



# USB-TTL-32 / USB-TTL-64

Hardware-Beschreibung

2023 September

# INDEX

<b><u>1. Einleitung</u></b>	<b>9</b>
1.1. Vorwort	10
1.2. Kundenzufriedenheit	10
1.3. Kundenresonanz	10
1.4. Kurzbeschreibung	11
1.5. Lieferumfang	12
<b><u>2. Inbetriebnahme</u></b>	<b>13</b>
2.1. Schritt 1 - Sicherheitshinweise	14
2.2. Schritt 2 - Anschluss der Stromversorgung	15
2.3. Schritt 3 - Verbinden mit dem PC oder Netzwerk	15
2.3.1. Verbindung via USB	15
2.4. Schritt 4 - Installation der Software und Treiber	16
2.5. Schritt 5 - Funktionstest	17
<b><u>3. Hardware Beschreibung</u></b>	<b>18</b>
3.1. Allgemeine technische Daten	19
3.1.1. USB-TTL-32	19
3.1.2. USB-TTL-64	20
3.2. Übersichtsbilder	21
3.2.1. Übersichtsbild eines USB-TTL-32	21
3.2.2. Übersichtsbild eines USB-TTL-64	22
3.3. Schnittstellen	23
3.3.1. USB	23
3.4. LEDs	24
3.4.1. Definition der LEDs	24
3.5. Pinbelegung	25
3.5.1. J1 - Pinbelegung USB-TTL-I/O 0-31	25
3.5.2. J2 - Pinbelegung USB-TTL-I/O 32-63	27
3.6. Blockschaltbilder	29

# INDEX

3.6.1. Blockschaltbild eines USB-TTL-32	29
3.6.2. Blockschaltbild eines USB-TTL-64	29
3.7. Anschlussbeispiele	30
3.7.1. Anschlussbeispiel eines USB-TTL-32 (Input)	30
3.7.2. Anschlussbeispiel eines USB-TTL-32 (Output)	31
3.8. Spannungspegel der TTL-I/O's konfigurieren	32
<b>4. Software Beschreibung</b>	<b>33</b>
4.1. Benutzen unserer Produkte	34
4.1.1. Ansteuerung über unsere DELIB Treiberbibliothek	34
4.1.2. Ansteuerung über mitgelieferte Testprogramme	34
4.1.3. Ansteuerung auf Protokollebene	35
4.1.4. DELIB CLI (command-line interface) für Windows	36
4.1.4.1. Konfiguration des DELIB CLI	38
4.1.4.2. DELIB CLI Beispiele	39
4.1.5. Ansteuerung über grafische Anwendungen	43
4.1.5.1. LabVIEW	43
4.1.5.2. ProfiLab	43
4.1.5.3. Licht24 Pro	44
4.1.6. Einbinden der DELIB in Programmiersprachen	45
4.1.6.1. Einbinden der DELIB in Visual-C/C++	45
4.1.6.2. Einbinden der DELIB in Visual-C/C++ (Visual Studio 2015)	47
4.1.6.3. Einbinden der DELIB in Visual-C#	50
4.1.6.4. Einbinden der DELIB in Delphi	51
4.1.6.5. Einbinden der DELIB in Visual-Basic (VB)	52
4.1.6.6. Einbinden der DELIB in Visual-Basic.NET (VB.NET)	53
4.1.6.7. Einbinden der DELIB in MS-Office (VBA)	54
4.1.6.8. Einbinden der DELIB in LabVIEW	56
4.1.6.8.1. Einbinden der DELIB in LabVIEW	56
4.1.6.8.2. Verwendung der VIs in LabVIEW	65
4.1.6.8.3. Setzen der Modul-ID in LabVIEW	67
4.1.6.9. Einbinden der DELIB in Java	69

# INDEX

4.2. DELIB Treiberbibliothek	70
4.2.1. Übersicht	71
4.2.1.1. Unterstützte Programmiersprachen	72
4.2.1.2. Unterstützte Betriebssysteme	73
4.2.1.3. SDK-Kit für Programmierer	73
4.2.2. DELIB Setup	74
4.2.3. DELIB Configuration Utility	80
4.2.3.1. Einführung	80
4.2.3.2. Neue Konfiguration erstellen oder vorhandene Konfiguration bearbeiten	81
4.2.3.2.1. Modul Konfiguration USB	82
4.2.3.2.1.1, Beispiel zur Konfiguration identischer USB-Module	84
4.2.3.3. Modul testen	89
4.2.3.4. Debug Optionen einstellen	93
4.2.4. Benutzung des Modulselectors	94
4.2.4.1. via USB	94
4.2.4.2. Modul Info	95
4.2.5. DELIB Module Config	97
4.2.5.1. Modul Konfigurationen	97
4.2.5.1.1. Modul-Infoseite	98
4.2.5.2. I/O-Test	99
4.2.5.2.1. Timeout Test-Funktion	99
4.2.5.2.2. Digital In	100
4.2.5.2.3. Digital Out	101
4.2.6. DELIB Module Demo	103
4.2.6.1. Auswahl des Moduls	104
4.2.6.2. Allgemein	105
4.2.6.2.1. Module Info	107
4.2.6.3. Digital Input	108
4.2.6.4. Digital Output	109
4.2.7. DT-Flasher	110
4.2.7.1. Über DEDITEC-Firmware	111
4.2.7.2. Auswahl des Moduls	111



# INDEX

4.2.7.3. Firmware Update durchführen	112
4.2.7.3.1. Flash-Files manuell aktualisieren	114
<b>4.3. DELIB Sample Sources (Windows Programmbeispiele)</b>	<b>115</b>
4.3.1. Installation DELIB Sample Sources	116
4.3.2. Benutzung der DELIB Sample Sources	120
4.3.2.1. Schritt 1 - Produktauswahl	121
4.3.2.2. Schritt 2 - Kategorieauswahl	122
4.3.2.3. Schritt 3 - Programmiersprachenauswahl	123
4.3.2.4. Schritt 4 - Quellcode	124
<b>4.4. DELIB für Linux</b>	<b>127</b>
4.4.1. Verwenden der DELIB Treiberbibliothek für Linux	130
4.4.1.1. Delib USB-Sample in Linux	130
4.4.1.2. Delib ETH-Sample in Linux	133
4.4.2. DELIB CLI (command-line interface) für Linux	137
4.4.2.1. Konfiguration des DELIB CLI	140
4.4.2.2. DELIB CLI Beispiele	143
<b><u>5. DELIB API Referenz</u></b>	<b>145</b>
5.1. Verfügbare DEDITEC Modul IDs	146
5.2. Verwaltungsfunktionen	149
5.2.1. DapiOpenModule	149
5.2.2. DapiCloseModule	150
5.2.3. DapiGetDELIBVersion	150
5.2.4. DapiSpecialCMDGetModuleConfig	151
5.2.5. DapiOpenModuleEx	154
5.2.6. DapiScanAllModulesAvailable	156
5.3. Fehlerbehandlung	157
5.3.1. DapiGetLastError	157
5.3.2. DapiGetLastErrorText	158
5.3.3. DapiClearLastError	159
5.3.4. DapiGetLastErrorByHandle	160
5.3.5. DapiClearLastErrorByHandle	161

# INDEX

5.4. Digitale Eingänge lesen	162
5.4.1. DapiDIGet1	162
5.4.2. DapiDIGet8	162
5.4.3. DapiDIGet16	163
5.4.4. DapiDIGet32	164
5.4.5. DapiDIGet64	165
5.4.6. DapiDIGetFF32	166
5.4.7. DapiDIGetCounter	167
5.4.8. DapiSpecialCounterLatchAll	168
5.4.9. DapiSpecialCounterLatchAllWithReset	169
5.4.10. DapiSpecialDIFilterValueSet	170
5.4.11. DapiSpecialDIFilterValueGet	171
5.4.12. Dapi_Special_DI_FF_Filter_Value_Set	172
5.4.13. Dapi_Special_DI_FF_Filter_Value_Get	173
5.5. Digitale Ausgänge verwalten	174
5.5.1. DapiDOSet1	174
5.5.2. DapiDOSet8	174
5.5.3. DapiDOSet16	175
5.5.4. DapiDOSet32	176
5.5.5. DapiDOSet64	177
5.5.6. DapiDOSet1_WithTimer	178
5.5.7. DapiDOReadback32	179
5.5.8. DapiDOReadback64	179
5.5.9. DapiDOSetBit32	180
5.5.10. DapiDOClrBit32	181
5.6. TTL-Ein-/Ausgangs Richtungen setzen mit DapiSpecialCommand	182
5.6.1. DAPI_SPECIAL_CMD_SET_DIR_DX_1	182
5.6.2. DAPI_SPECIAL_CMD_SET_DIR_DX_8	184
5.6.3. DAPI_SPECIAL_CMD_GET_DIR_DX_8	185
5.7. Ausgabe-Timeout verwalten	186
5.7.1. DapiSpecialCMDTimeout	186

# INDEX

5.7.1.1. DapiSpecialTimeoutSetValueSec	189
5.7.1.2. DapiSpecialTimeoutActivate	190
5.7.1.3. DapiSpecialTimeoutActivateAutoReactivate	191
5.7.1.4. DapiSpecialTimeoutActivateSecureOutputs	192
5.7.1.5. DapiSpecialTimeoutDeactivate	193
5.7.1.6. DapiSpecialTimeoutGetStatus	194
5.7.1.7. DapiSpecialTimeoutDoValueMaskWRSet32	195
5.7.1.8. DapiSpecialTimeoutDoValueMaskRDSet32	196
5.7.1.9. DapiSpecialTimeoutDoValueMaskWRClear32	197
5.7.1.10. DapiSpecialTimeoutDoValueMaskRDClear32	198
5.7.1.11. DapiSpecialTimeoutDoValueLoadDefault	199
<b>5.8. Testfunktionen</b>	<b>200</b>
5.8.1. DapiPing	200
<b>5.9. Register Schreib-Befehle</b>	<b>201</b>
5.9.1. DapiWriteByte	201
5.9.2. DapiWriteWord	202
5.9.3. DapiWriteLong	203
5.9.4. DapiWriteLongLong	204
<b>5.10. Register Lese-Befehle</b>	<b>205</b>
5.10.1. DapiReadByte	205
5.10.2. DapiReadWord	206
5.10.3. DapiReadLong	207
5.10.4. DapiReadLongLong	208
<b>5.11. Programmier-Beispiel</b>	<b>209</b>
<b>5.12. Delib Übersichtstabelle</b>	<b>211</b>
<b><u>6. Anhang</u></b>	<b>214</b>
6.1. Kontakt / Support	215
6.2. Umwelt und Entsorgung	215
6.3. Revisionen	216
6.4. Urheberrechte und Marken	217

# INDEX

# Einleitung

---



# **1. Einleitung**

## **1.1. Vorwort**

**Wir beglückwünschen Sie zum Kauf eines hochwertigen DEDITEC Produktes!**

Unsere Produkte werden von unseren Ingenieuren nach den heutigen geforderten Qualitätsanforderungen entwickelt. Wir achten bereits bei der Entwicklung auf flexible Erweiterbarkeit und lange Verfügbarkeit.

**Wir entwickeln modular!**

Durch eine modulare Entwicklung verkürzt sich bei uns die Entwicklungszeit und - was natürlich dem Kunden zu Gute kommt - wir verkaufen zu einem fairen Preis!

**Wir sorgen für eine lange Lieferverfügbarkeit!**

Sollten verwendete Halbleiter nicht mehr verfügbar sein, so können wir schneller reagieren. Bei uns müssen meistens nur Module redesigned werden und nicht das gesamte Produkt. Dies erhöht die Lieferverfügbarkeit.

## **1.2. Kundenzufriedenheit**

**Ein zufriedener Kunde steht bei uns an erster Stelle!**

Sollte mal etwas nicht zu Ihrer Zufriedenheit sein, wenden Sie sich einfach per Telefon oder Mail an uns.

Wir kümmern uns darum!

## **1.3. Kundenresonanz**

Die besten Produkte wachsen mit unseren Kunden. Für Anregungen oder Vorschläge sind wir jederzeit dankbar.

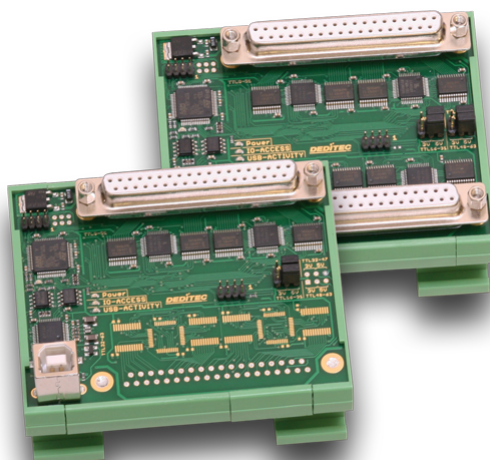
## 1.4. Kurzbeschreibung

Das USB-TTL-32 und das USB-TTL-64 finden dort Einsatz, wo über den USB-Bus direkt auf TTL-Ein- bzw. Ausgänge zugegriffen werden soll. Im Bereich der PC-Messtechnik eignen sich diese Module hervorragend für den Aufbau umfangreicher Automatisierungsprojekte, Steuerungsaufgaben oder Messverfahren. Durch den kostengünstigen Anschaffungspreis, eignen sich diese Module besonders gut für Einsteiger.

Digitale Datensignale können erfasst oder ausgegeben werden. Die Bereitstellung und Verarbeitung dieser Signale geschieht durch die Kundenapplikation auf dem Steuer PC. Als einfache Programmierschnittstelle bietet sich hier bspw. unsere DELIB API an, oder eine direkte Kommunikation über unser Ethernet-Protokoll.

Das im Lieferumfang enthaltene Konfigurationstool „DELIB-Module Config“ ermöglicht zusätzlich einen schnellen und unkomplizierten Einstieg bei der Inbetriebnahme.

Das Gehäuse besteht aus einem kompakten Kunststoff Profil und eignet sich zur Montage auf Hutschienen, wie sie typischerweise in Schaltschränken verwendet werden.



## 1.5. Lieferumfang

Folgende Artikel sind im Lieferumfang enthalten:

- USB-TTL-32 oder USB-TTL-64
- USB Kabel 1,5m
- Installations CD mit Handbüchern und Treibern



# Inbetriebnahme

---



## **2. Inbetriebnahme**

### **2.1. Schritt 1 - Sicherheitshinweise**

Bitte machen Sie sich vor der Inbetriebnahme Ihres DEDITEC Produktes mit diesem Handbuch vertraut und lesen Sie sich die nachfolgenden Punkte genau durch:

- Schäden, die durch Nichtbeachten dieser Bedienungsanleitung verursacht werden, führen zum Erlöschen der Gewährleistung bzw. Garantie dieses Produktes. Für Folgeschäden übernehmen wir keinerlei Haftung!
- Für Sach- oder Personenschäden, die durch unsachgemäße Handhabung oder Nichtbeachtung der Sicherheitshinweise entstehen könnten, übernehmen wir keinerlei Haftung!
- Vermeiden Sie ein direktes Berühren elektronischer Bauteile auf der Leiterplatine. Dies könnte zu elektrostatischen Entladungen führen und empfindliche Bauteile zerstören. Entladen Sie sich vorsichtshalber immer vor dem Berühren an einem elektrisch geerdeten Gegenstand.
- Eigenmächtige Umbauten oder technische Änderungen an diesem Produkt sind aus Sicherheits- und Zulassungsgründen (CE) nicht gestattet und führen zum Erlöschen der Gewährleistung bzw. Garantie.
- Betreiben Sie das Modul nicht außerhalb der maximal zulässigen technischen Daten.
- Das Produkt ist nicht für den Betrieb in feuchter oder nasser Umgebung geeignet.

## **2.2. Schritt 2 - Anschluss der Stromversorgung**

Bei unseren reinen USB-Modulen der Starter-Serie, wird das Modul über die USB-Schnittstelle mit der nötigen Versorgungsspannung von +5V versorgt.

Eine externe Versorgung wird nicht benötigt.

## **2.3. Schritt 3 - Verbinden mit dem PC oder Netzwerk**

### **2.3.1. Verbindung via USB**

Verbinden Sie das Modul, mit dem im Lieferumfang enthaltenen USB-Kabel, mit Ihrem PC oder USB-Hub.

Möchten Sie mehrere USB Module gleichzeitig an einem PC anschließen, muss zunächst jedem Modul eine eigene Modulnummer vergeben werden.

Siehe Kapitel → **Beispiel zur Konfiguration identischer USB-Module**

## 2.4. Schritt 4 - Installation der Software und Treiber

### Installation unter Windows:

Um dieses Produkt mit einem Windows basierten PC betreiben zu können, gehen Sie bitte wie folgt vor:

Installieren Sie zuerst die DELIB-Treiberbibliothek für Windows, indem Sie Datei "delib\_install.exe" von der DEDITEC-Treiber CD ausführen. Diese befindet sich Verzeichnis "\zip\delib\delib\_install.exe".

Alternativ können Sie die aktuellste DELIB Version auch von unserer Homepage herunterladen. → <http://www.deditec.de/delib>

### Installation unter Linux:

Um dieses Produkt mit einem Linux basierten PC betreiben zu können, gehen Sie bitte wie folgt vor:

Entpacken Sie das ZIP File "delib-linux.zip" von der DEDITEC Treiber-CD und kopieren Sie sich die delib.dll in Ihr Projektverzeichnis.

Alternativ können Sie die aktuellste DELIB Version auch von unserer Homepage herunterladen. → <https://www.deditec.de/media/zip/delib/delib-linux.zip>

## 2.5. Schritt 5 - Funktionstest

Mit unserem Tool "DELIB-Module Config" können Sie das Modul relativ schnell und einfach und ohne Programmierkenntnisse in Betrieb nehmen und auf dessen Funktionalität überprüfen.

Folgen Sie hierfür den Anweisungen im Kapitel "**DELIB-Module Config**".

# Hardware Beschreibung

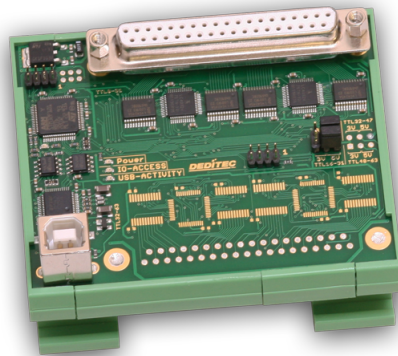
---



## 3. Hardware Beschreibung

### 3.1. Allgemeine technische Daten

#### 3.1.1. USB-TTL-32



##### Elektrische Daten:

Versorgungsspannung:	+5V (über USB-Bus)
Leistungsaufnahme:	max. 5W

##### Umgebung:

Umgebungstemperatur:	+10..+50 °C
Luftfeuchtigkeit:	90 %
Betauung:	Nicht erlaubt

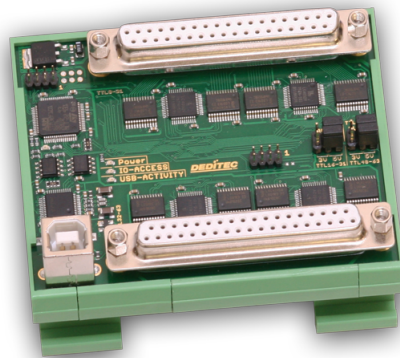
##### Mechanik:

Abmessungen in mm (LxBxH):	77 x 90 x 42
Befestigung:	Hutschiene TS 35 x 7,5 mm
Steckverbinder:	1*37 polige D-Sub Buchse

##### Ein-/Ausgänge:

Anzahl:	32 TTL I/Os
TTL-I/O:	In 8er-Blöcke als Ein- und Ausgang einstellbar
TTL-Pegel:	3,3V bis 5V über Jumper einstellbar 1,5V bis 5V über externe Pegeleinspeisung
Ausgangsstrom:	max. 5mA/Kanal

### 3.1.2. USB-TTL-64



#### Elektrische Daten:

Versorgungsspannung:	+5V (über USB-Bus)
Leistungsaufnahme:	max. 5W

#### Umgebung:

Umgebungstemperatur:	+10..+50 °C
Luftfeuchtigkeit:	90 %
Betauung:	Nicht erlaubt

#### Mechanik:

Abmessungen in mm (LxBxH):	77 x 90 x 42
Befestigung:	Hutschiene TS 35 x 7,5 mm
Steckverbinder:	2*37 polige D-Sub Buchse

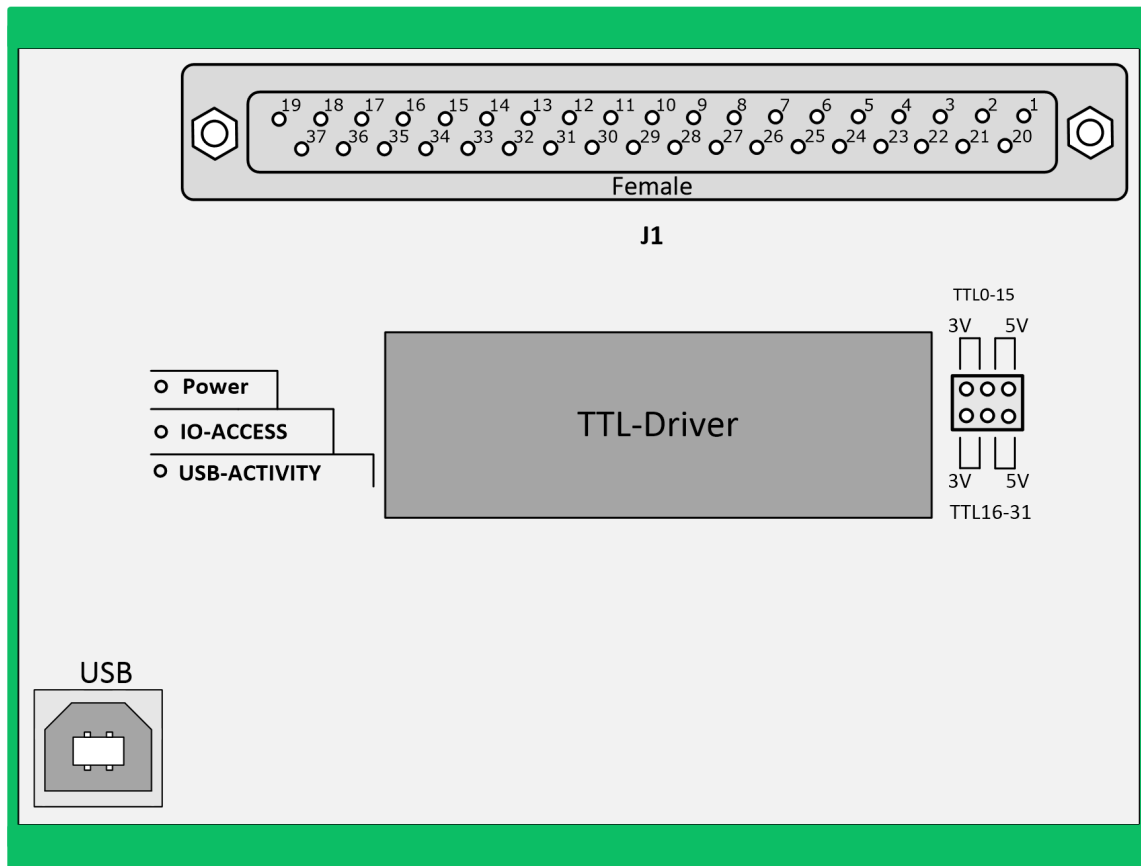
#### Ein-/Ausgänge:

Anzahl:	64 TTL I/Os
TTL-I/O:	In 8er-Blöcke als Ein- und Ausgang einstellbar
TTL-Pegel:	3,3V bis 5V über Jumper einstellbar 1,5V bis 5V über externe Pegeleinspeisung
Ausgangsstrom:	max. 5mA/Kanal

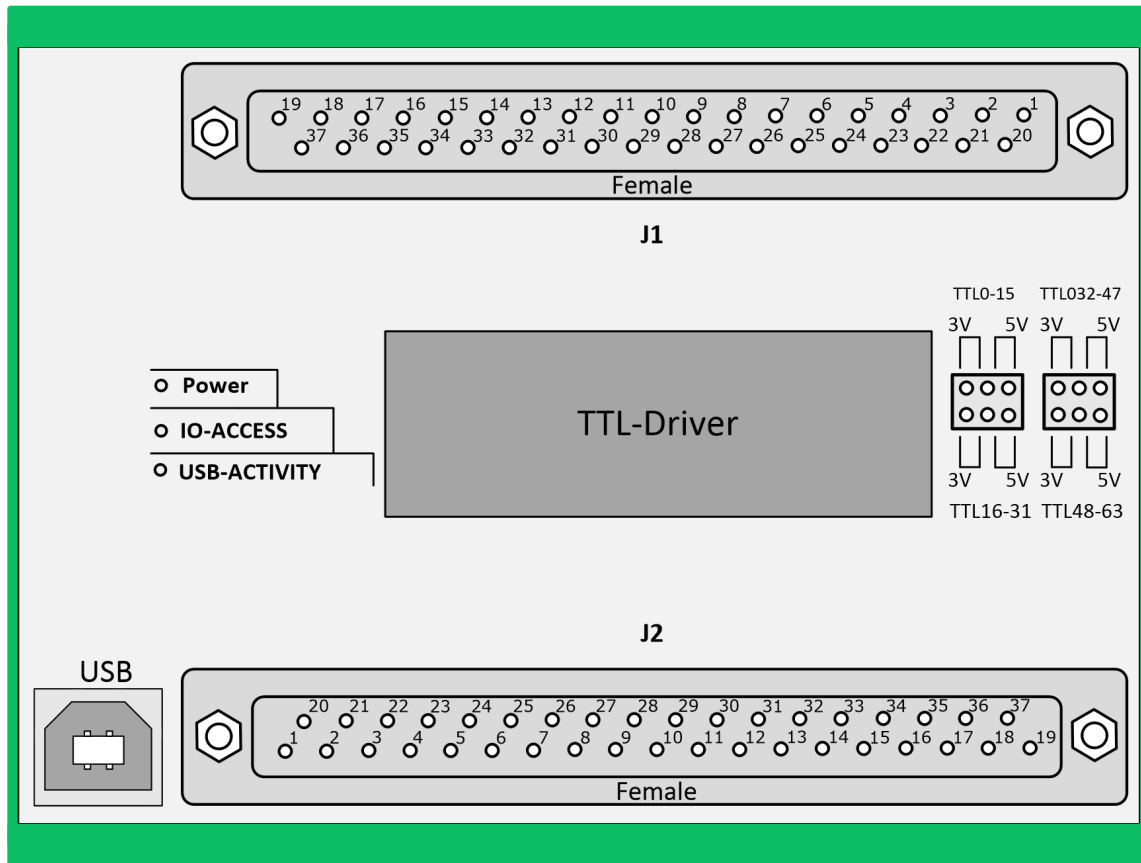


## 3.2. Übersichtsbilder

### 3.2.1. Übersichtsbild eines USB-TTL-32



### 3.2.2. Übersichtsbild eines USB-TTL-64



### 3.3. Schnittstellen

Die Standard Schnittstelle auf dem Modul ist USB.

#### 3.3.1. USB

##### Technische Daten:

Standard:	USB 1.1 / USB 2.0
Verbindungsaufbau:	USB Kabel Typ A auf Typ B
Zugriffszeit PC auf Modul*:	4,06 ms**

\* Berechnet mit 1000 Zugriffen auf das Modul über die DELIB Treiberbibliothek mit dem Befehl DapiDoSet32

\*\* durchschnittliche Zeit für 32-Bit Zugriffe

## **3.4. LEDs**

### **3.4.1. Definition der LEDs**

**LED Power:**

Leuchtet wenn sich das Modul in Betrieb befindet

**LED IO-Access:**

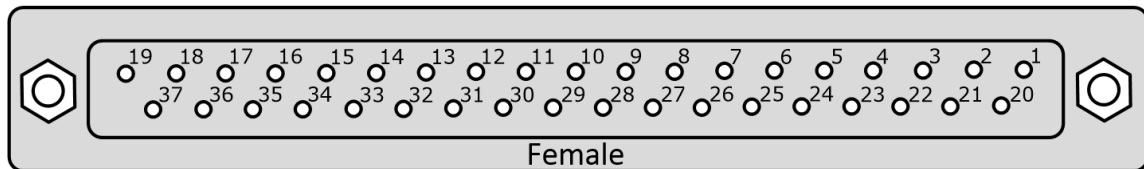
Signalisiert den Zugriff auf die TTL-I/Os

**LED USB-Activity:**

Zeigt an, dass eine Signalverarbeitung über den USB-Bus stattfindet

## 3.5. Pinbelegung

### 3.5.1. J1 - Pinbelegung USB-TTL-I/O 0-31



Pin	TTL I/O	Pin	TTL I/O
1	I/O 16	2	I/O 18
3	I/O 20	4	I/O 22
5	I/O 24	6	I/O 26
7	I/O 28	8	I/O 30
9	I/O 0	10	I/O 2*
11	I/O 4*	12	I/O 6
13	I/O 8	14	I/O 10
15	I/O 12	16	I/O 14
17	VIN 0-15	18	GND
19	GND	20	I/O 17
21	I/O 19	22	I/O 21
23	I/O 23	24	I/O 25
25	I/O 27	26	I/O 29
27	I/O 31	28	I/O 1
29	I/O 3	30	I/O 5
31	I/O 7	32	I/O 9

Pin	TTL I/O	Pin	TTL I/O
33	I/O 11	34	I/O 13
35	I/O 15	36	VIN 16-31
37	GND		

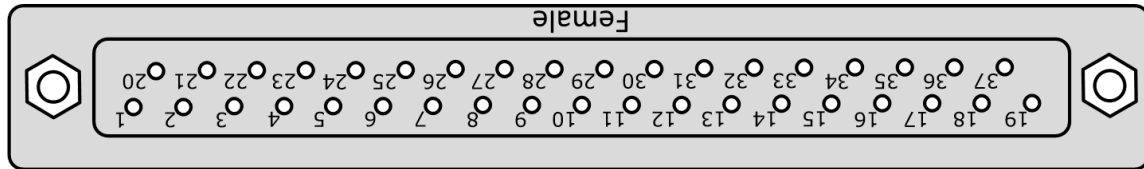
**Anmerkung:**

Der VIN-Pin dient dazu, eine eigene Spannung an die TTL-I/O's des Moduls zu legen. Diese Spannung kann zwischen 1,8V und 5V liegen.

Sind die TTL Pegel standardmäßig auf 3,3V oder 5V via Jumperbelegung eingestellt, dient der VIN-Pin als Ausgang.

\*: Bis einschließlich Revision 1.2 sind diese Pins vertauscht. In späteren Revisionen ist dieser Fehler behoben.

### 3.5.2. J2 - Pinbelegung USB-TTL-I/O 32-63



Pin	TTL I/O	Pin	TTL I/O
1	I/O 48	2	I/O 50
3	I/O 52	4	I/O 54
5	I/O 56	6	I/O 58
7	I/O 60	8	I/O 62
9	I/O 32	10	I/O 34*
11	I/O 36*	12	I/O 38
13	I/O 40	14	I/O 42
15	I/O 44	16	I/O 46
17	VIN 32-47	18	GND
19	GND	20	I/O 49
21	I/O 51	22	I/O 53
23	I/O 55	24	I/O 57
25	I/O 59	26	I/O 61
27	I/O 63	28	I/O 33
29	I/O 35	30	I/O 37
31	I/O 39	32	I/O 41
33	I/O 43	34	I/O 45

Pin	TTL I/O	Pin	TTL I/O
35	I/O 47	36	VIN 48-63
37	GND		

**Anmerkung:**

Der VIN-Pin dient dazu, eine eigene Spannung an die TTL-I/O's des Moduls zu legen. Diese Spannung kann zwischen 1,8V und 5V liegen.

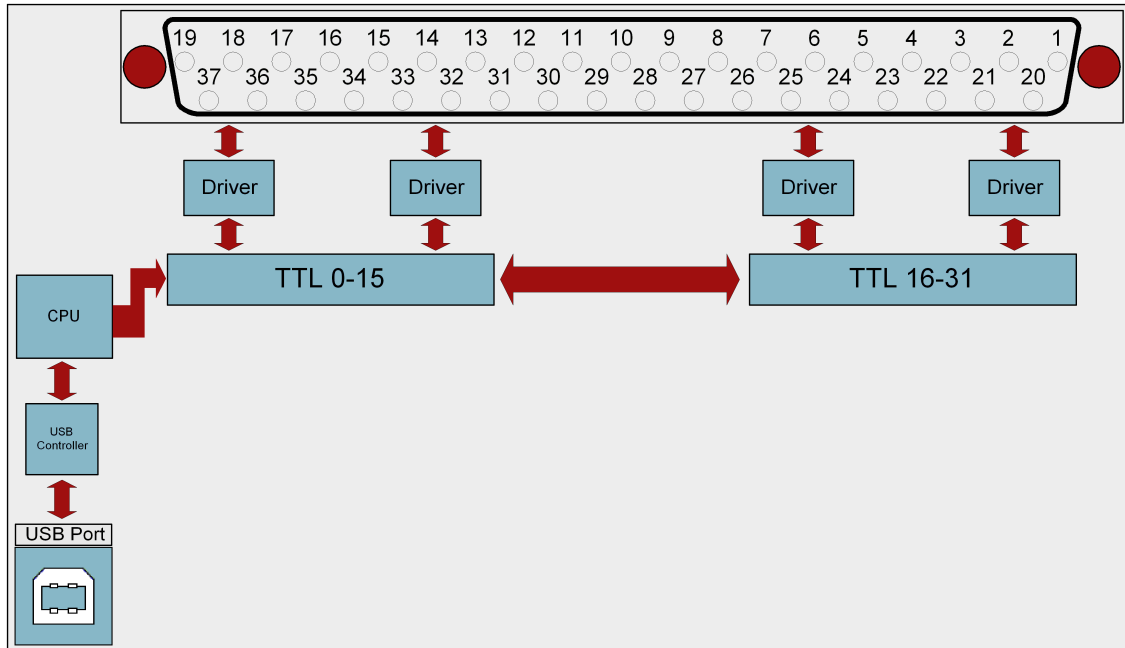
Sind die TTL Pegel standardmäßig auf 3,3V oder 5V via Jumperbelegung eingestellt, dient der VIN-Pin als Ausgang.

\*: Bis einschließlich Revision 1.2 sind diese Pins vertauscht. In späteren Revisionen ist dieser Fehler behoben.

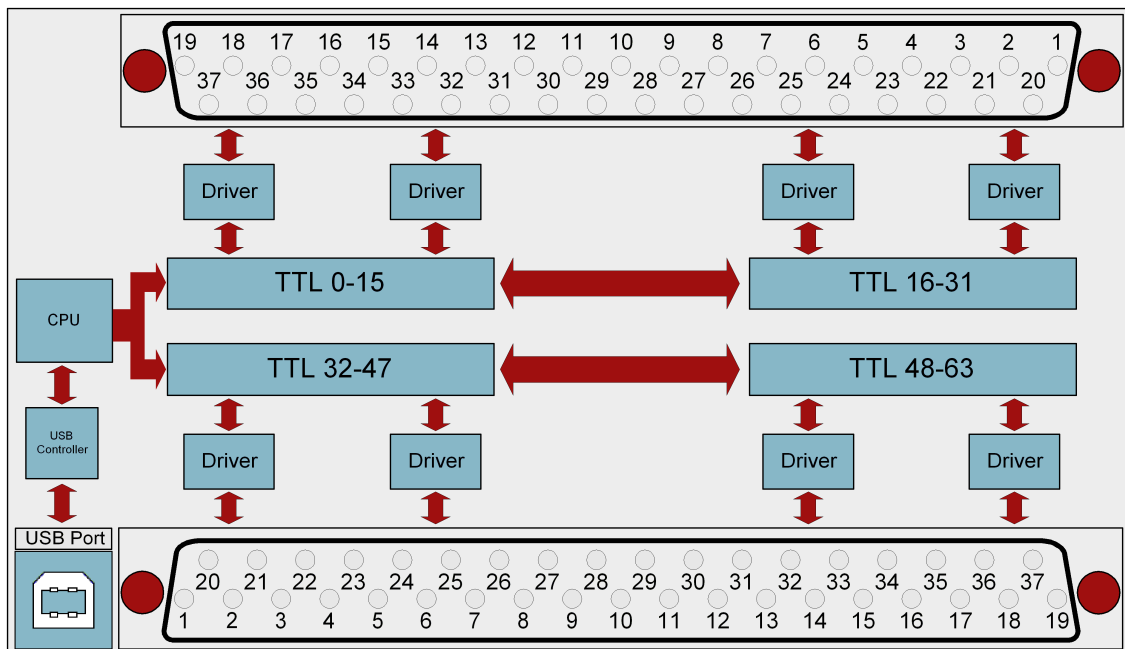


### 3.6. Blockschaltbilder

#### 3.6.1. Blockschaltbild eines USB-TTL-32

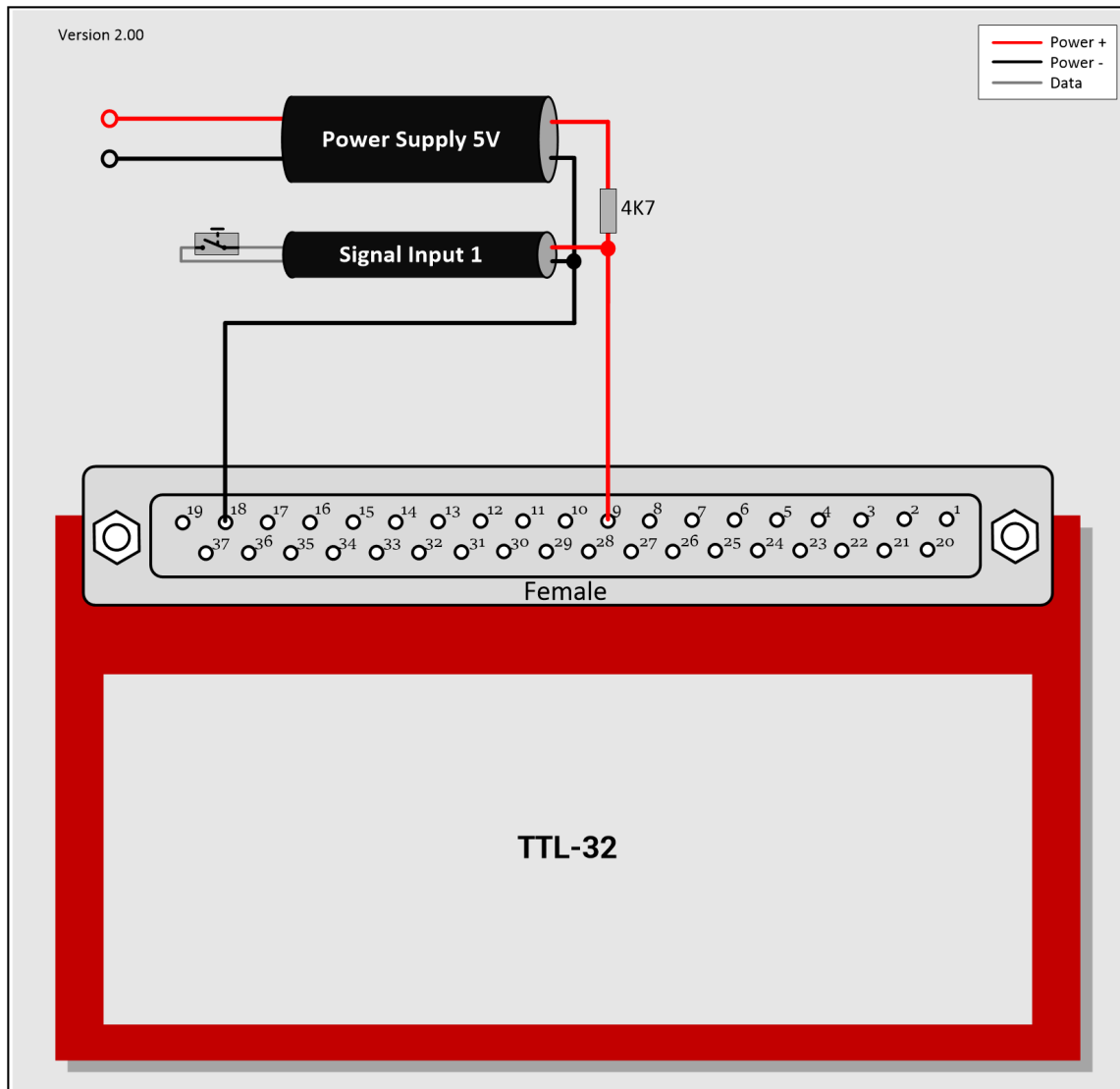


#### 3.6.2. Blockschaltbild eines USB-TTL-64

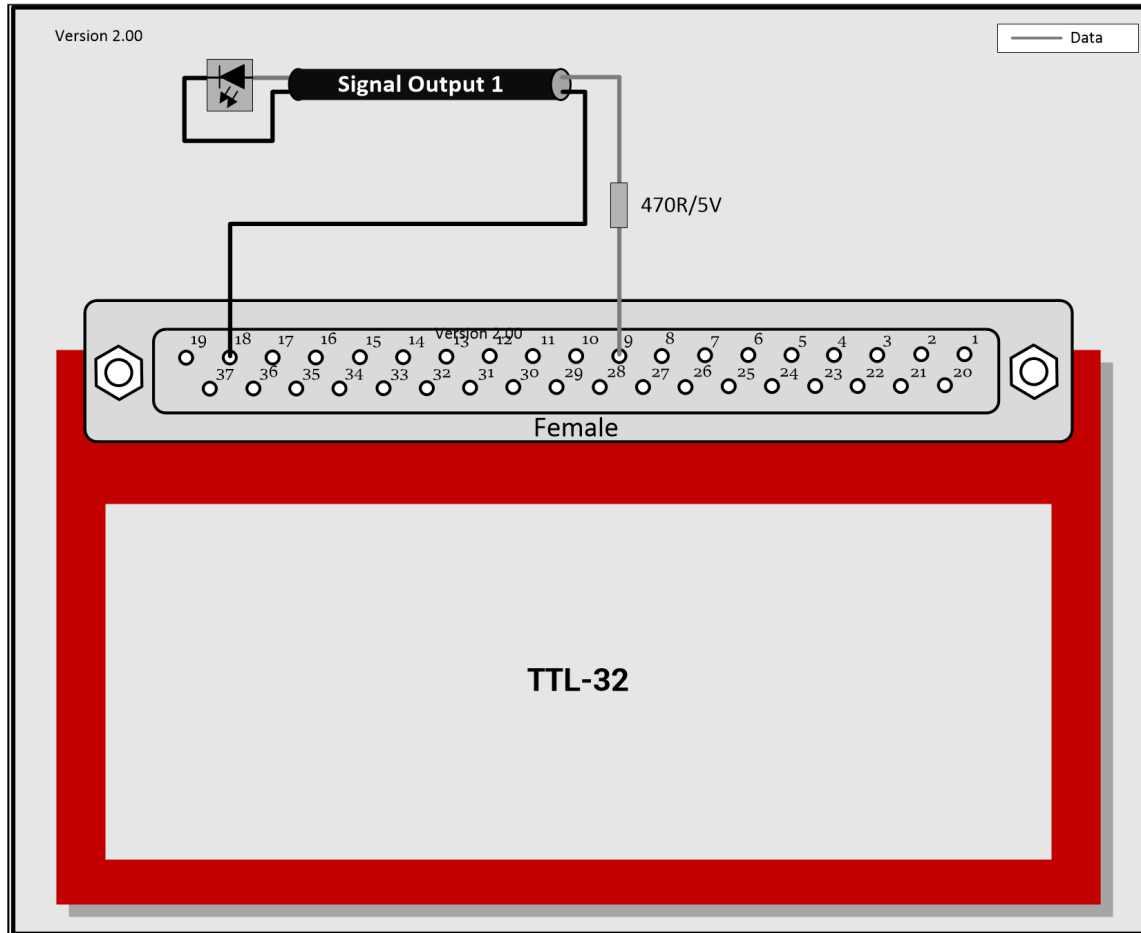


## 3.7. Anschlussbeispiele

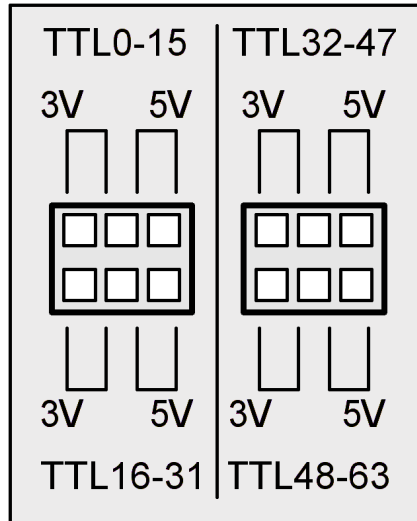
### 3.7.1. Anschlussbeispiel eines USB-TTL-32 (Input)



### 3.7.2. Anschlussbeispiel eines USB-TTL-32 (Output)



### 3.8. Spannungspegel der TTL-I/O's konfigurieren



#### TTL Pegel von 1,8V bis 5V:

Standardmäßig können Sie TTL Pegel wahlweise von 3,3V oder 5V über eine jeweilige Jumperbelegung einstellen.

Entfernen Sie die Jumper auf dem Modul, so können Sie eine eigene Spannung von 1,8V bis 5V an die TTL-I/O's des Moduls anlegen.

Wenn Sie eine eigene Spannung anlegen möchten, geschieht dies über den VIN-Pin (siehe Kapitel **Pinbelegung**).

Die TTL-I/O's des Moduls werden dabei in 16er Blöcken konfiguriert.

