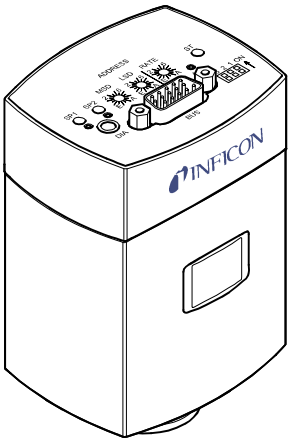
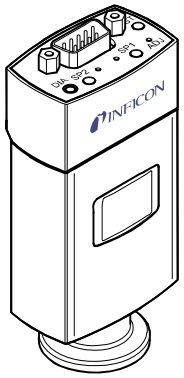
RS232C / RS485C

Serial Interface for Pirani Capacitance Diaphragm and Pirani Standard Gauges

PCG550, PCG552, PCG554
PSG550, PSG552, PSG554



RS485C



RS232C

General Information

The Serial Interface allows the communication of the digital INFICON Pirani Capacitance Diaphragm Gauges (PCG550, PCG552, PCG554) and the digital INFICON Pirani Standard Gauges (PSG550, PSG552, PSG554) with

- an INFICON Vacuum Gauge Controller (VGC series) or with
- another appropriate controller.

The RS232C or RS485C interface integrated in the gauge allows to digitally transmit measurement values and information on the gauge status as well as to make parameter settings.

Caution

Caution: data transmission errors

Any attempt to simultaneously operate the gauge via the Serial Interface and a fieldbus interface (DeviceNet, 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 and DeviceNet, Profibus, RS485C or diagnostic port.

Interface Protocol

The protocols of the RS232C and RS485C interfaces are identical except that in the RS485C protocol, the address is added.

The Serial Interface is used in Master-Slave mode. Without a corresponding request from the Master, the device does not transmit any data.

Instructions to the gauge are transmitted via binary protocol.

Data format

- binary
- 8 data bits
- 1 stop bit
- no parity bit
- no handshake

RS232 pin assignment

- RS232, TxD Pin 13
- RS232, RxD Pin 14
- Supply common Pin 4
  (sensor cable connector)

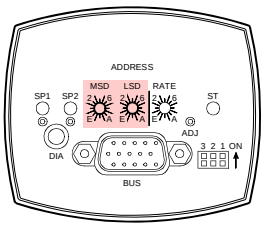
RS232 address and transmission rate

The address of the RS232C interface need not be set. For the protocol 0 is always used. The transmission rate can be set as parameter (→ 🗎 6).

RS485 pin assignment

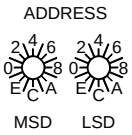
- RS485 B+ Pin 1
- RS485 A– Pin 2
- RS485 +5 V Pin 7
- RS485 GND Pin 8
  (sensor cable connector)

RS485 address <MSD>, <LSD>



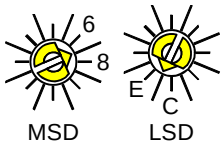
The node address is an unambiguous device address in the RS485 network. The gauge can only communicate with the network if the node address setting is correct. Valid address range: 0 … 255 in decimal form.

The node address setting is made on the gauge in hexadecimal form (00 … FF_{hex}) by means of two rotary switches.



<MSD>: upper half byte setting
<LSD>: lower half byte setting
Factory setting: 00_{hex}.

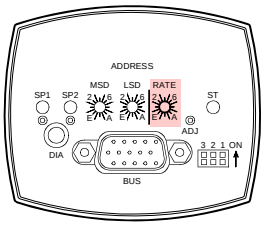
Example: Node address = $7D_{hex}$:



When the gauge is put into operation, the firmware queries its node address.

If the node address is modified during operation it is taken over immediately.

RS485 transmission rate <RATE>

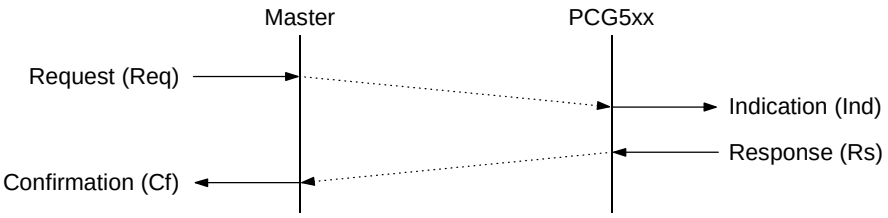


The transmission rate is set with the rotary switch <RATE>.

