Kommunikationsprotokoll für Manometer Leo-Record

1 Einleitung

In diesem Dokument wird das Kommunikationsprotokoll für den Daten-Logger LEO_RECORD von KELLER Druckmesstechnik beschrieben. Nebst diesen Daten-Loggern werden ausserdem noch andere Geräte, wie Transmitter oder Manometer, angeboten. Diese Produkte werden durch die Bezeichnung CLASS unterschieden. Innerhalb dieser Geräteklasse werden die einzelnen Gerätegruppen mit der Bezeichnung GROUP differenziert.

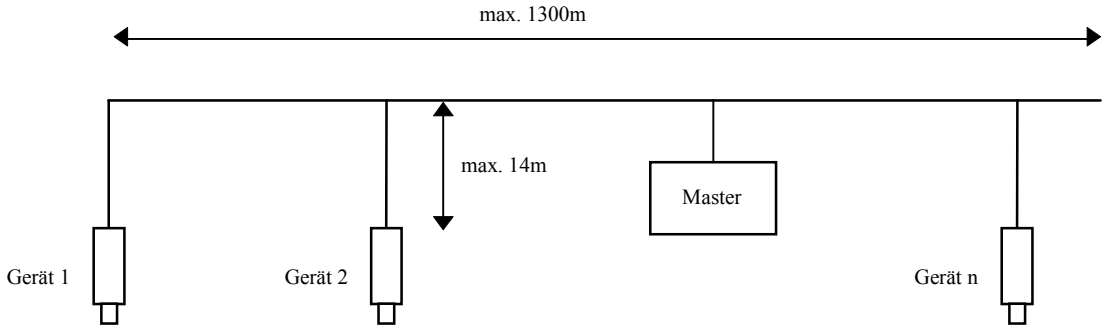
2 Bit-Übertragungsschicht (Physical Layer)

2.1 Einleitung

Als physikalische Verbindung wird die serielle Schnittstelle RS485 verwendet. Diese gewährleistet eine hohe Störsicherheit und ermöglicht den Aufbau einer flexiblen Busstruktur, d.h. mehrere Geräte können als Slaves durch einen Master verwaltet werden.
Um den Verdrahtungsaufwand so gering wie möglich zu halten, wird die RS485 im Halb-Duplex-Verfahren verwendet.
Somit werden nur 2 Leiter für die Kommunikation benötigt.

2.2 Charakteristik

Um mehrere Geräte an einer seriellen Schnittstelle zu betreiben, werden diese einfach alle parallel angeschlossen (RS485A, RS485B). Vor dem Einsetzen der Geräte in den Bus muss jedes Gerät mit einer anderen Adresse programmiert werden.
Es ist möglich, ein Netzwerk bis zu einer Länge von 1300 Meter mit max. 128 Geräten aufzubauen. Jede Steigleitung darf bis zu 14 m lang sein. Das verwendete Kabel soll der Spezifikation EIA RS485 entsprechen. Dies bedeutet z.B., dass die Leiter paarweise verdrillt sein müssen.



Folgender RS485 Treiber wird in den LEO_RECORD-Geräten eingesetzt: MAX3471

Diese Treiber sind slew-rate limitiert und benötigen keine Bias-Widerstände. In den meisten Anwendungen sind auch Abschlusswiderstände nicht erforderlich.
In den Geräten ist an beiden Leitungen (RS485A und RS485B) ein Spannungsschutz angebracht. Die common-mode Spannung gegenüber GND ist –7V ... + 12 V. Diese Spannung darf auf keinem Fall überschritten werden.

Weitere Informationen zu RS485: http://www.maxim-ic.com/MaximProducts/Interface/rs-485.htm
Informationen zur Verdrahtung: http://www.maxim-ic.com/appnotes.cfm/appnote_number/763

3 Sicherungsschicht (Data-Link-Layer)

An diesem Abschnitt wird beschrieben, wie der Datenaustausch auf diesem Bus erfolgt. Die Daten und deren Kontroll- und Steuerstrukturen sind zu Botschaften zusammengefasst. Diese bilden die kleinste Kommunikationseinheit, d.h. es können nur Botschaften zwischen den Geräten ausgetauscht werden. Da es sich um ein Halb-Duplex-Protokoll handelt, kann nur ein Gerät gleichzeitig den Bus als Sender benutzen. Alle anderen Geräte befinden sich dann im Empfängermodus. Als Master tritt z.B. ein PC oder ein Mikrokontroller auf, die Geräte sind die Slaves. Jeder Botschaftsaustausch erfolgt unter der Kontrolle des Masters. In der Botschaft ist die Adresse für den angesprochenen Slave enthalten.

Somit ergeben sich folgende 2 Möglichkeiten für den Datenaustausch:

a) Rundruf	Diese Art der Kommunikation erlaubt dem Master eine Botschaft an alle Slaves gleichzeitig zu senden. Jedoch erhält der Master keine Antwort, kann also nicht überprüfen, ob die Botschaft von jedem Slave korrekt empfangen wurde.
b) Datenaustausch	Diese Art der Kommunikation ermöglicht dem Master mit einem einzelnen Slave zu kommunizieren. Im Normalfall bedeutet dies die Übertragung von zwei Botschaften: der Master sendet eine Anforderung und der Slave antwortet daraufhin. Nur der Master selber darf eine Antwort anfordern. Die Anforderung empfängt jeder Slave, jedoch antwortet nur der ausgewählte. Die Antwort muss innerhalb einer festgelegten Zeit erfolgen, andernfalls wertet der Master den Versuch als erfolglos und muss ihn wiederholen.

3.1 Übertragungsformat der seriellen Schnittstelle

Die Informationen werden seriell über den Bus gesendet. Dabei gilt folgendes Format:

- **1 Start-Bit**
- **8 Daten-Bits (das niederwertigste Bit zuerst)**
- **1 Stopp-Bit**
- **kein Parity-Bit**
- **9600 Baud**

Somit ergeben sich 10 Bit pro Übertragungs-Byte.

3.2 Format einer Botschaft

3.2.1 Format der vom Master gesendeten Botschaft

Bemerkung zur Darstellung der Botschaften: Jeder Kasten stellt 1 Daten-Byte von 8 Bit dar, wenn nicht explizit anders angegeben.