# **Software Beschreibung**



**IV**

## **4. Software Beschreibung**

### **4.1. Benutzen unserer Produkte**

#### **4.1.1. Ansteuerung über unsere DELIB Treiberbibliothek**

Im Lieferumfang unserer DELIB Treiberbibliothek ist die DELIB-API und diverse Programme zur Konfiguration Test unserer Produkte enthalten.

Die API bietet Ihnen Zugriff auf alle Funktionen die Sie zur Kommunikation mit unseren Produkten benötigen.

Im Kapitel **DELIB API Referenz** finden Sie alle Funktionen unserer Treiberbibliothek erklärt und mit Anwendungsbeispielen versehen.

#### **4.1.2. Ansteuerung über mitgelieferte Testprogramme**

Mit unserem DELIB Module Config können Sie unsere Steuer- & Regelungstechnik Produkte ohne großen Konfigurationsaufwand auf Funktionalität testen.

Für ausführliche Informationen siehe Kapitel **DELIB\_Module\_Config**.

### 4.1.3. Ansteuerung auf Protokollebene

Für Produkte mit Ethernet-, CAN- oder Serieller-Schnittstelle bieten wir Ihnen unsere offenen Protokolle an.

Diese Protokolle können ohne unsere DELIB Treiberbibliothek auf Geräten mit entsprechender Schnittstelle verwendet werden. Der Weg über unsere Protokolle sind Betriebssystem unabhängig.

Unser Handbuch, Protokolle & Registerbelegung finden Sie hier:

Download PDF:

[http://www.deditec.de/pdf/manual\\_d\\_deditec\\_communication\\_protocols.pdf](http://www.deditec.de/pdf/manual_d_deditec_communication_protocols.pdf)

Online HTML-Manual:

[http://manuals.deditec.de/de/manual\\_deditec\\_communication\\_protocols/index.html](http://manuals.deditec.de/de/manual_deditec_communication_protocols/index.html)

Dieses Handbuch bietet eine komplette Übersicht über die benötigten Registeradressen unserer Module sowie den Aufbau der verschiedenen Kommunikationsprotokolle.

#### 4.1.4. DELIB CLI (command-line interface) für Windows

Da in manchen Programmiersprachen (wie zum Beispiel PHP) keine DLLs eingebunden werden können, gibt es hierzu ein extra Kommandozeilen-Befehl, der direkt aus dem Programm (mit den entsprechenden Parametern) heraus aufgerufen werden kann.

Der DELIB CLI Befehl für Windows befindet sich nach der Installation der DELIB Treiberbibliothek im Verzeichnis C:\Programme\DEDITEC\DELIB\programs\cli\.

##### Definition (Windows)

*delib\_cli command channel [value | unit ["nounit"]]*

**Hinweis:** Die einzelnen Parameter werden nur durch ein Leerzeichen getrennt. Groß und Kleinschreibung wird hierbei nicht beachtet.

##### Parameter

Befehl	Kanal	Wert		unit	nounit
di1	0, 1, 2, ...	-		hex	nounit
di8	0, 8, 16, ...				
di16					
di32					
ff	0, 32, ...	-		hex	nounit
do1	0, 1, 2, ...	0/1 (1-Bit Befehl)		-	-
do8	0, 8, 16, ...	8-Bit Wert	(Bit 0 für Kanal 1, Bit 1 für Kanal 2, ...)		
do16		16-Bit Wert			
do32		32-Bit Wert			



Befehl	Kanal	Wert	unit	nounit
ai	0, 1, 2, ...	-	hex, volt, mA	nounit
ao	0, 1, 2, ...	Ganz oder Hexadezimalzahl (beginnend mit 0x).	-	-

#### Return-Wert

##### Zustand der gelesenen digitalen Eingänge

In Kombination mit Parameter unit "hex" wird der Zustand als hex gelesen

##### Zustand der FlipFlips der digitalen Eingänge

In Kombination mit Parameter unit "hex" wird der Zustand als hex gelesen

##### Zustand der gelesenen analogen Eingänge

In Kombination mit Parameter unit "hex" wird der Zustand als hex gelesen

In Kombination mit Parameter unit "volt" wird die Spannung gelesen

In Kombination mit Parameter unit "mA" wird der Strom gelesen

#### 4.1.4.1. Konfiguration des DELIB CLI

Vor der ersten Verwendung des DELIB CLI muss die "delib\_cli.cfg" mit einem Texteditor bearbeitet werden.

##### Konfiguration unter Windows

Unter Windows befindet sich die "delib\_cli.cfg" nach der Installation der DELIB-Treiberbibliothek im Verzeichnis "C:\Programme\DEDITEC\DELIB\programs\cli".

##### Inhalt der "delib\_cli.cfg":

```
moduleID=14;  
moduleNR=0;  
RO-ETH_ipAddress=192.168.1.11;
```

##### moduleID

Als moduleID muss die entsprechende Nummer der eingesetzten Hardware eingetragen werden.

Diese Nummer kann der "delib.h" entnommen werden.

Unter Windows finden Sie diese im Verzeichnis C:\Programme\DEDITEC\DELIB\include\.

##### moduleNR

Die moduleNR wird im DELIB Configuration Utility vergeben.

Diese Nummer dient zur Identifizierung identischer Hardware.

Der Standardwert ist die 0.

##### RO-ETH\_ipAddress

Dieser Eintrag wird ausschließlich für die Verbindung zu unseren ETH Modulen benötigt.

Die IP-Adresse der ETH Module können über das DELIB Configuration Utility sowie über die Weboberfläche des Moduls eingestellt werden.

#### 4.1.4.2. DELIB CLI Beispiele

##### **Digitale Ausgänge**

```
delib_cli DO1 17 1
```

→ schaltet das 18. digitale Relais an

```
delib_cli DO1 3 0
```

→ schaltet das 4. digitale Relais aus

```
delib_cli DO8 0 255
```

→ schaltet die digitalen Relais 1 bis 8 an

```
delib_cli DO16 0 0
```

→ schaltet die digitalen Relais 1 bis 16 aus

```
delib_cli DO16 16 65535
```

→ schaltet die digitalen Relais 17 bis 32 an

```
delib_cli DO32 0 4294967295
```

→ schaltet die digitalen Relais 1 bis 32 an

### Digitale Eingänge

```
delib_cli DI1 3
```

Beispiel eines Rückgabewertes: 1

→ lese den Zustand des 4. digitalen Eingangs und gebe ihn zurück

```
delib_cli DI8 0 hex
```

Beispiel eines Rückgabewertes: **0xC8**

(auf den Kanälen 4, 7 und 8 liegt ein Signal an)

→ lese den Wert von digitalen Eingang 1-8 als hexadezimalzahl

```
delib_cli DI16 0 hex
```

Beispiel eines Rückgabewertes: **0xE0C0**

(auf den Kanälen 7,8, 14 ,15 und 16 liegt ein Signal an)

→ lese den Wert von digitalen Eingang 1-16 als hexadezimalzahl

Alternativ kann das Argument "nunit" an alle zu formatierenden Ausgabeanfragen wie folgendermaßen angehängen werden:

```
delib_cli DI8 0 hex nunit
```

Beispiel eines Rückgabewertes: **FF**

(auf den Kanälen 1-8 liegt ein Signal an)

→ lese den Wert von digitalen Eingang 1-8 als hexadezimalzahl

```
delib_cli FF 0
```

Beispiel eines Rückgabewertes: **192**

(auf den Kanälen 7 und 8 wurde eine Zustandsänderung erkannt)

→ lese den Wert der FlipFlops der digitalen Eingänge 1-32

```
delib_cli FF 32
```

Beispiel eines Rückgabewertes: **65535**

(auf den Kanälen 33 bis 64 wurde eine Zustandsänderung erkannt)

→ lese den Wert der FlipFlops der digitalen Eingänge 33-64

```
delib_cli FF 0 hex
```

Beispiel eines Rückgabewertes: **0xD00**

(auf Kanälen 9, 11 und 12 wurde eine Zustandsänderung erkannt)

→ lese den Wert der FlipFlops der digitalen Eingänge 1-32 als hexadezimalzahl

### **Analoge Ausgänge**

`delib_cli AO 7 4711`

→ setzt den dezimalen Wert 4711 auf den 8. analogen Ausgang

`delib_cli AO 6 0x4711`

→ setzt den hexadezimalen Wert 0x4AF1 auf den 7. analogen Ausgang

`delib_cli AO 7 3.7V`

→ setzt die Spannung des 8. analogen Ausgangs auf 3,7 Volt

(sowohl Komma "," als auch Punkt "." können zur Kommatrennung verwendet werden)

`delib_cli AO 7 13.3mA`

→ setzt den Strom des 8. analogen Ausgangs auf 13,3 Milliampere

(sowohl Komma "," als auch Punkt "." können zur Kommatrennung verwendet werden)

### Analoge Eingänge

```
delib_cli AI 2
```

Beispiel eines Rückgabewertes: **1234**

→ liest den Wert des 3. analogen Eingangs als dezimalzahl

```
delib_cli AI 2 hex
```

Beispiel eines Rückgabewertes: **0x1FA**

→ liest den Wert des 3. analogen Eingangs als hexadezimalzahl

```
delib_cli AI 2 V
```

Beispiel eines Rückgabewertes: **12.500000V**

→ liest die Spannung des 3. analogen Eingangs als kommazahl

```
delib_cli AI 2 mA
```

Beispiel eines Rückgabewertes: **20.551600mA**

→ liest den Strom des 3. analogen Eingangs als kommazahl

Alternativ kann auch hier das Argument "nunit" an alle zu formatierenden Ausgabeanfragen wie folgendermaßen angehängt werden:

```
delib_cli AI 3 hex nunit
```

Beispiel eines Rückgabewertes: **1FA**

→ liest den Wert des 4. analogen Eingangs als hexadezimalzahl

```
delib_cli AI 3 V nunit
```

Beispiel eines Rückgabewertes: **12.500000**

→ liest die Spannung des 4. analogen Eingangs als kommazahl

## **4.1.5. Ansteuerung über grafische Anwendungen**

### **4.1.5.1. LabVIEW**

Unsere DELIB API kann in LabVIEW importiert und verwendet werden. Alle Produkte die unsere DELIB API verwenden, sind somit mit LabVIEW kompatibel.

Folgendes Kapitel zeigt, wie Sie die DELIB API in LabVIEW einbinden können:  
Einbinden der DELIB in LabVIEW

### **4.1.5.2. ProfiLab**

Die ProfiLab Software der Firma Abacom unterstützt eine große Anzahl unserer Steuer- & Regelungstechnik Produkte.

Link zum Hersteller: <http://www.abacom-online.de/html/profilab-expert.html>

**Die folgenden I/Os werden unterstützt:**

#### **Digitale Ein-/Ausgänge**

- Relais
- MOSFET
- Optokoppler
- Bistabile-Relais

#### **Analoge Ein-/Ausgänge**

- Analog zu digital Wandler
- Digital zu analog Wandler

#### **TTL-I/Os**

- 8/32/64 TTL-Kanäle

#### **4.1.5.3. Licht24 Pro**

Die Licht24 Pro Software der Firma bksoft unterstützt ebenfalls eine hohe Anzahl unserer Produkte.

Mehr Informationen finden Sie unter: <http://www.bksoft.de/licht24pro.htm>



## 4.1.6. Einbinden der DELIB in Programmiersprachen

### 4.1.6.1. Einbinden der DELIB in Visual-C/C++

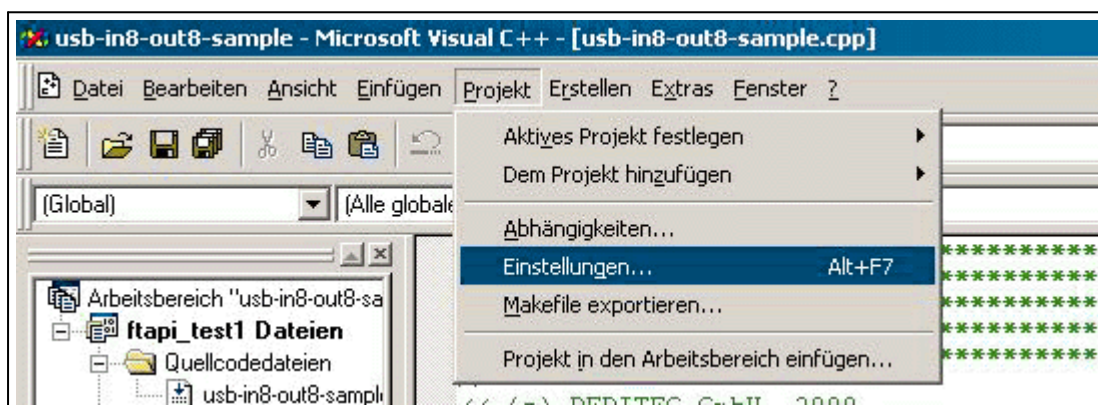
Zur Erleichterung für Verweise auf das DELIB-Include und das DELIB-Lib Verzeichnis werden bei installation der DELIB Umgebungsvariablen definiert.

DELIB\_LIB = C:\Programme\DEDITEC\DELIB\lib

DELIB\_INCLUDE = C:\Programme\DEDITEC\DELIB\include

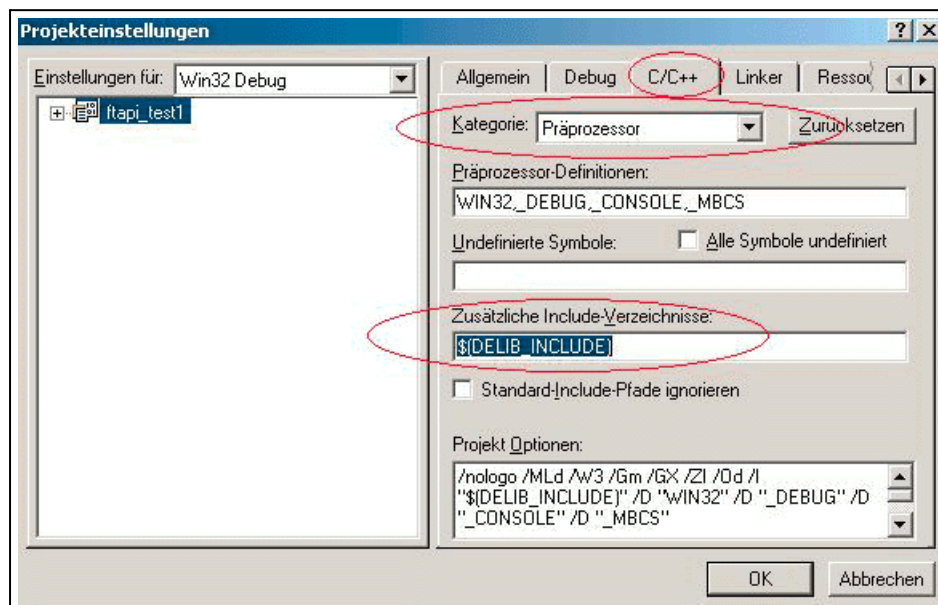
Diese werden im Folgenden in den Projekteinstellungen des Compilers eingetragen.

Visual-C/C++ Starten und im Menue "Projekt → Einstellungen" öffnen.



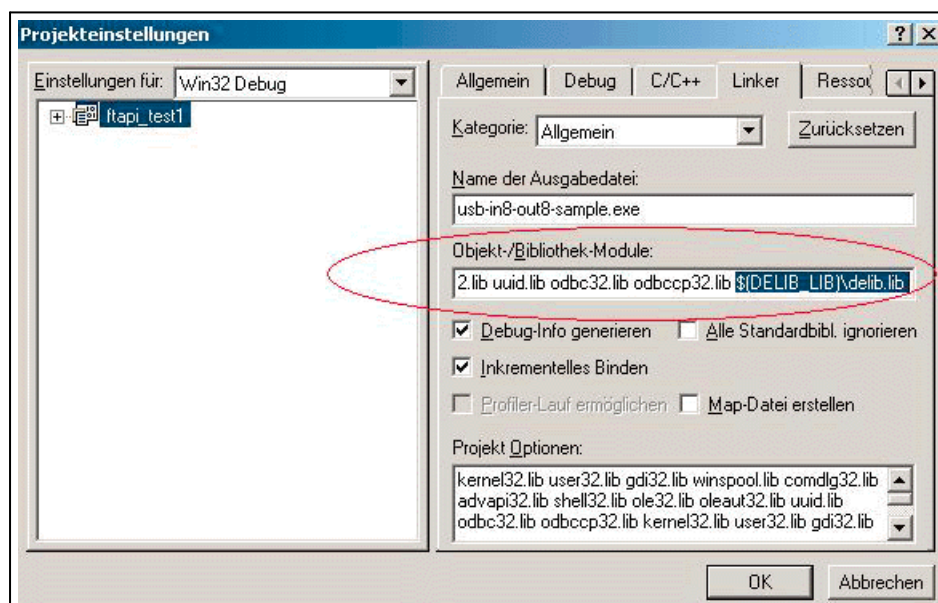
### DELIB.H Eintrag in den Visual-C/C++ Projekt Einstellungen

Unter dem Reiter "C/C++" die "Kategorie" Präprozessor auswählen und unter "Zusätzliche Include Verzeichnisse" "\$(DELIB\_INCLUDE)" eintragen.



### DELIB.LIB Eintrag in den Visual-C/C++ Projekt Einstellungen

Unter dem Reiter "Linker" bei "Objekt-/Bibliothek-Module" die vorhandene Zeile mit der Endung "\$(DELIB\_LIB)\delib.lib" erweitern.



#### 4.1.6.2. Einbinden der DELIB in Visual-C/C++ (Visual Studio 2015)

Zur Erleichterung für Verweise auf das DELIB-Include und das DELIB-Lib Verzeichnis werden bei Installation der DELIB Umgebungsvariablen definiert.

#### 32 Bit DELIB Installation

DELIB\_LIB = C:\Programme\DEDITEC\DELIB\lib

DELIB\_INCLUDE = C:\Programme\DEDITEC\DELIB\include

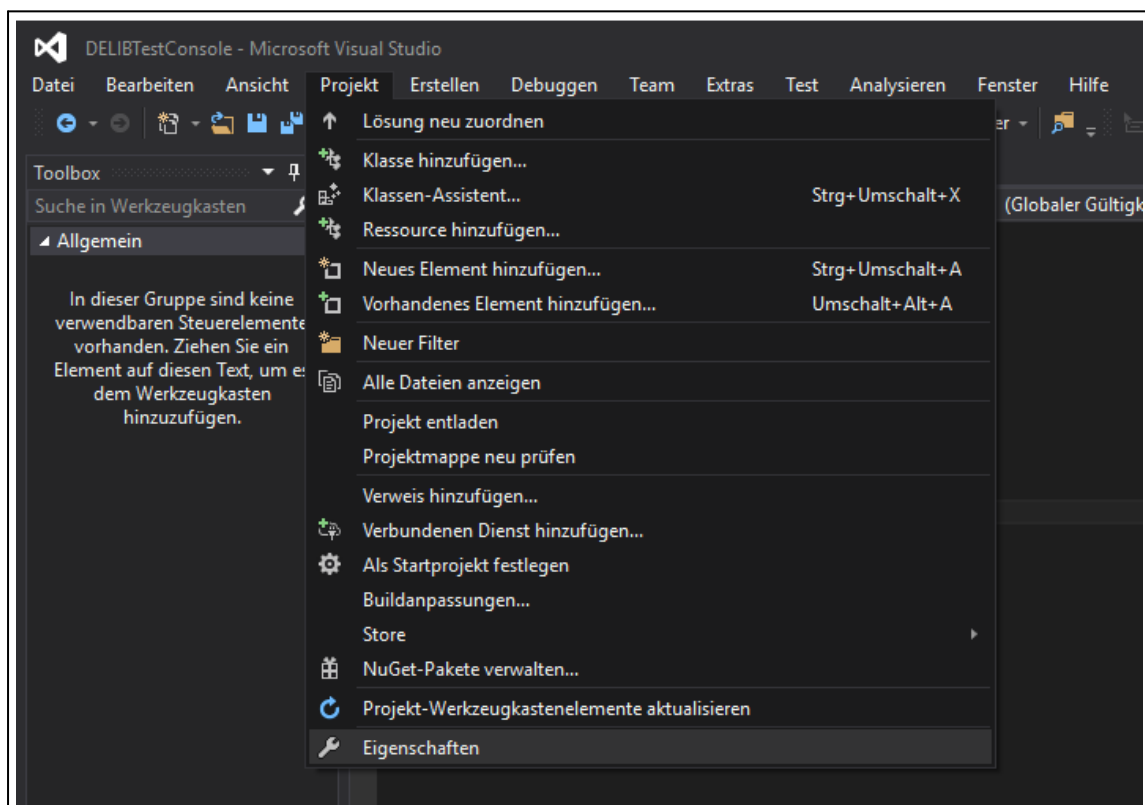
#### 64 Bit DELIB Installation

DELIB64\_LIB = C:\Programme\DEDITEC\DELIB64\lib

DELIB64\_INCLUDE = C:\Programme\DEDITEC\DELIB64\include

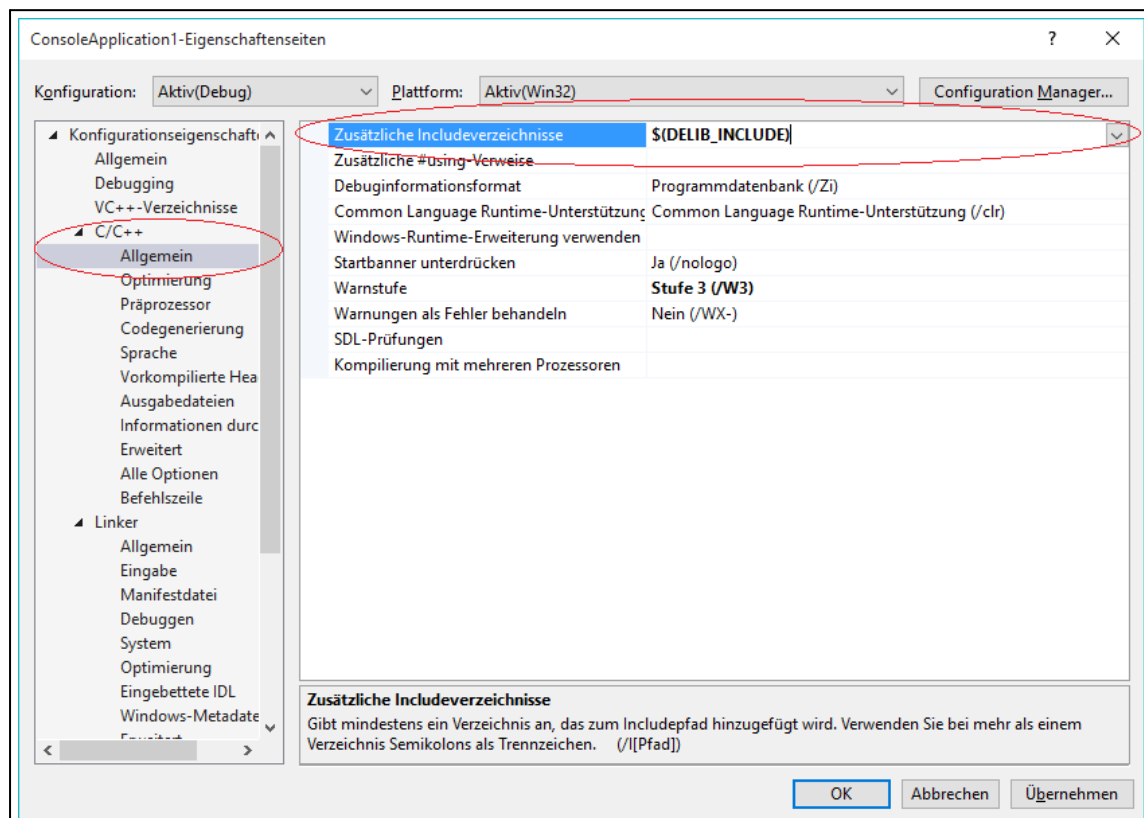
Diese werden im Folgenden in den Projekteinstellungen des Compilers eingetragen.

Visual-C/C++ Starten und im Menue "Projekt → Eigenschaften" öffnen.



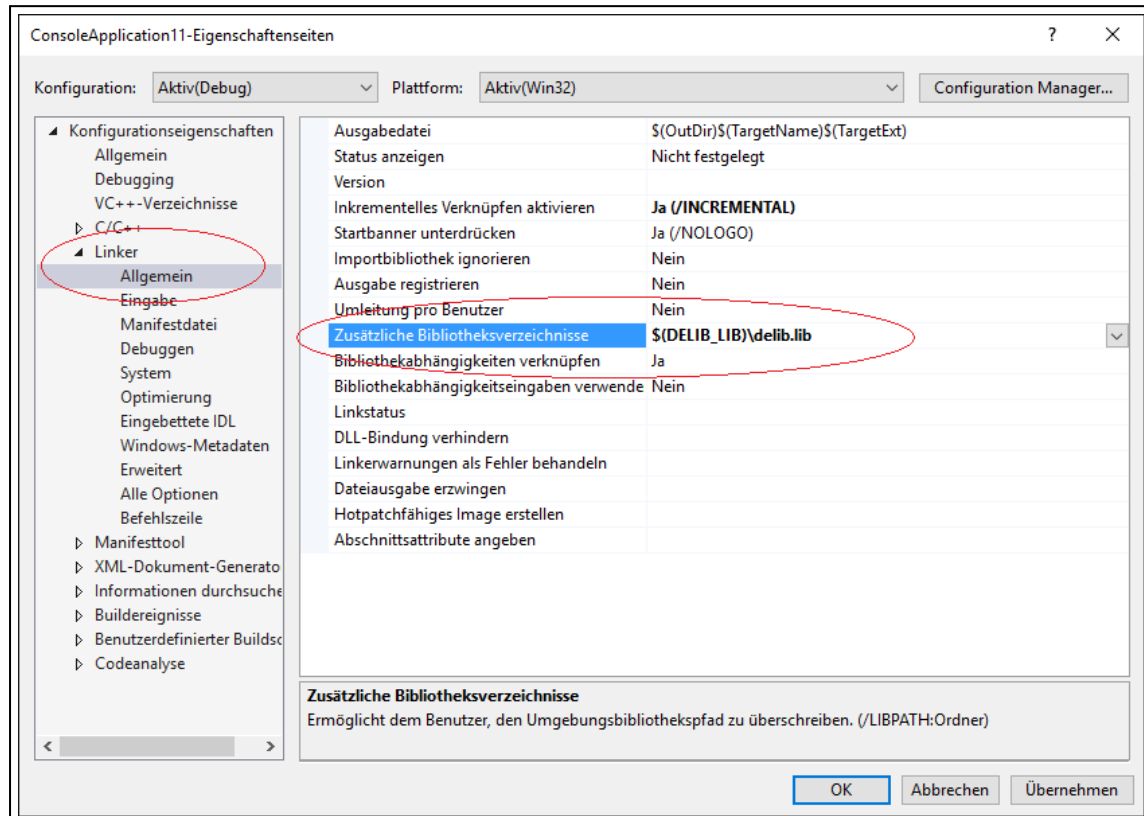
DELIB.H Eintrag in den Visual-C/C++ Projekt Einstellungen

Unter dem Reiter "C/C++" die "Kategorie" Allgemein auswählen und unter "Zusätzliche Include Verzeichnisse" "\$(DELIB\_INCLUDE)" eintragen.



DELIB.LIB Eintrag in den Visual-C/C++ Projekt Einstellungen

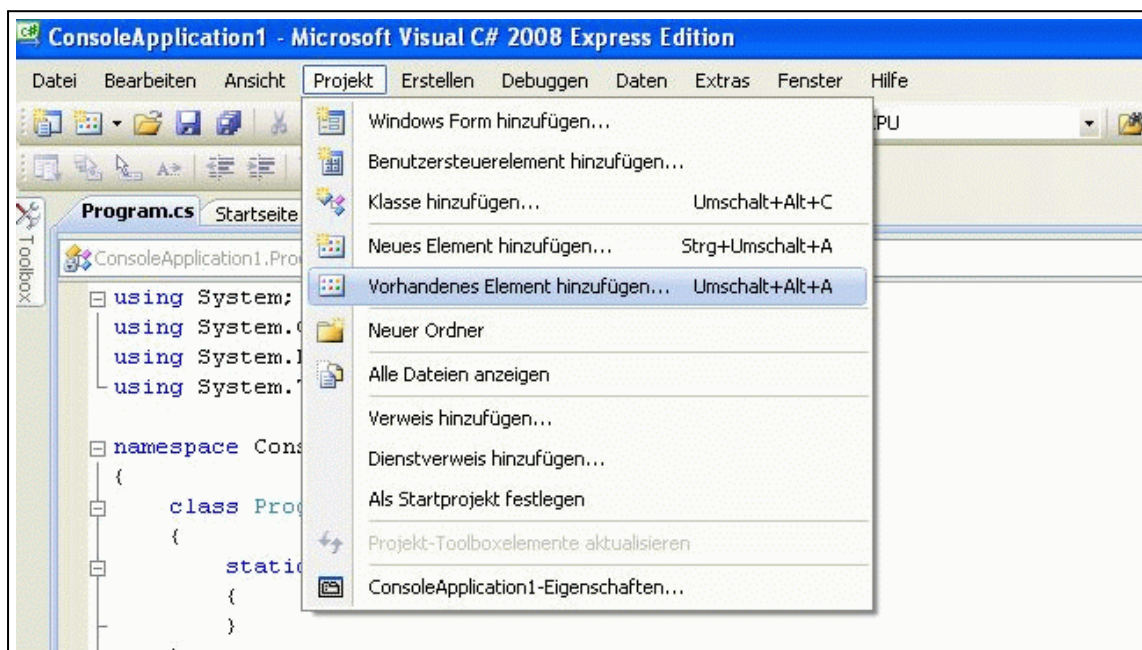
Unter dem Reiter "Linker" bei "Allgemein" "\$ (DELIB\_LIB)\delib.lib" eintragen.



#### 4.1.6.3. Einbinden der DELIB in Visual-C#

Die benötigte Datei für Visual-C# befindet sich im Verzeichnis  
C:\Programme\DEDITEC\DELIB\include.

Visual-C# starten und über das Menue "Projekt → Vorhandenes Element hinzufügen" im Verzeichnis C:\Programme\DEDITEC\DELIB\include\ die Datei delib.cs zum Importieren öffnen.



Folgenden Verweis in Ihrem Programm hinzufügen:

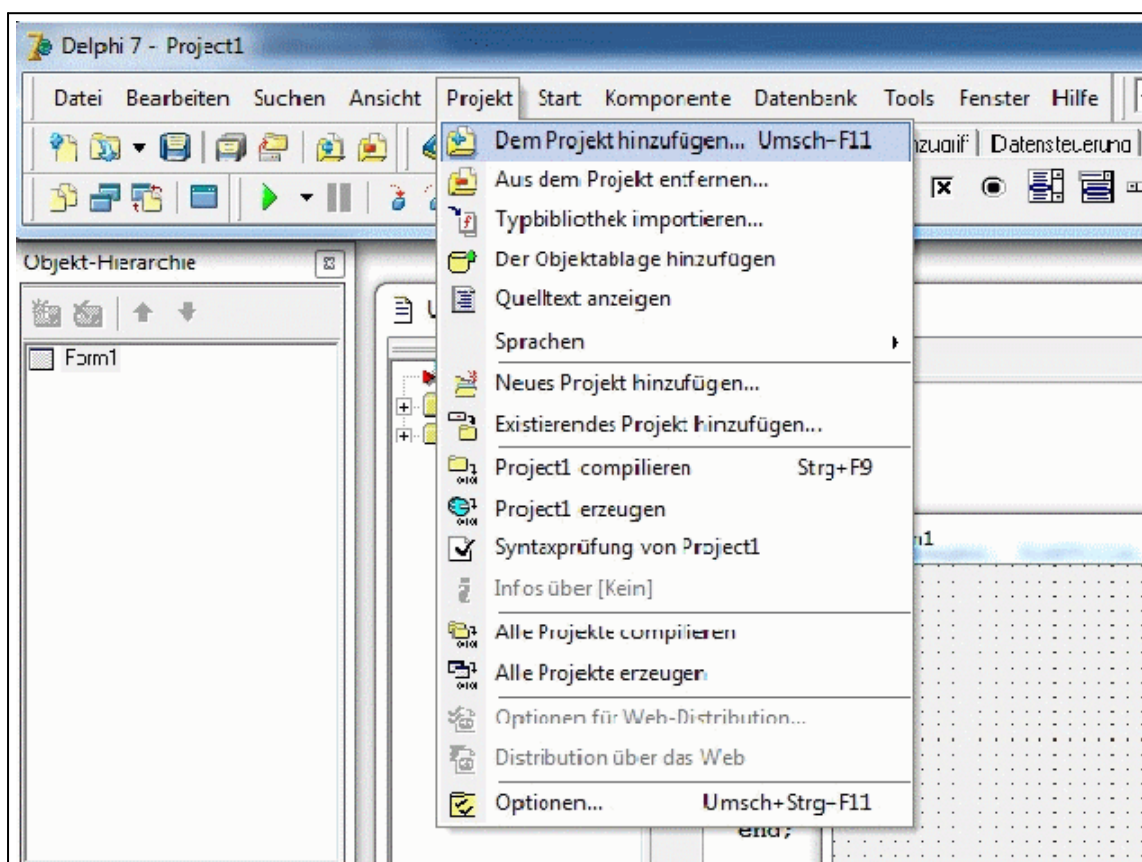
*using DeLib;*

#### 4.1.6.4. Einbinden der DELIB in Delphi

Die benötigte Datei für Delphi befindet sich im Verzeichnis

C:\Programme\DEDITEC\DELIB\include.

Delphi starten und über das Menue "Projekt → dem Projekt hinzufügen" im Verzeichnis C:\Programme\DEDITEC\DELIB\include\ die Datei delib.pas zum Importieren öffnen.

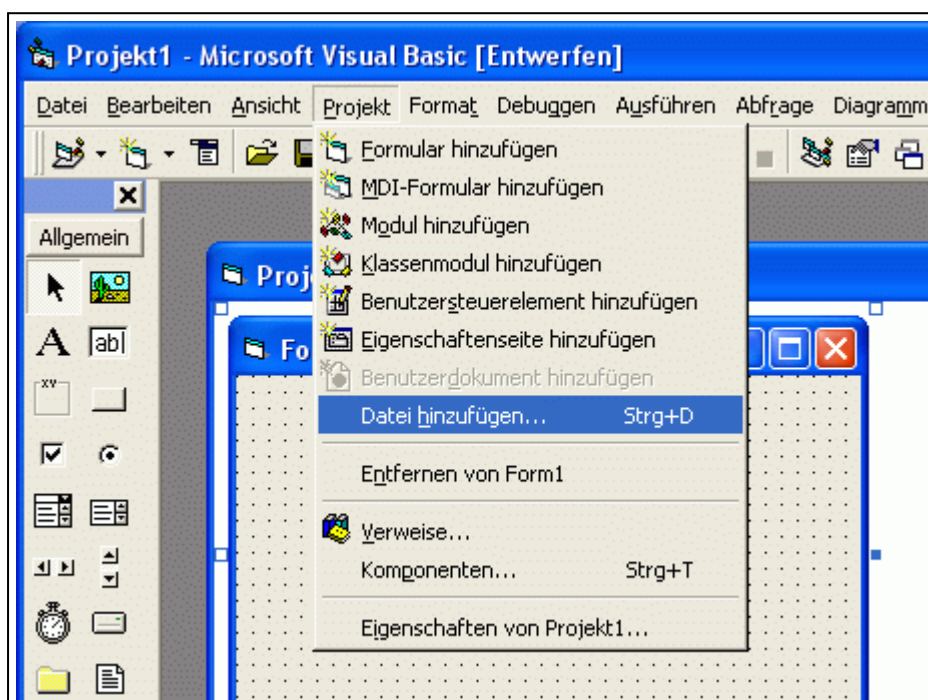




#### 4.1.6.5. Einbinden der DELIB in Visual-Basic (VB)

Die benötigte Datei für Visual-Basic befindet sich im Verzeichnis  
C:\Programme\DEDITEC\DELIB\include.

Visual Basic starten und über das Menue "Projekt → Datei hinzufügen..." im Verzeichnis C:\Programme\DEDITEC\DELIB\include\ die Datei delib.bas zum Importieren öffnen.

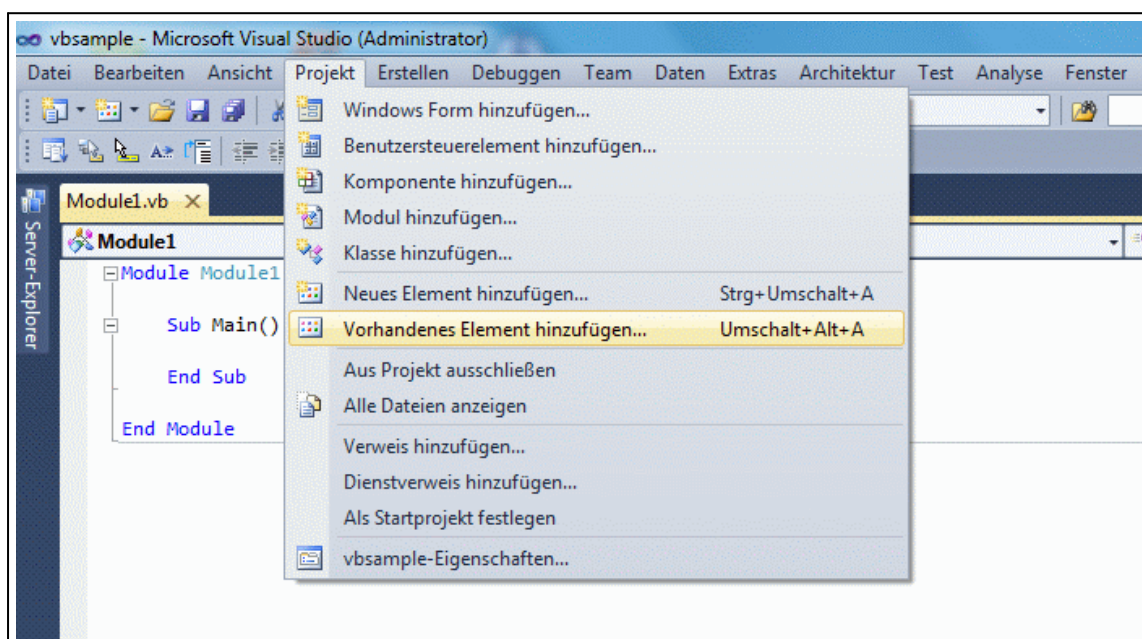




#### 4.1.6.6. Einbinden der DELIB in Visual-Basic.NET (VB.NET)

Die benötigte Datei für VB.NET befindet sich im Verzeichnis  
C:\Programme\DEDITEC\DELIB\include.

VB.NET starten und über das Menue "Projekt → Vorhandenes Element hinzufügen" im Verzeichnis C:\Programme\DEDITEC\DELIB\include\ die Datei delib.vb zum Importieren öffnen.

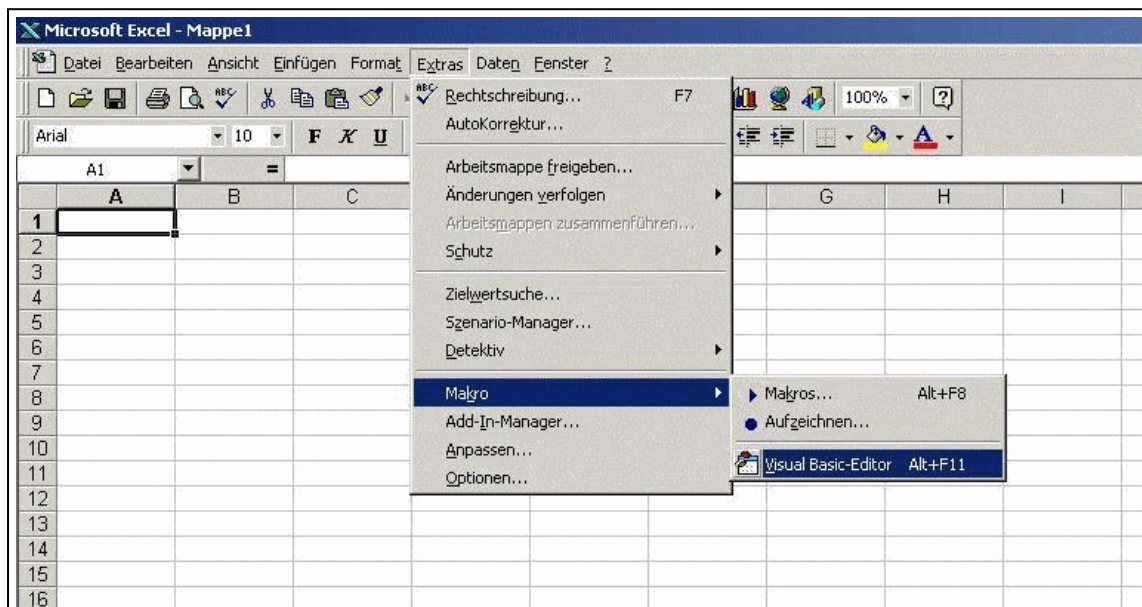


#### 4.1.6.7. Einbinden der DELIB in MS-Office (VBA)

Die benötigte Datei für VBA befindet sich im Verzeichnis

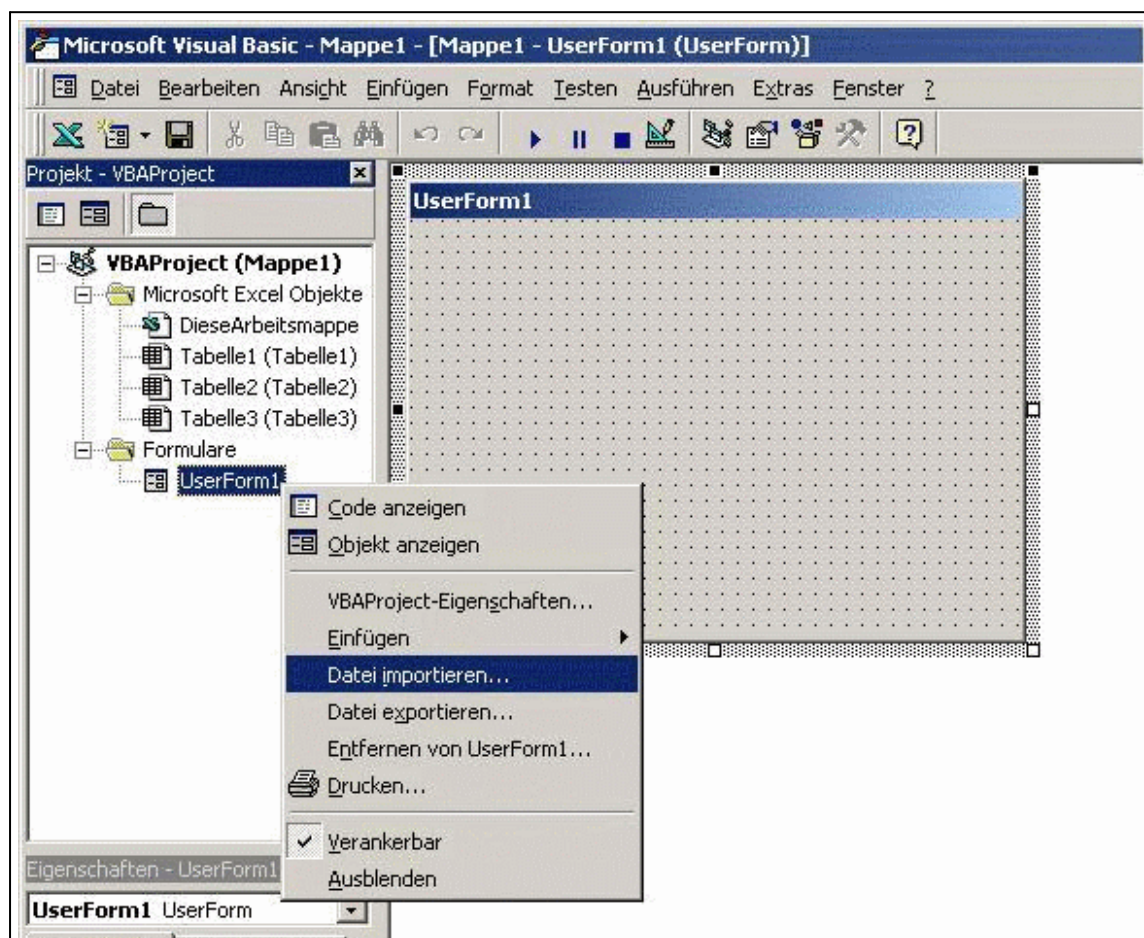
C:\Programme\DEDITEC\DELIB\include.

Microsoft Excel starten und über das Menue "Extras → Makro → Visual Basic Editor" öffnen.



## Erstellen der UserForm

Ein neues Arbeitsblatt (UserForm) über das Menue "Einfügen → UserForm" erstellen. Oben links im Projektmanager einen Rechtsklick auf "UserForm → Datei importieren". Im Verzeichnis C:\Programme\DEDITEC\DELIB\include die Datei delib.bas zum importieren öffnen.



#### 4.1.6.8. Einbinden der DELIB in LabVIEW

##### 4.1.6.8.1. Einbinden der DELIB in LabVIEW

Das LabVIEW-Beispielprogramm "Deditec\_Modul\_Control.vi" ist keine EXE-Datei und benötigt deshalb zur Ausführung die LabVIEW Entwicklungsumgebung.