Position	Baud
1	9600
2	19200
3	38400
4	57600
0	reserved
5 … F	reserved

If the transmission rate is modified during operation, a reset occurs and the new transmission rate is taken over.

Data Link Layer



The Data Link Layer uses the following terminology:

request (Req)
A request is an instruction (to read/or write) transmitted by the Master.

indication (Ind)
An indication means that the Slave (gauge) recognizes a request from the Master.

response (Rs)
A response is an answer from the Slave to the Master.

confirmation (Cf)
A confirmation is an acknowledgement of the response by the Master.

Protocol Frame

Every protocol layer of the communication protocol is represented in a protocol frame. The maximum length per frame is 64 byte. The data field of the data link layer consists of

- a command (Cmd)
- a parameter identifier (PID)
- a data field (Data).

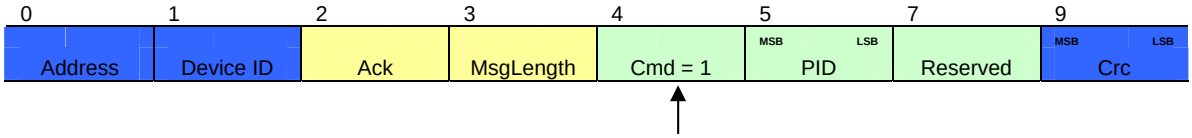
The command determines whether the data transmitted are read or write requests.

Command (Cmd)

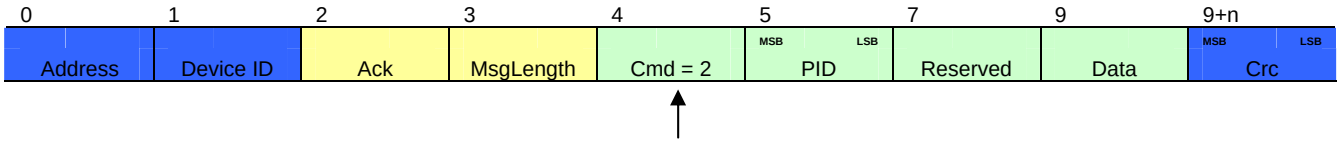
There are four different command types:

Cmd 1	Read request from the Master
Cmd 2	Read response from the gauge to a read request
Cmd 3	Write request
Cmd 4	Write response from the gauge to a write request

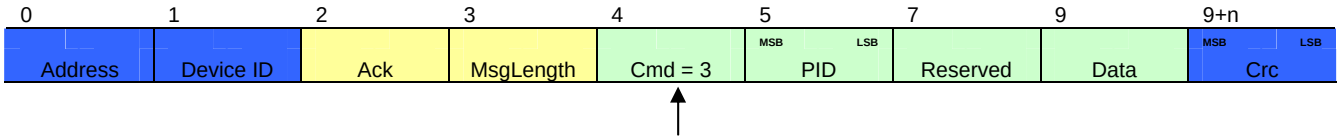
- Cmd 1: Read request from the Master



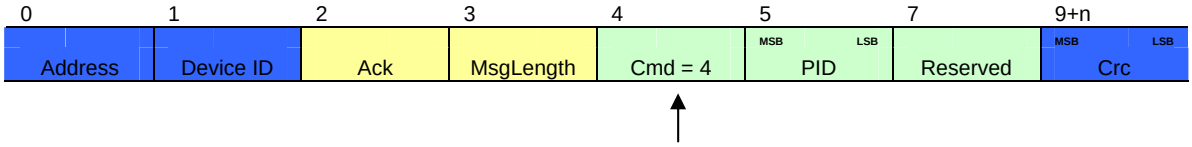
- Cmd 2: Response from the gauge to a read request



- Cmd 3: Write request from a Master



- Cmd 4: Response from the gauge to a write request



Parameter Identifier (PID)

The parameter identifier (PID) addresses a defined parameter of the device.

Data field (Data)

The data field contains the data of a request. In the case of a write request it contains the data transmitted from the Master to the gauge, in the case of a read request it contains the data to be transmitted from the gauge to the Master. Data are transmitted in Big Endian.

Medium Access Layer

The medium access layer contains the following data fields:

- RS485 address (for RS232 the address is 0)
- Device ID (Master 0, PCG5xx 2)
- Header with Ack and Message Length (all APDU data are counted: Cmd, PID, Reserved, Data)
- 16 bit CRC (example → 🗎 10)

Communication Error

If a communication error occurs during transmission the PID is set to 0xFFFF and an error byte is added.

