



DELIB
DEDITEC Treiber Bibliothek

2023 September

INDEX

1. Software Beschreibung	13
1.1. Benutzen unserer Produkte	14
1.1.1. Ansteuerung über unsere DELIB Treiberbibliothek	14
1.1.2. Ansteuerung über mitgelieferte Testprogramme	14
1.1.3. Ansteuerung auf Protokollebene	15
1.1.4. DELIB CLI (command-line interface) für Windows	16
1.1.4.1. Konfiguration des DELIB CLI	18
1.1.4.2. DELIB CLI Beispiele	19
1.1.5. Ansteuerung über grafische Anwendungen	23
1.1.5.1. LabVIEW	23
1.1.5.2. ProfiLab	23
1.1.5.3. Licht24 Pro	24
1.1.6. Einbinden der DELIB in Programmiersprachen	25
1.1.6.1. Einbinden der DELIB in Visual-C/C++	25
1.1.6.2. Einbinden der DELIB in Visual-C/C++ (Visual Studio 2015)	27
1.1.6.3. Einbinden der DELIB in Visual-C#	30
1.1.6.4. Einbinden der DELIB in Delphi	31
1.1.6.5. Einbinden der DELIB in Visual-Basic (VB)	32
1.1.6.6. Einbinden der DELIB in Visual-Basic.NET (VB.NET)	33
1.1.6.7. Einbinden der DELIB in MS-Office (VBA)	34
1.1.6.8. Einbinden der DELIB in LabVIEW	36
1.1.6.8.1. Einbinden der DELIB in LabVIEW	36
1.1.6.8.2. Verwendung der VIs in LabVIEW	45
1.1.6.8.3. Setzen der Modul-ID in LabVIEW	47
1.1.6.9. Einbinden der DELIB in Java	49
1.2. DELIB Treiberbibliothek	50
1.2.1. Übersicht	51
1.2.1.1. Unterstützte Programmiersprachen	52
1.2.1.2. Unterstützte Betriebssysteme	53
1.2.1.3. SDK-Kit für Programmierer	53
1.2.2. DELIB Setup	54

INDEX

1.2.3. DELIB Configuration Utility	60
1.2.3.1. Einführung	60
1.2.3.2. Neue Konfiguration erstellen oder vorhandene Konfiguration bearbeiten	61
1.2.3.2.1. Modul Konfiguration USB	62
1.2.3.2.1.1, Beispiel zur Konfiguration identischer USB-Module	64
1.2.3.2.2. Modul Konfiguration Ethernet	69
1.2.3.2.2.1, Automatische Suche	74
1.2.3.2.2.2, Verschlüsselung einrichten	78
1.2.3.2.2.1, Manuelle Konfiguration	78
1.2.3.2.2.2, Automatische Konfiguration	83
1.2.3.2.2.1, Authentifizierung	84
1.2.3.2.3. Modul Konfiguration CAN	88
1.2.3.2.4. Modul Konfiguration Seriell	91
1.2.3.3. Modul testen	93
1.2.3.4. Debug Optionen einstellen	97
1.2.4. Benutzung des Modulselectors	98
1.2.4.1. via USB	98
1.2.4.2. via Ethernet	99
1.2.4.3. via WiFi / WPS	101
1.2.4.4. Modul Info	102
1.2.5. DELIB Module Config	104
1.2.5.1. Modul Konfigurationen	104
1.2.5.1.1. Modul-Infoseite	105
1.2.5.1.2. Modul-Identifikation	106
1.2.5.1.3. LAN Netzwerkinformationen	107
1.2.5.1.4. LAN Netzwerkeinstellungen	108
1.2.5.1.5. WiFi Netzwerkinformationen	110
1.2.5.1.6. WiFi Netzwerkeinstellungen	112
1.2.5.1.7. WiFi WPS-Verbindung	114
1.2.5.1.8. NTP-Konfiguration	115
1.2.5.1.9. Serielle Konfiguration	117
1.2.5.1.10. I/O Kanal-Namen	119

INDEX

1.2.5.1.11. CAN Konfiguration	120
1.2.5.1.11.1, CAN Status	120
1.2.5.1.11.1, CAN Status Interface	120
1.2.5.1.11.2, CAN Statistik TX/RX	121
1.2.5.1.11.2, CAN Main	122
1.2.5.1.11.1, CAN Main Interface	122
1.2.5.1.11.2, CAN Main I/O Init	124
1.2.5.1.11.3, CAN TX/RX Modi	126
1.2.5.1.11.1, CAN TX Mode	126
1.2.5.1.11.2, CAN RX Mode	128
1.2.5.2. I/O-Test	129
1.2.5.2.1. Timeout Test-Funktion	129
1.2.5.2.2. Digital In	130
1.2.5.2.3. Digital Out	131
1.2.5.2.4. Analog In	133
1.2.5.2.5. Analog Out	134
1.2.6. DELIB Module Demo	136
1.2.6.1. Auswahl des Moduls	137
1.2.6.2. Allgemein	138
1.2.6.2.1. Module Info	140
1.2.6.3. Digital Input	141
1.2.6.4. Digital Output	142
1.2.6.5. Analog Input	143
1.2.6.6. Analog Output	144
1.2.7. CAN Configuration Utility	145
1.2.7.1. Auswahl des Moduls	147
1.2.7.2. Neue Konfiguration Erstellen, Laden, Speichern	149
1.2.7.3. Konfiguration auf das Modul übertragen	151
1.2.7.4. Statistiken vom Modul abfragen	152
1.2.7.5. Konfiguration	155
1.2.7.5.1. Modul Konfiguration	156
1.2.7.5.2. I/O Konfiguration	157
1.2.7.5.3. TX-Konfiguration	160

INDEX

1.2.7.5.3.1, Beispiel Interval	162
1.2.7.5.3.2, Beispiel Trigger	163
1.2.7.5.4. RX-Konfiguration	164
1.2.7.5.4.1, Beispiel RX-DA	166
1.2.7.5.4.2, Beispiel RX-DO	167
1.2.7.6. Aufbau der CAN-Pakete	168
1.2.7.6.1. Digitale Eingänge	168
1.2.7.6.2. Digitale Ausgänge	169
1.2.7.6.3. Digitale Eingangszähler (16-Bit)	170
1.2.7.6.4. Digitale Eingangszähler (48-Bit) - 32-Bit Paket	171
1.2.7.6.5. Digitale Eingangszähler (48-Bit) - 64-Bit Paket	172
1.2.7.6.6. Analoge Ein- / Ausgänge	173
1.2.7.6.6.1, Analoge Eingänge	173
1.2.7.6.6.2, Analoge Ausgänge	175
1.2.7.6.6.3, Beispiele	176
1.2.7.6.7. Temperatur Eingänge	179
1.2.7.6.8. Stepper	181
1.2.7.6.8.1, Command-Liste	181
1.2.7.6.8.2, Werte für par 1 zu Befehl SET_MOTORCHARACTERISTIC	184
1.2.7.6.8.3, Werte für par 1 zu Befehl GO_REFSWITCH	187
1.2.7.6.8.4, Beispiel	187
1.2.8. DT-Flasher	188
1.2.8.1. Über DEDITEC-Firmware	189
1.2.8.2. Auswahl des Moduls	189
1.2.8.3. Firmware Update durchführen	190
1.2.8.3.1. Flash-Files manuell aktualisieren	192
1.3. DELIB Sample Sources (Windows Programmbeispiele)	193
1.3.1. Installation DELIB Sample Sources	194
1.3.2. Benutzung der DELIB Sample Sources	198
1.3.2.1. Schritt 1 - Produktauswahl	199
1.3.2.2. Schritt 2 - Kategorieauswahl	200
1.3.2.3. Schritt 3 - Programmiersprachenauswahl	201
1.3.2.4. Schritt 4 - Quellcode	202

INDEX

1.4. DELIB für Linux	205
1.4.1. Verwenden der DELIB Treiberbibliothek für Linux	208
1.4.1.1. Delib USB-Sample in Linux	208
1.4.1.2. Delib ETH-Sample in Linux	211
1.4.2. DELIB CLI (command-line interface) für Linux	215
1.4.2.1. Konfiguration des DELIB CLI	218
1.4.2.2. DELIB CLI Beispiele	221
1.5. Weboberfläche	222
1.5.1. Konfiguration	224
1.5.1.1. Allgemein	224
1.5.1.2. Netzwerk Konfiguration	225
1.5.1.3. Benutzer Manager	226
1.5.1.4. Status	229
1.5.1.5. Sicherheit	230
1.5.2. Ein-/Ausgänge	232
1.5.2.1. Allgemein	232
1.5.2.2. Digitale Eingänge	233
1.5.2.3. Digitale Eingänge Zähler	235
1.5.2.4. Digitale Ausgänge	237
1.5.2.5. Analoge Eingänge	239
1.5.2.6. Analoge Ausgänge	240
1.5.2.7. Konfiguration	241
<u>2. DELIB API Referenz</u>	244
2.1. Verfügbare DEDITEC Modul IDs	245
2.2. Verwaltungsfunktionen	248
2.2.1. DapiOpenModule	248
2.2.2. DapiCloseModule	249
2.2.3. DapiGetDELIBVersion	249
2.2.4. DapiSpecialCMDGetModuleConfig	250
2.2.5. DapiOpenModuleEx	253
2.2.6. DapiScanAllModulesAvailable	255

INDEX

2.3. Fehlerbehandlung	256
2.3.1. DapiGetLastError	256
2.3.2. DapiGetLastErrorText	257
2.3.3. DapiClearLastError	258
2.3.4. DapiGetLastErrorByHandle	259
2.3.5. DapiClearLastErrorByHandle	260
2.4. Digitale Eingänge lesen	261
2.4.1. DapiDIGet1	261
2.4.2. DapiDIGet8	261
2.4.3. DapiDIGet16	262
2.4.4. DapiDIGet32	263
2.4.5. DapiDIGet64	264
2.4.6. DapiDIGetFF32	265
2.4.7. DapiDIGetCounter	266
2.4.8. DapiSpecialCounterLatchAll	267
2.4.9. DapiSpecialCounterLatchAllWithReset	268
2.4.10. DapiSpecialDIFilterValueSet	269
2.4.11. DapiSpecialDIFilterValueGet	270
2.4.12. Dapi_Special_DI_FF_Filter_Value_Set	271
2.4.13. Dapi_Special_DI_FF_Filter_Value_Get	272
2.5. Digitale Ausgänge verwalten	273
2.5.1. DapiDOSet1	273
2.5.2. DapiDOSet8	273
2.5.3. DapiDOSet16	274
2.5.4. DapiDOSet32	275
2.5.5. DapiDOSet64	276
2.5.6. DapiDOSet1_WithTimer	277
2.5.7. DapiDOReadback32	278
2.5.8. DapiDOReadback64	278
2.5.9. DapiDOSetBit32	279
2.5.10. DapiDOClrBit32	280

INDEX

2.6. Digitale Zähler Funktionen für RO-Counter Module	281
2.6.1. DapiCnt48ModeSet	281
2.6.2. DapiCnt48ModeGet	287
2.6.3. DapiCnt48CounterGet32	288
2.6.4. DapiCnt48CounterGet48	289
2.6.5. DapiSpecialCNT48ResetSingle	290
2.6.6. DapiSpecialCNT48ResetGroup8	291
2.6.7. DapiSpecialCNT48LatchGroup8	292
2.6.8. DapiSpecialCNT48DGet1	293
2.7. PWM Funktionen	294
2.7.1. DapiPWMOutSet	294
2.7.2. DapiPWMOutReadback	296
2.7.3. DAPI_SPECIAL_PWM_FREQ_SET	297
2.7.4. DAPI_SPECIAL_PWM_FREQ_READBACK	298
2.8. Pulsgenerator Ausgänge verwalten	299
2.8.1. DapiPulseGenSet	299
2.9. A/D Wandler Funktionen	301
2.9.1. DapiADSetMode	301
2.9.2. DapiADGetMode	303
2.9.3. DapiADGet	303
2.9.4. DapiADGetVolt	304
2.9.5. DapiADGetmA	304
2.9.6. DapiSpecialADReadMultipleAD	305
2.10. D/A Ausgänge verwalten	307
2.10.1. DapiDASetMode	307
2.10.2. DapiDAGetMode	309
2.10.3. DapiDASet	310
2.10.4. DapiDASetVolt	311
2.10.5. DapiDASetmA	312
2.10.6. DapiSpecialCmd_DA	313
2.11. PT100 Funktionen	315

INDEX

2.11.1. DapiTempGet	315
2.12. TTL-Ein-/Ausgangs Richtungen setzen mit DapiSpecialCommand	316
2.12.1. DAPI_SPECIAL_CMD_SET_DIR_DX_1	316
2.12.2. DAPI_SPECIAL_CMD_SET_DIR_DX_8	318
2.12.3. DAPI_SPECIAL_CMD_GET_DIR_DX_8	319
2.13. Watchdog Funktionen	320
2.13.1. DapiWatchdogEnable	320
2.13.2. DapiWatchdogDisable	321
2.13.3. DapiWatchdogRetrigger	322
2.14. Schrittmotoren Funktionen	323
2.14.1. Befehle mit DapiStepperCommand	323
2.14.1.1. DAPI_STEPPER_CMD_GO_POSITION	323
2.14.1.2. DAPI_STEPPER_CMD_GO_POSITION_RELATIVE	324
2.14.1.3. DAPI_STEPPER_CMD_SET_POSITION	325
2.14.1.4. DAPI_STEPPER_CMD_SET_FREQUENCY	326
2.14.1.5. DAPI_STEPPER_CMD_GET_FREQUENCY	327
2.14.1.6. DAPI_STEPPER_CMD_SET_FREQUENCY_DIRECTLY	328
2.14.1.7. DAPI_STEPPER_CMD_STOP	329
2.14.1.8. DAPI_STEPPER_CMD_FULLSTOP	330
2.14.1.9. DAPI_STEPPER_CMD_DISABLE	331
2.14.1.10. DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC	332
2.14.1.11. DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC	339
2.14.1.12. DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_EEPROM_SAVE	348
2.14.1.13. DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_EEPROM_LOAD	349
2.14.1.14. DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_LOAD_DEFAULT	350
2.14.1.15. DAPI_STEPPER_CMD_GO_REFSWITCH	351

INDEX

2.14.1.16. DAPI_STEPPER_CMD_GET_CPU_TEMP	353
2.14.1.17. DAPI_STEPPER_CMD_GET_MOTOR_SUPPLY_VOLTAGE	354
2.14.2. Status abfragen mit DapiStepperGetStatus	355
2.14.2.1. DAPI_STEPPER_STATUS_GET_ACTIVITY	355
2.14.2.2. DAPI_STEPPER_STATUS_GET_POSITION	356
2.14.2.3. DAPI_STEPPER_STATUS_GET_SWITCH	357
2.14.3. DapiStepperCommandEx	358
2.15. CAN Runtime Funktionen	359
2.15.1. RunTimeVarWriteToModule	359
2.16. Software FIFO verwalten	375
2.16.1. DapiSpecialCMDSWFifo	375
2.16.1.1. DapiSpecialSWFifoSetSubmodule	376
2.16.1.2. DapiSpecialSWFifoGetSubmodule	376
2.16.1.3. DapiSpecialSWFifoActivate	378
2.16.1.4. DapiSpecialSWFifoDeactivate	378
2.16.1.5. DapiSpecialSWFifoGetActivity	379
2.16.1.6. DapiSpecialSWFifoIOActivate	380
2.16.1.7. DapiSpecialSWFifoIODeactivate	380
2.16.1.8. DapiSpecialSWFifoInitAndClear	381
2.16.1.9. DapiSpecialSWFifoSetChannel	382
2.16.1.10. DapiSpecialSWFifoGetChannel	383
2.16.1.11. DapiSpecialSWFifoSetFrequencyHz	384
2.16.1.12. DapiSpecialSWFifoGetFrequencyHz	385
2.16.1.13. DapiSpecialSWFifoGetBytesFree	386
2.16.1.14. DapiSpecialSWFifoGetBytesPerSample	387
2.16.1.15. DapiSpecialSWFifoSetMode	388
2.16.1.16. DapiSpecialSWFifoGetMode	389
2.16.1.17. DapiSpecialSWFifoGetStatus	390
2.16.1.18. DapiSpecialSWFifoGetInstanceType	392
2.16.2. DapiWriteFifo	394
2.16.3. DapiReadFifo	395

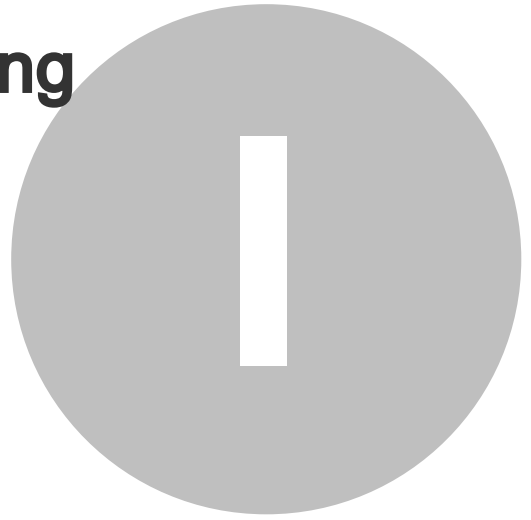
INDEX

2.17. Ausgabe-Timeout verwalten	399
2.17.1. DapiSpecialCMDTimeout	399
2.17.1.1. DapiSpecialTimeoutSetValueSec	402
2.17.1.2. DapiSpecialTimeoutActivate	403
2.17.1.3. DapiSpecialTimeoutActivateAutoReactivate	404
2.17.1.4. DapiSpecialTimeoutActivateSecureOutputs	405
2.17.1.5. DapiSpecialTimeoutDeactivate	406
2.17.1.6. DapiSpecialTimeoutGetStatus	407
2.17.1.7. DapiSpecialTimeoutDoValueMaskWRSet32	408
2.17.1.8. DapiSpecialTimeoutDoValueMaskRDSet32	409
2.17.1.9. DapiSpecialTimeoutDoValueMaskWRClr32	410
2.17.1.10. DapiSpecialTimeoutDoValueMaskRDClr32	411
2.17.1.11. DapiSpecialTimeoutDoValueLoadDefault	412
2.18. Testfunktionen	413
2.18.1. DapiPing	413
2.19. Register Schreib-Befehle	414
2.19.1. DapiWriteByte	414
2.19.2. DapiWriteWord	415
2.19.3. DapiWriteLong	416
2.19.4. DapiWriteLongLong	417
2.20. Register Lese-Befehle	418
2.20.1. DapiReadByte	418
2.20.2. DapiReadWord	419
2.20.3. DapiReadLong	420
2.20.4. DapiReadLongLong	421
2.21. Programmier-Beispiel	422
2.22. Delib Übersichtstabelle	424
<u>3. Verzeichnisstruktur der DELIB</u>	427
3.1. Include Verzeichnis	430
3.2. Library-Verzeichnis	430

INDEX

3.3. Library-Verzeichnis für Borland	430
3.4. Umgebungsvariablen	430
<u>4. Anhang</u>	431
4.1. Revisionen	432
4.2. Urheberrechte und Marken	434

Software Beschreibung



1. Software Beschreibung

1.1. Benutzen unserer Produkte

1.1.1. 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.

1.1.2. 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**.

1.1.3. 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

Online HTML-Manual:

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.

1.1.4. 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

1.1.4.1. 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.

1.1.4.2. 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

1.1.5. Ansteuerung über grafische Anwendungen

1.1.5.1. 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

1.1.5.2. 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

1.1.5.3. 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>

1.1.6. Einbinden der DELIB in Programmiersprachen

1.1.6.1. Einbinden der DELIB in Visual-C/C++

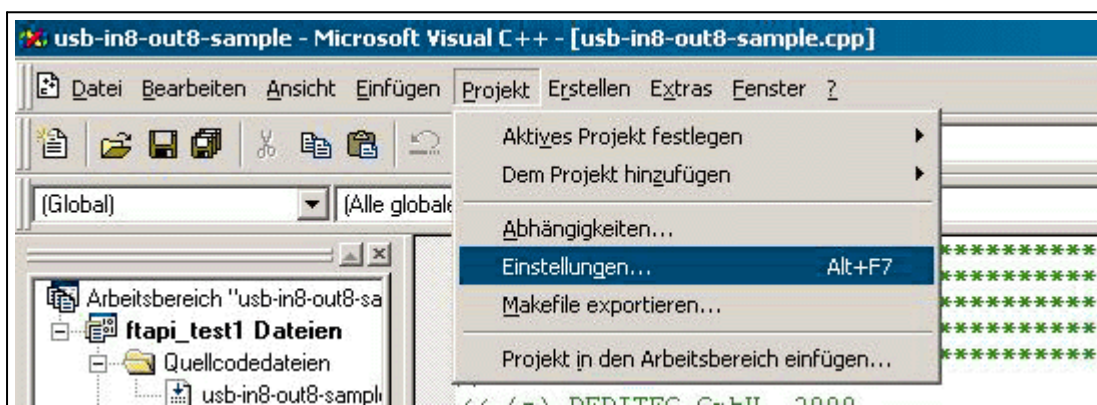
Zur Erleichterung für Verweise auf das DELIB-Include und das DELIB-Lib Verzeichnis werden bei installation der DELIB Umgebungsvariablen definiert.

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

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

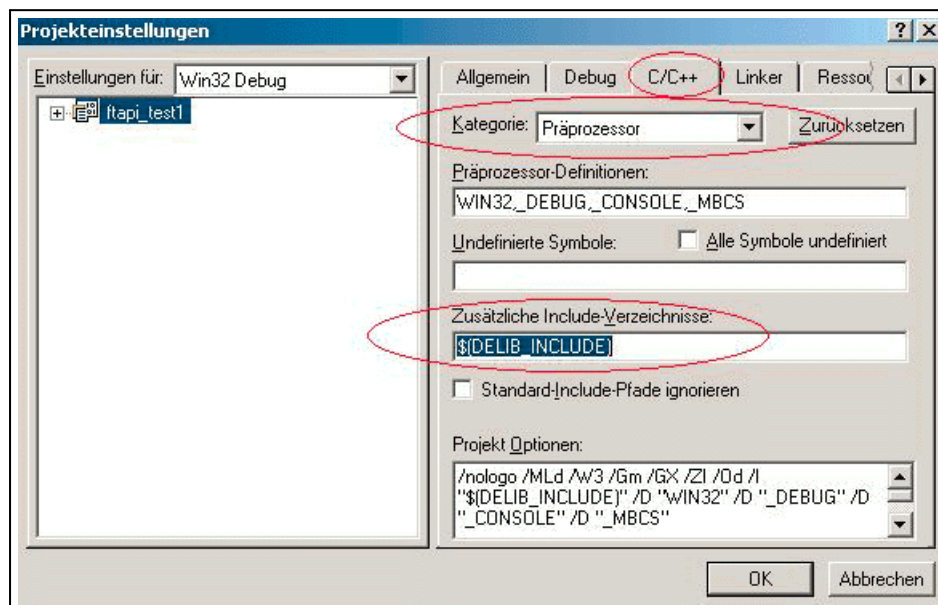
Diese werden im Folgenden in den Projekteinstellungen des Compilers eingetragen.

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



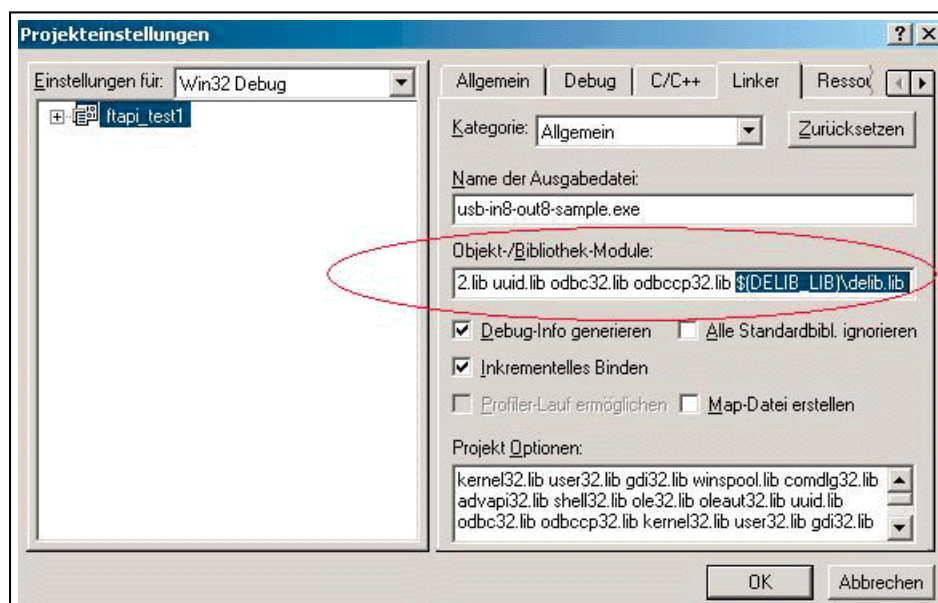
DELIB.H Eintrag in den Visual-C/C++ Projekt Einstellungen

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.



1.1.6.2. 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

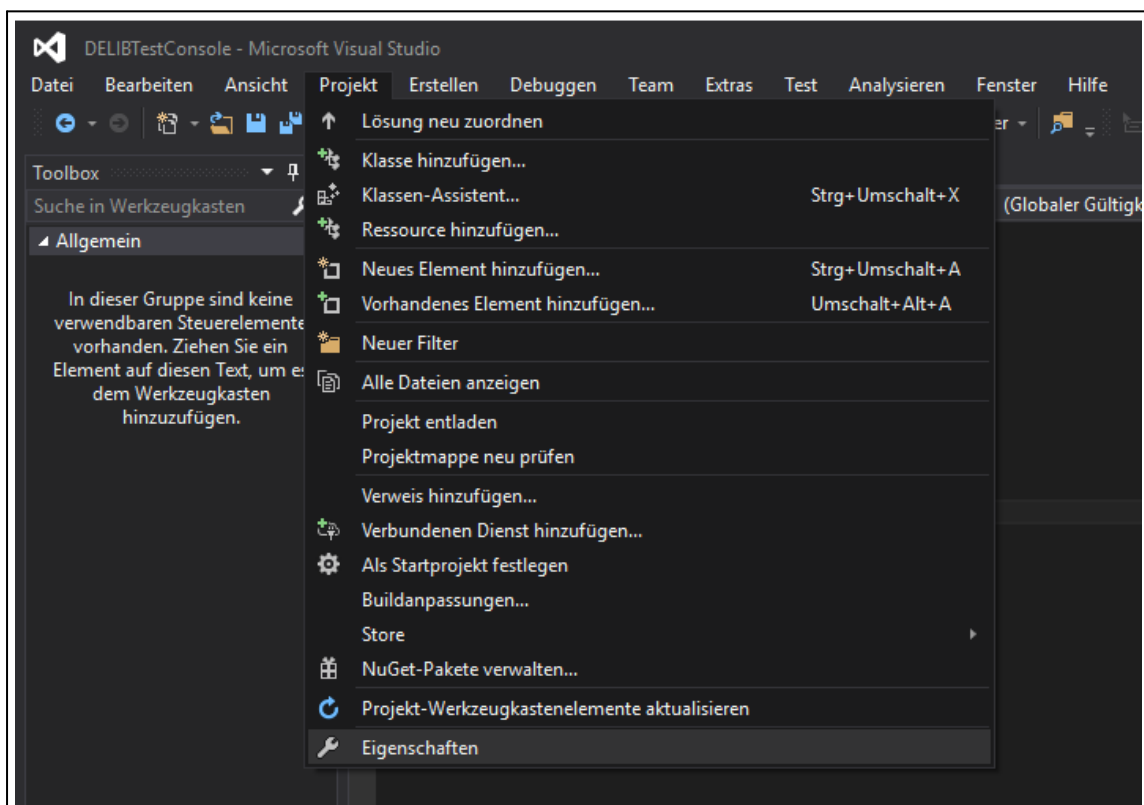
64 Bit DELIB Installation

DELIB64_LIB = C:\Programme\DEDITEC\DELIB64\lib

DELIB64_INCLUDE = C:\Programme\DEDITEC\DELIB64\include

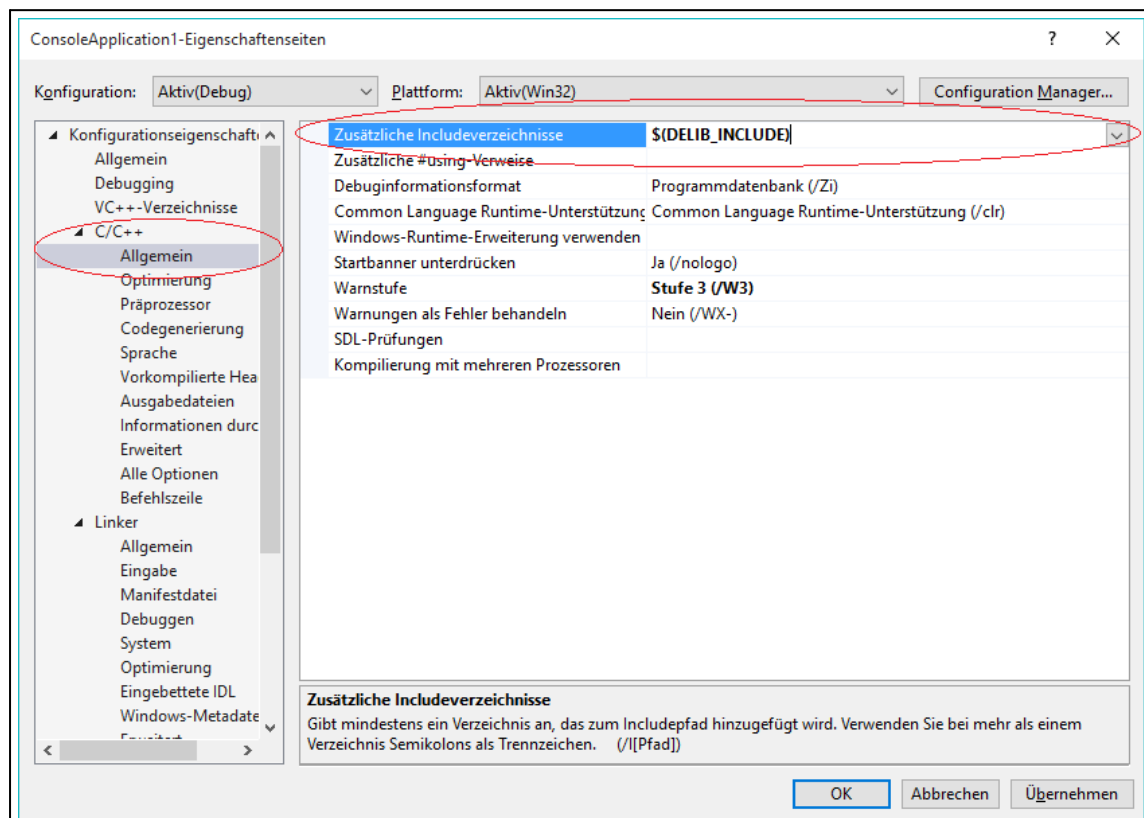
Diese werden im Folgenden in den Projekteinstellungen des Compilers eingetragen.

