Bus protocol functions

of Leo3 Manometers

1 Introduction

This document describes the bus functions of Leo3 Manometers. In addition, there is further information about the use of the PC-software driver (DLL) included.
A detailed description of the bus protocol is provided in the document „Bus protocol“. Please read the bus protocol description before using this document.

2 Description of the Functions

2.1 Function 48 : Confirmation for initialization

Request:

DEV. ADDR	48	CRC16 H	CRC16 L

Response:

DEV. ADDR	48	CLASS	GROUP	YEAR	WEEK	BUF	STAT	CRC16 H	CRC16 L

Remark:

After each power-on-reset of a device (caused by applying the power supply or after an interruption of the supply) and after a change of the device configuration (i.e. by a keyboard action), the device has to be initialized again. The request of any other function always leads to an answer with exception 32 as long as the confirmation of the initialization function was not sent to the device.

The response of the device after the confirmation of initialization function contains the following information :

CLASS, GROUP	device identification		
	currently, the following devices are defined :		
	CLASS	0	Prototype
		1	Transmitter
		2	Progres programmer
		3	Manometer
		5	Series 30
YEAR, WEEK	Software version		
BUF	Length of the transmission buffer		
STAT	Status		
	Bit	0	device was already initialized

2.2 Function 66 : Writing of a new device address

Request:

DEV. ADDR	66	NEW ADDR	CRC16 H	CRC16 L

Response:

DEV. ADDR	66	0	CRC16 H	CRC16 L

Exception:

32 device is not initialized (see function 48 : Confirmation of initialization)

Remark:

This function is only supported by devices that are bus operational. In these cases, it has to be taken care about the addresses. In a bus system, the address NEW ADDR may not be used by any of the other devices connected to the bus.

2.3 Function 69 : Reading the serial number

Request:

DEV. ADDR	69	CRC16 H	CRC16 L

Response:

DEV. ADDR	69	SN3	SN2	SN1	SN0	CRC16 H	CRC16 L

Exception:

32 device is not initialized (see function 48 : Confirmation of initialization)

Remark:

The serial number is defined by the manufacturer and it is unique. It is built by 4 bytes and calculated by the following formula :

SN = 256^3*SN3 + 256^2*SN2 + 256*SN1 + SN0

2.4 Function 73 : Reading the value of a selected channel

Request:

DEV. ADDR	73	CH	CRC16 H	CRC16 L

Response:

DEV. ADDR	73	EXP	M0	M1	M2	STAT	CRC16 H	CRC16 L

Exception:

2 if CH > 5
32 device is not initialized (see function 48 : Confirmation of initialization)

Remark:

The value is reported in the IEEE754 format.

CH	channel
0	---
1	P1
2	---
3	T
4	ToB1
5	---

(ToB = Top of Bridge)

Reference has no Unit, P1, P2 are in bar, T, ToB1, ToB2 are in °C.

The **STAT**-Byte contains the actual status.

Bit position	.7	.6	.5	.4	.3	.2	.1	.0
Name	/STD	ERR2	TOB2	TOB1	T	P2	P1	REF

If the bit **/STD** is set, the transmitter is in the adjust or in the power-up mode. Bit **/STD** cleared means, that the transmitter is in the normal measurement mode.

Bit **ERR2** set means, that a calculation error occurred during the DAC calculation.

The bits **REF**, **P1**, **P2**, **T**, **TOB1**, **TOB2** set means, that either a measurement or a calculation error occurred at the corresponding channel.

2.5 Function 95 : Calculating functions

Request:

DEV. ADDR	95	CMD	CRC16 H	CRC16 L

Response:

DEV. ADDR	95	0	CRC16 H	CRC16 L

Exception:

1 if in adjust mode
2 if CMD > 4
32 device is not initialized (see function 48 : Confirmation of initialization)

Remark:

With this function, the following actions can be initiated :

CMD	Action
0	Zero P1
1	Reset P1
2	Zero P2
3	Reset P2

3 Description of the PC-software driver (DLL)

3.1 General

This PC-software driver ***S30.DLL*** is running under the operating systems Windows 95, 98 and NT.

For any parameter of the functions the convention **stdcall** is used. This means, that
- all parameters are transferred via the stack,
- the last parameter in a parameter list (the one on the right) is calculated and put on the stack first and the first parameter in the list (most left) is the last one put on the stack.,
- the parameters will be removed from the stack by the function itself.

Some of the variables of the functions are declared by the leading statement ***var***. This means, that these variables are transferred by reference (as a pointer) instead by value.
The used variable types are defined as follows :

Type	Range	Format
Byte	0..255	8-bit without sign
Word	0..65535	16-bit without sign
Integer	-32768..32767	16-bit with sign
Longint	-2147483648.. 2147483647	32-bit with sign
Pbyte		Pointer to a byte
Single	$+/- 1.5x10^{-45}..3.4x10^{38}$	32-bit

3.2 The functions of the DLL

Any function returns a value, that reports, if the function was executed successfully or not. The following table shows the returned values. Only if a function was successful, the resulting parameters are valid and can be used for further calculations.

Returned value		Description
RS_OK	0	Function successfully executed; parameters are valid
RS_EX1	1	Function successfully executed; but Exception 1 was occurred
RS_EX2	2	Function successfully executed; but Exception 2 was occurred
RS_EX3	3	Function successfully executed; but Exception 3 was occurred
RS_EX32	32	Function successfully executed; but Exception 32 was occurred
RS_ERROR	-1	general error
RS_TXERROR	-2	transmission error
RS_RXERROR	-3	reception error (at the UART)
RS_TIMEOUT	-4	no data or less than expected data receipt
RS_BADDATA	-5	Bad data (i.e. wrong CRC16)

3.2.1 Port functions

The transmitters are connected to the PC by a serial interface (COM port). The port functions are used to open and close these ports. COM1 through COM9 are valid port numbers. In case the COM port could be opened, the function OpenComPort returns the value RS_OK. If not, it returns RS_ERROR. Any open COM port will be closed by the DLL automatically at the termination of the user program. As timeout-value you should use at least 200(ms). With function **OpenComBaud** you have the possibility to initialize the serial-interface to a required baudrate. Valid values are: 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 Baud.

function OpenComPort(intPort, intTimeout: Smallint): Smallint; **stdcall**; **export**;

function OpenComBaud(intPort, intTimeout: Smallint, longBaud, Longint): Smallint; **stdcall**; **export**;

function CloseComPort: Integer; **stdcall; export;**

3.2.2 Echo Function

This function is meant to whether use a hardware echo or not. The values are:

0 = echo off
1 = echo on (default)

Keller converters do make a hardware echo, therefore this value is set to <1> by default.Other converters may not echo and you need to clear the value <0> to make it work.

function EchoOn(bteEcho: Byte): Smallint; **stdcall; export;**

3.2.3 Protocol functions

The following functions contain the real bus functions. The order of the parameters are identical to the bus functions. Some parameters are built by several bytes. In order to have a better overview, they are summarized to better readable variables.

function F48(DeviceAddr: Byte; var Class, Group, Year, Week, Buffer, State: Byte): Integer; **stdcall; export;**

function F66(DeviceAddr, NewAddr: Byte): Integer; **stdcall; export;**

function F69(DeviceAddr: Byte; var SN: Longint): Integer; **stdcall; export;**

function F73(DeviceAddr, Channel: Byte; var Value: Single; var Stat: Byte): Integer; **stdcall; export;**

function F95(DeviceAddr, Cmd: Byte): Integer; **stdcall; export;**

function F95val(bteDeviceAddr, bteCmd: Byte; sinVal: Single): Smallint; **stdcall**; **export**;