PENKO Engineering B.V.

Your Partner for Fully Engineered Factory Solutions

EtherNet/IP™

Protocol description:
PENKO 1020 EtherNet/IP

PENKO
an ETC Company

Table of Contents

PENKO EtherNet/IP protocol

Introduction

EtherNet/IP(TM) is an industrial Ethernet network that combines standard Ethernet technologies with the media-independent Common Industrial Protocol or "CIP."

This document describes the PENKO EtherNet/IP implementation for the PENKO 1020 device range.

1 Basics

The EDS information, PENKO devices that support the protocol and the available classes.

1.1 EDS

This document describes the Electronic Data Sheet revision 1.2.0 and the 1020 class revision 1.2.

EDS filename: **1020 V1.2.0.EDS**

1.2 Devices

The following PENKO devices support this EDS version:

Device	EIP
1020 IND	Yes, starting at version **v1.5.0.9.0.5**
1020 BLT	Yes, starting at version **v1.5.0.9.0.5**
1020 CHK	Yes, starting at version **v1.5.0.9.0.5**
1020 MFL	Yes, starting at version **v1.5.0.9.0.5**
1020 FMD	Yes, starting at version **v1.5.0.9.0.5**

1.3 Classes

The following classes are supported:

Classes
Class 0x01 - Identity
Class 0x02 - Message router
Class 0x04 - Assembly
Class 0x06 - Connection manager
Class 0xF5 - TCP/IP
Class 0x300 - Weigher

For every class the following items are described:

- Class attributes
- Class services/actions for these attributes
- Instance attributes
- Instance services/actions for these attributes

2 Class 0x01 (1) - Identity

According to the EIP standard, every PENKO device has an Identity object. The devices support one instance of the Identity object.

2.1 Class Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Revision	UINT	1	Revision of this object.
2	Get	Max instance	UINT	1	Maximum instance number of an object currently created in this class level of the device.
3	Get	Number of instance	UINT	1	Number of object instances currently created at this class level of the device.
6	Get	Max ID Number Class Attributes	UINT	7	The attribute ID number of the last class attribute of the class definition implemented in the device.
7	Get	Maximum ID Number Instance Attributes	UINT	7	The attribute ID number of the last instance attribute of the class definition implemented in the device.

2.2 Class Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read class attribute 1,2,6,7
5	Reset	UINT	
		0	Emulate as closely as possible cycling power on the item the *Identity Object* represents. This value is the default if this parameter is omitted.
		1	Return to the factory default configuration, then emulate cycling power.
14	Get Attribute single	-	Read selected class attribute

2.3 Instance Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Vendor ID	UINT	1240	Identification of each vendor by number
2	Get	Device Type	UINT	12	Indication of general type of product
3	Get	Product Code	UINT	203	Identification of a particular product of an individual vendor
4	Get	Revision Major Revision Minor Revision	STRUCT OF UINT UINT	 1 4	Revision of the item the Identity Object represents
5	Get	Status	WORD		
6	Get	Serial Number	UDINT		Production generated serial number
7	Get	Product Name	SHORT STRING	Penko1020	Human readable identification

2.4 Instance Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read attribute 1-7
5	Reset	UINT 0 1	 Emulate as closely as possible cycling power on the item the *Identity Object* represents. This value is the default if this parameter is omitted. Return to the factory default configuration, and then emulate cycling power.
14	Get Attribute single	-	Read selected attribute
125	Execute PDI	Request BYTE ARRAY[] path Reply BYTE ARRAY[] path + BYTE ARRAY[] result	Executes the PDI interface, see examples chapter.

3 Class 0x02 (2) - Message router

The object within a node that distributes explicit message requests to the appropriate application objects.

3.1 Class Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Revision	UINT	1	Revision of this object.
2	Get	Max instance	UINT	1	Maximum instance number of an object currently created in this class level of the device.
3	Get	Number of instance	UINT	1	Number of object instances currently created at this class level of the device.
6	Get	Max ID Number Class Attributes	UINT	7	The attribute ID number of the last class attribute of the class definition implemented in the device.
7	Get	Maximum ID Number Instance Attributes	UINT	0	The attribute ID number of the last instance attribute of the class definition implemented in the device.

3.2 Class Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read class attribute 1,2,3,6,7
14	Get Attribute single	-	Read selected class attribute

ⓘ **Instance attributes and services are not applicable for this class**

4 Class 0x04 (4) - Assembly

Assembly objects provide the option of mapping data from attributes of different instances of various classes into one single attribute, an Assembly Object. This mapping is generally used for I/O messages to maximize the efficiency of the control data exchange on the network.

4.1 Class Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Revision	UINT	2	Revision of this object.
2	Get	Max instance	UINT	9	Maximum instance number of an object currently created in this class level of the device.
3	Get	Number of instance	UINT	9	Number of object instances currently created at this class level of the device.

4.2 Class Services

Service code	Name	Parameters	Description
14	Get Attribute single	-	Read selected attribute

4.3 Instance Attributes

Number	Access	Name	Data type	Value	Description
3	Get	Byte array[]	BYTE ARRAY[]		Instance depended data, see instance # See examples chapter

Available instances
Instance 0x0310 (784) Weigher configuration
Instance 0x0311 (785) Weigher
Instance 0x0321 (801) Heartbeat
Instance 0x0360 (864) Device configuration
Instance 0x0364 (868) Device in
Instance 0x0368 (872) Device out
Instance 0x0370 (880) Control configuration
Instance 0x0374 (884) Control in
Instance 0x0378 (888) Control out

Instance 0x0310 (784) Weigher configuration

Access	Name	Data type	Description
	Weigher Configuration	STRUCT OF BYTE[]	Reserved for future operation

Instance 0x0311 (785) Weigher

Access	Name	Data type	Description
Get	Weigher	STRUCT OF	
		DINT WEIGHER	Display rate weigher data
		DINT GROSS	Fast Gross weight
		DINT NET	Fast Net weight
		DINT TARE	Active Tare weight
		DINT WEIGHERx10	Display rate weigher data x10
		DINT GROSSx10	Fast Gross weight x10
		DINT NETx10	Fast Net weight x10
		DINT TAREx10	Active Tare weight x10
		WORD FORMAT	Format bits, see Weigher-Format word
		WORD STATUS	Status bits, see Weigher-Status word

Instance 0x0321 (801) Heartbeat

Access	Name	Data type	Description
	Heartbeat		Heartbeat

Instance 0x0360 (864) Device configuration

Access	Name	Data type	Description
Get/Set	Device Configuration	STRUCT OF	
	Indicator configuration	WORD offset read	Indicator configuration read offset Default read 1-20
	Register configuration	WORD offset read	Registers configuration offset Default read External Registers 1-10
	Markers	WORD offset input	Markers configuration offset Default read start at 401 (multiple of 8)

Instance 0x0364 (868) Device in

Access	Name	Data type	Description
Get	Device In	STRUCT OF	
	Weigher	DINT WEIGHER	Display rate weigher data
		DINT GROSS	Fast Gross weight
		DINT NET	Fast Net weight
		DINT TARE	Active Tare weight
		DINT WEIGHERx10	Display rate weigher data x10
		DINT GROSSx10	Fast Gross weight x10
		DINT NETx10	Fast Net weight x10
		DINT TAREx10	Active Tare weight x10
		WORD FORMAT	Format bits, see Weigher-Format word
		WORD STATUS	Status bits, see Weigher-Status word
	Indicator	ARRAY[20] OF INDICATOR	Read indicators, default start read at 1
	Register read	ARRAY OF DINT[10]	Registers [10], 1020 indicator: Register 1= Output level 1 Register 2= Output level 2 Register 3= Output level 3 Register 4= Output level 4 Register 5-10= spare= 0
Controller software	[1]Markers Input	BYTE ARRAY[4]	Markers 4x8=32 default read at 401-432
Indicator software	[2]Inputs [2]Outputs	BYTE ARRAY[2] BYTE ARRAY[2]	Inputs 1-16 Outputs 201-216

[1] *In case of controller software the instance numbers are used as markers input*

[2] *In case of indicator software the instance numbers are used as inputs and outputs*

Instance 0x0368 (872) Device out

Access	Name	Data type	Description
Set	Device Out	STRUCT OF	
	Weigher Control	BYTE[2]	Weigher control word, see also Weigher-Control word
	Reserved Control	BYTE[2]	Set to 0x0000

Instance 0x0370 (880) Control configuration

Access	Name	Data type	Description
Get/Set	Control Configuration	STRUCT OF	
	Indicator configuration	WORD offset read	Indicator configuration read offset, Default read at 1-10
	Register configuration	WORD offset read WORD offset write	Registers configuration offset, not used in 1020 indicator
	Markers	WORD offset input WORD offset output	Markers configuration offset Default read start at 401 (multiple of 8) Default write start at 433 (multiple of 8)

Instance 0x0374 (884) Control in

Access	Name	Data type	Description
Get	Control In	STRUCT OF	
	Weigher	DINT WEIGHER DINT GROSS DINT NET DINT TARE DINT WEIGHERx10 DINT GROSSx10 DINT NETx10 DINT TAREx10 WORD FORMAT WORD STATUS	Display rate weigher data Fast Gross weight Fast Net weight Active Tare weight Display rate weigher data x10 Fast Gross weight x10 Fast Net weight x10 Active Tare weight x10 Format bits, see Weigher-Format word Status bits, see Weigher-Status word
	Indicator	ARRAY[20] OF STRUCT OF INDICATOR	Read indicators, default start read at 1
	Register read	ARRAY OF DINT[10]	Registers [10], 1020 indicator : Register 1= Output level 1 Register 2= Output level 2 Register 3= Output level 3 Register 4= Output level 4 Register 5-10= spare= 0
	Markers Input	BYTE ARRAY[4]	Markers 4x8=32 default read at 401-432

Instance 0x0378 (888) Control out

Access	Name	Data type	Description
Set	Control Out	STRUCT OF	
	Weigher Control	ARRAY OF BYTE[2]	Weigher control word, see also Weigher-Control word
	Reserved Control	ARRAY OF BYTE[2]	Set to 0x0000
	Register write	ARRAY OF DINT[10]	Registers [10], 1020 indicator : Register 1= Output level 1 Register 2= Output level 2 Register 3= Output level 3 Register 4= Output level 4 Register 5-10= spare= 0
	Markers Output	BYTE ARRAY[4]	Markers 4x8=32 default write at 433-464

4.4 Indicator Data Type Definition

```
TYPE OF INDICATOR:
                        STRUCT OF
                                      BYTE FMTSTAT
                                      BYTE WEIGHT [3];
```

4.5 Weigher-Status word

Bit #	Called	Definition
0	OVERLOAD	Hardware overload/underload detected on loadcell
1	MAXLOAD	Overload detected on loadcell
2	STABLE	Weigher signal is stable
3	STABLE RANGE	Weigher signal is in stable range
4	ZERO SET	Weigher zero is corrected
5	ZERO CENTER	Weigher in center of zero range
6	ZERO RANGE	Weigher is in zero range, zero is possible
7	ZERO TRACK	Weigher signal is in zero tracking range, zero tracking is possible
8	TARE	Weigher tare is active
9	PTARE	Weigher preset tare is active
10	SAMPLE	Used by internal process handling
11	BAD CAL	Calibration is bad, invalid, not available
12	CAL ENABLED	Calibration is enabled, used by internal process handling
13	INDUSTRIAL	If set weigher runs in industrial mode, if reset weigher runs certified operation mode
14	NOT LEVEL	Weigher system in blocking, warming up or scale is not level
15	RESERVED	Reserved mode always 0