Byte 0, errors

Error Code	Description
1	Access error
2	Value higher max. or lower min.
3	Parameter not found
4	Length error
6	Memory access error
7	Memory access timeout

Protocol Description

The following parameters can be read via a serial interface.

Output pressure

PID	Name	Description	Factory setting	Min.	Max.	Type
221	Pressure	Output pressure in integer format				Fixs32en20
222	Pressure	Pressure in vacuum chamber in Real format				Real32
265	ATM Pressure	Ambient pressure in real format				Real32
466	Differential Pressure	Differential pressure outside and inside vacuum chamber				Real32
224	Data Unit	Changes the pressure unit of all Real32 pressures. 0 mbar 1 Torr 2 Pascal 3 micron 4 Counts	0	0	4	Uint8

Error

PID	Name	Description	Factory setting	Min.	Max.	Type
228	Device Exception	0 No error 1 Timeout EEPROM memory access 2 EEPROM CRC error 3 EEPROM error 4 Pirani filament rupture 5 Wrong filament material 6 CDG diaphragm rupture 8 ATM outside specification limits 11 Sensor does not match gauge	0			Uint8

General information

PID	Name	Description	Factory setting	Min.	Max.	Type
103	Reset	0 Write The sensor carries out a reset. 1 Write The sensor resets all parameters to their factory settings.				
104	Run Hours	Operating hours [0.25 h]				Fixs32en2
207	Serial Number	Serial number			4294967295	Uint32
208	Product Name	Product name	PCG550			String
209	Manufacturers Name	Name of manufacturer	INFICON AG			String
210	Manufacturers Model Number	Model number				String
218	Software Version	Software version				String
227	RS232 Baud Rate	9600, 19200, 38400, 57600	57600	9600	57600	Uint32
243	Display Direction	Display orientation 0 Setting for flange in bottom position 1 Setting for flange in top position	0	0	1	Uint8

Sensor details

PID	Name	Description	Factory setting	Min.	Max.	Type
223	Active Instance Number	Current sensor 1 CDG sensor active 2 Pirani sensor active 3 Mixed signal range Pirani and CDG sensor active				Uint8
33000	Pirani Full Scale	Pirani fullscale	1000			Fixs32en20
33001	Pirani Overrange Value	Pirani overrange status output limit	1000			Fixs32en20
33002	Pirani Underrange Value	Pirani underrange status output limit	5.00E-05			Fixs32en20
255	Pirani Safe State	Pressure reading in mbar that is output in the event of an error in Pirani mode 0 0 mbar 1 1500 mbar 2 The last valid value is retained. 3 The Pirani Safe State value is output.	0	0	3	Uint8
256	Pirani Safe State Value	Pressure reading in mbar that is output in the event of an error if the Pirani Safe State is set to 3.	0	0	2047	Fixs32en20
417	Pirani Adjust Flag	Executes a manual Pirani sensor adjustment. Set the value to "1" in order for the adjustment to be executed.	0			Uint8
236	CDG Safe State*	Pressure reading in mbar that is output in the event of an error in CDG mode. 0 0 mbar 1 1500 mbar 2 The last valid value is retained. 3 The CDG Safe State value is output.	0	0	3	Uint8
237	CDG Safe State Value*	Pressure reading in mbar that is output in the event of an error if the CDG Safe State is set to 3.	0	0	2047	Fixs32en20

(continued)

(concluded)

PID	Name	Description	Factory setting	Min.	Max.	Type
421	CDG Auto Zero Adjust*	0 No automatic CDG adjustment 1 Automatic CDG adjustment	1	0	1	Uint8
414	CDG Zero Adjust Flag*	Executes a manual CDG sensor adjustment. Set the value to "1" in order for the adjustment to be executed.	0			Uint8
34000	CDG Full Scale*	CDG full scale	1500			Fixs32en20
34001	CDG Overrange Value*	If the pressure exceeds this value the measurement is invalid.	1500			Fixs32en20
34002	CDG Underrange Value*	If the pressure is below this value the measurement is invalid.	1			Fixs32en20
264	ATM Pressure*	Output pressure in integer format				Fixs32en20
265	ATM Pressure*	Output pressure in Real format				Real32
267	ATM Full Scale*	ATM fullscale value	1150			Fixs32en20
270	ATM Overrange Value*	Limit for determining the overrange status	1150			Fixs32en20
271	ATM Underrange Value*	Limit for determining the underrange status	150			Fixs32en20
274	ATM Status Extention*	ATM sensor status Bit 2 underrange exceeded Bit 1 overrange Bit 0 exceeded reading invalid				Uint8
448	ATM Adjust Flag*	Adjustment of atmospheric pressure sensor Set the value to "1" in order for the adjustment to be executed. The adjustment has to be carried out while the vacuum chamber is at atmospheric pressure. When the atmospheric pressure sensor is adjusted it has to indicate the same pressure reading as the CDG pressure reading of the vacuum chamber at atmospheric pressure.	0			Uint8