Jede vom Master gesendete Botschaft besitzt folgendes Format:

DevAddr	0	Function code	n Byte Parameters (optional)	CRC16_H	CRC16_L

- `DevAddr`: Adresse des Gerätes.
  Die Adresse 0 ist für den Rundruf reserviert.
  Die Adressen 1...249 können für einen Busbetrieb benutzt werden.
  Die Adresse 250 ist transparent und für den Nicht-Busbetrieb reserviert. Jedes Gerät ist mit dieser Adresse ansprechbar.
  Die Adressen 251...255 sind für spätere Entwicklungen reserviert.
- `Function code`: Funktionsnummer
  Über die Funktionsnummer wird eine Funktion ausgewählt und vom Gerät ausgeführt. Die Funktionsnummer wird in 7 Bits verschlüsselt, Bit 7 ist immer = 0. Die Funktionen werden in diesem Dokument unter dem Kapitel „Beschreibung der Funktionen“ näher erläutert.
- `Parameters`:
  Es folgen die von der Funktion benötigten Parameter (n = 0 .. 6, je nach Funktion)
- `CRC16`: 16-Bit Prüfsumme
  Diese zwei Kontroll-Bytes dienen der Überprüfung der Integrität der empfangenen Daten. Wird ein Fehler festgestellt, wird die gesamte Botschaft verworfen. Das verwendete Prinzip zur CRC16-Berechnung ist im Anhang beschrieben. Es handelt sich um den CRC16-Standard.

Bemerkung: Die Länge einer Botschaft vom Master beträgt somit mindestens **4** Bytes.

3.2.2 Format der vom Slave gesendeten Botschaft

Eine vom Slave gesendete Botschaft besitzt folgendes Format:

DevAddr	X	Function code	n Byte Data (optional)	CRC16_H	CRC16_L

- `DevAddr`: Adresse des Gerätes.
  Diese Adresse entspricht der Adresse des antwortenden Gerätes.
- `Function code`:
  Die Funktionsnummer ist identisch mit der vom Master gesendeten Funktionsnummer. Ist das höchstwertige Bit X = 0, so konnte die Funktion korrekt ausgeführt werden. Ist das Bit X = 1, trat ein Ausnahmefehler auf.
- `Data`:
  Eventuell über die Funktion angeforderte Daten folgen nun.
- `CRC16`: 16-Bit Prüfsumme
  Siehe oben.

Bemerkung: Die Länge einer Botschaft vom Slave beträgt somit mindestens **5,** maximal **10** Bytes.

3.3 Prinzip des Botschaften-Austausches

3.3.1 Generelle Regeln

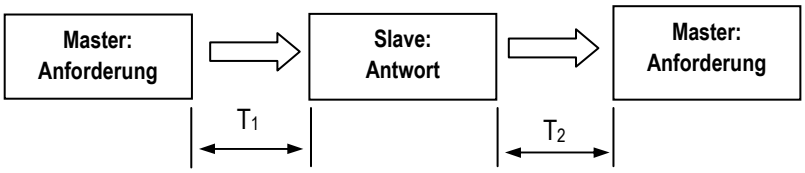
- Dieselbe Adresse darf nur **einem** am Bus angeschlossenen Gerät zugeteilt werden. Haben trotzdem zwei Geräte auf dem Bus die gleiche Adresse, so antworten beide, was zu einem Konflikt (Buskollision) führt.
- Der Master leitet jeden Datenaustausch ein. Das heisst, ein Gerät darf nur dann etwas senden, wenn es dazu vom Master aufgefordert wird.
- Eine Botschaft besteht aus mehreren Bytes. Diese müssen sowohl vom Master zum Slave wie auch vom Slave zum Master **ohne Unterbruch** gesendet werden.
- Das angesprochene Gerät muss innerhalb der Zeit T_1 antworten, sonst ist die Botschaft ungültig.
- Nachdem das Gerät geantwortet hat, muss 1ms gewartet werden, bevor ein neues Kommando gesendet werden darf.

Bit-Frame:

	ST	D0	D1	D2	D3	D4	D5	D6	D7	SP	ST	D0	…	D6	D7	SP

Wobei: ST: Start-Bit SP: Stop-Bit. D0 .. D7: 8 Daten-Bit

Message-Frame:



Antwort-Zeiten:

- T_1: Zeit nach dem Empfangen der Anfrage bis zum Beginn der Antwort.
  Min. 1ms bis max. 500ms für alle Funktionen und Geräte (Empfohlen 200ms).
- T_2: Zeit zur Herstellung der Empfangsbereitschaft des Slaves: 500 µs. Es wird daher empfohlen nach dem Empfang einer Botschaft mindestens 1ms zu warten, bis die nächste Botschaft zum Slave gesendet wird.

3.3.2 Fehlerbehandlung

Während eines Botschaftsaustausches zwischen Master und Slave können 2 Arten von Fehlern auftreten. Zum einen sind das die Übertragungsfehler und zum anderen die Ausnahmefehler.

3.3.2.1 Übertragungsfehler

Diese Fehler sind im Wesentlichen auf Leitungsstörungen zurückzuführen. Das Format der Botschaft ist falsch. Es ergeben sich folgende Möglichkeiten:

- Eine empfangene Botschaft ist zu klein.
- Eine Botschaft ist länger als der interne Übertragungspuffer es zulässt.
- Die Länge der Botschaft ist nicht korrekt und kann daher nicht richtig interpretiert werden.
- Die CRC16-Prüfsumme ist nicht korrekt.

Als Reaktion auf einen Übertragungsfehler werden alle empfangenen Daten ignoriert. Der Slave bleibt im Empfangsmodus während der Master einen neuen Datenaustausch initiieren muss.

3.3.2.2 Ausnahmefehler

Die Botschaft wurde korrekt empfangen (es liegt kein Übertragungsfehler vor), jedoch sind die übergebene Funktionsnummer bzw. die Parameter ungültig. Der Slave antwortet mit einem Ausnahmefehler, sofern die Botschaft nicht im Rundruf-Modus empfangen wurde.
Die als Antwort vom Slave gesendete Botschaft hat folgendes Format:

DevAddr	1	Function code	Exception code	CRC16_H	CRC16_L

Es sind 4 Arten von Ausnahmefehlern definiert :

- nicht implementierte Funktion 1
- falsche Parameter 2
- fehlerhafte Daten 3
- Initialisierung fehlt (F48) 32

Der Ausnahmefehler 32 tritt auf, wenn das Gerät neu aufgestartet ist und die Initialisierung nicht durchgeführt wurde. Dies geschieht jedes Mal, wenn das Gerät nach einer Versorgungsunterbrechung neu angeschlossen wird.

3.3.3 Aktivieren der Schnittstelle des LEO_RECORD

Um den Stromverbrauch des LEO_RECORD möglichst klein zu halten, wird die Schnittstelle während des normalen Betriebs deaktiviert.
Sie wird aktiviert, sobald ein Zugriff (Kommunikation) auf die Schnittstelle erfolgt. Die dabei gesendeten Daten gehen jedoch verloren und es wird keine Antwort gegeben. Die Botschaft muss daher nochmals gesendet werden.