4.6 Weigher-Control word

Bit #	Called	Definition
0	ZERO_RESET*	Reset the actual zero weight, condition only possible in noncertified mode
1	ZERO_SET*	Activate new zero weight, condition stable signal
2	TARE_OFF*	Switch actual tare weight off
3	TARE_ON*	Activate new tare weight, condition stable signal
4	TARE_TOGGLE*	Toggle the Tare weight on condition stable signal, off condition none
5-16	RESERVED	Reserved bits always 0

*Remark: action on rising edge of bit

4.7 Weigher-Format word

Bit number	Description
#15	Signed/unsigned
	0 = Unsigned
	1 = Signed
#14	Zero suppressing
	0 = Nonzero suppressing
	1 = Zero suppressing
#11 - #8	Display step size
	0000 = Step 1
	0001 = Step 2
	0010 = Step 5
	0011 = Step 10
	0100 = Step 20
	0101 = Step 50
	0110 = Step 100
	0111 = Step 200
	1000 = Step 500
	1001 = Step 1000
	1010 = Step 2000
	1011 = Step 5000
#2 - #0	Decimal point position
	000 = 000000
	001 = 00000.0
	010 = 0000.00
	011 = 000.000
	100 = 00.0000
	101 = 0.00000

4.8 Instance Services

Service code	Name	Parameters	Description
14	Get Attribute single	-	Read selected attribute
16	Set Attribute single	Data	Write selected attribute

4.9 Exclusive Owner Connections

Connection	Name	Assembly O->T	Assembly T->O	Assembly Configuration	Description
1	DEVICE	872	868	864	Read Weigher, Read Indicators, Read Registers, Read Markers and Weigher Control.
2	CONTROL	888	884	880	Read Weigher, Read Indicators, Read/Write Registers, Read/Write Markers, Weigher Control.

4.10 Input Only Connections

Connection	Name	Assembly O->T	Assembly T->O	Assembly Configuration	Description
1	Weigher	801	785	768	Weigher 1 live data

5 Class 0x06 (6) - Connection manager

The Connection manager describes connections supported by the PENKO device.

5.1 Class Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Revision	UINT	1	Revision of this object.
2	Get	Max instance	UINT	1	Maximum instance number of an object currently created in this class level of the device.
3	Get	Number of instance	UINT	1	Number of object instances currently created at this class level of the device.
6	Get	Max ID Number Class Attributes	UINT	7	The attribute ID number of the last class attribute of the class definition implemented in the device.
7	Get	Maximum ID Number Instance Attributes	UINT	0	The attribute ID number of the last instance attribute of the class definition implemented in the device.

5.2 Class Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read class attribute 1,2,3,6,7
14	Get Attribute single	-	Read selected class attribute

ⓘ **Instance attributes and services are not applicable for this class**

6 Class 0xF5 (245) - TCP/IP

The TCP/IP Interface object provides a mechanism for configuring a device's TCP/IP network interface. Examples of configurable items include the device's IP address, network mask and gateway address.

6.1 Class Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Revision	UINT	1	Revision of this object.
2	Get	Max instance	UINT	1	Maximum instance number of an object currently created in this class level of the device.
3	Get	Number of instance	UINT	1	Number of object instances currently created at this class level of the device.
6	Get	Max ID Number Class Attributes	UINT	7	The attribute ID number of the last class attribute of the class definition implemented in the device.
7	Get	Maximum ID Number Instance Attributes	UINT	6	The attribute ID number of the last instance attribute of the class definition implemented in the device.

6.2 Class Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read class attribute 1,2,3,6,7
14	Get Attribute single	-	Read selected class attribute

6.3 Instance Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Status	DWORD		Interface status, see Vol21.11 5-3.3.2.1
2	Get	Configuration Capability	DWORD		Interface capability flags, , see Vol21.11 5-3.3.2.2
3	Get	Configuration Control	DWORD		Interface control flags, , see Vol21.11 5-3.3.2.3.1
4	Get	Physical Link Object	STRUCT of UINT EPATH		Physical link object, see also Vol21.11 Path Size Path
5	Get	Interface configuration IP Address Network Gateway Address Name Server Name Server2 Domain Name	STRUCT OF UDINT UDINT UDINT UDINT UDINT STRING		
6	Get	Host Name	STRING		

6.4 Instance Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read class attribute 1,2,3,6,7
14	Get Attribute single	-	Read selected class attribute

7 Class 0x300 (768) - Weigher

The weigher class is a custom PENKO class used to read/write weigher data.

7.1 Class Attributes

Number	Access	Name	Data type	Value	Description
1	Get	Revision	UINT	2	Revision of this object.
2	Get	Max instance	UINT	1	Maximum instance number of an object currently created in this class level of the device.
3	Get	Number of instance	UINT	1	Number of object instances currently created at this class level of the device.
6	Get	Max ID Number Class Attributes	UINT	7	The attribute ID number of the last class attribute of the class definition implemented in the device.
7	Get	Maximum ID Number Instance Attributes	UINT	18	The attribute ID number of the last instance attribute of the class definition implemented in the device.

7.2 Class Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read class attribute 1,2,6,7
14	Get Attribute single	-	Read selected class attribute

7.3 Instance Attributes

Number	Access	Name	Data type	Value	Description
1	Get	WEIGHER	DINT		Display rate weigher data including multi range/interval step size
2	Get	FAST GROSS	DINT		Unfiltered weigher data gross
3	Get	FAST NET	DINT		Unfiltered weigher data net
4	Get	GROSS	DINT		Display rate weigher data gross
5	Get	NET	DINT		Display rate weigher data net
6	Get	TARE	DINT		Active tare weight
7	Get	PEAK	DINT		Measured peak weight since last reset
8	Get	VALLEY	DINT		Measured valley weight since last reset
9	Get	WEIGHER x10	DINT		Display rate weigher data x10
10	Get	FAST GROSS x10	DINT		Unfiltered weigher data gross x10
11	Get	FAST NET x10	DINT		Unfiltered weigher data net x10
12	Get	GROSS x10	DINT		Display rate weigher data gross x10
13	Get	NET x10	DINT		Display rate weigher data net x10
14	Get	TARE x10	DINT		Active tare weight x10
15	Get	PEAK x10	DINT		Measured peak weight since last reset x10
16	Get	VALLEY x10	DINT		Measured valley weight since last reset x10
17	Get	SAMPLE	DINT		Internal resolution
18	Get	STATUS	WORD		Weigher status bits Instance Attributes Status

7.4 Instance Attributes Status

Bit #	Called	Definition
0	OVERLOAD	Hardware overload/underload detected on loadcell
1	MAXLOAD	Overload detected on loadcell
2	STABLE	Weigher signal is stable
3	STABLE RANGE	Weigher signal is in stable range
4	ZERO SET	Weigher zero is corrected
5	ZERO CENTER	Weigher in center of zero range
6	ZERO RANGE	Weigher is in zero range, zero is possible
7	ZERO TRACK	Weigher signal is in zero tracking range, zero tracking is possible
8	TARE	Weigher tare is active
9	PTARE	Weigher preset tare is active
10	SAMPLE	Used by internal process handling
11	BAD CAL	Calibration is bad, invalid, not available
12	CAL ENABLED	Calibration is enabled, used by internal process handling
13	INDUSTRIAL	If set weigher runs in industrial mode, if reset weigher runs certified operation mode
14	NOT LEVEL	Weigher system in blocking, warming up or scale is not level
15	RESERVED	Reserved always 0

7.5 Instance Services

Service code	Name	Parameters	Description
1	Get Attribute all	-	Read attributes 1-18
14	Get Attribute single	-	Read selected attribute
50	Zero Set	-	Weigher zero function
51	Zero Reset	-	Weigher zero reset function
52	Tare On	-	Weigher tare on function
53	Tare Off	-	Weigher tare off function
54	Tare Toggle	-	Weigher tare toggle function
55	Preset Tare	Tare weight DINT	Weigher preset tare function
56	Hold Set	-	Weigher hold set function, Not available in revision 1.1
57	Peak Reset	-	Weigher peak reset function Not available in revision 1.1
58	Valley Reset	-	Weigher valley reset function Not available in revision 1.1
64	Calibrate Zero	Security code= 0x0055AAFF	Actual weight is 0
65	Calibrate Span	Security code= 0x0055AAFF Weight DINT	Calibrate Span Actual weight on scale
66	Calibrate mV	Security code= 0xFFAA5500 mV DINT Max.loadcell	Removed the deadload, the span is unattended fixed point 0.00000mV Maximum weigh performance loadcell
67	Calibrate Deadload	Security code= 0xFFAA5500 Weight DINT	Measure the deadload and correct, the span is unattended. Correction weight if scale is not empty
80	Register Functions	DINT in[4] DINT out[4]	See Register Functions In parameters Output parameters

See examples chapter

8 Register functions

By using register functions, all device parameters can be read and/or written.

In other PENKO protocols these functions use the device registers to send and receive data. With EtherNet/IP, complete blocks of data can be sent, making the use of the device registers unnecessary.

Register functions are found in the Weigher class, class 0x0300 (768), instance service 80.