* Parameter not available for PSG55x devices.

Setpoints

PID	Name	Description	Factory setting	Min.	Max.	Type
275	Setpoint 1 High Trip Point	Setpoint High Trip Point in mbar	1500	5.00E-04	1500	Fixs32en20
276	Setpoint 1 High Trip Point Enable	0 Off 1 On	1	0	1	Uint8
277	Setpoint 1 Low Trip Point	Setpoint Low Trip Point in mbar	5.00E-05	5.00E-05	1500	Fixs32en20
278	Setpoint 1 Low Trip Point Enable	0 Off 1 On	1	0	1	Uint8
279	Setpoint 1 Status	Status of Relay 1	0			Uint8
281	Setpoint 1 ATM Factor	Factor for ATM setpoint The atmospheric pressure is multiplied by this factor and used as Setpoint 1. To activate the ATM mode select the Setpoint 1 mode.	1.1	0	3	Fixs32en20
282	Setpoint 2 High Trip Point	Setpoint High Trip Point in mbar	1500	5.00E-04	1500	Fixs32en20
283	Setpoint 2 High Trip Point Enable	0 Off 1 On	1	0	1	Uint8
284	Setpoint 2 Low Trip Point	Setpoint Low Trip Point in mbar	5.00E-05	5.00E-05	1500	Fixs32en20
285	Setpoint 2 Low Trip Point Enable	0 Off 1 On	1	0	1	Uint8
286	Setpoint 2 Status	Status of Relay 1	0			Uint8
288	Setpoint 2 ATM Factor	Factor for the ATM setpoint The atmospheric pressure is multiplied by this factor and used as Setpoint 2. To activate the ATM mode select the Setpoint 2 mode.	1.1	0	3	Fixs32en20
455	Setpoint 1 Mode	Setpoint mode 0 Setpoint (Low / High Trip mMode) 4 Setpoint (Low / High Trip mode). In addition, the setpoint adjustment buttons are disabled. 1 The Low Trip Point is in ATM mode. 5 The Low Trip Point is in ATM mode. In addition, the setpoint adjustment buttons are disabled. 2 The High Trip Point is in ATM mode. 6 The High Trip Point is in ATM mode. In addition, the setpoint adjustment buttons are disabled. 3/7 Reserved function	0	0	7	Uint8
456	Setpoint 2 Mode	Setpoint mode 0 Setpoint (Low / High Trip mode) 4 Setpoint (Low / High Trip mode). In addition, the setpoint adjustment buttons are disabled. 1 The Low Trip Point is in ATM mode. 5 The Low Trip Point is in ATM mode. In addition, the setpoint adjustment buttons are disabled. 2 The High Trip Point is in ATM mode. 6 The High Trip Point is in ATM mode. In addition, the setpoint adjustment buttons are disabled. 3/7 Reserved function	0	0	7	Uint8

(continued)

(concluded)

PID	Name	Description	Factory setting	Min.	Max.	Type
457	High Trip Point 1 Hysteresis	Hysteresis High Trip Point 1 [mbar]	1.00E+01	5.00E-05	1500	Fixs32en20
458	Low Trip Point 1 Hysteresis	Hysteresis Low Trip Point 1 [mbar]	5.00E-05	5.00E-05	1500	Fixs32en20
459	High Trip Point 2 Hysteresis	Hysteresis High Trip Point 2 [mbar]	1.00E+01	5.00E-05	1500	Fixs32en20
460	Low Trip Point 2 Hysteresis	Hysteresis Low Trip Point 2 [mbar]	5.00E-05	5.00E-05	1500	Fixs32en20
461	Setpoint 1 Extended Status	Extended status of Setpoint 1 0 Not active 1 Low Trip Point active 2 High Trip Point active 3 High and Low Trip Point active	0			Uint8
462	Setpoint 2 Extended Status	Extended status of Setpoint 2 0 Not active 1 Low Trip Point active 2 High Trip Point active 3 High and Low Trip Point active	0			Uint8

Data Format

Fixs32enXX

To calculate the result from a Fixx32enXX value, multiply or divide it by a factor of 2^{XX}.