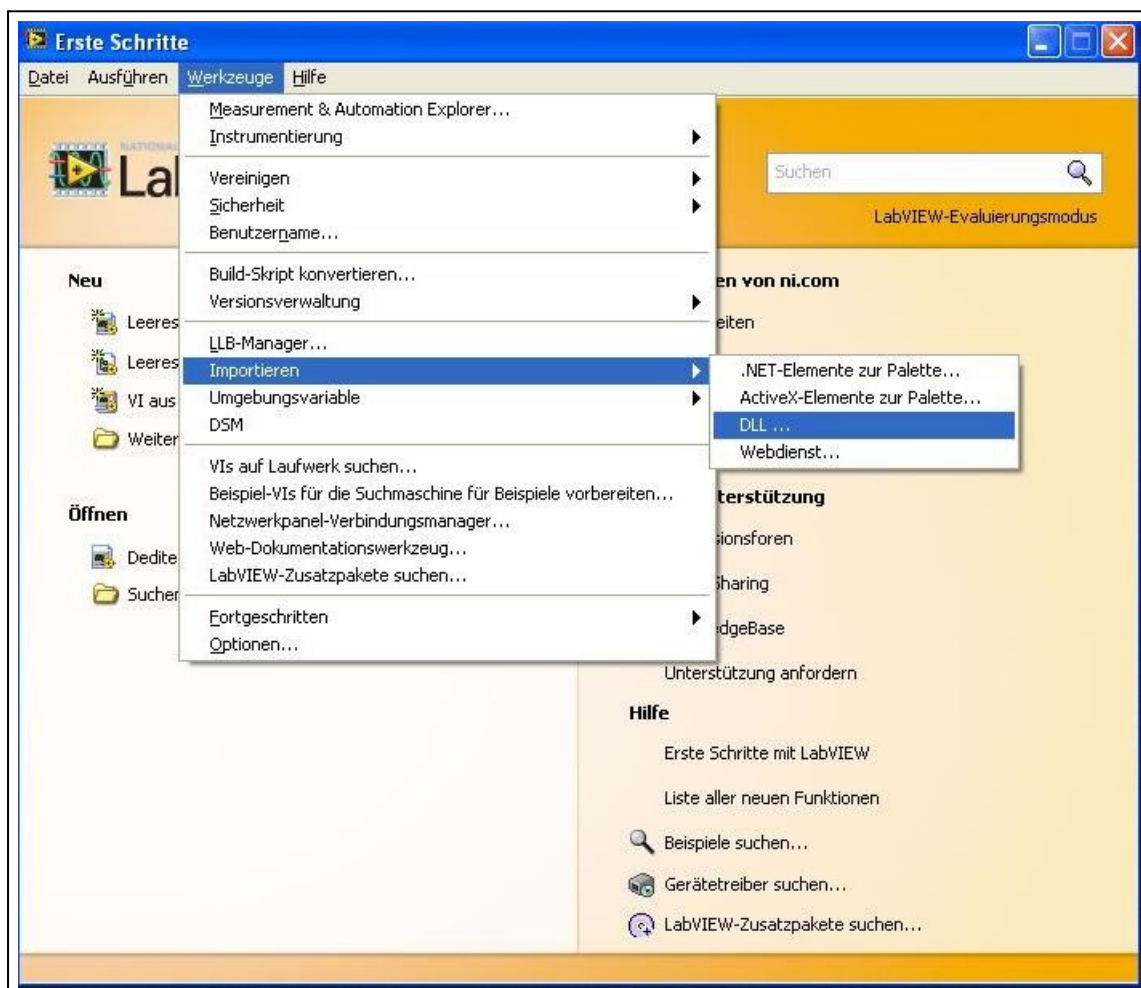
Beschreibung der Einbindung der "delib.dll" in LabVIEW Version 11

- Die benötigten Dateien für LabVIEW befinden sich im Verzeichnis

C:\Windows\System32\delib.dll und in

C:\Programme\DEDITEC\DELIB\include\delib.h

- LabVIEW starten und folgende Option auswählen "Werkzeuge → Importieren → DLL ..."



- Wählen Sie den Punkt "VIs für DLL erstellen" und drücken auf "Weiter"

[illegible]

- Im nächsten Fenster über die Browser-Buttons den Speicherort der delib.dll und der delib.h Datei angeben und mit "Weiter" fortfahren.

**DLL importieren**

DLL- und Header-Datei wählen

NATIONAL INSTRUMENTS

Bibliothek (\*.dll)  
C:\WINDOWS\system32\delib.dll

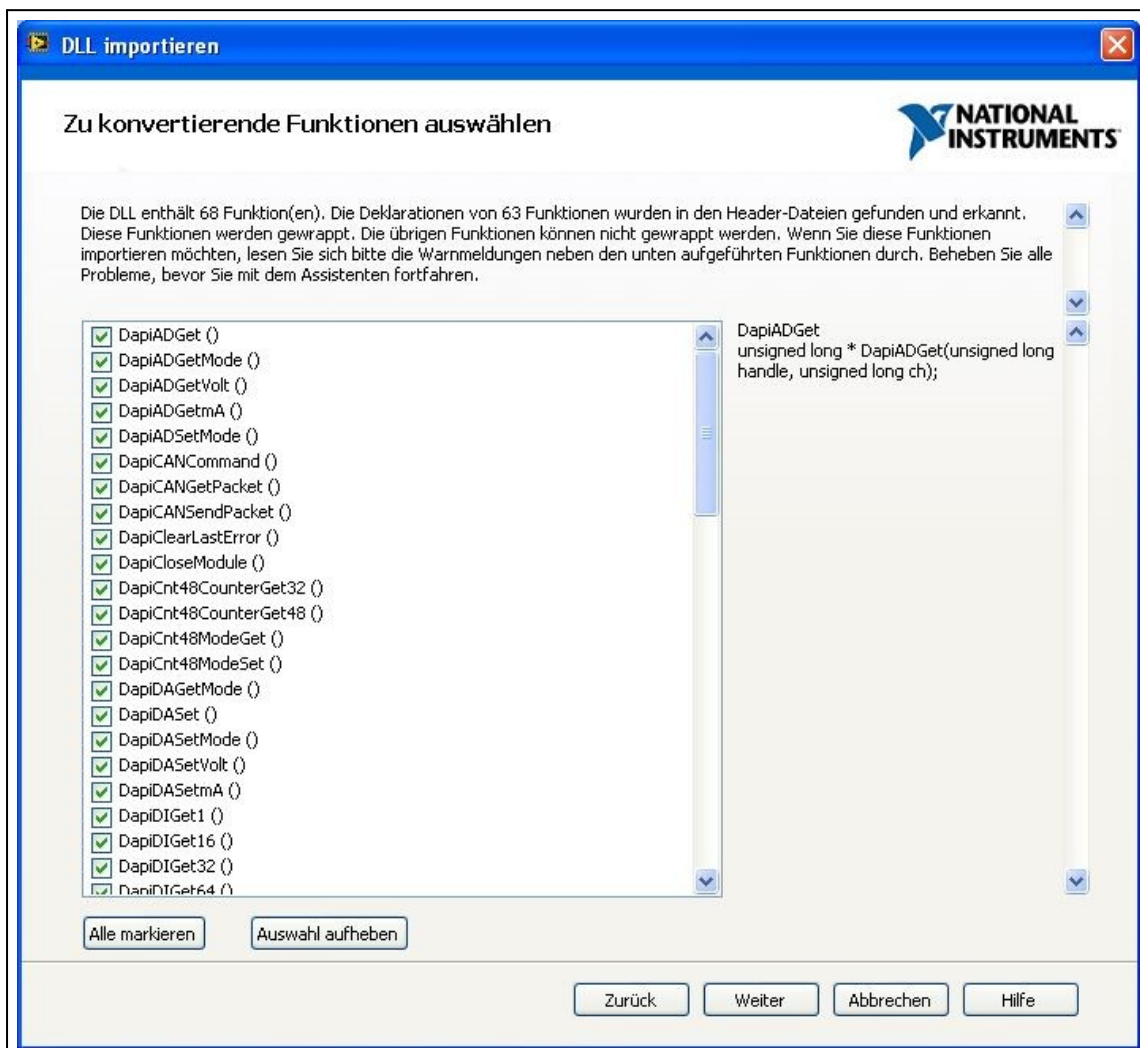
☐ DLL befindet sich nicht auf lokalem Computer

Header-Datei (.h)  
C:\Programme\DEDITEC\DELIB\include\delib.h

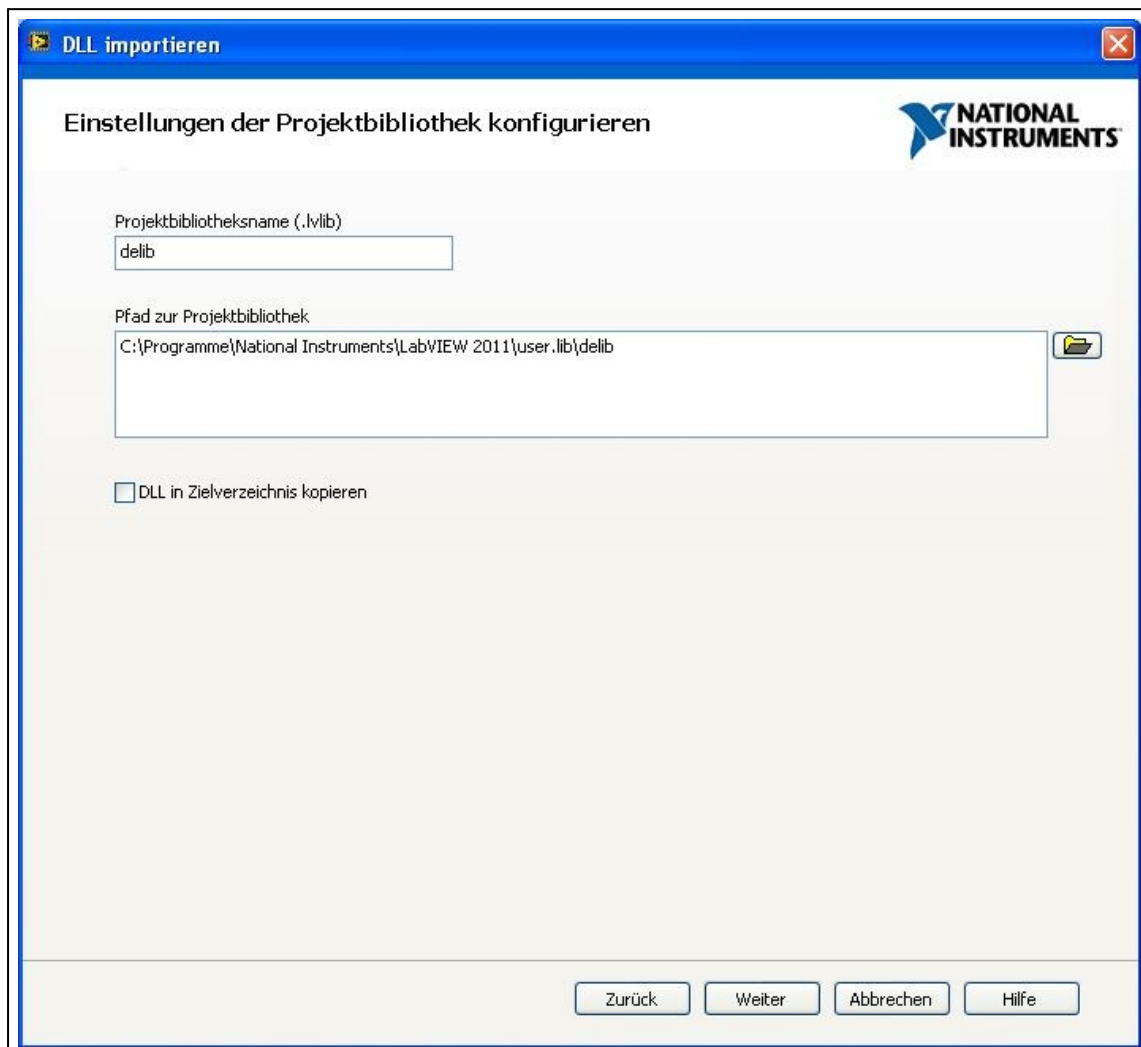
Zurück Weiter Abbrechen Hilfe



- Nochmals auf "Weiter" klicken um fortzufahren.
- Die Header-Datei wird nun analysiert. Anschließend fahren Sie im folgendem Fenster wieder mit "Weiter" fort.

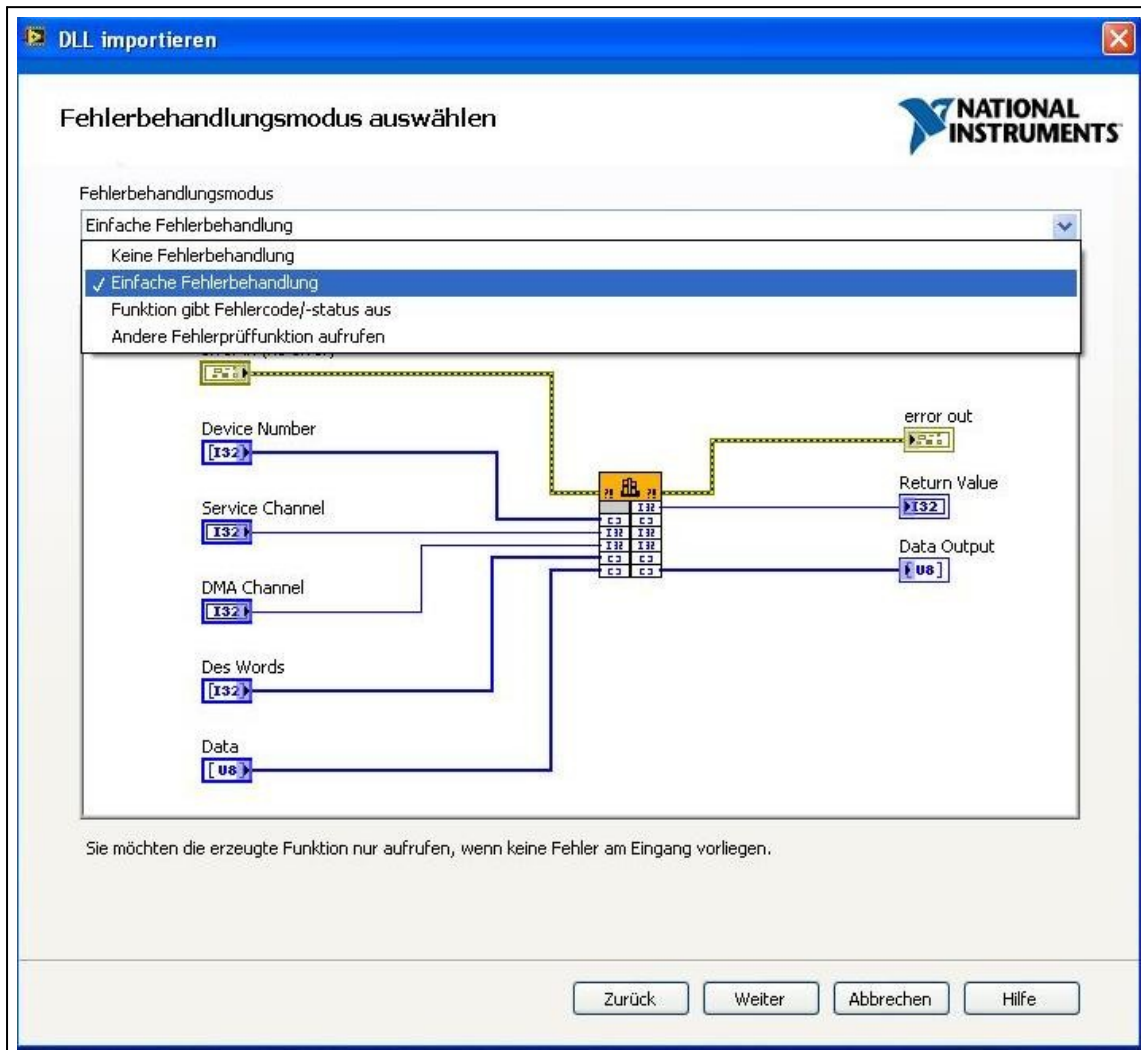


- Den weiteren Anweisungen folgen, bzw. die Konfiguration und den Speicherort für die VIs anpassen.

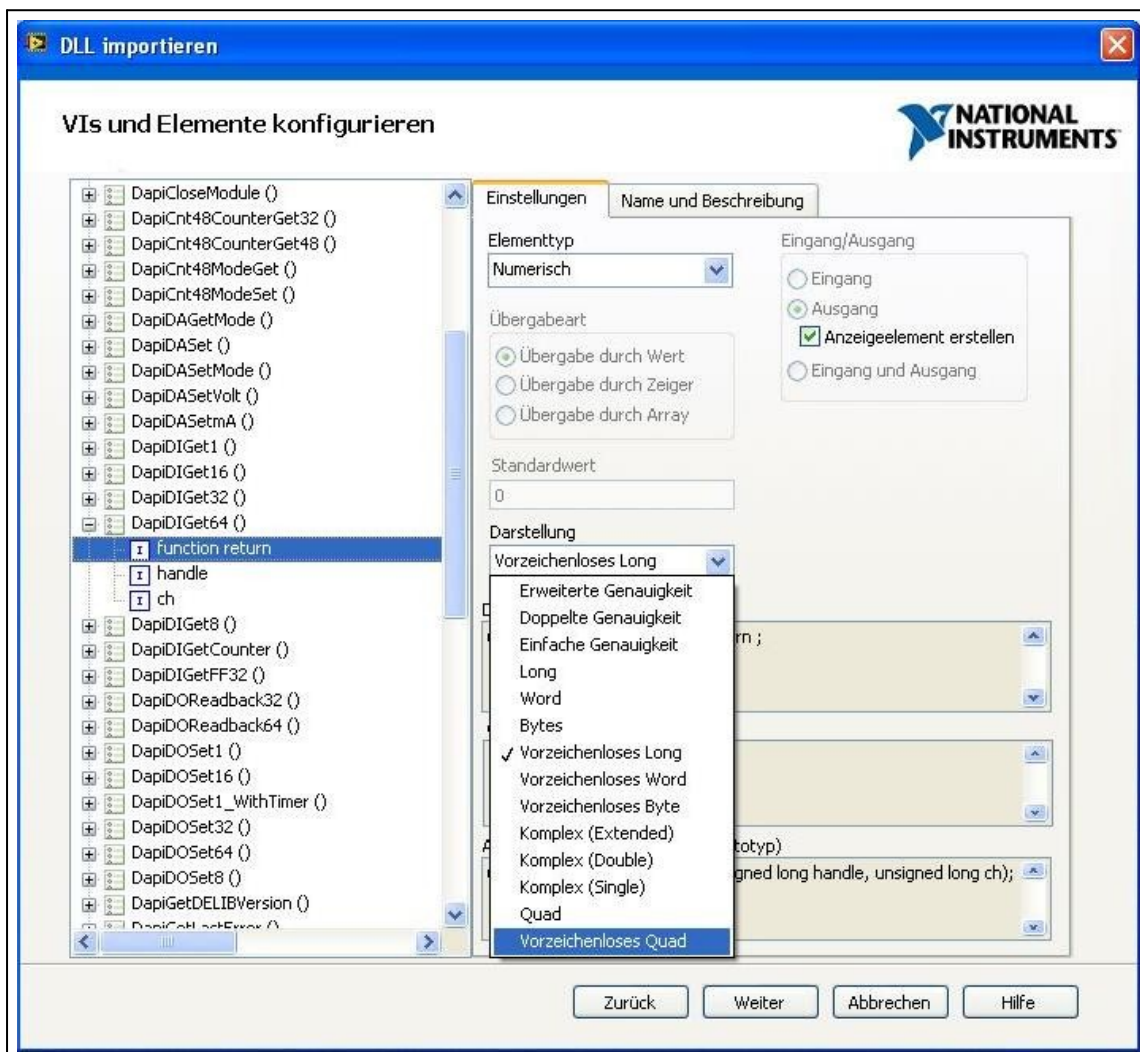




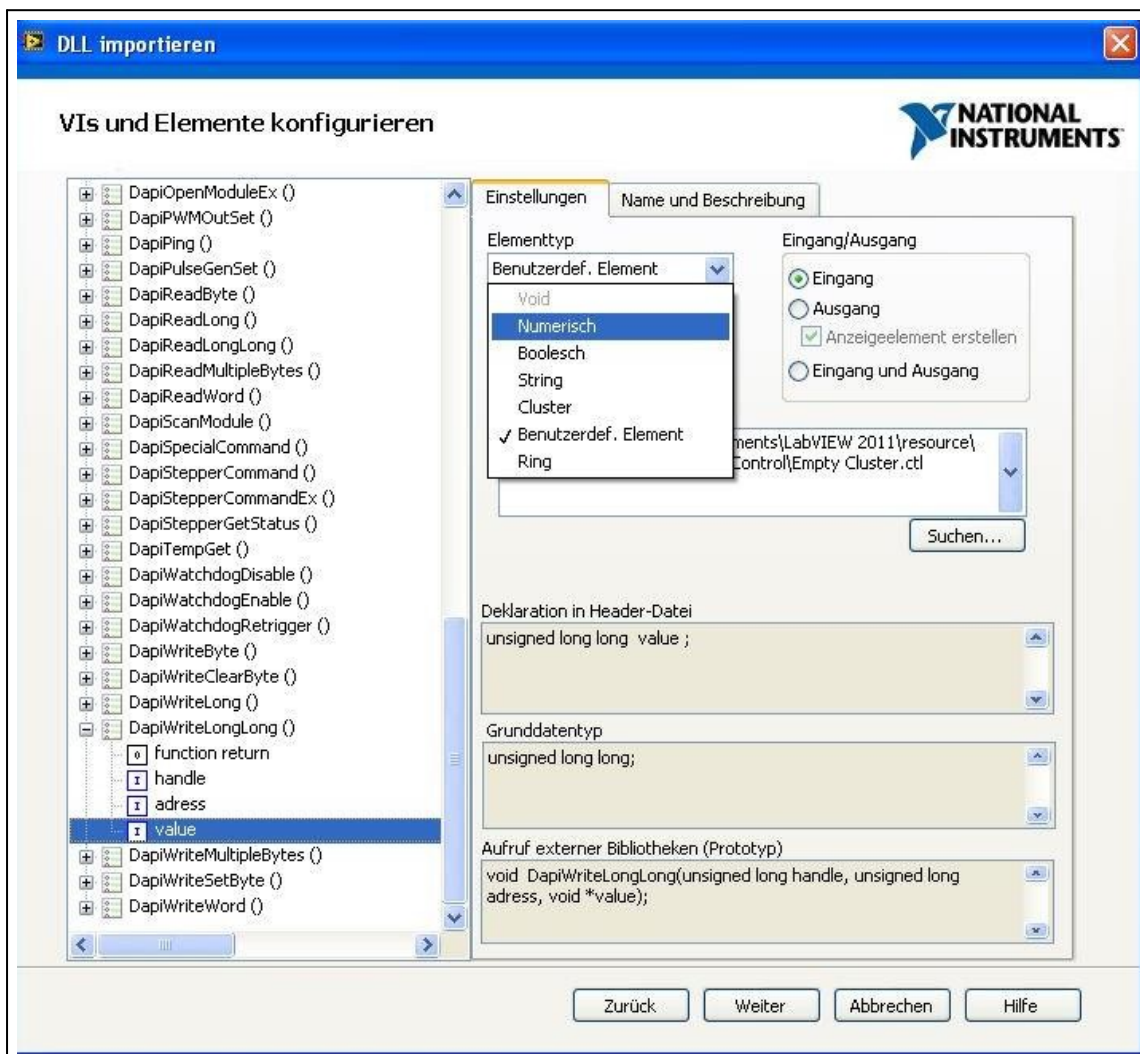
- Im folgendem Fenster wählen Sie im Drop-Down-Menü die Option "Einfache Fehlerbehandlung" aus und fahren mit "Weiter" fort.



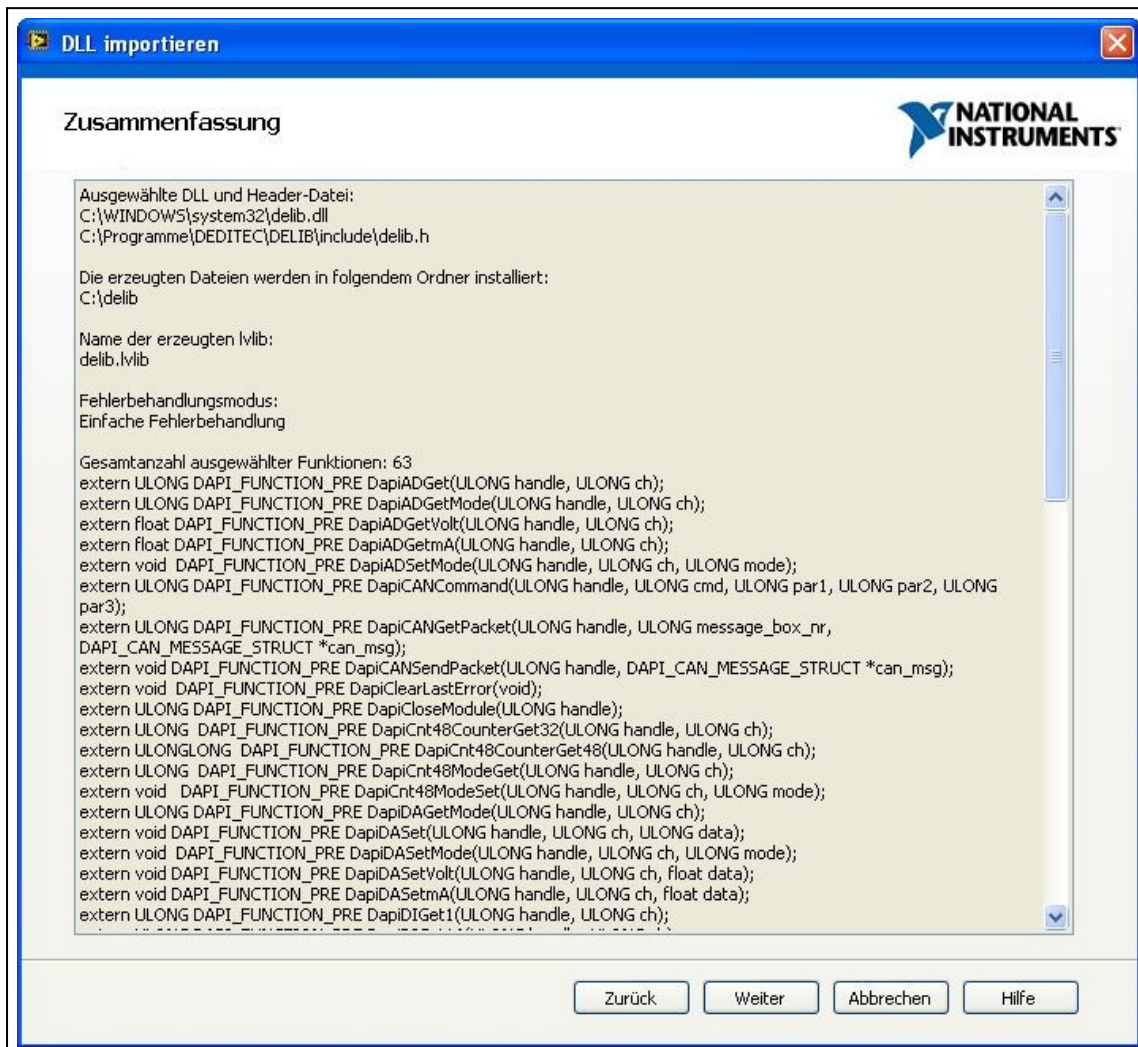
- Bei VIs die mit 64-Bit Werten arbeiten, muss die Darstellung von "Vorzeichenloses Long" in "Vorzeichenloses Quad" geändert werden.
- Folgende VIs müssen bearbeitet werden:
  - DapiCNT48CounterGet48 (function return)
  - DapiDIGet64 (function return)
  - DapiDOSet64 (data)
  - DapiDOReadBack64 (function return)



- Bei manchen VIs muss zusätzlich noch der Elementtyp auf "Numerisch" geändert werden und anschließend die Darstellung auf "Vorzeichenloses Quad"
- Folgende VIs müssen bearbeitet werden:
  - DapiWriteLongLong (value)
  - DapiReadLongLong (function return)



- Es erscheint eine Zusammenfassung der ausgeführten Schritte.
- Zum Fortfahren auf "Weiter" drücken.



- Die VIs werden nun erzeugt und können verwendet werden.

#### 4.1.6.8.2. Verwendung der VIs in LabVIEW

In unseren Beispielprogrammen werden bei manchen Funktionen sogenannte Defines als Übergabeparameter verwendet.

Diese Defines werden in LabVIEW nicht unterstützt.

Dieses Beispiel soll zeigen, wie solche Funktionen in LabVIEW genutzt werden können.

Als Beispiel dient uns hierbei die Funktion zur Konfiguration des Spannungsbereiches eines A/D Wandlers.

**Die Definition für die Funktion lautet:**

```
void DapiADSetMode(ULONG handle, ULONG ch, ULONG mode);
```

Für die Funktion sind die Spannungsbereiche in der DELIB Treiberbibliothek bereits vordefiniert.

```
// -----  
// A/D and D/A Modes  
  
#define ADDA_MODE_UNIPOL_10V 0x00  
#define ADDA_MODE_UNIPOL_5V 0x01  
#define ADDA_MODE_UNIPOL_2V5 0x02
```

**Beispielcode in C/C++:**

```
DapiADSetMode(handle, 0, ADDA_MODE_UNIPOL_5V);
```

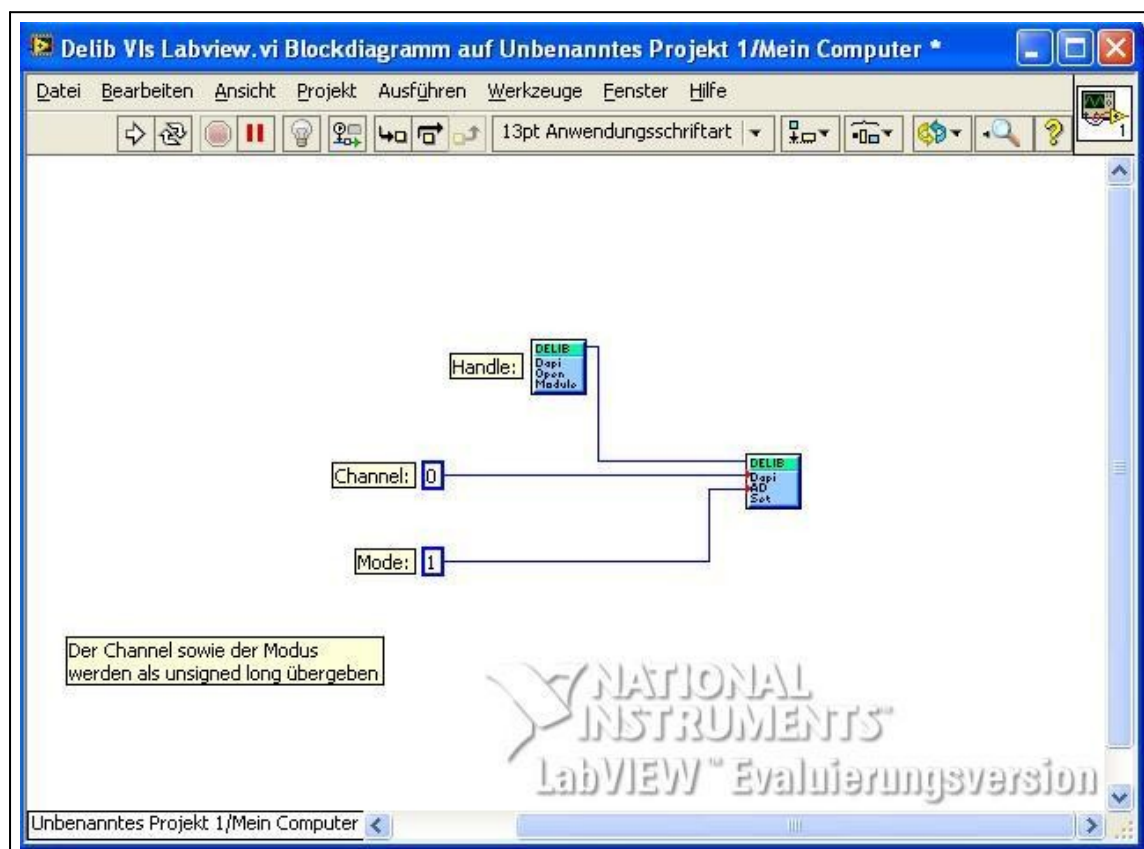
**Alternativ kann man auch folgende Schreibweise verwenden:**

```
DapiADSetMode(handle, 0, 0x01);
```

Hierbei wurde der Hexadezimalwert, den Sie aus der delib.h Datei entnehmen können, als Parameter für den Modus übergeben

Die delib.h Datei finden sie nach der Installation der DELIB Treiberbibliothek im Verzeichnis C:\Programme\Deditec\DELIB\Include

In LabVIEW könnte die Funktion dann so aussehen:





#### 4.1.6.8.3. Setzen der Modul-ID in LabVIEW

Im folgendem Beispiel wird das Ansprechen eines RO-ETH-Moduls in LabVIEW gezeigt.

Die Verbindung zum Modul wird mittels der Funktion DapiOpenModule hergestellt.

Die Definition für diese Funktion lautet:

*ULONG DapiOpenModule(ULONG moduleID, ULONG nr);*

Als Parameter für moduleID wird üblicherweise die Modul-ID (z.B. "RO\_ETH") des verwendeten Moduls übergeben.

Eine Übersicht aller möglichen Modul-IDs kann der Datei "delib.h" entnommen werden.

Die delib.h finden Sie nach der Installation der DELIB Treiberbibliothek im Verzeichnis C:\Programme\Deditec\DELIB\Include

```
// *****  
// *****  
//  
//  
#define DELIB_VERSION 0x0141 // Actual DELIB-Version  
  
// all Modul-ID's  
#define USB_Interface8 1 // USB-Controller8/USB-TTL-IN8-OUT8  
#define USB_CAN_STICK 2 // USB-CAN-Stick  
#define USB_LOGI_500 3 // USB-LOGI-500/USB-LOGI-250  
#define RO_USB2 4 // RO-CPU2 / 480 MBit/sec  
#define RO_SER 5 // RO-SER-Serie  
#define USB_BITP_200 6 // USB-BITP-200  
#define RO_USB1 7 // RO-USB-Serie  
#define RO_USB 7 // RO-USB-Serie  
#define RO_ETH 8 // RO-ETH-Serie  
#define USB_MINI_STICK 9 // USB-MINI-Stick-Serie  
#define USB_LOGI_18 10 // USB-LOGI-100  
#define RO_CAN 11 // RO-CAN-Serie  
#define USB_SPI_MON 12 // USB_SPI_MON  
#define USB_WATCHDOG 13 // USB_Watchdog  
#define USB_OPTOIN_8 14 // USB-OPTOIN8 / USB-RELAIS-8  
#define USB_RELAIS_8 14 // USB-OPTOIN8 / USB-RELAIS-8  
#define USB_OPTOIN_8_RELAIS_8 15 // USB-OPTOIN-8-RELAIS-8  
#define USB_OPTOIN_16_RELAIS_16 16 // USB-OPTOIN-16-RELAIS-16  
#define USB_OPTOIN_32 16 // USB-OPTOIN-16-RELAIS-16  
#define USB_RELAIS_32 16 // USB-OPTOIN-16-RELAIS-16  
#define USB_OPTOIN_32_RELAIS_32 17 // USB-OPTOIN-32-RELAIS-32  
#define USB_OPTOIN_64 17 // USB-OPTOIN-32-RELAIS-32  
#define USB_RELAIS_64 17 // USB-OPTOIN-32-RELAIS-32  
#define USB_TTL_32 18 // USB-TTL-32  
#define USB_TTL_64 18 // USB-TTL-64  
  
#define MAX_NR_OF_MODULES 18
```

### Beispiel in C:

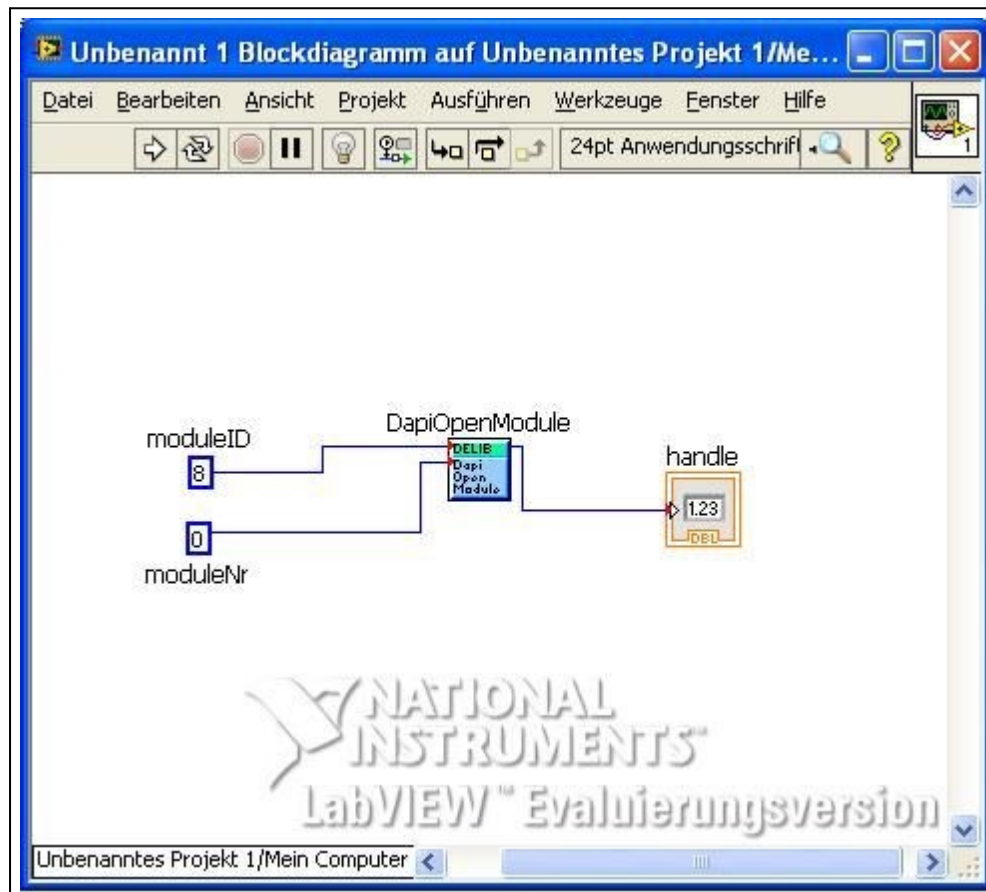
```
handle = DapiOpenModule(RO_ETH, 0); // öffnet ein RO-ETH-Modul mit Modul-Nr 0.
```

### Alternativ kann man auch folgende Schreibweise verwenden:

```
handle = DapiOpenModule(8, 0);
```

Da es in LabVIEW nicht möglich ist, diese "C-Defines" als Parameter für die Funktion DapiOpenModule zu übergeben, muss hier die alternative Schreibweise verwendet werden.

### Beispiel in Labview:





#### 4.1.6.9. Einbinden der DELIB in Java

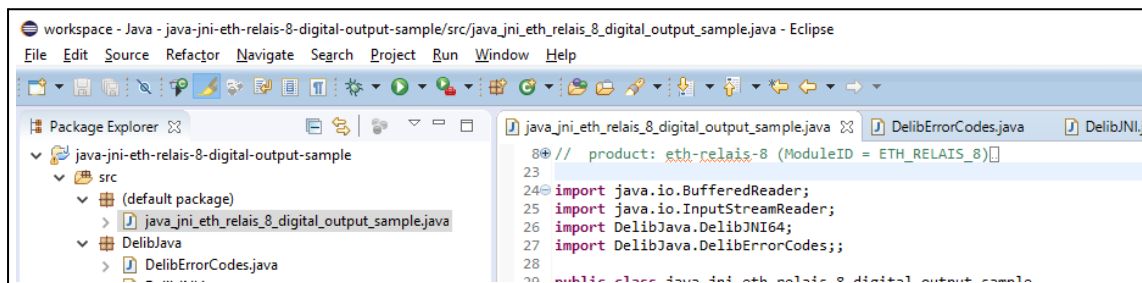
Die benötigten Dateien für Java befinden sich, je nach DELIB-Installation, in folgendem Verzeichnis

C:\Program Files (x86)\DEDITEC\DELIB\include\DelibJava (32 Bit Installation)

C:\Program Files\DEDITEC\DELIB64\include\DelibJava (64 Bit Installation)

Wird Eclipse verwendet, kann der DelibJava-Ordner einfach per Drag&Drop dem Projekt hinzugefügt werden.

Anschließend müssen die verwendeten Module noch importiert werden.



## 4.2. DELIB Treiberbibliothek

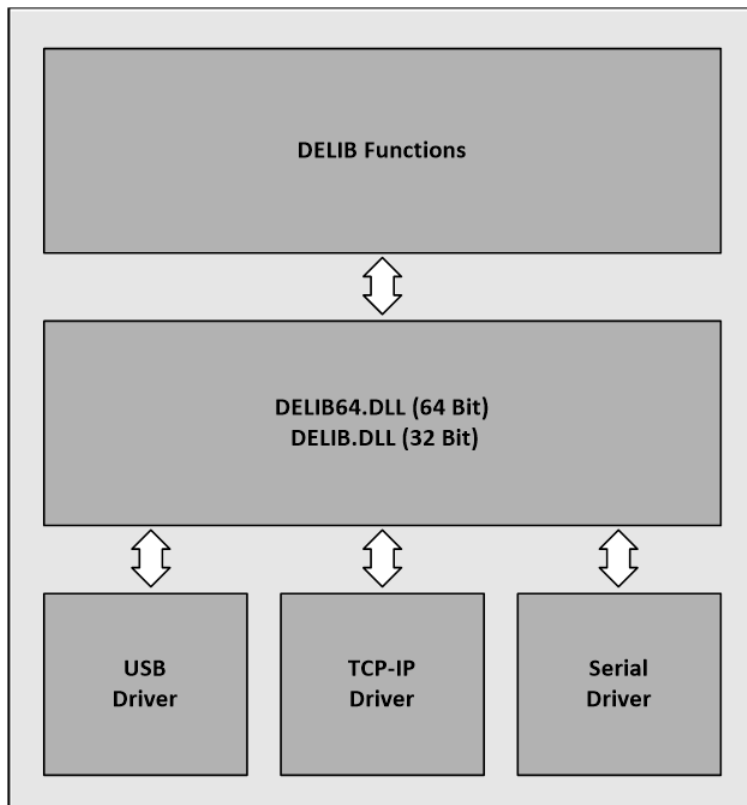
Die DELIB Treiberbibliothek enthält die DELIB-API und verschiedene Programme für den Konfigurationstest unserer Produkte.

Über die API haben Sie Zugriff auf alle Funktionen, die Sie zur Kommunikation mit unseren Produkten benötigen.

In dem Kapitel **DELIB API Referenz** finden Sie alle Funktionen unserer Treiberbibliothek erklärt und mit Anwendungsbeispielen versehen.

### 4.2.1. Übersicht

Die folgende Abbildung erläutert den Aufbau der DELIB Treiberbibliothek



Die DELIB Treiberbibliothek ermöglicht ein einheitliches Ansprechen von DEDITEC Hardware, mit der besonderen Berücksichtigung folgender Gesichtspunkte:

- Betriebssystem unabhängig
- Programmiersprachen unabhängig
- Produkt unabhängig

**Diese Versionen der Treiberbibliothek bieten wir an:**

- 32/64-Bit DELIB Treiberbibliothek für Windows
- 32/64-Bit DELIB Treiberbibliothek für Linux
- 32/64-Bit DELIB Treiberbibliothek ETH

#### **4.2.1.1. Unterstützte Programmiersprachen**

**Die folgenden Programmiersprachen werden von der DELIB Treiberbibliothek unterstützt:**

- C
- C++
- C#
- Delphi
- VisualBasic
- VB.NET
- MS-Office (VBA)
- Java (Plattformunabhängig, nur für Ethernet-Produkte)
- Java JNI (nur für Windows, alle Produkte werden unterstützt)

Falls von der Programmiersprache/Entwicklungsumgebung vorgesehen, unterstützen wir sowohl 32-Bit als auch 64-Bit Projekte.

#### **4.2.1.2. Unterstützte Betriebssysteme**

Die folgende Betriebssysteme sind mit unserer DELIB Treiberbibliothek kompatibel:

##### **32-Bit:**

- Windows 10
- Windows 7
- Windows 8
- Windows Server 2012
- Windows Server 2008
- Windows Vista
- Windows XP
- Windows Server 2003
- Windows 2000
- Linux

##### **64-Bit:**

- Windows 10 x64
- Windows 7 x64
- Windows 8 x64
- Windows Server 2012 x64
- Windows Server 2008 x64
- Windows Vista x64
- Windows XP x64
- Windows Server 2003 x64
- Linux x64

#### **4.2.1.3. SDK-Kit für Programmierer**

Integrieren Sie die DELIB in Ihre Anwendung. Auf Anfrage erhalten Sie von uns kostenlos Installationsskripte, die es ermöglichen, die DELIB Installation in Ihre Anwendung mit einzubinden.