The following parameters are used:

Parameter	Data type
Input - parameters	DINT [4]
Output - result	DINT [4]

Input:

Parameter	Data type	Description
1	DINT	Low 2 bytes = function code | High 2 bytes = 0
2	DINT	Input parameter, depending on function code
3	DINT	Input parameter, depending on function code
4	DINT	Input parameter, depending on function code

Output:

Result	Data type	Description
1	DINT	Low 2 bytes = function code | High 2 bytes = error code
2	DINT	Result, depending on function code
3	DINT	Result, depending on function code
4	DINT	Result, depending on function code

8.1 Function codes

The following function codes are present:

Name	Code	Description
NOP	0	No Operation
CAL_ZERO	1	Calibrate zero by weight
CAL_SPAN	2	Calibrate span by weight
CAL_MV	3	Calibrate in mV/V
CAL_DEADLOAD	4	Calibrate dead load by measuring weight
CAL_INSERT	5	Calibrate multipoint insert by measuring weight
CAL_POINT	6	Calibrate multipoint read point at parameter index
CAL_DELETE	7	Calibrate multipoint delete point at parameter index
CAL_GEOGRAPHIC_ORIGIN_SET	8	Calibrate set geographic origin calibration
CAL_GEOGRAPHIC_ORIGIN_GET	9	Calibrate get geographic origin calibration
CAL_GEOGRAPHIC_LOCAL_SET	10	Calibrate set geographic local calibration
CAL_GEOGRAPHIC_LOCAL_GET	11	Calibrate get geographic local calibration
IND_MAXLOAD_SET	101	Indicator set maximum load
IND_MAXLOAD_GET	102	Indicator get maximum load
PDI_PATH_SET	201	PDI path set
PDI_PROPERTY_SET	202	PDI property set
PDI_PROPERTY_GET	203	PDI property get
PRINT	301	Print ticket or line to printer
PRINT_SUBTOTAL	302	Subtotals to printer not supported by SGM series
PRINT_TOTAL	303	Totals to printer
PRINT_DAYTOTAL	304	Day totals to printer not supported by SGM series
PRINT_BATCHTOTAL	305	Batch totals to printer not supported by SGM series
PRINT_LAYOUT	306	Custom total layout to printer not supported by SGM series and 1020
PRINT_ALIBI	307	Print to Alibi memory not supported by SGM700 series
PRINT_ALIBIMEMORY	308	print full alibi memory to printer not supported by SGM700 series
PRINT_EVENTMEMORY	309	print full event memory to printer not supported by SGM700 series
TOTAL_TOTALIZE	401	Totalize actual stable weight
TOTAL_SUBTOTAL	402	Read or reset actual subtotal
TOTAL_TOTAL	403	Read or reset actual totals
TOTAL_DAYTOTAL	404	Read or reset actual day totals
TOTAL_BATCHTOTAL	405	Read or reset actual batch totals

RFN_PROCESS_RECIPE_GET	501	Read MFL/CHK/BLT recipe
RFN_PROCESS_RECIPE_SET	502	Write MFL/CHK/BLT recipe
RFN_PROCESS_CONFIG_GET	601	Read MFL/CHK/BLT configuration
RFN_PROCESS_CONFIG_SET	602	Write MFL/CHK/BLT configuration
RFN_PROCESS_DATA	701	Read MFL/CHK/BLT process data

8.2 Error codes

The following error codes are present:

Name	Code	Description
SUCCES	**0**	**Function successful**
WRN_WARNING	**1000**	**System warnings:**
WRN_TIMEOUT	1001	Generic time-out warning
WRN_TOLOW	1002	Generic parameter to low warning
WRN_TOHIGH	1003	Generic parameter to high warning
WRN_ZERO	1004	Generic parameter/result is zero warning
WRN_NOTZERO	1005	Generic parameter/result is not zero warning
WRN_POSITIVE	1006	Generic parameter is positive warning
WRN_NEGATIVE	1007	Generic parameter is negative warning
WRN_FULL	1008	Generic something is full warning
WRN_EMPTY	1009	Generic something is empty warning
WRN_NOTFOUND	1010	Generic search not found warning
WER_WARNING	**1100**	**Weigher warnings:**
WER_NO_TARE	1101	Zero tare level, tare rst
ERR_ERROR	**2000**	**System errors:**
ERR_PARAMETER_INCORRECT	2001	Generic parameter error
ERR_TIMEOUT	2002	Generic time-out error
ERR_TOLOW	2003	Generic parameter to low error
ERR_TOHIGH	2004	Generic parameter to high error
ERR_ZERO	2005	Generic parameter/result is zero error
ERR_NOTZERO	2006	Generic parameter/result is not zero error
ERR_POSITIVE	2007	Generic parameter is positive error
ERR_NEGATIVE	2008	Generic parameter is negative error
ERR_FULL	2009	Generic something is full error
ERR_EMPTY	2010	Generic something is empty error
ERR_NOTFOUND	2011	Generic search not found error
ERR_FILE_NOT_FOUND	2012	Generic file not found error
WER_ERROR	**2100**	**Weigher errors:**
WER_NOT_STABLE	2101	Weigher not stable

WER_ABOVE_MAXLOAD	2102	Parameter above max load
WER_BELOW_ZERO	2103	Parameter below zero
WER_NOT_IN_ZERO_RANGE	2104	Not in zero range
WER_ARITHMIC_OVERFLOW	2105	Arrhythmic overflow occurred
WER_ADC_OVERFLOW	2106	A/D reads all 1's
WER_ADC_UNDERFLOW	2107	A/D reads all 0's
WER_GAIN_NEGATIVE	2108	Gain ref. < zero ref.
WER_GAIN_OVERFLOW	2109	Gain limit
WER_SAVE	2110	Save errors:
WER_SAVE_FLASH_EXHAUSTED	2111	Flash ROM exhausted
WER_SAVE_CREATE_HEADER	2112	Error on header creation
WER_SAVE_DATA_WRITE	2113	Error on data write
WER_SAVE_HEADER_VALIDATE	2114	Header validation failed
WER_SAVE_DEACTIVATE	2115	Deactivate old data fail
WER_LOAD	2116	Load errors
WER_LOAD_NOT_FOUND	2117	Item not found in store
WER_LOAD_DATA_ERROR	2118	Error in stored data
WER_BAD_CALIBRATION	2119	No calibration available
WER_NOT_ENABLED	2120	Action not enabled
WER_MCAL_NOT_FOUND	2121	Multi-point not found
WER_MCAL_OVERFLOW	2122	Calibration table full
WER_TARE_ACTIVE	2123	Not allowed, tare active
WER_NOT_ALLOWED	2124	Action is not allowed
WER_ADC_NOPOWER	2125	ADC has no power
ERR_DOSER	**2200**	**Doser errors**
ERR_POSITION	**2300**	**Position errors**
ERR_SPCAPP	**2400**	**SPC-application errors**
ERR_SCOPE	**2500**	**Scope errors**
ERR_INTERPRETER	**2600**	**Interpreter errors**
ERR_USB	**3000**	**USB errors - use USB routines for returning error texts**
ERR_FLASH	**3100**	**FLASH file system errors**

8.3 Calibration functions

This chapter describes the calibration function codes.

8.3.1 CAL_ZERO

Calibrate zero by weight. Function code = 1.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
1	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
1	Not used	Not used	Not used

8.3.2 CAL_SPAN

Calibrate span by weight. Function code = 2.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
2	Span weight	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
2	Not used	Not used	Not used

Example - calibrate span on 1.200kg:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
2	1200	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
2	Not used	Not used	Not used

Example - calibrate span without loading scale - will result in **error**:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
2	1200	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
138215426	Not used	Not used	Not used

The result is **138215426**

- The function code (low word) is **2**
- The error code (high word) is **2109 - WER_GAIN_OVERFLOW, Gain limit**

8.3.3 CAL_MV

Theoretic calibration by millivolts. Function code = 3.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
3	Fixed point mV/V value	Maximum weight at mV/V	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
3	Not used	Not used	Not used

Example - theoretic calibration of load cell 200kg @ 2.0012mV/V:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
3	20012	200	Not used

8.3.4 CAL_DEADLOAD

Calibrate dead load by measuring weight. Function code = 4.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
4	Actual weight on scale	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
4	Not used	Not used	Not used

Example - calibration of dead load with 12kg on the scale:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
4	12	Not used	Not used

8.3.5 CAL_INSERT

Multipoint calibration up to 10 points. Insert or replace a calibration point. Function code = 5.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
5	Actual weight on scale	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
5	Not used	Not used	Not used

Example - add calibration point of 10.000kg - if the point already exists, its ADC value is replaced:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
5	10000	Not used	Not used

8.3.6 CAL_POINT

Multipoint calibration up to 10 points. Read the calibration point at index (1...10). Function code = 6.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
6	Index (1...10)	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
6	Index	Calibration reference weight	Calibration in mV

Example - read calibration point 1 (10.000kg @ 9.9975mV/V):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
6	1	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
6	1	10000	9.9975

8.3.7 CAL_DELETE

Multipoint calibration up to 10 points. Delete the calibration point at index (1...10). Function code = 7.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
7	Index (1...10)	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
7	Index	Not used	Not used

Example - delete calibration point 1:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
7	1	Not used	Not used

8.3.8 CAL_GEOGRAPHIC_ORIGIN_SET

Geographic correction. Set the origin calibration location. Function code = 8.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
8	Fixed point latitude degrees	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
8	Not used	Not used	Not used

Example - set origin latitude to 50.00 degrees:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
8	5000	Not used	Not used

8.3.9 CAL_GEOGRAPHIC_ORIGIN_GET

Geographic correction. Get the origin calibration location. Function code = 9.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
9	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
9	Fixed point latitude degrees	Not used	Not used

Example - get origin latitude (50.00 degrees):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
9	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
9	5000	Not used	Not used

8.3.10 CAL_GEOGRAPHIC_LOCAL_SET

Geographic correction. Set the actual scale location. Function code = 10.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
10	Fixed point latitude degrees	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
10	Not used	Not used	Not used

Example - set actual latitude to 50.00 degrees:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
10	5000	Not used	Not used

8.3.11 CAL_GEOGRAPHIC_LOCAL_GET

Geographic correction. Get the actual scale location. Function code = 11.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
11	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
11	Fixed point latitude degrees	Not used	Not used

Example - get location latitude (50.00 degrees):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
11	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
11	5000	Not used	Not used

8.4 Indicator functions

This chapter describes the indicator function codes.

8.4.1 IND_MAXLOAD_SET

Set the indicator maximum load. Function code = 101.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
101	Max load	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
101	Not used	Not used	Not used

Example - set the maximum load to 10.020kg:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
101	10020	Not used	Not used

8.4.2 IND_MAXLOAD_GET

Get the indicator maximum load. Function code = 102.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
102	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
102	Max load	Not used	Not used

Example - get the maximum load (10.020kg):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
102	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
102	10020	Not used	Not used

8.5 PDI functions

This chapter describes the PDO function codes.

8.5.1 PDI_PATH_SET

Set the PDI path to perform the action on. Function code = 201.