Example Fixs32en20

The hysteresis of the High Trip Point Setpoint 1 is to be set to 10 mbar. For this purpose 10 is multiplied by 2^{20} (Fixs32en20). The value 10485760 is therefore transmitted.

Examples

The following examples show how read and write requests are made.

Read request (reading a pressure)

The required parameter has PID 221.

From Master to PCG5xx:

0	1	2	3	4	5	7	9	9+n
Address	Device ID	Ack	MsgLength	Cmd	PID (MSB LSB)	Reserved	Data	Crc (MSB LSB)
0x00	0x00	0x00	0x05	0x01	0x00DD	0x0000	-	0xAB21

From PCG5xx to Master:

0	1	2	3	4	5	7	9	9+n
Address	Device ID	Ack	MsgLength	Cmd	PID (MSB LSB)	Reserved	Data	Crc (MSB LSB)
0x00	0x02	0x01	0x09	0x02	0x00DD	0x0000	0x375A05BF	0xD9BB

Write request (setting a unit)

The required parameter has PID 224. To set the unit to Torr, 1 must be written.

From Master to PCG5xx:

0	1	2	3	4	5	7	9	9+n
Address	Device ID	Ack	MsgLength	Cmd	PID (MSB LSB)	Reserved	Data	Crc (MSB LSB)
0x00	0x00	0x00	0x06	0x03	0x00E0	0x0000	0x01	0x346D

From PCG5xx to Master:

0	1	2	3	4	5	7	9	9+n
Address	Device ID	Ack	MsgLength	Cmd	PID (MSB LSB)	Reserved	Data	Crc (MSB LSB)
0x00	0x02	0x01	0x05	0x04	0x00E0	0x0000	-	0x94EA

Calculating CRC

The data packages are secured with CRC16 in Little Endian.