Zehn Sekunden nach der letzten Kommunikation, deaktiviert das LEO_RECORD die Schnittstelle wieder.
Bei aktiver Schnittstelle werden auch fortlaufend alle verfügbaren Kanäle gemessen (jede Sekunde), somit kann über die Funktion 73 immer der aktuelle Messwert ohne Verzögerung ausgelesen werden.

Beispiel:
Schnittstelle des LEO_RECORD ist deaktiviert.

- Der PC sendet Initialisierungsbefehl (Funktion 48):
  → Das LEO_RECORD antwortet nicht (Timeout-Fehler).Die Schnittstelle wird aktiviert.
- Der PC sendet erneut Initialisierungsbefehl:
  → Das LEO_RECORD Antwortet.

4 Beschreibung der Funktionen

In diesem Abschnitt werden die Funktionen des Busprotokolls für den Daten-Logger (Gerät 5.5) beschrieben.

Übersicht:

Funktion 30: Float lesen
Funktion 31: Float schreiben
Funktion 48: Initialisieren
Funktion 66: Schreiben der neuen BUS-Adresse
Funktion 68: Lesen des RECORD-ROMs (nicht Bus-fähig)
Funktion 69: Lesen der Serienummer
Funktion 73: Lesen des Wertes des gewählten Kanals
Funktion 92: Lesen der Record-Konfiguration
Funktion 93: Schreiben der Record-Konfiguration
Funktion 95: Nullungsfunktionen
Funktion 100: Konfigurationen lesen

4.1 Funktion 30: Float lesen

Anforderung:

DevAddr	30	Nr.	CRC16_H	CRC16_L

Antwort:

DevAddr	30	B3	B2	B1	B0	CRC16_H	CRC16_L

Ausnahmefehler:

2 wenn Nr. > 111
3 wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert

Bemerkung:
Mit dieser Funktion kann jeder Koeffizient im IEEE754-Format (Gleitkomma-Format 4-Byte B0 .. B3) gelesen werden
→ Informationen zu IEEE754: siehe Anhang.
Nicht benutzte Koeffizienten enthalten undefinierte Werte (NaN).

N	Koeffizient	N	Koeffizient	N	Koeffizient	N	Koeffizient
64	**P1OFFS**	65	**P1Gain**	66	**P2OFFS**	67	**P2Gain**
78	ADJUST_File	79	ADJUST_File				
80	P1_MIN	81	P1_MAX	82	P2_MIN	83	P2_MAX
84	T_MIN	85	T_MAX	86	TOB1_MIN	87	TOB1_MAX
88	TOB2_MIN	89	TOB2_MAX				
96	RC_ModusVal1	97	RC_ModusVal2			99	AdjustDate
100	100....111 frei für Kunden lesen und schreiben						

Nr. 64 .. 67: Offset/Gain -Kalibration in bar, die mit **F95 /F31** verändert werden können: **default = 0/1**.
Nr. 80... 87: Abgleich-Informationen nur lesen
Nr. 96 & 97: Werte für ereignisgesteuerte Aufzeichnung (RECORD): nur im RAM
Nr. 98..111: „Custom-Value“ frei für Kunden lesen und schreiben (Installationsdaten)

Skalierung der Kanäle P1 und P2:
P1 und P2 sind linear skalierbar mit Nullpunkt und Verstärkungsfaktor: **Wert = Verstärkungsfaktor * Wert + Offset**
Standard-Werte: Offset = 0, Verstärkungsfaktor = 1.0
Die Offset-Werte können auch mit Funktion 95 beeinflusst werden (siehe Funktion 95).
Der Verstärkungsfaktor sollte **nur für Kalibrationszwecke** verwendet werden und nicht zur Änderung von Druck-Einheiten. Dies sollte unbedingt auf der Seite des Masters erfolgen.

4.2 Funktion 31: Float schreiben

Anforderung:

DevAddr	31	Nr.	B3	B2	B1	B0	CRC16_H	CRC16_L

Antwort:

DevAddr	31	0	CRC16_H	CRC16_L

Ausnahmefehler:

2 Schreiben nicht erlaubt
3 Wenn falsche Länge der Botschaft
32 Wenn Gerät noch nicht initialisiert

Bemerkung:
Informationen zu der Skalierung der Kanäle: Siehe Funktionen 73 und 95. Informationen, welche Kanäle aktiv sind: Siehe Funktion 100.

4.3 Funktion 48 : Initialisieren

Anforderung:

DevAddr	48	CRC16_H	CRC16_L

Antwort:

DevAddr	48	CLASS	GROUP	YEAR	WEEK	BUF	STAT	CRC16_H	CRC16_L

Ausnahmefehler:

3 wenn falsche Länge

Bemerkung:
Nach jedem Einschalten des Gerätes, welches durch Anlegen der Versorgungsspannung oder nach einem Spannungsunterbruch erfolgt, muss das Gerät mit dieser Funktion initialisiert werden. Der Aufruf einer anderen Funktion führt zu einem **Ausnahmefehler 32**.

Folgende Angaben werden zurückgegeben:

CLASS Gerätekennung
5: Digitale Drucktransmitter

GROUP Unterteilung innerhalb einer Geräteklasse
5: Daten-Logger LEO_RECORD

YEAR, WEEK Firmware-Version

BUF Länge des internen Übertragungspuffers

STAT Status-Angaben
0: Gerät wird nach dem Einschalten das erste mal angesprochen.
1: Gerät war schon initialisiert

4.4 Funktion 66 : Schreiben und lesen der BUS-Adresse

Anforderung:

DevAddr	66	NewAddr	CRC16_H	CRC16_L

Antwort:

DevAddr	66	ActAddr	CRC16_H	CRC16_L

Ausnahmefehler:

3 Wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert

Bemerkung:
Diese Funktion programmiert die Geräte-Adresse auf NewAddr. Zur Bestätigung wird die Adresse in ActAddr zurückgegeben. Es ist darauf zu achten, dass die neue Adresse NewAddr nicht schon durch einen anderen Busteilnehmer verwendet wird.

Zulässige Adressen: 1 .. 249. Die Adresse 250 ist transparent. Das heisst: Jedes Gerät, unabhängig der eingestellten Adresse, antwortet auf die Adresse 250. Die *transparente* DevAddr = 250 darf somit nur im Einzelbetrieb verwendet werden!

Zum **Lesen der Geräte-Adresse**, wenn z.B. die Adresse nicht bekannt ist, wird als DevAddr der Wert 250 und als NewAddr der Wert 0 übergeben. Als Rückgabe wird die aktuelle Adresse zurückgegeben.