Request:

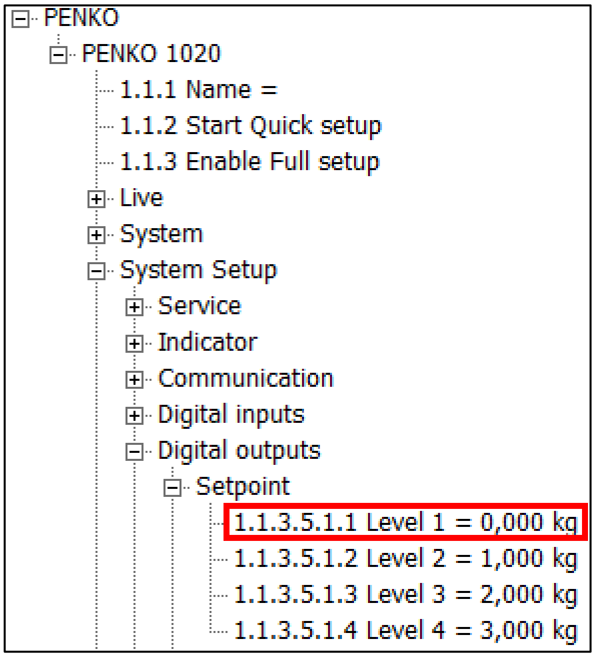
Parameter 1	Parameter 2	Parameter 3	Parameter 4
201	Path no. 1,2,3,4	Path no. 5,6,7,8	Path no. 9,10,11,12

Reply:

Result 1	Result 2	Result 3	Result 4
201	Path no. 1,2,3,4	Path no. 5,6,7,8	Path no. 9,10,11,12

PDI (PENKO Device Interface) represents the device configuration in a tree structure. Every property has its own unique path number. The tree is used in the PENKO configuration tools Pi Mach II and PDI Client, both available at www.penko.com/software

For example, a part of the PENKO 1020 looks like this:



```
⊟ PENKO
   ⊟ PENKO 1020
        1.1.1 Name =
        1.1.2 Start Quick setup
        1.1.3 Enable Full setup
      ⊞ Live
      ⊞ System
      ⊟ System Setup
         ⊞ Service
         ⊞ Indicator
         ⊞ Communication
         ⊞ Digital inputs
         ⊟ Digital outputs
            ⊟ Setpoint
                 1.1.3.5.1.1 Level 1 = 0,000 kg
                 1.1.3.5.1.2 Level 2 = 1,000 kg
                 1.1.3.5.1.3 Level 3 = 2,000 kg
                 1.1.3.5.1.4 Level 4 = 3,000 kg
```

Setpoint 1 has path number 1.1.3.5.1.1. This results in the following input parameters:

Fill the path with zeros to make 12 numbers: 1.1.3.5.1.1.0.0.0.0.0.0

Parameter 1	PDI_PATH_SET	201
Parameter 2	0x01 0x01 0x03 0x05	16843525
Parameter 3	0x01 0x01 0x00 0x00	16842752
Parameter 4	0x00 0x00 0x00 0x00	0

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
201	16843525	16842752	0

Reply:

Result 1	Result 2	Result 3	Result 4
201	16843525	16842752	0

The PDI path is now set.

If the path is not found, all zeros are returned.

8.5.2 PDI_PROPERTY_SET

Set a PDI property for the selected PDI path. Function code = 202.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
202	Property value	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
202	Not used	Not used	Not used

Example - set setpoint 1 to 0.500kg (path must be selected with PDI_PATH_SET):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
202	500	Not used	Not used

8.5.3 PDI_PROPERTY_GET

Get a PDI property from the selected PDI path. Function code = 203.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
203	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
203	Property value integer/string	Property value string optional	Property value string optional

Example - get setpoint 1 (0.500kg) (path must be selected with PDI_PATH_SET):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
203	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
203	500	Not used	Not used

Example - get the software version number (1.4.3.9.0.1) (path must be selected with PDI_PATH_SET):

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
203	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
203	825111598	858667310	808333568

The result is a string containing the version number:

Result 2	825111598	0x31 0x2E 0x34 0x2E	1.4.
Result 3	858667310	0x33 0x2E 0x39 0x2E	3.9.
Result 4	808333568	0x30 0x2E 0x31 0x00	0.1

1.4.3.9.0.1

8.6 Printer functions

This chapter describes the printer function codes.

8.6.1 PRINT

Print ticket or line layout, depending on set layout in device. Function code = 301.

For the SGM series printing is only available at the SGM720/820 and SGM750/850.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
301	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
301	Gross weight	Net weight	Tare weight

Example - print:

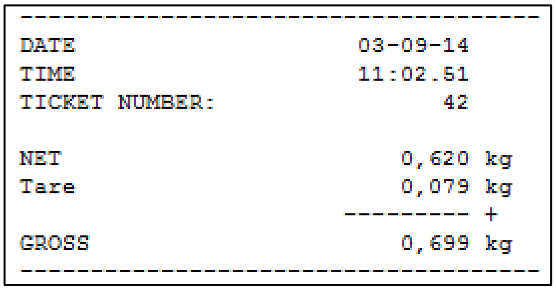
Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
301	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
301	699	620	79

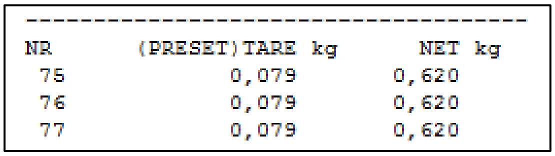
Device printer layout setting: **ticket**



```
-------------------------------------
DATE                        03-09-14
TIME                        11:02.51
TICKET NUMBER:                    42

NET                          0,620 kg
Tare                         0,079 kg
                       ---------- +
GROSS                        0,699 kg
-------------------------------------
```

Device printer layout setting: **line**



```
------------------------------------
NR       (PRESET)TARE kg      NET kg
 75             0,079      0,620
 76             0,079      0,620
 77             0,079      0,620
```

8.6.2 PRINT_SUBTOTAL

Print subtotal to printer. Function code = 302.

Not applicable for the SGM7xx/8xx series.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
302	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
302	Subtotal gross weight	Subtotal net weight	Subtotal tare weight

Example - print subtotal:

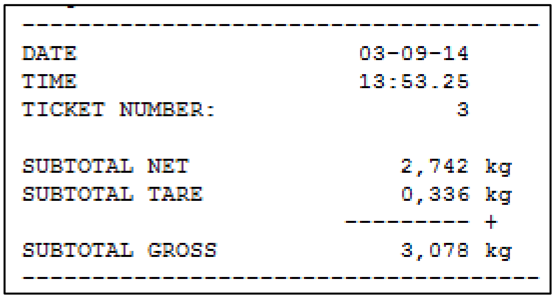
Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
302	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
302	3078	2742	336

Printed ticket:



```
-------------------------------------
DATE                     03-09-14
TIME                     13:53.25
TICKET NUMBER:                  3

SUBTOTAL NET                2,742 kg
SUBTOTAL TARE               0,336 kg
                        ---------- +
SUBTOTAL GROSS              3,078 kg
-------------------------------------
```

8.6.3 PRINT_TOTAL

Print total to printer. Function code = 303.

For the SGM series printing is only available at the SGM720/820 (Ethernet) and SGM750/850 (serial).

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
303	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
303	Total gross weight	Total net weight	Total tare weight

Example - print total:

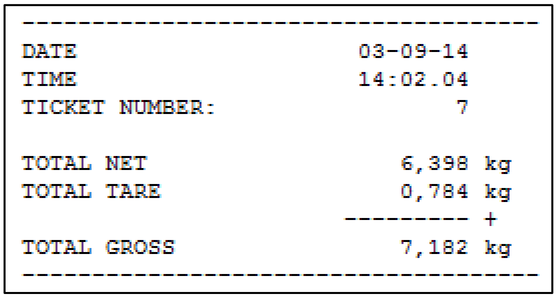
Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
303	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
303	7182	6398	784

Printed ticket:



```
---------------------------------------
DATE                        03-09-14
TIME                        14:02.04
TICKET NUMBER:                     7

TOTAL NET                      6,398 kg
TOTAL TARE                     0,784 kg
                          ---------- +
TOTAL GROSS                    7,182 kg
---------------------------------------
```

8.6.4 PRINT_DAYTOTAL

Print day total to printer. Function code = 304.

Not applicable for the SGM7xx/8xx series.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
304	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
304	Day total gross weight	Day total net weight	Day total tare weight

Example - print day total:

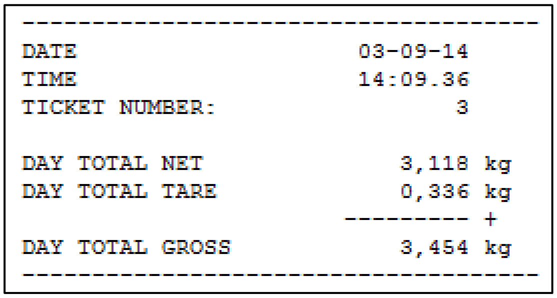
Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
304	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
304	3454	3118	336

Printed ticket:



```
---------------------------------------
DATE                         03-09-14
TIME                         14:09.36
TICKET NUMBER:                      3

DAY TOTAL NET                   3,118 kg
DAY TOTAL TARE                  0,336 kg
                           ---------- +
DAY TOTAL GROSS                 3,454 kg
---------------------------------------
```

8.6.5 PRINT_BATCHTOTAL

Print batch total to printer. Function code = 305.

Not applicable for the SGM7xx/8xx series.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
305	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
305	Batch total gross weight	Batch total net weight	Batch total tare weight

Example - print batch total:

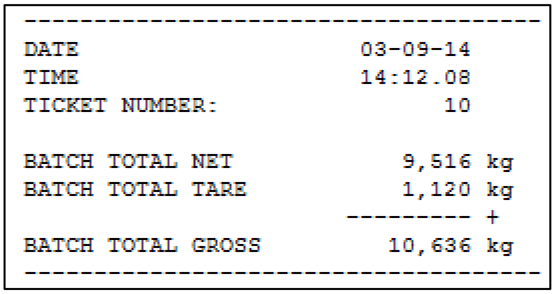
Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
305	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
305	10636	9516	1120

Printed ticket:



```
---------------------------------------
DATE                        03-09-14
TIME                        14:12.08
TICKET NUMBER:                    10

BATCH TOTAL NET                9,516 kg
BATCH TOTAL TARE               1,120 kg
                          --------- +
BATCH TOTAL GROSS             10,636 kg
---------------------------------------
```

8.6.6 PRINT_LAYOUT

Print to Printer function custom layout 1-n. Function code = 306.

Not supported by 1020 and SGM series.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
306	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
306	Layout number 1-n	Not used	Not used

Example - print with custom layout:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
306	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
306	1	Not used	Not used

8.6.7 PRINT_ALIBI

Print to Alibi memory function. Store the actual stable weight in Alibi memory. Function code = 307.

For the SGM series Alibi memory is only available at the SGM8xx series.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
307	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
307	UID*	Gross/Net	Preset (Tare)

** UID can exceed the maximum positive value of the LONG data type (signed). Use the DWORD data type (unsigned).*

Example - write to Alibi memory, with active tare:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
307	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
307	1944985600	1315	112

Corresponding Alibi records:

Entry	Code	Date/Value	Time/Unit	UID
00001/00004	Alibi 001	03-09-14	15:00:46	**1944985600**
00002/00004	Net	**1.315**	kg	3803586561
00003/00004	Tare	**0.112**	kg	1269178371
00004/00004	Gross	1.427	kg	0718544901

Example - write to Alibi memory, without active tare:

Reply:

Result 1	Result 2	Result 3	Result 4
307	1660428288	1711	0

Corresponding Alibi records:

Entry	Code	Date/Value	Time/Unit	UID
00001/00002	Alibi 001	03-09-14	15:02:34	**1660428288**
00002/00002	Gross	**1.711**	kg	1133518849

8.6.8 PRINT_ALIBIMEMORY

Print the complete Alibi memory to a printer. Function code = 308.

