.

### 8.1.11 Auto Zero Enable (ID 41)

This parameter defines, if the CDG Zero pressure is automatically and periodically calibrated.

### 8.1.12 Overrange (ID 44)

This parameter contains the highest valid Process Value (PV) of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

### 8.1.13 Underrange (ID 45)

This parameter contains the lowest valid Process Value (PV) of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

## 8.2 Sensor Analog Input Function Block (Pirani)

This function block describes the functionality of the Pirani measuring part of the gauge.

The PSG55x does not have a Capacitance Diaphragm Gauge (CDG) and therefore this function block will be implemented as Instance 1.

The following attributes are supported:

| ID | Name | Structure | Data type | Bytes | Access | Store | → |
|---|---|---|---|---|---|---|---|
| 15 | AIBlockAdjust Command | Record | UINT8 | 2 | 1_R/W<br>2_R/W | N | 37 |
| 16 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 38 |
| 19 | ProcessValue (PV) | Simple | *) | - | 0_XI<br>1_R<br>2_R | D | 38 |
| 20 | Status | Simple | UINT8 | 1 | 0_XI<br>1_R<br>2_R | D | 38 |
| 21 | Data Type | Simple | UINT8 | 1 | 2_R/W | N | 38 |
| 22 | Data Unit | Simple | UINT16 | 2 | 2_R/W | N | 38 |
| 23 | Reading Valid | Simple | Boolean | 1 | 1_R<br>2_R | D | 39 |
| 24 | Full Scale | Simple | *) | - | 1_R<br>2_R | N | 39 |
| 39 | Safe State | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 39 |
| 40 | Safe Value | Simple | *) | - | 1_R/W<br>2_R/W | N | 39 |
| 44 | Overrange | Simple | *) | - | 1_R<br>2_R | N | 39 |
| 45 | Underrange | Simple | *) | - | 1_R<br>2_R | N | 39 |

*) According to data type value (ID 21)

### 8.2.1 Analog Input Block Adjust Command (ID 15)

| Byte | Name | Structure | Data Type | Bytes | Access | Store |
|---|---|---|---|---|---|---|
| 0 | Adjust Command | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N |
| 1 | Target Value | Simple | *) | 1 | 1_R/W<br>2_R/W | N |

*) According to data type value (ID 21)

| Adjust Command | Name | Description |
|---|---|---|
| 0 | Zero Adjust | Used to calibrate pirani zero pressure value. A target value has not to been used. |
| 1 | Gain Adjust | Used to calibrate pirani fullscale pressure value. A target value has not to been used. |
| 2…255 | Reserved | Reserved, no action |

| Target value | Description |
|---|---|
| – | Not available |

Zero adjust algorithm

To perform the Zero Adjust pump down to a value 2 decades below the minimum pressure range of the gauge and then start the Zero Adjust Service.

Gain adjust algorithm

To perform the Gain Adjust vent to atmosphere and then start the Gain Adjust Service.

### 8.2.2 Block Type ID (ID 16)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Sensor Analog Input Function Block" is set to "2". The other defined block types are listed in Appendix B.

### 8.2.3 Process Value (ID 19)

The process value contains the measurement value of the Heat Transfer Vacuum Gauge (Pirani) transducer instance, transformed on the basis of the chosen data unit. If the device is not in state EXECUTING (ID 25, Device Block), the Process value is set to the value specified by the Safe State (ID 39). This could be the Safe Value, Last Value, Full Scale, Zero, device specific, or vendor specific.

Conversion between different Data Units:

p [mbar] = p [Torr] * 1.33322369
p [Torr] = p [mbar] / 1.33322369
p [Pa] = p [Torr] * 133.322369

Conversion between different Data Types:

$p\,[\text{Counts}] = 3511 * \log (p\,[\text{mbar}] * 2^{20}) + 1$
$p\,[\text{mbar}] = 10^{((p\,[\text{Counts}] - 1) / 3511)} / 2^{20}$

### 8.2.4 Status (ID 20)

This parameter provides the Alarm and Warning State of this block instance. Because Alarm and Warning Trip Points are not implemented, this value is always "0".

### 8.2.5 Data Type (ID 21)

This parameter determines the data type of ProcessValue and all related parameters. Two data types are supported: Float and Integer16.

This parameter can be changed via master class 2 acyclic data transfer only if the device is not in cyclic data transfer with master class 1. The data type is valid for all block instances of the device. This means that all parameters determining the data type will have the same value. A change of one data type parameter changes all others to the same value.

It is possible to adjust the data type during cyclic data transfer start up (setting of configuration data, CFG-String. In this case the data type value is updated with the value requested by the configuration data.

The data type can be changed from Float to Integer16. After startup it is set to the value stored in the EEPROM. The default setting is "Float".

| Coding | Data type |
|---|---|
| 3 | INT16 |
| 8 | FLOAT (default) |

### 8.2.6 Data Unit (ID 22)

The device supports the four data units described below:

| Coding (hex) | Unit |
|---|---|
| 1001 | COUNTS |
| 1301 | Torr |
| 1308 | mbar |
| 1309 | Pascal |

For safety reasons, it is not possible to change the pressure unit while the gauge is cyclically interchanging data with a DP/V0 master.

The data unit setting can only be modified when the gauge is in the IDLE status.

In cyclic data traffic, the data unit must be set in the User Parameter Data. All settings previously made in acyclic data traffic are overwritten.
→ 12 "User Parameter Data"

If the data unit is set in one instance, that data unit setting applies to all instances. Likewise, the data unit setting made in the User Parameter Data is valid for all instances.