4.5 Funktion 68 : Lesen des RECORD-ROMs (nicht Bus-fähig)

Anforderung:

DevAddr	68	Page_H	Page_L	Index	CRC16_H	CRC16_L

Antwort:

DevAddr	68	DATA 0	…	DATA 7/63	CRC16_H	CRC16_L

Ausnahmefehler:

2 wenn Page > RecRomLastPagePhysik
3 wenn falsche Länge
32 wenn Gerät noch nicht initialisiert

Bemerkung:

Index: Wenn Index = 0 , dann werden nur die ersten 8 Byte der Page (Header) übermittelt (DATA 0...7).
Wenn Index = 1 , dann wird die ganze Page übermittelt (DATA 0...63).
Page : Aufteilung und Grösse siehe Funktion 93 ! (0...RecRomLastPagePhysik)

Da diese Funktion nicht der Protokoll-Vereinbarung entspricht (maximal 10 Übertragungsbyte), darf sie nicht verwendet werden, wenn mehrere Geräte in einem Bus zusammengeschlossen sind.

4.6 Funktion 69 : Lesen der Serienummer

Anforderung:

DevAddr	69	CRC16_H	CRC16_L

Antwort:

DevAddr	69	SN3	SN2	SN1	SN0	CRC16_H	CRC16_L

Ausnahmefehler:

3 Wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert.

Bemerkung:

Die Serienummer wird werksseitig vergeben. Sie besteht aus 4 Byte und wird wie folgt berechnet :

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

4.7 Funktion 73 : Wert eines Kanals lesen (Gleitkomma)

Anforderung:

DevAddr	73	CH	CRC16_H	CRC16_L

Antwort:

DevAddr	73	B3	B2	B1	B0	STAT	CRC16_H	CRC16_L

Ausnahmefehler:

2 wenn CH > 5
3 Wenn falsche Länge der Botschaft
32 wenn das Gerät noch nicht initialisiert ist

Bemerkung:
Ein Gerät kann bis zu fünf Signale (Kanäle) messen:
Zwei unabhängige Drucksensoren P1 und P2. Dazu jeweils die Temperaturen des Drucksensors TOB1 und TOB2. Die Temperaturen der Drucksensoren (TOB1, TOB2) werden für die Temperaturkompensation des Drucksignals benötigt. Zusätzlich kann ein Temperaturfühler (T) gemessen werden.
P1 – P2 ist ein berechneter Kanal.
Bei einem Standard Drucktransmitter sind nur die Kanäle P1 und TOB1 verfügbar. Mit der Funktion 100 können Sie auslesen, welche Kanäle aktiv sind.

Der Messwert wird im IEEE754-Format (4-Byte B0 ... B3) zurückgegeben.

CH	Bezeichnung	Beschreibung	Einheit
0	P1 - P2	Differenz der beiden Drucksensoren	bar
1	P1	Druck von Drucksensor 1	bar
2	P2	Druck von Drucksensor 2	bar
3	T	Zusätzlicher Temperaturfühler	°C
4	TOB1	Temperatur von Drucksensor 1	°C
5	TOB2	Temperatur von Drucksensor 2	°C

Das **STAT**-Byte enthält den aktuellen Status.

Bit-Position	.7	.6	.5	.4	.3	.2	.1	.0
Bezeichnung	/STD	---	TOB2	TOB1	T	P2	P1	---

Ein gesetztes Bit **/STD** gibt an, ob sich der Transmitter im Abgleich- bzw. Power-up-Modus befindet, andernfalls befindet er sich im Standard-Modus.

Ein gesetztes Bit **P1**, **P2**, **T**, **TOB1**, **TOB2** zeigt an, dass ein Mess- oder Berechnungsfehler im entsprechenden Kanal auftrat.

LEO-Record SW-Version >= Jahr 11

CH	Bezeichnung	Beschreibung	Einheit
20	V_{Bat}	Batteriespannung	Volt

Die Batterie des LEO-Records (Tadiran SL-760) hat eine Leerlaufspannung von ca 3,65Volt, leer < 3.1Volt

Das **STAT**-Byte beim Auslesen von Index 20 (Batteriespannung) hat folgende Bedeutung.

Bit-Position	.7	.6	.5	.4	.3	.2	.1	.0
Bezeichnung	---	---	---	---	---	---	ExternVcc	BatLow

ExternVcc = 1 ==> Externe Speisung angeschlossen
BatLow = 1 dann Anzeige BatLow in LCD bzw Batterie leer

4.8 Funktion 92 : Lesen der Record-Konfiguration

Anforderung:

DevAddr	92	Index	CRC16_H	CRC16_L

Antwort:

DevAddr	92	PARA0	PARA1	PARA2	PARA3	PARA4	CRC16_H	CRC16_L

Ausnahmefehler:

2 wenn Index > 8
3 wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert

Bemerkung:
Indextabelle siehe Funktion 93

4.9 Funktion 93 : Schreiben der Record-Konfiguration

Anforderung:

DevAddr	93	Index	PARA0	PARA1	PARA2	PARA3	PARA4	CRC16_H	CRC16_L

Antwort:

DevAddr	93	0	CRC16_H	CRC16_L

Ausnahmefehler:

2 wenn Index >9
3 wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert

Bemerkung:

Alle Variablen sind lesbar.

Markierte Variablen sind beschreibbar, sie werden im RAM des Gerätes gespeichert. Dies bedeutet, dass sie bei einem Batterie-Wechsel verloren gehen.

Index	Para0	Para1	Para2	Para3	Para4	Bemerkung
0	FUNC	---	---	---	---	---
1	CFG	REC_CTRL	EE_CTRL	PAGE_H	PAGE_L	---
2	RecRomFirstPagePhysik_H	RecRomFirstPagePhysik_L	RecRomLastPagePhysik_H	RecRomLastPagePhysik_L	RecRomAnzTextPage	EEPROM Info
3	Time_Byte3	Time_Byte2	Time_Byte1	Time_Byte0	---	Aktuelle Zeit
4	Start_Time_Byte3	Start_Time_Byte2	Start_Time_Byte1	Start_Time_Byte0		Record_StartTime
5	LoadFixCounter_LH	LoadFixCounter_LL	---	---	---	* Bis 0310
6	LoadModusMessCounter_H	LoadModusMessCounter_L	LoadFastMessCounter_H	LoadFastMessCounter_H	LoadSaveCounter	---
7	ModusSelect_H	ModusSelect_L	ModusChannel	---	---	---
8	RecChannels _H	RecChannels _L	Available RecordCH_H	Available RecordCH_L	Bat_Capacity	---
*9	LoadFixCounter_HH	LoadFixCounter_HL	LoadFixCounter_LH*	LoadFixCounter_LL*	---	* Ab 03.10

**** ab Software-Version 03.10: Total 4 Byte LoadFixCounter. Bis 03.10 Index 5 benützen, danach Index 9 benützen***

Index 0:

FUNC

Bit	7	6	5	4	3	2	1	0
Beschr.						f_RecStart	f_RecStop	f_RecFree

f_RecFree = 1, dann ist der komplette Speicher zum beschreiben freigegeben.
Wenn 0 dann wird kein „alter Record" überschrieben und es steht nicht der ganze Speicherplatz zur Verfügung.

f_RecStop = 1, ein aktiver Record wird gestoppt.

f_RecStart = 1, ein Record wird gestartet (wenn Bedingung erfüllt gestartet)

Index 1:

CFG

Bit	7	6	5	4	3	2	1	0
Beschr.								fRE_CFG_Continue

Wenn **fRE_CFG_Continue** gesetzt, dann ist Endlosespeicherung aktiviert. (Nur schreiben)

REC_CTRL

Bit	7	6	5	4	3	2	1	0
Beschr.	fRE_Error	fRE_Active	fRE_Start				fRE_Memoryfull	fRE_Continue

fRE_Continue =1 endlose Speicherung. 0 = Aufzeichnen bis Speicher voll.
fRE_Memoryfull = 1 Es ist kein Platz mehr verfügbar. Freigeben mit f_RecFree
fRE_Start = 1 Startbedingungen gesetzt, warten auf Start (Trigger)
fRE_Active = 1 Recordkonfiguration geschrieben, Record läuft oder wartet auf Start
fRE_Error = 1 es ist ein Fehler aufgetreten, Record wird gestoppt.

EE_CTRL

Bit	7	6	5	4	3	2	1	0
Beschr.					f_AckError	f_LoBatError		

PAGE_H/L Aktive Page, auf die gespeichert wird

Index 2: Verfügbarer Record-Speicher

Index 3: **Time_Byte0…3:** Aktuelle Zeit im LEO_RECORD

Index 4: **Start_Time_Byte0…3:** Ist die aktuelle Zeit grösser als die Start-Zeit wird die Aufzeichnung ermöglicht.

Index 5: **LoadFixCounter:** Intervall für die FixeRekordaufzeichnung in s.

Index 6:

LoadModusMessCounter_H/L: Intervall für die Ereignisüberprüfung in s

LoadFastMessCounter_H/L: Intervall wenn das Ereignis (Ereignisüberprüfung) eingetroffen ist.

LoadSaveCounter: Nach LoadSaveCounter Anzahl Messungen wird der Messwert (Mittelwert) gespeichert. (Mittelwerterfassung immer im 1s Takt)

Index 7:

ModusSelect_H/L: Anhand des Bits wird die Aufzeichnungsart gewählt.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Beschr.											CH< Val1	CH> Val1	Delta CH	On Off	Intervall	Fix Intervall

ModusChannel: Die Modus-Bedingung wird anhand des Wertes des gewählten Kanals geprüft.

Index 8:

Available RecordCH_H/L & RecordChannels_H/L: die Kanäle sind Verfügbar, bzw. werden aufgezeichnet.

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Beschr.	---	CH 14	CH 13	CH 12	CH 11	CH 10	*CH9	*CH8	CH7	CH6	TOB2	TOB1	T	P2	P1	P1-P2

LeoRecord mit MinMax-Funktion

***CH8 *CH9** Beim Leo-RecordMinMax sind diese 2 Kanäle für die Speicherung der Min und Max Werte.
Die MinMax-Werte werden nur im FixIntervall-Modus gespeichert.
Ist ein Minoder ein Max-Kanal aktiviert, so misst das Leo-Record jede Sekunde den aktuellen Druck und führt den Min-Max-Wert nach.
Ist Zeit zum speichern (FixIntervall) werden die aktuellen , selektierten Kanäle gespeichert und die MinMax – Werte geresetet. D.H. Min und Max wird auf den aktuellen Messwert gesetzt.

Bat_Capacity Batterie-Ladung (Kapazität) 0...100%

4.10 Funktion 95 : Befehle für Setzen des Nullpunktes

Anforderungen:

Anforderung a:

DevAddr	95	CMD	CRC16_H	CRC16_L

Anforderung b mit Sollwert:

DevAddr	95	CMD	*B3*	*B2*	*B1*	*B0*	CRC16_H	CRC16_L

Antwort:

DevAddr	95	0	CRC16_H	CRC16_L

Ausnahmefehler:

1 wenn im Power-up-Modus
2 wenn CMD > 3
3 wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert

Bemerkung:
Feste Anzahl Bytes bei Modemkommunikation (Adresse 251): 5

Mit der Funktion 95 können die folgenden Aktionen durchgeführt werden:

CMD	Bedeutung
0	Nullpunkt von P1 setzten
1	Nullpunkt von P1 auf Standardwert zurücksetzen
2	Nullpunkt von P2 setzten
3	Nullpunkt von P2 auf Standardwert zurücksetzen

CMD 0, 2:
Nullpunkt-Werte für die Druckkanäle P1 und P2. Diese Werte können auch mit der Funktion 30 gelesen und mit der Funktion 31 beschrieben werden. Offset P1_Offset, P2_Offset oder CH0_Offset werden im RAM, sowie im EEPROM verändert.

Anforderung a: Der Nullpunkt wird so berechnet, dass der aktuelle Messwert = 0 wird.
Anforderung b: Der Nullpunkt wird so berechnet, dass der aktuelle Messwert gleich dem Sollwert wird.

CMD 1, 3: Rücksetzen des Nullpunktes auf Werkseinstellung
Die Nullpunkt-Werte werden auf 0 zurückgesetzt .

4.11 Funktion 100 : Konfiguration lesen

Anforderung:

DevAddr	100	Index	CRC16_H	CRC16_L

Antwort:

DevAddr	100	PARA0	PARA1	PARA2	PARA3	PARA4	CRC16_H	CRC16_L

Ausnahmefehler:

2 wenn Index > 8
3 wenn falsche Länge der Botschaft
32 wenn Gerät noch nicht initialisiert

Bemerkung:
CFG_P & CFG_T = Verfügbare Kanäle

Index	Para0	Para1	Para2	Para3	Para4
2	CFG_P	CFG_T	---	---	CNT_TCOMP

Index 2:

Kanaldefinition für Cfg_P und Cfg_T

.7	.6	.5	.4	.3	.2	.1	.0
CH7	CH6	TOB2	TOB1	T	P2	P1	P1-P2

Die in **CFG_P** abgelegten Kanäle werden maximal einmal pro Sekunde gemessen oder im Recordintervall.
Die in **CFG_T** abgelegten Kanäle werden nur gemessen, wenn Sie zum Aufzeichnen ausgewählt wurden, oder die Temperaturkompensation (Temperaturkorrektur vom Drucksignal) ausgeführt wird.