For the SGM series Alibi/printing is only available at the SGM820 and SGM850.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
308	Not used	Not used	Not used

Reply:

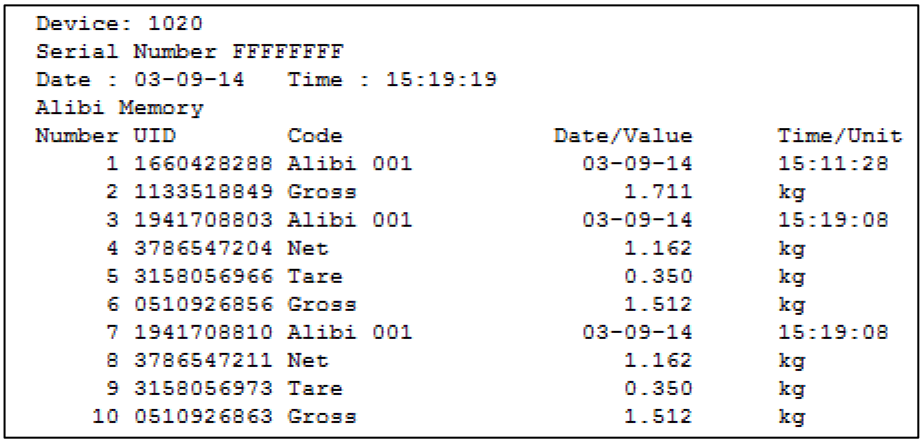
Result 1	Result 2	Result 3	Result 4
308	Not used	Not used	Not used

Example - print the complete Alibi memory:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
308	Not used	Not used	Not used

Printed ticket:



```
Device: 1020
Serial Number FFFFFFFF
Date : 03-09-14   Time : 15:19:19
Alibi Memory
Number UID        Code               Date/Value      Time/Unit
     1 1660428288 Alibi 001            03-09-14      15:11:28
     2 1133518849 Gross                   1.711      kg
     3 1941708803 Alibi 001            03-09-14      15:19:08
     4 3786547204 Net                     1.162      kg
     5 3158056966 Tare                    0.350      kg
     6 0510926856 Gross                   1.512      kg
     7 1941708810 Alibi 001            03-09-14      15:19:08
     8 3786547211 Net                     1.162      kg
     9 3158056973 Tare                    0.350      kg
    10 0510926863 Gross                   1.512      kg
```

8.6.9 PRINT_EVENTMEMORY

Print the complete Event log to a printer. Function code = 309.

For the SGM series Event/printing is only available at the SGM820 and SGM850.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
309	Not used	Not used	Not used

Reply:

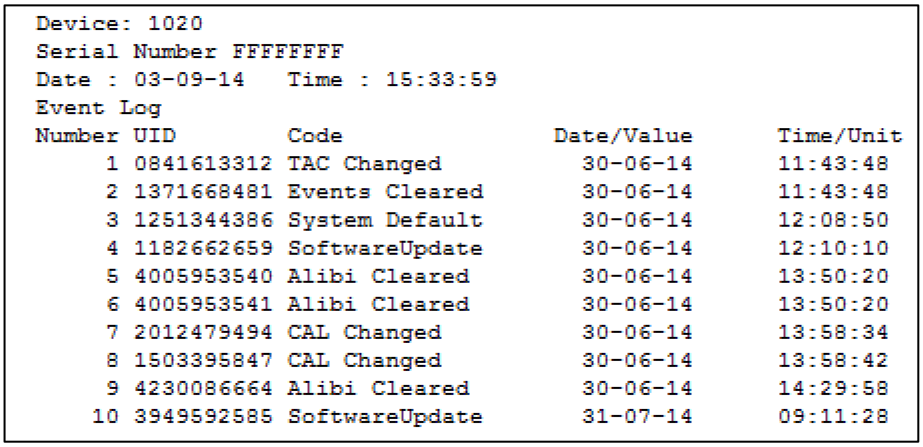
Result 1	Result 2	Result 3	Result 4
309	Not used	Not used	Not used

Example - print the complete Event log:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
309	Not used	Not used	Not used

Printer ticket:



```
Device: 1020
Serial Number FFFFFFFF
Date : 03-09-14   Time : 15:33:59
Event Log
Number UID        Code                 Date/Value      Time/Unit
     1 0841613312 TAC Changed            30-06-14      11:43:48
     2 1371668481 Events Cleared         30-06-14      11:43:48
     3 1251344386 System Default         30-06-14      12:08:50
     4 1182662659 SoftwareUpdate         30-06-14      12:10:10
     5 4005953540 Alibi Cleared          30-06-14      13:50:20
     6 4005953541 Alibi Cleared          30-06-14      13:50:20
     7 2012479494 CAL Changed            30-06-14      13:58:34
     8 1503395847 CAL Changed            30-06-14      13:58:42
     9 4230086664 Alibi Cleared          30-06-14      14:29:58
    10 3949592585 SoftwareUpdate         31-07-14      09:11:28
```

8.7 Total functions

This chapter describes the total function codes.

8.7.1 TOTAL_TOTALIZE

Totalize actual stable weight. Function code = 401.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
401	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
401	Added gross weight	Added net weight	Added tare weight

Example - add actual stable weight to total:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
401	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
401	1512	1162	350

8.7.2 TOTAL_SUBTOTAL

Get the subtotal weights. Function code = 402.

Not applicable for the SGM7xx/8xx series.

Leave parameter 2 empty to read the subtotal weights.

Set parameter 2 to **0x55AA55AA** to reset the subtotal weights.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
402	Optional	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
402	Subtotal gross weight	Subtotal net weight	Subtotal tare weight

Example - read the subtotal weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
402	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
402	12096	9296	2800

Example - reset the subtotal weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
402	1437226410	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
402	12096	9296	2800

When reading again, all subtotal weights are 0.

8.7.3 TOTAL_TOTAL

Get the total weights. Function code = 403.

Leave parameter 2 empty to read the total weights.

Set parameter 2 to **0x55AA55AA** to reset the total weights.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
403	Optional	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
403	Total gross weight	Total net weight	Total tare weight

Example - read the total weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
403	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
403	12096	9296	2800

Example - reset the total weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
403	1437226410	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
403	12096	9296	2800

When reading again, all total weights are 0.

8.7.4 TOTAL_DAYTOTAL

Get the day total weights. Function code = 404.

Not applicable for the SGM7xx/8xx series.

Leave parameter 2 empty to read the day total weights.

Set parameter 2 to **0x55AA55AA** to reset the day total weights.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
404	Optional	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
404	Day total gross weight	Day total net weight	Day total tare weight

Example - read the day total weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
404	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
404	12096	9296	2800

Example - reset the day total weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
404	1437226410	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
404	12096	9296	2800

When reading again, all day total weights are 0.

8.7.5 TOTAL_BATCHTOTAL

Get the batch total weights. Function code = 405.

Not applicable for the SGM7xx/8xx series.

Leave parameter 2 empty to read the batch total weights.

Set parameter 2 to **0x55AA55AA** to reset the batch total weights.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
405	Optional	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
405	Batch total gross weight	Batch total net weight	Batch total tare weight

Example - read the batch total weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
405	Not used	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
405	12096	9296	2800

Example - reset the batch total weights:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
405	1437226410	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
405	12096	9296	2800

When reading again, all batch total weights are 0.

8.8 Controller functions

This chapter describes the controller functions for the belt weigher, check weigher and mono filler.

8.8.1 RFN_PROCESS_RECIPE_GET

Get the value of the selected recipe parameter. Function code = 501.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
501	Recipe param	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
501	Recipe param	Value	Not used

Example - get the value of recipe parameter 1:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
501	1	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
501	1	2000	Not used

8.8.2 RFN_PROCESS_RECIPE_SET

Set the value of the selected recipe parameter. Function code = 502.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
502	Recipe param	Value	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
502	Recipe param	Not used	Not used

Example - set the value of recipe parameter 2 to 500:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
502	2	500	Not used

8.8.3 RFN_PROCESS_CONFIG_GET

Get the value of the selected configuration parameter. Function code = 601.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
601	Config param	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
601	Config param	Value	Not used

Example - get the value of configuration parameter 1:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
601	1	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
601	1	2000	Not used

8.8.4 RFN_PROCESS_CONFIG_SET

Set the value of the selected configuration parameter. Function code = 602.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
602	Config param	Value	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
602	Config param	Not used	Not used

Example - set the value of configuration parameter 2 to 500:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
602	2	500	Not used

8.8.5 RFN_PROCESS_DATA

Get the value of the selected process data parameter. Function code = 701.

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
701	Process param	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
701	Process param	Value	Not used

Example - get the value of process parameter 1:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
701	1	Not used	Not used

Reply:

Result 1	Result 2	Result 3	Result 4
701	1	2000	Not used

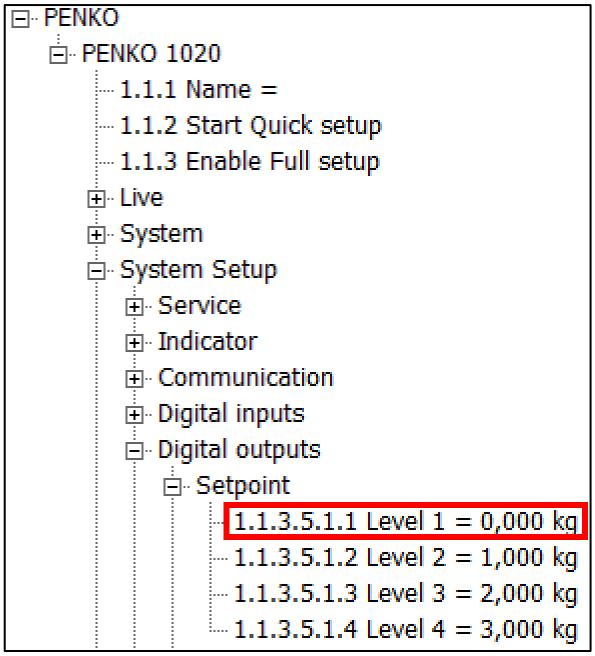
9 Examples

These examples are made with the ODVA EtherNet/IP™ Protocol Conformance Test.

9.1 Class 0x01 execute PDI

PDI (PENKO Device Interface) represents the device configuration in a tree structure. Every property has its own unique path number. The tree is used in the PENKO configuration tools Pi Mach II and PDI Client, both available at www.penko.com/software

For example, a part of the PENKO 1020 looks like this:



- ⊟ PENKO
  - ⊟ PENKO 1020
    - 1.1.1 Name =
    - 1.1.2 Start Quick setup
    - 1.1.3 Enable Full setup
    - ⊞ Live
    - ⊞ System
    - ⊟ System Setup
      - ⊞ Service
      - ⊞ Indicator
      - ⊞ Communication
      - ⊞ Digital inputs
      - ⊟ Digital outputs
        - ⊟ Setpoint
          - 1.1.3.5.1.1 Level 1 = 0,000 kg
          - 1.1.3.5.1.2 Level 2 = 1,000 kg
          - 1.1.3.5.1.3 Level 3 = 2,000 kg
          - 1.1.3.5.1.4 Level 4 = 3,000 kg

Consult the PENKO PDI documentation for more information, available at www.penko.com

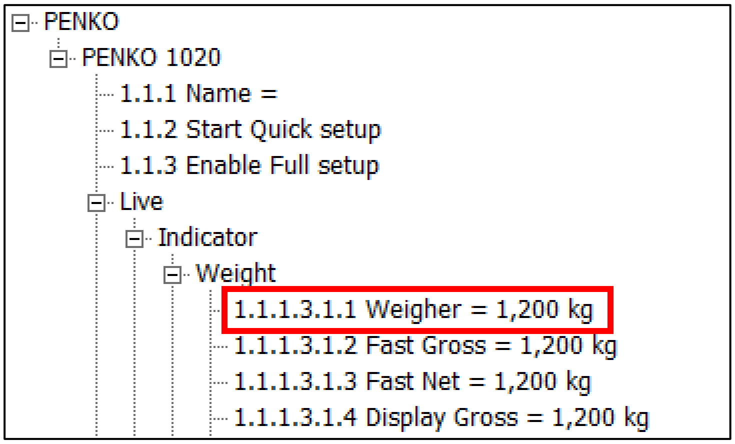
The PDI request exists out of the PDI command, a PDI function and a path. The command is a fixed value; 0xB4. The PDI functions are shown in the following table. The path indicates the parameter.