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



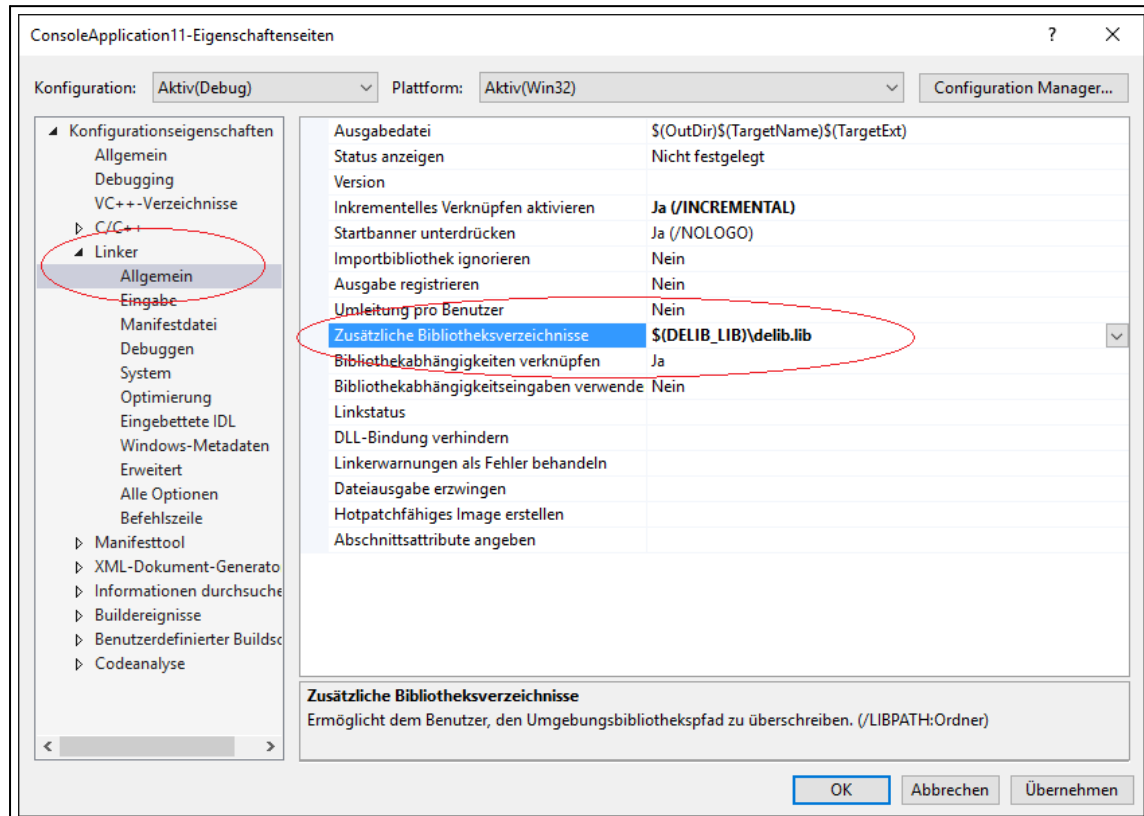
DELIB.H Eintrag in den Visual-C/C++ Projekt Einstellungen

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

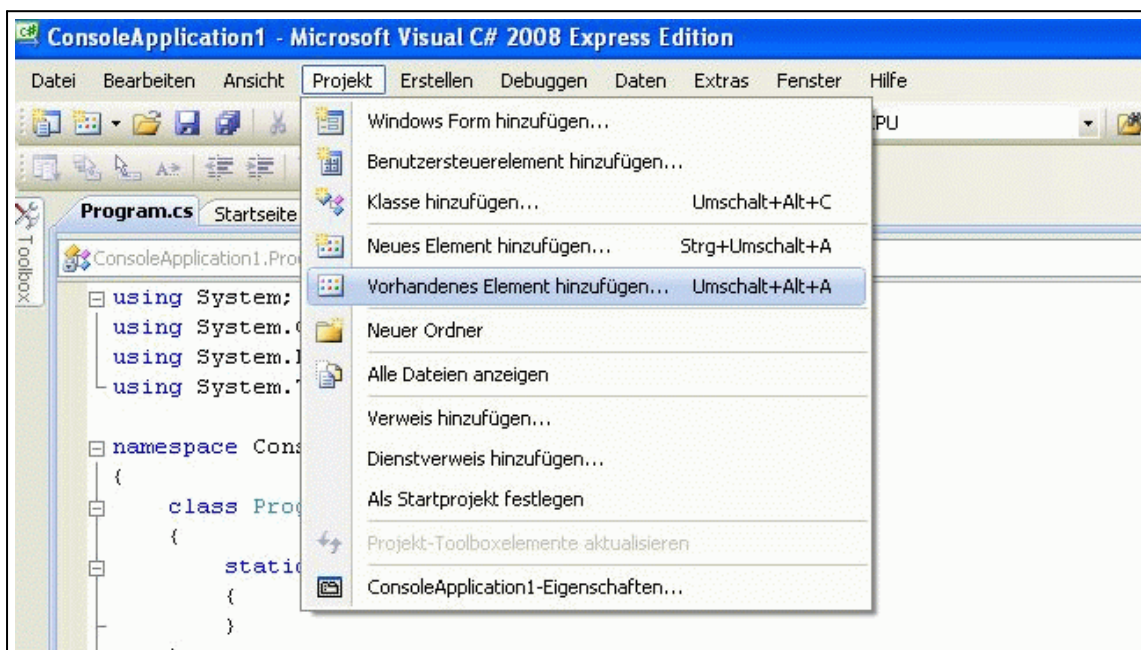
Unter dem Reiter "Linker" bei "Allgemein" "\$ (DELIB_LIB)\delib.lib" eintragen.



1.1.6.3. 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:

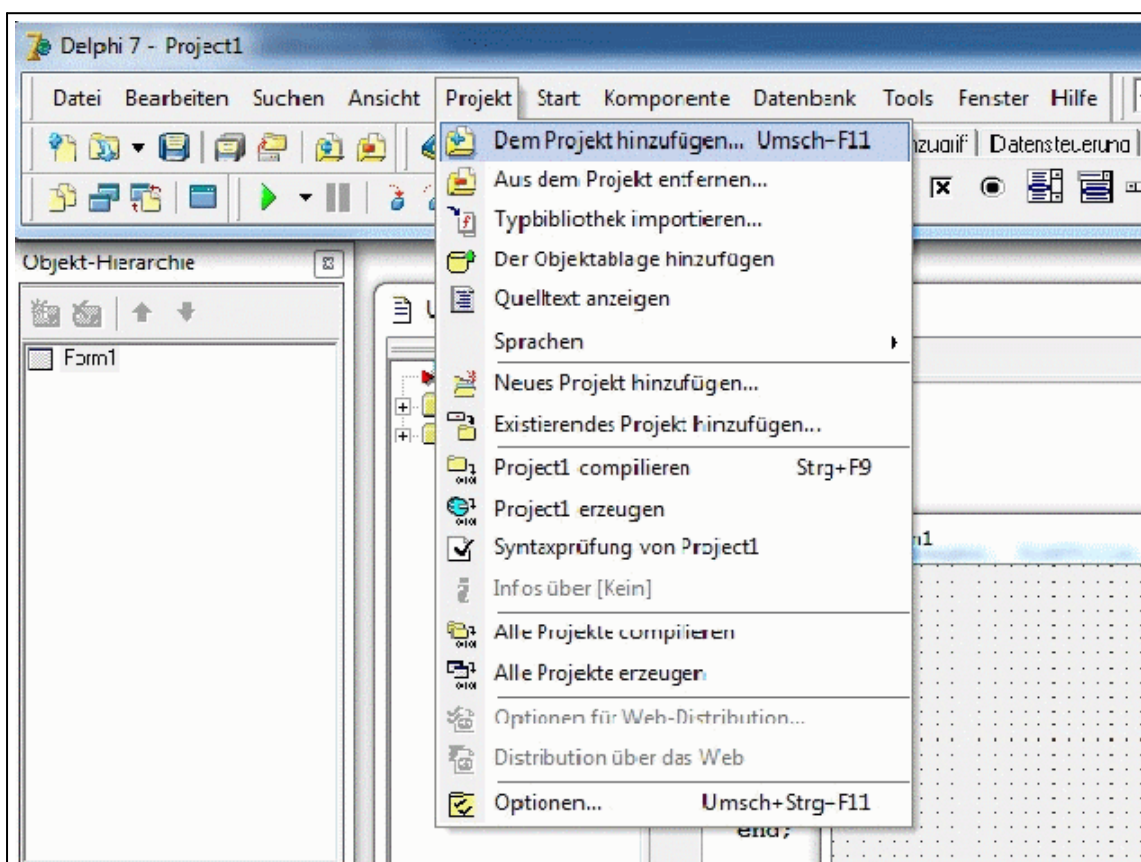
using DeLib;

1.1.6.4. Einbinden der DELIB in Delphi

Die benötigte Datei für Delphi befindet sich im Verzeichnis

C:\Programme\DEDITEC\DELIB\include.

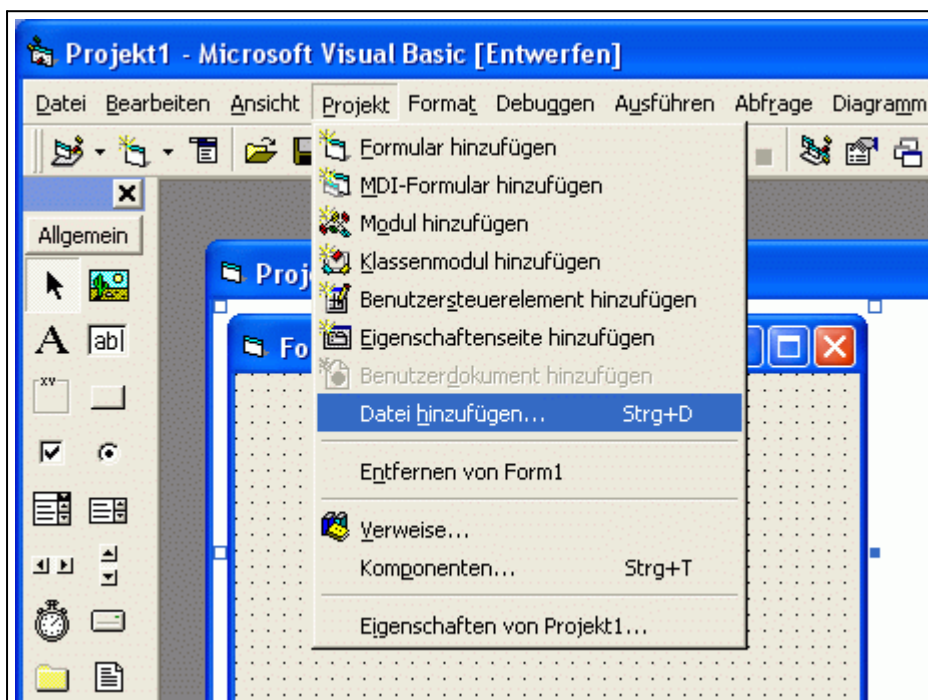
Delphi starten und über das Menue "Projekt → dem Projekt hinzufügen" im Verzeichnis C:\Programme\DEDITEC\DELIB\include\ die Datei delib.pas zum Importieren öffnen.



1.1.6.5. Einbinden der DELIB in Visual-Basic (VB)

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

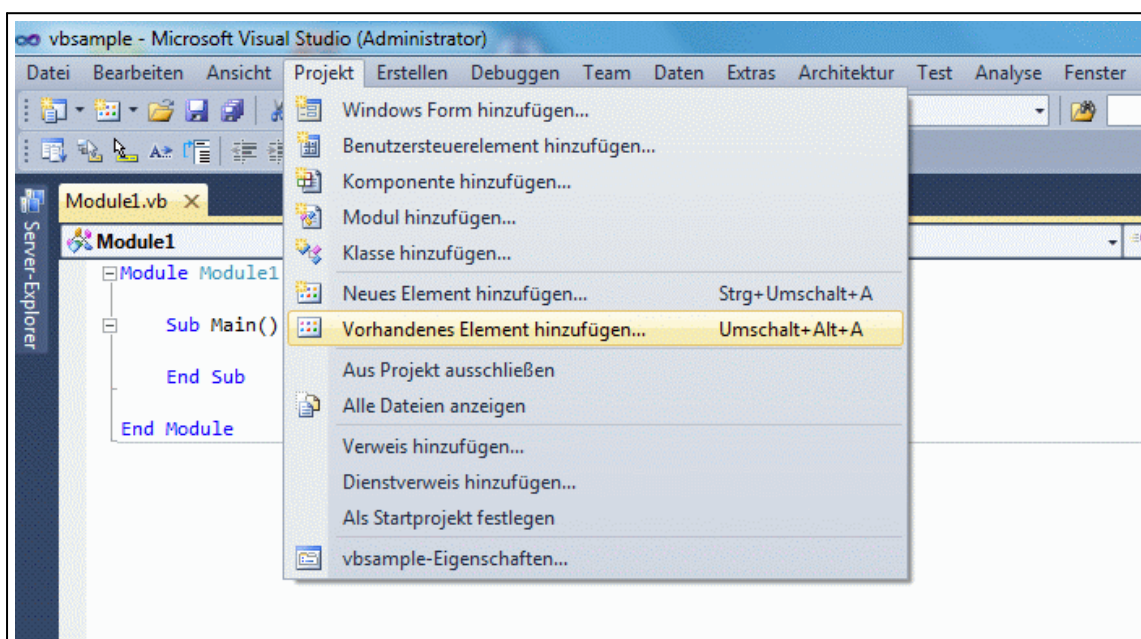
Visual Basic starten und über das Menue "Projekt → Datei hinzufügen..." im Verzeichnis C:\Programme\DEDITEC\DELIB\include\ die Datei delib.bas zum Importieren öffnen.



1.1.6.6. 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.

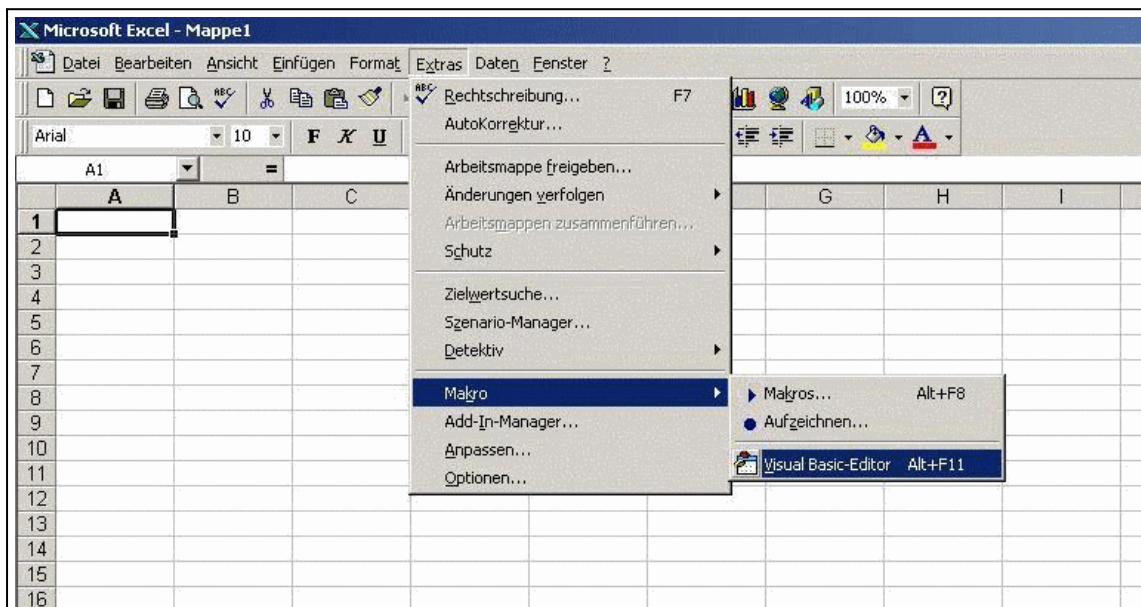


1.1.6.7. Einbinden der DELIB in MS-Office (VBA)

Die benötigte Datei für VBA befindet sich im Verzeichnis

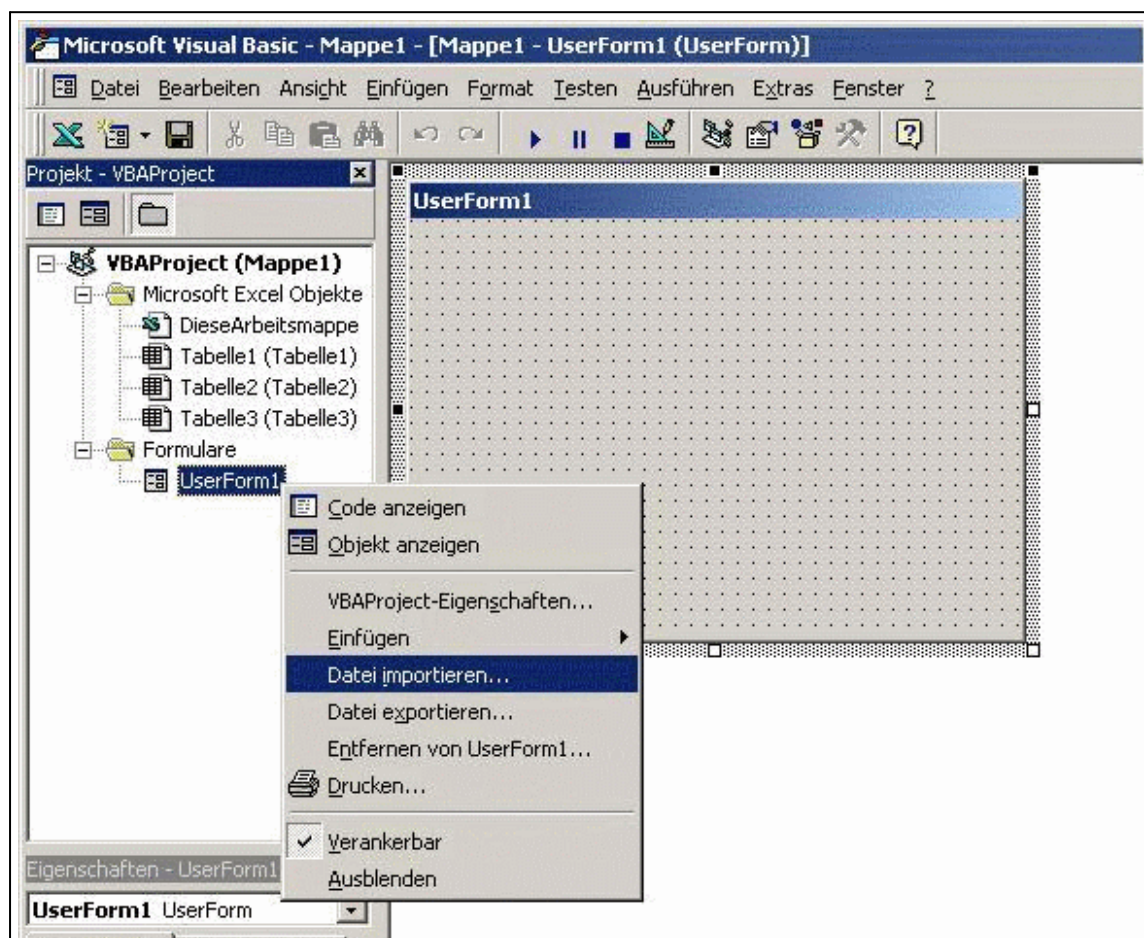
C:\Programme\DEDITEC\DELIB\include.

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.



1.1.6.8. Einbinden der DELIB in LabVIEW

1.1.6.8.1. Einbinden der DELIB in LabVIEW

Das LabVIEW-Beispielprogramm "Deditec_Modul_Control.vi" ist keine EXE-Datei und benötigt deshalb zur Ausführung die LabVIEW Entwicklungsumgebung.

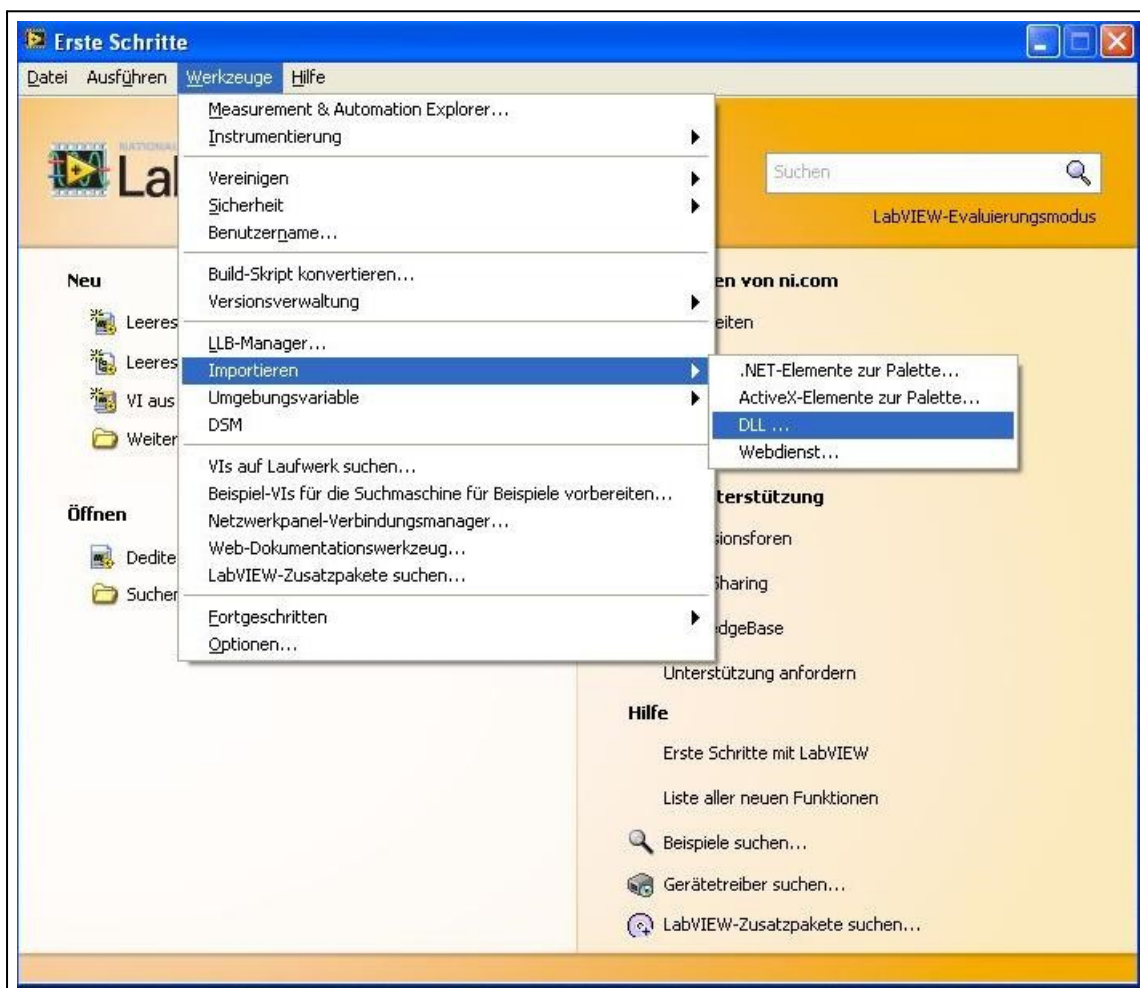
Beschreibung der Einbindung der "delib.dll" in LabVIEW Version 11

- 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

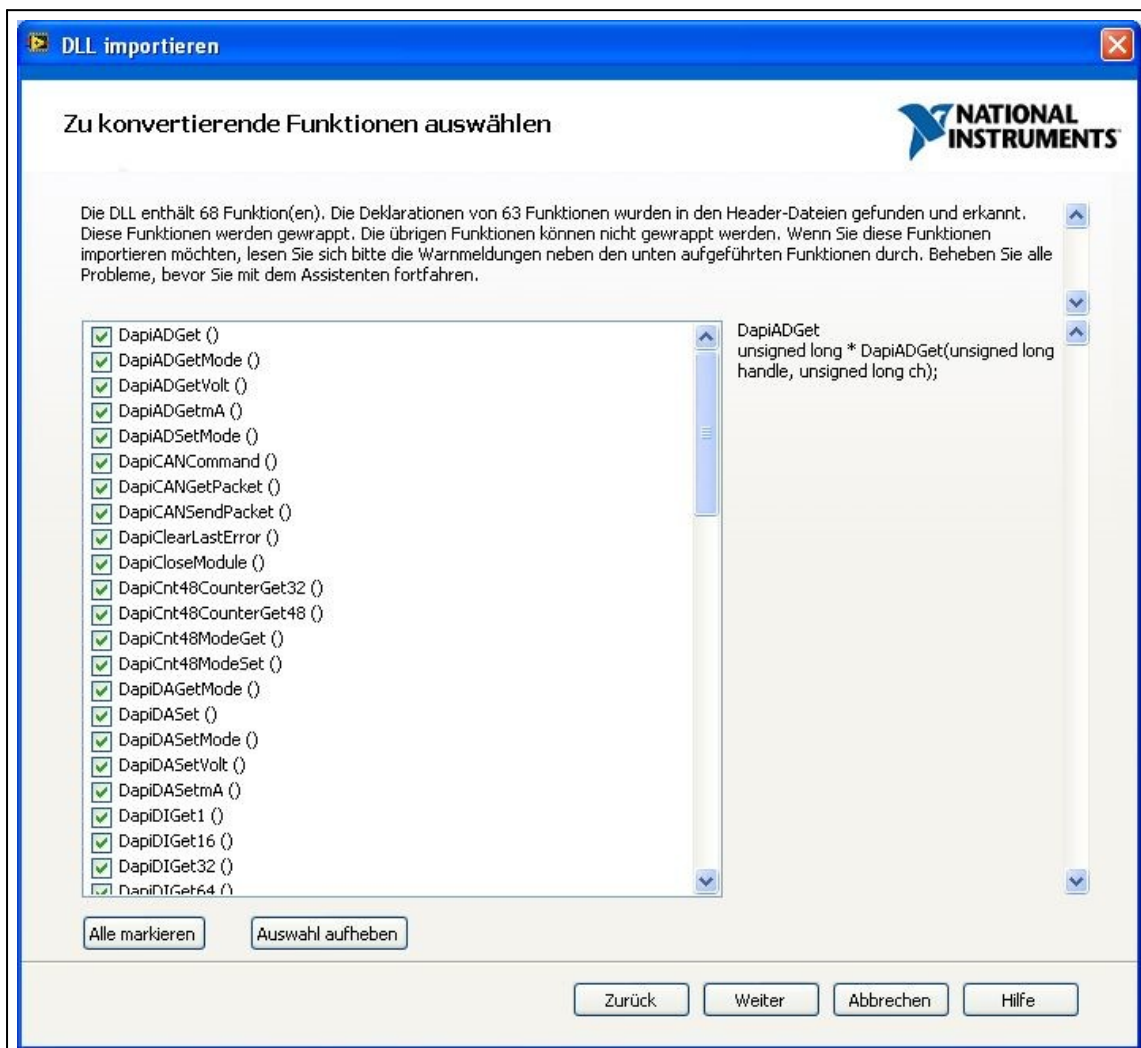
Bibliothek (*.dll)
C:\WINDOWS\system32\delib.dll