#### 4.2.2. DELIB Setup

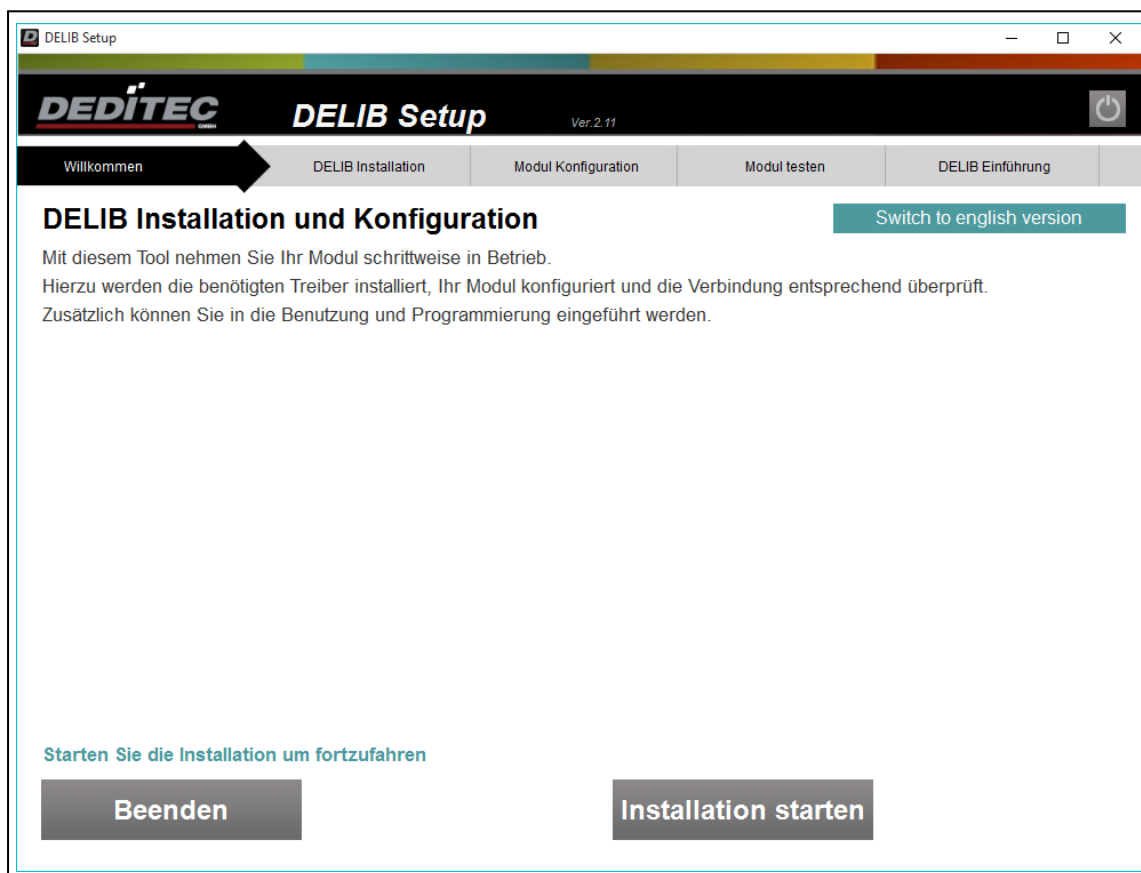
Das DELIB Setup führt Sie durch die Installation unserer DELIB Treiberbibliothek.

Anschließend werden Sie durch den Konfigurationsvorgang sowie Funktionstest für unsere verschiedenen Produkte geführt.

Die aktuelle Version des DELIB Setups finden Sie auf unserer Homepage zum Download.

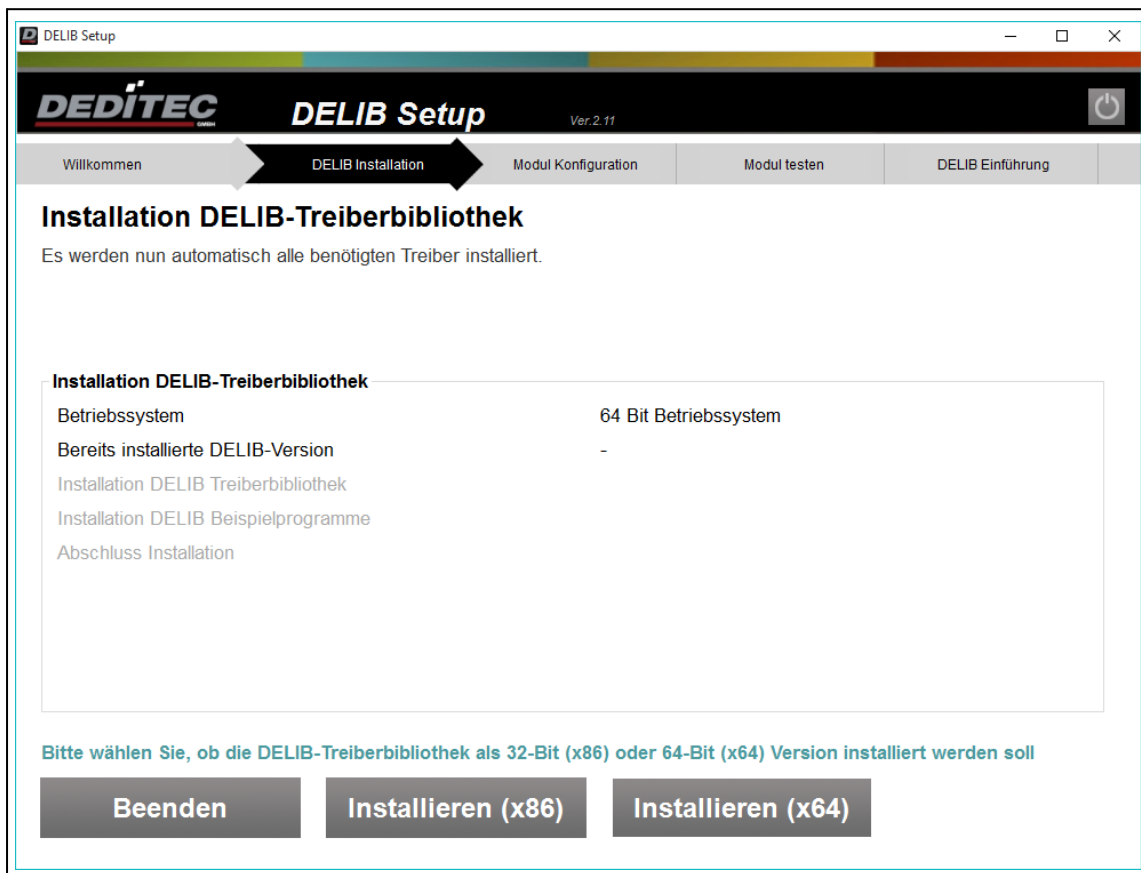
Link: <https://www.deditec.de/delib>

Das DELIB Setup führt Sie Schritt für Schritt durch die Installation der DELIB Treiberbibliothek einschließlich der Konfiguration und Inbetriebnahme der Produkte.

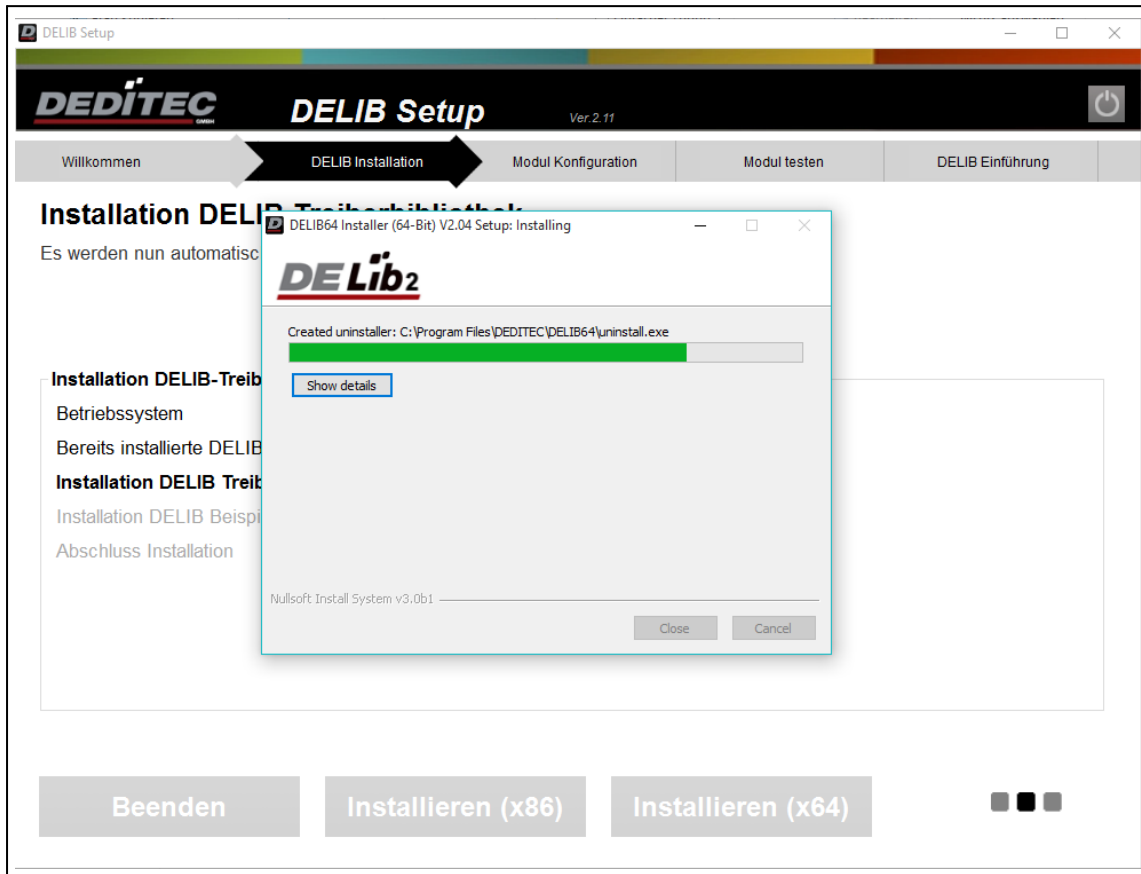


Das DELIB Setup prüft das Betriebssystem und ob bereits Versionen der DELIB Treiberbibliothek installiert sind.

Anschließend können Sie wählen ob Sie die 32-Bit oder 64-Bit Version der DELIB Treiberbibliothek installieren möchten.

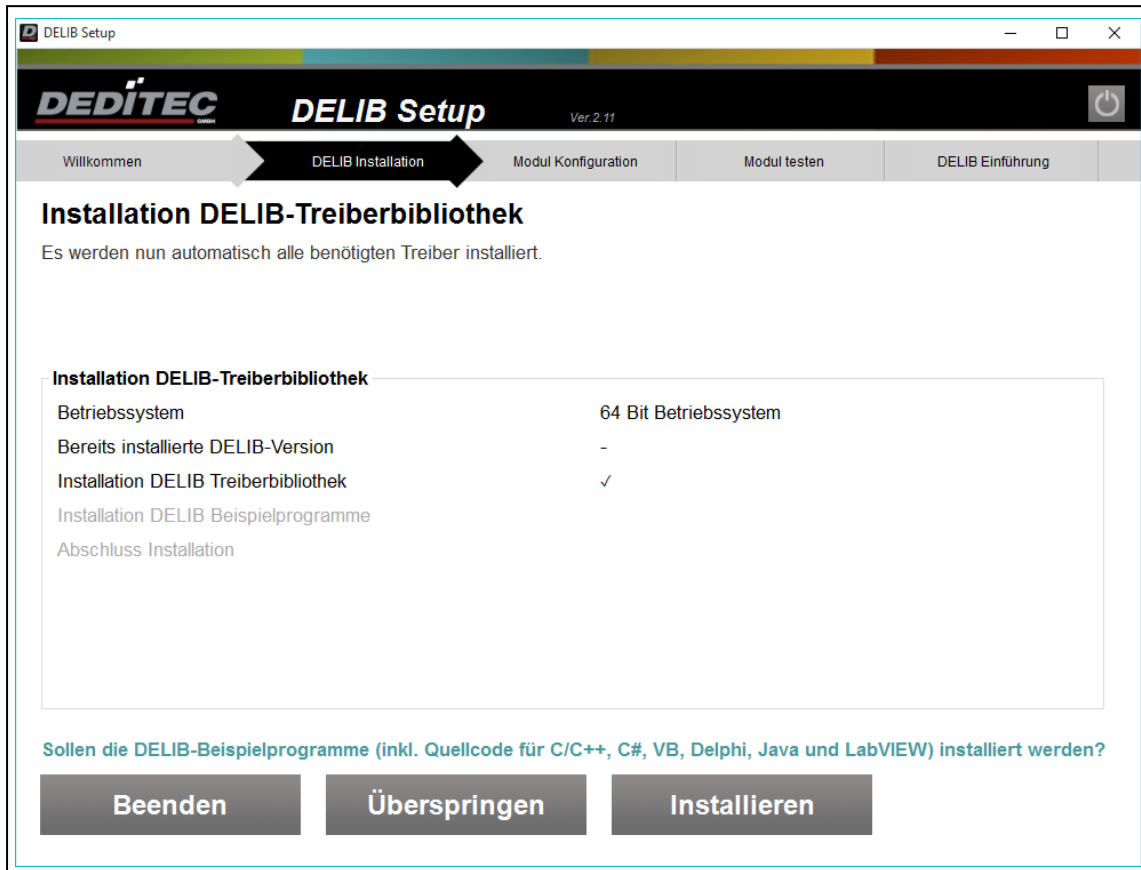


Installationsfortschritt der Treiberbibliothek.

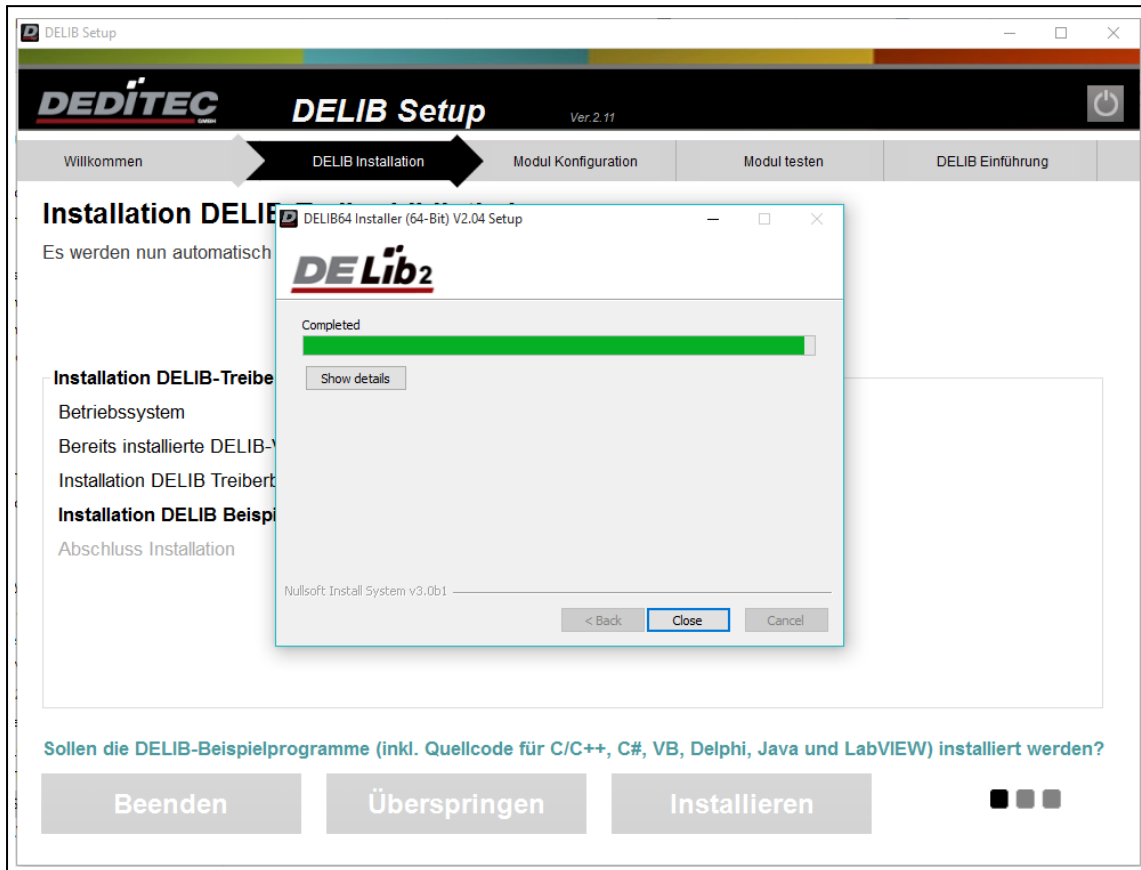




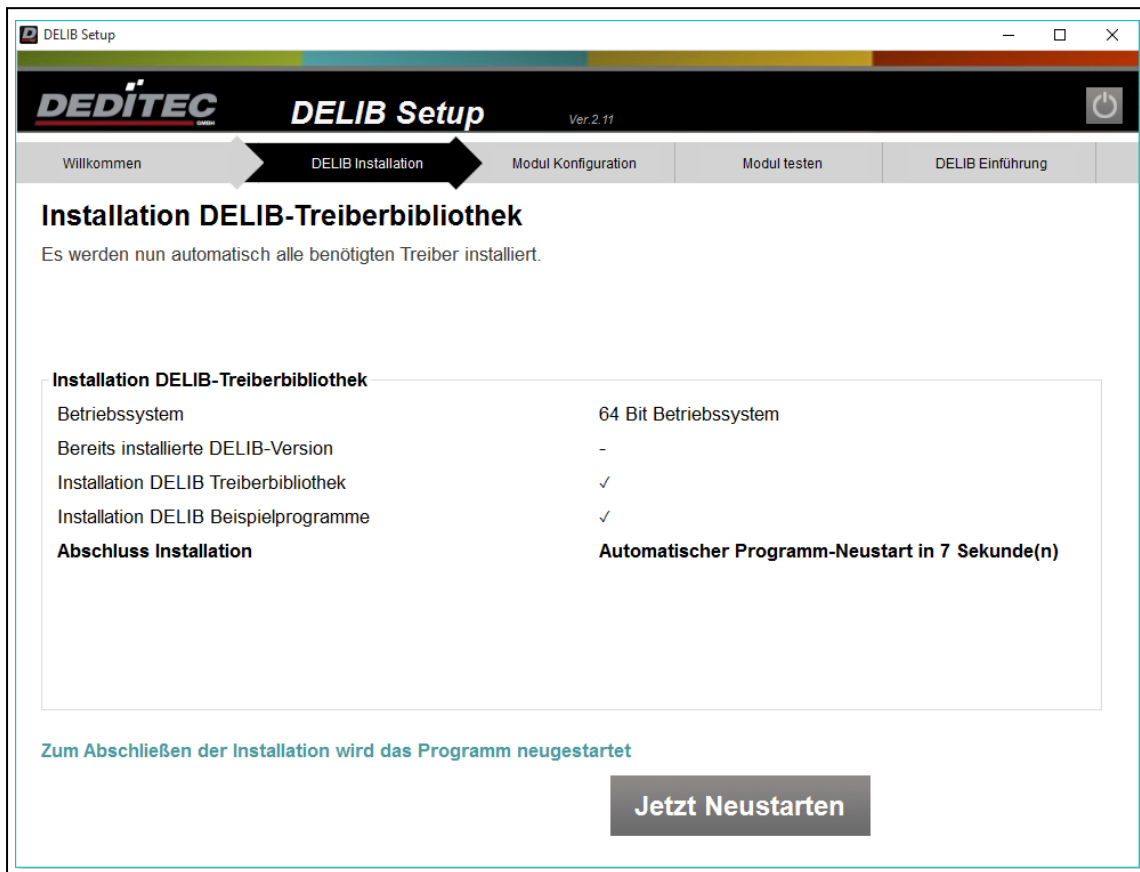
Zusätzlich können Sie wählen, ob Sie die DELIB Beispielprogramme installieren.



Installationsfortschritt der DELIB Beispielprogramme.



Für den Abschluss der Installation wird das Programm neu gestartet. Im nächsten Schritt, wird mit dem DELIB Configuration Utility das Produkt konfiguriert und getestet.



### **4.2.3. DELIB Configuration Utility**

#### **4.2.3.1. Einführung**

Das DELIB Configuration Utility ermöglicht die Konfiguration der Ethernet-, CAN- oder seriellen Schnittstelle eines Produktes.

Die Konfiguration ist für die erste Inbetriebnahme erforderlich.

Ausgenommen sind Produkte mit USB-Schnittstelle. Diese müssen nur konfiguriert werden, falls Sie mehrere identische USB Produkte an einem PC betreiben möchten.

Das DELIB Configuration Utility ist in der Installation der DELIB Treiberbibliothek enthalten.

#### **Standardpfad:**

32-Bit: C:\Program Files (x86)\DEDITEC\DELIB\programs\delib-configuration-utility.exe

64-Bit: C:\Program Files\DEDITEC\DELIB64\programs\delib-configuration-utility\_x64.exe

Sie können das DELIB Configuration Utility auch über das Startmenü unter "Alle Programme" → "DEDITEC" → "DELIB Configuration Utility" öffnen.

#### 4.2.3.2. Neue Konfiguration erstellen oder vorhandene Konfiguration bearbeiten

Für eine neue Konfiguration wählen Sie in der linken Auswahlbox unter "Neues Modul" die gewünschte Schnittstelle aus.

Möchten Sie eine bestehende Konfiguration bearbeiten, finden Sie rechts die Auswahlbox der vorhandenen Konfigurationen.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar includes the DEDITEC logo, the application name, version 'Ver.2.11', and 'DELIB 2.101 (64 Bit)'. The main menu has five tabs: 'Modul Auswahl' (active), 'Modul Konfiguration', 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The 'Modul Konfiguration' section is titled 'Modul Konfiguration' and includes a 'Switch to english version' button. Below the title, there are two columns: 'Neues Modul' and 'Konfigurierte Module'. The 'Neues Modul' column has a 'Schnittstelle' label and a list of radio buttons for 'USB', 'ETH', 'CAN', and 'SER'. The 'Konfigurierte Module' column has a 'Module' label and a list showing 'RO-ETH (1)' and 'RO-SER (0)'. At the bottom, there is a prompt 'Wählen Sie die Schnittstelle oder ein bereits konfiguriertes Modul aus' and two buttons: 'Beenden' and 'Weiter'.

DELIB Configuration Utility

**DEDITEC** **DELIB Configuration Utility** Ver.2.11 **DELIB 2.101** (64 Bit)

Modul Auswahl | Modul Konfiguration | Konfiguration testen | Modul testen | DELIB Einführung

### Modul Konfiguration

Switch to english version

Mit dem DELIB Configuration Utility lassen sich DEDITEC Module in wenigen Schritten konfigurieren.  
Um ein neues Modul zu konfigurieren, wählen Sie bitte die Schnittstelle (links) aus.  
Um eine bereits erstellte Konfiguration für ein Modul zu editieren, wählen Sie das entsprechende Modul (rechts) aus.

Neues Modul	Konfigurierte Module
<p>Schnittstelle</p> <ul style="list-style-type: none"><li><input type="radio"/> USB</li><li><input type="radio"/> ETH</li><li><input type="radio"/> CAN</li><li><input type="radio"/> SER</li></ul>	<p>Module</p> <ul style="list-style-type: none"><li>RO-ETH (1)</li><li>RO-SER (0)</li></ul>

Wählen Sie die Schnittstelle oder ein bereits konfiguriertes Modul aus

**Beenden** **Weiter**

#### 4.2.3.2.1. Modul Konfiguration USB

Die Konfiguration von USB-Modulen ist nur nötig, um mehrere Module einer USB-Produktfamilie (z.B. 2x USB-RELAIS-8) in einem System verwenden zu können.

Befindet sich nur ein USB-Modul, oder mehrere USB-Module aus unterschiedlichen Produktfamilien (z.B. RO-USB-016 und USB-RELAIS-8) im System, ist keine Konfiguration nötig, da die Produkte über die Modul-ID eindeutig identifizierbar sind.

Nach einem Klick auf das Produkt, welches Sie konfigurieren möchten, werden Ihnen rechts alle möglichen Einstellungen angezeigt.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar includes the DEDITEC logo, the application name, version 'Ver. 2.19', and 'DELIB 2.19 (64 Bit)'. The main menu has five tabs: 'Modul Auswahl', 'Modul Konfiguration' (which is active), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The active tab displays the title 'Konfiguration Module mit USB Schnittstelle' and a manual icon. Below this, there is explanatory text about changing the module number. A table titled 'Gefundene USB-Module' lists one module: 'USB-OPT/REL-8' with number '0'. To the right of the table are input fields for 'Aktuelle Modul-Nr.' (0), 'Neue Modul-Nr.' (1), and 'Übertragungsversuche (für alle USB-Module)' (5). A 'Neue Modul-Nr. setzen' button is located between the 'Neue Modul-Nr.' and 'Übertragungsversuche' fields. At the bottom of the window are three large buttons: 'Hauptmenü', 'Test', and 'Fertig'.

Module	Nr
USB-OPT/REL-8	0

Aktuelle Modul-Nr.

Neue Modul-Nr.

Übertragungsversuche (für alle USB-Module)

Die "Aktuelle Modul-Nr" bezieht sich auf die im Modul gespeicherte Modul Nummer. Diese Nummer dient zur Identifikation und muss für identische USB-Produkte unterschiedlich konfiguriert werden.

Mit dem Punkt "Neue Module-Nr" kann dem Produkt eine neue Nummer zwischen 0 und 255 zugewiesen werden. Im Auslieferungszustand haben alle Produkte die Modul-Nummer 0.

Mit "Neue Module-Nr. setzen" wird die aktuell ausgewählte Neue Modul-Nummer auf das Modul geschrieben

Über die Auswahlbox "Übertragungsversuche (für alle USB-Module)" können Sie festlegen, wie oft die DELIB im Falle eines Fehlers versucht mit dem Modul zu kommunizieren.

#### 4.2.3.2.1.1. Beispiel zur Konfiguration identischer USB-Module

Um mehrere identische USB-Module (USB-Module mit gleicher Modul-ID) in einem System verwenden zu können, muss jedem Modul mit dem DELIB Configuration Utility eine eindeutige Modul-Nr. zugeordnet werden.

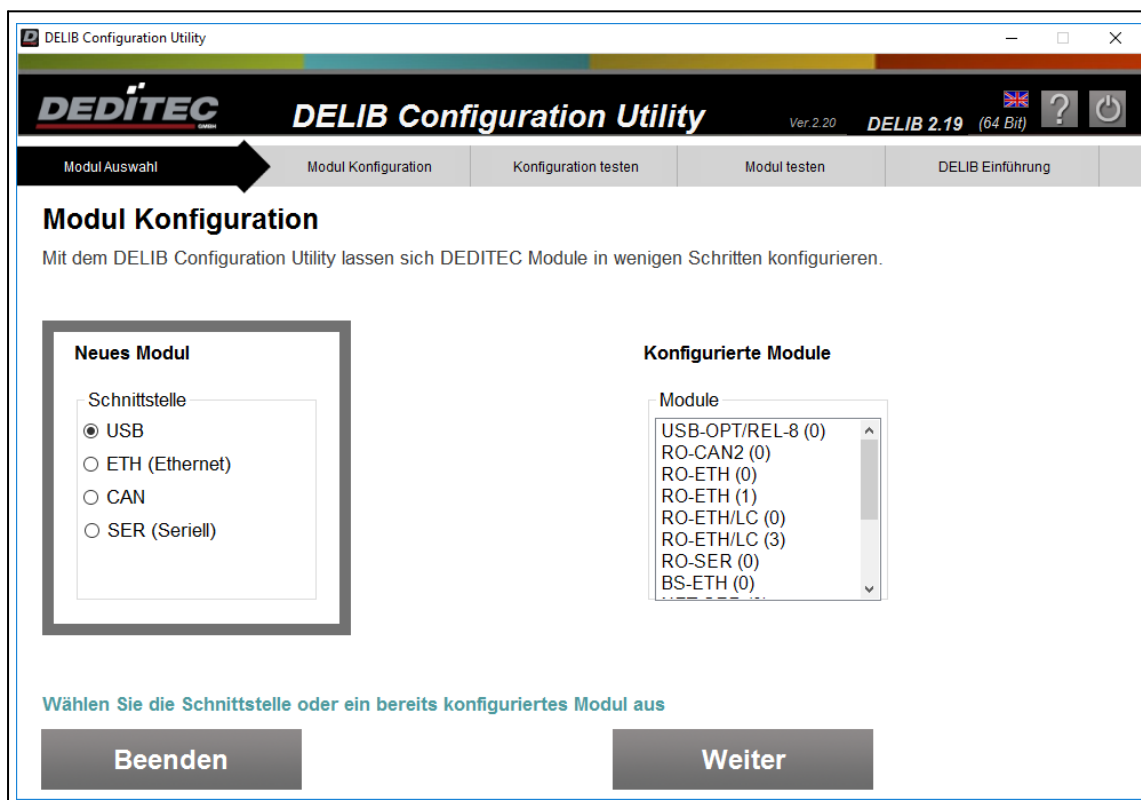
Befindet sich nur ein USB-Modul, oder mehrere USB-Module mit unterschiedlicher Modul-ID (z.B. RO-USB-016 und USB-RELAIS-8) im System, ist keine Konfiguration nötig, da die Produkte über die ID eindeutig identifizierbar sind.

Folgendes Beispiel zeigt die Konfiguration von zwei USB-OPTOIN-8 Modulen im gleichen System.

#### Schritt 1

Verbinden Sie zunächst nur ein USB-OPTOIN-8 mit dem PC und starten Sie das DELIB Configuration Utility.

Wählen Sie im linken Bereich die USB-Schnittstelle aus und klicken auf "Weiter".



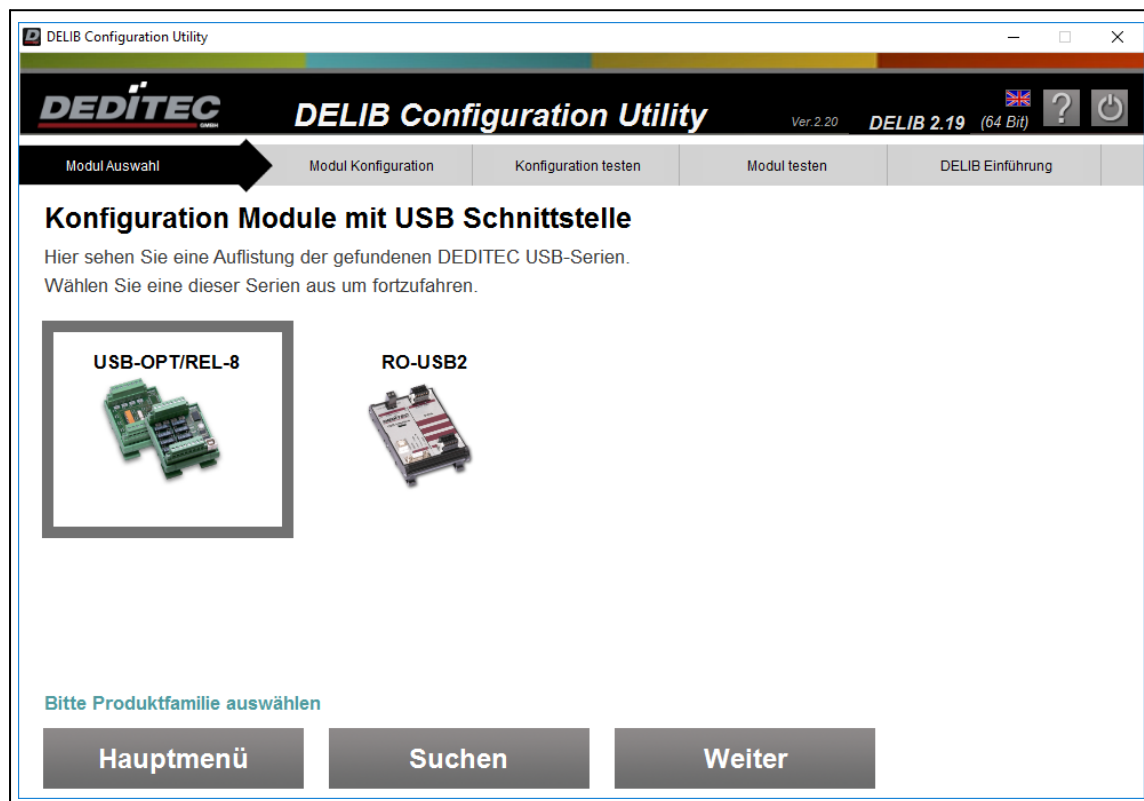


## Schritt 2

Sind mehrere USB-Module verschiedener DEDITEC-USB-Serien angeschlossen, muss in diesem Schritt die entsprechende Produktfamilie ausgewählt werden.

In diesem Beispiel sind Module der RO-USB2-Serie und USB-OPT/REL-8-Serie angeschlossen.

Dieser Schritt entfällt, wenn die angeschlossenen Module der gleichen Serie angehören.



### Schritt 3

1. Wählen Sie das entsprechende USB-Modul aus.
2. Ändern Sie die Neue Modul-Nr. auf "1". Im Auslieferungszustand ist diese Nummer bereits mit "0" vordefiniert.
3. Mit Neue Modul-Nr. setzen wird die neue Module-Nr. im Modul gespeichert.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The interface has a dark header with the 'DEDITEC' logo, the title 'DELIB Configuration Utility', version 'Ver. 2.20', and 'DELIB 2.19 (64 Bit)'. Below the header is a navigation bar with five tabs: 'Modul Auswahl', 'Modul Konfiguration' (active), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration Module mit USB Schnittstelle'. It contains instructions: 'Hier kann die Modul-Nr. des ausgewählten Moduls geändert werden. Die Modul-Nr. dient zur Unterscheidung mehrerer identischer USB-Module einer USB-Serie.' Below this is a table titled 'Gefundene USB-Module' with two columns: 'Module' and 'Nr.'. The first row shows 'USB-OPT/REL-8' and '0'. Below the table is an 'Aktualisieren' button. To the right of the table are three input fields: 'Aktuelle Modul-Nr.' with value '0', 'Neue Modul-Nr.' with a dropdown menu showing '1', and 'Übertragungsversuche (für alle USB-Module)' with a dropdown menu showing '5'. A 'Neue Modul-Nr. setzen' button is located between the 'Neue Modul-Nr.' and 'Übertragungsversuche' fields. At the bottom of the window are three buttons: 'Hauptmenü', 'Test', and 'Weiter'.

Module	Nr
USB-OPT/REL-8	0

Aktualisieren

Aktuelle Modul-Nr. 0

Neue Modul-Nr. 1

Neue Modul-Nr. setzen

Übertragungsversuche (für alle USB-Module) 5

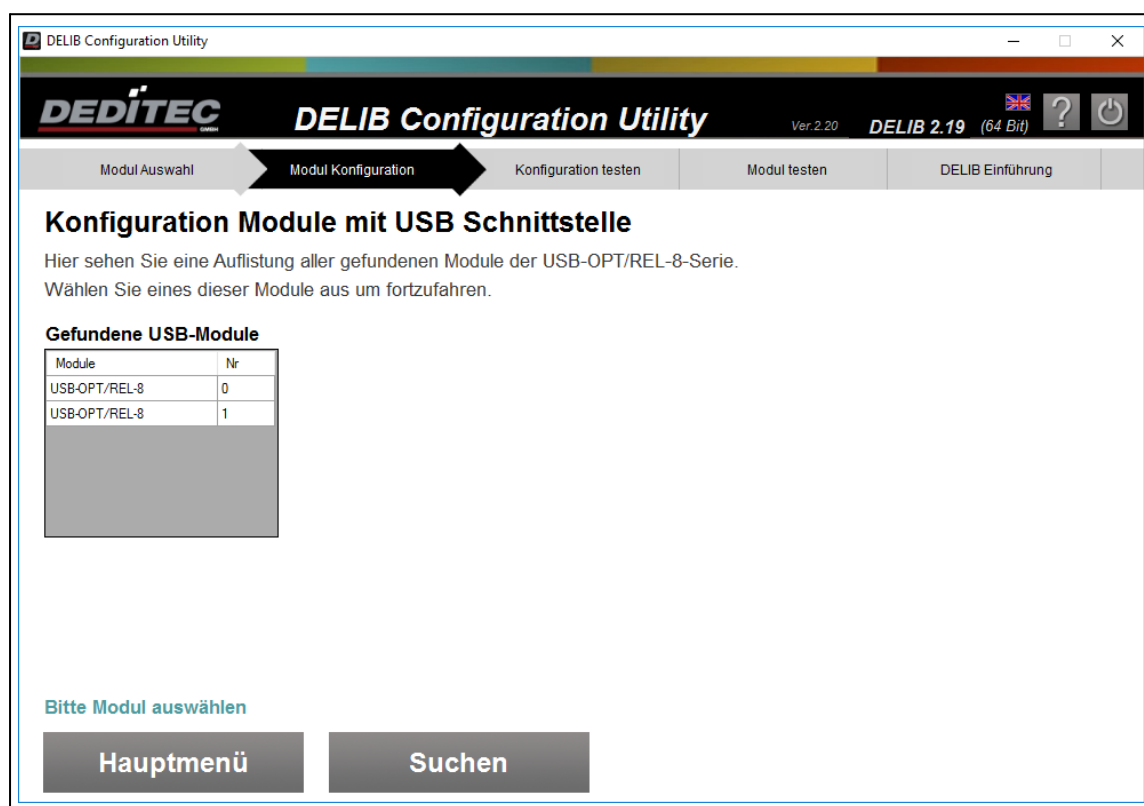
Hauptmenü Test Weiter

#### Schritt 4

Schließen Sie nun zusätzlich das zweite USB-OPTOIN-8 an den PC an.

Da im Auslieferungszustand die Modul-Nr bereits mit "0" vordefiniert und somit unterschiedlich zur Modul-Nr des ersten Moduls (Module-Nr = 1) ist, sind beide Module konfiguriert und betriebsbereit.

Mit Aktualisieren werden nun beide Module angezeigt.



## Schritt 5

Nachfolgend finden Sie Hinweise, was bei der Programmierung der beiden Module beachten werden muss.

Alle Module werden einheitlich mit dem Befehl DapiOpenModule geöffnet. Dieser Befehl ist wie folgt definiert:

```
ULONG DapiOpenModule(ULONG moduleID, ULONG nr);
```

### Ansprechen des USB-OPTOIN-8 mit der NR 0

```
ulong handle;  
handle = DapiOpenModule(USB_OPTOIN_8, 0);  
//öffnet das Modul USB-OPTOIN-8 mit der NR 0
```

### Ansprechen des USB-OPTOIN-8 mit der NR 1

```
ulong handle;  
handle = DapiOpenModule(USB_OPTOIN_8, 1);  
//öffnet das Modul USB-OPTOIN-8 mit der NR 1
```

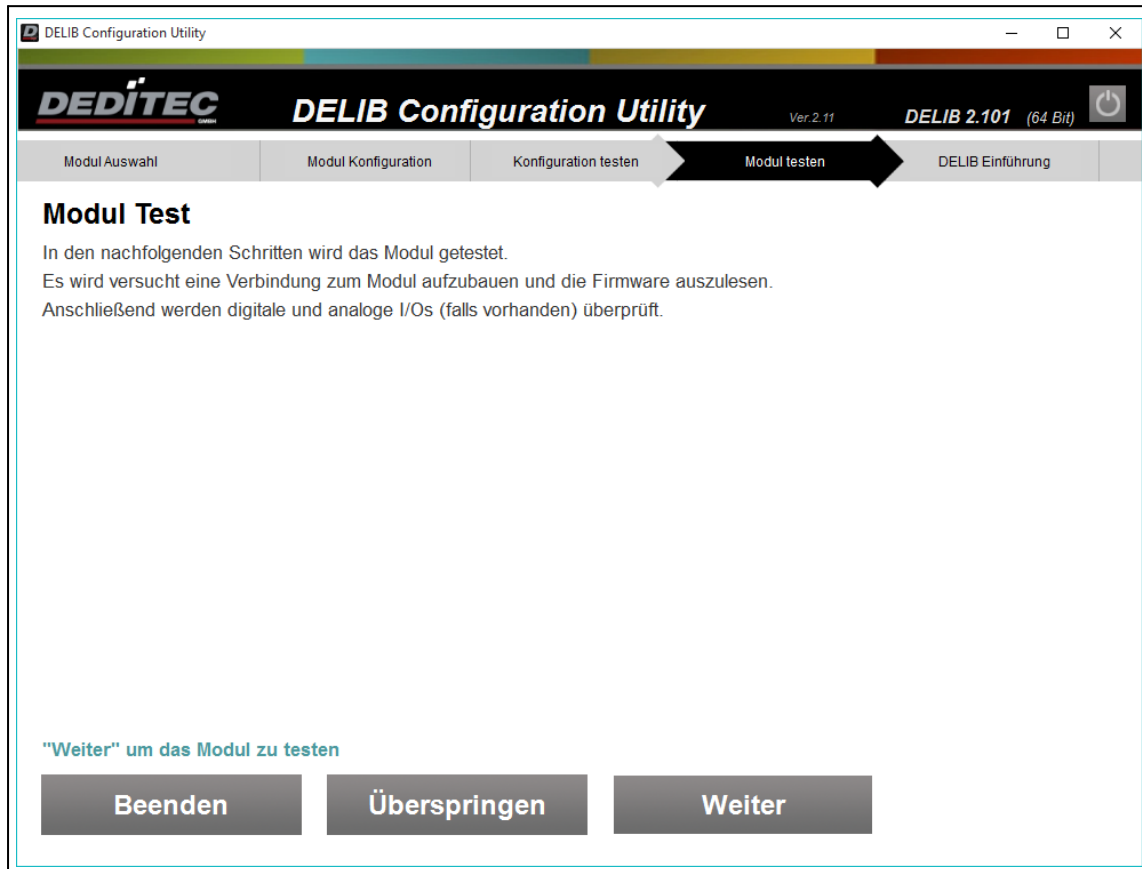
### Hinweis:

Alle Module haben als Werkseinstellung die NR "0".

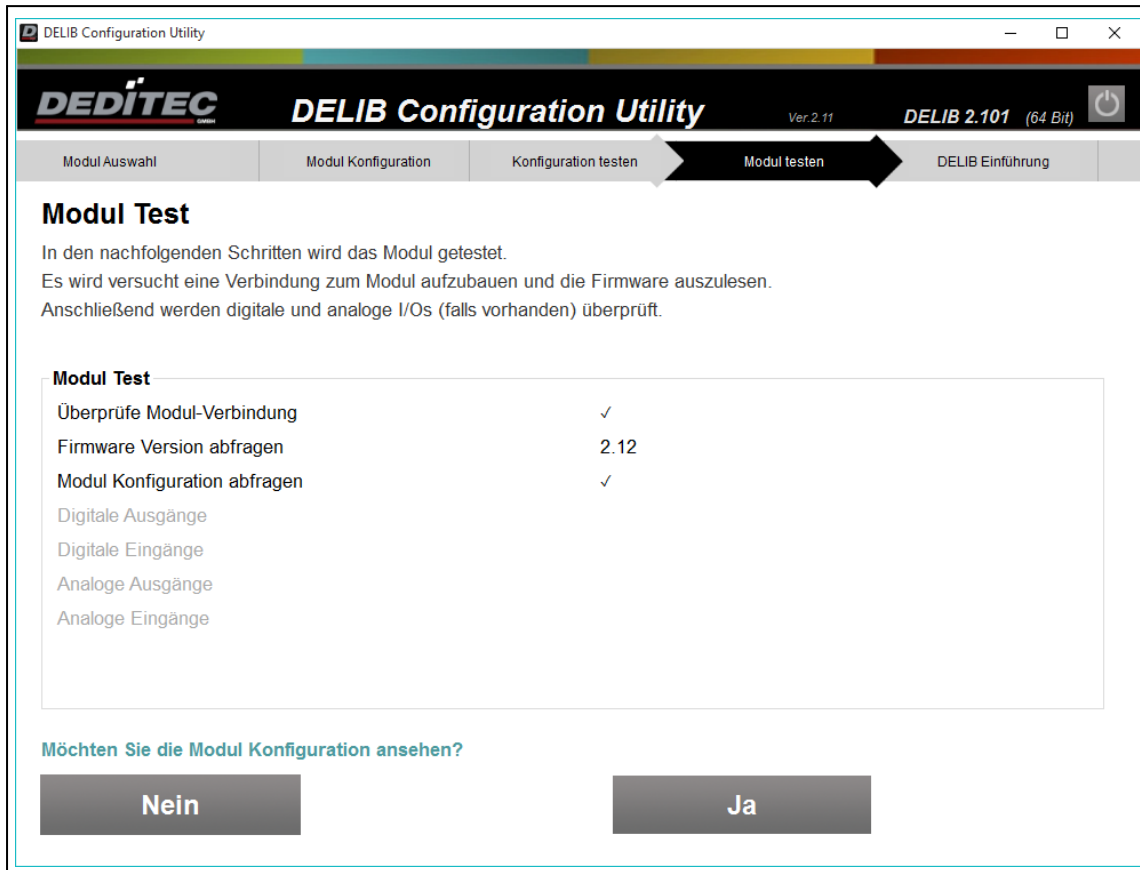
Wenn Sie mehrere identische USB-Module verwenden möchten, muss den Modulen nacheinander, eine eindeutige NR zugeordnet werden.

#### 4.2.3.3. Modul testen

Nachdem die Konfiguration der Schnittstelle durchgeführt wurde, kann anschließend das Produkt getestet werden.



Test der Firmware und Anzeige der Modul-Info.



Die Modul-Info zeigt alle Eigenschaften des Produktes. Neben der Anzahl der vorhandenen I/Os werden auch die unterstützten Software-Features angezeigt.