The PDI reply exists out of the full request and the result.

PDI function	Description
0x00 PDI_AVAIL	Function available
0x01 PDI_ENUMERATE	Enumerate structure
0x02 PDI_PROPERTY	Property structure
0x03 PDI_GET	Value read
0x04 PDI_SET	Value write
0x05 PDI_SETANDREPLY	Value write and get reply code

Read weigher value

The weigher value is found at PDI path 1.1.1.3.1.1



```
⊟ PENKO
  ⊟ PENKO 1020
      1.1.1 Name =
      1.1.2 Start Quick setup
      1.1.3 Enable Full setup
    ⊟ Live
      ⊟ Indicator
        ⊟ Weight
            1.1.1.3.1.1 Weigher = 1,200 kg
            1.1.1.3.1.2 Fast Gross = 1,200 kg
            1.1.1.3.1.3 Fast Net = 1,200 kg
            1.1.1.3.1.4 Display Gross = 1,200 kg
```

The used PDI function is PDI_GET

Request:

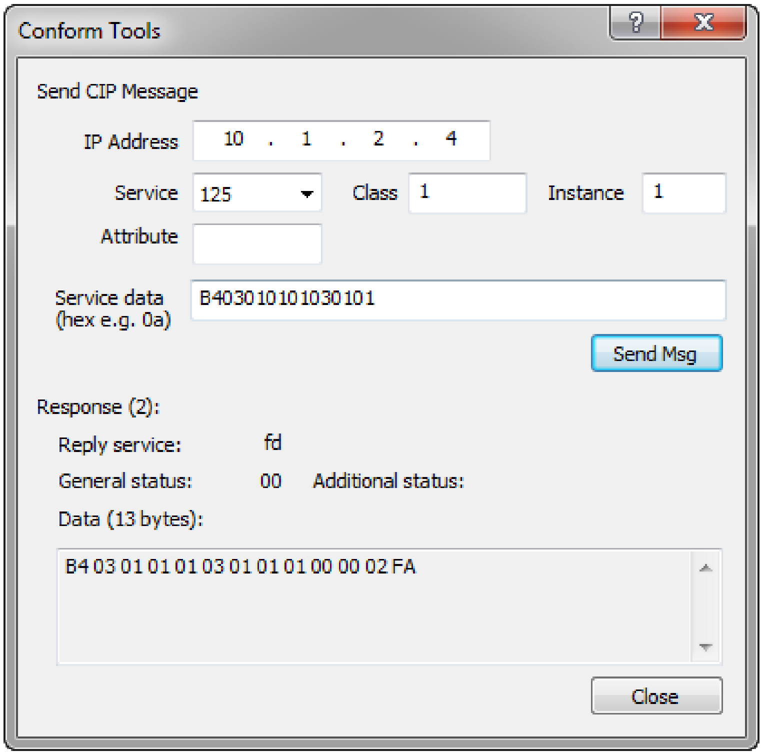
Command	Function	Path
0xB4	0x03	0x01 01 01 03 01 01

Reply:

Command	Function	Path	Status	Property value
0xB4	0x03	0x01 01 01 03 01 01	0x01	0x00 00 02 FA

- Status 0x01 is OK
- The weight is 0x00 00 02 FA → 762

With test tool:



IP Address: 10 . 1 . 2 . 4

Service: 125 Class: 1 Instance: 1

Attribute:

Service data (hex e.g. 0a): B403010101030101

Response (2):

Reply service: fd

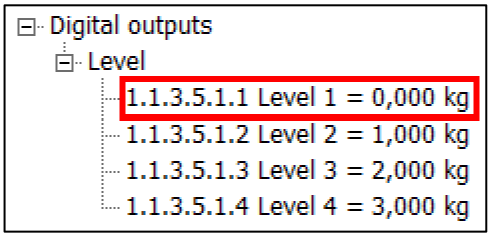
General status: 00 Additional status:

Data (13 bytes):

B4 03 01 01 01 03 01 01 01 00 00 02 FA

Write setpoint 1

Setpoint 1 is found at PDI path 1.1.3.5.1.1



```
⊟ Digital outputs
   ⊟ Level
      1.1.3.5.1.1 Level 1 = 0,000 kg
      1.1.3.5.1.2 Level 2 = 1,000 kg
      1.1.3.5.1.3 Level 3 = 2,000 kg
      1.1.3.5.1.4 Level 4 = 3,000 kg
```

The used PDI function is PDI_SET

A null terminator is added to separate the path and the value.

To set this setpoint to 300, the following request is sent:

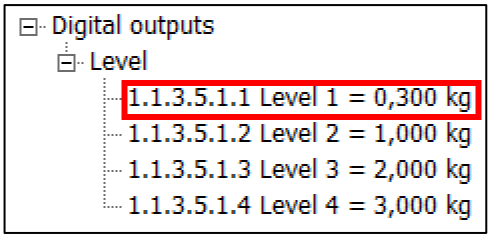
Command	Function	Path	Terminator	Property value
0xB4	0x04	0x01 01 03 05 01 01	0x00	0x00 00 01 2C

The reply will be as follows:

Command	Function	Path	Terminator	Property value	Save OK
0xB4	0x04	0x01 01 03 05 01 01	0x00	0x00 00 01 2C	0x01

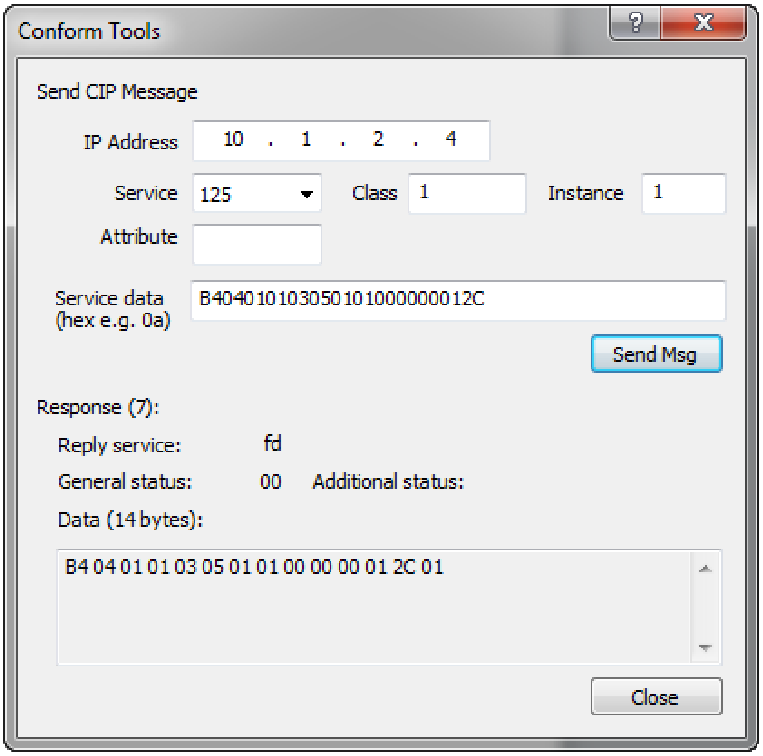
- The set value is 0x00 00 01 2C → decimal 300
- Save OK 0x01 is succeeded

Changed value:



⊟ Digital outputs
⊟ Level
1.1.3.5.1.1 Level 1 = 0,300 kg
1.1.3.5.1.2 Level 2 = 1,000 kg
1.1.3.5.1.3 Level 3 = 2,000 kg
1.1.3.5.1.4 Level 4 = 3,000 kg

With test tool:



Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 125 Class 1 Instance 1

Attribute

Service data (hex e.g. 0a) B404010103050101000000012C

Response (7):

Reply service: fd

General status: 00 Additional status:

Data (14 bytes):

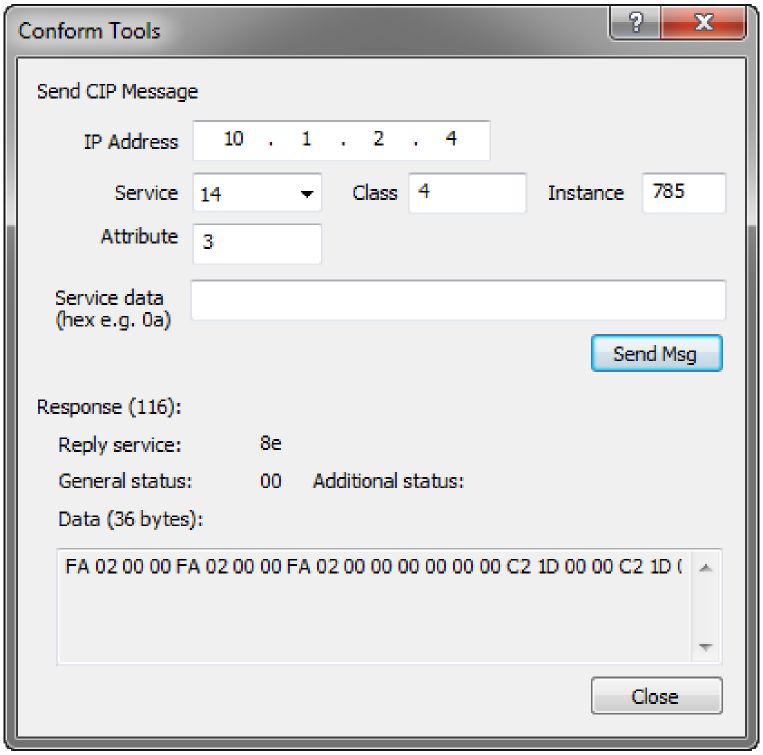
B4 04 01 01 03 05 01 01 00 00 00 01 2C 01

9.2 Class 0x04 read/write weigher data

The assembly class contains several instances to read or write weigher data. The used attribute number for all instances is 3.

Get weigher data

Instance 785 holds a list of weigher data.



Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 14 Class 4 Instance 785

Attribute 3

Service data (hex e.g. 0a)

Response (116):

Reply service: 8e

General status: 00 Additional status:

Data (36 bytes):

```
FA 02 00 00 FA 02 00 00 FA 02 00 00 00 00 00 00 C2 1D 00 00 C2 1D (
```

The following data is returned. Mind the byte order within every DINT and WORD.