### 8.2.7 Reading Valid (ID 23)

This parameter indicates that the pressure reading is within a valid range. This means that the following conditions are fulfilled:

- The gauge is in the EXECUTING status (Device Block, ID 25)
- The exception status contains no manufacturer warning or alarm
- The transducer block contains no sensor alarm (ID 103)

If this value is set to zero, the pressure reading is not valid. In such a case, either check Exception Status (ID 26, Device Block) to find out whether there is an error or check the Status Extension (ID 102, Transducer Block) to find out whether the measured value is out of the specified measuring range (overrange or underrange mode).

### 8.2.8 Full Scale (ID 24)

This parameter contains the valid maximum measurement scale of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

### 8.2.9 Safe State (ID 39)

When the gauge is not in the EXECUTING status (ID 25, Device Block) or if there is a device error, a value defined by Safe State is output as pressure value. You can select among the following Safe State values:

| Option | Coding | ProcessValue (PV) behavior |
|---|---|---|
| Zero | 0 | The PV is set to 0. |
| Full Scale | 1 | The PV is set to the full scale value (ID 24). |
| Hold Last Value | 2 | The PV is set to the last valid value obtained in the EXECUTING status. |
| Use Safe Value | 3 | The PV is set to the Safe Value (ID 40). |

### 8.2.10 Safe Value (ID 40)

The Safe Value is the value output with the Process Value Parameter (ID 19) when an error occurs or the gauge goes to the NOT EXECUTING status. If this value is set to zero, it will remain on zero when the data unit is changed.

### 8.2.11 Overrange (ID 44)

This parameter contains the highest valid Process Value (PV) of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

### 8.2.12 Underrange (ID 45)

This parameter contains the lowest valid Process Value (PV) of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

## 8.3 Sensor Analog Input Function Block (ATM)

This function block describes the functionality of the Atmosphere Pressure Sensor measuring part of the gauge.

The PSG55x and some variants of the PCG55x do not have a Atmosphere Pressure Sensor (ATM) and this function block (ATM) is not supported.

The following attributes are supported:

| ID | Name | Structure | Data type | Bytes | Access | Store | → |
|---|---|---|---|---|---|---|---|
| 15 | AIBlockAdjust Command | Record | UINT8 | 2 | 1_R/W<br>2_R/W | N | 40 |
| 16 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 41 |
| 19 | ProcessValue (PV) | Simple | *) | - | 0_XI<br>1_R<br>2_R | D | 41 |
| 20 | Status | Simple | UINT8 | 1 | 0_XI<br>1_R<br>2_R | D | 41 |
| 21 | Data Type | Simple | | 1 | 2_R/W | N | 41 |
| 22 | Data Unit | Simple | UINT16 | 2 | 2_R/W | N | 41 |
| 23 | Reading Valid | Simple | Boolean | 1 | 1_R<br>2_R | D | 42 |
| 24 | Full Scale | Simple | *) | - | 1_R<br>2_R | N | 42 |
| 39 | Safe State | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 42 |
| 40 | Safe Value | Simple | *) | - | 1_R/W<br>2_R/W | N | 42 |
| 44 | Overrange | Simple | *) | - | 1_R<br>2_R | N | 42 |
| 45 | Underrange | Simple | *) | - | 1_R<br>2_R | N | 42 |

*) According to data type value (ID 21)

### 8.3.1 Analog Input Block Adjust Command (ID 15)

| Byte | Name | Structure | Data Type | Bytes | Access | Store |
|---|---|---|---|---|---|---|
| 0 | Adjust Command | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N |
| 1 | Target Value | Simple | *) | 1 | 1_R/W<br>2_R/W | N |

*) According to data type value (ID 21)

| Adjust Command | Name | Description |
|---|---|---|
| 0<br>2…255 | Reserved | Reserved, no action |
| 1 | Gain Adjust | If this service is performed, the ATM sensor value will be set to the same value as at that time measured by the diaphragm gauge. |

| Target value | Description |
|---|---|
| 0…255 | Not available |

Gain adjust algorithm

To perform the Gain Adjust vent to atmosphere and then start the Gain Adjust Service.

### 8.3.2 Block Type ID (ID 16)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Sensor Analog Input Function Block" is set to "2". The other defined block types are listed in Appendix B.

### 8.3.3 Process Value (ID 19)

The process value contains the measurement value of the Atmosphere Pressure Sensor (ATM), transformed on the basis of the chosen data unit. If the device is not in state EXECUTING (ID 25, Device Block), the Process value is set to the value specified by the Safe State (ID 39). This could be the Safe Value, Last Value, Full Scale, Zero, device specific, or vendor specific.

Conversion between different Data Units:

p [mbar] = p [Torr] * 1.33322369
p [Torr] = p [mbar] / 1.33322369
p [Pa] = p [Torr] * 133.322369

Conversion between different Data Types:

p [Counts] = p [mbar] * $2^4$
p [mbar] = p [Counts] / $2^4$

### 8.3.4 Status (ID 20)

The atmosphere pressure sensor cannot trigger alarms and warnings because the atmosphere pressure changes are to small. Therefore the ATM sensor supports no alarm/warning trip points and consequently this parameter is always set to "0".

### 8.3.5 Data Type (ID 21)

This parameter determines the data type of ProcessValue and all related parameters. Two data types are supported: Float and Integer16.

This parameter can be changed via master class 2 acyclic data transfer only if the device is not in cyclic data transfer with master class 1. The data type is valid for all block instances of the device. This means that all parameters determining the data type will have the same value. A change of one data type parameter changes all others to the same value.

It is possible to adjust the data type during cyclic data transfer start up (setting of configuration data, CFG-String. In this case the data type value is updated with the value requested by the configuration data.