DT\_ModuleInfo
×

General		Digital I/O		Analog I/O		Special	
SW_FEATURE_1	e3373007	Digital Inputs	8	Analog Inputs	16	Stepper	2
HW_INTERFACE_1	01000003	Digital Outputs	8	Analog Outputs	4		
Firmware-Revision	2.11	Digital In-/Outputs	0	Temperature Inputs	4		
Main-Module:	RO	Digital Input FlipFlops	8				
		Digital Input Counter	8				
		Pulse Gen Outputs	0				
		CNT8	0				
		Digital PWM Outputs	0				

Features-General		Features-Digital I/O		Features-Analog I/O		Features-Special	
Supported by FW	OK	DI Commands	OK	DA Commands	OK	Watchdog Commands	-
Dev IO registry error	OK	DI CNT Commands	OK	AD Commands	OK	Stepper Commands	OK
AD FIFO	OK	DI CNT Latch Feature	-	Pt100 Commands	OK		
Set-Clr Bit Commands	OK	DI FF Commands	OK				
EEPROM RN23	-	DO Commands	OK				
EEPROM E2_2K	-	DO Time Commands	OK				
DX1 Mode	-	PWM Commands	-				
Support Channel Names	OK	TTL Commands	-				
HW-INT Supported by FW	OK	PulseGen Commands	-				
ETH	OK	CNT8 Commands	-				
CAN	-	Auto-Off Timeout Commar	OK				
RS232	-						
RS485	-						
USB1	-						
USB2	-						

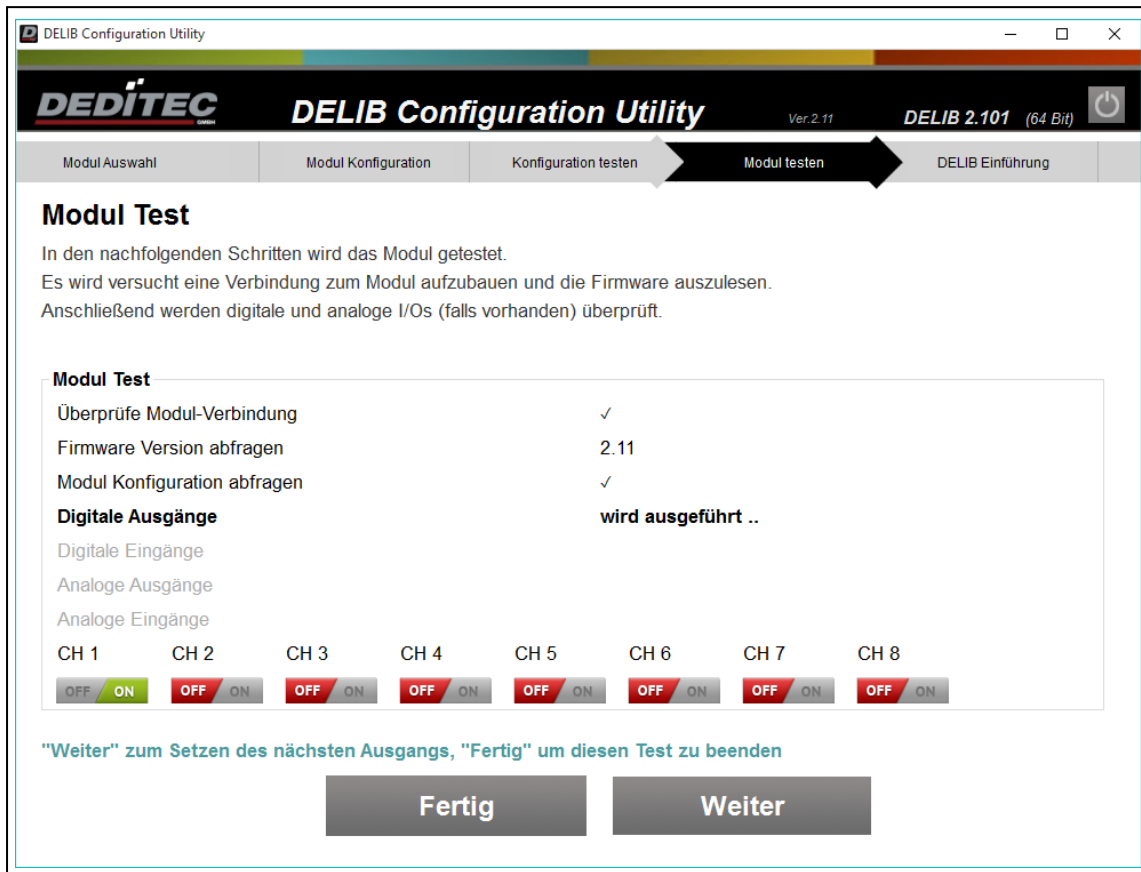
Submodule-Info's

Sub: 0 - RO-AD16DA4  
Sub: 1 - RO-08\_R8  
Sub: 2 - RO-STEPPER2  
Sub: 3 - RO-PT100

FW:2.11  
FW:2.10  
FW:1.29  
FW:1.03

EXIT

Es folgt ein Test der I/Os. In diesem Beispiel werden die digitalen Ausgänge geschaltet.



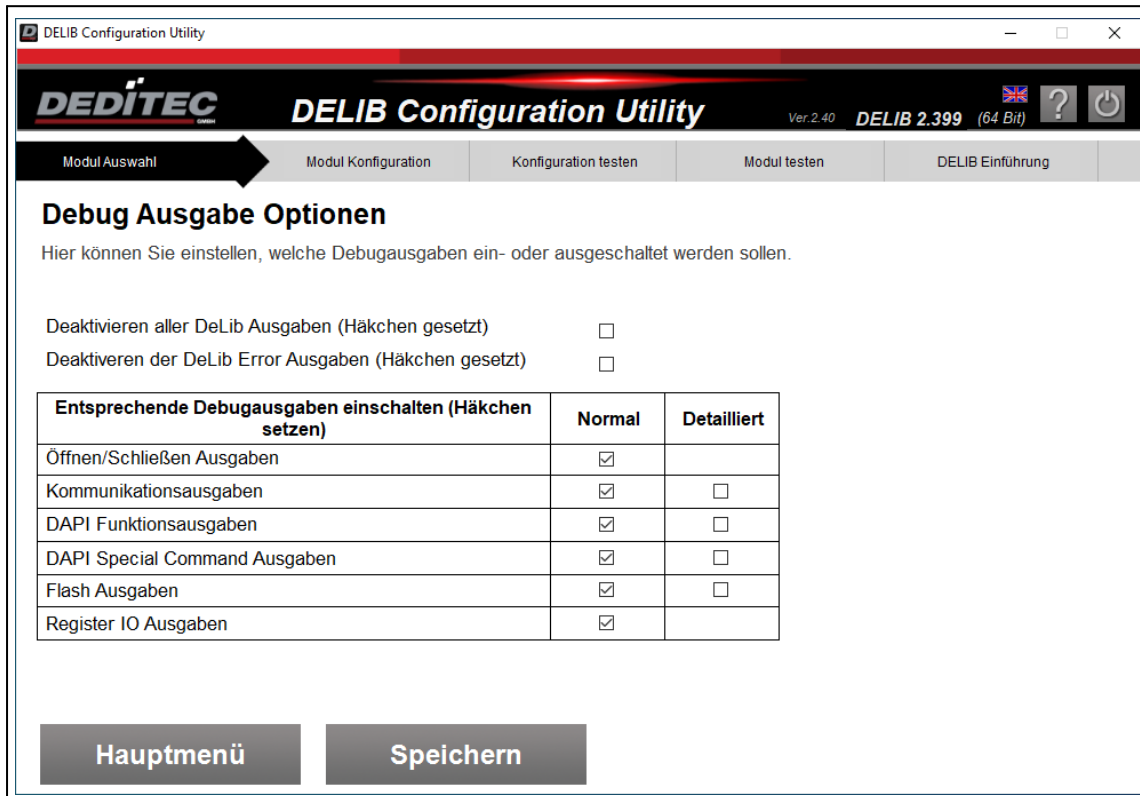
Wurden alle Tests erfolgreich durchlaufen, ist das Produkt einsatzbereit.



#### 4.2.3.4. Debug Optionen einstellen

Über den Knopf "Debug Ausgabe Optionen" gelangen Sie in das folgende Optionsmenü.

Dort können Sie einstellen, welche Debugausgaben Sie ein- oder ausschalten möchten.



The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The main header features the 'DEDITEC' logo, the title 'DELIB Configuration Utility', and version information 'Ver. 2.40 DELIB 2.399 (64 Bit)'. Below the header is a navigation bar with five tabs: 'Modul Auswahl' (selected), 'Modul Konfiguration', 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Debug Ausgabe Optionen' and contains the instruction: 'Hier können Sie einstellen, welche Debugausgaben ein- oder ausgeschaltet werden sollen.' Below this are two checkboxes: 'Deaktivieren aller DeLib Ausgaben (Häkchen gesetzt)' and 'Deaktivieren der DeLib Error Ausgaben (Häkchen gesetzt)', both currently unchecked. A table follows, titled 'Entsprechende Debugausgaben einschalten (Häkchen setzen)'. The table has three columns: the first lists the debug output types, the second is 'Normal', and the third is 'Detailliert'. The 'Normal' column has checkboxes checked for all listed items, while the 'Detailliert' column has checkboxes unchecked for all listed items. At the bottom of the window are two buttons: 'Hauptmenü' and 'Speichern'.

Entsprechende Debugausgaben einschalten (Häkchen setzen)	Normal	Detailliert
Öffnen/Schließen Ausgaben	<input checked="" type="checkbox"/>	
Kommunikationsausgaben	<input checked="" type="checkbox"/>	<input type="checkbox"/>
DAPI Funktionsausgaben	<input checked="" type="checkbox"/>	<input type="checkbox"/>
DAPI Special Command Ausgaben	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Flash Ausgaben	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Register IO Ausgaben	<input checked="" type="checkbox"/>	

#### 4.2.4. Benutzung des Moduleselectors

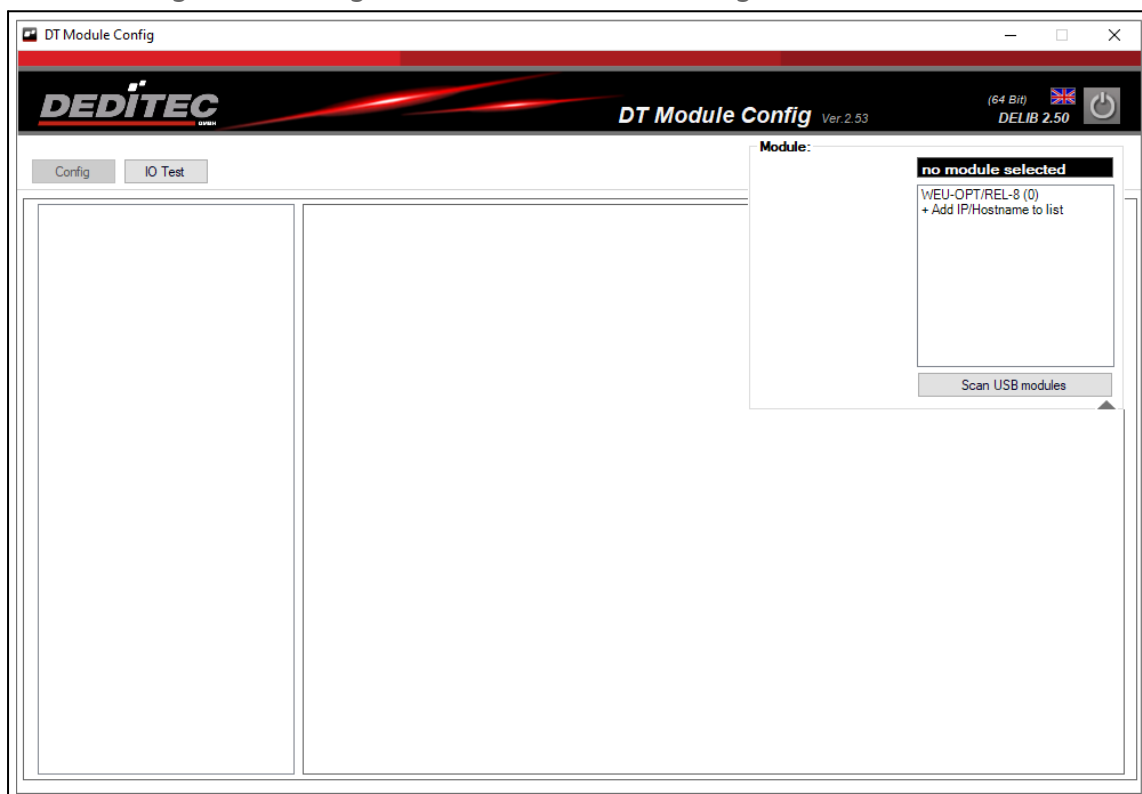
Um unsere Produkte mit der DEDITEC-Software benutzen zu können, müssen diese über den Modul Selector ausgewählt werden.

Je nach Modul, kann dies über verschiedene Schnittstellen bewerkstelligt werden.

##### 4.2.4.1. via USB

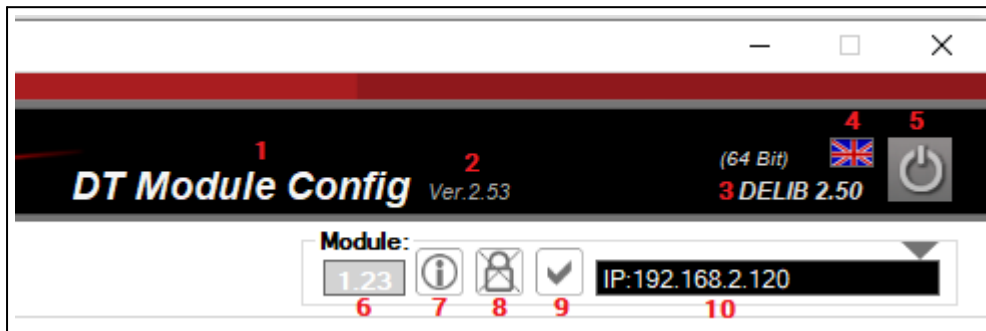
Haben Sie das Modul über die USB-Schnittstelle mit dem PC verbunden, kann das Modul direkt über einen Klick auf den Modul Selector in der rechten oberen Ecke ausgewählt werden.

Anschließend können Sie im Netzwerkbereich unter LAN - Konfiguration oder WiFi - Konfiguration die gewünschte Netzwerkkonfiguration vornehmen.



#### 4.2.4.2. Modul Info

Bei einer erfolgreichen Verbindung mit dem Modul werden nun im Bereich des Modul Selector verschiedene Information, wie unten beschrieben, dargestellt.



#### Beschreibung:

1. Zeigt den Namen der verwendeten DEDITEC Software an
2. Zeigt die aktuell verwendete Versionsnummer der Software an
3. Zeigt die aktuell verwendete DELIB Version an
4. Durch einen Klick auf das Fahnenymbol lässt sich die Sprache zwischen deutsch und englisch ändern
5. Schließt das Programm
6. Zeigt die aktuell verwendete Firmware Ihres Moduls an
7. Durch einen Klick auf die Informationsschaltfläche, öffnet sich das Informationsfenster des Moduls (s. Bild unten)
8. Zeigt an, ob eine ver- oder entschlüsselte Kommunikation mit dem Modul stattfindet
9. Zeigt den Kommunikationsstatus mit dem Modul an
10. Je nach Verbindungsart wird hier die IP oder der Boardname des aktuell verwendeten Moduls angezeigt

## Informationsfenster

Je nach angeschlossenem Modul werden hier Informationen zu dem verwendeten Interface und den Submodulen angezeigt.

Unter Anderem können Sie hier die Anzahl der angeschlossenen Ein- bzw. Ausgänge einsehen und welche DEDITEC Befehle unterstützt werden.

The screenshot shows a window titled "DT\_ModuleInfo" with a close button (X) in the top right corner. The window displays information for a module, organized into four main sections: General, Digital I/O, Analog I/O, and Special. Each section has a list of parameters and their values.

General		Digital I/O		Analog I/O		Special	
SW_FEATURE_1	0300a0c5	Digital Inputs	0	Analog Inputs	0	Stepper	0
HW_INTERFACE_1	00000103	Digital Outputs	8	Analog Outputs	0		
Firmware-Revision	1.23	Digital In-/Outputs	0	Temperature Inputs	0		
Main-Module:	-	Digital Input FlipFlops	0				
		Digital Input Counter	0				
		Pulse Gen Outputs	0				
		CNT8	0				
		Digital PWM Outputs	0				

Features-General		Features-Digital I/O		Features-Analog I/O		Features-Special	
Supported by FW	OK	DI Commands	-	DA Commands	-	Watchdog Commands	-
Dev IO registry error	OK	DI CNT Commands	-	AD Commands	-	Stepper Commands	-
RO-AD FIFO	-	DI CNT Latch Feature	-	Pt100 Commands	-		
NET-Software FIFO	-	DI FF Commands	-				
Set-Clr Bit Commands	OK	DO Commands	OK				
EEPROM RN23	-	DO Time Commands	-				
EEPROM E2_2K	-	PWM Commands	-				
DX1 Mode	-	TTL Commands	-				
Support Channel Names	-	PulseGen Commands	-				
HW-INT Supported by FW	OK	CNT8 Commands	-				
ETH	OK	Auto-Off Timeout	OK				
CAN	-	Auto-Off Timeout Mask	OK				
RS232/485	-	FTDI Userbyte 6	0x0				
USB1	-						
USB2	-						

An "EXIT" button is located in the bottom right corner of the window.

In diesem Beispiel wurde ein WEU-RELAIS-8 aus unserer Startet-Serie mit 8 digitalen Ausgängen über Ethernet angeschlossen.

#### **4.2.5. DELIB Module Config**

Das Module Config ist eine neue Anwendung zur Konfiguration und zum Testen unserer Produkte. Dieses Programm ist im Installationspaket unserer DELIB Treiberbibliothek enthalten.

##### **4.2.5.1. Modul Konfigurationen**

Im Konfigurations-Bereich können Konfigurationseinstellungen des Moduls eingesehen oder geändert werden.

#### 4.2.5.1.1. Modul-Infoseite

Mit dem Module Config lässt sich Ihr WEU-Modul nicht nur schnell und einfach konfigurieren, Sie können sich auch alle wichtigen Modulinformationen auf nur einen Blick anzeigen lassen.

**Info**  
Hier finden Sie Informationen zu Ihren Moduleigenschaften

Modul-Name	WEU-08-R8
Modul-ID	40 (dezimal) / 0x0028 (hex)
Firmware-Version	Ver. 1,23

Form für Module Overview (Ver. 1.01)

#### Modul-Name

Zeigt den Namen des aktuell verwendeten DEDITEC Modules an.

#### Modul-ID

Zeigt die ID Ihres verwendeten Moduls an. Diese wird für das Programmieren eigener Software mit DEDITEC Befehlen benötigt.

#### Firmware-Revision

Zeigt die aktuelle auf dem Modul installierte Firmware-Version an.

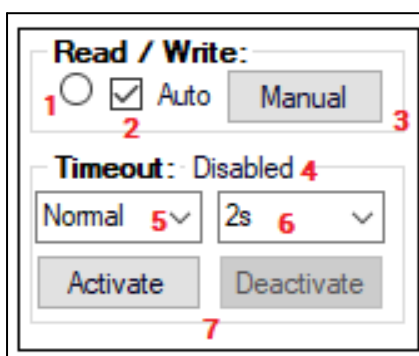
#### 4.2.5.2. I/O-Test

Im I/O Bereich können Tests an den Modulen vorgenommen werden.

##### 4.2.5.2.1. Timeout Test-Funktion

Im "Read/Write" Bereich können Einstellungen am Timeout vorgenommen werden.

Mit diesen Funktionen lässt sich ein Timeout-Fall auslösen.



- 1 Das Feld zeigt durch wiederholtes Blinken an, ob eine Verbindung zum Modul besteht. Bleibt das Feld leer, ist die Kommunikation unterbrochen.
- 2 Durch das Entfernen des Häkchens, wird die Verbindung unterbrochen und somit ein Timeout-Fall ausgelöst. Ein erneutes Setzen des Hakens stellt die Verbindung wieder her.
- 3 Da die Benutzeroberfläche in einem Timeout-Fall nicht automatisch aktualisiert wird, kann dies mit einem Klick auf den "Manual" Knopf ausgelöst werden.
- 4 Hier wird der aktuelle Timeout-Status angezeigt.
- 5 Hier kann der gewünschte Timeout-Mode eingestellt werden. Mehr Informationen siehe Kapitel: **DapiSpecialCMDTimeout**.
- 6 Hier können Sie die Zeit einstellen, in welcher der Timeout ausgelöst werden soll.
- 7 "Activate" aktiviert den Timeout. Durch "Deactivate" wird er deaktiviert.

#### 4.2.5.2.2. Digital In

Hier finden Sie Informationen zu den Digitalen Eingängen, sowie den Eingangszähler und den FlipFlop-Filter.

### Digital In [0 .. 15]

Hier können Sie den Status der digitalen Eingangskanäle ablesen

	State on/off	Counter	FlipFlop		State on/off	Counter	FlipFlop
Kanal 0	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 8	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 1	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 9	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 2	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 10	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 3	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 11	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 4	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 12	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 5	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 13	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 6	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 14	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>
Kanal 7	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>	Kanal 15	<input type="checkbox"/>	<input type="text" value="0"/>	<input type="checkbox"/>

☐ Read with reset

☐ Auto Refresh

FormIODigitalIn (Ver. 1.10)

#### State on/off

Zeigt den aktuellen Zustand der einzelnen Eingangskanäle.

#### Counter

Zeigt die Zählerstände der Eingangszähler an.

#### FlipFlop

Zeigt die Änderung der Eingangszustände seit dem letzten Auslesen an.

#### Read with reset

Mit dieser Option wird festgelegt, ob die Zähler beim nächsten Lesen zurückgesetzt werden sollen.



#### 4.2.5.2.3. Digital Out

Hier können Sie die einzelnen digitalen Ausgänge Ihres Modules an- und ausschalten.

Eine LED an jedem Ausgangsrelais auf dem Board Ihres Modules, zeigt den aktuellen Status des Ausganges an (LED an = Relais an).

### Digital Out [0 .. 15]

Hier können Sie die digitalen Ausgänge des Modules ein- oder ausschalten

	On / Off		Readback	Timer	
Kanal 0	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 1	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 2	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 3	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 4	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 5	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 6	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 7	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>

	On / Off		Readback	Timer	
Kanal 8	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 9	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 10	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 11	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 12	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 13	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 14	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>
Kanal 15	<input type="button" value="on"/>	<input type="button" value="off"/>	<input type="checkbox"/>	<input type="text" value="0 s"/>	<input type="button" value="set"/>

☒ Invert DO-Timer  
on (data1) / off (data0)

☐ Auto Refresh

FormIODigitalOut (Ver. 1.10)

#### On/Off

Schaltet das jeweilige Ausgangsrelais an oder aus.

#### Readback

Zeigt den aktuellen Status des jeweiligen Relais an (on oder off).

#### Switch all states OFF / Switch all states ON

Mit diesen Knöpfen, lassen sich alle Ausgänge des Modules gleichzeitig an- oder ausschalten.

### **Kanäle schalten mit Timer-Funktion**

(Wird nur angezeigt, wenn es vom Modul unterstützt wird)

Geben Sie im Timer Bereich eine Zeit (in Sekunden) an, nach der die Relais ein oder ausgeschaltet werden sollen. Mit "set" starten Sie den Timer.

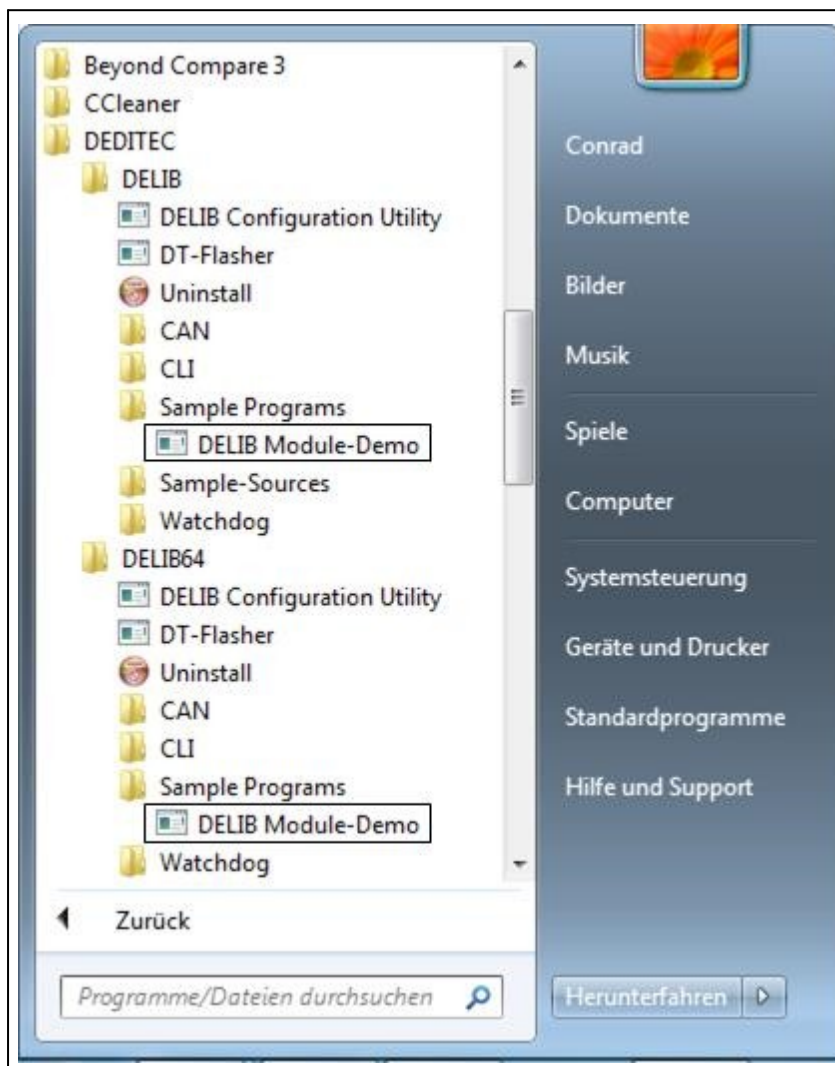
### **Invert DO-Timer**

Ist diese Option aktiviert, wird das Relais nach Ablauf des Timers deaktiviert. Ist diese Option deaktiviert, wird das Relais nach Ablauf des Timers aktiviert.

#### 4.2.6. DELIB Module Demo

Nach Installation der DELIB Treiberbibliothek kann das Programm DELIB Module Demo auf folgendem Weg gestartet werden:

Start → Programme → DEDITEC → DELIB oder DELIB64 → Sample Programs → DELIB Module Demo.

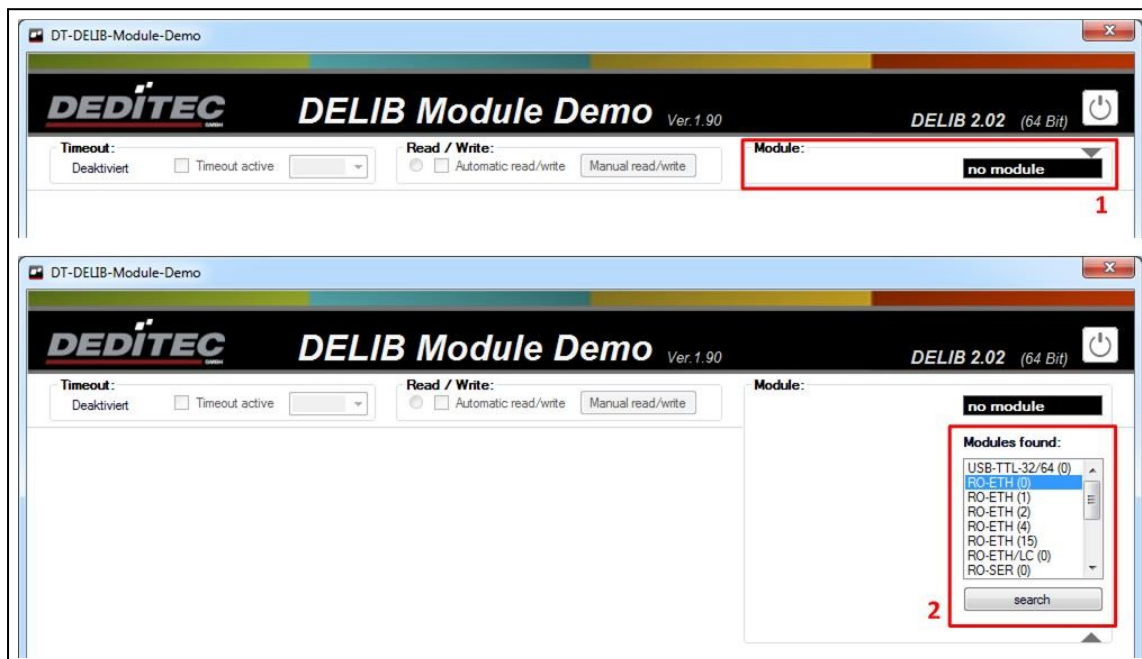


Das Programm DELIB Module Demo ist ein All-in-One Tool mit dem sämtliche I/Os aller Produkte aus unserem S&R Bereich gesteuert und getestet werden können.

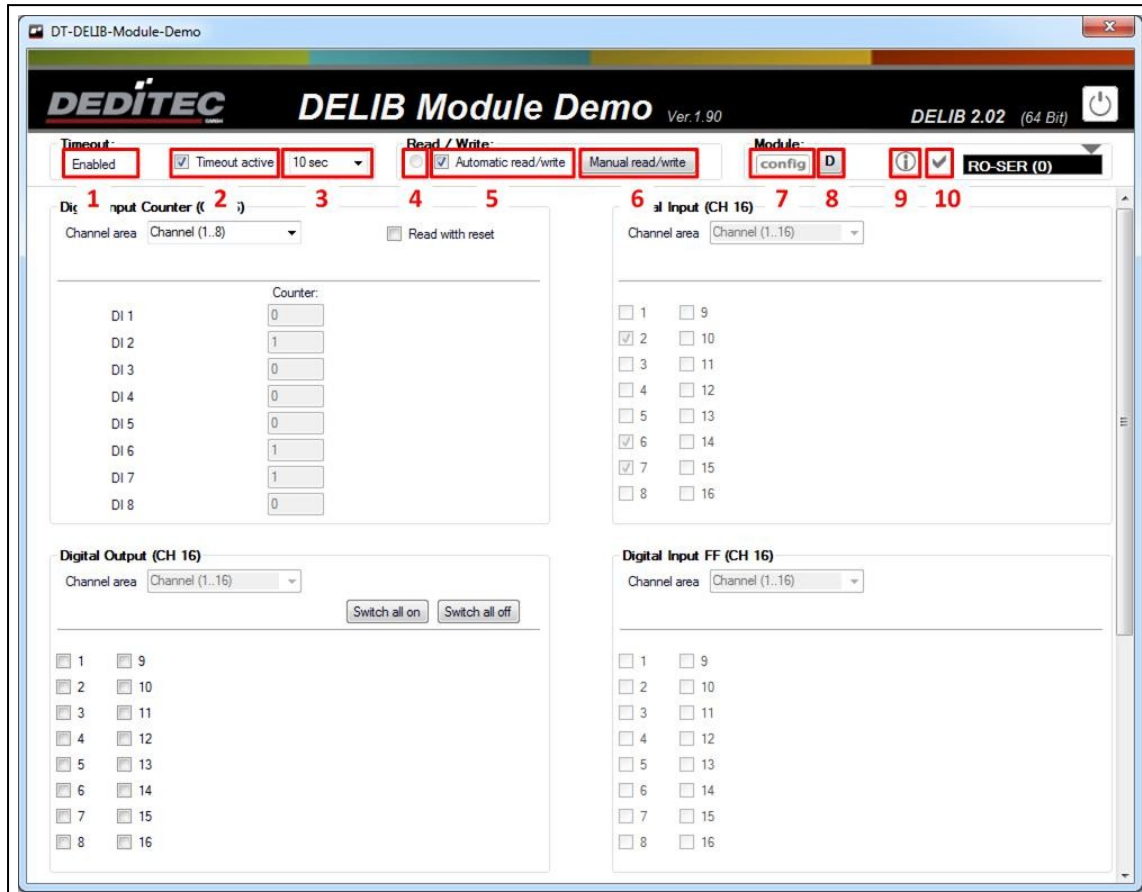
#### 4.2.6.1. Auswahl des Moduls

Bei Programmstart muss ein Modul ausgewählt werden.

1. Klicken Sie den "Module-Selector" an. Sie erhalten eine Auflistung der verfügbaren/angeschlossenen Module.
2. Wählen Sie nun das gewünschte Modul aus.



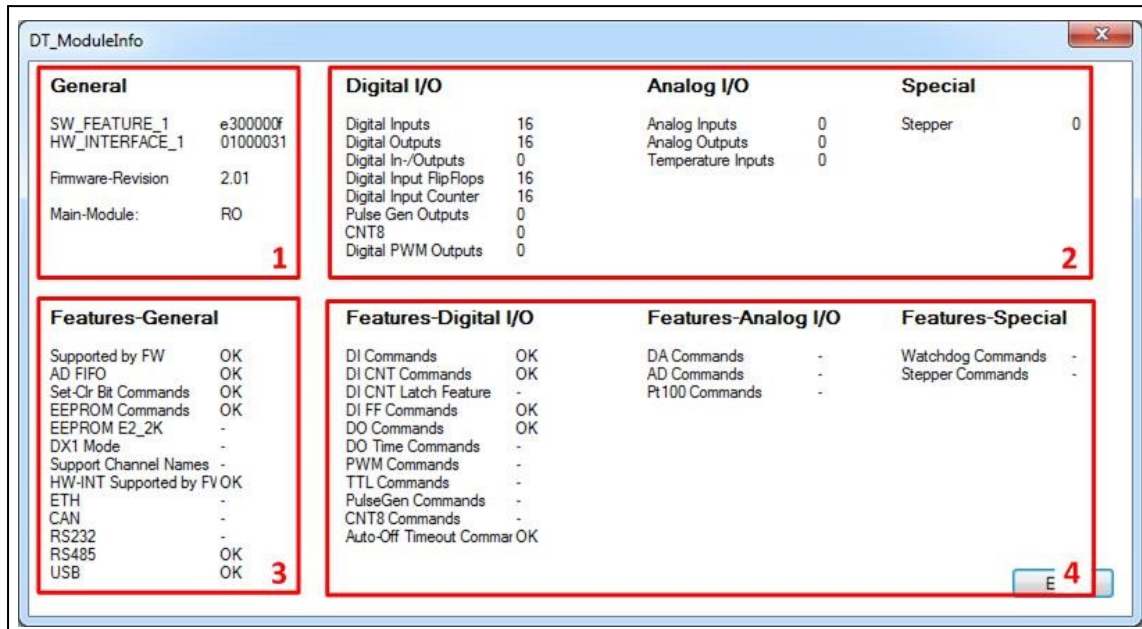
#### 4.2.6.2. Allgemein



1. Timeout-Status ("Disabled", "Enabled" oder "Occured").
2. Aktiviert oder deaktiviert den Timeout-Schutz.
3. Timeout-Zeit für den Timeout-Schutz.
4. Status für "Automatic read/write" (Blinkt, wenn "Automatic read/write" aktiviert ist).
5. Mit dem Haken bei "Automatic read/write" wird festgelegt, ob die Messdaten automatisch gelesen/geschrieben werden sollen.

6. Manuelles Lesen/Schreiben. Nur aktiv, wenn "Automatic read/write" deaktiviert ist.
7. Hier kann das aktuell ausgewählte Modul via DELIB Configuration Utility konfiguriert werden.
8. Sofern vom Modul unterstützt, erhalten Sie hierüber detaillierte Debug-Informationen.

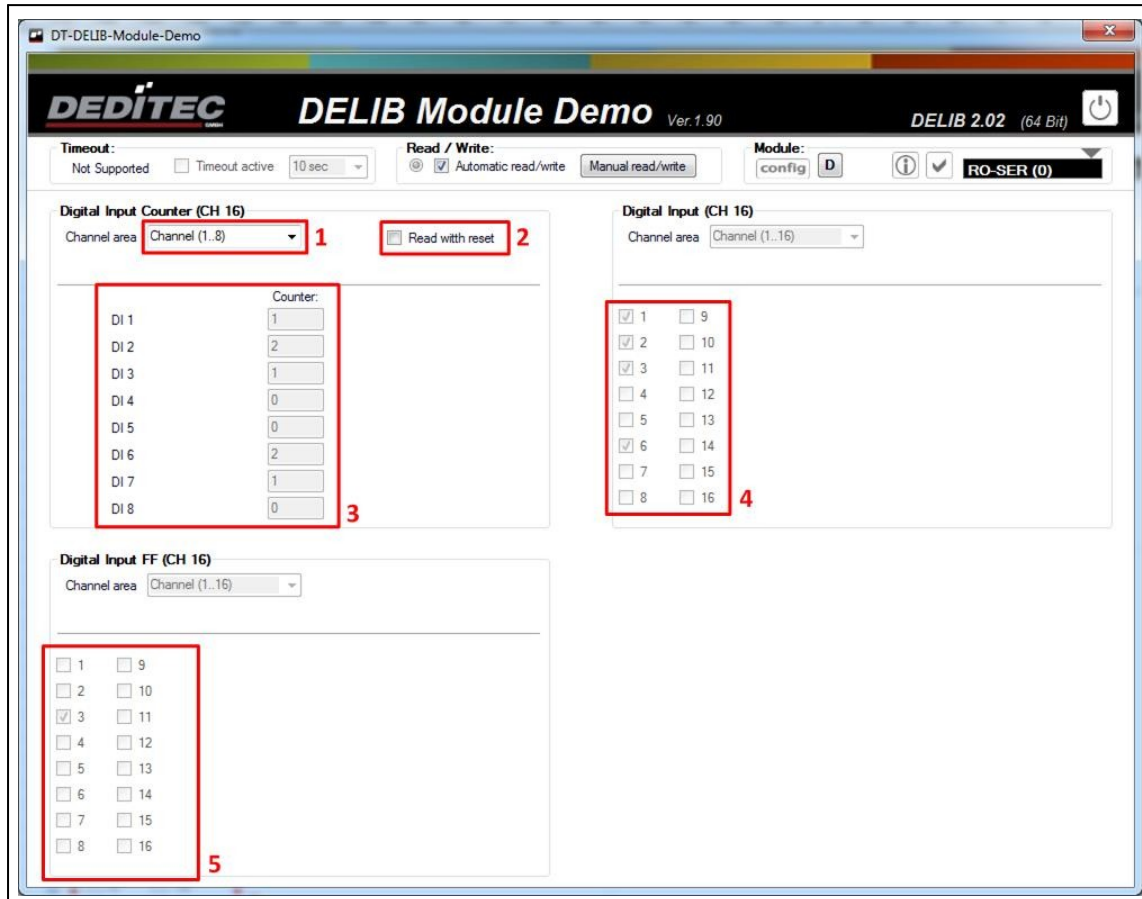
#### 4.2.6.2.1. Module Info



Dieses Beispiel zeigt die erweiterten Informationen des Moduls RO-SER-016-M16.

1. Allgemeine Informationen des ausgewählten Moduls.
2. Anzahl der angeschlossenen I/O-Kanäle.
3. Übersicht der unterstützten Interface DELIB Kommandos.
4. Übersicht der unterstützten I/O DELIB Kommandos.

#### 4.2.6.3. Digital Input

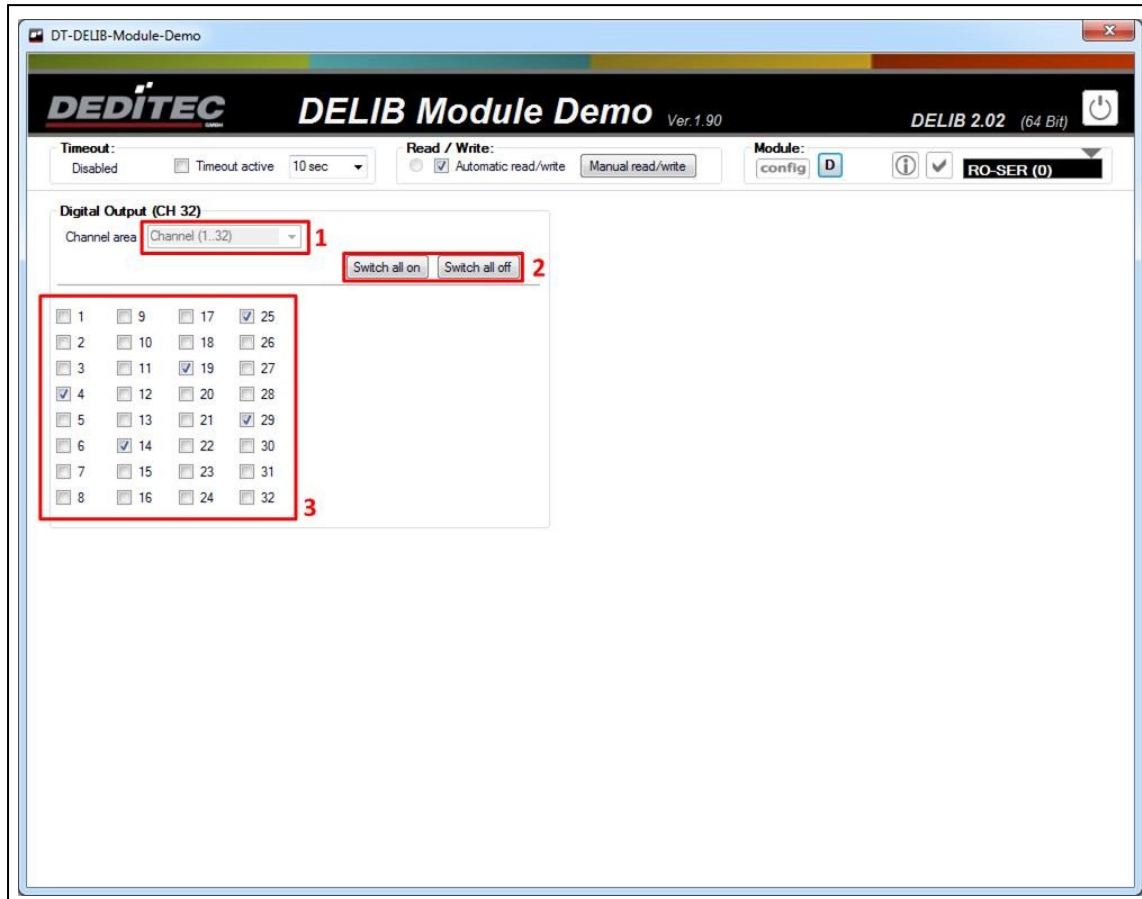


Dieses Beispiel zeigt die digitalen Eingänge eines RO-SER-016 Moduls.

1. Auswahl des Kanal-Bereichs, der angezeigt werden soll.
2. Mit dem Haken bei "Read with reset" wird festgelegt, ob die Zähler beim nächsten Lesen resettet werden.
3. Zählerstände der Eingangszähler.
4. Zustände der Eingänge.
5. Änderung der Eingangszustände (seit dem letzten Auslesen).



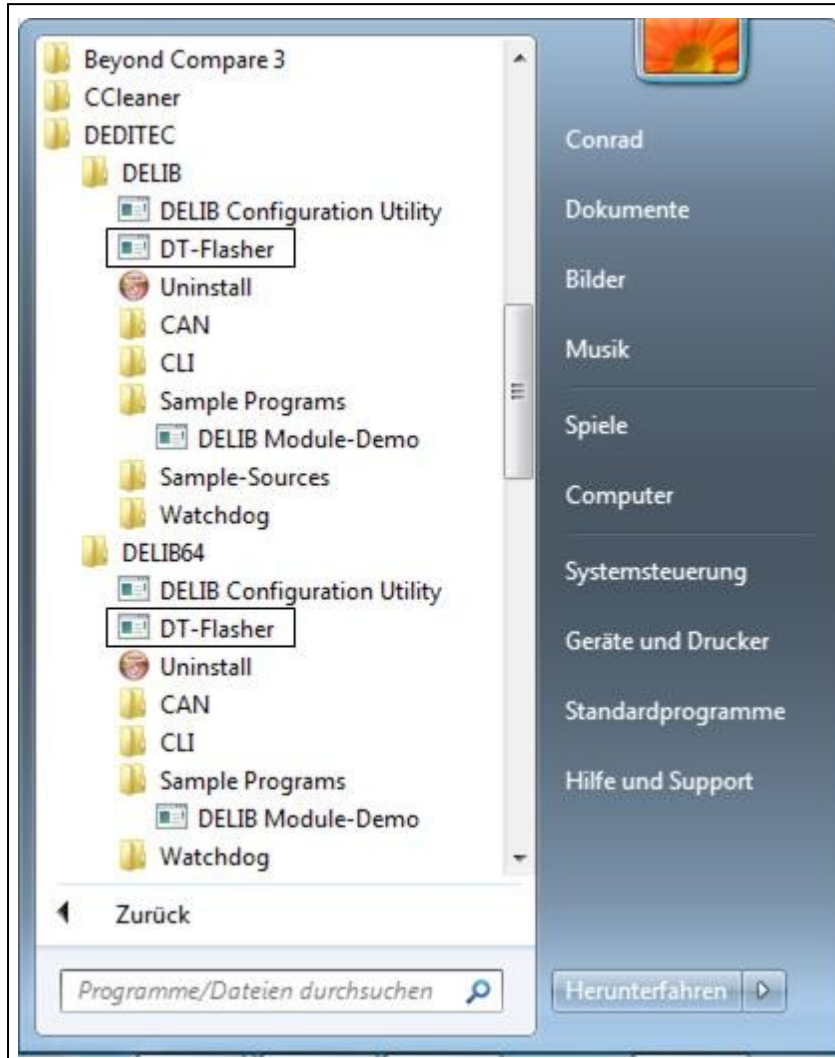
#### 4.2.6.4. Digital Output



Dieses Beispiel zeigt die digitalen Ausgänge eines RO-SER-M32 Moduls.