Data	Data type	Value HEX	Value
WEIGHER	DINT	00 00 02 FA	762
GROSS	DINT	00 00 02 FA	762
NET	DINT	00 00 02 FA	762
TARE	DINT	00 00 00 00	0
WEIGHERx10	DINT	00 00 1D C2	7618
GROSSx10	DINT	00 00 1D C2	7618
NETx10	DINT	00 00 1D C2	7618
TAREx10	DINT	00 00 00 00	0
FORMAT	WORD	C0 03	Bit# 0, 1, 14 and 15*
STATUS	WORD	20 CC	Bit# 2, 3, 6, 7 and 13**

** Decimal point position = 3, display step-size = 1, zero-suppressing = true, signed = true*

*** Stable = true, stable range = true, zero range = true, zero track = true, industrial = true*

Control weigher

Instance 872 controls the device.

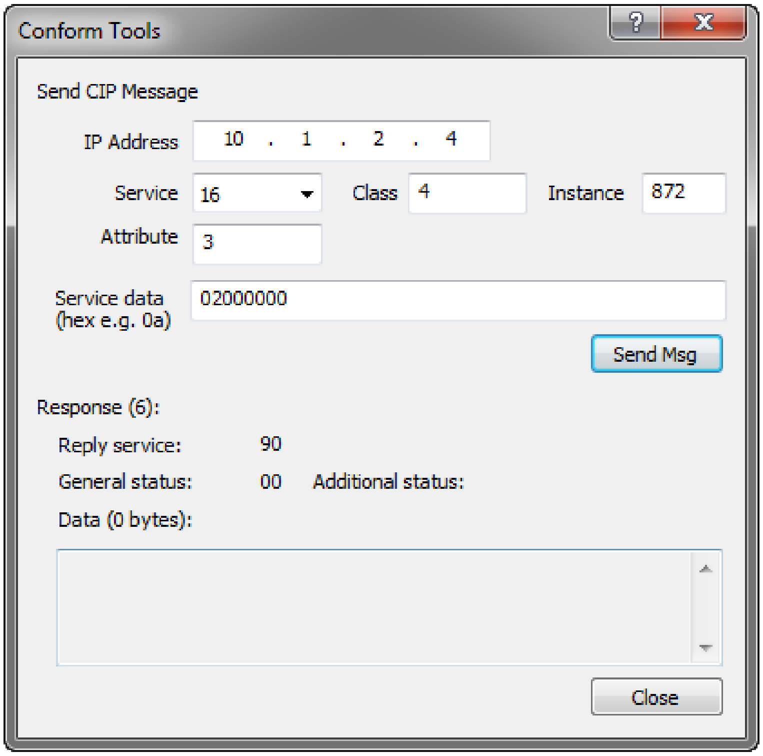
Request:

Control WORD	Reserved WORD
0x00 00*	0x00 00

** Mind the byte order within the WORD*

Set zero is control bit# 1, 0x00 02

Control WORD	Reserved WORD
0x02 00	0x00 00



Conform Tools

Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 16 Class 4 Instance 872

Attribute 3

Service data (hex e.g. 0a) 02000000

Response (6):

Reply service: 90

General status: 00 Additional status:

Data (0 bytes):

Reset zero is control bit# 0, 0x00 01

Control WORD	Reserved WORD
0x01 00	0x00 00

Set tare is control bit# 3, 0x00 08

Control WORD	Reserved WORD
0x08 00	0x00 00

Reset tare is control bit# 2, 0x00 04

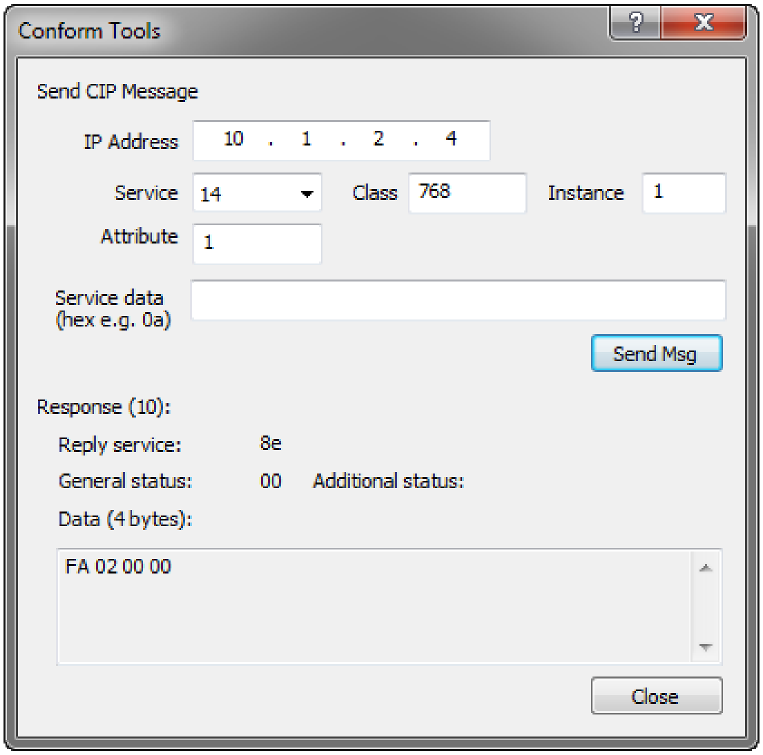
Control WORD	Reserved WORD
0x04 00	0x00 00

9.3 Class 0x300 read/write weigher data

The weigher class is a custom PENKO class used to read/write weigher data.

Get weigher value

Instance attribute 1 holds the weigher value. Instance service 14 reads a single instance attribute.



Conform Tools

Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 14 Class 768 Instance 1

Attribute 1

Service data (hex e.g. 0a)

Response (10):

Reply service: 8e

General status: 00 Additional status:

Data (4 bytes):

FA 02 00 00

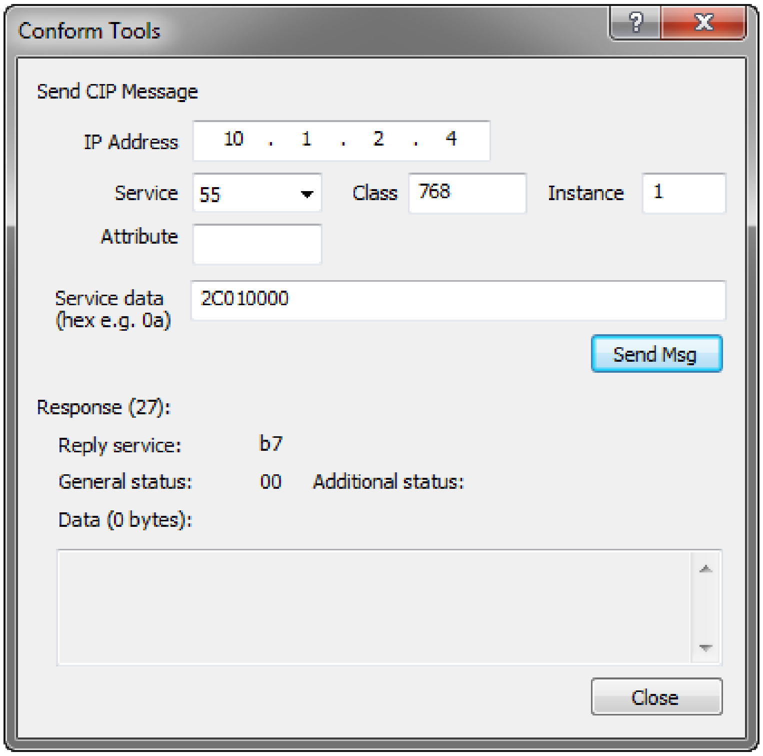
The reply 0x00 00 02 FA → 762

Set the preset tare

Instance service 55 sets the preset tare value.

Enter the value as parameter. Data type is DINT.

Set the preset tare to 300 → 00 00 01 2C



IP Address 10 . 1 . 2 . 4

Service 55 Class 768 Instance 1

Attribute

Service data (hex e.g. 0a) 2C010000

Response (27):

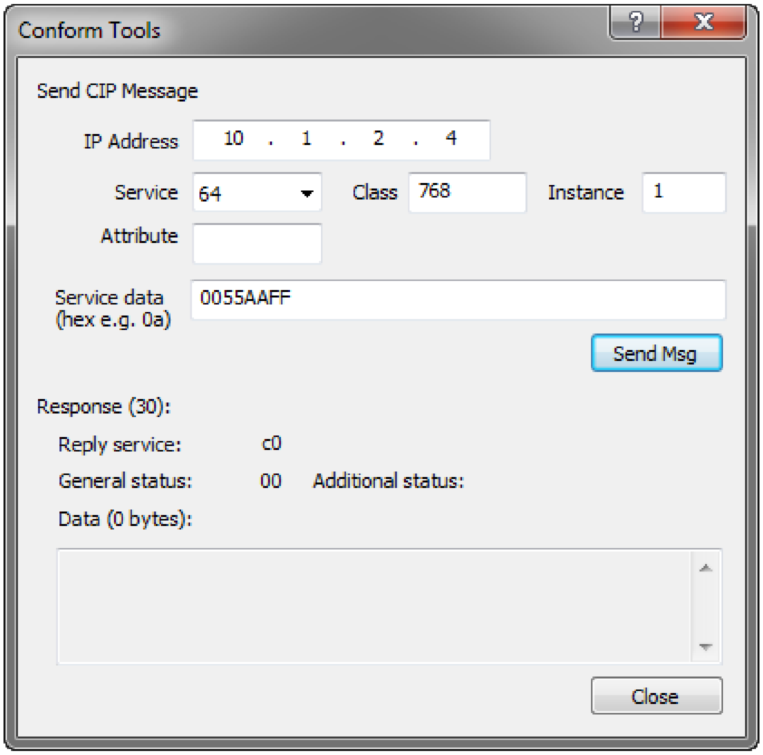
Reply service: b7

General status: 00 Additional status:

Data (0 bytes):

Calibrate zero

Instance service 64 calibrates the zero point. For calibration actions a security code is needed. This is FFAA5500.



Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 64 Class 768 Instance 1

Attribute

Service data (hex e.g. 0a) 0055AAFF

Response (30):

Reply service: c0

General status: 00 Additional status:

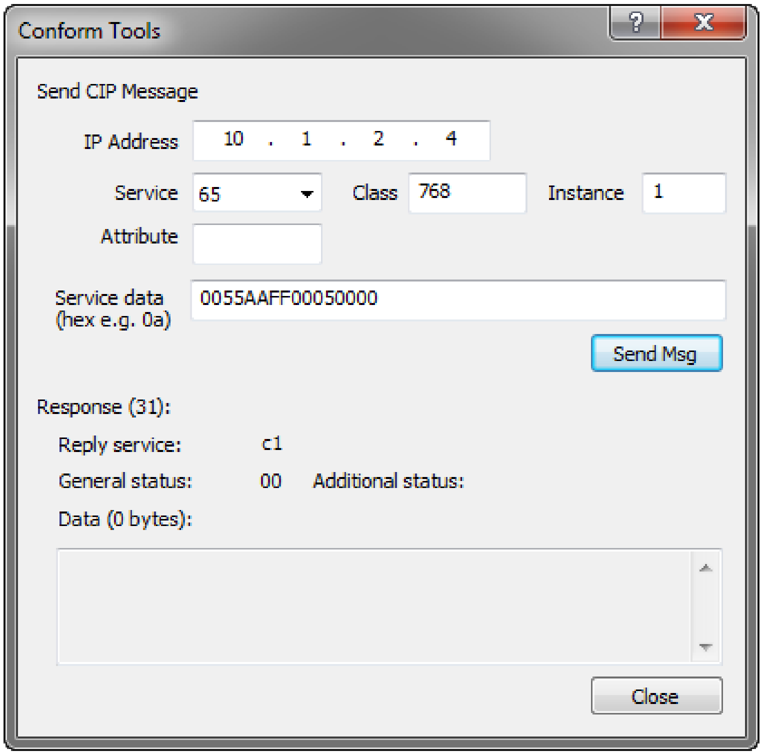
Data (0 bytes):

Calibrate span

Instance service 65 calibrates the span point. For calibration actions a security code is needed. This is FFAA5500.

Place the span weight after the security code.

Calibrate span at 1280 → 00 00 05 00



IP Address 10 . 1 . 2 . 4

Service 65 Class 768 Instance 1

Attribute

Service data (hex e.g. 0a) 0055AAFF00050000

Response (31):

Reply service: c1

General status: 00 Additional status:

Data (0 bytes):

Use register function

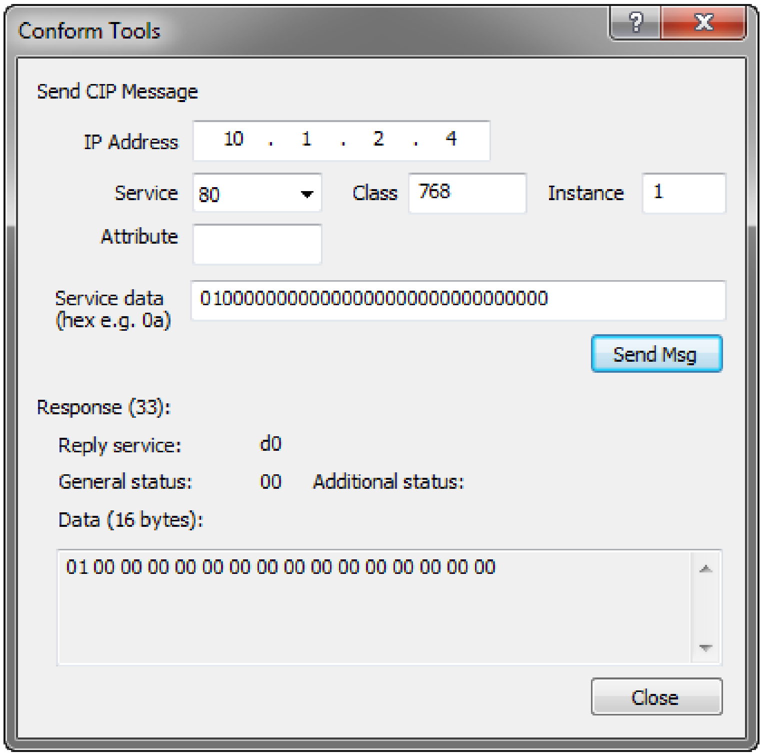
Instance service 80 executes the register functions. Enter the input values as parameter.

Each input value has to be entered as DINT[4]. Not used parameter has to be entered as 0.

Calibrating the zero point is function code 1.The input values are:

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
0x00 00 00 01	0x00 00 00 00	0x00 00 00 00	0x00 00 00 00



Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 80 Class 768 Instance 1

Attribute

Service data (hex e.g. 0a) 01000000000000000000000000000000

Response (33):

Reply service: d0

General status: 00 Additional status:

Data (16 bytes):

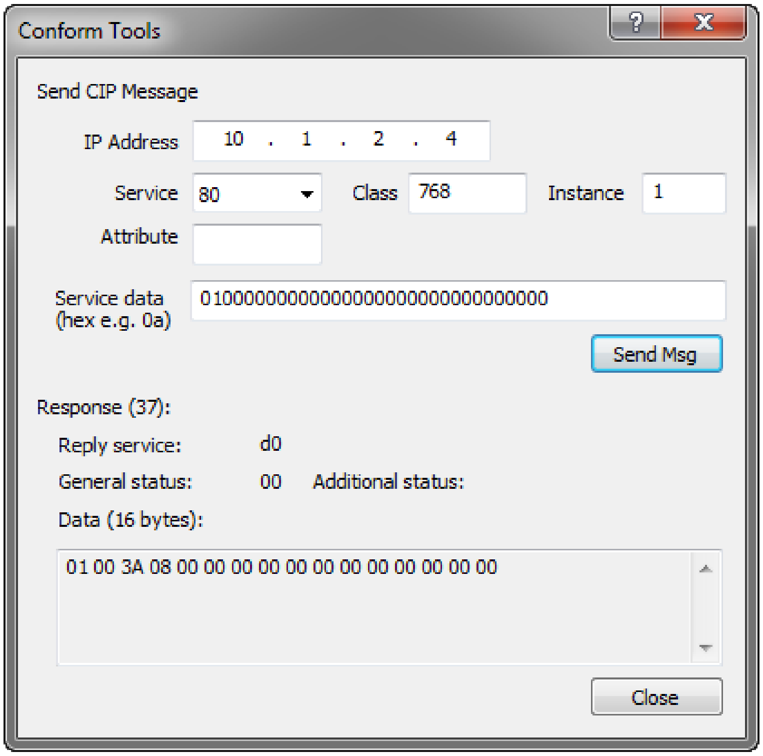
01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

The reply shows the function code.

Reply:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
0x00 00 00 01	0x00 00 00 00	0x00 00 00 00	0x00 00 00 00

In case an error occurs, the error code is passed in the reply.



IP Address 10 . 1 . 2 . 4

Service 80 Class 768 Instance 1

Attribute

Service data (hex e.g. 0a) 0100000000000000000000000000000

Response (37):

Reply service: d0

General status: 00 Additional status:

Data (16 bytes):

```
01 00 3A 08 00 00 00 00 00 00 00 00 00 00 00 00
```

Reply:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
0x08 3A 00 01	0x00 00 00 00	0x00 00 00 00	0x00 00 00 00

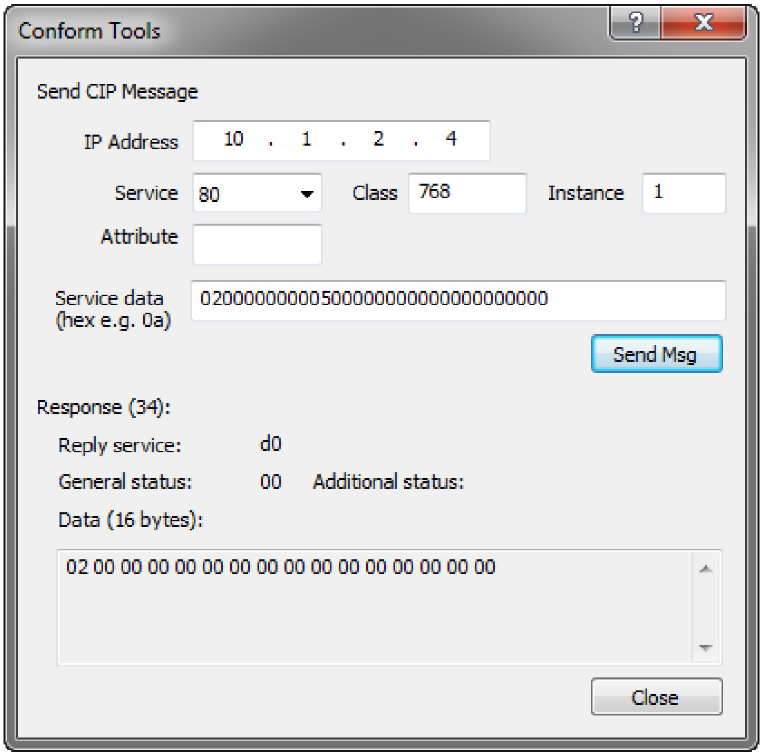
Error code = 0x083A = 2106 = A/D reads all 1's, Hardware Overload on loadcell

Calibrating the gain point is function code 2. Parameter 2 is used to enter the weight.

Calibrate span at 1280 → 00 00 05 00

Request:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
0x00 00 00 02	0x00 00 05 00	0x00 00 00 00	0x00 00 00 00



Conform Tools

Send CIP Message

IP Address 10 . 1 . 2 . 4

Service 80 Class 768 Instance 1

Attribute

Service data (hex e.g. 0a) 02000000000500000000000000000000

Response (34):

Reply service: d0

General status: 00 Additional status:

Data (16 bytes):

```
02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

The reply shows the function code.

Reply:

Parameter 1	Parameter 2	Parameter 3	Parameter 4
0x00 00 00 02	0x00 00 00 00	0x00 00 00 00	0x00 00 00 00

About PENKO

At PENKO Engineering we specialize in weighing. Weighing is inherently chemically correct, independent of consistency, type or temperature of the raw material. This means that weighing any kind of material guaranties consistency and thus, it is essential to sustainable revenue generation in any industry. As a well-established and proven solution provider, we strive for the ultimate satisfaction of custom design and/or standard applications, increasing your efficiencies and saving you time, saving you money.

Whether we are weighing raw materials, components in batching, ingredients for mixing or dosing processes, - or weighing of static containers and silos, or - in-motion weighing of railway wagons or trucks, by whatever means required during a process, we are essentially forming vital linkages between processes and businesses, anywhere at any time. We design, develop and manufacture state of the art technologically advanced systems in accordance with your strategy and vision. From the initial design brief, we take a fresh approach and a holistic view of every project, managing, supporting and/or implementing your system every step of the way. Curious to know how we do it? www.penko.com

Certifications

PENKO sets high standards for its products and product performance which are tested, certified and approved by independent expert and government organizations to ensure they meet – and even – exceed metrology industry guidelines. A library of testing certificates is available for reference on:
http://penko.com/nl/publications_certificates.html

PENKO Professional Services

PENKO is committed to ensuring every system is installed, tested, programmed, commissioned and operational to client specifications. Our engineers, at our weighing center in Ede, Netherlands, as well as our distributors around the world, strive to solve most weighing-system issues within the same day. On a monthly basis PENKO offers free training classes to anyone interested in exploring modern, high-speed weighing instruments and solutions. Training sessions on request: **www.penko.com/training**

PENKO Alliances

PENKO’s worldwide network: Australia, Brazil, China, Denmark, Germany, Egypt, Finland, France, India, Italy, Netherlands, Norway, Poland, Portugal, Slovakia, Spain, Syria, Turkey, United Kingdom, South Africa, Slovakia Sweden and Switzerland, Singapore.
A complete overview you will find on: **www.penko.com/dealers**