The data type can be changed from Float to Integer16. After startup it is set to the value stored in the EEPROM. The default setting is "Float".

| Coding | Data type |
|---|---|
| 3 | INT16 |
| 8 | FLOAT (default) |

### 8.3.6 Data Unit (ID 22)

The device supports the four data units described below:

| Coding (hex) | Unit |
|---|---|
| 1001 | COUNTS |
| 1301 | Torr |
| 1308 | mbar |
| 1309 | Pascal |

For safety reasons, it is not possible to change the pressure unit while the gauge is cyclically interchanging data with a DP/V0 master.

The data unit setting can only be modified when the gauge is in the IDLE status.

In cyclic data traffic, the data unit must be set in the User Parameter Data. All settings previously made in acyclic data traffic are overwritten.
→ 12 "User Parameter Data"

If the data unit is set in one instance, that data unit setting applies to all instances. Likewise, the data unit setting made in the User Parameter Data is valid for all instances.

### 8.3.7 Reading Valid (ID 23)

This parameter indicates that the pressure reading is within a valid range. This means that the following conditions are fulfilled:

- The gauge is in the EXECUTING status (Device Block, ID 25)
- The exception status contains no manufacturer warning or alarm
- The transducer block contains no sensor alarm (ID 103)

If this value is set to zero, the pressure reading is not valid. In such a case, check the Status Extension (ID 102, Transducer Block) to find out whether the measured value is out of the specified measuring range (overrange or underrange mode).

### 8.3.8 Full Scale (ID 24)

This parameter contains the valid maximum measurement scale of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

### 8.3.9 Safe State (ID 39)

When the gauge is not in the EXECUTING status (ID 25, Device Block) or if there is a device error, a value defined by Safe State is output as pressure value. You can select among the following Safe State values:

| Option | Coding | ProcessValue (PV) behavior |
|---|---|---|
| Zero | 0 | The PV is set to 0. |
| Full Scale | 1 | The PV is set to the full scale value (ID 24). |
| Hold Last Value | 2 | The PV is set to the last valid value obtained in the EXECUTING status. |
| Use Safe Value | 3 | The PV is set to the Safe Value (ID 40). |

### 8.3.10 Safe Value (ID 40)

The Safe Value is the value output with the Process Value Parameter (ID 19) when an error occurs or the gauge goes to the NOT EXECUTING status. If this value is set to zero, it will remain on zero when the data unit is changed.

### 8.3.11 Overrange (ID 44)

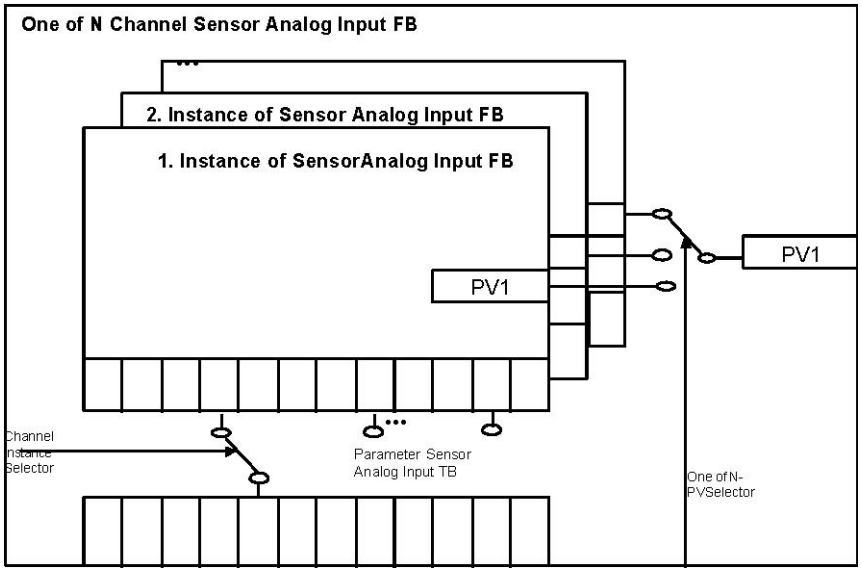
This parameter contains the highest valid Process Value (PV) of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

### 8.3.12 Underrange (ID 45)

This parameter contains the lowest valid Process Value (PV) of the device in terms of the currently selected data type (ID 21) and data unit (ID 22).

# 9 OneOfN Analog Input Function Block

The figure below shows the basic design of the analog input function block.




The following attributes are supported:

| ID | Name | Structure | Data type | Bytes | Access | Store | → 🗎 |
|---|---|---|---|---|---|---|---|
| 46 | Channel Instance Selector | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 43 |
| 47 | OneOfN PV Selector | Simple | UINT8 | 1 | 1_R<br>2_R | N | 43 |

## 9.1 Information on the Individual Indices

### 9.1.1 Channel Instance Selector (ID 46)

Provides access to the selected parameter instances via the function block parameter interface. The channel instance number starts with 1. The selector determines which function/transducer block parameter set is mapped into the "OneOfN Analog Input Function Block" or "OneOfN Vacuum Gauge Transducer Block" address space.

### 9.1.2 OneOfN PV Selector (ID 47)

Selects the PV of the function block instance which will be provided to the master class 1 using the MS0 cyclic data exchange mechanism. The channel instance number starts with 1. The selector defines the Analog Sensor Function Block Instance which Process Value is used as output using cyclic data exchange → 🗎 15).