CRC polynominal	0x8408
CRC initial value	0xFFFF

CRC Example Code

To illustrate how the checksum can be calculated and verified, an example of a code is given below:

```
using System;

class Program
{
    static void Main()
    {
    // the following test array is a valid pcg5xx frame with crc16 at the end
    byte[] arr = new byte[] { 0x00, 0x00, 0x00, 0x05, 0x01, 0x00, 0xDD, 0x00, 0x00, 0xAB, 0x21};
    UInt16 crc;
    Boolean b;
    // Calculate the crc of the test array arr, of course without crc (therefore length minus 2)
    crc = Crc16.Create(arr, (Byte)(arr.Length - 2));
    // Check, if the test array has a correct crc at the end (it is correct, therefore the returnvalue is
true)
    b = Crc16.Check(arr, (Byte)arr.Length);
    }
}

public class Crc16
{
    // initial value for crc16
    public static UInt16 inital = 0xFFFF;

    // function to create a crc16
    public static UInt16 Create(Byte[] buffer, Byte length)
    {
        UInt16 crc16 = new UInt16();
        UInt16 i = 0;

        // Initial Value for CRC calculation
        crc16 = Crc16.inital;

        while (i < length)
        {
            crc16 = (UInt16)((Crc16.crc16Tab[(crc16 ^ buffer[i++]) & (Byte)0xFF]) ^ (crc16 >> 8));
        }
        return crc16;
    }

    // function to check a buffer with a crc16 at the end
    public static Boolean Check(Byte[] buffer, Byte length)
    {
        UInt16 crc16 = Crc16.inital;
        UInt16 i = 0;
        // calculate crc for the buffer without crc
        while (i < length)
        {
            crc16 = (UInt16)((Crc16.crc16Tab[(crc16 ^ buffer[i++]) & (Byte)0xFF]) ^ (crc16 >> 8));
        }
        if (crc16 == 0)
        {
            return true;
        }
        return false;
    }
    // crc array
    public static UInt16[] crc16Tab =
    {
        0x0000, 0x1189, 0x2312, 0x329B, 0x4624, 0x57AD, 0x6536, 0x74BF,
        0x8C48, 0x9DC1, 0xAF5A, 0xBED3, 0xCA6C, 0xDBE5, 0xE97E, 0xF8F7,
        0x1081, 0x0108, 0x3393, 0x221A, 0x56A5, 0x472C, 0x75B7, 0x643E,
        0x9CC9, 0x8D40, 0xBFDB, 0xAE52, 0xDAED, 0xCB64, 0xF9FF, 0xE876,
        0x2102, 0x308B, 0x0210, 0x1399, 0x6726, 0x76AF, 0x4434, 0x55BD,
        0xAD4A, 0xBCC3, 0x8E58, 0x9FD1, 0xEB6E, 0xFAE7, 0xC87C, 0xD9F5,
        0x3183, 0x200A, 0x1291, 0x0318, 0x77A7, 0x662E, 0x54B5, 0x453C,
        0xBDCB, 0xAC42, 0x9ED9, 0x8F50, 0xFBEF, 0xEA66, 0xD8FD, 0xC974,
        0x4204, 0x538D, 0x6116, 0x709F, 0x0420, 0x15A9, 0x2732, 0x36BB,
        0xCE4C, 0xDFC5, 0xED5E, 0xFCD7, 0x8868, 0x99E1, 0xAB7A, 0xBAF3,
        0x5285, 0x430C, 0x7197, 0x601E, 0x14A1, 0x0528, 0x37B3, 0x263A,
        0xDECD, 0xCF44, 0xFDDF, 0xEC56, 0x98E9, 0x8960, 0xBBFB, 0xAA72,
```

```
        0x6306, 0x728F, 0x4014, 0x519D, 0x2522, 0x34AB, 0x0630, 0x17B9,
        0xEF4E, 0xFEC7, 0xCC5C, 0xDDD5, 0xA96A, 0xB8E3, 0x8A78, 0x9BF1,
        0x7387, 0x620E, 0x5095, 0x411C, 0x35A3, 0x242A, 0x16B1, 0x0738,
        0xFFCF, 0xEE46, 0xDCDD, 0xCD54, 0xB9EB, 0xA862, 0x9AF9, 0x8B70,
        0x8408, 0x9581, 0xA71A, 0xB693, 0xC22C, 0xD3A5, 0xE13E, 0xF0B7,
        0x0840, 0x19C9, 0x2B52, 0x3ADB, 0x4E64, 0x5FED, 0x6D76, 0x7CFF,
        0x9489, 0x8500, 0xB79B, 0xA612, 0xD2AD, 0xC324, 0xF1BF, 0xE036,
        0x18C1, 0x0948, 0x3BD3, 0x2A5A, 0x5EE5, 0x4F6C, 0x7DF7, 0x6C7E,
        0xA50A, 0xB483, 0x8618, 0x9791, 0xE32E, 0xF2A7, 0xC03C, 0xD1B5,
        0x2942, 0x38CB, 0x0A50, 0x1BD9, 0x6F66, 0x7EEF, 0x4C74, 0x5DFD,
        0xB58B, 0xA402, 0x9699, 0x8710, 0xF3AF, 0xE226, 0xD0BD, 0xC134,
        0x39C3, 0x284A, 0x1AD1, 0x0B58, 0x7FE7, 0x6E6E, 0x5CF5, 0x4D7C,
        0xC60C, 0xD785, 0xE51E, 0xF497, 0x8028, 0x91A1, 0xA33A, 0xB2B3,
        0x4A44, 0x5BCD, 0x6956, 0x78DF, 0x0C60, 0x1DE9, 0x2F72, 0x3EFB,
        0xD68D, 0xC704, 0xF59F, 0xE416, 0x90A9, 0x8120, 0xB3BB, 0xA232,
        0x5AC5, 0x4B4C, 0x79D7, 0x685E, 0x1CE1, 0x0D68, 0x3FF3, 0x2E7A,
        0xE70E, 0xF687, 0xC41C, 0xD595, 0xA12A, 0xB0A3, 0x8238, 0x93B1,
        0x6B46, 0x7ACF, 0x4854, 0x59DD, 0x2D62, 0x3CEB, 0x0E70, 0x1FF9,
        0xF78F, 0xE606, 0xD49D, 0xC514, 0xB1AB, 0xA022, 0x92B9, 0x8330,
        0x7BC7, 0x6A4E, 0x58D5, 0x495C, 0x3DE3, 0x2C6A, 0x1EF1, 0x0F78
    };
}
```

Original: German tira59d1 (2010-07)

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

www.inficon.com