5 Anhang

5.1 Schnittstellen-Konverter

Zum Anschluss an einen PC kann die Serielle RS232 oder die USB-Schnittstelle verwendet werden. KELLER bietet dafür Konverter an. Es gibt aber auch im freien Handel diverse Produkte. Wenn Sie mit KELLER Software arbeiten, wird folgendes vorausgesetzt:

- Der Konverter muss die Umschaltung Senden / Empfangen automatisch steuern.
- KELLER Konverter haben ein Hardware-Echo, d. h. dass die gesendete Botschaft sofort als Echo wieder empfangen wird. Bei einigen Softwaren von KELLER wird dieses Echo vorausgesetzt.
- Konverter ohne Abschlußwiderstände betreiben !

5.2 Gleitkomma-Format IEEE754

Da die Übertragung der Daten Byte-weise (8-Bit Daten) erfolgt, werden die Gleitkomma-Werte wie folgt abgebildet:
B0: Bit 0..7; B1: Bit 8..15, B2: Bit 16..23, B3: Bit 24..31

Abbildung nach IEEE754:

s	e[30:23]	f[22:0]
31	30 23	22 0

Wenn Sie die von KELLER erhältliche DLL verwenden, müssen Sie sich nicht um die Umwandlung kümmern, da dies in der DLL gekapselt ist. Wenn Sie jedoch direkt die Geräte ansprechen wollen, müssen sie die einzelnen Bytes in einen Gleitkommawert konvertieren.
Um aus den einzelnen Bytes einen Gleitkomma-Wert zu erhalten gehen sie wie folgt vor:

1. Datenstruktur definieren, in der auf dem gleichen Speicherplatz ein Array von 4 Bytes und ein 32-Bit Gleitkomma-Wert definiert ist.
2. Die Bytes in das Byte-Array schreiben.
3. Den Gleitkomma-Wert auslesen.

Sie müssen also nichts umrechnen, da der Compiler die Interpretation übernimmt. Einige Mikrokontroller haben eine andere Datenstruktur für Gleitkomma-Werte. In solchen Fällen muss eine Anpassung erfolgen.

Weitere Informationen finden Sie unter:
http://cch.loria.fr/documentation/IEEE754/numerical_comp_guide/ncg_math.doc.html - 556

5.3 Memory-Map

Der gesamte Speicher wird in Pages unterteilt. Die Anzahl Pages kann je nach Geräte-Typ variieren. Beim YEAR.WEEK 02.35 sind es total 2048 Pages, bei der Version 03.15 sind 4096 Pages enthalten.
Jede Page besteht aus 64 Byte. Die ersten 8 Byte entsprechen dem Header. Alle nachfolgenden Byte sind die eigentlichen Daten. Ein Datensatz besteht aus vier Byte. Diese beinhalten die Messgrösse und den entsprechenden Messwert.

Gesamt-Speicher: 2047 oder 4096 Pages

Page	Beschreibung
0	Daten-Speicher
1	
2	
...	
...	
...	
X-1	Ende Daten-Speicher
X	Record-Text
...	
...	
...	
2047	Ende Record-Text

1 Page mit 64 Byte :

Byte	Beschreibung
0	Header
1	
2	
3	
4	
5	
6	
7	
8	Data 1
9	
10	
11	
12	Data 2
13	
14	
15	
...	...
...	...
...	...
60	Data 14
61	
62	
63	

Header
Der Header besteht aus 8 Byte.

Byte	Beschreibung des Header
0	Starterkennung / Überlaufzähler / Startzeiger
1	Restliche 8 Bit des Startzeigers
2	Absolute Zeit 1. Byte1
3	Absolute Zeit 2. Byte2
4	Absolute Zeit 3. Byte3
5	Absolute Zeit 4. Byte4
6	Reserviert
7	Reserviert

Absolute Zeit in Sekunden seit 01.01.2000
(2^32*Byte1+2^16*Byte2+2^8*Byte3+Byte4)

Byte 0:

7	6	5	4	3	2	1	0

Bit 7: **Starterkennung** – Wenn dieses Bit gesetzt ist bedeutet dies, dass diese Page der Anfang eines Records ist.

Bit 5 + 6: **Überlaufzähler** – Wird inkrementiert, sobald die letzte Page erreicht wurde und wieder bei der ersten begonnen wird.

Bit 0 .. 4: Zusammen mit dem ganzen zweiten Byte des Header bildet dies den **Startzeiger**. Er zeigt auf die Page, auf welcher der Record begonnen hat. 5 Bit + 8 Bit = 13 Bit

Data x

Ein Datenpaket besteht aus 4 Byte. Es gibt vier Varianten von Datenpaketen:

a) Datenpaket mit Meßgröße, Zeitabstand und Meßwert

Byte	Beschreibung	
0	Bit 7..4: Messgrösse ‚0000' bis ‚1110'	Bit 3..0: Zeitabstand zum vorherigen Datenpaket (0..15s)
1	Messwert 1. Byte	
2	Messwert 2. Byte	
3	Messwert 3. Byte	

Der Messwert ist ein 4 Byte Float, wobei das letzte Byte weggelassen wird.

b) Datenpaket Zeitabstand

Byte	Beschreibung	
0	Bit 7..4: ‚1111'	Bit 3..0: ‚0000'
1	Zeitabstand zum vorhergehenden Datenpaket 1. Byte	
2	Zeitabstand zum vorhergehenden Datenpaket 2. Byte	
3	‚0000 0000'	

Ein Datenpaket mit Zeitabstand wird dann abgespeichert, wenn der Zeitabstand zum vorhergehenden Datenpaket grösser als 15 Sekunden ist. Zeitabstand = 256* 1.Byte + 2.Byte = Bereich (0.. 65'535 s)

c) Datenpaket mit Textinhalt

Byte	Beschreibung	
0	Bit 7..4: ‚1111'	Bit 3..0: ‚0100'
1	8-Bit-ASCII-Zeichen	
2	8-Bit-ASCII-Zeichen	
3	8-Bit-ASCII-Zeichen	

Es ist möglich ein Textpaket mit 3 Zeichen (3 Byte) abzulegen.

d) Leeres Datenpaket

Byte	Beschreibung	
0	Bit 7 .. 4: ‚1111'	Bit 3 .. 0: ‚1111'
1	‚xxxx xxxx'	
2	‚xxxx xxxx'	
3	‚xxxx xxxx'	

Ein leeres Datenpaket kennzeichnet ein leeres Datenpaket oder das Ende eines Records.