1. Auswahl des Kanal-Bereichs, der angezeigt werden soll.
2. Hiermit werden alle Ausgänge des aktuellen Kanal-Bereichs ein- bzw. ausgeschaltet.
3. Hier können bestimmte Ausgänge gezielt ein- bzw. ausgeschaltet werden.

#### 4.2.7. DT-Flasher



Nach Installation der DELIB Treiberbibliothek kann das Programm DT-Flasher auf folgendem Weg gestartet werden:

Start → Programme → DEDITEC → DELIB oder DELIB64 → DT-Flasher.

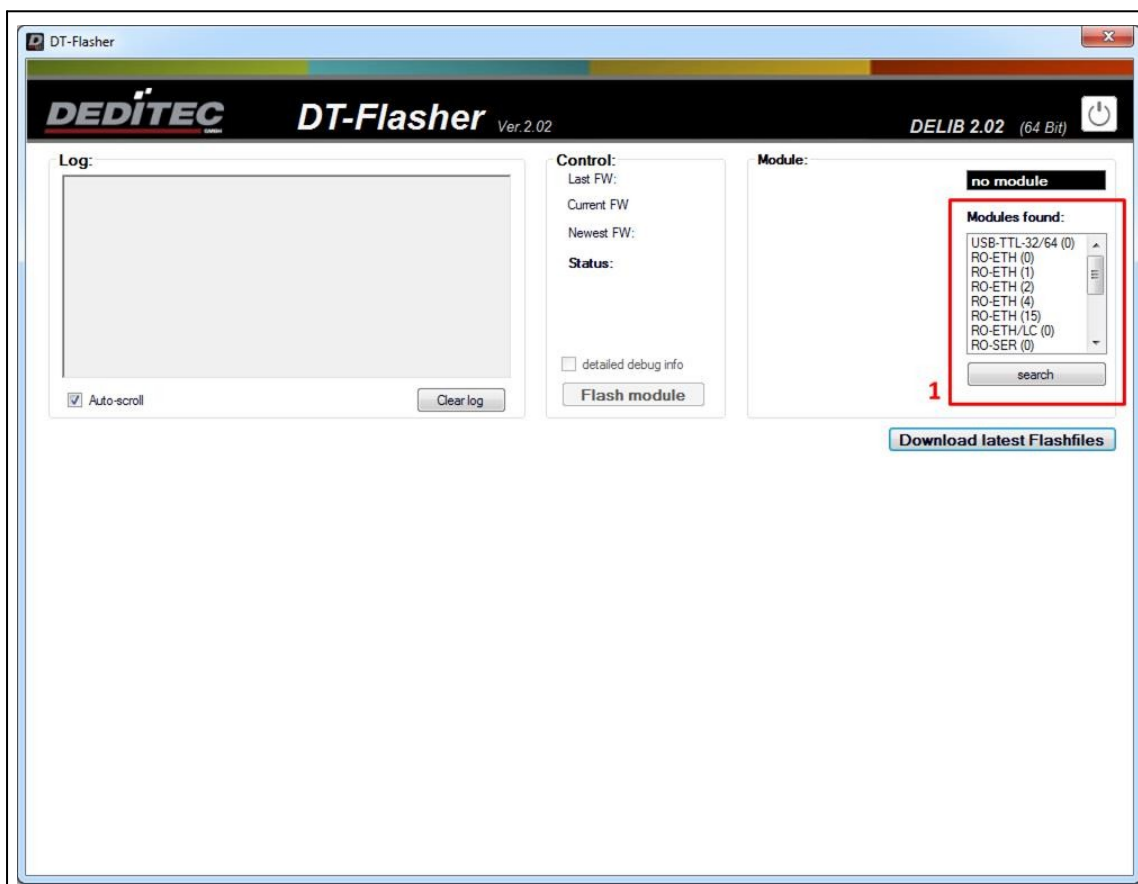
#### 4.2.7.1. Über DEDITEC-Firmware

Die meisten DEDITEC Produkte verfügen über einen eigenen Microcontroller. Dieser Prozessor ist für die Steuerung aller Abläufe der Hardware verantwortlich. Um die für den Prozessor benötigte Firmware im Nachhinein zu ändern, stellen wir unser kostenloses Tool DT-Flasher zur Verfügung. Mit diesem Tool hat der Kunde die Möglichkeit neu veröffentlichte Firmware-Versionen, direkt bei sich vor Ort auf das Modul zu übertragen.

#### Hinweis:

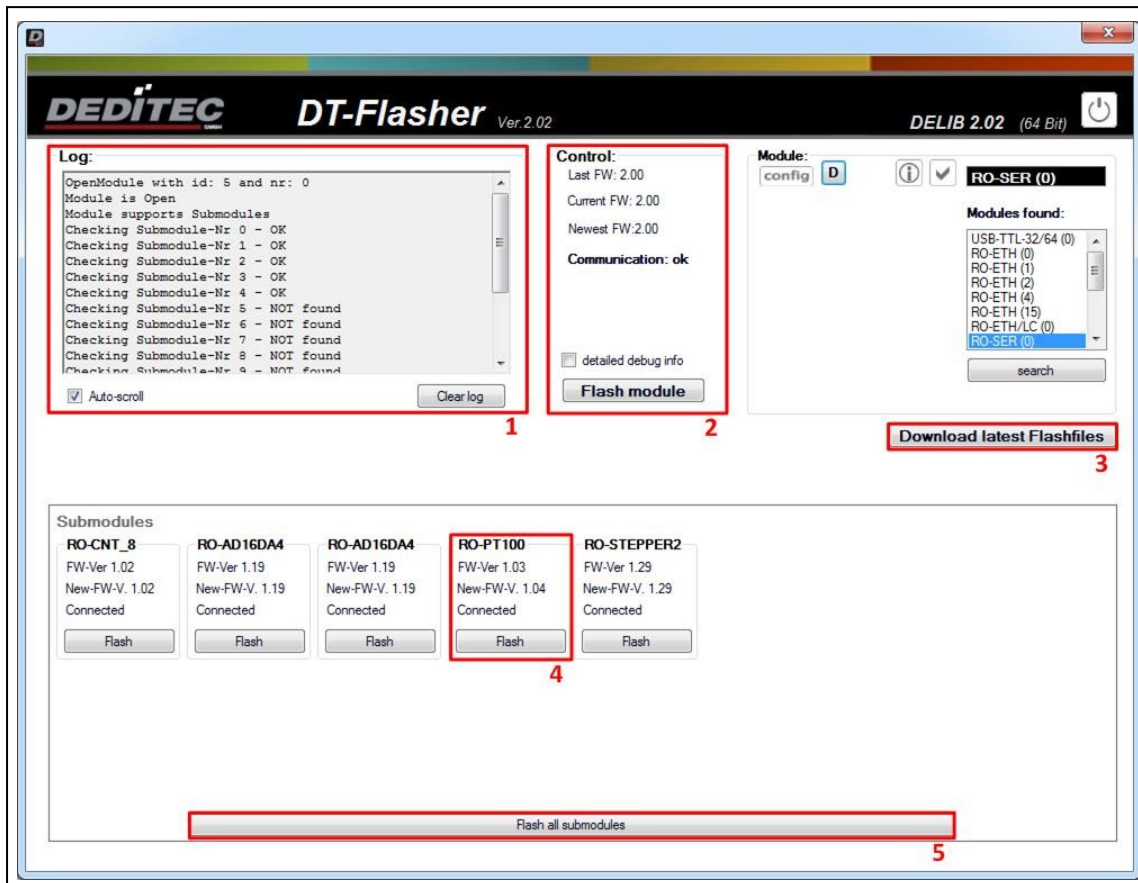
Da neue Firmware-Versionen in der Regel neue Funktionen für Ihr Produkt "freischalten", empfehlen wir daher ein regelmäßiges Firmware-Update Ihrer DEDITEC Produkte.

#### 4.2.7.2. Auswahl des Moduls



1. Wählen Sie bei Programmstart das Modul aus, welches Sie mit einer neuen Firmware updaten möchten. Hierzu finden Sie eine Auflistung aller verfügbaren Module im "Module-Selector"

#### 4.2.7.3. Firmware Update durchführen



Dieses Beispiel zeigt das Modul RO-SER-CNT8-AD32-DA8-PT100-4-STEPPER2 vor einem Firmware-Update.

1. Logbuch - Alle Meldungen während des Firmware-Updates werden hier angezeigt. Über Auto-scroll wird festgelegt, ob immer automatisch bis zum letzten Ereignis heruntergescrollt werden soll. Über Clear log wird das gesamte Logbuch gelöscht.

2. Hier erhalten Sie Informationen zum Interface-Modul (in diesem Beispiel das RO-SER-Interface). Newest FW zeigt die neuste Firmware-Version an, die für das Modul verfügbar ist. Current FW zeigt die Version an, die aktuell auf dem Modul vorhanden ist. Nachdem das Modul erfolgreich geflasht wurde, zeigt Last FW die Version an, die vor dem Firmware-Update aufgespielt war. Ist der Haken bei detailed debug info gesetzt, werden während des Firmware Updates detaillierte Meldungen ins Logbuch(1) geschrieben. Mit Flash module wird das Firmware Update für das Interface-Modul gestartet.

3. Hierüber können Sie direkt aus der Anwendung heraus die aktuellsten Firmware Versionen, sogenannte Flash-Files, herunterladen.

4. Firmware Version zeigt die aktuelle Firmware Version des Submoduls. New-FW-Ver zeigt die neuste Version an, die für dieses Submodul verfügbar ist. Über den Button Flash wird das Firmware Update für das jeweilige Submodul durchgeführt.

5. Über den Button Flash all submodules wird das Firmware Update für alle angeschlossenen Submodule durchgeführt.

#### **4.2.7.3.1. Flash-Files manuell aktualisieren**

In manchen Fällen ist es nötig, die Flash-Files manuell zu aktualisieren, z.B. wenn am PC keine Administratoren-Rechte verfügbar sind.

##### **Schritt 1**

Downloaden Sie die aktuellste Version der Flash-Files unter

[http://www.deditec.de/zip/deditec-flash\\_files.zip](http://www.deditec.de/zip/deditec-flash_files.zip)

##### **Schritt 2**

Entpacken Sie das heruntergeladene ZIP-Archiv, je nach DELIB Installation, in folgendes Verzeichnis:

x86

C:\Program Files(x86)\DEDITEC\DELIB\programs\

x64

C:\Program Files\DEDITEC\DELIB\programs

### 4.3. DELIB Sample Sources (Windows Programmbeispiele)

Die DELIB Sample Sources bieten Beispielprogramme inklusive Quellcode zu nahezu allen DEDITEC Produkten.

Um den Schnelleinstieg mit unseren Modulen zu vereinfachen, finden Sie Quellcodes zu folgenden Programmiersprachen:

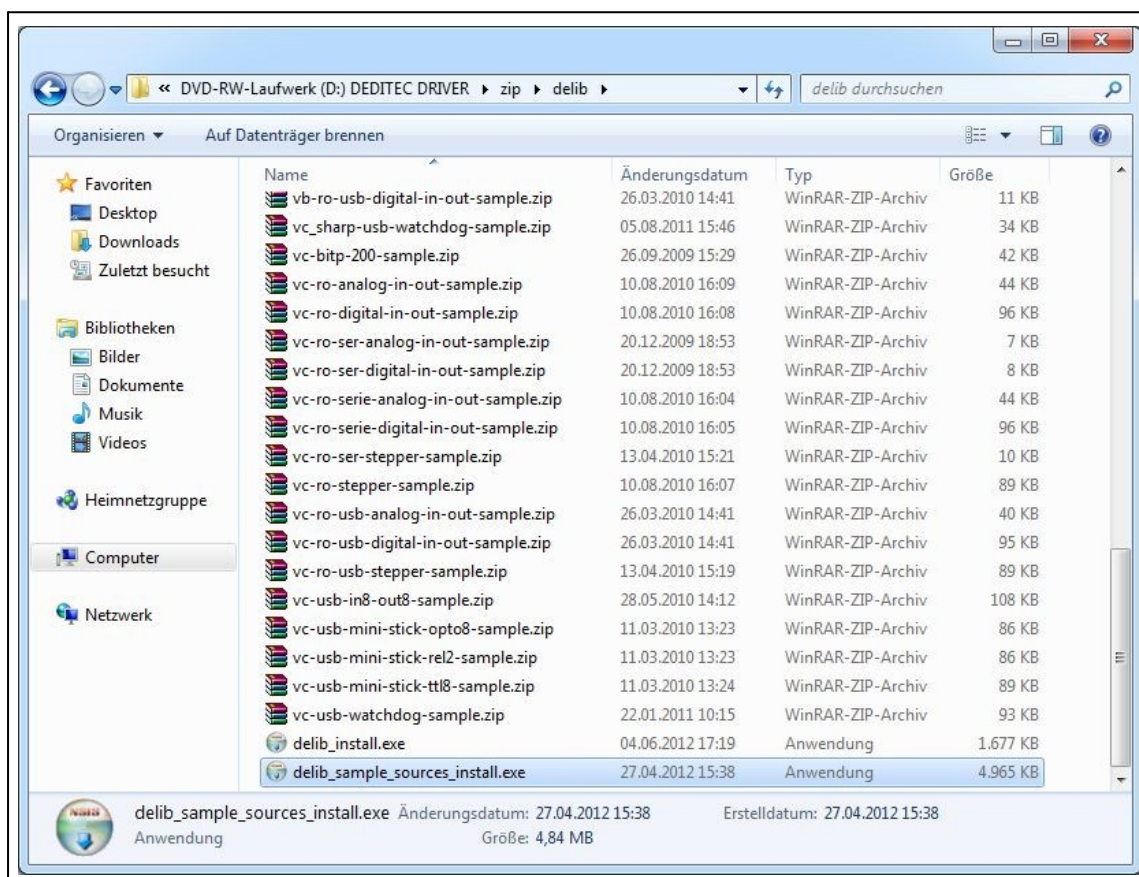
- C
- C++
- C#
- Delphi
- VisualBasic
- VB.NET
- MS-Office
- LabVIEW
- Java

### 4.3.1. Installation DELIB Sample Sources

Die DELIB Sample Sources können entweder während der Durchführung des DELIB Setups installiert werden oder als eigenständiges Setup.

Legen Sie die DEDITEC Driver-CD in das Laufwerk und starten Sie `delib_sample_sources_install.exe`.

Eine aktuelle Version der Sample Sources finden Sie auch im Internet unter <http://www.deditec.de/de/delib>

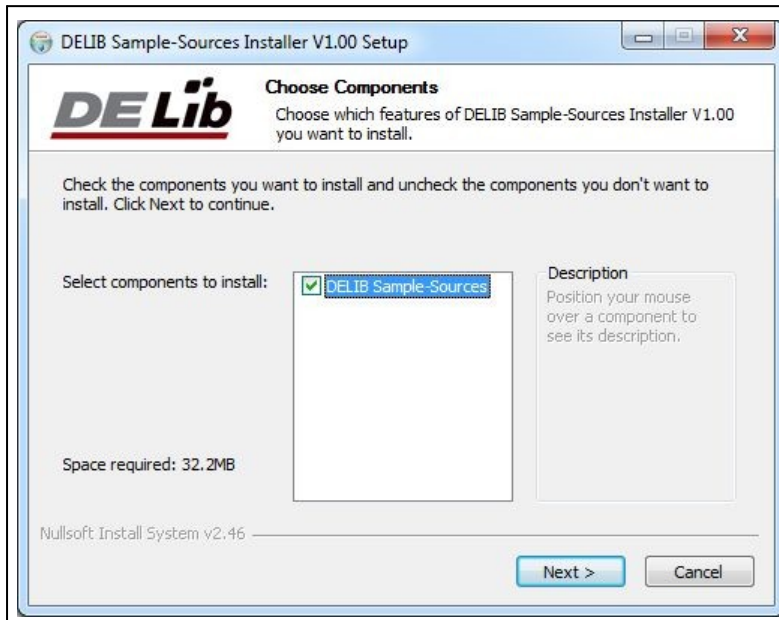




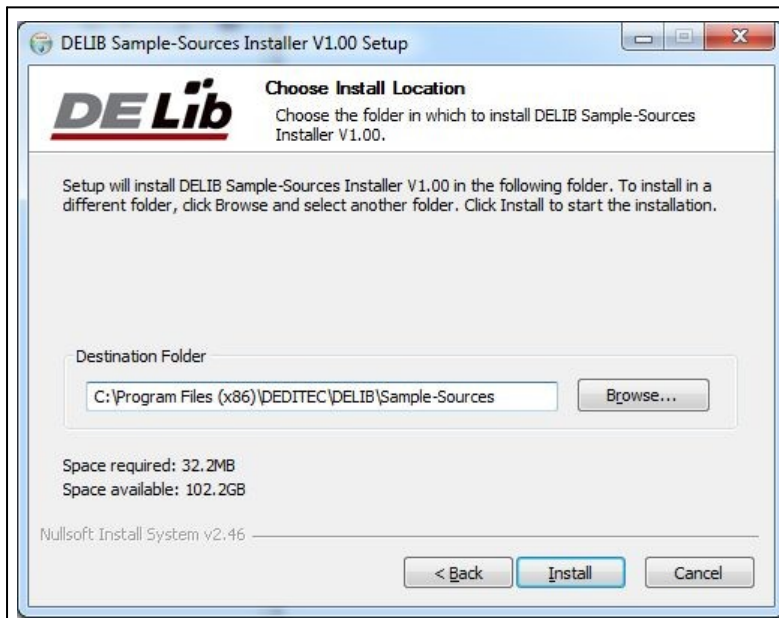
Startbild des DELIB Sample Sources Installer



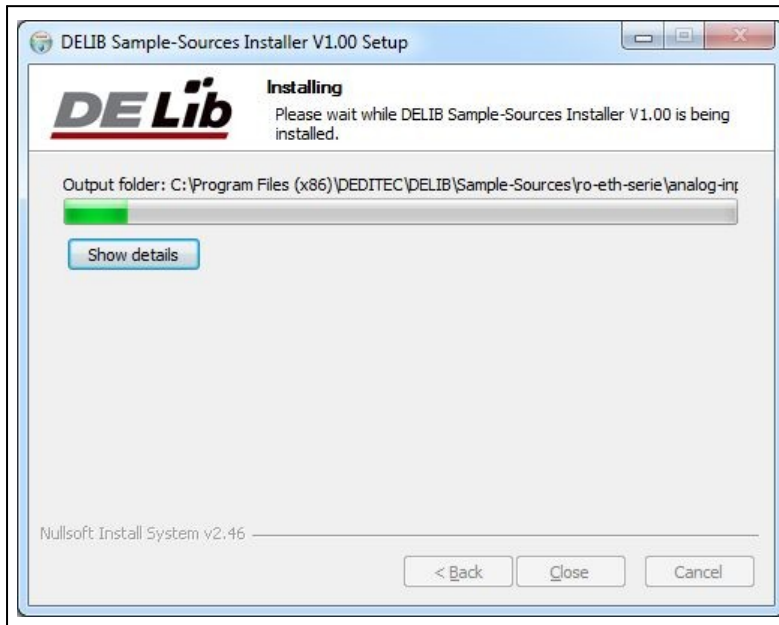
Drücken Sie Next.



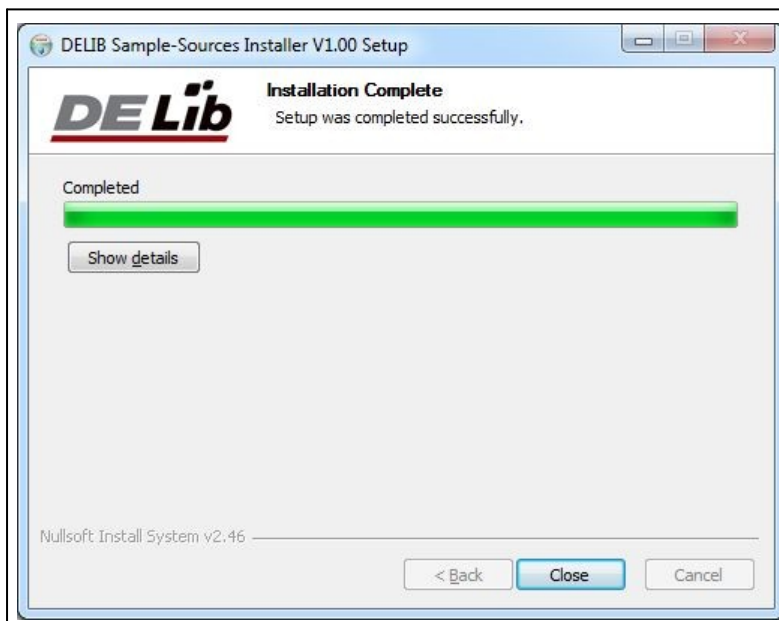
Wählen Sie den Installationsordner und drücken Sie Install.



Die DELIB Sample Sources werden nun installiert.



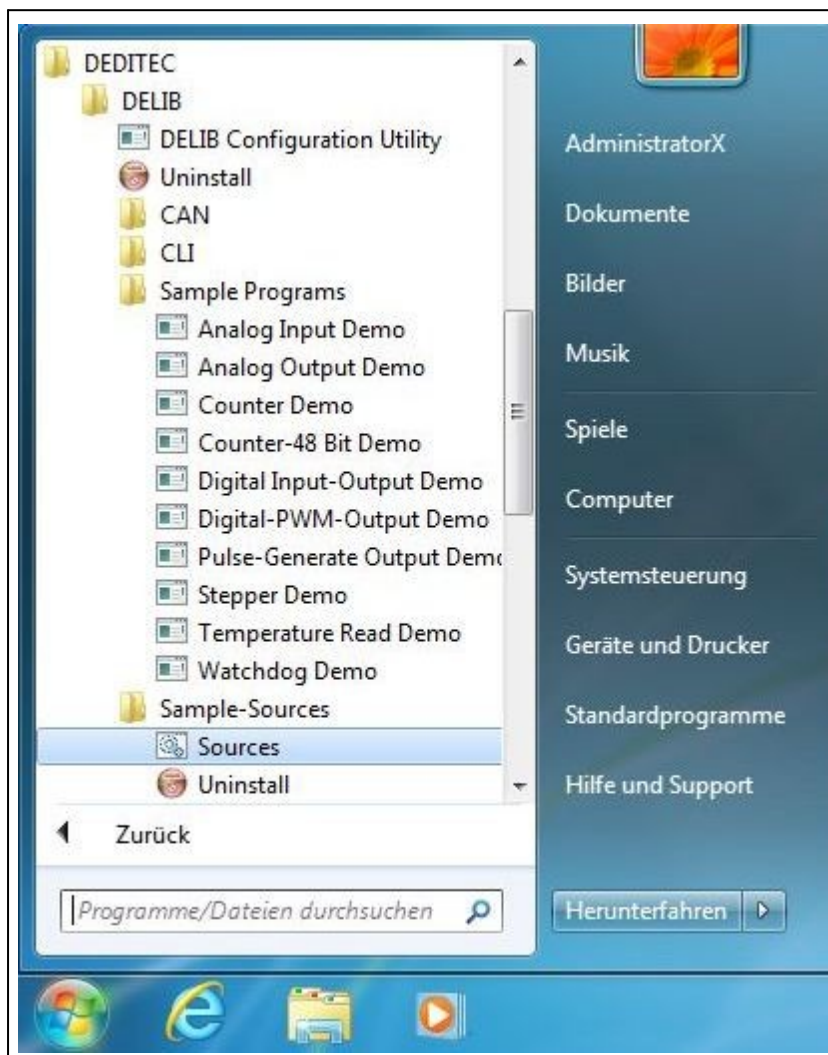
Die DELIB Sample Sources wurden erfolgreich installiert. Drücken Sie Close um die Installation zu beenden.



### 4.3.2. Benutzung der DELIB Sample Sources

Nach Installation der DELIB Sample Sources finden Sie diese unter

Start → Programme → DEDITEC → DELIB → Sample-Sources → Sources

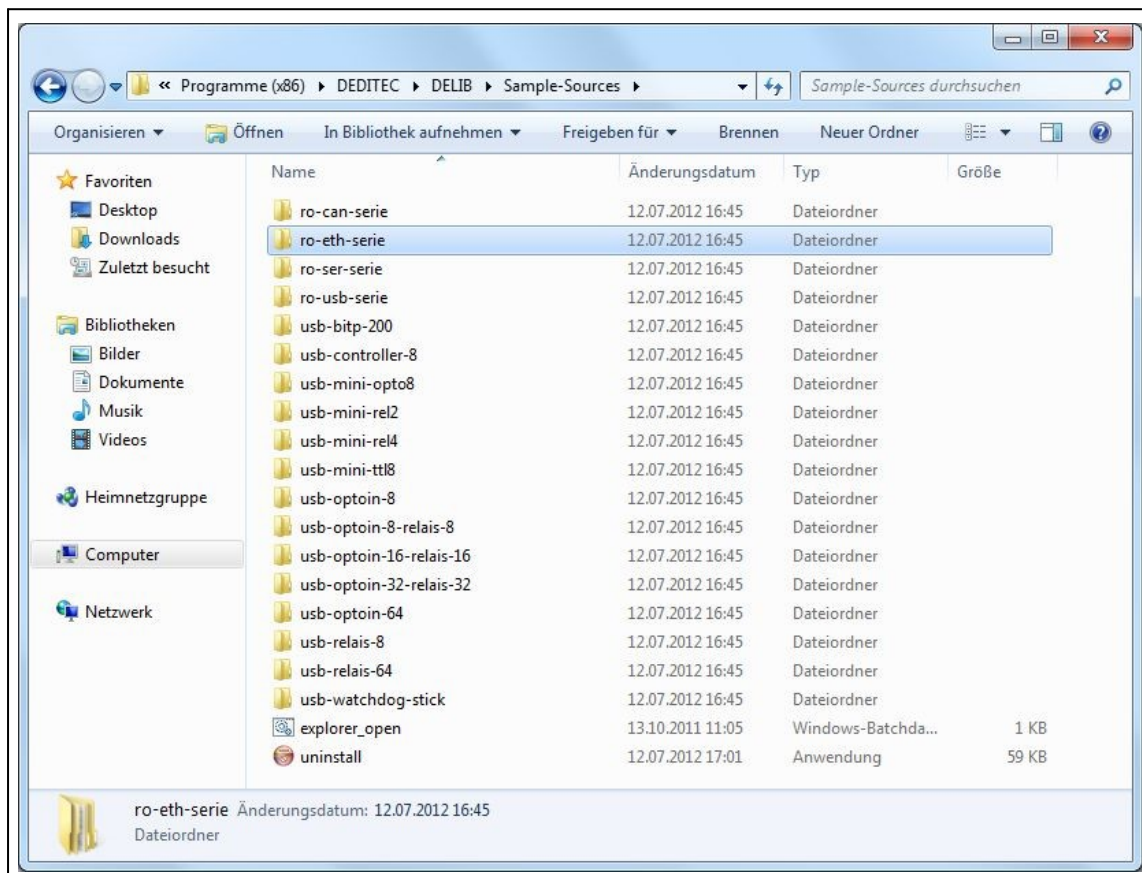


Nun öffnet sich der Windows-Explorer mit einer Übersicht aller Produkte für die ein Beispielprogramm verfügbar ist.

#### 4.3.2.1. Schritt 1 - Produktauswahl

Sie benötigen beispielsweise eine Hilfestellung zur Programmierung der digitalen Eingänge eines RO-ETH-Moduls (z.B. RO-ETH-016) in der Programmiersprache Visual-C.

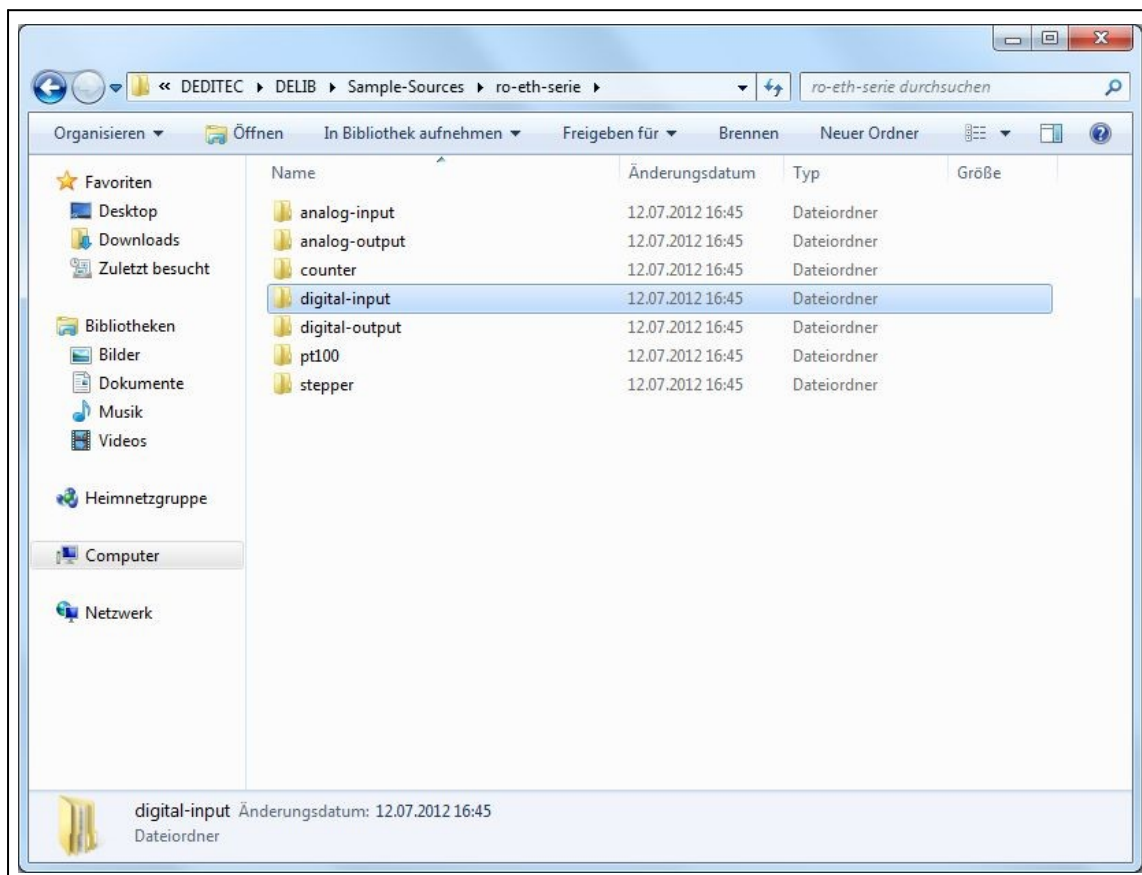
Da es sich um ein RO-ETH-Produkt handelt, wählen bzw. öffnen Sie den Ordner ro-eth-serie.



#### 4.3.2.2. Schritt 2 - Kategorieauswahl

Im nächsten Schritt, finden Sie eine Übersicht der verfügbaren Kategorien für das ausgewählte Produkt.

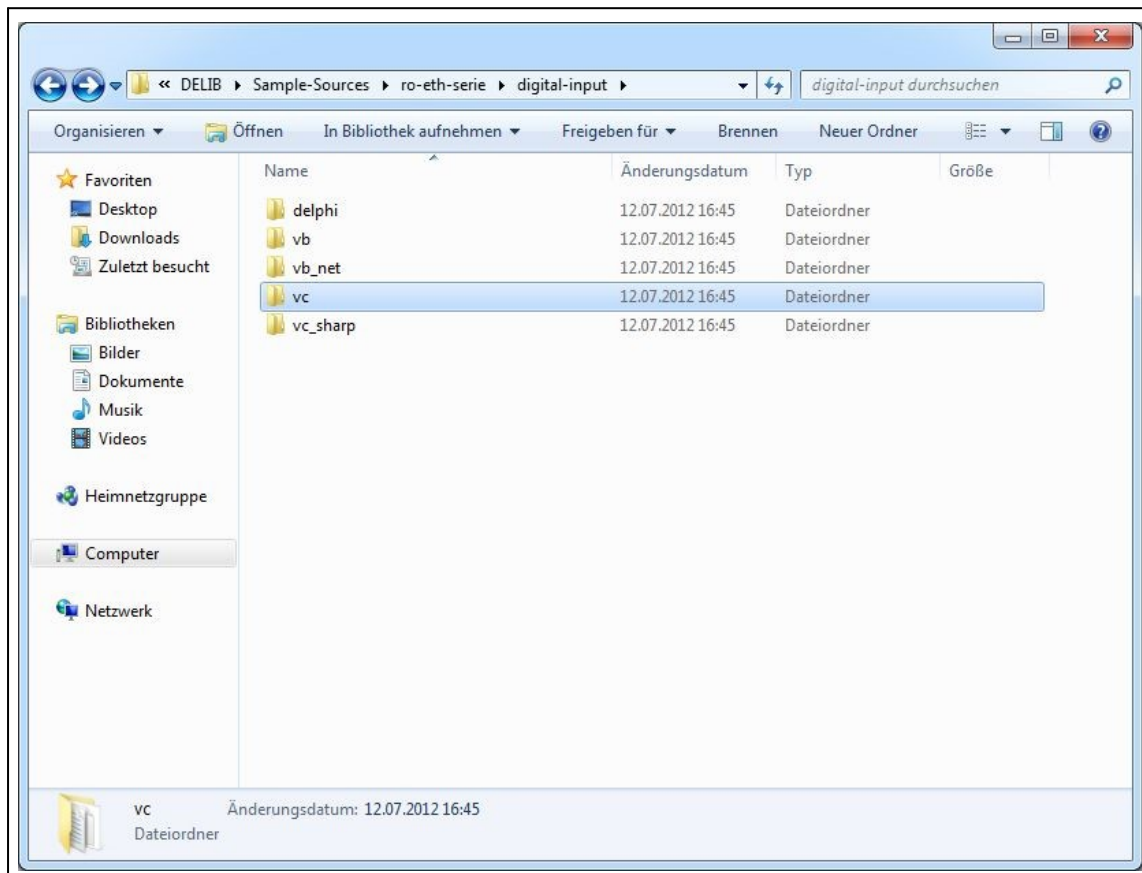
Da wir uns in diesem Beispiel auf die digitalen Eingänge konzentrieren, wählen Sie die Kategorie digital-input



#### 4.3.2.3. Schritt 3 - Programmiersprachenauswahl

In diesem Schritt sehen Sie alle verfügbaren Programmierbeispiele der gewählten Kategorie, sortiert nach Programmiersprachen.

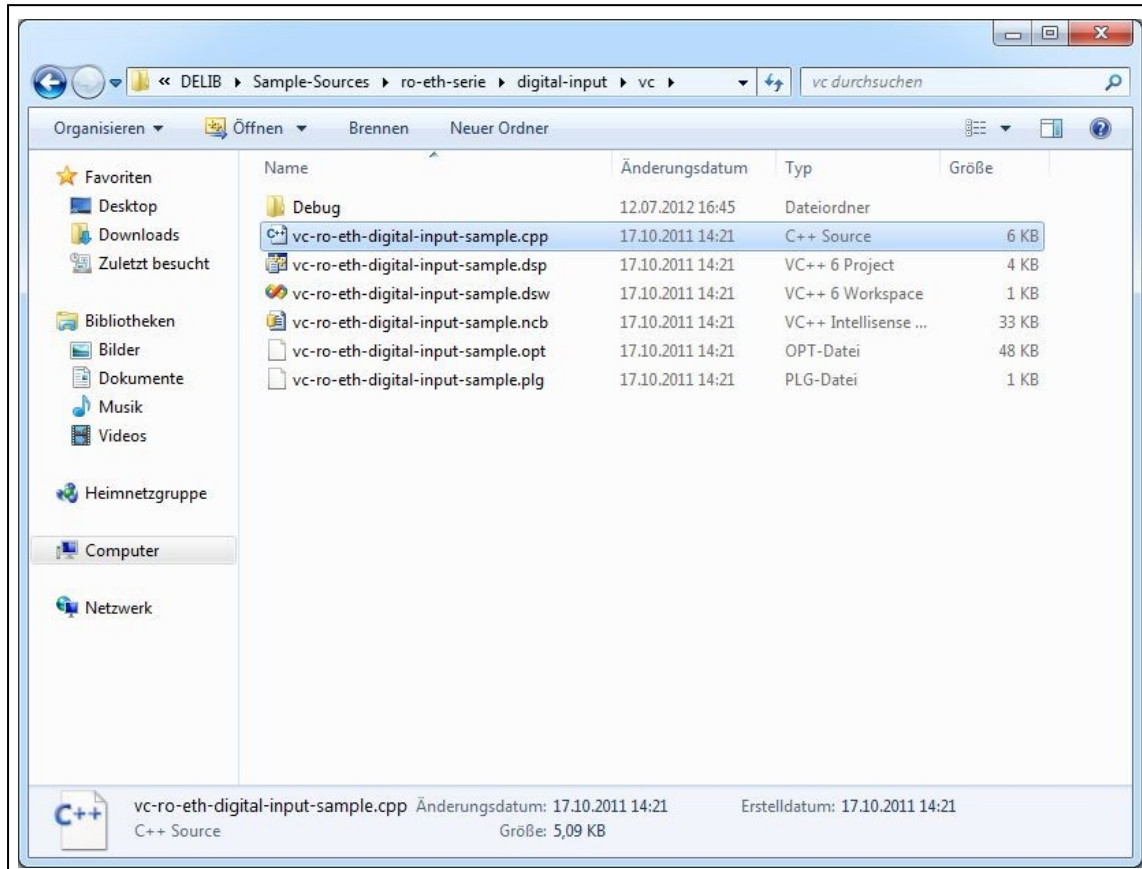
Da wir uns in diesem Beispiel auf die Programmiersprache Visual-C konzentrieren, öffnen Sie den Ordner vc.





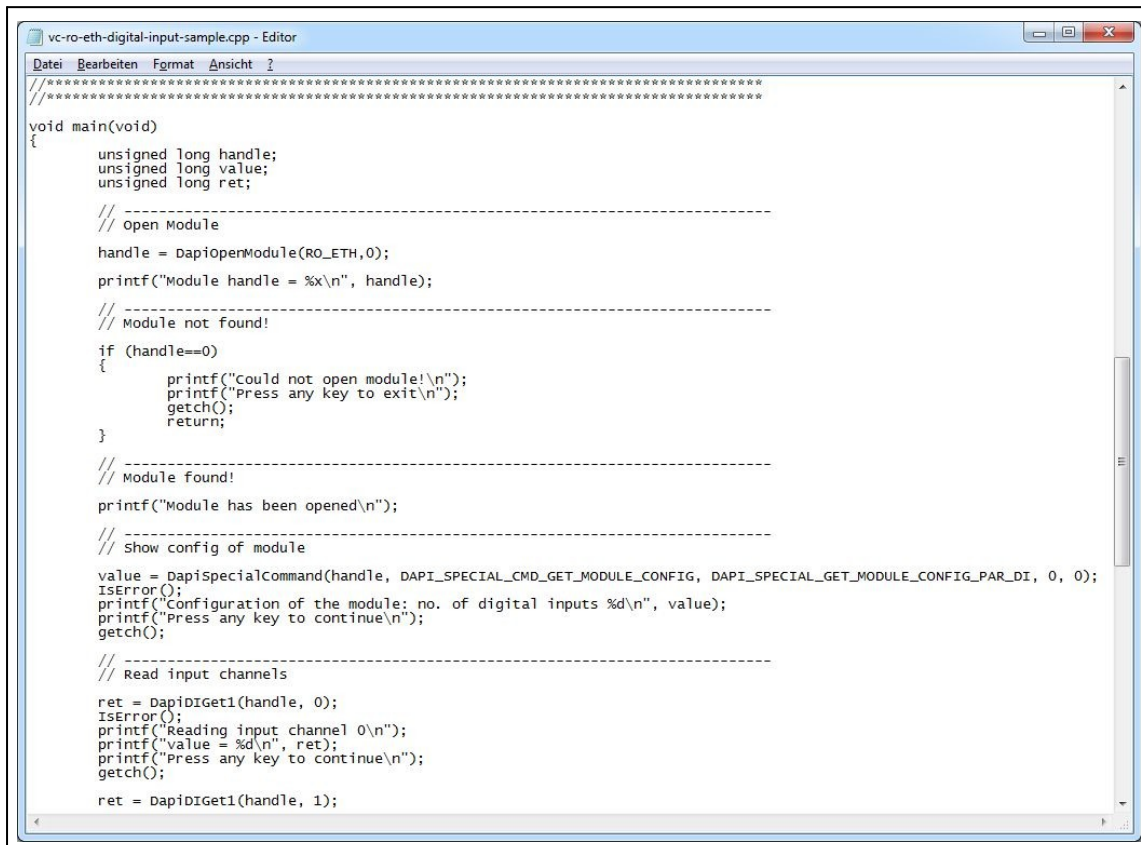
#### 4.3.2.4. Schritt 4 - Quellcode

Nach Auswahl der Programmiersprache erhalten Sie folgende Übersicht:





Den Quellcode des Beispielprogramms (in diesem Fall .cpp-Datei) können Sie nun mit einem beliebigen Text-Editor öffnen.



```
vc-ro-eth-digital-input-sample.cpp - Editor
Datei Bearbeiten Format Ansicht ?
//*****
//*****
void main(void)
{
    unsigned long handle;
    unsigned long value;
    unsigned long ret;

    // -----
    // Open Module

    handle = DapiOpenModule(RO_ETH,0);
    printf("Module handle = %x\n", handle);

    // -----
    // Module not found!

    if (handle==0)
    {
        printf("Could not open module!\n");
        printf("Press any key to exit\n");
        getch();
        return;
    }

    // -----
    // Module found!

    printf("Module has been opened\n");

    // -----
    // Show config of module

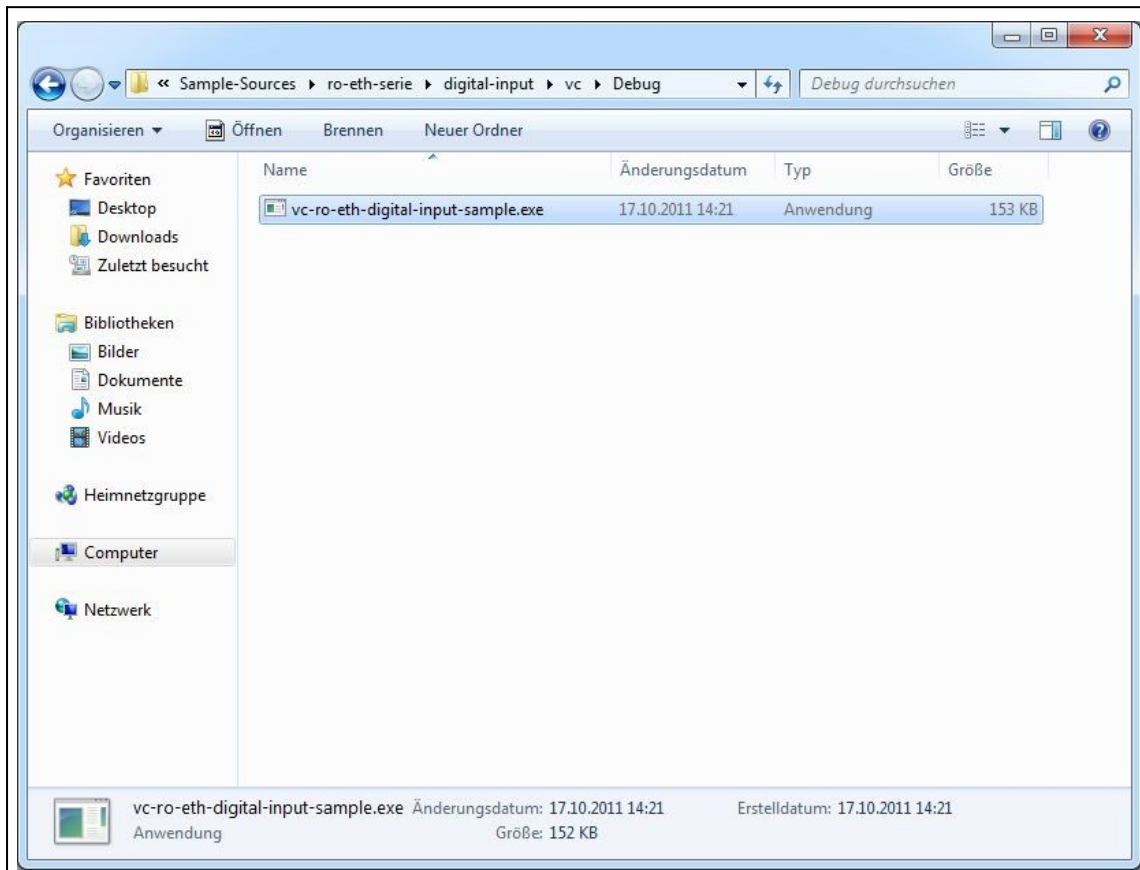
    value = DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_GET_MODULE_CONFIG, DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_DI, 0, 0);
    if (value < 0)
    {
        printf("Configuration of the module: no. of digital inputs %d\n", value);
        printf("Press any key to continue\n");
        getch();
    }

    // -----
    // Read input channels

    ret = DapiGetI(handle, 0);
    if (ret < 0)
    {
        printf("Reading input channel 0\n");
        printf("value = %d\n", ret);
        printf("Press any key to continue\n");
        getch();
    }

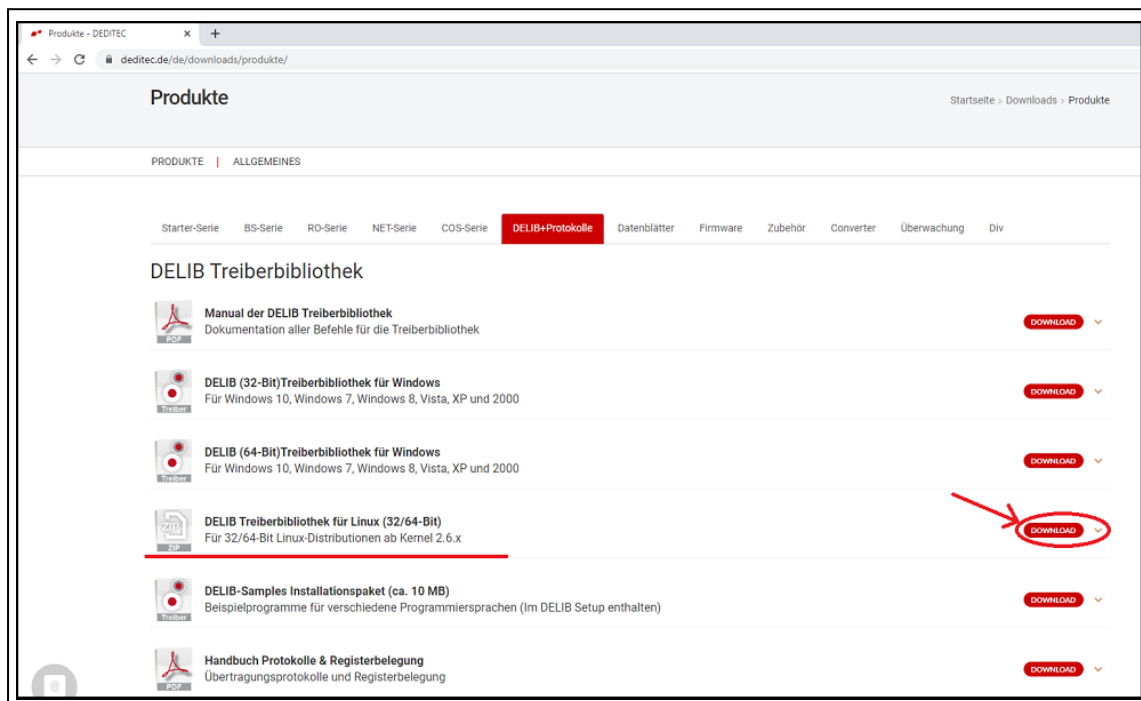
    ret = DapiGetI(handle, 1);
```

Zusätzlich finden Sie im Ordner debug ein bereits kompiliertes und ausführbares Programm zu diesem Projekt.