# 10 Discrete Output Function Block (Relay 1 and 2)

The Discrete Output Function Block is used for switching the relays on and off. The following attributes are supported:

| ID | Name | Structure | Data Type | Bytes | Access | Store | → 🗎 |
|---|---|---|---|---|---|---|---|
| 16 | Block Type ID | Simple | OSTRING(n) | 1 | 1_R<br>2_R | N | 44 |
| 19 | Setpoint Value | Simple | UINT8 | 1 | 0_XO<br>2_R | D | 44 |
| 20 | Status | Simple | Boolean | 1 | 2_R | D | 44 |

## 10.1 Information on the Individual Indices

### 10.1.1 Block Type ID (ID 16)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Discrete Output Function Block" is set to "7". The other defined block types are listed in Appendix B.

### 10.1.2 Setpoint Value (ID 19)

This parameter contains the value which is forwarded to the relay:

| Setpoint value | Relay |
|---|---|
| 0 | Off |
| 1 | On |
| 2…255 | Off |

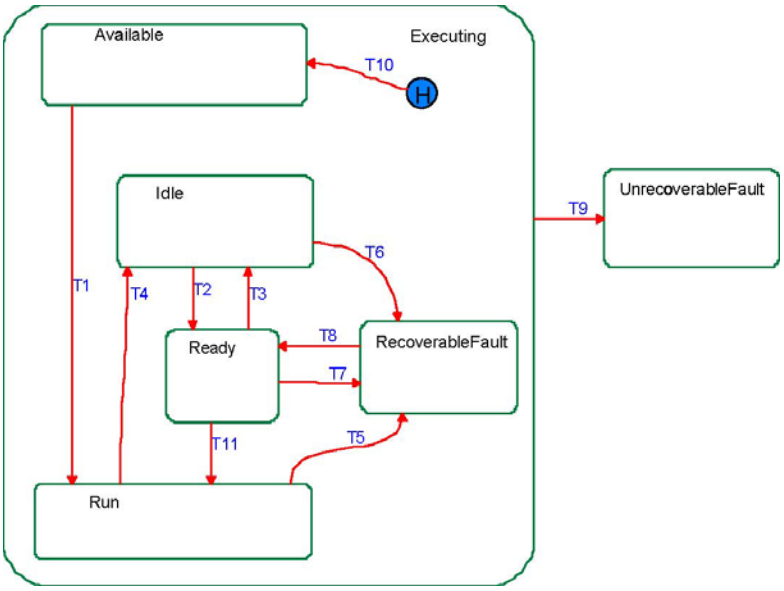
### 10.1.3 Status (ID 20)

This parameter indicates the state of the Discrete Output Function Block Instance.

| Status | Description |
|---|---|
| 0 | Ok |
| 1 | Error or alarm |

## 10.2 Discrete Output Function Block Behavior

The figure below shows the state chart of the discrete output function block.




| State name | Description |
|---|---|
| Available | This is the initial state after power on. The Device Block is in Executing state (→ 🗎 32). |
| Idle | This state cuts the signal flow between the SetpointValue of the certain Discrete Output function block instance and its dedicated underlying actuation hardware. The behavior is according the IdleActionSelector parameter. |
| Ready | The connection between the SetpointValue parameter of the certain Discrete Output function block instance and its dedicated underlying actuation hardware is recovered. The instance is waiting for new SetpointValue parameter values. |
| Run | The instance is working in its dedicated way. |
| Recoverable Fault | There is a temporary fault. The reason is device and manufacturer specific. |
| Unrecoverable Fault | There is a fault with no automatic return. This fault can force a Critical Fault according to page 32 or refers to the certain instance only. |
| Executing | This state is defined on page 32, i.e. the state machine is working in this device state only. |

# 11 Trip Point Function Block (Relay 1 and 2)

The Trip Point Object models the action of trip points for a device, often corres-pondding to physical outputs (Discrete Output Object). Each Trip Point channel has a pointer (Analog Sensor Input Object Instance ID) to a source Analog Sensor Input Instance.

A trip point value, designated as a High or Low Trip Point, is compared to the specified instance of an Analog Input Sensor Process Value parameter. This trip point is intended to be used as a process control indicator only, as distinguished from the Analog Input Sensor Function Block Object's Alarm and Warning trip points.

## 11.1 Trip Point Function Block behaviors

Two different functionalities are implemented: Setpoint function and Atmosphere (ATM) detection. The functionality of the trip points is set with the parameter "Setpoint Mode" (ID 202) (→ 🗎 49).

### 11.1.1 Setpoint function

If "Low Trip Enable" (ID 20) is set, "Low Trip Point" is compared to the input value to generate a trip point condition.

Status will be set if the input (ProcessValue) is at or below the "Low Trip Point". If the pressure (input) increases above "Low Trip Point + Low Trip Point Hysteresis", the Status will be reset.

This behavior is similar at the "High Trip Point".

If "High Trip Enable" (ID 18) is set, "High Trip Point" is compared to the input value to generate a trip point condition.

Status will be set if the input (ProcessValue) is at or above the "High Trip Point". If the pressure (input) decreases below "High Trip Point - High Trip Point Hysteresis", the Status will be reset.

### 11.1.2 Atmosphere detection

This functionality is used to compare the pressure measured by the CDG with the atmospheric pressure.

The PSG55x and some variants of the PCG55x do not have an Atmosphere Pressure Sensor (ATM) and this function (Atmosphere detection) is not supported.

The parameter "Percentage of ATM" (ID 201) is used to define a value "Percentage of Atmosphere". This value will be multiplied with the pressure from the Analog Sensor Instance x (ATM Sensor). Then this result will be copied into parameter "High Trip Point" or "Low Trip Point", depending on the value of parameter 202 (Setpoint Mode).

Status has the same behavior as in the description of the "Setpoint function".