☐ DLL befindet sich nicht auf lokalem Computer

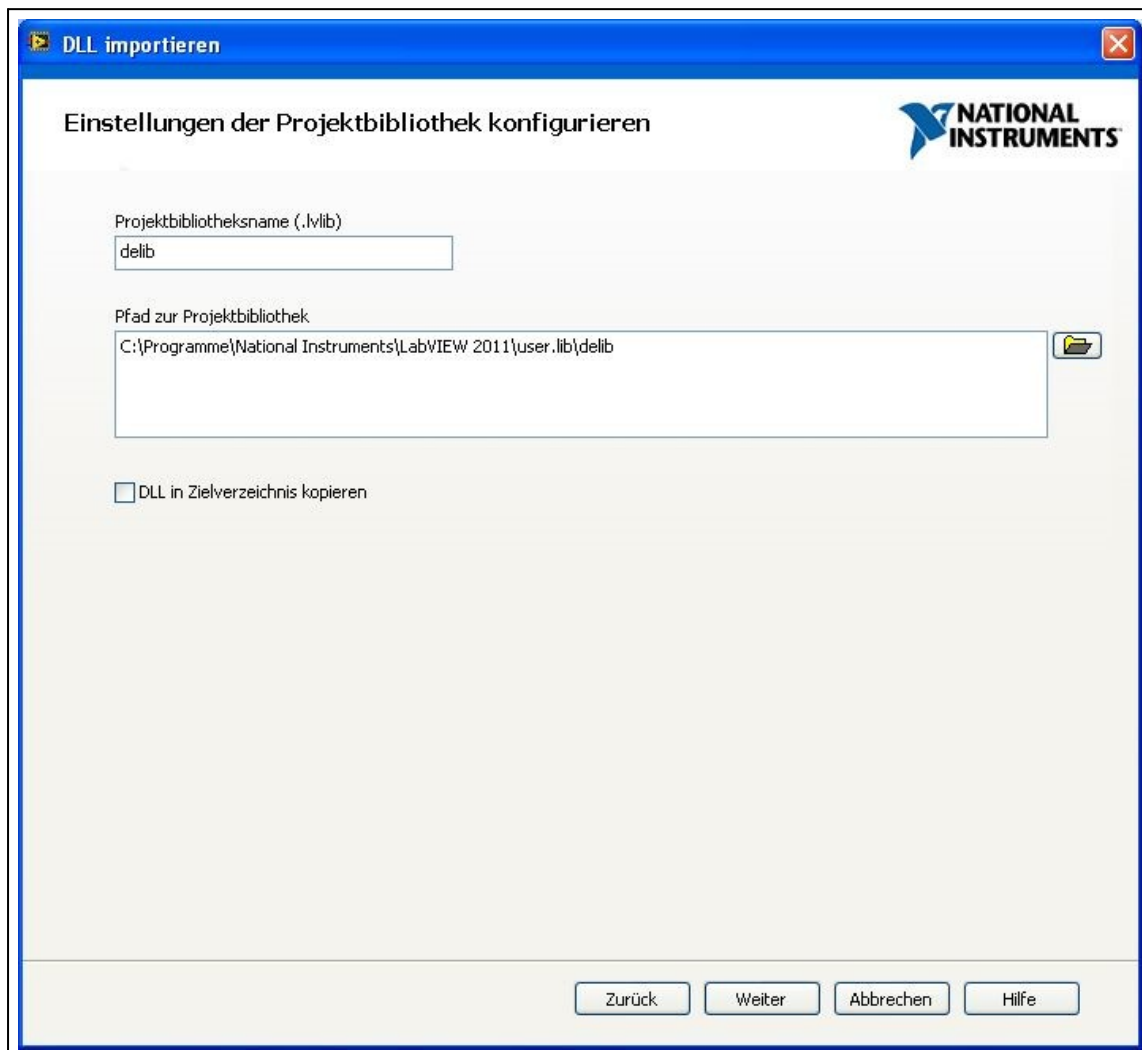
Header-Datei (.h)
C:\Programme\DEDITEC\DELIB\include\delib.h

Zurück Weiter Abbrechen Hilfe

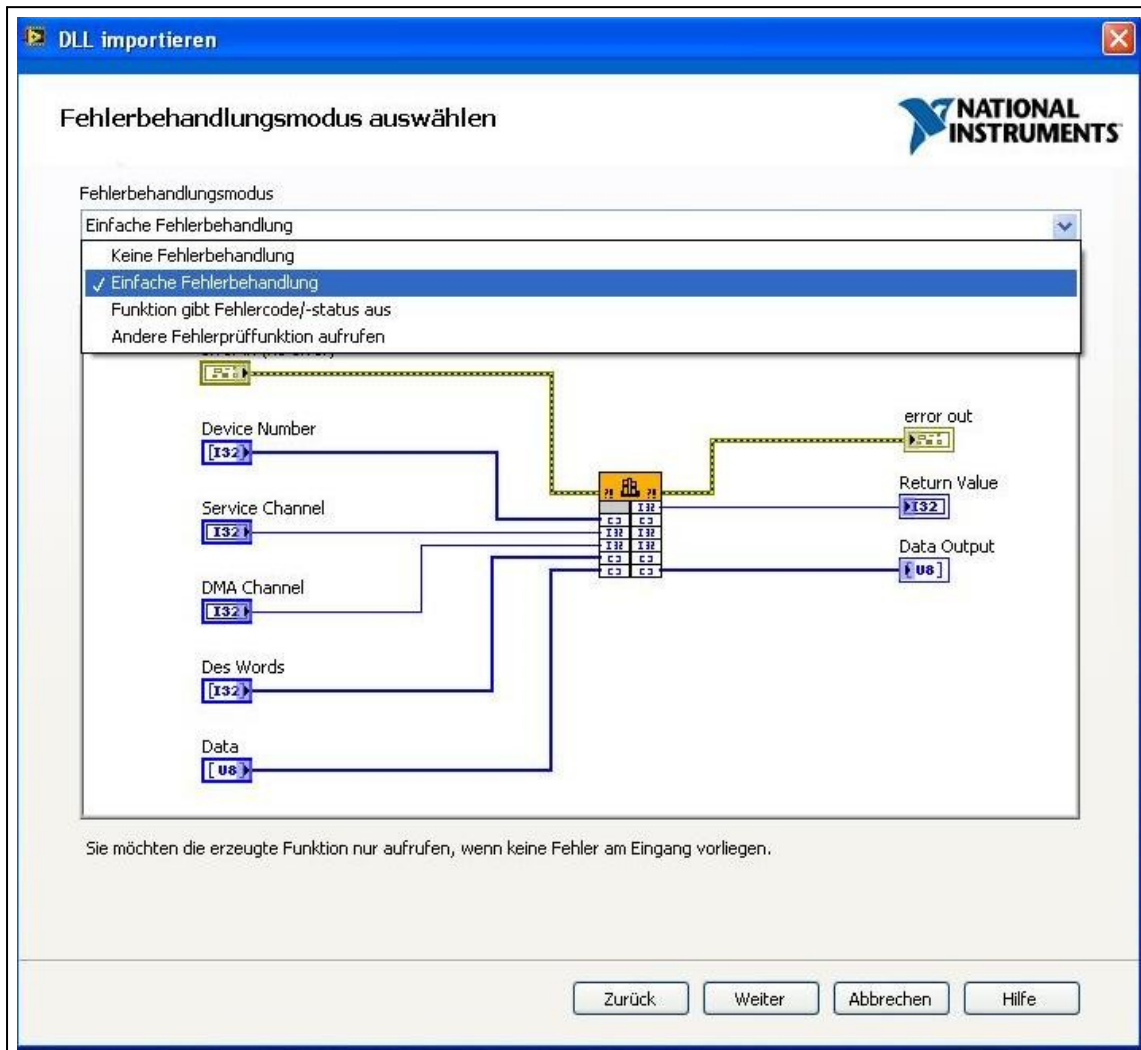
- Nochmals auf "Weiter" klicken um fortzufahren.
- Die Header-Datei wird nun analysiert. Anschließend fahren Sie im folgendem Fenster wieder mit "Weiter" fort.



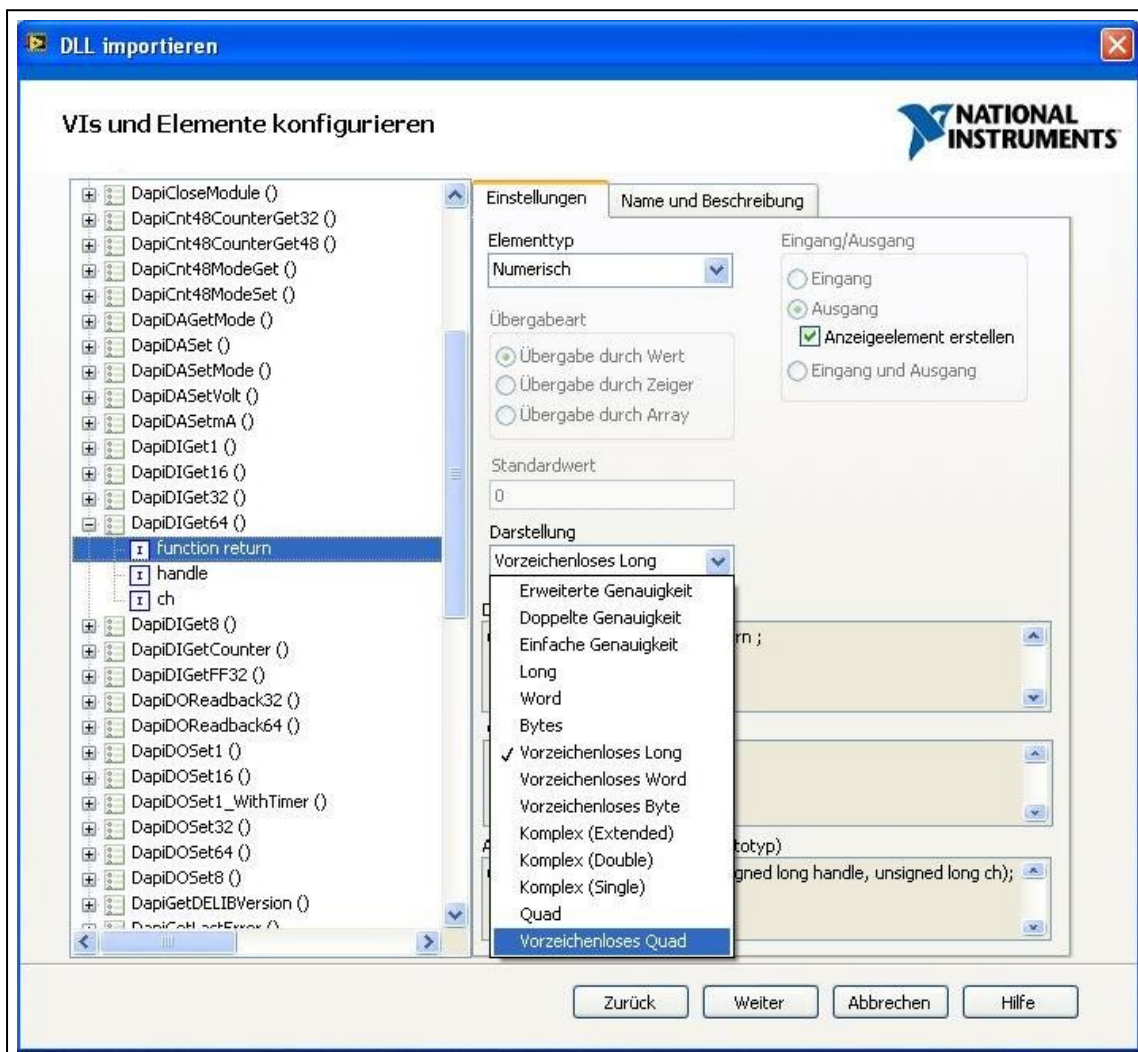
- Den weiteren Anweisungen folgen, bzw. die Konfiguration und den Speicherort für die VIs anpassen.



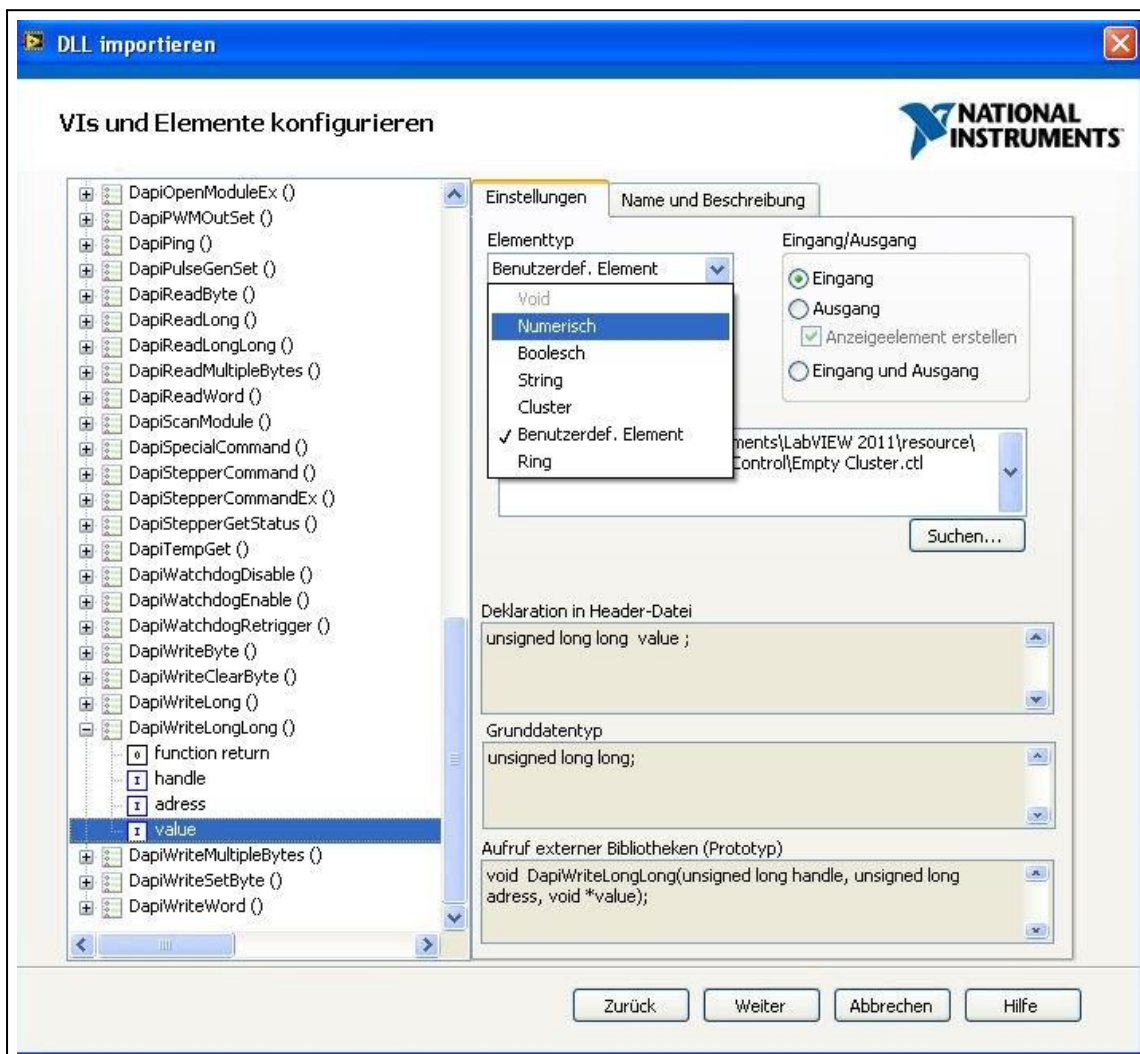
- Im folgendem Fenster wählen Sie im Drop-Down-Menü die Option "Einfache Fehlerbehandlung" aus und fahren mit "Weiter" fort.



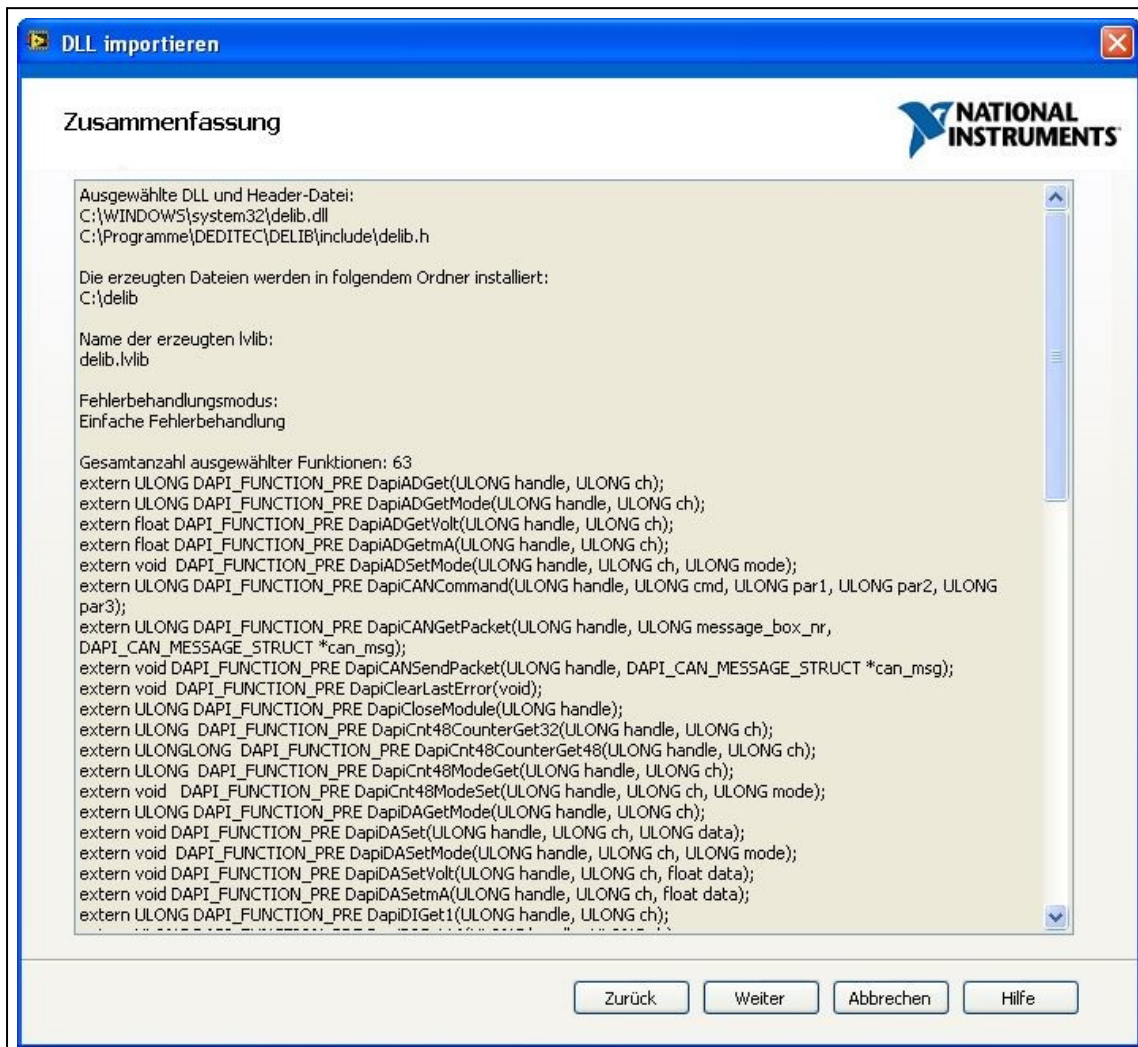
- Bei VIs die mit 64-Bit Werten arbeiten, muss die Darstellung von "Vorzeichenloses Long" in "Vorzeichenloses Quad" geändert werden.
- Folgende VIs müssen bearbeitet werden:
 - DapiCNT48CounterGet48 (function return)
 - DapiDIGet64 (function return)
 - DapiDOSet64 (data)
 - DapiDOReadBack64 (function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

1.1.6.8.2. 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);
```

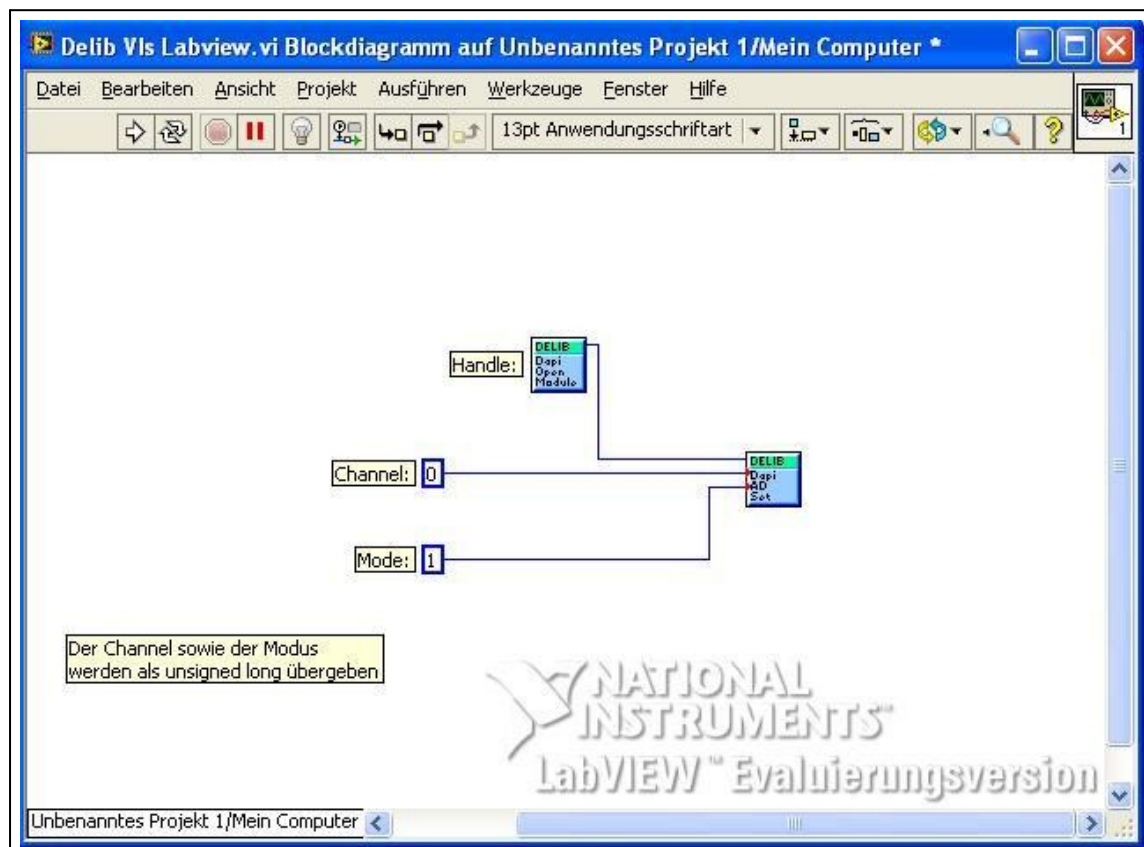
Alternativ kann man auch folgende Schreibweise verwenden:

```
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:



1.1.6.8.3. 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:

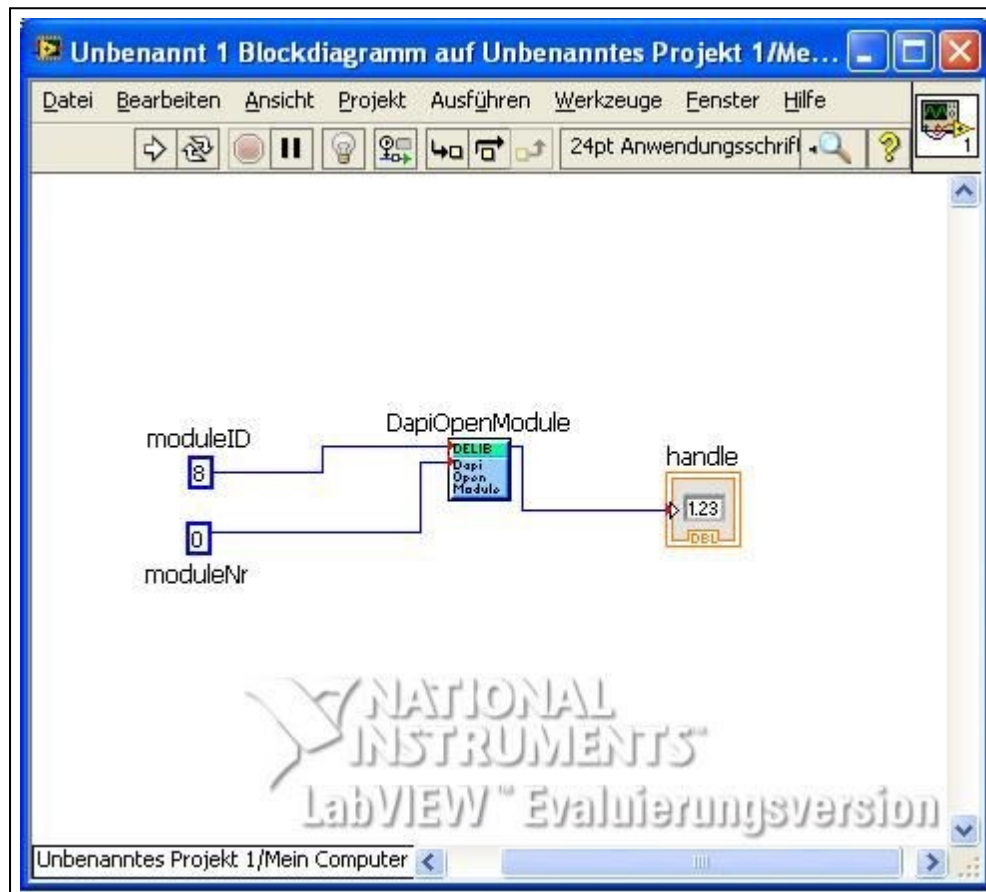
```
handle = DapiOpenModule(RO_ETH, 0); // öffnet ein RO-ETH-Modul mit Modul-Nr 0.
```

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:



1.1.6.9. Einbinden der DELIB in Java

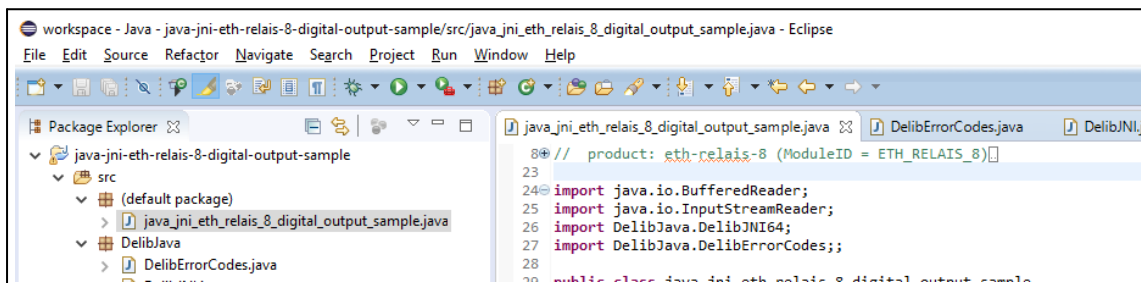
Die benötigten Dateien für Java befinden sich, je nach DELIB-Installation, in folgendem Verzeichnis

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.