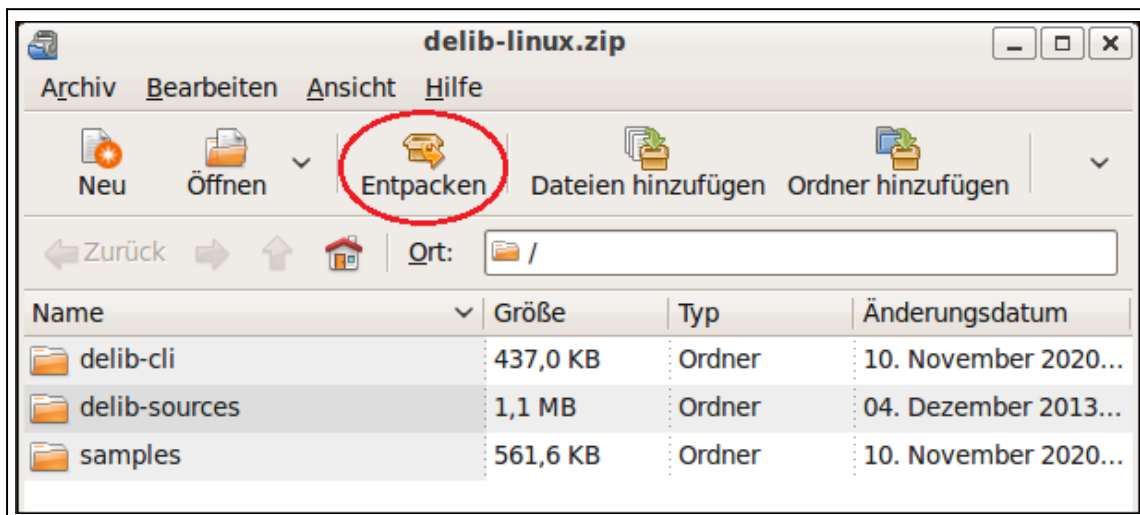


## 4.4. DELIB für Linux

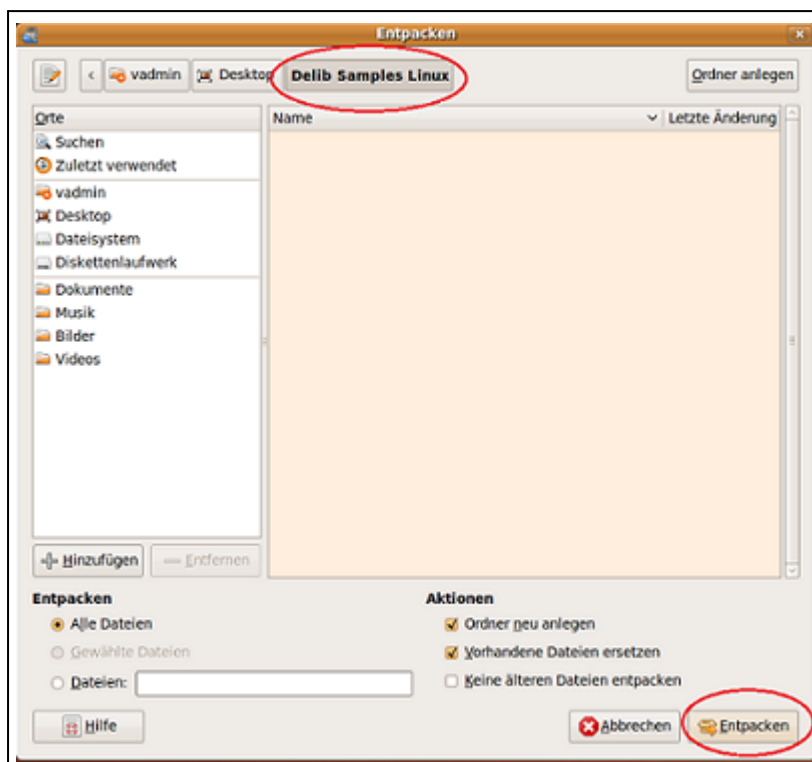
Laden Sie sich die Delib Linux Treiberbibliothek unter "[www.deditec.de/de/downloads/produkte/](http://www.deditec.de/de/downloads/produkte/)" im Reiter „DELIB+Protokolle“ oder unter "[www.deditec.de/media/zip/delib/delib-linux.zip](http://www.deditec.de/media/zip/delib/delib-linux.zip)" direkt auf ihr Linux-System.



Entpacken Sie die "delib-linux.zip" in einen beliebigen Zielordner. Doppelklicken Sie dafür auf die Zip-Datei und benutzen Sie dann den "Entpacken"-Knopf in der oberen Menüleiste.



Wählen Sie Ihren Zielordner aus und klicken Sie dann auf den "Entpacken"-Knopf.



## 4.4.1. Verwenden der DELIB Treiberbibliothek für Linux

### 4.4.1.1. Delib USB-Sample in Linux

#### Voreinstellungen

In diesem Programmbeispiel wird ein USB\_REL AIS\_8 Modul angesprochen. Sollten Sie ein anderes Modul verwenden, müssen Sie in der Datei

„./samples/usb\_sample/source/usb\_sample.c“ bei dem Befehl „DapiOpenModule“ ihr Modul angeben. Die genaue Bezeichnung können Sie der „delib.h“ entnehmen. Diese finden sie im Verzeichnis „./delib-sources/delib/library/delib/delib.h“

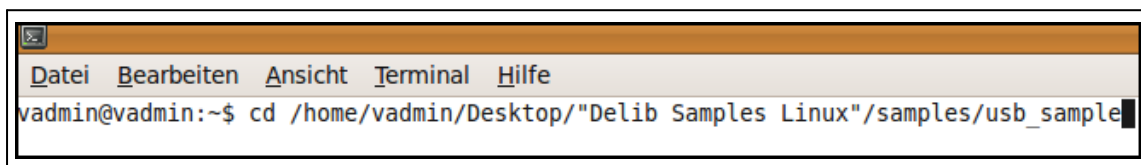
```
23
24 #include <stdio.h>
25 #include <stdlib.h>
26 #include <unistd.h>
27
28 #include "../../delib-sources/delib/library/delib/delib.h"
29
30 int main()
31 {
32     ULONG i;
33     ULONG handle=0;
34
35     printf("\n\n");
36     printf("-----\n");
37     printf("-----\n");
38     printf("-----\n");
39     printf("WICHTIG !!!\n");
40     printf("Dieses Programm bitte mit admin-Rechten ausfuehren\n");
41     printf("Also: sudo ./delib-test-digital-io <return>\n");
42     printf("-----\n");
43     printf("-----\n");
44     printf("-----\n");
45     printf("\n\n");
46
47     printf("-----\n");
48     printf("Try to open USB_REL AIS_8\n");
49     handle = DapiOpenModule(USB_REL AIS_8, 0);
50
51     if(handle == 0)
52     {
53         // Module not found
54         printf("Handle = 0x%x\n", (unsigned long) handle);
55         return 0;
56     }
57
58     printf("Handle = 0x%x\n", (unsigned long) handle);
59 }
```

## Kompilieren des USB-Samples

Für das Kompilieren des Testprogramms öffnen Sie ein Terminalfenster und navigieren mit dem Befehl

"cd /<Verzeichnispfad>" zunächst in das "/samples/usb\_sample" Verzeichnis.

Tipp: Sollten in Ihrem Ordernamen Leerzeichen enthalten sein, geben Sie diese wie im unteren Beispiel dargestellt in " " an.



```

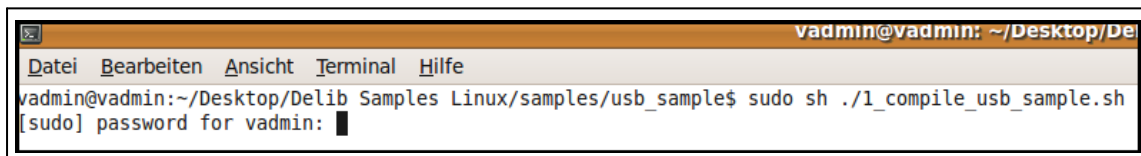
Datei Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~$ cd /home/vadmin/Desktop/"Delib Samples Linux"/samples/usb_sample

```

Zum Kompilieren öffnen Sie nun das darin enthaltene Shell-Skript mit dem Befehl

„sudo sh ./1\_compile\_usb\_sample.sh“.

Geben Sie, falls nötig, Ihr Benutzerkennwort ein.



```

vadmin@vadmin: ~/Desktop/Del
Datei Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/usb_sample$ sudo sh ./1_compile_usb_sample.sh
[sudo] password for vadmin:

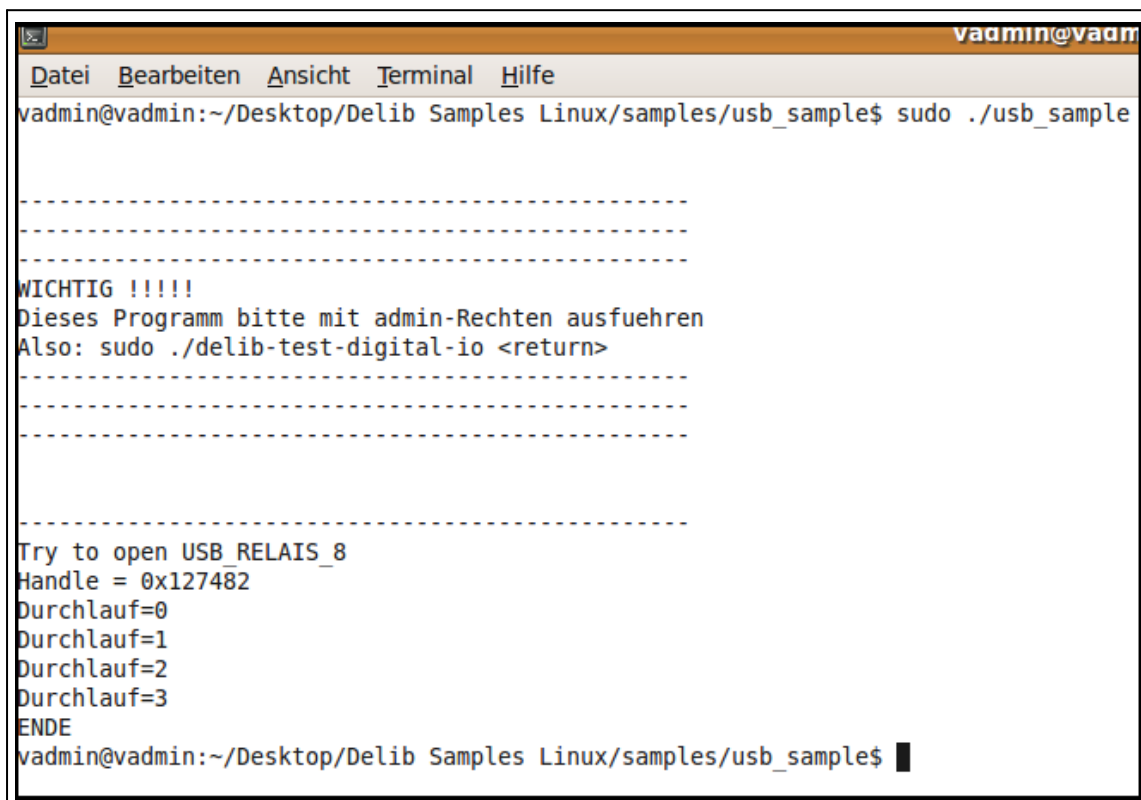
```

Bei erfolgreicher Kompilierung sollte nun "compiling successful" im Terminalfenster erscheinen.

Es wurde die Datei "usb\_sample" dem Verzeichnis hinzugefügt.

Jetzt können Sie das Beispielprogramm mit "sudo ./usb\_sample" ausführen.

**WICHTIG!!** Sie benötigen für das Ausführen Admin-Rechte. Benutzen Sie deshalb den Befehl mit "sudo"



```
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/usb_sample$ sudo ./usb_sample

-----
WICHTIG !!!!!
Dieses Programm bitte mit admin-Rechten ausfuehren
Also: sudo ./delib-test-digital-io <return>
-----

-----

Try to open USB_RELAIS_8
Handle = 0x127482
Durchlauf=0
Durchlauf=1
Durchlauf=2
Durchlauf=3
ENDE
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/usb_sample$
```

Das Programm wird nun ausgeführt.

In diesem Beispiel werden alle digitalen Ausgänge des USB\_RELAIS\_8 in einer Schleife an und wieder ausgeschaltet.



#### 4.4.1.2. Delib ETH-Sample in Linux

##### Voreinstellungen

Bei diesem Programmbeispiel wird das Modul mit der IP "192.168.1.21" angesprochen. Diese können Sie in der Datei

„./samples/ethernet\_sample/source/eth\_sample.c“ ändern (siehe Bild unten).

Falls Sie ein Kennwort für eine verschlüsselte TCP Verbindung voreingestellt haben, können Sie dieses ebenfalls dort eintragen (siehe Bild unten). Haben Sie kein Passwort angegeben, können Sie diese Zeile unverändert lassen.

Die Konfiguration der ETH-Module können über das DELIB Configuration Utility, sowie über die Weboberfläche des Moduls eingestellt werden.

```
26 #include <string.h>
27 #include <unistd.h>
28
29 #include "../delib-sources/delib/library/delib/delib.h"
30
31 int main()
32 {
33     unsigned long i;
34     unsigned long handle;
35     unsigned long ret;
36     DAPI_OPENMODULEEX_STRUCT open_buffer;
37
38     strcpy((char*) open_buffer.address, "192.168.1.21"); // hostname
39     open_buffer.timeout = 5000; // 5000 msec
40     open_buffer.portno = 9912; // using default port
41
42     #ifdef ENABLE_TCP_ENCRYPTION
43         open_buffer.encryption_type = DAPI_OPEN_MODULE_ENCRYPTION_TYPE_ADMIN; // encrypted communication with admin priv
44         strcpy((char*) open_buffer.encryption_password, "myPassword"); // password for encrypted communication
45     #else
46         open_buffer.encryption_type = DAPI_OPEN_MODULE_ENCRYPTION_TYPE_NONE; // Falls vorher eingestellt, geben Sie hier das Passwort
47     #endif // Ihrer verschlüsselten TCP-Verbindung an.
48
49     handle = DapiOpenModuleEx(ETHERNET_MODULE, 0, (unsigned char*) &open_buffer, DAPI_OPEN_MODULE_OPTION_USE_EXBUFFER);
50
51     if(handle == 0)
52     {
```

Sollten Sie ein Modul ohne digitale Eingänge verwenden, müssen Sie die Zeilen wie unten dargestellt, in der gleichen Datei auskommentieren.

```
57     for(i=0; i!=4; ++i)
58     {
59         printf("Durchlauf = %ld\n", i);
60
61         DapiDOSet8(handle, 0, 0xff);
62
63         usleep(1000 * 500);          // 500 msec sleep
64
65         DapiDOSet8(handle, 0, 0);
66
67         usleep(1000 * 500);          // 500 msec sleep
68         //ret = DapiDIGet8(handle, 0);
69         //printf("DI0-7 = 0x%lx\n", ret);
70
71         usleep(1000 * 500);          // 500 msec sleep
72     }
73
74
```

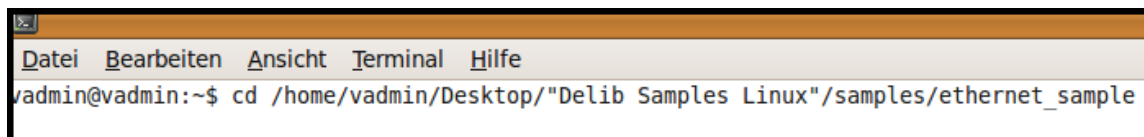
Auskommentieren,  
falls keine digitalen  
Eingänge vorhanden

### Kompilieren des ETH-Samples

Für das Kompilieren des Testprogramms, öffnen Sie ein Terminalfenster und navigieren mit dem Befehl

"cd /<Verzeichnispfad>" zunächst in das "/samples/ethernet\_sample" Verzeichnis.

Tipp: Sollten in Ihrem Ordnernamen Leerzeichen enthalten sein, geben Sie diese wie im unteren Beispiel dargestellt in " " an.

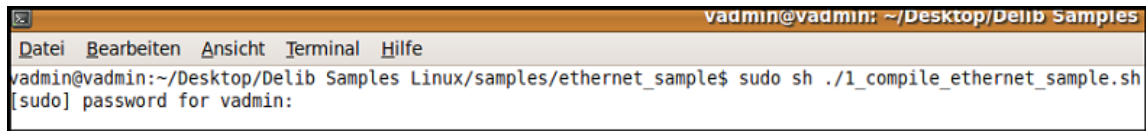
A screenshot of a terminal window with a menu bar containing 'Datei', 'Bearbeiten', 'Ansicht', 'Terminal', and 'Hilfe'. The terminal text shows a user 'vadmin' at host 'vadmin' in the directory '~' executing the command 'cd /home/vadmin/Desktop/"Delib Samples Linux"/samples/ethernet\_sample'.

```
vadmin@vadmin:~$ cd /home/vadmin/Desktop/"Delib Samples Linux"/samples/ethernet_sample
```

Zum Kompilieren öffnen Sie nun das gewünschte Shell-Skript mit dem Befehl „sudo sh ./<DATEINAME>“

- Möchten Sie das Modul über eine unverschlüsselte TCP Verbindung ansteuern, verwenden Sie die Datei „1\_compile\_ethernet\_sample.sh“
- Möchten Sie das Modul über eine verschlüsselte TCP Verbindung ansteuern, verwenden Sie die Datei „2\_compile\_ethernet\_sample\_with\_encryption.sh“

Geben Sie, falls nötig, Ihr Benutzerkennwort ein.



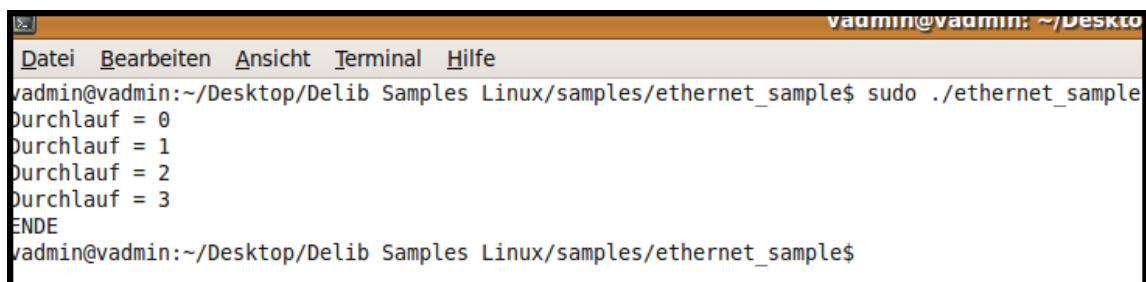
```
vadmin@vadmin: ~/Desktop/Delib Samples
Datei Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/ethernet_sample$ sudo sh ./1_compile_ethernet_sample.sh
[sudo] password for vadmin:
```

Bei erfolgreicher Kompilierung sollte nun "compiling successful" im Terminalfenster erscheinen.

Es wurde die Datei "ethernet\_sample" dem Verzeichnis hinzugefügt.

Jetzt können Sie das Beispielprogramm mit "sudo ./ethernet\_sample" ausführen.

**WICHTIG!!** Sie benötigen für das Ausführen Admin-Rechte. Benutzen Sie deshalb den Befehl mit "sudo".



```
vadmin@vadmin: ~/Desktop
Datei Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/ethernet_sample$ sudo ./ethernet_sample
Durchlauf = 0
Durchlauf = 1
Durchlauf = 2
Durchlauf = 3
ENDE
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/ethernet_sample$
```

Das Programm wird nun ausgeführt.

In diesem Beispiel werden alle Ausgänge des Moduls in einer Schleife an und wieder ausgeschaltet.

#### 4.4.2. DELIB CLI (command-line interface) für Linux

Der DELIB CLI Befehl für Linux befindet sich nach Entpacken des Zip-Archivs "delib-linux-cli" im Ordner /deditec-cli/ .

Definition für USB-Module (Linux)

```
sudo delib_cli [command] [channel] [value | unit ["nunit"]]
```

Definition für ETH-Module (Linux)

```
delib_cli [command] [channel] [value | unit ["nunit"]]
```

**Hinweis:**

Die einzelnen Parameter werden nur durch ein Leerzeichen getrennt.

Groß und Kleinschreibung wird hierbei nicht beachtet.

## Parameter

Befehl	Kabal	Wert		unit	nounit
di1	0, 1, 2, ...	-		hex	nounit
di8	0, 8, 16, ...				
di16					
di32					
ff	0, 32, ...	-		hex	nounit
do1	0, 1, 2, ...	0/1 (1-Bit Befehl)		-	-
do8	0, 8, 16, ...	8-Bit Wert	(Bit 0 für Kanal 1, Bit 1 für Kanal 2, ...)		
do16		16-Bit Wert			
do32		32-Bit Wert			
ai	0, 1, 2, ...	-		hex, volt, mA	nounit
ao	0, 1, 2, ...	Ganz oder Hexadezimalzahl (beginnend mit 0x).		-	-

### **Return-Wert**

#### **Zustand der gelesenen digitalen Eingänge**

In Kombination mit Parameter unit "hex" wird der Zustand als hex gelesen

#### **Zustand der FlipFlips der digitalen Eingänge**

In Kombination mit Parameter unit "hex" wird der Zustand als hex gelesen

#### **Zustand der gelesenen analogen Eingänge**

In Kombination mit Parameter unit "hex" wird der Zustand als hex gelesen

In Kombination mit Parameter unit "volt" wird die Spannung gelesen

In Kombination mit Parameter unit "mA" wird der Strom gelesen

#### 4.4.2.1. Konfiguration des DELIB CLI

##### Voreinstellungen

Vor der ersten Verwendung des DELIB CLI muss die "delib\_cli.cfg" mit einem Texteditor bearbeitet werden.

Sie finden die "delib\_cli.cfg" im Verzeichnis "/delib\_cli/".

##### Inhalt der "delib\_cli.cfg":

```
moduleID=14;  
moduleNR=0;  
RO-ETH_ipAddress=192.168.1.11;
```

##### moduleID

Als moduleID muss die entsprechende Nummer der eingesetzten Hardware eingetragen werden.

Diese Nummer kann der "delib.h" entnommen werden.

Unter Linux finden Sie diese im Zip-Archiv des "delib-linux" unter dem Pfad "delib-sources\delib\library\delib".

##### moduleNR

Die moduleNR wird im DELIB Configuration Utility vergeben.

Diese Nummer dient zur Identifizierung identischer Hardware.

Der Standardwert ist 0.

##### RO-ETH\_ipAddress

Dieser Eintrag wird ausschließlich für die Verbindung zu unseren ETH-Modulen benötigt.

Die IP-Adresse der ETH-Module können über das DELIB Configuration Utility sowie über die Weboberfläche des Moduls eingestellt werden.

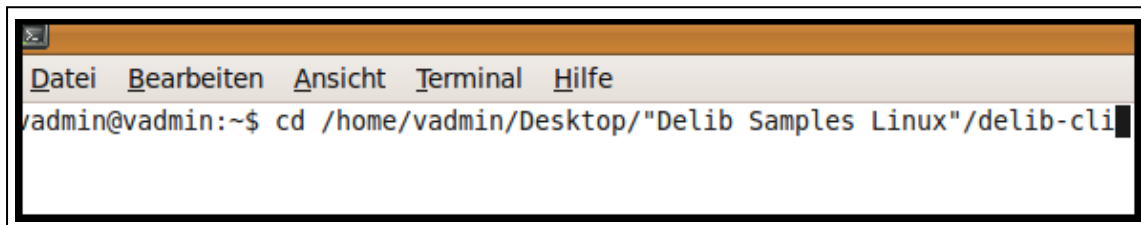


### Kompilieren des Delib-CLI-Samples

Für das Kompilieren des Testprogramms, öffnen Sie ein Terminalfenster und navigieren mit dem Befehl

"cd /<Verzeichnispfad>" zunächst in das "../delib\_cli/" Verzeichnis.

Tip: Sollten in Ihrem Ordernamen Leerzeichen enthalten sein, geben Sie diese wie im unteren Beispiel dargestellt in " " an.

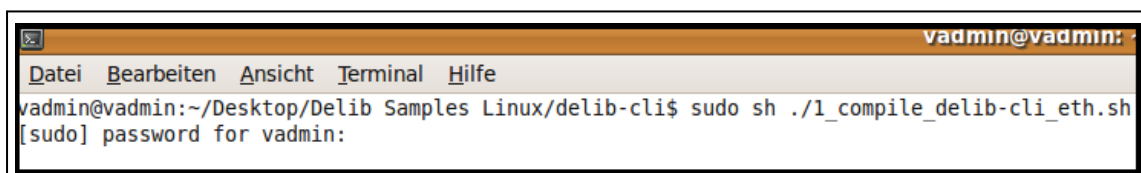


```
File Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~$ cd /home/vadmin/Desktop/"Delib Samples Linux"/delib-cli
```

Zum Kompilieren öffnen Sie nun das gewünschte Shell-Skript mit dem Befehl „sudo sh ./<DATEINAME>“

- ETH - "1\_compile\_delib-cli\_eth.sh"
- USB - "2\_compile\_delib-cli\_usb.sh"

Geben Sie, falls nötig, Ihr Benutzerkennwort ein.

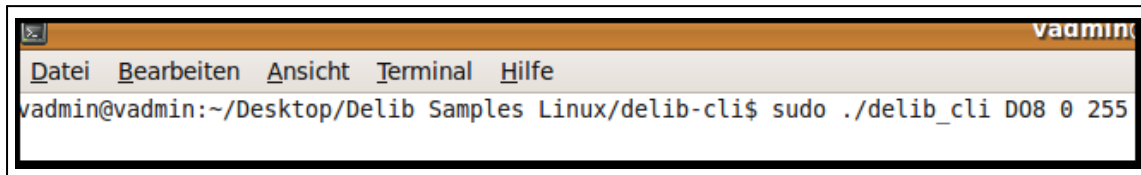


```
vadmin@vadmin:
File Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~/Desktop/Delib Samples Linux/delib-cli$ sudo sh ./1_compile_delib-cli_eth.sh
[sudo] password for vadmin:
```

Bei erfolgreicher Kompilierung sollte nun "compiling successfull" im Terminalfenster erscheinen. Es wurde die Datei "delib\_cli" im Verzeichnis erstellt. Jetzt können Sie das Beispielprogramm mit

"sudo ./delib\_cli [command] [channel] [value | unit ["nounit"]] " ausführen.

**WICHTIG!!** Sie benötigen für das Ausführen Admin-Rechte. Benutzen Sie deshalb den Befehl mit "sudo".

A terminal window with a brown title bar containing a window icon and the text 'vadmin'. Below the title bar is a menu bar with the items 'Datei', 'Bearbeiten', 'Ansicht', 'Terminal', and 'Hilfe'. The main area of the terminal shows the command prompt 'vadmin@vadmin:~/Desktop/Delib Samples Linux/delib-cli\$' followed by the command 'sudo ./delib\_cli D08 0 255'.

```
vadmin@vadmin:~/Desktop/Delib Samples Linux/delib-cli$ sudo ./delib_cli D08 0 255
```

#### 4.4.2.2. DELIB CLI Beispiele

##### Digitale Ausgänge

```
sudo delib_cli DO1 17 1
```

→ schaltet das 18. digitale Relais eines USB-Moduls an

```
sudo delib_cli DO1 3 0
```

→ schaltet das 4. digitale Relais eines RO-ETH-Moduls aus

##### Digitale Eingänge

```
sudo delib_cli DI1 3
```

Beispiel eines Rückgabewertes: **1**

→ lese den Zustand des 4. digitalen Eingangs eines USB-Moduls und gebe ihn zurück

```
sudo delib_cli DI8 0 hex
```

Beispiel eines Rückgabewertes: **0xFF**

(auf den Kanälen 1 bis 8 liegt ein Signal an)

→ lese den Wert von digitalen Eingang 1-8 eines RO-ETH-Moduls als hexadezimalzahl

```
sudo delib_cli FF 0
```

Beispiel eines Rückgabewertes: **192**

(auf den Kanälen 7 und 8 wurde eine Zustandsänderung erkannt)

→ lese den Wert der FlipFlops der digitalen Eingänge 1-32

```
sudo delib_cli FF 32
```

Beispiel eines Rückgabewertes: **65535**

(auf den Kanälen 33 bis 64 wurde eine Zustandsänderung erkannt)

→ lese den Wert der FlipFlops der digitalen Eingänge 33-64

```
sudo delib_cli FF 0 hex
```

Beispiel eines Rückgabewertes: **0xD00**

(auf Kanälen 9, 11 und 12 wurde eine Zustandsänderung erkannt)

→ lese den Wert der FlipFlops der digitalen Eingänge 1-32 als hexadezimalzahl

### Analoge Ausgänge

```
sudo delib_cli AO 7 4711
```

→ setzt den dezimalen Wert 4711 auf den 8. analogen Ausgang eines USB-Moduls

```
sudo delib_cli AO 6 0x4711
```

→ setzt den hexadezimalen Wert 0x4AF1 auf den 7. analogen Ausgang eines RO-ETH-Moduls

### Analoge Eingänge

```
sudo delib_cli AI 2
```

Beispiel eines Rückgabewertes: **1234**

→ liest den Wert des 3. analogen Eingangs als dezimalzahl eines USB-Moduls

```
sudo delib_cli AI 2 hex
```

Beispiel eines Rückgabewertes: **0x1FA**

→ liest den Wert des 3. analogen Eingangs als hexadezimalzahl eines RO-ETH-Moduls

## **DELIB API Referenz**



## 5. DELIB API Referenz

### 5.1. Verfügbare DEDITEC Modul IDs

Hier finden Sie eine Auflistung mit allen verfügbaren Modul IDs.

Diese ID wird benötigt, um beispielsweise das Modul zu öffnen und einen "handle" zu erhalten.

Mehr Informationen dazu finden Sie im Kapitel **DapiOpenModule**.

Modul Name	ID
USB_Interface8	1
USB_CAN_STICK	2
USB_LOGI_500	3
USB_SER_DEBUG	4
RO_SER	5
USB_BITP_200	6
RO_USB1	7
RO_USB	7
RO_ETH	8
USB_MINI_STICK	9
USB_LOGI_18	10
RO_CAN	11
USB_SPI_MON	12
USB_WATCHDOG	13
USB_OPTOIN_8	14

Modul Name	ID
USB_RELAIS_8	14
USB_OPTOIN_8_RELAIS_8	15
USB_OPTOIN_16_RELAIS_16	16
USB_OPTOIN_32	16
USB_RELAIS_32	16
USB_OPTOIN_32_RELAIS_32	17
USB_OPTOIN_64	17
USB_RELAIS_64	17
BS_USB_8	15
BS_USB_16	16
BS_USB_32	17
USB_TTL_32	18
USB_TTL_64	18
RO_ETH_INTERN	19
BS_SER	20
BS_CAN	21
BS_ETH	22
NET_ETH	23
RO_CAN2	24
RO_USB2	25
RO_ETH_LC	26

Modul Name	ID
ETH_RELAIS_8	27
ETH_OPTOIN_8	27
ETH_O4_R4_ADDA	28
ETHERNET_MODULE	29
ETH_TTL_64	30
NET_USB2	31
NET_ETH_LC	32
NET_USB1	33
NET_SER	34
NET_CAN_OPEN	35
NET_RAS_PI	36
USB_CANOPEN_STICK	37
ETH_CUST_0	38
WEU_RELAIS_8	39
WEU_OPTO_8	39
WEU_E_RELAIS_8	40
BS_WEU	41
BS_WEU_E	42



## 5.2. Verwaltungsfunktionen

### 5.2.1. DapiOpenModule

#### Beschreibung

Diese Funktion öffnet ein bestimmtes Modul.

#### Definition

*ULONG DapiOpenModule(ULONG moduleID, ULONG nr);*

#### Parameter

moduleID=Gibt das Modul an, welches geöffnet werden soll (siehe delib.h)

nr=Gibt an, welches (bei mehreren Modulen) geöffnet werden soll.

nr=0 → 1. Modul

nr=1 → 2. Modul

#### Return-Wert

handle=Entsprechender Handle für das Modul

handle=0 → Modul wurde nicht gefunden

#### Bemerkung

Der von dieser Funktion zurückgegebene Handle wird zur Identifikation des Moduls für alle anderen Funktionen benötigt.

#### Programmierbeispiel

```
// USB-Modul öffnen
handle = DapiOpenModule(RO_USB1, 0);
printf("handle = %x\n", handle);
if (handle==0)
{
// USB Modul wurde nicht gefunden
printf("Modul konnte nicht geöffnet werden\n");
return;
}
```

### 5.2.2. DapiCloseModule

#### Beschreibung

Dieser Befehl schließt ein geöffnetes Modul.

#### Definition

*ULONG DapiCloseModule(ULONG handle);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls.

#### Return-Wert

Keiner

#### Programmierbeispiel

```
// Modul schliessen  
DapiCloseModule(handle);
```

### 5.2.3. DapiGetDELIBVersion

#### Beschreibung

Diese Funktion gibt die installierte DELIB-Version zurück.

#### Definition

*ULONG DapiGetDELIBVersion(ULONG mode, ULONG par);*

#### Parameters

mode=Modus, mit dem die Version ausgelesen wird (muss 0 sein).

par=Dieser Parameter ist nicht definiert (muss 0 sein).

#### Return-Wert

version=Versionsnummer der installierten DELIB-Version [hex].

#### Programmierbeispiel

```
version = DapiGetDELIBVersion(0, 0);  
//Bei installierter Version 1.32 ist Version = 132(hex)
```

#### 5.2.4. DapiSpecialCMDGetModuleConfig

##### Beschreibung

Diese Funktion gibt die Hardwareausstattung (Anzahl der Ein- und Ausgangskanäle) des Moduls zurück.

##### Definition

```
ULONG DapiSpecialCommand(ULONG handle,  
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG, par, 0, 0);
```

##### Parameter

handle=Dies ist der handle eines offenen Moduls

##### Querying the number of digital input channels

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_DI

##### Query number of digital input flip-flops

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_DI\_FF

##### Query number of digital input counters (16-bit counter)

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_DI\_COUNTER

##### Query number of digital input counters (48-bit counter)

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_CNT48

##### Querying the number of digital output channels

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_DO

##### Querying the number of digital pulse generator outputs

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_PULSE\_GEN

##### Querying the number of digital PWM outputs

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_PWM\_OUT

##### Querying the number of digital input/output channels

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_DX

**Querying the number of analog input channels**

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_AD

**Querying the number of analog output channels**

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_DA

**Query number of temperature channels**

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_TEMP

**Query number of stepper channels**

par=DAPI\_SPECIAL\_GET\_MODULE\_CONFIG\_PAR\_STEPPER

**Return value**

**Querying the number of digital input channels**

return=number of digital input channels

**Query number of digital input flip-flops**

return=number of digital input flip-flops

**Query number of digital input counters (16-bit counter)**

return=number of digital input counters (16-bit counter)

**Query number of digital input counters (48-bit counter)**

return=number of digital input counters (48-bit counter)

**Querying the number of digital output channels**

return=number of digital output channels

**Querying the number of digital pulse generator outputs**

return=number of digital pulse generator outputs

**Querying the number of digital PWM outputs**

return=number of digital PWM outputs

#### **Querying the number of digital input/output channels**

return=number of digital input/output channels

#### **Querying the number of analog input channels**

return=number of analog input channels

#### **Querying the number of analog output channels**

return=number of analog output channels

#### **Query number of temperature channels**

return=number of temperature channels

#### **Query number of stepper channels**

return=number of stepper channels

#### **Programmierbeispiele**

```
ret=DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG,
DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_DI, 0, 0);
//Returns the number of digital input channels

ret=DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG,
DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_DO, 0, 0);
//Returns the number of digital output channels

ret=DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG,
DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_DX, 0, 0);
//Returns the number of digital input/output channels

ret=DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG,
DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_AD, 0, 0);
//Returns the number of analog input channels

ret=DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG,
DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_DA, 0, 0);
//Returns the number of analog output channels

ret=DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_GET_MODULE_CONFIG,
DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_STEPPER, 0, 0);
//Returns the number of stepper channels
```

### 5.2.5. DapiOpenModuleEx

#### Beschreibung

Diese Funktion öffnet gezielt ein Modul mit Ethernet-Schnittstelle. Dabei können die Parameter IP-Adresse, Portnummer, die Dauer des Timeouts und der Encryption Type bestimmt werden.

Das Öffnen des Moduls geschieht dabei unabhängig von den im DELIB Configuration Utility getroffenen Einstellungen.

#### Definition

```
ULONG DapiOpenModuleEx(ULONG moduleID, ULONG nr, unsigned char*  
exbuffer, 0);
```

#### Parameter

moduleID = Gibt das Modul an, welches geöffnet werden soll (siehe delib.h)

nr = Gibt an, welches Modul (bei mehreren Modulen) geöffnet werden soll.

nr = 0 → 1. Modul

nr = 1 → 2. Modul

exbuffer = Buffer für IP-Adresse, Portnummer, Dauer des Timeouts und der Encryption Type

#### Return-Wert

handle = Entsprechender Handle für das Modul

handle = 0 → Modul wurde nicht gefunden

### Bemerkung

Der von dieser Funktion zurückgegebene Handle wird zur Identifikation des Moduls für alle anderen Funktionen benötigt.

Dieser Befehl wird von allen Modulen mit Ethernet-Schnittstelle unterstützt.

Universelle Ethernet moduleID

### Die moduleID:

ETHERNET\_MODULE = 29

ist eine universelle Ethernet moduleID und kann benutzt werden, um jedes Ethernet Produkt anzusprechen.

### Encryption Type

Folgende Encryption Types stehen zur Verfügung:

DAPI\_OPEN\_MODULE\_ENCRYPTION\_TYPE\_NONE = 0

DAPI\_OPEN\_MODULE\_ENCRYPTION\_TYPE\_NORMAL = 1

DAPI\_OPEN\_MODULE\_ENCRYPTION\_TYPE\_ADMIN = 2

### Programmierbeispiel

```
// Open ETH-Module with parameter
DAPI_OPENMODULEEX_STRUCT open_buffer;

strcpy((char*) open_buffer.address, "192.168.1.10");
open_buffer.portno = 0;
open_buffer.timeout = 5000;
open_buffer.encryption_type = 0;

handle = DapiOpenModuleEx(RO_ETH, 0, (unsigned char*)
&open_buffer, 0);
printf("Module handle = %x\n", handle);
```

### 5.2.6. DapiScanAllModulesAvailable

#### Beschreibung

Mit dieser Funktion lassen sich alle am USB-Bus angeschlossenen Module scannen.

Hierbei wird die Modul-ID und die Modul-Nr. jedes gefundenen Modules ermittelt.

#### Definition

*ULONG DapiScanAllModulesAvailable(uint nr)*

#### Parameter

nr = 0: Es wird nach allen am USB-Bus angeschlossenen Module gesucht  
nr = i: Auslesen der einzelnen angeschlossenen Module

#### Return-Wert

Gibt die Anzahl der gefunden Module wieder.

#### Programmierbeispiel

```
no_of_modules =  
DT.Delib.DapiScanAllModulesAvailable(0);  
for (i = 1; i <= no_of_modules; i++)  
{  
    ret = DapiScanAllModulesAvailable(i);  
    moduleID = ret & 0x0000ffff;  
    moduleNr = (ret >> 16) & 0xff;  
}
```



## 5.3. Fehlerbehandlung

### 5.3.1. DapiGetLastError

#### Beschreibung

Diese Funktion liefert den letzten erfassten Fehler. Sofern ein Fehler aufgetreten ist, muss dieser mit **DapiClearLastError()** gelöscht werden, da sonst jeder Aufruf von DapiGetLastError() den "alten" Fehler zurückgibt.

Sollen mehrere Module verwendet werden, empfiehlt sich die Verwendung von **DapiGetLastErrorByHandle()**.

#### Definition

```
ULONG DapiGetLastError();
```

#### Parameter

Keine

#### Return-Wert

Fehler Code

0=kein Fehler. (siehe delib\_error\_codes.h)

#### Programmierbeispiel

```
BOOL IsError()
{
    unsigned char msg[500];
    unsigned long error_code = DapiGetLastError();
    if (error_code != DAPI_ERR_NONE)
    {
        DapiGetLastErrorText((unsigned char*) msg,
        sizeof(msg));
        printf("Error Code = 0x%x * Message = %s\n",
        error_code, msg);
        DapiClearLastError();
        return TRUE;
    }
    return FALSE;
}
```

### 5.3.2. DapiGetLastErrorText

#### Beschreibung

Diese Funktion liest den Text des letzten erfassten Fehlers. Sofern ein Fehler aufgetreten ist, muss dieser mit **DapiClearLastError()** gelöscht werden, da sonst jeder Aufruf von DapiGetLastErrorText() den "alten" Fehler zurückgibt.

#### Definition

*ULONG DapiGetLastErrorText(unsigned char \* msg, unsigned long msg\_length);*

#### Parameter

msg = Buffer für den zu empfangenden Text

msg\_length = Länge des Text Buffers

#### Programmierbeispiel

```
BOOL IsError()
{
    unsigned char msg[500];
    unsigned long error_code = DapiGetLastError();

    if (error_code != DAPI_ERR_NONE)
    {
        DapiGetLastErrorText((unsigned char*) msg,
        sizeof(msg));
        printf("Error Code = 0x%x * Message = %s\n",
        error_code, msg);

        DapiClearLastError();

        return TRUE;
    }

    return FALSE;
}
```

### 5.3.3. DapiClearLastError

#### Beschreibung

Diese Funktion löscht den letzten Fehler, der mit **DapiGetLastError()** erfasst wurde.

#### Definition

```
void DapiClearLastError();
```

#### Parameter

Keine

#### Return-Wert

Keine

#### Programmierbeispiel

```
BOOL IsError()
{
    unsigned char msg[500];
    unsigned long error_code = DapiGetLastError();

    if (error_code != DAPI_ERR_NONE)
    {
        DapiGetLastErrorText((unsigned char*) msg,
        sizeof(msg));
        printf("Error Code = 0x%x * Message = %s\n",
        error_code, msg);

        DapiClearLastError();

        return TRUE;
    }

    return FALSE;
}
```

### 5.3.4. DapiGetLastErrorByHandle

#### Beschreibung

Diese Funktion liefert den letzten erfassten Fehler eines bestimmten Moduls (handle). Sofern ein Fehler aufgetreten ist, muss dieser mit **DapiClearLastErrorByHandle()** gelöscht werden, da sonst jeder Aufruf von DapiGetLastErrorByHandle() den "alten" Fehler zurückgibt.

#### Definition

*ULONG DapiGetLastErrorByHandle(ULONG handle);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls.

#### Return-Wert

Fehler Code

0=kein Fehler. (siehe delib\_error\_codes.h)

#### Programmierbeispiel

```
BOOL IsError(ULONG handle)
{
    unsigned long error_code =
    DapiGetLastErrorByHandle(handle);

    if (error_code != DAPI_ERR_NONE)
    {
        printf("Error detected on handle 0x%x - Error
Code = 0x%x\n", handle, error_code);

        DapiClearLastErrorByHandle(handle);

        return TRUE;
    }

    return FALSE;
}
```

### 5.3.5. DapiClearLastErrorByHandle

#### Beschreibung

Diese Funktion löscht den letzten Fehler eines bestimmten Moduls (handle), der mit **DapiGetLastErrorByHandle()** erfasst wurde.