## 11.2 Trip Point Function Block (Relay 1 and Relay 2)

The following attributes are supported:

| ID | Name | Structure | Data type | Bytes | Access | Store | → |
|---|---|---|---|---|---|---|---|
| 16 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 47 |
| 17 | High Trip Point | Simple | According Data Type value (ID 29) | - | 1_R/W<br>2_R/W | N | 47 |
| 18 | High Trip Enable | Simple | Boolean | 1 | 1_R/W<br>2_R/W | N | 47 |
| 19 | Low Trip Point | Simple | According Data Type value (ID 29) | - | 1_R/W<br>2_R/W | N | 47 |
| 20 | Low Trip Enable | Simple | Boolean | 1 | 1_R/W<br>2_R/W | N | 47 |
| 21 | Status | Simple | UINT8 | 1 | 0_XI<br>1_R<br>2_R | D | 48 |
| 26 | Discrete Output Function Block Instance | Simple | UINT8 | 1 | 1_R<br>2_R | N | 48 |
| 28 | Data Units | Simple | UINT16 | 2 | 2_R/W | N | 48 |
| 29 | Data Type | Simple | UINT8 | 1 | 2_R/W | N | 48 |
| 31 | High Trip Point Hysteresis | Simple | According Data Type value (ID 29) | - | 1_R/W<br>2_R/W | N | 48 |
| 32 | Low Trip Point Hysteresis | Simple | According Data Type value (ID 29) | - | 1_R/W<br>2_R/W | N | 49 |
| 201 | Factor of ATM | Simple | FLOAT | 4 | 1_R/W<br>2_R/W | N | 49 |
| 202 | Setpoint Mode | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 49 |

### 11.2.1 Block Type ID (ID 16)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Trip Point Function Block" is set to "17". The other defined block types are listed in Appendix B.

### 11.2.2 High Trip Point (ID 17)

This parameter defines the value at or above which a trip point condition will occur.

### 11.2.3 High Trip Enable (ID 18)

This parameter enables/disables the High Trip Point setting.

| Parameter value | Description |
|---|---|
| 0 | disable High Trip Point setting |
| 1 | enable High Trip Point setting |

### 11.2.4 Low Trip Point (ID 19)

This parameter defines the value at or below which a trip point condition will occur.

### 11.2.5 Low Trip Enable (ID 20)

This parameter enables/disables the Low Trip Point setting.

| Parameter value | Description |
|---|---|
| 0 | disable Low Trip Point setting |
| 1 | enable Low Trip Point setting |

### 11.2.6 Status (ID 21)

This parameter contains the state of this object instance. The functionality is described on page 46.

| Parameter value | Description |
|---|---|
| 0 | Trip Point condition doesn't exist (unasserted) |
| 1 | Trip Point condition exists (asserted) |

### 11.2.7 Discrete Output Function Block Instance (ID 26)

This parameter specifies the associated Discrete Output Object Instance.

| ID | Meaning in the base unit |
|---|---|
| Instance 1 in Slot 2 | Relay 1<br>Trip Point Object Instance 1 in the base unit |
| Instance 1 in Slot 3 | Relay 2<br>Trip Point Object Instance 1 in the base unit |

The instance is always one, because each Discrete Output Function Block (Slot 2 and Slot 3) has only one instance.

### 11.2.8 Data Units (ID 28)

This parameter defines the units of Trip Point, Hysteresis, etc. according to the following table:

| Parameter value (hex) | Unit |
|---|---|
| 1001 | COUNTS |
| 1301 | Torr |
| 1308 | mbar |
| 1309 | Pascal |

For safety reasons, it is not possible to change the pressure unit while the gauge is cyclically interchanging data with a DP/V0 master.

The data unit setting can only be modified when the gauge is in the IDLE status.

In cyclic data traffic, the data unit must be set in the User Parameter Data. All settings previously made in acyclic data traffic are overwritten.
→ 12 "User Parameter Data"

If the data unit is set in one instance, that data unit setting applies to all instances. Likewise, the data unit setting made in the User Parameter Data is valid for all instances.

### 11.2.9 Data Type (ID 29)

This parameter defines the data type of Trip Point, Hysteresis, etc. and all related parameters.

The parameter can be changed via master class 2 acyclic data transfer only if the device is not in cyclic data transfer with master class 1. The data type is valid for all block instances of the device i.e. all parameter determining the data type shall have the same value. A change of one data type parameter changes all other parameters to the same value.

| Parameter value | Type |
|---|---|
| 3 | INT16 |
| 8 | FLOAT (default) |

### 11.2.10 High Trip Point Hysteresis (ID 31)

The High Trip Point Hysteresis value specifies the amount by which the Analog Input Sensor ProcessValue must recover in order to clear a trip point condition.

For example: A High Trip Point of 100 and a High Trip Point Hysteresis value of 2 will result in a trip point condition being set when the (Analog Input Sensor) ProcessValue goes at or above 100 and cleared when ProcessValue drops below 98.

### 11.2.11 Low Trip Point Hysteresis (ID 32)

The Low Trip Point Hysteresis value specifies the amount by which the Analog Input Sensor ProcessValue must recover in order to clear a trip point condition.

For example: A Low Trip Point of 100 and a Low Trip Point Hysteresis value of 2 will result in a trip point condition being set when the (Analog Input Sensor) ProcessValue falls at or below 100 and cleared when ProcessValue increases above 102.

### 11.2.12 Factor of ATM (ID 201)

This value is only used for the calculation of Trip Point value in "ATM Mode".

The basic device multiplies this value with the pressure from the Atmosphere Pressure sensor (Analog Sensor Instance x (ATM Sensor)) and copies the result into the variable "High Trip Point" or "Low Trip Point", depending on the parameter "Setpoint Mode" (ID 202).