1.2. DELIB Treiberbibliothek

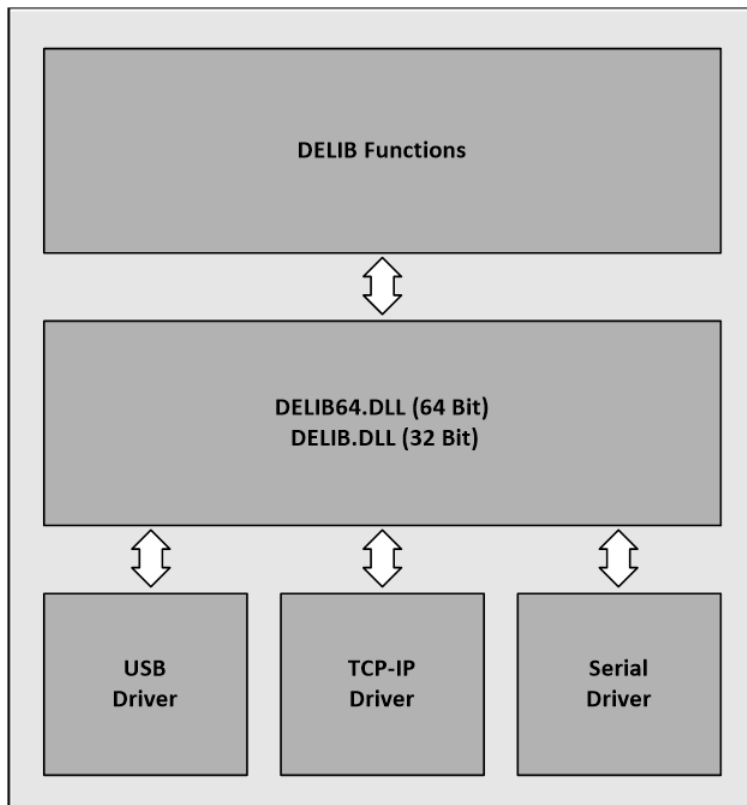
Die DELIB Treiberbibliothek enthält die DELIB-API und verschiedene Programme für den Konfigurationstest unserer Produkte.

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.

1.2.1. 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

1.2.1.1. 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.

1.2.1.2. 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

1.2.1.3. 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.

1.2.2. DELIB Setup

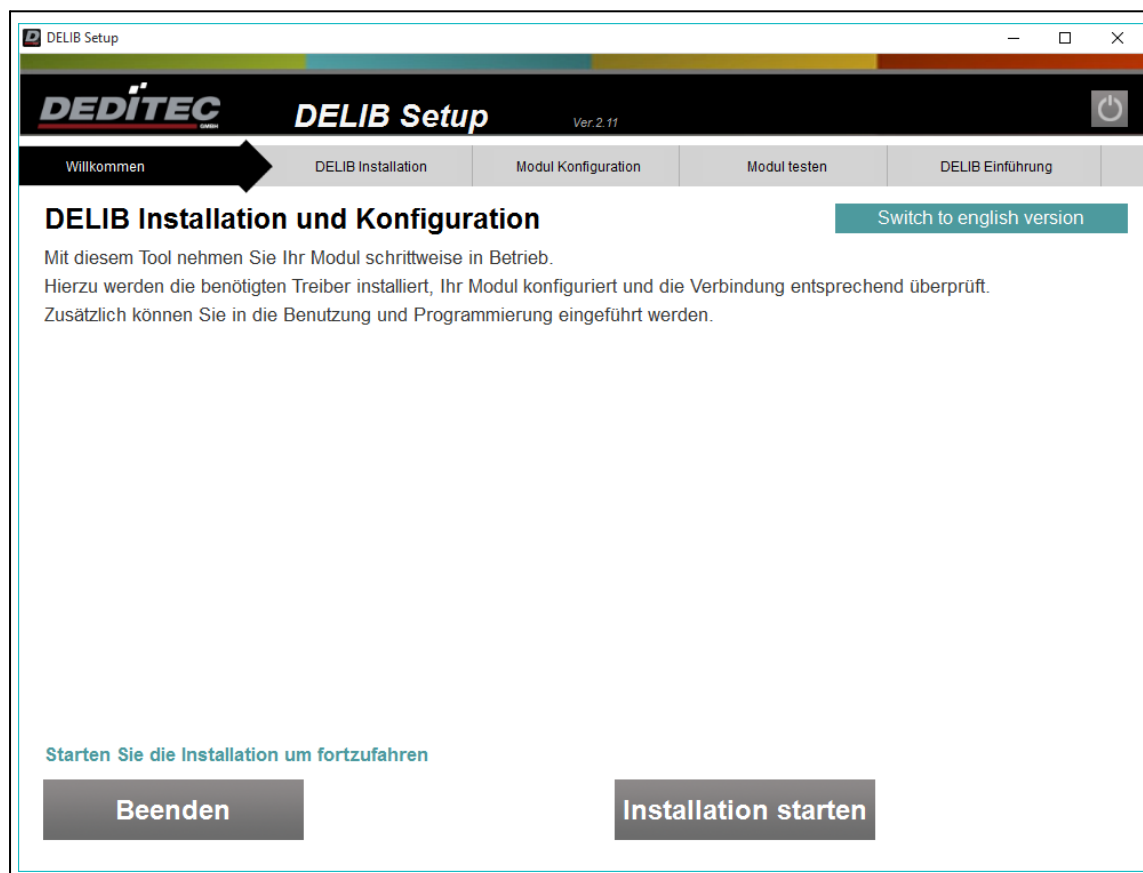
Das DELIB Setup führt Sie durch die Installation unserer DELIB Treiberbibliothek.

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.

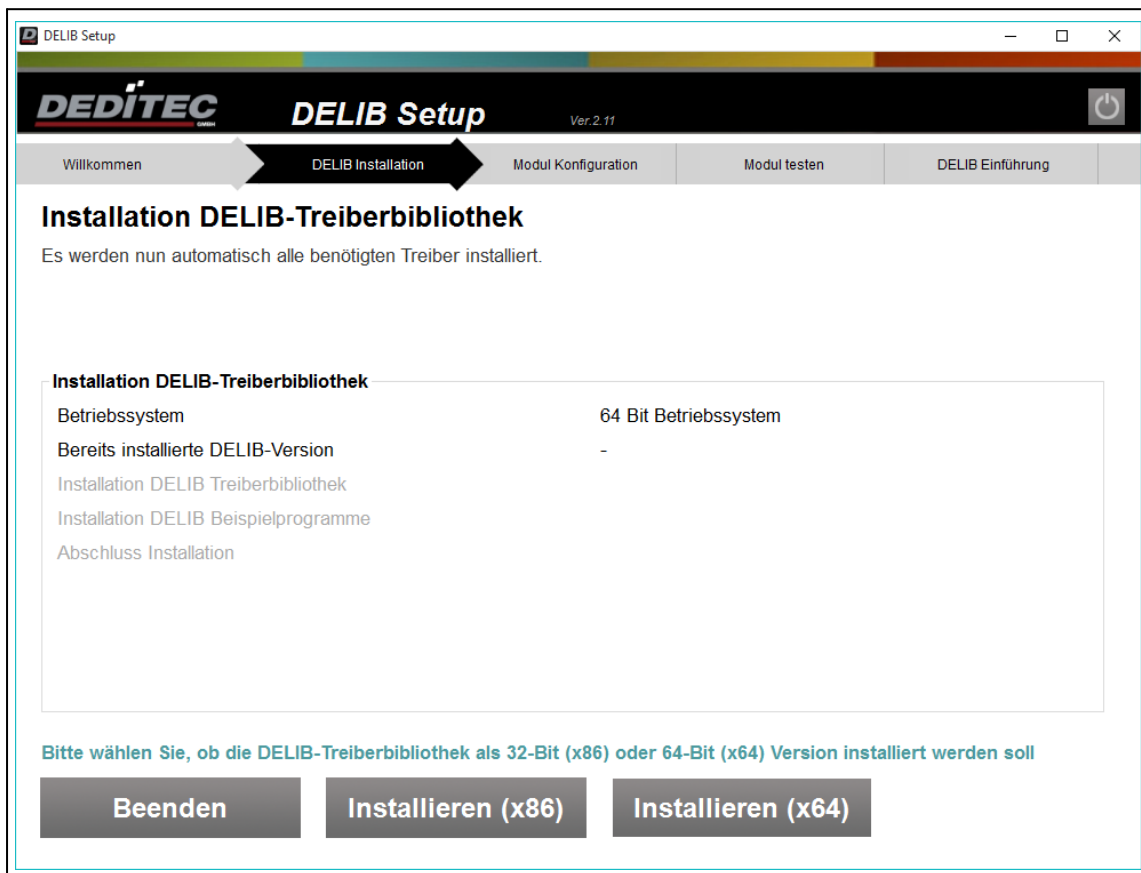
Link: <https://www.deditec.de/delib>

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

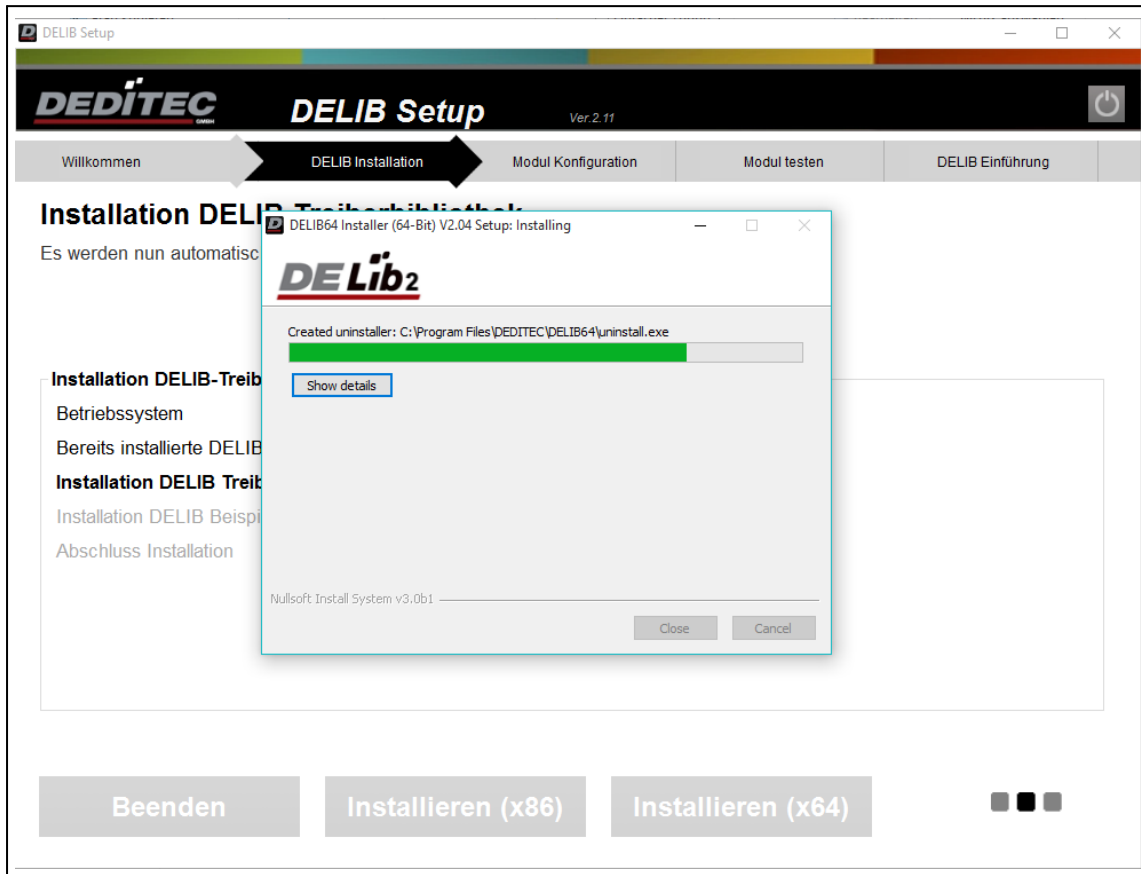


Das DELIB Setup prüft das Betriebssystem und ob bereits Versionen der DELIB Treiberbibliothek installiert sind.

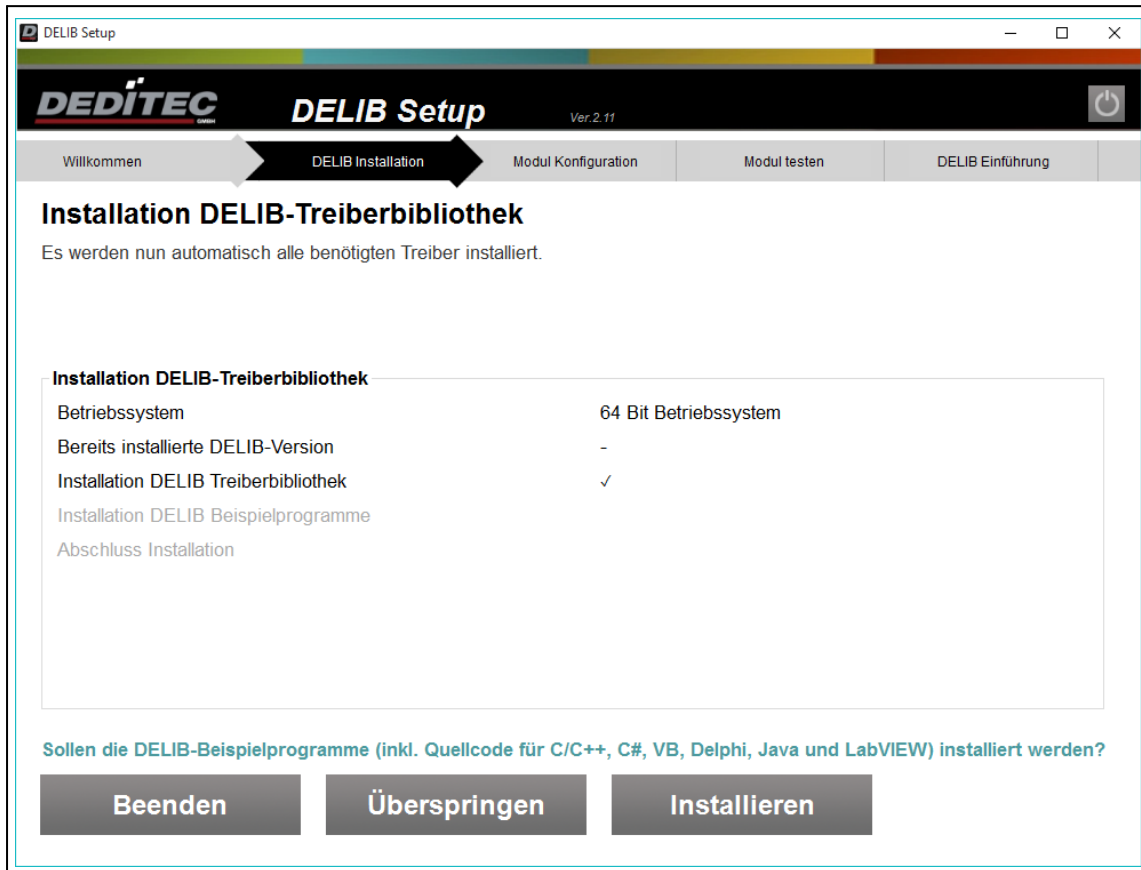
Anschließend können Sie wählen ob Sie die 32-Bit oder 64-Bit Version der DELIB Treiberbibliothek installieren möchten.



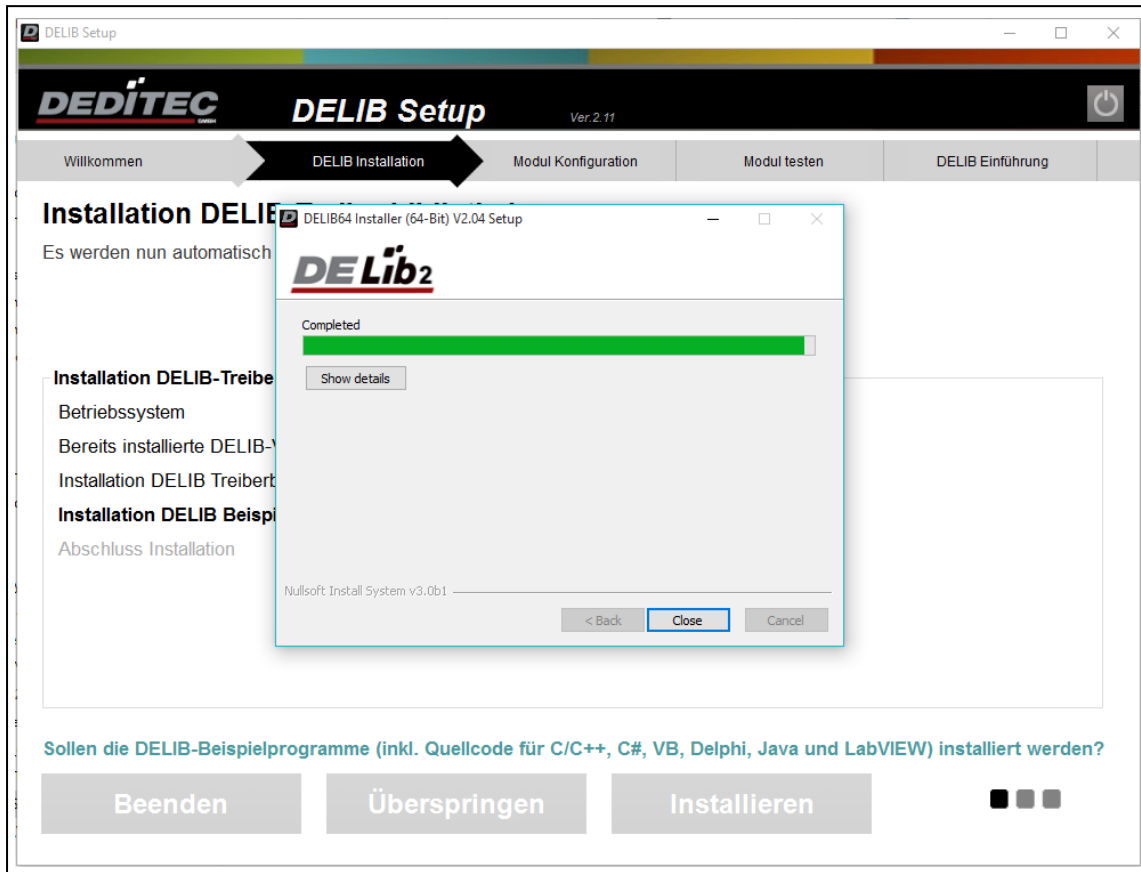
Installationsfortschritt der Treiberbibliothek.



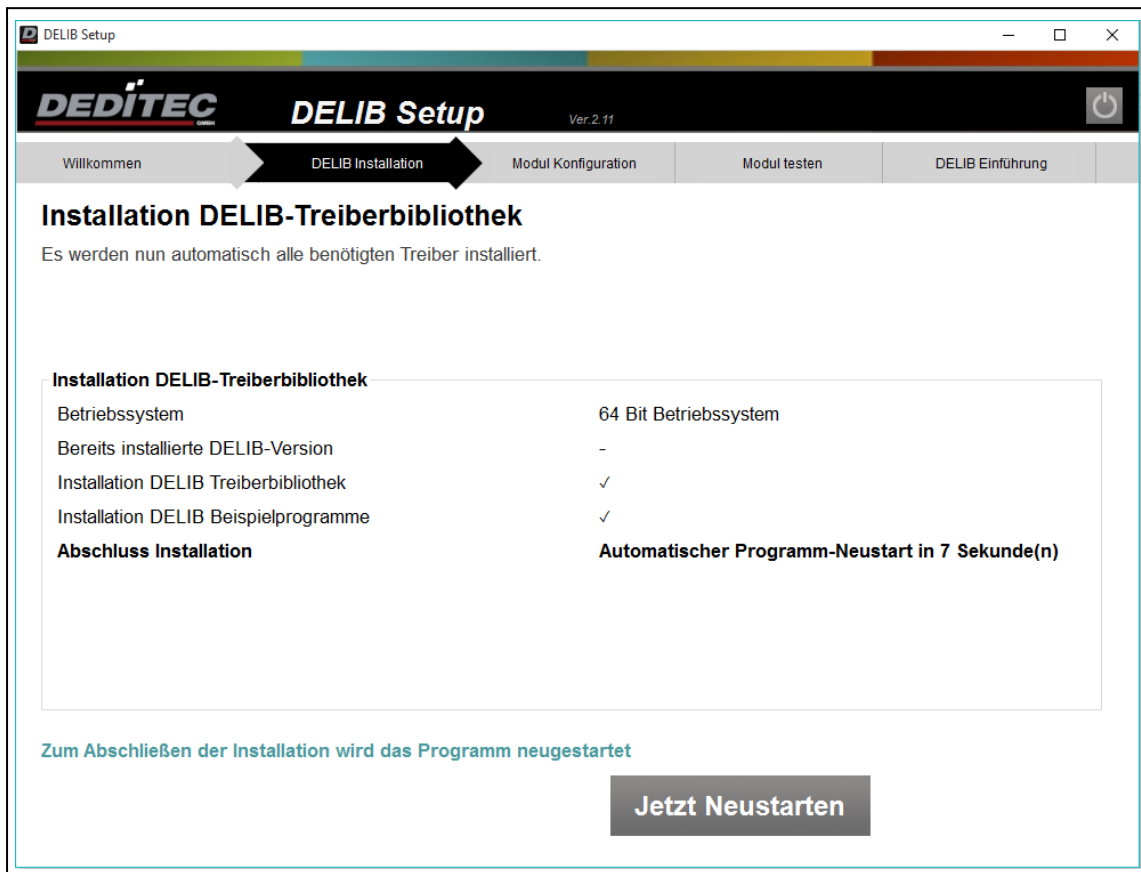
Zusätzlich können Sie wählen, ob Sie die DELIB Beispielprogramme installieren.



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.



1.2.3. DELIB Configuration Utility

1.2.3.1. 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.

1.2.3.2. 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 contains instructions in German and a 'Switch to english version' button. It is divided into two columns: 'Neues Modul' with a list of interfaces (USB, ETH, CAN, SER) and 'Konfigurierte Module' with a list of existing configurations (RO-ETH (1), RO-SER (0)). At the bottom, there are two buttons: 'Beenden' and 'Weiter'.

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
Schnittstelle <input type="radio"/> USB <input type="radio"/> ETH <input type="radio"/> CAN <input type="radio"/> SER	Module RO-ETH (1) RO-SER (0)

Wählen Sie die Schnittstelle oder ein bereits konfiguriertes Modul aus

Beenden **Weiter**

1.2.3.2.1. 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 subtitle '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, there is a section 'Gefundene USB-Module' with a table showing one module: 'USB-OPT/REL-8' with number '0'. To the right of the table are input fields for 'Aktuelle Modul-Nr.' (set to 0), 'Neue Modul-Nr.' (a dropdown menu set to 1), and 'Übertragungsversuche (für alle USB-Module)' (a dropdown menu set to 5). A teal button labeled 'Neue Modul-Nr. setzen' is positioned 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.

Neue Modul-Nr. setzen

Übertragungsversuche (für alle USB-Module)

Hauptmenü Test Fertig

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.

1.2.3.2.1.1. 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".

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The main header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver. 2.20', and 'DELIB 2.19 (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 'Modul Konfiguration' tab is active, displaying the title 'Modul Konfiguration' and a subtitle 'Mit dem DELIB Configuration Utility lassen sich DEDITEC Module in wenigen Schritten konfigurieren.' The interface is divided into two main sections. On the left, under 'Neues Modul', there is a 'Schnittstelle' (Interface) section with four radio button options: 'USB' (selected), 'ETH (Ethernet)', 'CAN', and 'SER (Seriell)'. On the right, under 'Konfigurierte Module', there is a 'Module' list box containing the following items: 'USB-OPT/REL-8 (0)', 'RO-CAN2 (0)', 'RO-ETH (0)', 'RO-ETH (1)', 'RO-ETH/LC (0)', 'RO-ETH/LC (3)', 'RO-SER (0)', and 'BS-ETH (0)'. At the bottom of the window, there is a blue instruction text: 'Wählen Sie die Schnittstelle oder ein bereits konfiguriertes Modul aus'. Below this text are two large buttons: 'Beenden' (End) on the left and 'Weiter' (Next) on the right.

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 main header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver. 2.20', and 'DELIB 2.19 (64 Bit)' with a language selector (UK flag) and help/power icons. A progress bar at the top indicates the current step: 'Modul Konfiguration' (highlighted with a black arrow), followed by 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. Below the progress bar, the section title is 'Konfiguration Module mit USB Schnittstelle'. The instructions state: 'Hier kann die Modul-Nr. des ausgewählten Moduls geändert werden. Die Modul-Nr. dient zur Unterscheidung mehrerer identischer USB-Module einer USB-Serie.' On the left, under 'Gefundene USB-Module', there is a table with two columns: 'Module' and 'Nr.'. The first row shows 'USB-OPT/REL-8' and '0'. Below the table is an 'Aktualisieren' button. On the right, there are input fields: 'Aktuelle Modul-Nr.' with a value of '0', 'Neue Modul-Nr.' with a dropdown menu showing '1', and 'Übertragungsversuche (für alle USB-Module)' with a dropdown menu showing '5'. A teal button labeled 'Neue Modul-Nr. setzen' is positioned between the 'Neue Modul-Nr.' and 'Übertragungsversuche' fields. At the bottom of the window, there are three large buttons: 'Hauptmenü', 'Test', and 'Weiter'.