#### Definition

```
void DapiClearLastErrorByHandle();
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls.

#### Return-Wert

Keine

#### Programmierbeispiel

```
BOOL IsError(ULONG handle)
{
    unsigned long error_code =
    DapiGetLastErrorByHandle(handle);

    if (error_code != DAPI_ERR_NONE)
    {
        printf("Error detected on handle 0x%x - Error
        Code = 0x%x\n", handle, error_code);

        DapiClearLastErrorByHandle(handle);

        return TRUE;
    }

    return FALSE;
}
```

## 5.4. Digitale Eingänge lesen

### 5.4.1. DapiDIGet1

#### Beschreibung

Dieser Befehl liest einen einzelnen digitalen Eingang.

#### Definition

*ULONG DapiDIGet1(ULONG handle, ULONG ch);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, der gelesen werden soll (0, 1, 2, 3, .. )

#### Return-Wert

Zustand des Eingangs (0/1)

### 5.4.2. DapiDIGet8

#### Beschreibung

Dieser Befehl liest gleichzeitig 8 digitale Eingänge.

#### Definition

*ULONG DapiDIGet8(ULONG handle, ULONG ch);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, ab dem gelesen werden soll (0, 8, 16, 24, .. )

#### Return-Wert

Zustand der gelesen Eingänge

### 5.4.3. DapiDIGet16

#### Beschreibung

Dieser Befehl liest gleichzeitig 16 digitale Eingänge.

#### Definition

*ULONG DapiDIGet16(ULONG handle, ULONG ch);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, ab dem gelesen werden soll (0, 16, 32, ...)

#### Return-Wert

Zustand der gelesenen Eingänge

#### 5.4.4. DapiDIGet32

##### Beschreibung

Dieser Befehl liest gleichzeitig 32 digitale Eingänge.

##### Definition

*ULONG DapiDIGet32(ULONG handle, ULONG ch);*

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, ab dem gelesen werden soll (0, 32, 64, ..)

##### Return-Wert

Zustand der gelesenen Eingänge

##### Programmierbeispiel

```
unsigned long data;
// -----
// Einen Wert von den Eingängen lesen (Eingang 1-31)
data = (unsigned long) DapiDIGet32(handle, 0);
// Chan Start = 0
printf("Eingang 0-31 : 0x%x\n", data);
printf("Taste für weiter\n");
getch();
// -----
// Einen Wert von den Eingängen lesen (Eingang 32-64)
data = (unsigned long) DapiDIGet32(handle, 32);
// Chan Start = 32
printf("Eingang 32-64 : 0x%x\n", data);
printf("Taste für weiter\n");
getch();
```



#### 5.4.5. DapiDIGet64

##### Beschreibung

Dieser Befehl liest gleichzeitig 64 digitale Eingänge.

##### Definition

*ULONG DapiDIGet64(ULONG handle, ULONG ch);*

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, ab dem gelesen werden soll (0, 64, ..)

##### Return-Wert

Zustand der gelesenen Eingänge

#### 5.4.6. DapiDIGetFF32

##### Beschreibung

Dieser Befehl liest die Flip-Flops der Eingänge aus und setzt diese zurück (Eingangszustands-Änderung).

##### Definition

*ULONG DapiDIGetFF32(ULONG handle, ULONG ch);*

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, ab dem gelesen werden soll (0, 32, 64, ..)

##### Return-Wert

Zustand von 32 Eingangszustandsänderungen

### 5.4.7. DapiDIGetCounter

#### Beschreibung

Dieser Befehl liest den Eingangszähler eines digitalen Eingangs.

#### Definition

*ULONG DapiDIGetCounter(ULONG handle, ULONG ch, ULONG mode);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangs an, ab dem gelesen werden soll

mode=0 (Normale Zählfunktion)

mode=DAPI\_CNT\_MODE\_READ\_WITH\_RESET (Zähler auslesen und direktes Counter resettet)

mode=DAPI\_CNT\_MODE\_READ\_LATCHED (Auslesen des gespeicherten Zählerstandes)

#### Return-Wert

Angabe des Zählerwertes

#### Programmierbeispiel

```
value = DapiDIGetCounter(handle, 0 , 0);  
// Zähler von DI Chan 0 wird gelesen  
  
value = DapiDIGetCounter(handle, 1 , 0);  
// Zähler von DI Chan 1 wird gelesen  
  
value = DapiDIGetCounter(handle, 8 ,0);  
// Zähler von DI Chan 8 wird gelesen  
  
value = DapiDIGetCounter(handle, 0 ,  
DAPI_CNT_MODE_READ_WITH_RESET);  
// Zähler von DI Chan 0 wird gelesen UND resettet  
  
value = DapiDIGetCounter(handle, 1 ,  
DAPI_CNT_MODE_READ_LATCHED);  
// Auslesen des gespeicherten Zählerstandes von DI Chan 1
```

#### 5.4.8. DapiSpecialCounterLatchAll

##### Beschreibung

Dieser Befehl speichert die Zählerstände aller Eingangszähler gleichzeitig in ein Zwischenspeicher (Latch).

So können anschließend alle Zählerstände des Latches nacheinander ausgelesen werden.

Besonderheit hierbei ist, dass ein gleichzeitiges "Einfrieren" der Zählerstände möglich ist und die eingefrorenen Stände (Latch) dann einzeln nacheinander ausgelesen werden können.

##### Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_COUNTER,  
DAPI_SPECIAL_COUNTER_LATCH_ALL, 0, 0);
```

##### Parameter

Keine

##### Return-Wert

Keiner

##### Bemerkung

Module, die von diesen Befehlen unterstützt werden, können Sie unserer **DELIB Übersichtstabelle** entnehmen.

##### Programmierbeispiel

```
DapiSpecialCommand(ULONG handle,  
DAPI_SPECIAL_CMD_COUNTER,  
DAPI_SPECIAL_COUNTER_LATCH_ALL, 0, 0);
```

#### 5.4.9. DapiSpecialCounterLatchAllWithReset

##### Beschreibung

Dieser Befehl speichert die Zählerstände aller Eingangszähler gleichzeitig in einen Zwischenspeicher (Latch).

Zusätzlich werden die Zählerstände der Eingangszähler im Anschluss zurückgesetzt.

##### Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_COUNTER,  
DAPI_SPECIAL_COUNTER_LATCH_ALL_WITH_RESET, 0, 0);
```

##### Parameter

Keine

##### Return-Wert

Keiner

##### Bemerkung

Module, die von diesen Befehlen unterstützt werden, können Sie unserer

**DELIB Übersichtstabelle** entnehmen.

##### Programmierbeispiel

```
DapiSpecialCommand(ULONG handle,  
DAPI_SPECIAL_CMD_COUNTER,  
DAPI_SPECIAL_COUNTER_LATCH_ALL_WITH_RESET, 0, 0);
```

#### 5.4.10. DapiSpecialDIFilterValueSet

##### Beschreibung

Dieser Befehl setzt einen Eingansfilter in [ms], in welchem Zeitintervall Störimpulse bei digitalen Eingangskanälen, gefiltert werden.

##### Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FILTER_VALUE_SET, ULONG time_ms, 0);
```

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

time\_ms=Zeitintervall [ms], indem digitale Eingangskanäle gelesen werden.

##### Bemerkung

Standardwert: 0ms

Wertebereich: 0(=off) , 1(ms) - 254(ms)

Module, die von diesen Befehlen unterstützt werden, können Sie unserer **DELIB Übersichtstabelle** entnehmen.

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FILTER_VALUE_SET, 5, 0);  
// Setzt das Zeitintervall auf 5ms  
  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FILTER_VALUE_SET, 150, 0);  
// Setzt das Zeitintervall auf 150ms
```

#### 5.4.11. DapiSpecialDIFilterValueGet

##### Beschreibung

Dieser Befehl gibt den vorher festgelegten Wert des Zeitintervalls zur Filterung von Störimpulsen bei digitalen Eingangskanäle in [ms] zurück.

##### Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FILTER_VALUE_GET, 0, 0);
```

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

##### Return-Wert

Zeit [ms]

##### Bemerkung

Module, die von diesen Befehlen unterstützt werden, können Sie unserer

**DELIB Übersichtstabelle** entnehmen.

##### Programmierbeispiel

```
value = DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FILTER_VALUE_GET, 0, 0);  
//Gibt das Zeitintervall zum Auslesen der digitalen Eingangskanäle zurück.
```

#### 5.4.12. Dapi\_Special\_DI\_FF\_Filter\_Value\_Set

##### Beschreibung

Dieser Befehl setzt einen Filter [ms], in welchem Zeitintervall die Eingangs-Flip-Flops und die Eingangs-Zähler, abgefragt werden.

##### Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FF_FILTER_VALUE_SET, ULONG time_ms, 0);
```

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

time\_ms=Zeitintervall [ms], indem digitale Eingangskanäle abgetastet werden.

##### Bemerkung

Dieser Befehl unterstützt nur Impulszeiten zwischen 5ms und 255ms.

Wird keine Zeit gesetzt, ist der Default-Wert 100ms.

Module, die von diesen Befehlen unterstützt werden, können Sie unserer **DELIB Übersichtstabelle** entnehmen.

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FF_FILTER_VALUE_SET, 5, 0);  
// Setzt das Zeitintervall auf 5ms  
  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FF_FILTER_VALUE_SET, 150, 0);  
// Setzt das Zeitintervall auf 150ms
```



### 5.4.13. Dapi\_Special\_DI\_FF\_Filter\_Value\_Get

#### Beschreibung

Dieser Befehl gibt den vorher festgelegten Wert des Zeitintervalls zur Abtastung der Eingangs-Flip-Flops und der Eingangs-Zähler in [ms] zurück.

#### Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FF_FILTER_VALUE_GET, 0, 0);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

#### Return-Wert

Zeit [ms]

#### Bemerkung

Module, die von diesen Befehlen unterstützt werden, können Sie unserer

**DELIB Übersichtstabelle** entnehmen.

#### Programmierbeispiel

```
value = DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DI,  
DAPI_SPECIAL_DI_FF_FILTER_VALUE_GET, 0, 0);  
//Gibt das Zeitintervall zum Abtasten der digitalen Eingangskanäle zurück.
```

## 5.5. Digitale Ausgänge verwalten

### 5.5.1. DapiDOSet1

#### Beschreibung

Dieser Befehl setzt einen einzelnen Ausgang.

#### Definition

```
void DapiDOSet1(ULONG handle, ULONG ch, ULONG data);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des zu setzenden Ausgangs an (0 .. )

data=Gibt den Datenwert an, der geschrieben wird (0 / 1)

#### Return-Wert

Keiner

### 5.5.2. DapiDOSet8

#### Beschreibung

Dieser Befehl setzt gleichzeitig 8 digitale Ausgänge.

#### Definition

```
void DapiDOSet8(ULONG handle, ULONG ch, ULONG data);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 8, 16, 24, 32, ..)

data=Gibt die Datenwerte an, die geschrieben werden

#### Return-Wert

Keiner

### 5.5.3. DapiDOSet16

#### Beschreibung

Dieser Befehl setzt gleichzeitig 16 digitale Ausgänge.

#### Definition

```
void DapiDOSet16(ULONG handle, ULONG ch, ULONG data);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 16, 32, ..)

data=Gibt die Datenwerte an, die geschrieben werden

#### Return-Wert

Keiner

#### 5.5.4. DapiDOSet32

##### Beschreibung

Dieser Befehl setzt gleichzeitig 32 digitale Ausgänge.

##### Definition

```
void DapiDOSet32(ULONG handle, ULONG ch, ULONG data);
```

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 32, 64, ..)

data=Gibt die Datenwerte an, die geschrieben werden

##### Return-Wert

Keiner

##### Programmierbeispiel

```
// Einen Wert auf die Ausgänge schreiben
data = 0x0000ff00; // Ausgänge 9-16 werden auf 1
gesetzt
DapiDOSet32(handle, 0, data); // Chan Start = 0
printf("Schreibe auf Ausgänge Daten=0x%x\n", data);
printf("Taste für weiter\n");
getch();
// -----

// Einen Wert auf die Ausgänge schreiben
data = 0x80000000; // Ausgang 32 wird auf 1 gesetzt
DapiDOSet32(handle, 0, data); // Chan Start = 0
printf("Schreibe auf Ausgänge Daten=0x%x\n", data);
printf("Taste für weiter\n");
getch();
// -----

// Einen Wert auf die Ausgänge schreiben
data = 0x80000000; // Ausgang 64 wird auf 1 gesetzt
DapiDOSet32(handle, 32, data); // Chan Start = 32
printf("Schreibe auf Ausgänge Daten=0x%x\n", data);
printf("Taste für weiter\n");
getch();
```

### 5.5.5. DapiDOSet64

#### Beschreibung

Dieser Befehl setzt gleichzeitig 64 digitale Ausgänge.

#### Definition

```
void DapiDOSet64(ULONG handle, ULONG ch, ULONG data);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 64, ..)

data=Gibt die Datenwerte an, die geschrieben werden

#### Return-Wert

Keiner

### 5.5.6. DapiDOSet1\_WithTimer

#### Beschreibung

Diese Funktion setzt einen Digitalausgang (ch) auf einen Wert (data - 0 oder 1) für eine bestimmte Zeit in ms.

#### Definition

```
void DapiDOSet1_WithTimer(ULONG handle, ULONG ch, ULONG data, ULONG  
time_ms);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 32, 64, ..)

data=Gibt die Datenwerte an, die geschrieben werden

time\_ms=Gibt die Zeit an, in der der Ausgang gesetzt wird [ms]

#### Return-Wert

Keiner

#### Bemerkung:

Dieser Befehl wird nur von unserem RO-O8-R8 Modul unterstützt.

Dieser Befehl verliert seine Gültigkeit, sofern er mit anderen Werten überschrieben wird.

Möchte man den Befehl deaktivieren, dann muss er mit time\_ms=0 überschrieben werden.

Module, die von diesen Befehlen unterstützt werden, können Sie unserer

**DELIB Übersichtstabelle** entnehmen.

#### Programmierbeispiel

```
DapiDOSet1_WithTimer(handle, 2, 1, 1000);  
//Setting channel 2 for 1000msec to 1
```

### 5.5.7. DapiDOReadback32

#### Beschreibung

Dieser Befehl liest die 32 digitalen Ausgänge zurück.

#### Definition

*ULONG DapiDOReadback32(ULONG handle, ULONG ch);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem zurückgelesen werden soll (0, 32, 64, ..)

#### Return-Wert

Zustand von 32 Ausgängen.

### 5.5.8. DapiDOReadback64

#### Beschreibung

Dieser Befehl liest die 64 digitalen Ausgänge zurück.

#### Definition

*ULONG DapiDOReadback64(ULONG handle, ULONG ch);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, ab dem zurückgelesen werden soll (0, 64, ..)

#### Return-Wert

Zustand von 64 Ausgängen.

### 5.5.9. DapiDOSetBit32

#### Beschreibung

Mit diesem Befehl können Ausgänge gezielt auf 1 geschaltet werden, ohne die Zustände der benachbarten Ausgänge zu ändern.

#### Definition

```
void DapiDOSetBit32(uint handle, uint ch, uint data);
```

#### Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll

data = Gibt den Datenwert an, der geschrieben werden soll (bis zu 32 Bit)

#### Return-Wert

Keiner

#### Bemerkung:

Nur die Bits mit einer Wertigkeit von 1 im data Parameter werden vom Befehl berücksichtigt.

#### Programmierbeispiel

```
data = 0x1; // Ausgang 0 wird auf 1 gesetzt, der Zustand von Ausgang
1-31 bleibt unberührt
DapiDOSetBit32(handle, 0, data);
data = 0xf; // Ausgang 0-3 wird auf 1 gesetzt, der Zustand von Ausgang
4-31 bleibt unberührt
DapiDOSetBit32(handle, 0, data);
data = 0xff; // Ausgang 0-7 wird auf 1 gesetzt, der Zustand von
Ausgang 8-31 bleibt unberührt
DapiDOSetBit32(handle, 0, data);
data = 0xff000000; // Ausgang 23-31 wird auf 1 gesetzt, der Zustand
von Ausgang 0-22 bleibt unberührt
DapiDOSetBit32(handle, 0, data);
```



### 5.5.10. DapiDOClrBit32

#### Beschreibung

Mit diesem Befehl können Ausgänge gezielt auf 0 geschaltet werden, ohne die Zustände der benachbarten Ausgänge zu ändern.

#### Definition

```
void DapiDOClrBit32(uint handle, uint ch, uint data);
```

#### Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll

data = Gibt den Datenwert an, der geschrieben werden soll (bis zu 32 Bit)

#### Return-Wert

Keiner

#### Bemerkung:

Nur die Bits mit einer Wertigkeit von 1 im data Parameter werden vom Befehl berücksichtigt.

#### Programmierbeispiel

```
data = 0x1; // Ausgang 0 wird auf 0 gesetzt, der Zustand von Ausgang  
1-31 bleibt unberührt  
DapiDOSetBit32(handle, 0, data);  
data = 0xf; // Ausgang 0-3 wird auf 0 gesetzt, der Zustand von Ausgang  
4-31 bleibt unberührt  
DapiDOSetBit32(handle, 0, data);  
data = 0xff; // Ausgang 0-7 wird auf 0 gesetzt, der Zustand von  
Ausgang 8-31 bleibt unberührt  
DapiDOSetBit32(handle, 0, data);  
data = 0xff000000; // Ausgang 23-31 wird auf 0 gesetzt, der Zustand  
von Ausgang 0-22 bleibt unberührt  
DapiDOSetBit32(handle, 0, data);
```

## 5.6. TTL-Ein-/Ausgangs Richtungen setzen mit DapiSpecialCommand

### 5.6.1. DAPI\_SPECIAL\_CMD\_SET\_DIR\_DX\_1

#### Beschreibung

Dieser Befehl setzt die Richtung von 8 hintereinanderliegenden TTL-Ein/Ausgängen (1-Bit weise).

#### Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_SET_DIR_DX_1,  
    ULONG ch, ULONG dir, 0);
```

#### Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Muss immer 0 sein!

dir = Gibt die Richtung für 8 Kanäle an (1=output / 0=input) / Bit 0 steht für Kanal 0, Bit 1 für Kanal 1 ...

#### Bemerkung

Nicht kompatibel mit USB-TTL-32/64.

Verwenden Sie für diese Module den DAPI\_SPECIAL\_CMD\_SET\_DIR\_DX\_8 Befehl.

#### Programmierbeispiel

```
DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x01 , 0);  
// Set Dir of TTL-I/O CH0 to output, others to input  
DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x02 , 0);  
// Set Dir of TTL-I/O CH1 to output, others to input  
DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x04 , 0);  
// Set Dir of TTL-I/O CH2 to output, others to input  
DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x08 , 0);  
// Set Dir of TTL-I/O CH3 to output, others to input  
DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x10 , 0);  
// Set Dir of TTL-I/O CH4 to output, others to input
```

```
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x20 , 0);  
// Set Dir of TTL-I/O CH5 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x40 , 0);  
// Set Dir of TTL-I/O CH6 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x80 , 0);  
// Set Dir of TTL-I/O CH7 to output, others to input  
  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0x0f , 0);  
// Set Dir of TTL-I/O CH0-3 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_1, 0, 0xff , 0);  
// Set Dir of TTL-I/O CH0-7 to output, others to input
```

### 5.6.2. DAPI\_SPECIAL\_CMD\_SET\_DIR\_DX\_8

#### Beschreibung

Dieser Befehl setzt die Richtung von bis zu 64 hintereinanderliegenden TTL-Ein/Ausgängen (8-Bit weise).

1-Bit repräsentiert dabei 8 TTL-Ein/Ausgänge.

#### Definition

```
void DapiSpecialCommand(ULONG handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_8, ULONG ch, ULONG dir, 0);
```

#### Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Muss immer 0 sein!

dir = (8-Bit) gibt die Richtung für bis zu 64 hintereinanderliegende TTL-Ein/Ausgänge an. (1=output / 0=input)

#### Bemerkung

Nur kompatibel mit USB-TTL-32/64.

Verwenden Sie für andere TTL-Produkte den DAPI\_SPECIAL\_CMD\_SET\_DIR\_DX\_1 Befehl.

#### Programmierbeispiel

```
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_8, 0, 0x1 , 0);  
// Set Dir of TTL-I/O CH0-7 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_8, 0, 0x3 , 0);  
// Set Dir of TTL-I/O CH0-15 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_8, 0, 0xc , 0);  
// Set Dir of TTL-I/O CH16-31 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_8, 0, 0x33 , 0);  
// Set Dir of TTL-I/O CH0-15 and CH32-47 to output, others to input  
DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SET_DIR_DX_8, 0, 0xff , 0);  
// Set Dir of TTL-I/O CH0-63 to output, others to input
```

### 5.6.3. DAPI\_SPECIAL\_CMD\_GET\_DIR\_DX\_8

#### Beschreibung

Dieser Befehl liest die Richtung von bis zu 64 hintereinanderliegenden TTL-Ein/Ausgängen (8-Bit weise).

#### Definition

ULONG                      DapiSpecialCommand(ULONG                      handle,  
DAPI\_SPECIAL\_CMD\_GET\_DIR\_DX\_8, ULONG ch, ULONG dir, 0);

#### Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Muss immer 0 sein!

dir = Muss immer 0 sein!

#### Bemerkung

Nur kompatibel mit USB-TTL-32/64.

#### Return-Wert

Richtungszustand von 64 Kanälen.

Bit 0: Richtung von TTL 0-7	/ 1=Output, 0=Input
Bit 1: Richtung von TTL 8-15	/ 1=Output, 0=Input
Bit 2: Richtung von TTL 16-23	/ 1=Output, 0=Input
Bit 3: Richtung von TTL 24-31	/ 1=Output, 0=Input
Bit 4: Richtung von TTL 32-39	/ 1=Output, 0=Input
Bit 5: Richtung von TTL 40-47	/ 1=Output, 0=Input
Bit 6: Richtung von TTL 48-55	/ 1=Output, 0=Input
Bit 7: Richtung von TTL 56-63	/ 1=Output, 0=Input

#### Programmierbeispiel

```
ULONG ret = DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_GET_DIR_DX_8, 0, 0, 0);  
// Liest die Richtung von 64 Kanälen aus
```

## 5.7. Ausgabe-Timeout verwalten

### 5.7.1. DapiSpecialCMDTimeout

#### Beschreibung

Dieser Befehl dient zum Einstellen der Timeout-Schutz-Funktion.

Es gibt seit 2021 drei unterschiedliche Timeout-Methoden.

#### "normalen" Timeout

Dies ist der Timeout, den unsere Module schon seit 2009 besitzen.

Vorgehensweise für den Timeout-Befehl:

Der Timeout wird per Befehl aktiviert.

Findet dann ein sogenanntes Timeout-Ereignis statt (Pause zwischen zwei Zugriffen auf das Modul ist größer, als die erlaubte Timeout-Zeit) passiert Folgendes:

- Alle Ausgänge werden ausgeschaltet
- Der Timeout-Status geht auf "2"
- Die Timeout-LED geht an (bei Modulen, die solch einen Status haben)

Weitere Zugriffe auf die Ausgänge sind dann weiterhin möglich, aber der Timeout ist nicht weiter aktiv. Erst wieder, wenn er wieder aktiviert wurde.

### **"auto reactivate" Timeout**

Dies ist ein seit 2021 implementierter Timeout-Modus, der nach Auftreten des Timeout-Ereignisses den Timeout automatisch wieder aktiviert.

Vorgehensweise für den Timeout-Befehl:

Der Timeout wird per Befehl aktiviert.

Findet dann ein sogenanntes Timeout-Ereignis statt (Pause zwischen zwei Zugriffen auf das Modul ist größer, als die erlaubte Timeout-Zeit) passiert Folgendes:

- Alle Ausgänge werden ausgeschaltet
- Der Timeout-Status geht auf "4"
- Die Timeout-LED geht an (bei Modulen, die solch einen Status haben)

Weitere Zugriffe auf die Ausgänge sind dann weiterhin möglich. UND der Timeout ist weiter aktiv. Bei erneuter Zeitüberschreitung der Timeout-Zeit werden die Ausgänge wieder ausgeschaltet.

### **"secure outputs" Timeout**

Dies ist ein seit 2021 implementierter Timeout-Modus, der nach Auftreten des Timeout-Ereignisses einen Schreibenden Zugriff auf die Ausgänge verhindert. Somit wird sichergestellt, dass die Software erst einmal einen "sicheren" Zustand der Ausgänge wiederherstellen muss, da der Timeout-Mechanismus des Moduls die Ausgänge auf vordefinierte Werte verändert hat.

Vorgehensweise für den Timeout-Befehl:

Der Timeout wird per Befehl aktiviert.

Findet dann ein sogenanntes Timeout-Ereignis statt (Pause zwischen zwei Zugriffen auf das Modul ist größer, als die erlaubte Timeout-Zeit) passiert Folgendes:

- Alle Ausgänge werden ausgeschaltet
- Der Timeout-Status geht auf "6"
- Die Timeout-LED geht an (bei Modulen, die solch einen Status haben)

Weitere Zugriffe auf die Ausgänge sind NICHT möglich. Erst nach erneutem Aktivieren des Timeouts oder Deaktivieren des Timeouts können die Ausgänge beschrieben werden.

### **Definition**

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, par1, par2);*

### **Parameter**

handle=Dies ist das Handle eines geöffneten Moduls

cmd = auszuführende Funktion

par1 = Wert, der an die Funktion übergeben wird

par2 = Wert, der an die Funktion übergeben wird



#### 5.7.1.1. DapiSpecialTimeoutSetValueSec

##### Beschreibung

Dieser Befehl dient zum Setzen der Timeout-Zeit.

##### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, par1, par2);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_SET\_VALUE\_SEC

par1 = Sekunden [s]

par2 = Millisekunden [100ms] (Wert 6 = 600ms)

##### Bemerkung

Der zulässige Wertebereich der Zeitangabe liegt zwischen 0,1 Sekunden und 6553 Sekunden

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_SET_VALUE_SEC, 3, 7);  
//Die Zeit des Timeouts wird auf 3,7sek gesetzt.
```

### 5.7.1.2. DapiSpecialTimeoutActivate

#### Beschreibung

Dieser Befehl aktiviert den "normalen" Timeout.

Nach dem Timeout-Ereignis werden..

- ..alle Ausgänge ausgeschaltet
- ..der Timeout-Status auf "2" gesetzt
- ..die Timeout-LED angeschaltet (bei Modulen, die solch einen Status haben)

Weitere Zugriffe auf die Ausgänge sind dann weiterhin möglich, aber der Timeout ist nicht weiter aktiv.

Erst wieder, wenn er wieder aktiviert wurde.

#### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

#### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_ACTIVATE

#### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_ACTIVATE, 0, 0);  
//Der "normale" Timeout wird aktiviert.
```

### 5.7.1.3. DapiSpecialTimeoutActivateAutoReactivate

#### Beschreibung

Dieser Befehl aktiviert den "auto reactivate" Timeout.

In diesem Modus wird der Timeout nach dem Timeout-Ereignis automatisch wieder aktiviert.

Nach dem Timeout-Ereignis werden..

- ..alle Ausgänge ausgeschaltet
- ..der Timeout-Status auf "4" gesetzt
- ..die Timeout-LED angeschaltet (bei Modulen, die solch einen Status haben)

Weitere Zugriffe auf die Ausgänge sind dann weiterhin möglich UND der Timeout ist weiter aktiv.

Bei erneuter Zeitüberschreitung der Timeout-Zeit werden die Ausgänge wieder ausgeschaltet.

#### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

#### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_ACTIVATE\_AUTO\_REACTIVATE

#### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_ACTIVATE_AUTO_REACTIVATE, 0, 0);  
//Der "auto reactivate" Timeout wird aktiviert.
```

#### 5.7.1.4. DapiSpecialTimeoutActivateSecureOutputs

##### Beschreibung

Dieser Befehl aktiviert den "secure" Timeout.

In diesem Modus wird ein schreibender Zugriff auf die Ausgänge nach einem Timeout-Ereignis verhindert.

Somit wird sichergestellt, dass die Software erst einmal einen "sicheren" Zustand der Ausgänge wiederherstellen muss,

da der Timeout-Mechanismus des Moduls die Ausgänge auf vordefinierte Werte verändert hat.

Nach dem Timeout-Ereignis werden..

- ..alle Ausgänge ausgeschaltet
- ..der Timeout-Status auf "6" gesetzt
- ..die Timeout-LED angeschaltet (bei Modulen, die solch einen Status haben)

Weitere Zugriffe auf die Ausgänge sind NICHT möglich. Erst nach erneutem Aktivieren des

Timeouts oder Deaktivieren des Timeouts können die Ausgänge beschrieben werden.

##### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_ACTIVATE\_SECURE\_OUTPUTS

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_ACTIVATE_SECURE_OUTPUTS, 0, 0);  
//Der "secure" Timeout wird aktiviert.
```

#### 5.7.1.5. DapiSpecialTimeoutDeactivate

##### Beschreibung

Dieser Befehl deaktiviert den Timeout.

##### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_DEACTIVATE

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_DEACTIVATE, 0, 0);  
//Der Timeout wird deaktiviert.
```

### 5.7.1.6. DapiSpecialTimeoutGetStatus

#### Beschreibung

Dieser Befehl dient dem Auslesen des Timeout-Status.

#### Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_GET_STATUS, 0, 0);
```

#### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_GET\_STATUS

#### Return-Wert

Return = 0 (Timeout ist deaktiviert)

#### Werte für den "normalen" Timeout

Return = 1 (Timeout "normal" ist aktiviert)

Return = 2 (Timeout "normal" hat stattgefunden)

#### Werte für den "auto reactivate" Timeout

Return = 3 (Timeout "auto reactivate" ist aktiviert)

Return = 4 (Timeout "auto reactivate" hat ein oder mehrmals stattgefunden)

#### Werte für den "secure" Timeout

Return = 5 (Timeout "secure" ist aktiviert)

Return = 6 (Timeout "secure" hat stattgefunden. In diesem Status wird ein Schreiben auf die Outputs verhindert)

#### Programmierbeispiel

```
unsigned long status = DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_GET_STATUS, 0, 0);  
printf("Status = %lu\n", status);  
//Abfrage des Timeout-Status mit Ausgabe.
```

#### 5.7.1.7. DapiSpecialTimeoutDoValueMaskWRSet32

##### Beschreibung

Dieser Befehl bestimmt die Ausgänge, die bei einem Timeout gesetzt werden sollen.

##### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, ch, par2);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_DO\_VALUE\_MASK\_WR\_SET32

ch = Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 32, 64, ..)

par2 = [32 Bit] Gibt die Ausgänge an, welche bei einem Timeout aktiviert werden sollen

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_WR_SET32, 0, 0xff);  
//Die ersten 8 Relais werden im Timeout Fall eingeschaltet.
```

#### 5.7.1.8. DapiSpecialTimeoutDoValueMaskRDSet32

##### Beschreibung

Dieser Befehl dient dem Auslesen der übergebenen Werte.

##### Definition

*ULONG DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_DO\_VALUE\_MASK\_RD\_SET32

##### Return-Wert

[32 Bit] Wert der dem SET-Befehl übergeben wird

##### Programmierbeispiel

```
long value = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_TIMEOUT,
DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_RD_SET32, 0, 0);
printf("%0x\n", value);
//Der Wert der dem SET-Befehl übergeben wurde, wird ausgelesen und
dargestellt.
```



#### 5.7.1.9. DapiSpecialTimeoutDoValueMaskWRClr32

##### Beschreibung

Dieser Befehl bestimmt die Ausgänge, die bei einem Timeout ausgeschaltet werden sollen.

##### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, ch, par2);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_DO\_VALUE\_MASK\_WR\_CLR32

ch = Gibt die Nummer des Ausgangs an, ab dem geschrieben werden soll (0, 32, 64, ..)

par2 = [32 Bit] Gibt die Ausgänge an, welche bei einem Timeout deaktiviert werden sollen

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_WR_CLR32, 0, 0xff);  
//Die ersten 8 Relais werden im Timeout Fall ausgeschaltet.
```

#### 5.7.1.10. DapiSpecialTimeoutDoValueMaskRDClr32

##### Beschreibung

Dieser Befehl dient dem Auslesen der übergebenen Werte.

##### Definition

*ULONG DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_DO\_VALUE\_MASK\_RD\_CLR32

##### Return-Wert

[32 Bit] Wert der dem CLR-Befehl übergeben wird

##### Programmierbeispiel

```
long value = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_TIMEOUT,
DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_RD_CLR32, 0, 0);
printf("%0x\n", value);
//Der Wert der dem CLR-Befehl übergeben wurde, wird ausgelesen und
dargestellt.
```

#### 5.7.1.11. DapiSpecialTimeoutDoValueLoadDefault

##### Beschreibung

Setzt die SET- und CLR-Werte auf den Default-Wert zurück.

(SET-Wert = 0, CLR-Wert = FFFFFFFF)

##### Definition

*DapiSpecialCommand(handle, DAPI\_SPECIAL\_CMD\_TIMEOUT, cmd, 0, 0);*

##### Parameter

cmd = DAPI\_SPECIAL\_TIMEOUT\_DO\_VALUE\_LOAD\_DEFAULT

##### Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_TIMEOUT,  
DAPI_SPECIAL_TIMEOUT_DO_VALUE_LOAD_DEFAULT, 0, 0);  
//SET- und CLR-Werte werden auf den Default-Wert gesetzt.
```

## 5.8. Testfunktionen

### 5.8.1. DapiPing

#### Beschreibung

Dieser Befehl prüft die Verbindung zu einem geöffneten Modul.

#### Definition

*ULONG DapiPing(ULONG handle, ULONG value);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

value=Übergebener Testwert, im Wertebereich von 0-255 (8-Bit), an das Modul

#### Return-Wert

Hier muß der mit "value" übergebene Testwert zurückkommen

## 5.9. Register Schreib-Befehle

### 5.9.1. DapiWriteByte

#### Beschreibung

Dieser Befehl führt einen direkten Register Schreibbefehl auf das Modul aus.

#### Definition

```
void DapiWriteByte(ULONG handle, ULONG adress, ULONG value);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

value=Gibt den Datenwert an, der geschrieben wird (8 Bit)

#### Return-Wert

Keiner

#### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

### 5.9.2. DapiWriteWord

#### Beschreibung

Dieser Befehl führt einen direkten Register Schreibbefehl auf das Modul aus.

#### Definition

```
void DapiWriteWord(ULONG handle, ULONG adress, ULONG value);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

value=Gibt den Datenwert an, der geschrieben wird (16 Bit)

#### Return-Wert

Keiner

#### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

### 5.9.3. DapiWriteLong

#### Beschreibung

Dieser Befehl führt einen direkten Register Schreibbefehl auf das Modul aus.

#### Definition

```
void DapiWriteLong(ULONG handle, ULONG adress, ULONG value);
```

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

value=Gibt den Datenwert an, der geschrieben wird (32 Bit)

#### Return-Wert

Keiner

#### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

#### 5.9.4. DapiWriteLongLong

##### Beschreibung

Dieser Befehl führt einen direkten Register Schreibbefehl auf das Modul aus.

##### Definition

```
void DapiWriteLongLong(ULONG handle, ULONG adress, ULONGLONG value);
```

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

value=Gibt den Datenwert an, der geschrieben wird (64 Bit)

##### Return-Wert

Keiner

##### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.



## 5.10. Register Lese-Befehle

### 5.10.1. DapiReadByte

#### Beschreibung

Dieser Befehl führt einen direkten Register Lese-Befehl auf das Modul aus.

#### Definition

*ULONG DapiReadByte(ULONG handle, ULONG adress);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

#### Return-Wert

Inhalt des zu lesenden Registers (8 Bit)

#### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

### 5.10.2. DapiReadWord

#### Beschreibung

Dieser Befehl führt einen direkten Register Lese-Befehl auf das Modul aus.

#### Definition

*ULONG DapiReadWord(ULONG handle, ULONG adress);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

#### Return-Wert

Inhalt des zu lesenden Registers (16 Bit)

#### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

### 5.10.3. DapiReadLong

#### Beschreibung

Dieser Befehl führt einen direkten Register Lese-Befehl auf das Modul aus.

#### Definition

*ULONG DapiReadLong(ULONG handle, ULONG adress);*

#### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

#### Return-Wert

Inhalt des zu lesenden Registers (32 Bit)

#### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

#### Programmbeispiel

```
char v0, v1, v2, v3;
uint ver;
float fw_ver;

ver = (uint)DapiReadLong(handle, 0xffff4);

v3 = (char)((ver >> 24) & 0xff);
v2 = (char)((ver >> 16) & 0xff);
v1 = (char)((ver >> 8) & 0xff);
v0 = (char)((ver >> 0) & 0xff);

fw_ver = (((float)v0) - '0') * 10 + (((float)v1) - '0')
+ (((float)v2) - '0') / 10 + (((float)v3) - '0') / 100;
// Hier wird die Firmware-Version des Modules ausgelesen.
```

#### 5.10.4. DapiReadLongLong

##### Beschreibung

Dieser Befehl führt einen direkten Register Lese-Befehl auf das Modul aus.

##### Definition

*ULONGLONG DapiReadLongLong(ULONG handle, ULONG adress);*

##### Parameter

handle=Dies ist das Handle eines geöffneten Moduls

adress=Adresse, auf die zugegriffen werden soll

##### Return-Wert

Inhalt des zu lesenden Registers (64 Bit)

##### Bemerkung

Dies sollte nur von erfahrenen Programmieren benutzt werden. So kann auf alle zur Verfügung stehenden Register direkt zugegriffen werden.

## 5.11. Programmier-Beispiel

```
// *****
// *****
//
// (c) DEDITEC GmbH, 2009
//
// web: http://www.deditec.de
//
// mail: vertrieb@deditec.de
//
// dtapi_prog_beispiel_input_output.cpp
//
// *****
// *****
//
// Folgende Bibliotheken beim Linken mit einbinden: delib.lib
// Dies bitte in den Projekteinstellungen
// (Projekt/Einstellungen/Linker(Objekt-
// Bibliothek-Module) .. letzter Eintrag konfigurieren
#include <windows.h>
#include <stdio.h>
#include "conio.h"
#include "delib.h"
// *****
// *****

void main(void)
{
    unsigned long handle;
    unsigned long data;
    unsigned long anz;
    unsigned long i;
    unsigned long chan;
    // -----
    // USB-Modul öffnen
    handle = DapiOpenModule(USB_Interface8,0);
    printf("USB_Interface8 handle = %x\n", handle);
    if (handle==0)
    {
        // USB Modul wurde nicht gefunden
        printf("Modul konnte nicht geöffnet werden\n");
        printf("TASTE für weiter\n");
        getch();
        return;
    }
    // Zum Testen - ein Ping senden
    // -----
    printf("PING\n");
    anz=10;
    for(i=0;i!=anz;++i)
    {
        data=DapiPing(handle, i);
        if(i==data)
        {
            // OK
            printf(".");
        }
        else
        {

```

```

// No answer
printf("E");
}
}
printf("\n");

// -----
// Einen Wert auf die Ausgänge schreiben
data = 255;
DapiWriteByte(handle, 0, data);
printf("Schreibe auf Adresse=0 daten=0x%x\n", data);
// -----
// Einen Wert auf die Ausgänge schreiben
data = 255;
DapiWriteByte(handle, 1, data);
printf("Schreibe auf Adresse=0 daten=0x%x\n", data);
// -----
// Einen Wert auf die Ausgänge schreiben
data = 255;
DapiWriteByte(handle, 2, data);
printf("Schreibe auf Adresse=2 daten=0x%x\n", data);
// -----
// Einen Wert von den Eingängen lesen
data = (unsigned long) DapiReadByte(handle, 0);
printf("Gelesene Daten = 0x%x\n", data);
// -----
// Einen A/D Wert lesen
chan=11; // read chan. 11
data = DapiReadWord(handle, 0xff010000 + chan*2);
printf("Adress=%x, ret=%x volt=%f\n", chan, data, ((float) data) / 1024*5); //
Bei 5 Volt Ref
// -----
// Modul wieder schliessen
DapiCloseModule(handle);
printf("TASTE für weiter\n");
getch();
return ;
}

```

## 5.12. Delib Übersichtstabelle

Befehle	Verfügbar für
DAPI_SPECIAL_CMD_SET_DIR_DX_1	USB-MINI-TTL8
DAPI_SPECIAL_CMD_SET_DIR_DX_8	USB-MINI-TTL8 USB-TTL32 USB-TTL64 ETH-TTL64
DAPI_SPECIAL_CMD_GET_DIR_DX_1	wird nicht unterstützt
DAPI_SPECIAL_CMD_GET_DIR_DX_8	wird nicht unterstützt

Befehle	Verfügbar für	Geht nicht bei
DAPI_SPECIAL_CMD_TIMEOUT DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_WR_SET32 DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_RD_SET32 DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_WR_CLR32 DAPI_SPECIAL_TIMEOUT_DO_VALUE_MASK_RD_CLR32 DAPI_SPECIAL_TIMEOUT_DO_VALUE_LOAD_DEFAULT	ETH-TTL64 ETH-RELAIS8 USB-RELAIS8 RO-SERIE BS-SERIE NET-SERIE USB-TTL-64	USB-Mini-Stick
DAPI_SPECIAL_TIMEOUT_SET_VALUE_SE DAPI_SPECIAL_TIMEOUT_ACTIVATE DAPI_SPECIAL_TIMEOUT_DEACTIVATE DAPI_SPECIAL_TIMEOUT_GET_STATUS	alle Module	

Befehl	Starter USB*	Starter ETH**	RO Serie	BS Serie	NET Serie	Sonstiges
DAPI_SPECIAL_COUNTER_ LATCH_ALL			x			
DAPI_SPECIAL_COUNTER_ LATCH_ALL_WITH_RESET			x			
DapiDOSet1_WithTimer			x			
DAPI_SPECIAL_CMD_SW_FIFO DAPI_SPECIAL_SW_FIFO_INIT_ AND_CLEAR ... DAPI_SPECIAL_SW_FIFO_ IO_DEACTIVATE					x	
DAPI_SPECIAL_CMD_AD DAPI_SPECIAL_RO_AD_ FIFO_ACTIVATE ... DAPI_SPECIAL_RO_AD_ FIFO_INIT			x			

\*: USB-OPTOIN8, USB-Mini-Stick, USB-TTL-64

\*\* : ETH-TTL64, ETH-OPTOIN8, ETH-RELAIS8



Befehl	Starter USB*	Starter ETH**	RO Serie	BS Serie	NET Serie	Sonstiges
DAPI_SPECIAL_DI_FF_FILTER DAPI_SPECIAL_DI_FF_FILTER_ VALUE_SET DAPI_SPECIAL_DI_FF_FILTER_ VALUE_GET	5-255	1-255	1-255	1-255	1-255	
DAPI_SPECIAL_DI_FILTER DAPI_SPECIAL_DI_FILTER_ VALUE_SET DAPI_SPECIAL_DI_FILTER_ VALUE_GET	x	0, 1-254	0, 1-254	0, 1-254	0, 1-254	
DAPI_SPECIAL_CMD_GET_ INTERNAL_STATISTIC	x	x	x	x		

\*: USB-OPTOIN8, USB-Mini-Stick, USB-TTL-64

\*\* : ETH-TTL64, ETH-OPTOIN8, ETH-RELAIS8

Befehle	Verfügbar für
DAPI_SPECIAL_CMDEXT_CAN_WR_RUNTIME_ _VALUE DAPI_SPECIAL_CMDEXT_CAN_RD_RUNTIME_ VALUE	NET-CPU-PRO, BS-WEU, RO-ETH-LC

# Anhang

---



## **6. Anhang**

### **6.1. Kontakt / Support**

Wenn Sie Fragen zum Produkt haben oder Unterstützung bei der Inbetriebnahme brauchen, erreichen Sie uns unter folgenden Rufnummern:

#### **Support Software**

Tel. +49 (0) 22 32 / 50 40 8 – 20

#### **Support Hardware**

Tel. +49 (0) 22 32 / 50 40 8 – 30

#### **Support via E-mail**

[support@deditec.de](mailto:support@deditec.de)

### **6.2. Umwelt und Entsorgung**

Sie können das defekte oder veraltete Produkt am Ende seiner Lebensdauer wieder an uns zurück senden. Als Hersteller und Vertreiber von Elektronikbaugruppen übernehmen wir für Sie die fachgerechte Entsorgung nach den geltenden gesetzlichen Bestimmungen. Nutzen Sie hierfür am besten unser Rücksendeformular auf der Homepage:

[Rücksendeformular](#)

### 6.3. Revisionen

Rev 3.00	DEDITEC Design Update
Rev 2.03	Kapitel "Software" und "DELIB API Referenz" überarbeitet
Rev 2.02	Index hinzugefügt
Rev 2.01	Kapitel "Firmware Update" hinzugefügt, Kapitel "Software" überarbeitet
Rev 2.00	Erste DEDITEC Anleitung

## 6.4. Urheberrechte und Marken

Linux ist eine registrierte Marke von Linus Torvalds.

USB ist eine registrierte Marke von USB Implementers Forum Inc.

LabVIEW ist eine registrierte Marke von National Instruments.

Intel ist eine registrierte Marke von Intel Corporation.

AMD ist eine registrierte Marke von Advanced Micro Devices, Inc.

ProfiLab ist eine registrierte Marke von ABACOM Ingenieurbüro GbR.

ispVM System ist eine registrierte Marke von Lattice Semiconductor Corporation

Windows, Visual-C/C++, -C#, -Basic, -Basic.NET und Visual-Studio sind registrierte Marken von Microsoft Corporation.

Delphi ist eine registrierte Marke von Borland Software Corporation.

Java ist eine registrierte Marke von Oracle Corporation.