### 11.2.13 Setpoint Mode (ID 202)

The following different modes are available:

| Mode | Functionality Low Trip Point | Functionality High Trip Point |
|---|---|---|
| 0 | Setpoint Mode:<br>Value settable by pushbutton and Profibus | Setpoint Mode:<br>Value settable by pushbutton and Profibus |
| 1 | ATM Mode:<br>Trip point value calculated by Factor of ATM | Setpoint Mode:<br>Value settable by pushbutton and Profibus |
| 2 | Setpoint Mode:<br>Value settable by pushbutton and Profibus | ATM Mode:<br>Trip point value calculated by Factor of ATM |
| 4 | Setpoint Mode:<br>Value settable only by Profibus | Setpoint Mode:<br>Value settable only by Profibus |

If only one High Trip Point or Low Trip Point is enabled: Pushbutton works on that Trip Point which is enabled.

If both High Trip Point and Low Trip Point are enabled: Pushbutton works only on Low Trip Point

# 12 Transducer Blocks

## 12.1 CDG Transducer Block

| ID | Name | Structure | Data type | Bytes | Access | Store | → 🗎 |
|---|---|---|---|---|---|---|---|
| 101 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R 2_R | N | 50 |
| 102 | Status Extension | Simple | UINT8 | 1 | 1_R 2_R | V | 50 |
| 103 | Sensor Alarm | Array | UINT8 | 2 | 1_R 2_R | V | 50 |
| 104 | Sensor Warning | Array | UINT8 | 2 | 1_R 2_R | V | 50 |

### 12.1.1 Block Type ID (ID 101)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "CDG Transducer Block" is set to "14". The other defined block types are listed in Appendix B.

### 12.1.2 StatusExtension (ID 102)

This bit mapped byte parameter provides the following additional information of the PCG55x sensor (CDG):

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Res. | Res. | Res. | Res. | Res. | Underrange Exceeded | Overrange Exceeded | Reading Invalid |

Res. = Reserved

- Underrange Exceeded
  This bit is set ("1") if the ProcessValue PV is below the parameter Underrange (ID 45) of the CDG sensor instance (→ 🗎 36].
- Overrange Exceeded
  This bit is set ("1") if the ProcessValue PV is above the parameter Overrange (ID 44) of the CDG sensor instance (→ 🗎 36).
- Reading invalid
  This bit represents the logical inversion of the parameter "Reading Valid" (ID 23) of the CDG sensor instance (→ 🗎 36).

### 12.1.3 Sensor Alarm (ID 103)

This parameter contains the following information:

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| Byte 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SF |
| Byte 1 | 0 | 0 | 0 | 0 | 0 | 0 | EF | 0 |

- SF = Sensor Element Failure
- EF = Electronics Failure

### 12.1.4 Sensor Warning (ID 104)

This byte is always "0". (The CDG Sensor does not produce any warnings)

## 12.2 Pirani Transducer Block

| ID | Name | Structure | Data type | Bytes | Access | Store | → 🗎 |
|---|---|---|---|---|---|---|---|
| 101 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 51 |
| 102 | Status Extension | Simple | UINT8 | 1 | 1_R<br>2_R | V | 51 |
| 103 | Sensor Alarm | Array | UINT8 | 2 | 1_R<br>2_R | V | 51 |
| 104 | Sensor Warning | Array | UINT8 | 2 | 1_R<br>2_R | V | 51 |
| 201 | Pirani OFF | Simple | UINT8 | 1 | 1_R/W<br>2_R/W | N | 51 |

### 12.2.1 Block Type ID (ID 101)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "Pirani Transducer Block" is set to "13". The other defined block types are listed in Appendix B.

### 12.2.2 StatusExtension (ID 102)

This bit mapped byte parameter provides the following additional information of the PxG55x sensor (PIR):

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Res. | Res. | Res. | Res. | Res. | Underrange Exceeded | Overrange Exceeded | Reading Invalid |

Res. = Reserved

- Underrange Exceeded
  This bit is set ("1") if the ProcessValue PV is below the parameter Underrange (ID 45) of the Pirani sensor instance (→ 🗎 39).
- Overrange Exceeded
  This bit is set ("1") if the ProcessValue PV is above the parameter Overrange (ID 44) of the Pirani sensor instance (→ 🗎 39).
- Reading invalid
  This bit represents the logical inversion of the parameter "Reading Valid" (ID 23) of the Pirani sensor instance (→ 🗎 39).

### 12.2.3 Sensor Alarm (ID 103)

This parameter contains the following information:

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| Byte 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SF |
| Byte 1 | 0 | 0 | 0 | 0 | 0 | 0 | EF | 0 |

- SF = Sensor Element Failure
- EF = Electronics Failure

### 12.2.4 Sensor Warning (ID 104)

This byte is always "0". (The PIR Sensor does not produce any warnings)

### 12.2.5 Pirani OFF (ID 201)

This parameter is used to switch the Pirani off or on (by using an automatic or manual mode) in case of special gases used in the process.

| Parameter value | Description |
|---|---|
| 0 | Pirani is always on |
| 1 | Pirani switches off automatically<br>(by default at pressures above 12 mbar - depends on the system configuration) |
| 2 | Pirani is always off |

## 12.3 ATM Transducer Block

This transducer block can only be accessed if the ATM sensor is part of the device configuration (ID 24) (→ 🗎 25).

| ID | Name | Structure | Data type | Bytes | Access | Store | → 🗎 |
|---|---|---|---|---|---|---|---|
| 101 | Block Type ID | Simple | OSTRING(n) | 4 | 1_R<br>2_R | N | 52 |
| 102 | Status Extension | Simple | UINT8 | 1 | 1_R<br>2_R | V | 52 |
| 103 | Sensor Alarm | Array | UINT8 | 2 | 1_R<br>2_R | V | 52 |
| 104 | Sensor Warning | Array | UINT8 | 2 | 1_R<br>2_R | V | 52 |

### 12.3.1 Block Type ID (ID 101)

The Block Type ID Parameter contains an ID which describes the block type. The block type ID of the "ATM Transducer Block" is set to "256". The other defined block types are listed in Appendix B.