Module	Nr.
USB-OPT/REL-8	0

Aktuelle Modul-Nr. 0

Neue Modul-Nr. 1

Übertragungsversuche (für alle USB-Module) 5

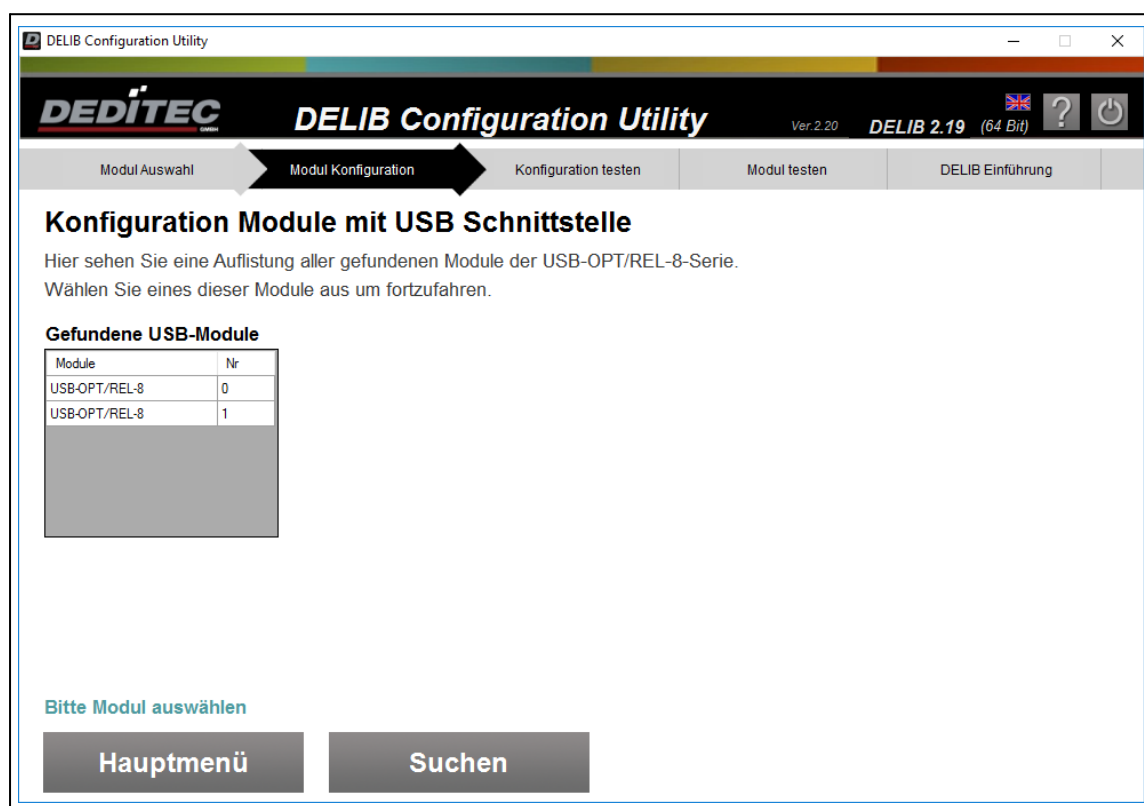
Hauptmenü Test Weiter

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:

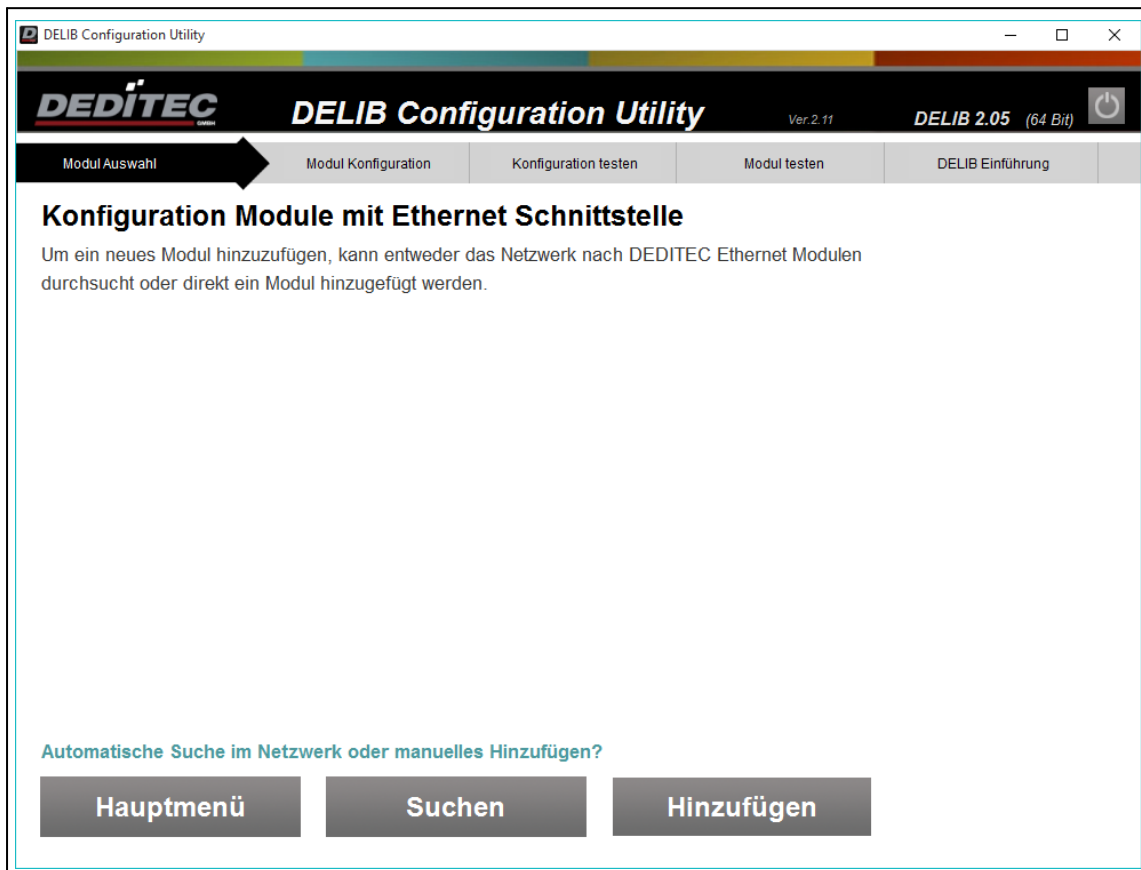
Alle Module haben als Werkseinstellung die NR "0".

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

1.2.3.2.2. Modul Konfiguration Ethernet

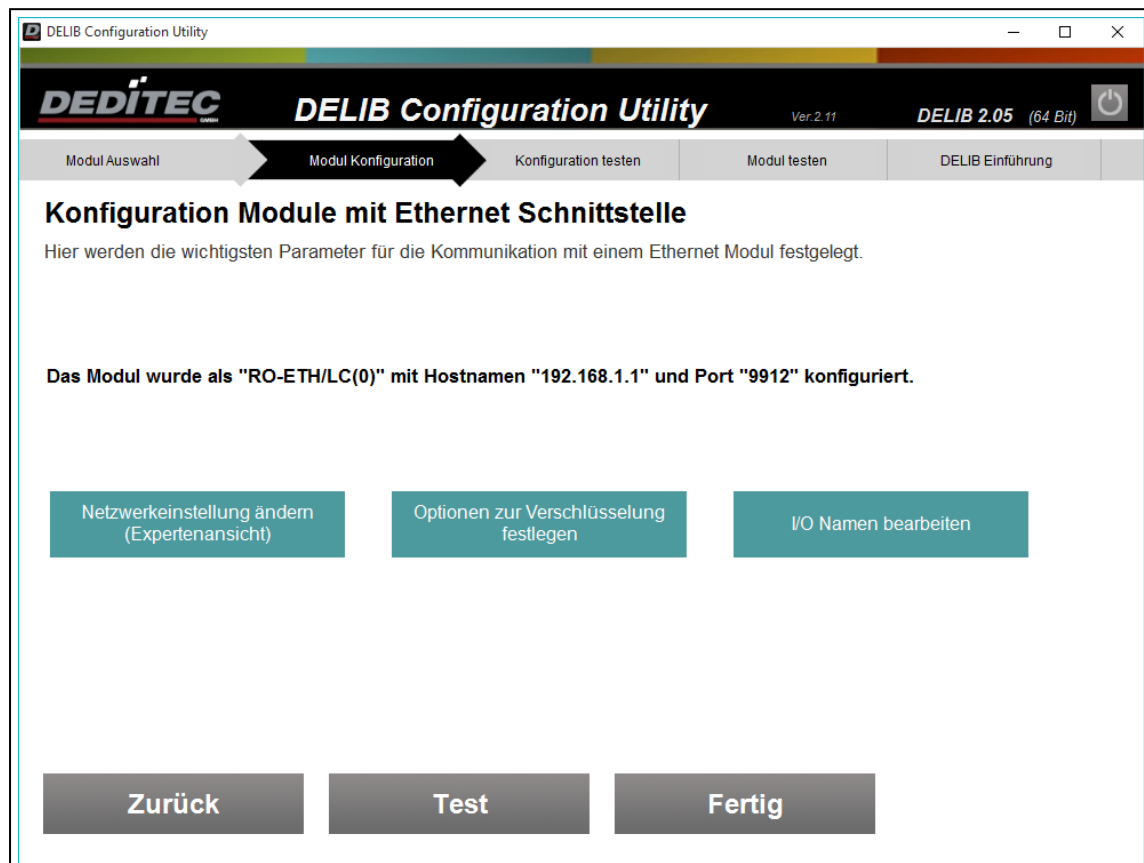
Bei der Konfiguration von Ethernet Produkten kann das Netzwerk automatisch nach DEDITEC Produkten durchsucht werden.

Auch eine manuelle Konfiguration kann durchgeführt werden.



Im letzten Schritt der Konfiguration können Sie weitere Optionen für die Kommunikationsverschlüsselung oder der I/O Namensvergabe durchführen.

Wenn Sie sich für eine manuelle Konfiguration entschließen, beginnen Sie an dieser Stelle.



Bei der Verschlüsselung kann zwischen Deaktiviert, User oder Admin ausgewählt werden.

Deaktiviert

- unverschlüsselte Kommunikation
- kein Zugriff auf System-Einstellungen

User

- verschlüsselte Kommunikation
- Lese-Zugriff auf System-Einstellungen

Admin

- verschlüsselte Kommunikation
- Schreib-Lese-Zugriff auf System-Einstellungen

In das Feld Passwort muss das gewünschte Passwort für die Verschlüsselung eingetragen werden.

Durch einen Klick auf den Button "Einstellungen zur Verschlüsselung auf das Modul übertragen" werden Sie aufgefordert eine Hardware-Taste am Produkt zu betätigen. Erst nach dem Betätigen dieser Taste werden die Verschlüsselungsoptionen vom Produkt übernommen.

siehe Kapitel → **Authentifizierung**

The screenshot shows the 'DELIB Configuration Utility' window. The title bar includes the DEDITEC logo, the application name, version 'Ver. 2.11', and 'DELIB 2.05 (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 Ethernet Schnittstelle' and a subtitle 'Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.' Below this, a status message reads: 'Das Modul wurde als "RO-ETH/LC(0)" mit Hostnamen "192.168.1.1" und Port "9912" konfiguriert.' The interface contains several interactive elements: a button 'Netzwerkeinstellung ändern (Expertenansicht)' on the left; a 'Verschlüsselung' dropdown menu set to 'deaktiviert' with a 'Passwort' text input field below it; a button 'I/O Namen bearbeiten' on the right; and a central button 'Einstellungen zur Verschlüsselung auf das Modul übertragen'. At the bottom, there are three large buttons: 'Zurück', 'Test', and 'Fertig'.

Unsere Ethernet-Produkte, bieten Ihnen die Möglichkeit, für die digitalen und analogen I/Os, Namen zu vergeben. Diese werden beispielsweise auf der Weboberfläche oder in unserer App verwendet.

DELIB Configuration Utility

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

Modul Auswahl → **Modul Konfiguration** → Konfiguration testen → Modul testen → DELIB Einführung

Konfiguration Module mit Ethernet Schnittstelle

Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.

I/O Namen Konfiguration

Auswahl I/O-Typ: Digital Output Kanal-Bereich: Channel (1..8)

Kanal	Name	Kanal	Name
CH 01	Example name	CH 05	Digital Output 5
CH 02	Digital Output 2	CH 06	Digital Output 6
CH 03	Digital Output 3	CH 07	Digital Output 7
CH 04	Digital Output 4	CH 08	Digital Output 8

Zurück Speichern

Hinweis

Zum Speichern der Kanalnamen wird eine Admin-Kommunikation benötigt.
Die maximale Zeichenlänge beträgt 16.

1.2.3.2.2.1. Automatische Suche

Verhalten des DELIB Configuration Utility bei der Verwendung mehrerer Netzwerkadapter in einem Windowssystem.

Bei der Suche nach DEDITEC Ethernet Modulen über den primären Netzwerkadapter (ETH0) des PCs werden die Netzwerkeinstellungen des DEDITEC Produktes ignoriert. Dies bedeutet, dass unsere Ethernet Produkte selbst dann gefunden werden, wenn diese bedingt durch die IP-Adresse und Subnetzmaske nicht im selben Netzwerk sind.

Ist das Ethernet Modul hingegen an einem anderen, bzw. nicht primären Netzwerkadapter (z.B. ETH1) angeschlossen, muss die Netzwerkkonfiguration des Ethernet-Moduls gültig sein, so dass das Modul gefunden wird.

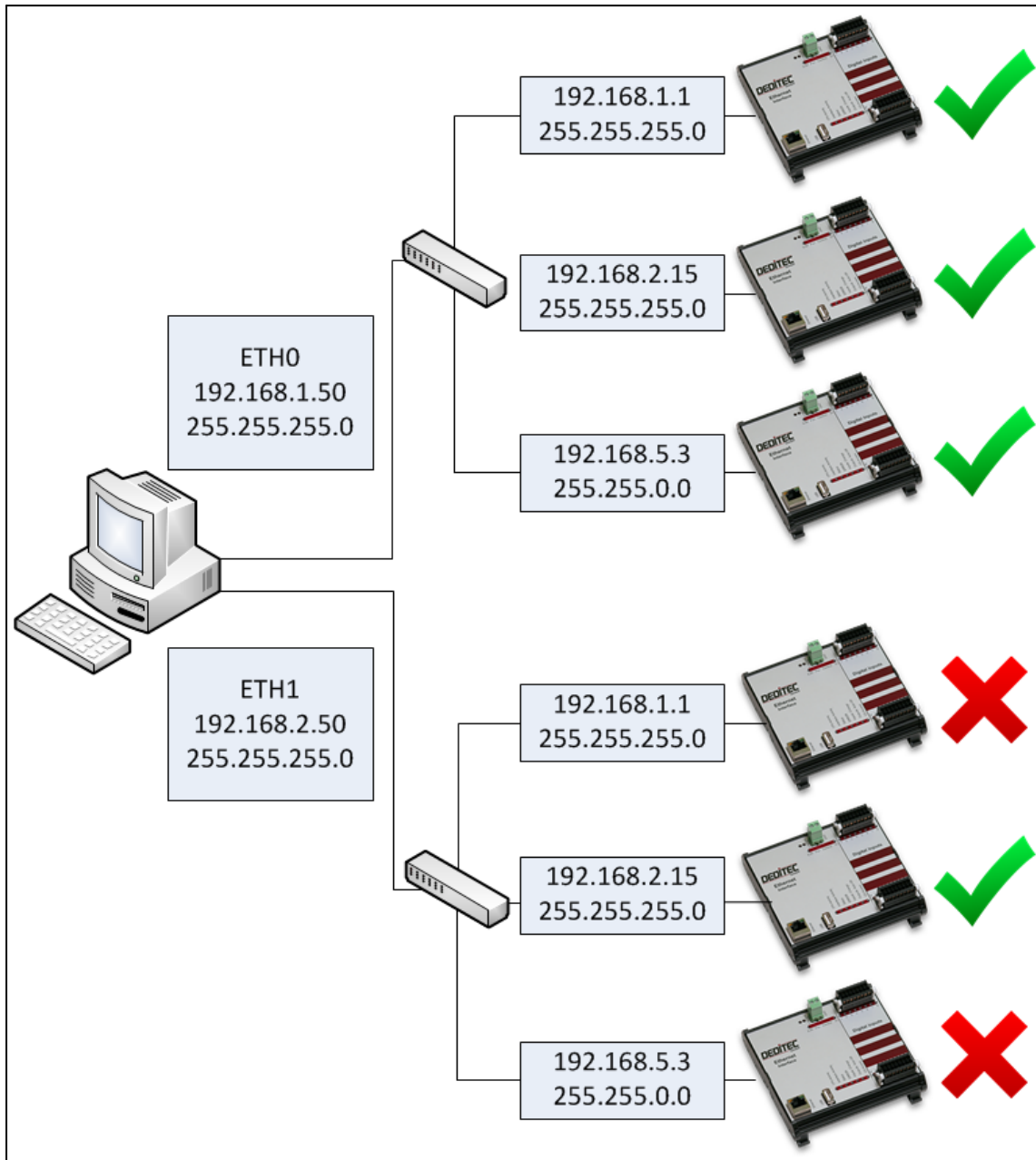
Achtung:

Wird ein Laptop verwendet, ist meistens der integrierte WLAN-Adapter als primärer Netzwerkadapter konfiguriert.

Hierdurch werden Module, die per LAN-Kabel mit dem Laptop verbunden sind, oft nicht erkannt, da die Netzwerkkonfiguration nicht gültig ist.

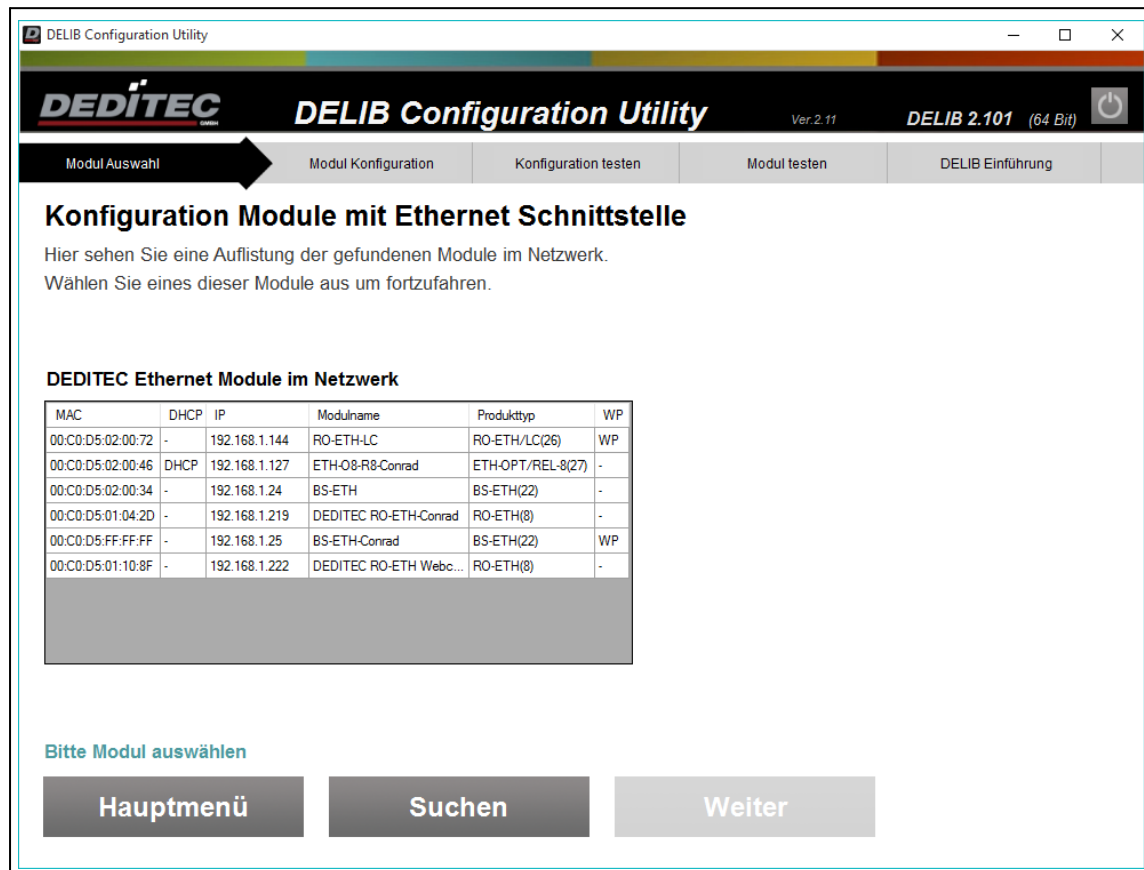
Zur Modul-Einrichtung empfiehlt es sich daher, den WLAN-Adapter kurzzeitig zu deaktivieren.

Folgende Grafik zeigt eine Übersicht der automatischen Suche.



Nach der automatischen Suche werden alle gefundenen DEDITEC Ethernet Produkte in einer Übersicht dargestellt.

Die Übersicht zeigt die MAC-Adresse, den DHCP Status, die IP, den Modulnamen, den Produkttypen sowie den Status der Write-Protection.



Mit einem Klick auf ein Modul kann die Netzwerkkonfiguration für dieses Modul geändert werden. Bitte beachten Sie, dass die Write-Protection ausgeschaltet sein muss, wenn Sie die Einstellungen speichern möchten. Die Write-Protection kann über die Authentifizierung temporär deaktiviert werden.

siehe Kapitel → **Authentifizierung**

Konfiguration Module mit Ethernet Schnittstelle

An dieser Stelle kann die aktuelle Netzwerk Konfiguration des Moduls (rechts) geändert werden.
 Mit "Übernehmen" wird diese Konfiguration an das Modul übertragen.
 Klicken Sie auf "Weiter" um die Kommunikation mit dem Modul zu testen.

DEDITEC Ethernet Module im Netzwerk

MAC	DHCP	IP	Modulname	Produkttyp	WP
00:C0:D5:02:00:72	-	192.168.1.1	RO-ETH-LC	RO-ETH/LC(26)	-
00:C0:D5:02:00:46	-	192.168.1.171	ETH-O8-R8-Conrad	ETH-OPT/REL-8(27)	-
00:C0:D5:FF:FF:FF	-	192.168.1.25	BS-ETH-Conrad	BS-ETH(22)	-
00:C0:D5:02:00:34	-	192.168.1.24	BS-ETH	BS-ETH(22)	-
00:C0:D5:01:10:8F	-	192.168.1.222	DEDITEC RO-ETH Webc...	RO-ETH(8)	-
00:C0:D5:01:04:2D	-	192.168.1.219	DEDITEC RO-ETH-Conrad	RO-ETH(8)	-

Aktuelle Modul Konfiguration

Board Name:

☐ IP-Adresse automatisch beziehen (DHCP)

IP Adresse:

Netzmaske:

Std.-Gateway:

Hinweis

Bitte beachten Sie, dass in diesem Schritt nur die Modul-Konfiguration geändert wird.

Wurde das Modul bereits im DELIB-Configuration Utility konfiguriert, muss diese Konfiguration an die neue Modul-Konfiguration angepasst werden.

1.2.3.2.2.2. Verschlüsselung einrichten

Um wichtige Modul-Einstellungen (z.B. Netzwerkkonfiguration) über TCP bearbeiten zu können, muss die Kommunikation zwischen Modul und PC im sogenannten verschlüsselten Admin-Modus laufen.

Hierzu muss im Modul und auf der PC-Seite ein Passwort zur Verschlüsselung konfiguriert werden.

Diese Konfiguration kann wahlweise über den automatischen Assisten oder manuell erstellt werden.

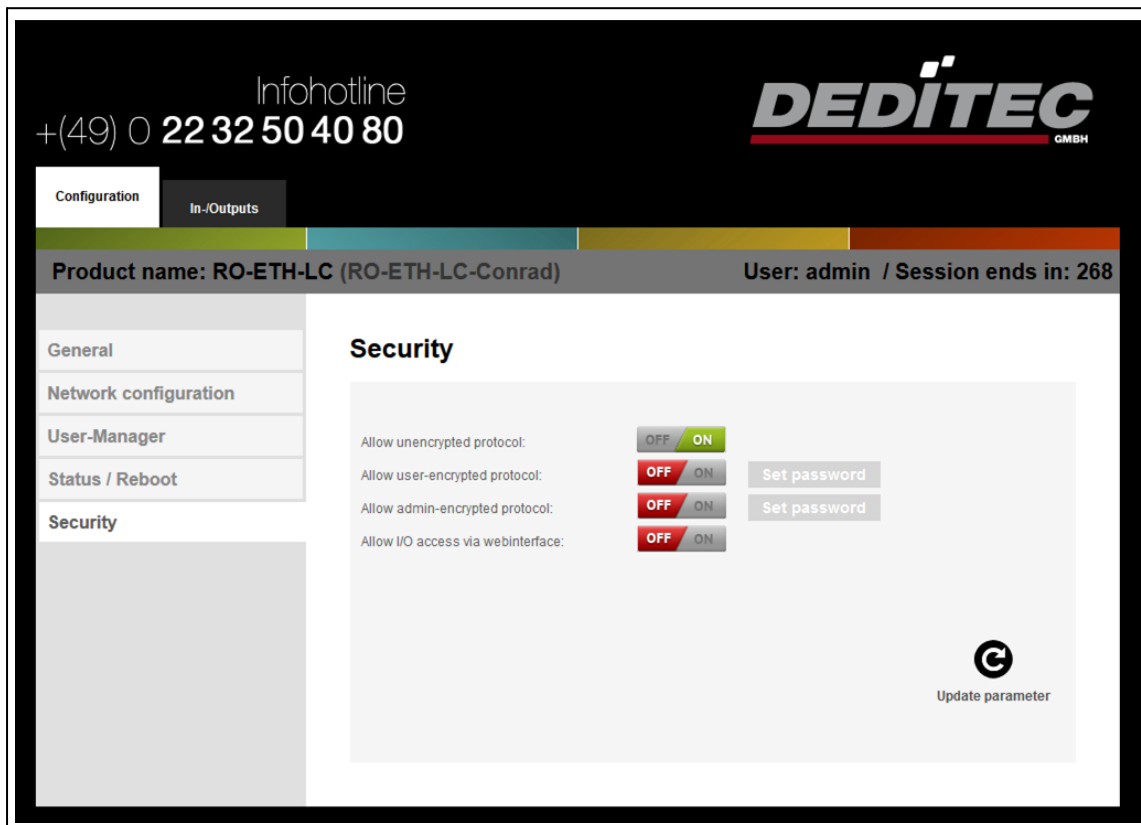
Manuelle Konfiguration

Bei der manuellen Konfiguration, muss jeweils auf der Modul- und PC-Seite ein Passwort zur Verschlüsselung festgelegt werden.

Schritt 1 - Konfiguration Modul-Seite

Zuerst wird das Modul via Weboberfläche konfiguriert. Geben Sie die IP-Adresse des Moduls (Auslieferungszustand 192.168.1.1) in einem Internet-Browser ein. Sollte eine Authentifizierung notwendig sein, loggen Sie sich ein (Auslieferungszustand: user=admin + passwort=admin).

Öffnen Sie nun die Sicherheitseinstellungen (**Configuration** → **Security**)



Falls deaktiviert, aktivieren Sie die Einstellung Allow admin-encrypted protocol. Anschließend können Sie mit Set password ein neues Passwort zur Verschlüsselung einstellen.

Mit Update parameter wird, sofern ein neues Passwort korrekt eingegeben wurde, die Konfiguration abgeschlossen.

Hinweis

Bei der Eingabe des Passwortes sind alle Zeichen erlaubt.

The screenshot displays the DEDITEC web interface. At the top, there is an 'Infohotline' number: +(49) 0 22 32 50 40 80. The DEDITEC GMBH logo is in the top right corner. Below the header, there are two tabs: 'Configuration' (selected) and 'In-/Outputs'. A status bar shows 'Product name: RO-ETH-LC (RO-ETH-LC-Conrad)' and 'User: admin / Session ends in: 223'. On the left, a sidebar menu lists 'General', 'Network configuration', 'User-Manager', 'Status / Reboot', and 'Security' (selected). The main content area is titled 'Security' and contains several settings: 'Allow unencrypted protocol:' with a toggle switch set to 'ON'; 'Allow user-encrypted protocol:' with a toggle switch set to 'OFF' and a 'Set password' button; 'Allow admin-encrypted protocol:' with a toggle switch set to 'ON'; 'New password:' and 'Confirm password:' fields, both masked with dots; and 'Allow I/O access via webinterface:' with a toggle switch set to 'OFF'. At the bottom right of the main content area, there is a circular arrow icon and the text 'Update parameter'.

Schritt 2 - Konfiguration PC-Seite

Starten Sie das DELIB Configuration Utility und wählen Sie das gewünschte Ethernet-Modul aus.

Hier bei Verschlüsselung "Admin" auswählen.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The main header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver. 2.19', and 'DELIB 2.19 (64 Bit)' with a power icon. Below the header is a navigation bar with five tabs: 'Modul Auswahl', 'Modul Konfiguration' (which is active and highlighted with a black arrow), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration Module mit Ethernet Schnittstelle' and includes a sub-header 'Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.' and a 'Manual' icon. Under the 'Netzwerkkonfiguration' section, there are several input fields: 'Modul Auswahl' (a dropdown menu showing 'RO-ETH/LC(0)'), 'Aktiviert' (checked checkbox), 'Hostname verwenden' (unchecked checkbox), 'IP Adresse' (text field with '192.168.1.25'), 'Port' (text field with '9912'), and 'Timeout [msec]' (text field with '5000'). A 'Verschlüsselung' dropdown menu is set to 'deaktiviert'. To the right of these fields is a teal button labeled 'I/O Namen bearbeiten'. At the bottom of the window are three large buttons: 'Hauptmenü', 'Test', and 'Fertig'.

Im neu erscheinenden Passwort Feld das Passwort aus Schritt 1 eintragen.
Mit Test kann die neue verschlüsselten Admin-Modus Kommunikation getestet werden.

Das DELIB Configuration Utility kann nun geschlossen werden.

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', the version 'Ver. 2.19', and 'DELIB 2.19 (64 Bit)' with a power icon. A progress bar at the top is divided into five segments: 'Modul Auswahl', 'Modul Konfiguration', 'Konfiguration testen' (highlighted with a black arrow), 'Modul testen', and 'DELIB Einführung'. Below the header, the section 'Konfiguration Module mit Ethernet Schnittstelle' is displayed, followed by the instruction: 'Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.' A 'Manual' icon is visible on the right. Under the 'Netzwerkkonfiguration' heading, the following fields are present: 'Modul Auswahl' (dropdown menu showing 'RO-ETH/LC(0)'), 'Aktiviert' (checked checkbox), 'Hostname verwenden' (unchecked checkbox), 'IP Adresse' (text field with '192.168.1.25'), 'Verschlüsselung' (dropdown menu showing 'Admin'), 'Port' (text field with '9912'), 'Passwort' (password field with '*****'), and 'Timeout [msec]' (text field with '5000'). There are two teal buttons: 'I/O Namen bearbeiten' and 'Einstellungen zur Verschlüsselung auf das Modul übertragen'. At the bottom, a green status message reads: 'Modul-Kommunikation ist OK! Aktuelle Modul-Firmware ist 2.20'. Three large buttons are at the very bottom: 'Hauptmenü', 'Test', and 'Fertig'.