Vorgehen für das Lesen des Inhaltsverzeichnisses eines Records

1. Lesen der aktiven Page

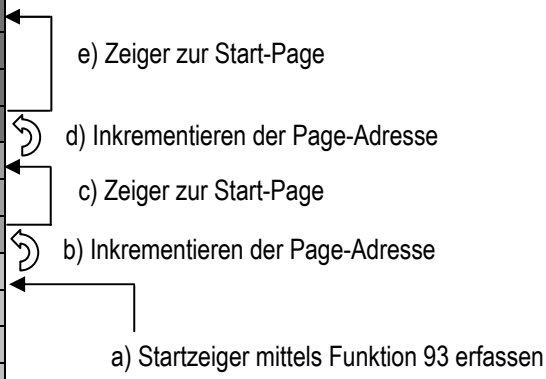
Mittels Funktion 92 Index 1 wird die Adresse der aktuellen Page ausgelesen

2. Lesen der vorherigen Pages

Nun wird die Adresse der aktuellen Page inkrementiert und bei dieser Adresse der Startzeiger überprüft. Der Startzeiger befindet im Header der Page, welcher mittels Funktion 67 ausgelesen werden kann. Mittels Startzeiger kann der Start des Records gefunden werden. Aus der Differenz zwischen momentaner Page und Start-Page lässt sich errechnen wie gross der Record ist. Im Header der Start-Page ist auch die Start-Zeit des Records angegeben.

Durch weiteres inkrementieren der Page-Adresse und demselben Vorgehen wie oben lassen sich nun die weiteren Records finden. Es muss jedoch auf Überläufe geachtet werden.

Page	Beschreibung
0	Start-Page von Record 1
1	Record 1
2	Record 1
3	Record 1
4	Start-Page von Record 2
5	Record 2
6	Record 2
7	Start-Page von Record 3
8	Record 3
9	Record 3
10	Record 3
...	



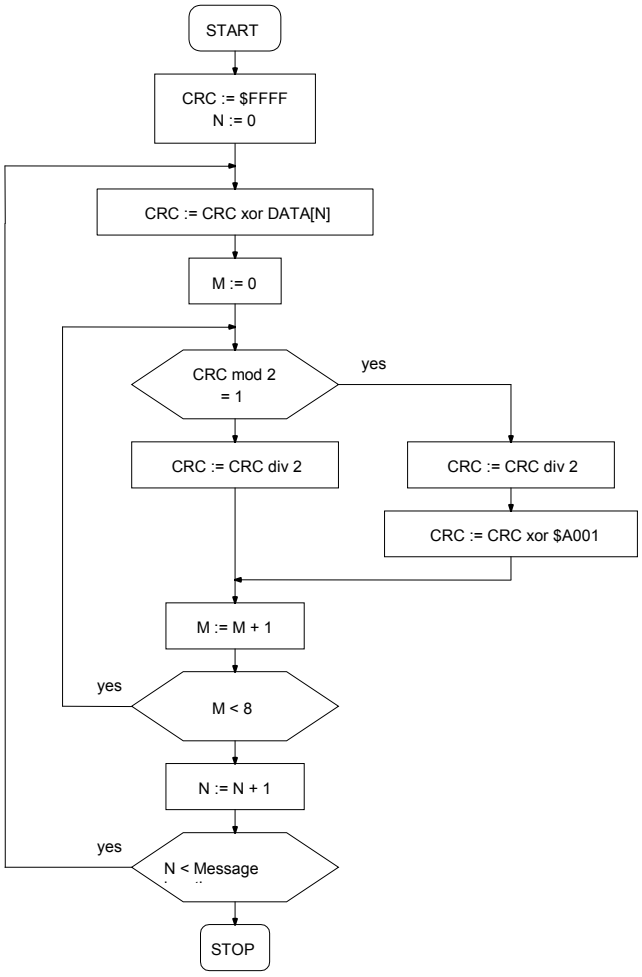
5.4 Berechnung der CRC16-Prüfsumme

Die Prüfsumme kann entweder berechnet, oder aus einer Tabelle abgeleitet werden.
Hier ein Beispiel der CRC16 Berechnung in C:

```
//////////////////////////////////////////////////////////////////////////
// CRC-16 calculation in C
//
// Calculation of CRC-16 checksum over a amount of bytes in the serial buffer.
// The calculation is done without the 2byte from crc16 (recive-mode).
// SC_Buffer[]: Byte-Buffer for the serial interface. Type: unsigned char (8bit)
// SC_Amount : Amount of Bytes which should be transmitted or are received (without CRC16)
//
//////////////////////////////////////////////////////////////////////////
void CalcCRC16(unsigned char* CRC_H, unsigned char* CRC_L)
{
    // locals
    unsigned int Crc;
    unsigned char  n, m, x;

    // initialisation
    Crc= 0xFFFF;
    m= SC_Amount;
    x= 0;

    // loop over all bits
    while(m>0)
    {
        Crc^= SC_Buffer[x];
        for(n=0; n<8; n++)
        {
           if(Crc&1)
           {
               Crc>>= 1;
               Crc^= 0xA001;
           }
           else
               Crc>>= 1;
        }
        m--;
        x++;
    }
    // result
    *CRC_H= (Crc>>8)&0xFF;
    *CRC_L= Crc&0xFF;
}// end CalcCRC16
```



Für die Funktion 48 mit der Geräte-Adresse 250 ergibt sich somit : CRC16_H= 4, CRC16_L= 67.

5.5 Beschreibung des Softwaretreibers (DLL)

5.5.1 Allgemeines

Die zur Verfügung stehende DLL ***DCXc.dll*** ist für KELLER Geräte mit Record-Funktion und Recordspeicher ausgelegt und stellt alle Funktionen (F36/F67/F68/F92/F93) zum Zugreifen auf die Geräte zur Verfügung. Die **S30c.dll** stellt die allgemeinen Funktionen (F30/F31/F48/F66/F69/F73/F100) für KELLER-Geräte zur Verfügung, ohne die Funktionen (F36/F67/F68/F92/F93) die für die Record-Parametrierung oder Zugriff für den Zugriff auf den Recordspeicher benötigt werden. Die DCXc.dll sowie die S30c.dll wurde auf den Betriebssystemen Windows 95, 98, NT und 2000 getestet.

Es stehen für folgende Programmiersprachen Beispiele zur Verwendung dieser DLL zur Verfügung:
LabVIEW / C++ / Delphi / VB / VBA

Für die Übergabe der Parameter an die Funktionen wird die Aufrufskonvention **stdcall** verwendet. Dies bedeutet, dass:
- alle Parameter über den Stack übergeben werden,
- der am Weitesten rechts stehende Parameter als erstes berechnet und übergeben wird, der am Weitesten links stehende als letztes,
- die Funktion selbst die Parameter aus dem Stack löscht.