### 12.3.2 StatusExtension (ID 102)

This bit mapped byte parameter provides the following additional information of the ATM sensor (ATM):

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Res. | Res. | Res. | Res. | Res. | Underrange Exceeded | Overrange Exceeded | Reading Invalid |

Res. = Reserved

- Underrange Exceeded
  This bit is set ("1") if the ProcessValue PV is below the parameter Underrange (ID 45) of the ATM sensor instance (→ 🗎 42).
- Overrange Exceeded
  This bit is set ("1") if the ProcessValue PV is above the parameter Overrange (ID 44) of the ATM sensor instance (→ 🗎 42).
- Reading invalid
  This bit represents the logical inversion of the parameter "Reading Valid" (ID 23) of the ATM sensor instance (→ 🗎 42).

### 12.3.3 Sensor Alarm (ID 103)

This parameter contains the following information:

| | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|---|
| Byte 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | SF |
| Byte 1 | 0 | 0 | 0 | 0 | 0 | 0 | EF | 0 |

- SF = Sensor Element Failure
- EF = Electronics Failure

### 12.3.4 Sensor Warning (ID 104)

This byte is always "0". (The ATM Sensor does not produce any warnings).

# 13 OneOfN Vacuum Gauge Transducer Block

The following attributes are supported:

| ID | Name | Structure | Data type | Bytes | Access | Store | → |
|---|---|---|---|---|---|---|---|
| 120 | OneOfN Status Extension | Simple | UINT8 | 1 | 1_R<br>2_R | V | 53 |

## 13.1 Information on the Individual Indices

### 13.1.1 OneOfN Status Extension (ID 120)

This parameter indicates whether the overrange or underrange of the gauge is exceeded.

If the gauge is operated in its overrange or underrange, the corresponding bit and additionally the bit "Reading Invalid" is set.

If an error occurs, the bit "Reading Invalid" as well as the corresponding error bits in Device Block (ID 26, 27, 28) are set.

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Underrange exceeded | Overrange exceeded | Reading invalid |

# Appendix A: Definitions

## Data types

| Abbreviation | Range | Data type |
|---|---|---|
| INT8 | $-2^7$ … $(2^7 - 1)$ | Integer 1 byte |
| INT16 | $-2^{15}$ … $(2^{15} - 1)$ | Integer 2 byte |
| INT32 | $-2^{31}$ … $(2^{31} - 1)$ | Integer 4 byte |
| UINT8 | 0 … $(2^8 - 1)$ | Unsigned integer 1 byte |
| UINT16 | 0 … $(2^{16} - 1)$ | Unsigned integer 2 byte |
| UINT32 | 0 … $(2^{31} - 1)$ | Unsigned integer 4 byte |
| FLOAT | $\pm 3.402 \times 10^{38}$ | Floating point, IEEE 754 Short Real Number, 4 byte |
| VSTRING(n) | | Visible string, ISO 646 and ISO 2375 |
| OSTRING(n) | | Octet string |
| Boolean | 0 ... 1 | Boolean |

## Definitions

| Term | Meaning |
|---|---|
| Byte | Number of bytes used by a data structure (integer value) |
| Store | This parameter defines whether the values are stored in non-volatile memory (→ store characteristics) |
| Default | Manufacturer-defined value |

## Store characteristics

| Abbreviation | Meaning |
|---|---|
| V | "Volatile": Value is not saved to the NVRAM or EEPROM and is lost in the event of a power failure |
| N | "Nonvolatile": Value is saved to the NVRAM or EEPROM and is not lost in the event of a power failure |
| D | "Dynamic": Value may frequently change and do not need to be persistent |

## Data access

| Abbreviation | Meaning |
|---|---|
| 1_R/W | Acyclically readable and writeable by a Master Class 1 |
| 2_R/W | Acyclically readable and writeable by a Master Class 2 |
| 1/2_R/W | Acyclically readable and writeable by a Master Class 1 and 2 |
| 1_R | Acyclically readable by a master Class 1 |
| 2_R | Acyclically readable by a master Class 2 |
| 1/2_R | Acyclically readable by a master Class 1 and 2 |
| 1_W | Acyclically writeable by a master Class 1 |
| 2_W | Acyclically writeable by a master Class 2 |
| 1/2_W | Acyclically writeable by a master Class 1 and 2 |
| 0_XI | Cyclic output data with master Class 1 |

Excerpts from:
"PROFIBUS Profile for SEMI"
→ 🕮 [4]

The following table explains terms used in connection with the Profibus.