Automatische Konfiguration

Starten Sie das DELIB Configuration Utility und wählen Sie das gewünschte Ethernet-Modul aus.

DELIB Configuration Utility

DELETEDEC DELIB Configuration Utility Ver. 2.19 DELIB 2.19 (64 Bit)

Modul Auswahl Modul Konfiguration Konfiguration testen Modul testen DELIB Einführung

Konfiguration Module mit Ethernet Schnittstelle

Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.

Netzwerkkonfiguration

Modul Auswahl: RO-ETH/LC(0)

☒ Aktiviert ☐ Hostname verwenden

IP Adresse: 192.168.1.25 Verschlüsselung: Admin

Port: 9912 Passwort: *****

Timeout [msec]: 5000

I/O Namen bearbeiten

Einstellungen zur Verschlüsselung auf das Modul übertragen

Hauptmenü Test Fertig

- 1) Bei Verschlüsselung Admin auswählen
- 2) Im neu erscheinenden Passwort Feld ein Passwort eintragen
- 3) Mit Einstellungen zur Verschlüsselung auf das Modul übertragen, wird die aktuelle Verschlüsselungs-Einstellung auf das Modul übertragen

Hinweis

Bei der Eingabe des Passwortes sind alle Zeichen erlaubt.

Idealerweise sollte sich das Passwort aus einer Kombination von Groß- und Kleinbuchstaben, Zahlen, sowie Sonderzeichen zusammensetzen.

Um das Modul vor unrechtmäßigen Zugriffen zu schützen, muss beim Editieren von System-Einstellungen eine Authentifizierung durchgeführt werden.

Je nach Produkttyp werden Sie aufgefordert, den Firmware-Reset-Taster des Moduls für eine gewisse Zeit zu betätigen (RO-ETH und RO-CPU-800) oder DIP-Schalter-Einstellungen zu verändern (alle anderen Produkte, z.B. RO-ETH/LC, NET-ETH, ETH-RELAIS-8, ...).

Im nachfolgenden Beispiel wird gezeigt, wie die Authentifizierung bei einem RO-ETH/LC Modul durchgeführt wird.

Im ersten Schritt werden Sie aufgefordert, die Stellung von DIP-Schalter 2 zu invertieren.

DELIB Configuration Utility

Ver. 2.19 DELIB 2.19 (64 Bit)

Modul Auswahl Modul Konfiguration Konfiguration testen Modul testen DELIB Einführung

Konfiguration Module mit Ethernet Schnittstelle

Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.

Netzwerkkonfiguration

Modul Auswahl: RO-ETH/LC(0)

☒ Aktiviert ☐ Hostname verwenden

IP Adresse: 192.168.1.25 Verschlüsselung: Admin

Port: 9912 Passwort: *****

Timeout [msec]: 5000

I/O Namen bearbeiten

Einstellungen zur Verschlüsselung auf das Modul übertragen

Schritt 1: DIP-Schalter 2 (Schreibschutz) des ETH-Moduls auf Stellung 1 (0=OFF, 1=ON) setzen

Authentifizierung abbrechen

Hauptmenü Test Fertig

Im zweiten Schritt wird DIP-Schalter2 wieder in die Ausgangsstellung zurückgesetzt.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver. 2.19', and 'DELIB 2.19 (64 Bit)' with a power icon. A progress bar at the top indicates the current step: 'Modul Konfiguration'. Below the header, a navigation bar contains five tabs: 'Modul Auswahl', 'Modul Konfiguration' (active), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration Module mit Ethernet Schnittstelle' and includes a sub-header 'Netzwerkconfiguration'. It contains several input fields: 'Modul Auswahl' (dropdown menu showing 'RO-ETH/LC(0)'), 'Aktiviert' (checked checkbox), 'Hostname verwenden' (unchecked checkbox), 'IP Adresse' (text field with '192.168.1.25'), 'Verschlüsselung' (dropdown menu showing 'Admin'), 'Port' (text field with '9912'), 'Passwort' (password field with '*****'), and 'Timeout [msec]' (text field with '5000'). There are three buttons: 'I/O Namen bearbeiten', 'Einstellungen zur Verschlüsselung auf das Modul übertragen', and 'Authentifizierung abbrechen'. At the bottom, there are three large buttons: 'Hauptmenü', 'Test', and 'Fertig'. A status bar at the very bottom shows three small squares.

DELIB Configuration Utility

DEDITEC DELIB Configuration Utility Ver. 2.19 DELIB 2.19 (64 Bit)

Modul Auswahl Modul Konfiguration Konfiguration testen Modul testen DELIB Einführung

Konfiguration Module mit Ethernet Schnittstelle

Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.

Netzwerkconfiguration

Modul Auswahl: RO-ETH/LC(0)

☒ Aktiviert ☐ Hostname verwenden

IP Adresse: 192.168.1.25 Verschlüsselung: Admin I/O Namen bearbeiten

Port: 9912 Passwort: *****

Timeout [msec]: 5000 Einstellungen zur Verschlüsselung auf das Modul übertragen

Schritt 2: DIP-Schalter 2 (Schreibschutz) des ETH-Moduls auf Stellung 0 (0=OFF, 1=ON) setzen Authentifizierung abbrechen

Hauptmenü Test Fertig

Die Authentifizierung wurde erfolgreich abgeschlossen. Sie sind nun temporär berechtigt, System-Einstellungen zu editieren.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The main header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver. 2.19', and 'DELIB 2.19 (64 Bit)' with a power icon. A navigation bar contains five tabs: 'Modul Auswahl', 'Modul Konfiguration' (which is active and highlighted with a black arrow), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration Module mit Ethernet Schnittstelle'. Below this, it says 'Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.' and includes a 'Manual' icon. The 'Netzwerkconfiguration' section contains the following fields and controls: 'Modul Auswahl' (dropdown menu showing 'RO-ETH/LC(0)'), 'Aktiviert' (checked checkbox), 'Hostname verwenden' (unchecked checkbox), 'IP Adresse' (text field with '192.168.1.25'), 'Verschlüsselung' (dropdown menu showing 'Admin'), 'Port' (text field with '9912'), 'Passwort' (password field with '*****'), 'Timeout [msec]' (text field with '5000'), and a button 'I/O Namen bearbeiten'. There are also two buttons: 'Einstellungen zur Verschlüsselung auf das Modul übertragen' and a button with three dots. At the bottom, there are three buttons: 'Hauptmenü', 'Test', and 'Fertig'. A status message 'Authentifizierung erfolgreich' is displayed in green text.

DELIB Configuration Utility

DEDITEC **DELIB Configuration Utility** Ver. 2.19 **DELIB 2.19** (64 Bit)

Modul Auswahl Modul Konfiguration Konfiguration testen Modul testen DELIB Einführung

Konfiguration Module mit Ethernet Schnittstelle

Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.

Netzwerkconfiguration

Modul Auswahl: RO-ETH/LC(0)

☒ Aktiviert ☐ Hostname verwenden

IP Adresse: 192.168.1.25 Verschlüsselung: Admin I/O Namen bearbeiten

Port: 9912 Passwort: *****

Timeout [msec]: 5000

Einstellungen zur Verschlüsselung auf das Modul übertragen

Authentifizierung erfolgreich

Hauptmenü Test Fertig

Nach erfolgreicher Authentifizierung werden abschließend die Verschlüsselungseinstellungen auf das Modul übertragen.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver. 2.19', and 'DELIB 2.19 (64 Bit)' with a power icon. A progress bar at the top indicates the current step is 'Modul Konfiguration', with other steps being 'Modul Auswahl', 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration Module mit Ethernet Schnittstelle' and includes a sub-header 'Netzwerkconfiguration'. Below this, there are input fields for 'Modul Auswahl' (set to 'RO-ETH/LC(0)'), 'IP Adresse' (192.168.1.25), 'Port' (9912), 'Timeout [msec]' (5000), 'Verschlüsselung' (Admin), and 'Passwort' (masked with asterisks). There are checkboxes for 'Aktiviert' (checked) and 'Hostname verwenden' (unchecked). Two teal buttons are present: 'I/O Namen bearbeiten' and 'Einstellungen zur Verschlüsselung auf das Modul übertragen'. A green status message at the bottom reads 'Einstellungen zur Verschlüsselung erfolgreich übertragen'. At the very bottom are three large buttons: 'Hauptmenü', 'Test', and 'Fertig'.

DELIB Configuration Utility

DEDITEC DELIB Configuration Utility Ver. 2.19 DELIB 2.19 (64 Bit)

Modul Auswahl Modul Konfiguration Konfiguration testen Modul testen DELIB Einführung

Konfiguration Module mit Ethernet Schnittstelle

Hier werden die wichtigsten Parameter für die Kommunikation mit einem Ethernet Modul festgelegt.

Netzwerkconfiguration

Modul Auswahl: RO-ETH/LC(0)

☒ Aktiviert ☐ Hostname verwenden

IP Adresse: 192.168.1.25 Verschlüsselung: Admin

Port: 9912 Passwort: *****

Timeout [msec]: 5000

I/O Namen bearbeiten

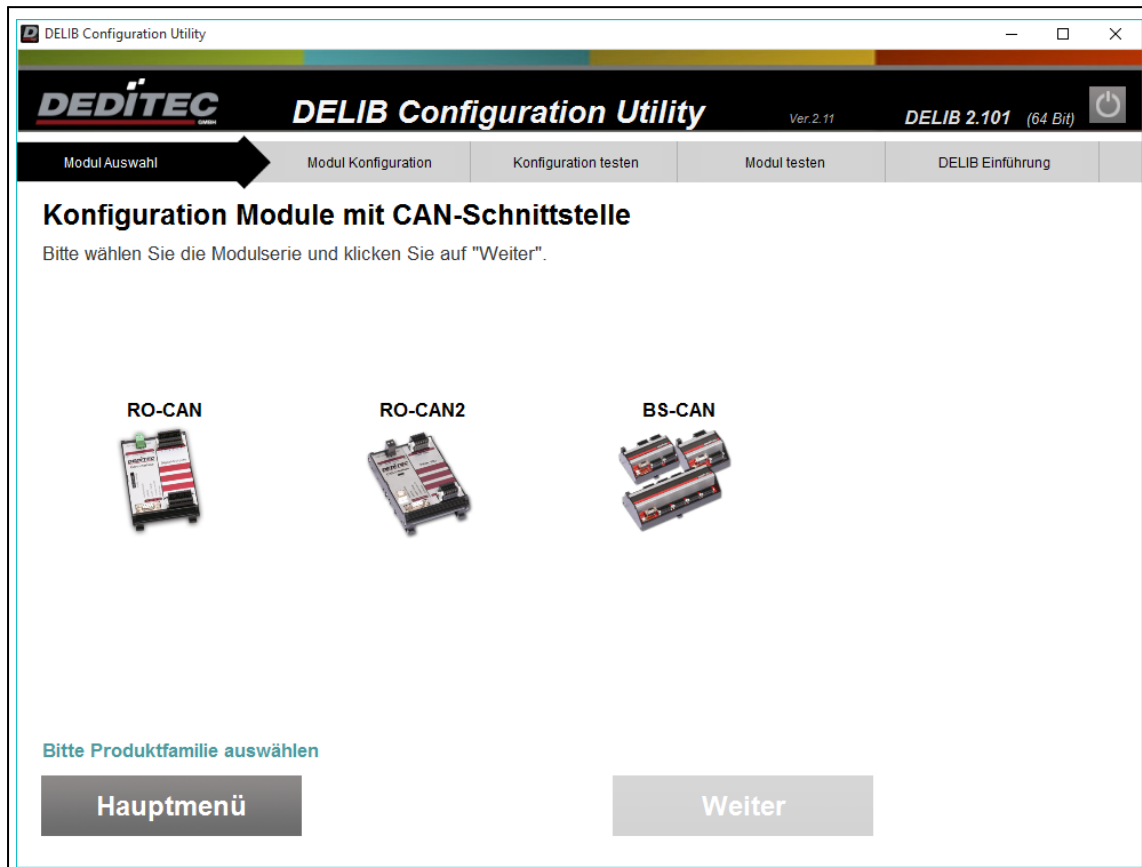
Einstellungen zur Verschlüsselung auf das Modul übertragen

Einstellungen zur Verschlüsselung erfolgreich übertragen

Hauptmenü Test Fertig

1.2.3.2.3. Modul Konfiguration CAN

Wenn Sie eine neue CAN-Konfiguration erstellen, müssen Sie zuerst die Produktfamilie auswählen.



Bei der Konfiguration von RO-CAN Modulen geschieht die Konfiguration des Produktes über eine serielle Verbindung. Hierzu muss der entsprechende COM Port festgelegt werden.

Zusätzlich muss die Anzahl der Übertragungsversuche im Falle eines Kommunikationsfehlers angegeben werden.

Mit dem Button "Test" wird geprüft, ob eine Kommunikation über den angegebenen COM Port möglich ist.

The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The interface has a top header with the 'DEDITEC' logo, the title 'DELIB Configuration Utility', version 'Ver. 2.11', and 'DELIB 2.101 (64 Bit)' with a power icon. Below the header is a navigation bar with five steps: 'Modul Auswahl', 'Modul Konfiguration' (highlighted with a black arrow), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration Module mit CAN-Schnittstelle' and contains the text: 'Die Konfiguration der CAN-Schnittstelle erfolgt seriell. Hierzu muss ein COM-Port ausgewählt werden. Zusätzlich kann die Anzahl der maximalen Übertragungsversuche im Falle eines Kommunikationsfehlers festgelegt werden.' Below this text is a section 'Modul Konfiguration' with three dropdown menus: 'Modul Auswahl' set to 'RO-CAN(0)', 'COM Port' set to 'COM1', and 'Übertragungsversuche' set to '5'. At the bottom are three buttons: 'Zurück', 'Test', and 'Fertig'.

Die Konfiguration eines RO-CAN2 oder BS-CAN Moduls findet über die USB-Schnittstelle statt. Wie bei einem USB-Interface muss bei der Verwendung identischer Produkte eine unterschiedliche Modul-Nr zur eindeutigen Identifikation vergeben werden.

siehe Kapitel **Beispiel zur Konfiguration identischer USB-Module**

Zusätzlich zur Modul-Nr kann auch die Anzahl an Übertragungsversuchen im Falle eines Fehlers festgelegt werden.

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', 'Modul Konfiguration' (which is active and highlighted with a black arrow), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The active tab displays the 'Konfiguration Module mit USB Schnittstelle' section. Below the title, there is explanatory text: 'Hier kann die Modul-Nr. des ausgewählten Moduls geändert werden. Die Modul-Nr. dient zur Unterscheidung mehrerer identischer USB-Module einer USB-Serie. Zusätzlich kann die Anzahl der maximalen Übertragungsversuche im Falle eines Kommunikationsfehlers festgelegt werden.' On the left, a table titled 'Gefundene USB-Module' lists one module: 'RO-CAN2' with 'Nr.' 1. To the right of the table are three input fields: 'Aktuelle Modul-Nr.' with the value '1', 'Neue Module-Nr.' with a dropdown menu showing '0', and 'Übertragungsversuche (für alle USB-Module)' with a dropdown menu showing '5'. At the bottom, there is a blue instruction: 'Klicken Sie "Weiter" zum Übernehmen der neuen Modul-Nr.' and three buttons: 'Hauptmenü', 'Suchen', and 'Weiter'.

Module	Nr
RO-CAN2	1

Aktuelle Modul-Nr.

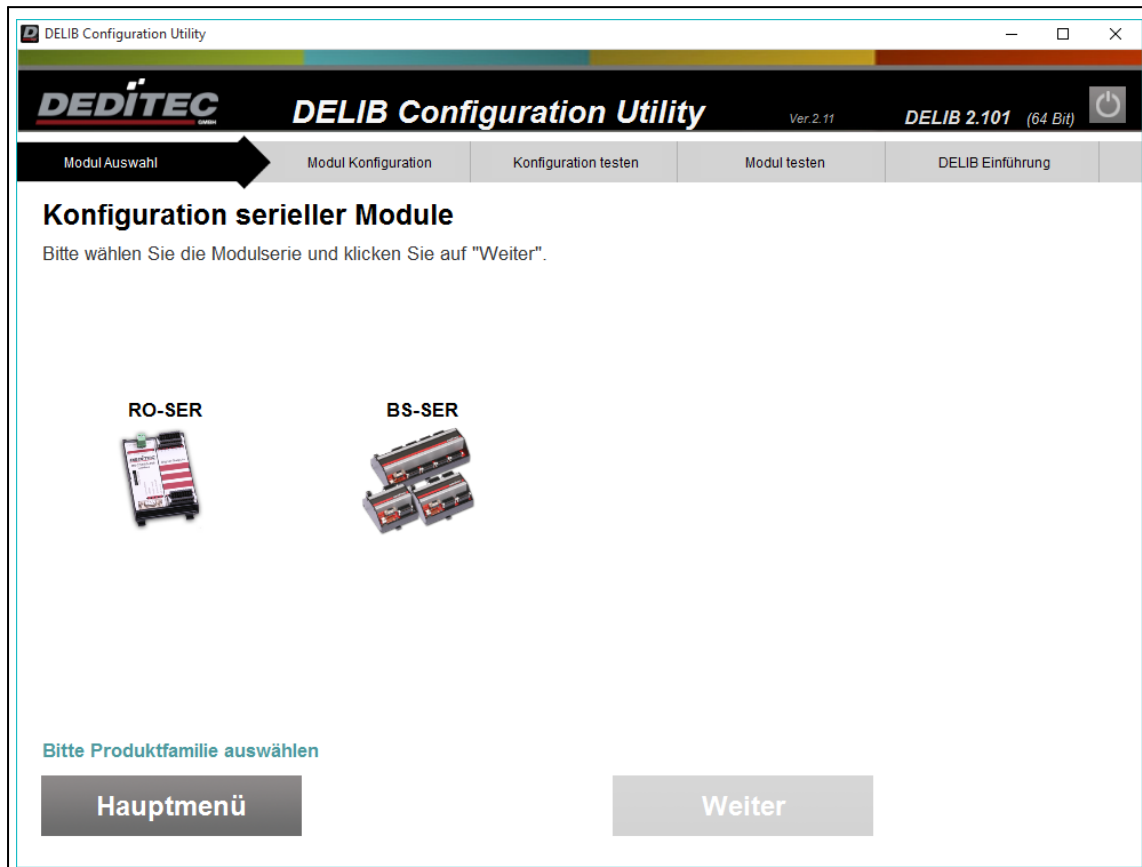
Neue Module-Nr.

Übertragungsversuche (für alle USB-Module)

Klicken Sie "Weiter" zum Übernehmen der neuen Modul-Nr.

1.2.3.2.4. Modul Konfiguration Seriell

Wenn Sie eine neue serielle Konfiguration erstellen, müssen Sie zuerst die Produktfamilie auswählen.



Bei der Konfiguration der seriellen Produkte muss der entsprechende COM-Port festgelegt werden.

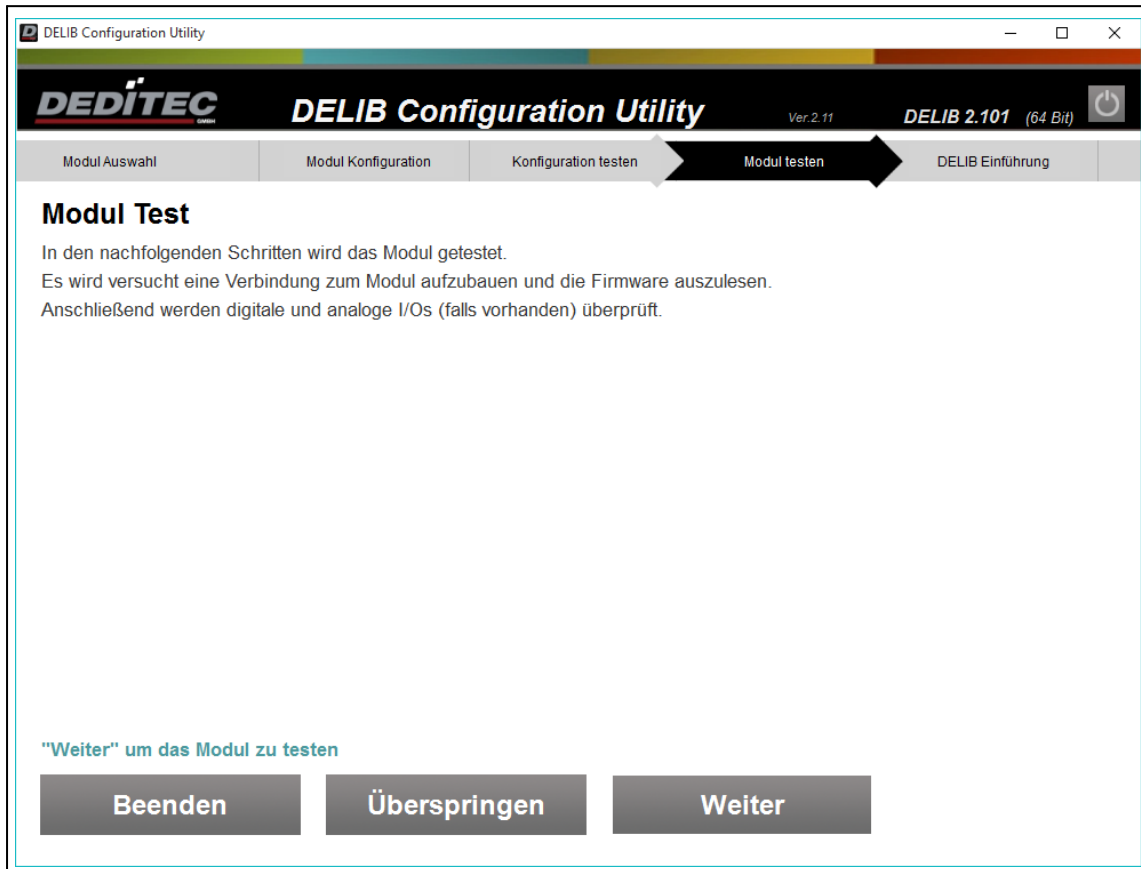
Zusätzlich muss die Anzahl der Übertragungsversuche im Falle eines Kommunikationsfehlers angegeben werden.

Mit dem Button "Test" wird geprüft, ob eine Kommunikation über den angegebenen COM-Port möglich ist.

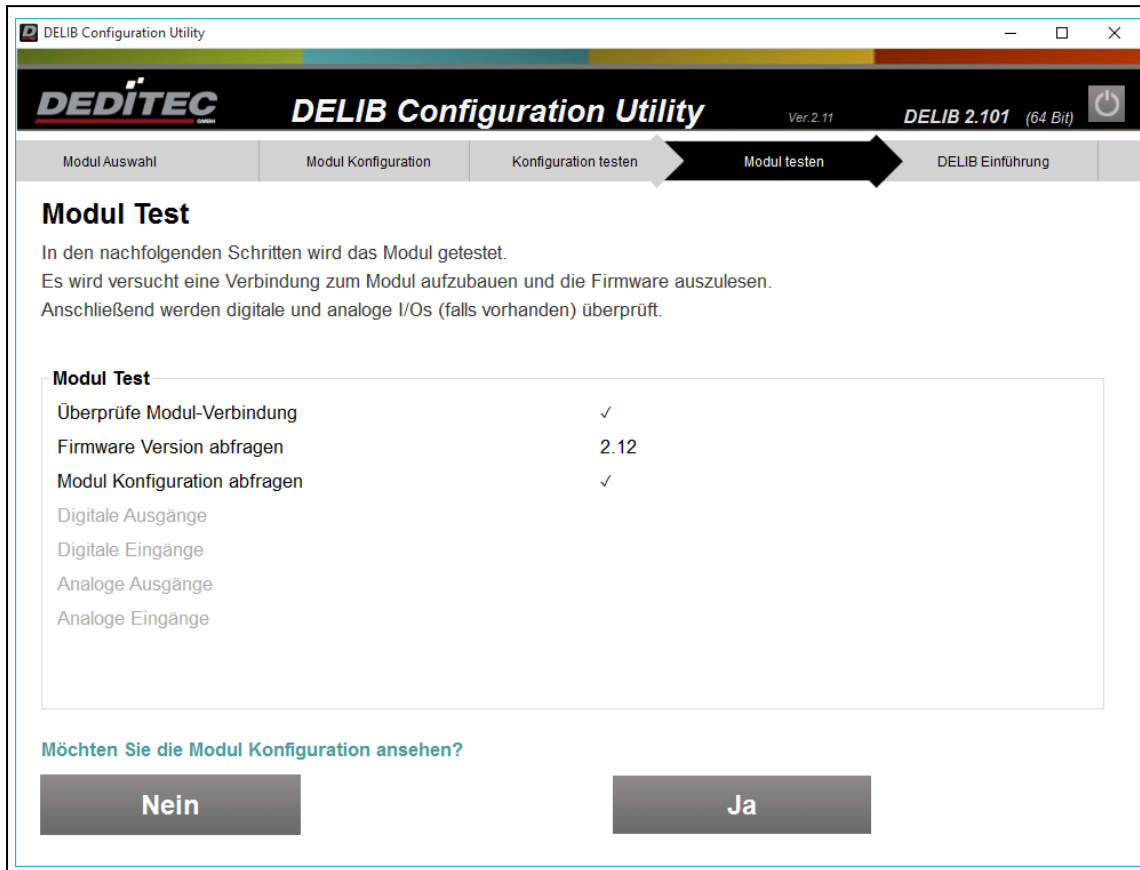
The screenshot shows the 'DELIB Configuration Utility' window. The title bar reads 'DELIB Configuration Utility'. The main header features the 'DEDITEC' logo, the product name 'DELIB Configuration Utility', the version 'Ver.2.11', and the model 'DELIB 2.101 (64 Bit)' with a power icon. Below the header is a navigation bar with five tabs: 'Modul Auswahl', 'Modul Konfiguration' (which is active and highlighted with a black arrow), 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The main content area is titled 'Konfiguration serieller Module' and contains the instruction: 'Hier werden die wichtigsten Parameter für die Kommunikation mit einem seriellen Modul festgelegt.' Under the 'Modul Konfiguration' section, there are three configuration items: 'Modul Auswahl' set to 'RO-SER(0)', 'COM Port' set to 'COM1', and 'Übertragungsversuche' set to '5'. At the bottom of the window are three buttons: 'Zurück', 'Test', and 'Fertig'.