Wie den nachfolgenden Deklarationen der Funktionen entnommen werden kann, sind viele Variablen mit dem vorangestellten Wort *var* deklariert. Dies bedeutet, dass diese Variablen als Zeiger (Pointer) übergeben werden und nicht als Wert. Die bei der Deklaration verwendeten Typen sind nachfolgend beschrieben:

Typ	Bereich	Format
Byte	0..255	8-Bit ohne Vorzeichen
Word	0..65535	16-Bit ohne Vorzeichen
Smallint	-32768..32767	16-Bit mit Vorzeichen
Longint	-2147483648.. 2147483647	32-Bit mit Vorzeichen
Pbyte		Zeiger auf Byte
Single	+/- $1.5x10^{-45}$..$3.4x10^{38}$	32-Bit

5.5.2 Die Funktionen der (DCXc.dll) DLL

Jede Funktion liefert einen Wert zurück der angibt, ob die gewünschte Funktion erfolgreich ausgeführt wurde oder nicht. Nachfolgend sind alle möglichen Rückgabewerte aufgeführt. Nur bei erfolgreicher Ausführung sind die zurückgegebenen Parameter gültig und dürfen weiterverarbeitet werden.

Rückgabewert		Beschreibung
RS_OK	0	Funktion erfolgreich ausgeführt; Rückgabeparameter sind gültig
RS_EX1	1	Funktion erfolgreich ausgeführt; jedoch Ausnahmefehler 1 aufgetreten
RS_EX2	2	Funktion erfolgreich ausgeführt; jedoch Ausnahmefehler 2 aufgetreten
RS_EX3	3	Funktion erfolgreich ausgeführt; jedoch Ausnahmefehler 3 aufgetreten
RS_EX32	32	Funktion erfolgreich ausgeführt; jedoch Ausnahmefehler 32 aufgetreten
RS_BROADCAST	100	Rundruf
RS_ERROR	-1	allgemeiner Fehler
RS_TXERROR	-2	Sende-Fehler
RS_RXERROR	-3	Empfangs-Fehler im UART
RS_TIMEOUT	-4	keine oder zu wenige Daten empfangen
RS_BADDATA	-5	Daten fehlerhaft (z.B. CRC16 fehlerhaft)

5.5.2.1 Port-Funktionen

Die Geräte werden über eine serielle Schnittstelle mit dem PC verbunden. Die Port-Funktionen **`OpenComPort & CloseComPort`** dienen dem Öffnen und Schliessen dieser Schnittstelle. Zulässig sind die Ports 1 bis 9 (COM1..COM9). Für die Timeout-Zeit sollte die Standardeinstellung verwendet werden (Timeout = 0). Konnte der gewünschte Port geöffnet werden, liefert die Funktion **OpenComPort** den Wert RS_OK zurück, andernfalls RS_ERROR.
Ein geöffneter Port wird bei Beendigung des Programms automatisch geschlossen.

5.5.2.2 Echo-Funktion

Schnittstellenkonverter von KELLER Druckmesstechnik liefern jeweils ein Echo, der vom PC gesendeten Botschaft.
Diese Funktion **`EchoOn`** hat den Standard-Wert 1 (Echo On), um mit den von KELLER ausgelieferten Konvertern arbeiten zu können. Werden andere Konverter verwendet, welche kein Hardware-Echo liefern, muss die Funktion auf 0 = Echo Off gesetzt werden.

5.5.2.3 Protokoll-Funktionen der DCXc.dll / Funktionsdeklarationen

Die folgenden Funktionen kapseln die oben beschriebenen Busfunktionen. Die Reihenfolge der Parameter sind identisch. Die CRC16 Prüfsumme wird in der DLL berechnet und überprüft, deshalb fällt sie weg. Einige Parameter bestehen aus mehreren Bytes. Diese werden zur besseren Übersicht zusammengefasst. Die unterschiedliche Anforderungen a und b der Funktion 95 werden aufgeteilt in zwei Funktionen: F95 und F95val

```
function OpenComPort(intPort, intTimeout: Smallint): Smallint; stdcall; external DLL;

function CloseComPort: Smallint; stdcall; external DLL;

function EchoOn(bEcho: Boolean): Smallint; stdcall; external DLL;

function F30(bteDeviceAddr, bteCoeffNo: Byte; var sinCoeff: Single): Smallint; stdcall; external
DLL;

function F31(bteDeviceAddr, bteCoeffNo: Byte; sinCoeff: Single): Smallint; stdcall; external DLL;

function F36(bteDeviceAddr: Byte; wrdPage: Word; btePos: Byte; bteAmount: Byte;pbteData: PByte):
Smallint; stdcall; external DLL;

function F48(bteDeviceAddr: Byte; var bteClass, bteGroup, bteYear, bteWeek, bteBuffer, bteState:
Byte): Smallint; stdcall; external DLL;

function F66(bteDeviceAddr, bteNewAddr: Byte; var bteActualAddr: Byte): Smallint; stdcall; exter-
nal DLL;

function F67(bteDeviceAddr: Byte; wrdPage: Word; btePos: Byte; bteAmount: Byte; pbteData: PByte):
Smallint; stdcall; external DLL;

function F68(bteDeviceAddr: Byte; wrdPage: Word; bteIndex: Byte; pbteData: PByte): Smallint;
stdcall; external DLL;

function F69(bteDeviceAddr: Byte; var linSN: Longint): Smallint; stdcall; external DLL;

function F73(bteDeviceAddr, bteChannel: Byte; var sinValue: Single; var bteStat: Byte): Smallint;
stdcall; external DLL;

function F92(bteDeviceAddr, bteIndex: Byte; var btePara0, btePara1, btePara2, btePara3, btePara4:
Byte): Smallint; stdcall; external DLL;

function F93(bteDeviceAddr, bteIndex: Byte; btePara0, btePara1, btePara2, btePara3,btePara4:
Byte): Smallint; stdcall; external DLL;

function F95(bteDeviceAddr, bteCmd: Byte): Smallint; stdcall; external DLL;

function F95val(bteDeviceAddr, bteCmd: Byte; sinVal: Single): Smallint; stdcall; external DLL;

function F100(bteDeviceAddr, bteIndex: Byte; var btePara0, btePara1, btePara2, btePara3, btePara4:
Byte): Smallint; stdcall; external DLL;
```

5.6 Support

Für die Implementierung unter Windows steht eine DLL mit diversen Referenz-Implemetationen zur Verfügung.

Zur Konfiguration und zum Auslesen von bis zu 16 Geräten steht Ihnen kostenlos die Software READ30 oder CCS30 zur Verfügung.

Alle Informationen verfügbar unter:
http://www.keller-druck.com

KELLER AG für Druckmesstechnik
St. Gallerstrasse 119 • CH-8404 Winterthur
Tel: ++41 52 235 25 25
http://www.keller-druck.com

5.7