| Term | Meaning |
|---|---|
| Alert Elements | Alert Elements are used to communicate notification messages from slave to master when warnings, alarms or events are detected. |
| Application | A *software functional unit* consisting of an interconnected aggregation of *function blocks, events and objects,* which may be distributed and which may have *interfaces* with other *applications*. |
| Characteristic | A characteristic is a property or characteristic of an *entity*.<br>(Au) In block applications a block interface is defined by input/output parameters. These parameters have characteristics called parameter characteristics. Examples are access rights and identification names.<br>(IT) The UML defines characteristics as a feature within a classifier that describes a range of values that instances of the classifier may hold. It is a property of a class instance (object). |
| Block<br>(Block Instance) | A logical processing unit of software comprising an individual, named copy of the block and associated parameters specified by a block type, which persists from one invocation of the block to the next. Concept similar to the class/ object approach, but well suited to the automation requirements. |
| Class | (IT) A class represents a template for several objects and describes how these objects are structured internally. Objects of the same class have the same definition both for their operations and for their information structures. |
| Configuration (of a system/device) | A step in system design: selecting functional units, assigning their locations and identifiers and defining their interconnections. |
| Data Structure | An *aggregate* whose elements need not be of the same *data type*, and each of them is uniquely referenced by an *offset identifier.* |
| Data Type | A data item with certain characteristics and permissible operations on that data, e.g. INT8. |
| Device | A physical entity capable of performing one or more specified functions in a particular context and delimited by its interfaces. |
| Direction of Data | *Input data* are transmitted from the device to the bus. *Output data* are transmitted from the bus to the device. |
| Direction of Flow | A positive set point causes a flow from P to A. |
| Entity | A particular thing, such as a person, place, *process*, object, concept, association or *event*. |
| Function | (1) A specific purpose of an entity.<br>(2) One of a group of actions performed by an entity. |
| Function Block | A named *block* consisting of one or more input, output and contained parameters. Function blocks represent the basic automation functions performed by an application which is as independent as possible from the specifics of I/O devices and the network. Each function block processes input parameters according to a specified algorithm and an internal set of contained parameters. They produce output parameters that are available for use within the same function block application or by other function block applications. |
| Function Block Application | Application of an automation system performed by a Device Block, Function Block, Transducer Block and accompanied elements. |

Excerpts from:
"PROFIBUS Profile for SEMI"
(cont.)

| Term | Meaning |
|---|---|
| Instance | A set of data related to an invocation of a function block or a class. |
| Internal Resolution (ir) | The internal resolution is 16383 ($3FFF_{hex}$) for 100% and -16384 ($C000_{hex}$) for -100% of the range. |
| Mode | Determines the block operating mode and available modes for a block instance. |
| Object | (IT) A software entity having identity, attributes and behavior. |
| Parameter | A *variable* that is given a constant value for a specified *application* and that may denote the *application*. |
| Device Block | A Device Block is a named block. Hardware specific parameters of a field device, which are associated with a resource, are made visible through the Device Block. Similar to transducer blocks, they insulate function blocks from the physical hardware by a set of implementation independent hardware parameters. |
| Record | A set of *data items* of different data types treated as a unit. |
| Resource | A resource is considered to be a logical subdivision within the software (and possibly hardware) structure of a device. Resources have independent control of their operation. The definition of a resource may be modified without affecting other resources within a device. A resource accepts and processes data and/or events from the process and/or communication interfaces and returns data and/or events to the process and/or communication interfaces, as specified by the applications utilizing the resource. An interoperable network view of applications is provided through device resources. Each resource specifies the network visible aspects of one or more local applications (or parts of distributed applications). |
| Simple Variable | A single variable which is characterized by a defined Data Type. |
| Substitute Value | In case an optional parameter has not been implemented, the device behaves according to the substitute value for this parameter. |
| Transducer Block | Transducer Block is a named block. Transducer blocks insulate function blocks from the specifics of I/O devices, such as sensors, actuators, and switches. Transducer blocks control access to I/O devices through a device independent interface defined for use by function blocks. Transducer blocks also perform functions, such as calibration and linearization, on I/O data to convert it to a device independent representation. Their interface to function blocks is defined as one or more implementation independent I/O channels. |
| Variable | A *software* entity that may assume any one of a set of values. The values of a variable are usually restricted to a certain data type. |

# Appendix B: Block Type

Currently defined Block Types

| Block Name | Block Type ID |
|---|---|
| Device Block | 1 |
| Sensor Analog Input Function Block | 2 |
| One of N Channel Sensor Analog Input Function Block | 3 |
| Multi Channel Sensor Analog Input Function Block | 4 |
| Discrete Input Function Block | 5 |
| Actuation Analog Output Function Block | 6 |
| Discrete Output Function Block | 7 |
| Analog Output Function Block | 8 |
| Single Stage Controller Function Block | 9 |
| Gas Calibration Transducer Block | 10 |
| Flow Transducer Block | 11 |
| Sensor Analog Input Ambient Temperature Transducer Block | 12 |
| Heat Transfer Vacuum Gauge | 13 |
| Diaphragm Gauge | 14 |
| Cold Cathode Ion Gauge | 15 |
| Hot Cathode Ion Gauge | 16 |
| Trip Point Function Block | 17 |
| OneOfN Vacuum Pressure Gauge Transducer Block | 18 |
| Reserved | 19 … $2^8$ - 1 |
| Manufacturer-specific | $2^8$ … $2^{16}$ - 1 |

# Appendix C: Literature


🕮 [1] www.inficon.com
Operating Manual
PCG550, PCG552, PCG554
tina56d1 (German)
tina56e1 (English)
INFICON AG, LI–9496 Balzers, Liechtenstein

🕮 [2] www.inficon.com
Operating Manual
PSG550, PSG552, PSG554
tina60d1 (German)
tina60e1 (English)
INFICON AG, LI–9496 Balzers, Liechtenstein

🕮 [3] www.profibus.com
Profibus user organization

🕮 [4] www.profibus.com
Profibus profile guidelines
Part 1 - Identification & Maintenance Functions

🕮 [5] IEC 61158 Type 3 elements: Industrial communication networks – Fieldbus specifications
IEC 61784: Industrial communication networks – Fieldbus profiles

Notes

Original: English tira56e1 (2010-02)

t i r a56e1

INFICON
LI–9496 Balzers
Liechtenstein
Tel +423 / 388 3111
Fax +423 / 388 3700
reachus@inficon.com

www.inficon.com