1.2.3.3. 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		

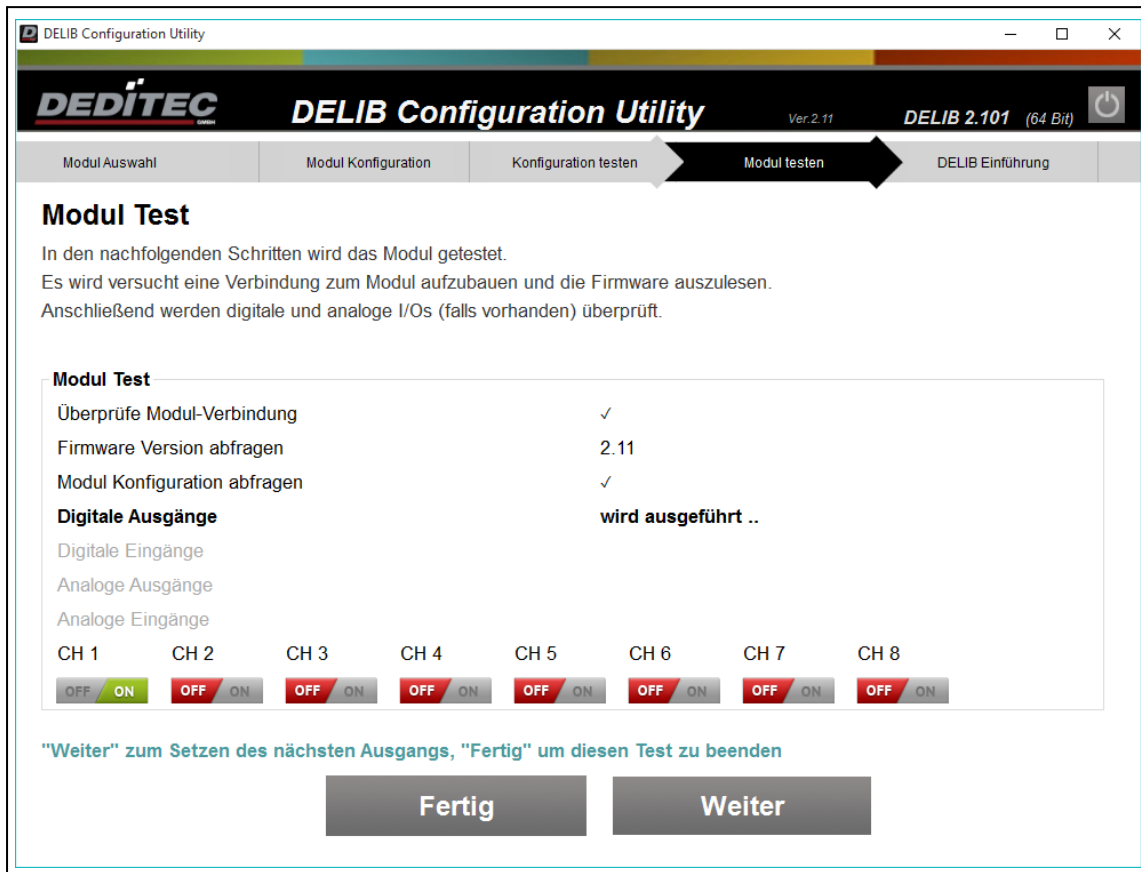
Features-General	Features-Digital I/O	Features-Analog I/O	Features-Special
Supported by FW OK	DI Commands OK	DA Commands OK	Watchdog Commands -
Dev IO registry error OK	DI CNT Commands OK	AD Commands OK	Stepper Commands OK
AD FIFO OK	DI CNT Latch Feature -	Pt100 Commands OK	
Set-Clr Bit Commands OK	DI 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 FV OK	PulseGen Commands -		
ETH OK	CNT8 Commands -		
CAN -	Auto-Off Timeout Commar OK		
RS232 -			
RS485 -			
USB1 -			
USB2 -			

Submodule-Info's

Sub: 0 - RO-AD16DA4	FW:2.11
Sub: 1 - RO-08_R8	FW:2.10
Sub: 2 - RO-STEPPER2	FW:1.29
Sub: 3 - RO-PT100	FW:1.03

EXIT

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

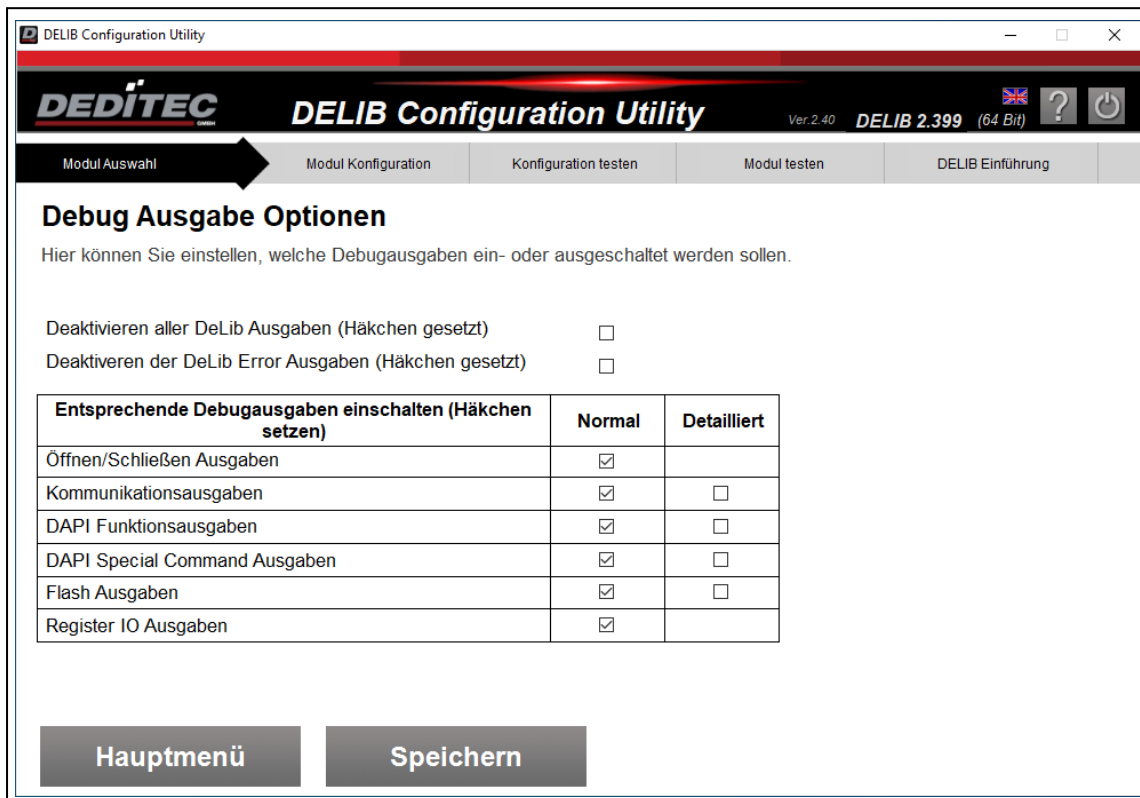


Wurden alle Tests erfolgreich durchlaufen, ist das Produkt einsatzbereit.

1.2.3.4. 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', 'Modul Konfiguration', 'Konfiguration testen', 'Modul testen', and 'DELIB Einführung'. The 'Modul Konfiguration' tab is active. 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 column lists the debug output types, the second column is 'Normal', and the third column 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"/>	

1.2.4. Benutzung des Moduleselectors

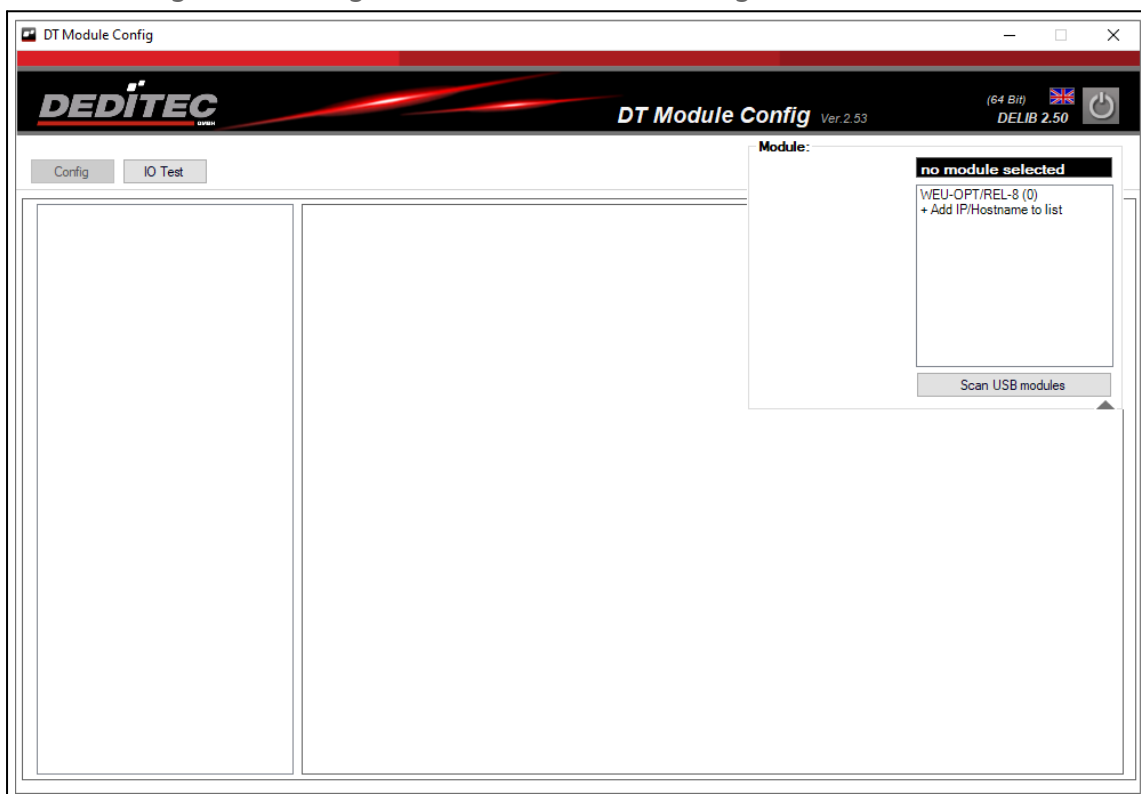
Um unsere Produkte mit der DEDITEC-Software benutzen zu können, müssen diese über den Modul Selector ausgewählt werden.

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

1.2.4.1. 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.

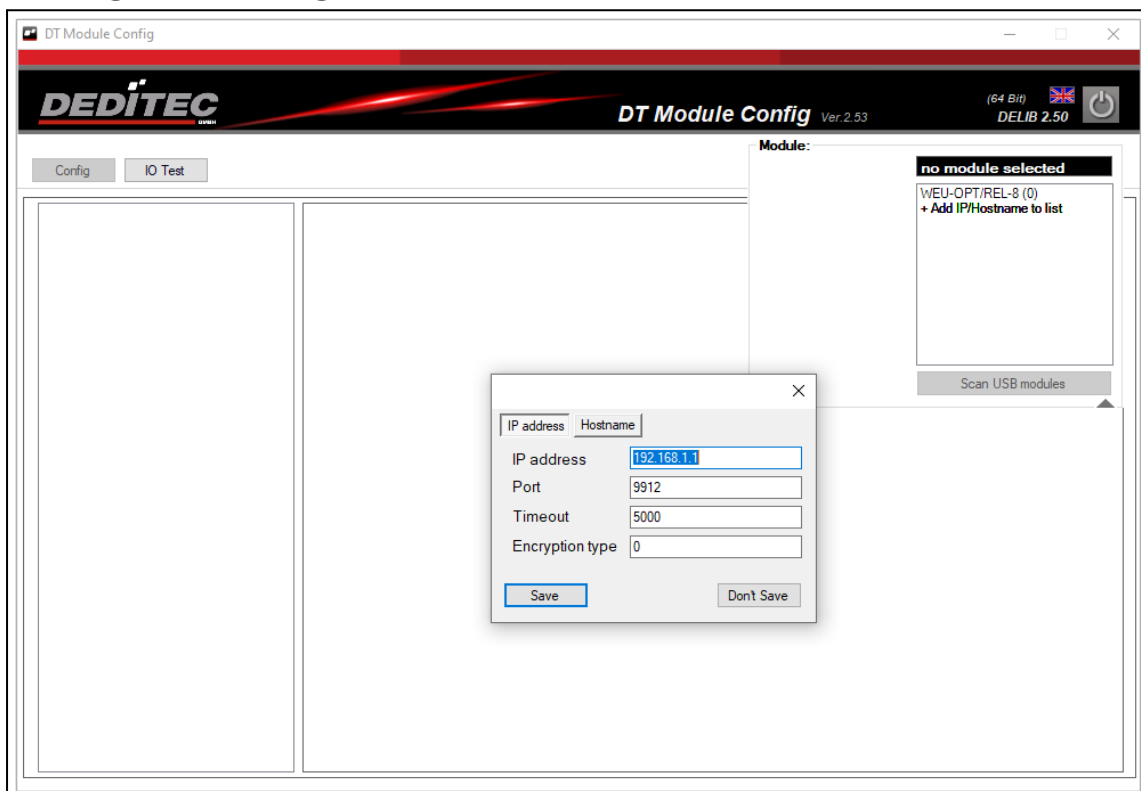


1.2.4.2. via Ethernet

Sollten Sie Ihr Modul über die Ethernet-Schnittstelle angeschlossen haben, können Sie das Modul direkt über die im Netzwerk eingebundene IP-Adresse finden.

Fragen Sie hierfür gegebenenfalls Ihren Systemadministrator.

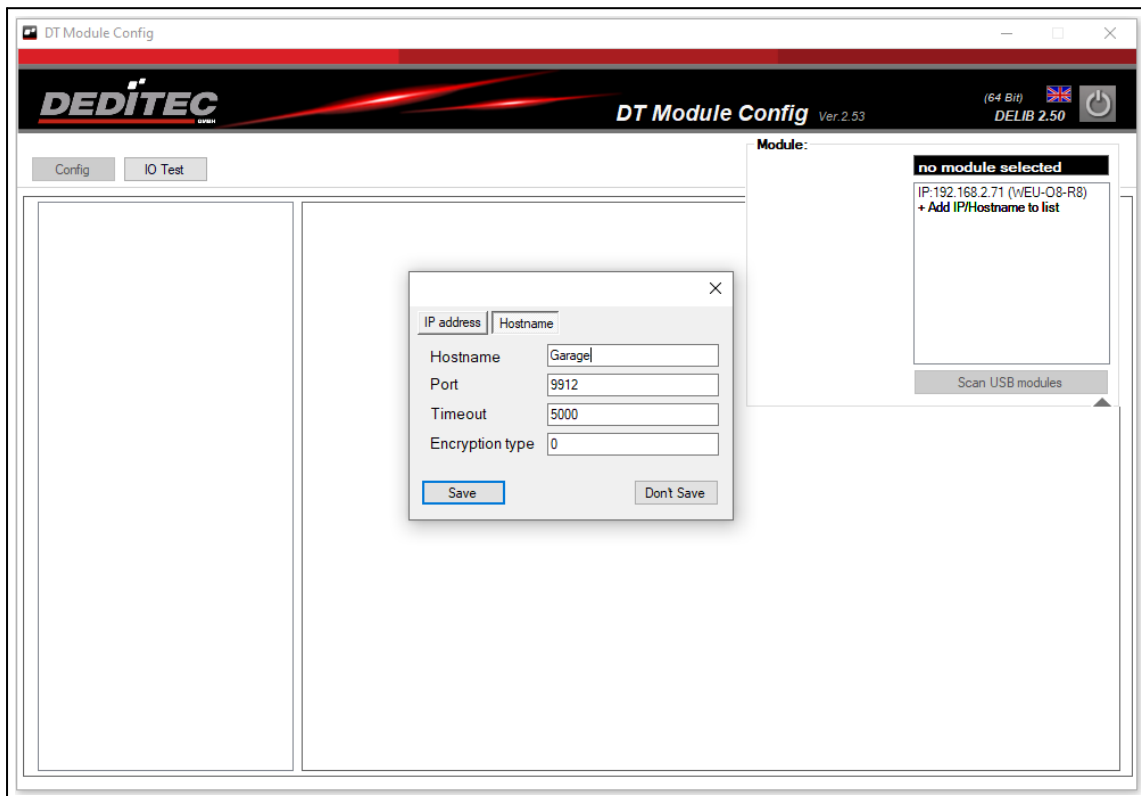
Klicken Sie im Modul Selector auf "Add IP/Hostname to list", tragen Sie dort die automatisch erhaltene IP-Adresse unter dem Reiter "IP Adresse" ein und bestätigen Sie die Eingabe mit "Save".



Ist Ihr Modul im DHCP-Modus (siehe Kapitel: **LAN Netzwerkkonfiguration**) können Sie dieses auch mit Hilfe des Board Namens verbinden.

Diesen finden Sie im Modul Config im Bereich "LAN - Netzwerkinformation".

Für eine Verbindung per Board-Name klicken Sie im Modul Selector auf "Add IP/Hostname to list". Tragen Sie dort den Namen unter dem Reiter "Hostname" ein und bestätigen Sie die Eingabe mit "Save".



1.2.4.3. via WiFi / WPS

WiFi (nur bei WEU-Modulen)

Um das Modul per WiFi zu verbinden, muss dieses im Vorfeld mit USB oder Ethernet verbunden werden.

Nun kann unter dem Menüpunkt WiFi-Konfiguration WiFi aktiviert werden. Die an das Modul vergebene IP-Adresse finden Sie unter WiFi-Info.

WPS (nur bei WEU-Modulen)

Ist das Modul noch nicht mit dem PC-Netzwerk verbunden, führen Sie den Verbindungsaufbau, wie im Kapitel CFG-Taster beschrieben, durch.

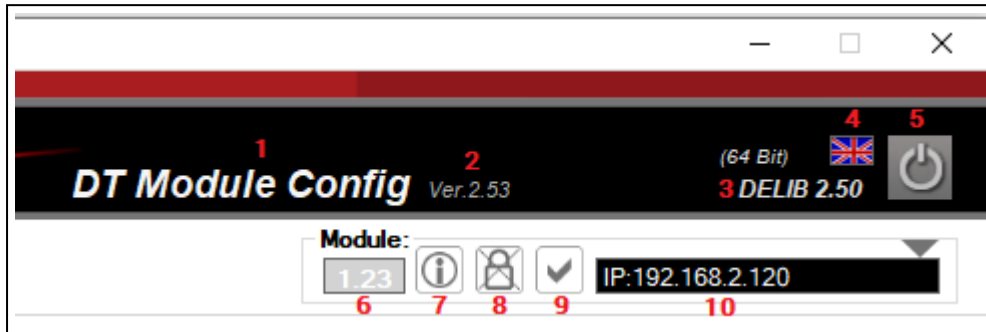
Klicken Sie danach im Modul Selector auf "Add IP/Hostname to list", tragen Sie dort die automatisch erhaltene IP-Adresse unter dem Reiter "IP Adresse" ein und bestätigen Sie die Eingabe mit "Save".

Fragen Sie hierfür gegebenenfalls Ihren Systemadministrator.

Sie können eine WPS-Verbindung auch mit Hilfe des Module Config starten. Führen Sie dafür die Schritte wie im Kapitel: WiFi WPS-Verbindung beschrieben aus.

1.2.4.4. 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 Fahnnensymbol 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.

1.2.5. 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.

1.2.5.1. Modul Konfigurationen

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

1.2.5.1.1. 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

FormConfigModuleOverview (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.

1.2.5.1.2. Modul-Identifikation

Identifizieren Sie das Modul, welches Sie gerade mit dem Modul Config ansprechen, um Verwechslungen vorzubeugen.

Dies ist besonders hilfreich, wenn mehrere Module gleichzeitig in Betrieb sind.

Durch das Betätigen von "Start" wird die Identifikation gestartet.

Es fängt nun die Status-LED wiederholt an zu blinken.

Dieser Vorgang wird durch drücken von "Stop" beendet.

Identifikation

Hier können Sie Ihr Modul identifizieren

Identify module

Modulidentifikation läuft!

Mit der Identifikations-Funktion können Sie feststellen, welches Ihrer Module momentan durch das Modul Config angesteuert wird. Dies ist besonders hilfreich, bei der gleichzeitigen Verwendung mehrerer Module. Durch das Betätigen der 'Start' Taste blinken folgende LEDs zur Identifikation

- bei unseren ETH_LC-Modulen blinkt die 'Int.Act' LED
- bei unseren WEU-Modulen blinkt die 'Status-LED'

Das Bedienen der 'Stop' Taste beendet diesen Vorgang.

FormConfModulIdent (Ver. 1.01)

1.2.5.1.3. LAN Netzwerkinformationen

Alle wichtigen LAN Netzwerkinformationen auf einen Blick.

Auf dieser Informationsseite finden Sie die aktuellen LAN-Einstellungen Ihres Modules.

Info
Hier finden Sie die Netzwerkinformationen des ausgewählten Ethernet-Produktes

MAC address	40:F5:20:44:60:EB
Board name (Hostname in DHCP mode)	WEU-Q8-R8
LAN status	Static-IP success
DHCP active	<input type="checkbox"/>
IP address	192.168.2.71
Subnet mask	255.255.255.0
Default gateway	192.168.2.254
TCP port	9912

☐ Auto Refresh

FormOfnNetwork LANInfo (Ver. 1.01)

MAC-Adresse

Die MAC-Adresse ist die physikalische Adresse des Produktes und ist fest mit der Hardware verbunden.

Board Name

Zeigt den aktuellen Board Name Ihres Modules an.

LAN-Status

Hier wird der Verbindungsstatus Ihres angeschlossenen Moduls angezeigt.

Sollte bei Ihnen der Status "Query not supported (FW-Update)" dargestellt werden, benötigt Ihre Modul eine aktuellere Firmware.

DHCP active

Zeigt an, ob das Modul über DHCP verbunden ist.

IP-Adresse, Netzmaske, Standard Gateway und TCP-Port


Zeigt die aktuelle Netzwerkkonfiguration, mit der das Modul verbunden ist, an.

1.2.5.1.4. LAN Netzwerkeinstellungen

Hier können Sie Änderungen an den Netzwerkeinstellungen des ausgewählten WEU-Moduls vornehmen.

Konfiguration

Hier können Sie die Netzwerkeinstellungen des ausgewählten Ethernet-Produktes ändern

Board name (Hostname in DHCP mode)	<input type="text" value="WEU-08-R8"/>
Network configuration protection active 	<input type="checkbox"/>
DHCP active	<input type="checkbox"/>
IP address	<input type="text" value="192.168.2.71"/>
Subnet mask	<input type="text" value="255.255.255.0"/>
Default gateway	<input type="text" value="192.168.2.254"/>
TCP port	<input type="text" value="9912"/>

Form("ToNetwork1_AbIConfin_Ver. 1.01")

Board Name

Der Board Name kann zur Geräteidentifizierung genutzt werden. Ist DHCP aktiv, wird der Board Name als Hostname verwendet.

Diese Option ist besonders bei der Verwendung mehrerer Module sehr hilfreich.

So können Sie zum Beispiel einem Modul einen speziellen Board Name wie "Garage" oder "Gartenlaube" vergeben. Im Modul Selector können Sie das Modul dann unter diesem Namen direkt ansteuern.

Mehr Infos zum Anbinden des Moduls per Board Name

siehe Kapitel: **Benutzung des Modulselectors**

Network configuration protection active

Wenn diese Option aktiviert ist, können Netzwerkkonfiguration nur noch über die Weboberfläche geändert werden.

Dies verhindert unautorisierten Zugriff auf die Netzwerkkonfiguration (Beispielsweise über das Modul Config).

DHCP active

Ist diese Option aktiviert, versucht das Gerät beim Start eine gültige IP-Adresse von einem DHCP Server im Netzwerk zu beziehen.

Der Board Name wird als Hostname verwendet.

IP address, Subnet mask, Default gateway und TCP port

Diese Einstellungen werden verwendet, wenn DHCP deaktiviert ist. Fragen Sie gegebenenfalls bitte Ihren Systemadministrator.

Werkseinstellungen laden

Hier wird die IP-Konfiguration auf die Werkseinstellungen zurückgesetzt. Diese sehen wie folgt aus:

Werkeinstellungen	
Board name	Modul abhängig
Network protection	off
DHCP	off
IP-Adresse	192.168.1.1
Subnet mask	255.255.255.0
Default gateway	192.168.1.254
TCP Port	9912

1.2.5.1.5. WiFi Netzwerkinformationen

Alle wichtigen WiFi Netzwerkinformationen auf einen Blick.

Auf dieser Informationsseite finden Sie die aktuellen WiFi-Einstellungen Ihres Moduls.

Info

Hier finden Sie die Netzwerkinformationen des ausgewählten WiFi-Produktes

MAC address	40:F5:20:44:60:E8
Board name (Hostname in DHCP mode)	WEU-O8-R8_W
WLAN status	starting WIFI connection..
WLAN active	<input checked="" type="checkbox"/>
IP address	0.0.0.0
Subnet mask	0.0.0.0
Default gateway	0.0.0.0
TCP port	9912
Router name	TESTssid
Password	TESTpwd

☒ Auto Refresh

FormToNetworkWiFiInfo (Ver. 1.01)

MAC-Adresse

Die MAC-Adresse ist die physikalische Adresse des Produkts und ist fest mit der Hardware verbunden.

Board Name

Zeigt den aktuellen Board Name Ihres Moduls an.

WLAN-Status

Hier wird der Verbindungsstatus Ihres angeschlossenen Moduls angezeigt.
Sollte bei Ihnen der Status "Query not supported (FW-Update)" dargestellt werden, benötigt Ihr Modul eine aktuellere Firmware.

WLAN active

Zeigt an, ob das Modul über WLAN verbunden ist.

IP-Adresse, Netzmaske, Standard Gateway und TCP-Port

Zeigt die aktuelle Netzwerkkonfiguration mit der das Modul verbunden ist, an.

Router name

Zeigt an, welcher Router Name zum Verbinden via WLAN verwendet wird.

Password


Zeigt das verwendete Routerpasswort an.

1.2.5.1.6. WiFi Netzwerkeinstellungen

Hier können Sie die Änderungen an den WiFi-Einstellungen Ihres WEU-Moduls vornehmen.

Konfiguration

Hier können Sie die Netzwerkeinstellungen des ausgewählten WiFi-Produktes ändern

Board name (Hostname in DHCP mode)	<input type="text" value="WEU-08-R8_W"/>
WLAN active	<input type="checkbox"/>
Router name	<input type="text" value="TESTssid"/>
Password	<input type="text" value="TESTpwd"/>
TCP port	<input type="text" value="9912"/> 

Form für Netzwerk-WiFi Konfig (Ver. 1.01)

Board name

Der Board Name kann zur Geräteidentifizierung genutzt werden. Ist DHCP aktiv, wird der Board Name als Hostname verwendet.

Diese Option ist besonders bei der Verwendung mehrerer Module sehr hilfreich.

So können Sie zum Beispiel einem Modul einen speziellen Board Name wie "Garage" oder "Gartenlaube" vergeben. Im Modul Selector können Sie das Modul dann unter diesem Namen direkt ansteuern.

(Mehr Infos zum Anbinden des Modules per Board Name siehe Kapitel: Benutzung des Modulselectors)

WLAN active

Mit dieser Option können Sie das WLAN Ihres Moduls aktivieren oder deaktivieren.

Router name

Hier können Sie den Routername eintragen, welcher bei einer Verbindung via WLAN verwendet werden soll.

Fragen Sie hierfür gegebenenfalls Ihren Systemadministrator.

Password

Hier können Sie das Routerpasswort des verwendeten Routers eingetragen werden.

Fragen Sie hierfür gegebenenfalls Ihren Systemadministrator.

TCP port

Hier wird der verwendete TCP-Port dargestellt. Eine Änderung des Ports kann nur bei den LAN Netzwerkkonfigurationen vorgenommen werden.

Werkseinstellungen laden

Hier wird die WiFi-Konfiguration auf die Werkseinstellungen zurückgesetzt. Diese sehen wie folgt aus:

Werkseinstellungen	
Board name	Modul abhängig
WLAN active	off
Routername	TESTssid
Password	TESTpwd

1.2.5.1.7. WiFi WPS-Verbindung

Hier können Sie Ihr WEU-Modul mit Hilfe der WPS-Funktion mit Ihrem PC-Netzwerk verbinden.

Klicken Sie hierfür den WPS-Knopf an Ihrem Router. Informationen dazu finden Sie im Handbuch Ihres Netzwerkgerätes.

Klicken Sie während des Suchlaufs Ihres Routers auf den "WPS-Start"-Knopf im Modul Config oder betätigen Sie den Taster direkt auf dem Board Ihres WEU-Moduls.

Mehr Informationen über den CFG-Taster siehe Kapitel **CFG-Taster**

Bei einer erfolgreichen Verbindung erscheint nun der verwendete Router Name und das Passwort.

WPS

Hier können Sie eine Verbindung per WPS-Funktion mit Ihrem WiFi-Modul herstellen

WLAN status	starting WPS connection..
Router name	
Password	

WPS-Start

WPS-Stop

Mit Hilfe der WPS-Funktion, lässt sich Ihr Modul mit nur wenigen Schritten schnell und einfach automatisch mit Ihrem Router verbinden.

Für eine erfolgreiche Verbindung mit dem Netzwerk müssen folgende Punkte durchgeführt werden:

1. Betätigen Sie die 'WPS-Taste' an Ihrem Router (Hilfe dazu finden Sie im Handbuch des Routers)
2. Klicken Sie anschließend auf den obigen 'WPS-Start'-Knopf im Module Config
3. Das Modul verbindet sich nun automatisch mit Ihrem Router

Der aktuellen Status Ihrer Verbindung wird bei 'WLAN Status' angezeigt.
Das Bedienen der 'Stop-Taste' beendet diesen Vorgang.

☒ Auto Refresh

FormCtoNetworkWiFiWPS (Ver. 1.01)

1.2.5.1.8. NTP-Konfiguration

Hier können Änderungen am NTP-Service vorgenommen werden.

NTP-Konfiguration

Hier können Sie Änderungen am NTP Service vornehmen

NTP service active	<input checked="" type="checkbox"/>
Server	<input type="text" value="0.de.pool.ntp.org"/>
Port	<input type="text" value="123"/>
Timezone	<input type="text" value="(GMT) Greenwich Mean Time: Dublin, Edinburgh"/>

FirmenNetzwerkNTP (Ver. 1.01)

NTP service active

Ist diese Option aktiviert, wird der NTP-Service aktiviert.

Server

Hier können Sie den NTP-Server, der verwendet werden soll, einstellen.

Port

Hier können Sie den NTP-Port, der verwendet werden soll, einstellen.

Timezone

Hier können Sie die Zeitzone, die vom Modul verwendet werden soll, einstellen.

Werkseinstellungen laden

Hier wird die TCP-Verschlüsselungseinstellungen auf die Werkseinstellungen zurückgesetzt. Diese sehen wie folgt aus:

Werkseinstellungen	
NTP service active	on
Server	0.de.pool.ntp.org
Port	123
Timezone	(GMT) Greenwich Mean Time: Dublin, Edinburgh, Lisbon, London

1.2.5.1.9. Serielle Konfiguration

Hier können Sie die gesamte Konfiguration unserer Produkte mit serieller Schnittstelle vornehmen.

Seriell

Hier können Sie die gesamte Konfiguration unserer Produkte mit serieller Schnittstelle vornehmen

Special mode active	<input type="checkbox"/>
Baud rate	<input type="text" value="625000"/>
RS485 modul address	<input type="text" value="0"/>
Echo active	<input type="checkbox"/>
Register modus active	<input type="checkbox"/>

FormC3aSenell (Ver 1.01)

Vorzugsmodus (Special mode)

Im Vorzugsmodus wird das Modul automatisch mit folgenden Einstellungen betrieben:

Baud rate: 115200

Modul-Nr. 0

Echo = Off

Register-Mode = On

Baud rate

Ist der Vorzugsmodus deaktiviert, kann die Geschwindigkeit der Kommunikation festgelegt werden.

625000

250000

125000

115200

57600

50000

38400

19200

9600

4800

2400

1200

600

300

RS485 Modul-adresse

Adresse für die Identifikation im RS485 Bus.

Echo

Seriell empfangene Zeichen werden vom Modul zurückgesendet.

Registermodus

Deaktivieren Sie den Registermodus um den Textmodus zu aktivieren.

1.2.5.1.10. I/O Kanal-Namen

Hier können Sie die Kanalnamen Ihres Haupt- bzw. Submoduls einstellen.

I/O Kanal-Namen

Hier können Sie die Namen der einzelnen I/O-Kanäle ändern und speichern

Ch. 0	<input type="text" value="Kanal 0"/>	Ch. 8	<input type="text" value="Kanal 8"/>
Ch. 1	<input type="text" value="Kanal 1"/>	Ch. 9	<input type="text" value="Kanal 9"/>
Ch. 2	<input type="text" value="Kanal 2"/>	Ch. 10	<input type="text" value="Kanal 10"/>
Ch. 3	<input type="text" value="Kanal 3"/>	Ch. 11	<input type="text" value="Kanal 11"/>
Ch. 4	<input type="text" value="Kanal 4"/>	Ch. 12	<input type="text" value="Kanal 12"/>
Ch. 5	<input type="text" value="Kanal 5"/>	Ch. 13	<input type="text" value="Kanal 13"/>
Ch. 6	<input type="text" value="Kanal 6"/>	Ch. 14	<input type="text" value="Kanal 14"/>
Ch. 7	<input type="text" value="Kanal 7"/>	Ch. 15	<input type="text" value="Kanal 15"/>

Set default channel name

Einstellung speichern

FormCfnSubmoduleIOnames (Ver. 1.01)

Sie können hier sämtliche Kanäle Ihres Haupt- oder Submoduls individuell benennen und speichern.

Hinweis:

Der Kanalname darf maximal 16 Zeichen lang sein.

Set default channel name

Schreibt den oben abgebildeten Text als Namensvorschlag in die Textfelder.
Zum Übernehmen auf das Modul muss zusätzlich gespeichert werden.

1.2.5.1.11. CAN Konfiguration

1.2.5.1.11.1. CAN Status

In diesem Bereich finden Sie alle Informationen rund um den Status des CAN-Interfaces und den TX- und RX-Paketen.

Hier werden Ihnen alle wichtigen Informationen zu Ihrem CAN-Interface angezeigt

Interface
Hier finden Sie Informationen über den Status des CAN-Interface.

CAN-Baudrate	1000000 Bit/sec
DT-CAN-CMD-MODE - Modul-Address	100 [hex]
DT-CAN-CMD-MODE - Response-Address	200 [hex]
CAN is active	1
Use Ext ID	1
CAN config mode selection	1

☐ Auto Refresh

FormCfoCANStatus (Ver. 1.00)

CAN-Baudrate

Zeigt die aktuell eingestellte Baud rate Ihres CAN-Interfaces an

DT-CAN-CMD-MODE- Module-Address

DT-CAN-CMD-MODE - Response-Address

CAN is active

(Diese Funktion wird nur dargestellt, wenn es von Ihrem Modul unterstützt wird)

Use EXT ID

CAN config mode selection

Hier werden alle wichtigen Statistiken zu den TX- und RX-Paketen, wie zum Beispiel die Anzahl der gesendeten und empfangenen CAN-Pakete und deren Übertragungsgeschwindigkeit angezeigt.

Statistik TX/RX

Hier finden Sie Informationen über die gesendeten und empfangenen CAN-Pakete.

DT-CAN-CMD-MODE

Frames total

TX0

RX0

TX-Modes	Frames total	Frames/Second	Ø Time/Frame	RX-Modes	Frames total
TX-1	0	0	-	RX-1	0
TX-2	0	0	-	RX-2	0
TX-3	0	0	-	RX-3	0
TX-4	0	0	-	RX-4	0
TX-5	0	0	-	RX-5	0
TX-6	0	0	-	RX-6	0
TX-7	0	0	-	RX-7	0
TX-8	0	0	-	RX-8	0

☒ Auto Refresh

FormCfoCANStatistic (Ver. 1.00)

1.2.5.1.11.2. CAN Main

In diesem Bereich können Sie Einstellungen direkt am CAN-Interface vornehmen.

Mit Hilfe dieser Einstellungen lässt sich das CAN Interface konfigurieren.

Interface

Hier können Sie Einstellungen am CAN-Interface vornehmen.

Baudrate	<input type="text" value="1 MBit/s"/>
Address Mode	<input type="text" value="11 Bit Mode"/>
DT-CAN-CMD-MODE - Modul-Address [hex]	<input type="text" value="FFFFFF"/>
DT-CAN-CMD-MODE - Response-Address [hex]	<input type="text" value="FFFFFF"/>

Einstellung speichern

FormCfaCANConfiaStd (Ver. 1.00)

Baudrate

Hier kann die Baudrate eingestellt werden, mit der das Modul kommunizieren soll.

Address Mode

Der Address Mode gibt vor, wie viel Bit zur Adressierung verwendet werden.

DT-CAN-CMD-MODE - Modul-Address[hex]

Diese Modul-Address legt fest, unter welcher Adresse das Modul im CAN-Bus identifiziert wird.

DT-CAN-CMD-MODE - Response-Address[hex]

Response-Address gibt vor, an welche Modul-Adresse eine Bestätigung gesendet wird, sobald ein Paket empfangen wurde.

Diese Einstellungen dienen der Konfiguration der angeschlossenen Submodule. Es kann der jeweilige Filter / Modus eingestellt werden, in dem die angeschlossenen Submodule gestartet werden.

I/O-Init

Hier können Sie die CAN-Einstellungen der angeschlossenen Submodule konfigurieren.

A/D Mode	16 Bit / $\pm 20V$
A/D Filter	None
D/A Mode	16 Bit / $\pm 10V$
Counter Mode	16 Bit Counter (default)
Timeout-Mode	Deactivate
Output Timeout	0.0 sec
CNT48 Mode	Read on rising edge (x1)
CNT48 Submode	Read
CNT48 Filter	20 ns

Einstellung speichern

FormCfaCANConfiaIO (Ver. 1.00)

A/D Mode

Der Wertebereich gibt den Bereich an, in dem analoge Signale, digital (z.B. im Bereich 0-10V) umgesetzt werden.

A/D Filter

Hier kann der A/D Filter eingestellt werden.

D/A Mode

Der Wertebereich gibt den Bereich an, in dem digitale Signale, analog (z.B. im Bereich 0-10V) umgesetzt werden.

Counter Mode

Ist zuständig für den Counter Modus. Wahlweise kann mit 16-Bit hochgezählt werden, oder mit je 8-Bit hoch- und herunter gezählt werden.

Timeout-Mode

Hier kann der Timeout-Mode eingestellt werden. Mehr Informationen zu den einzelnen Modi finden Sie im Kapitel: DapiSpecialCMDTimeout. Wird kein Timeout gewünscht kann dieser mit "Deactivate", deaktiviert werden.

Output Timeout

Gibt die Zeit vor, nach der die Ausgänge abschalten, wenn ein Modul nicht mehr erreicht werden kann.

CNT48 Mode

Stellt ein, welcher Counter Modus benutzt werden soll. Es stehen hierbei 6 verschiedene Modi zur Auswahl.

CNT48 Submode

In Abhängigkeit von dem unter "CNT48 Mode" gewählten Modus, stehen hier entsprechende Submodi zur Auswahl.

CNT48 Filter

Stellt den Filter ein, wie lange ein Signal mindestens sein muss, damit dieses als High erkannt wird. Es kann aus 16 vorgegebenen Filtern zwischen 20ns und 5ms ausgewählt werden.

1.2.5.1.11.3. CAN TX/RX Modi

Hier können Sie Einstellungen an den TX- und RX-Paketen vornehmen.

Hier können Sie Einstellungen an den TX-Paketen vornehmen.

TX-Mode[1]

Hier können Sie Einstellungen der TX-Paket-Konfiguration ändern.

Activate	<input checked="" type="checkbox"/>
Trigger Mode	Interval
Interval	1 * 1ms
Address Mode	11 Bit Mode
Send to CAN-ID [hex]	200
Data to send	OPTO-IN 1-64

Werkseinstellung laden

Einstellung speichern

FormCfCANConfiaTX (Ver. 1.00)

Activate

Aktiviert diesen TX-Mode

Trigger Mode

Gibt an mit welchem Modus das senden stattfinden soll. Zur Auswahl stehen die Modi "Interval" , "RX-Event" und "Fast as possible".

Intervall

Wird der Interval-Modus eingestellt, können sie zusätzlich angeben, in welchem Zeitintervall gesendet werden soll.

Address Mode

Gibt den Bit-Mode an welcher verwendet werden soll. Ausgewählt werden kann zwischen 11-Bit Mode und 29-Bit Mode

Send to CAN-ID[hex]

Sendet die CAN-Pakete an diese Adresse. In dem obigen Beispiel wird an Adresse 0x200 gesendet.

Data to send

Hier können Sie angeben, welche Daten an die zuvor eingestellte Adresse gesendet werden sollen.

Hier können Sie Einstellungen an den RX-Paketen vornehmen.

RX-Mode[1]

Hier können Sie Einstellungen der RX-Paket-Konfiguration ändern.

Activate	<input checked="" type="checkbox"/>
Address Mode	11 Bit Mode
Receive at CAN-ID [hex]	100
Data to send	Digital Out 1-64

Werkseinstellung laden

Einstellung speichern

FormCfoCANConfioRX (Ver. 1.00)

Activate

Aktiviert diesen RX-Mode

Address Mode

Hier kann der Address Mode 11-Bit oder 29-Bit eingestellt werden.

Receive at CAN-ID[hex]

Gibt die Empfängeradresse an. In den obigen Beispiel wird ein CAN-Paket auf der Adresse 0x100 empfangen.

Data to send

Wird ein Paket an der eingestellten Adresse empfangen, wird der Inhalt des Datenpaketes an die digitalen Ausgänge 1-64 weitergeleitet, woraufhin dort die Ausgänge ein- oder ausgeschaltet werden.

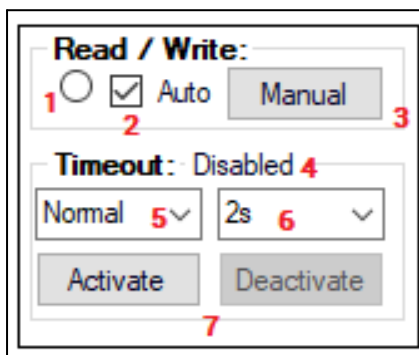
1.2.5.2. I/O-Test

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

1.2.5.2.1. 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.

1.2.5.2.2. 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.

1.2.5.2.3. 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		On / Off	Readback	Timer
Kanal 0	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 8	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 1	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 9	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 2	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 10	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 3	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 11	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 4	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 12	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 5	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 13	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 6	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 14	<input type="button" value="on"/> <input type="button" value="off"/>	0 s <input type="button" value="set"/>
Kanal 7	<input type="button" value="on"/> <input type="button" value="off"/>	<input type="checkbox"/>	0 s <input type="button" value="set"/>		Kanal 15	<input type="button" value="on"/> <input type="button" value="off"/>	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.

1.2.5.2.4. Analog In

Hier können Sie Einstellungen am A/D-Mode ändern und testen.

Analog In [0 .. 15]

Hier können Sie den A/D - Mode der Kanäle ändern

	Value		Value
Kanal 0	<input type="text" value="0,027"/> V	Kanal 8	<input type="text" value="0,027"/> V
Kanal 1	<input type="text" value="0,026"/> V	Kanal 9	<input type="text" value="0,025"/> V
Kanal 2	<input type="text" value="0,025"/> V	Kanal 10	<input type="text" value="0,024"/> V
Kanal 3	<input type="text" value="0,025"/> V	Kanal 11	<input type="text" value="0,025"/> V
Kanal 4	<input type="text" value="0,025"/> V	Kanal 12	<input type="text" value="0,024"/> V
Kanal 5	<input type="text" value="0,025"/> V	Kanal 13	<input type="text" value="0,024"/> V
Kanal 6	<input type="text" value="0,025"/> V	Kanal 14	<input type="text" value="0,025"/> V
Kanal 7	<input type="text" value="0,025"/> V	Kanal 15	<input type="text" value="0,024"/> V

Set mode for all channels

Mode Readback

Set filter for all channels

Filter Readback

☒ Auto Refresh

FormIOAnalogIn (Ver. 1.10)

Value

Liest den wert an dem jeweiligen A/D-Kanals aus.

Set mode for all channels

Auswahl des Spannungs- /Strombereichs für alle Kanäle, in dem gemessen werden soll. Es werden nur Modi angezeigt, die von Ihrem Modul unterstützt werden.

Mode Readback

Liest den aktuell verwendeten A/D-Mode aus dem Modul.

Set filter for all channels

Auswahl des anzuwendenden A/D-Filters für alle Kanäle.

Filter Readback

Liest den aktuell verwendeten A/D-Filter aus dem Modul.

1.2.5.2.5. Analog Out

In diesem Bereich können Sie Einstellungen an den D/A-Kanälen Ihres Moduls vornehmen und testen.

Analog Out [0 .. 13]

Hier können Sie den D/A - Mode der Kanäle ändern

	Value	Mode	Readback
Kanal 0	<input type="text" value="0"/>	16 Bit / 0-10V ▾	<input type="text" value="0"/> V
Kanal 1	<input type="text" value="1"/>	16 Bit / 0-10V ▾	<input type="text" value="0,999908"/> V
Kanal 2	<input type="text" value="2"/>	16 Bit / 0-10V ▾	<input type="text" value="1,999969"/> V
Kanal 3	<input type="text" value="3"/>	16 Bit / 0-10V ▾	<input type="text" value="2,999878"/> V
Kanal 4	<input type="text" value="4"/>	16 Bit / 0-10V ▾	<input type="text" value="3,999939"/> V
Kanal 5	<input type="text" value="5"/>	16 Bit / 0-10V ▾	<input type="text" value="5"/> V
Kanal 6	<input type="text" value="6"/>	16 Bit / 0-10V ▾	<input type="text" value="5,999908"/> V
Kanal 7	<input type="text" value="7"/>	16 Bit / 0-10V ▾	<input type="text" value="6,999969"/> V

	Value	Mode	Readback
Kanal 8	<input type="text" value="8"/>	16 Bit / 0-10V ▾	<input type="text" value="7,999878"/> V
Kanal 9	<input type="text" value="9"/>	16 Bit / 0-10V ▾	<input type="text" value="8,999939"/> V
Kanal 10	<input type="text" value="10"/>	16 Bit / 0-10V ▾	<input type="text" value="9,999847"/> V
Kanal 11	<input type="text" value="11"/>	16 Bit / 0-10V ▾	<input type="text" value="9,999847"/> V
Kanal 12	<input type="text" value="12"/>	16 Bit / 0-10V ▾	<input type="text" value="9,999847"/> V
Kanal 13	<input type="text" value="13"/>	16 Bit / 0-10V ▾	<input type="text" value="9,999847"/> V

Set mode for all channels

16 Bit / 0-10V ▾

Set

☒ Auto Refresh

FormIOAnalogOut (Ver. 1.10)

Value

Hier kann der Wert eingetragen werden, der an dem jeweiligen D/A-Kanal ausgegeben werden soll.

Mode

Auswahl des Spannungs-/ Strombereichs in dem der Wert des jeweiligen Kanals ausgegeben werden soll. Es können nur Modi ausgewählt werden, die vom Modul unterstützt werden.

Readback

Liest den Ist-Wert des jeweiligen D/A-Kanals zurück.

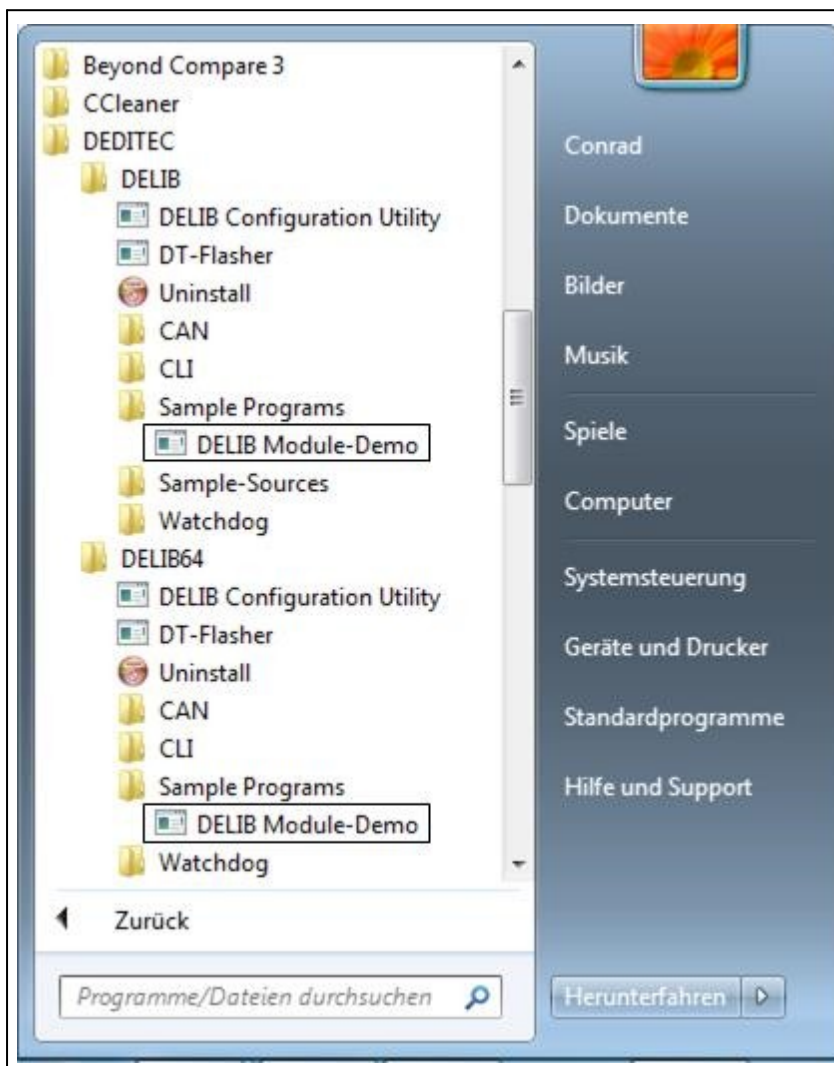
Set mode for all channels

Mit dieser Funktion lässt sich der Spannungs-/ Strombereich für alle verfügbaren Kanäle auf einmal ändern.

1.2.6. 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.

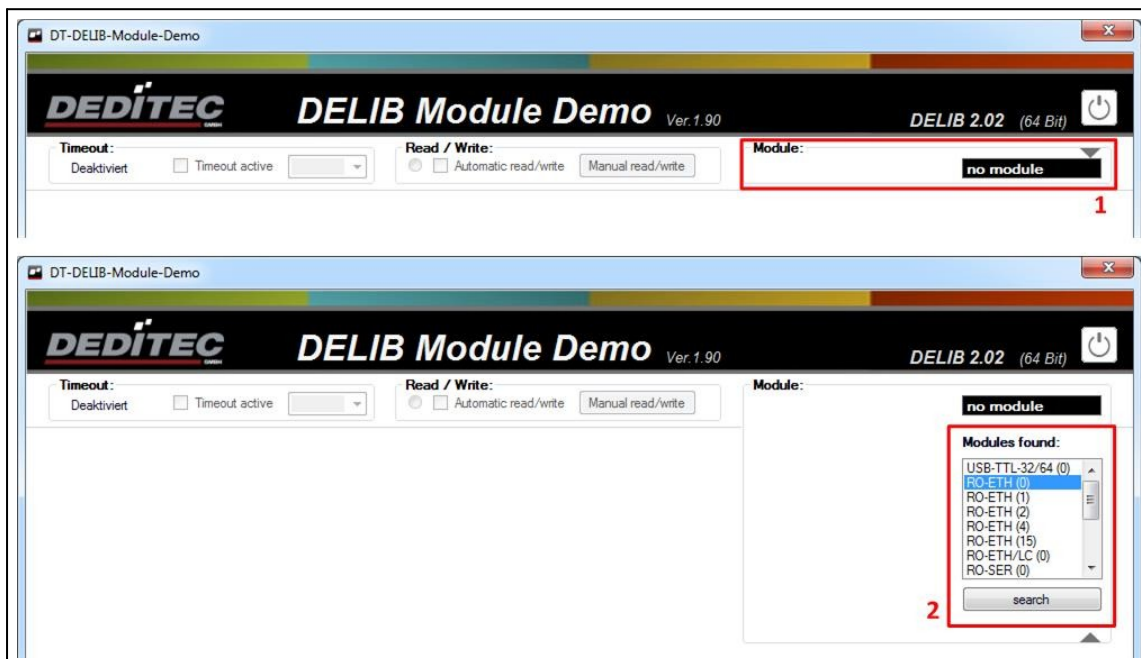


Das Programm DELIB Module Demo ist ein All-in-One Tool mit dem sämtliche I/Os aller Produkte aus unserem S&R Bereich gesteuert und getestet werden können.

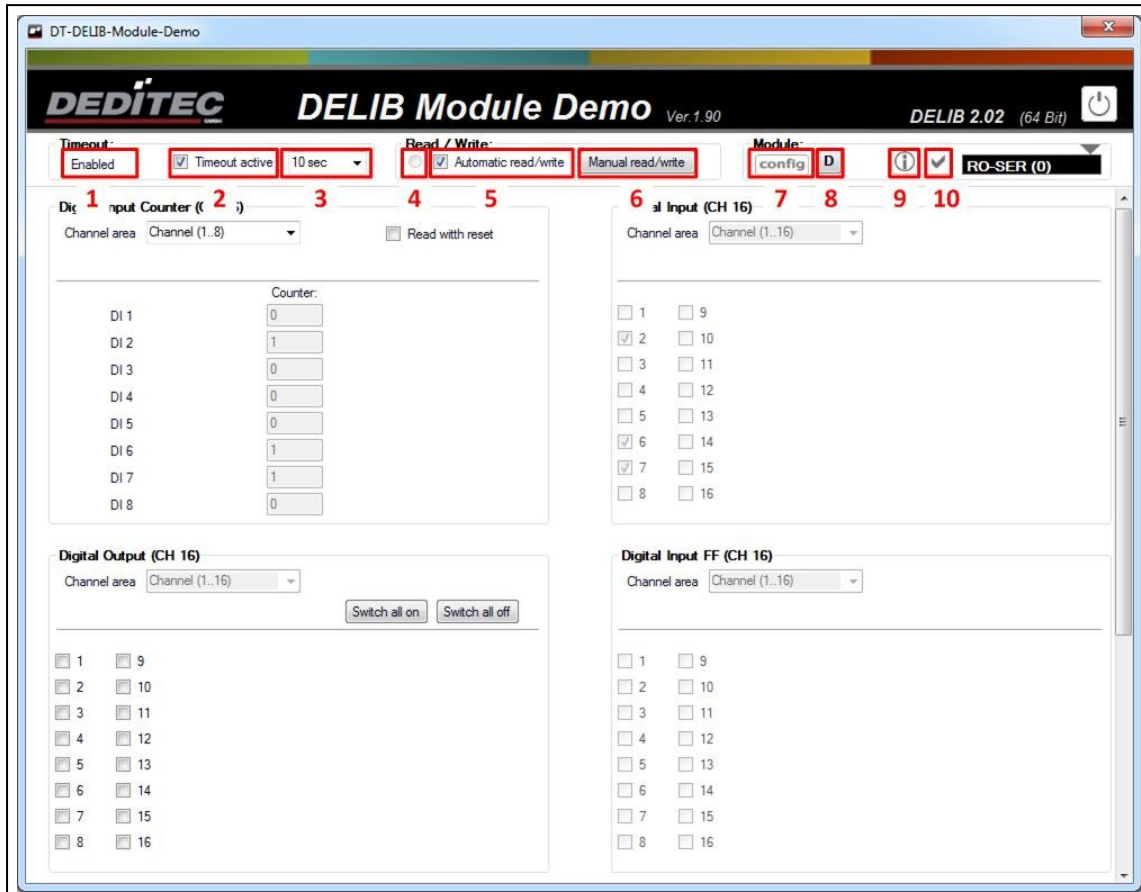
1.2.6.1. Auswahl des Moduls

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.



1.2.6.2. 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.

The screenshot displays the DT_ModuleInfo application window, which is divided into four main panels, each containing specific module information. The panels are labeled with red numbers 1 through 4 in the bottom right corner.

Panel 1: General

SW_FEATURE_1	e300000f
HW_INTERFACE_1	01000031
Firmware-Revision	2.01
Main-Module:	RO

Panel 2: Digital I/O, Analog I/O, and Special

Digital I/O	Analog I/O	Special
Digital Inputs	Analog Inputs	Stepper
Digital Outputs	Analog Outputs	
Digital In-/Outputs	Temperature Inputs	
Digital Input FlipFlops		
Digital Input Counter		
Pulse Gen Outputs		
CNT8		
Digital PWM Outputs		

Panel 3: Features-General

Supported by FW	OK
AD FIFO	OK
Set-Clr Bit Commands	OK
EEPROM Commands	OK
EEPROM E2_2K	-
DX1 Mode	-
Support Channel Names	-
HW-INT Supported by FV	OK
ETH	-
CAN	-
RS232	-
RS485	OK
USB	OK

Panel 4: Features-Digital I/O, Features-Analog I/O, and Features-Special

Features-Digital I/O	Features-Analog I/O	Features-Special
DI Commands	DA Commands	Watchdog Commands
DI CNT Commands	AD Commands	Stepper Commands
DI CNT Latch Feature	Pt100 Commands	
DI FF Commands		
DO Commands		
DO Time Commands		
PWM Commands		
TTL Commands		
PulseGen Commands		
CNT8 Commands		
Auto-Off Timeout Commar		

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.

1.2.6.3. Digital Input

The screenshot shows the 'DT-DELIB-Module-Demo' window. At the top, it displays 'DEDITEC DELIB Module Demo Ver. 1.90' and 'DELIB 2.02 (64 Bit)'. Below this, there are settings for 'Timeout' (Not Supported, Timeout active, 10 sec), 'Read / Write' (Automatic read/write, Manual read/write), and 'Module' (config, D, RO-SER (0)).

The main interface is divided into three sections:

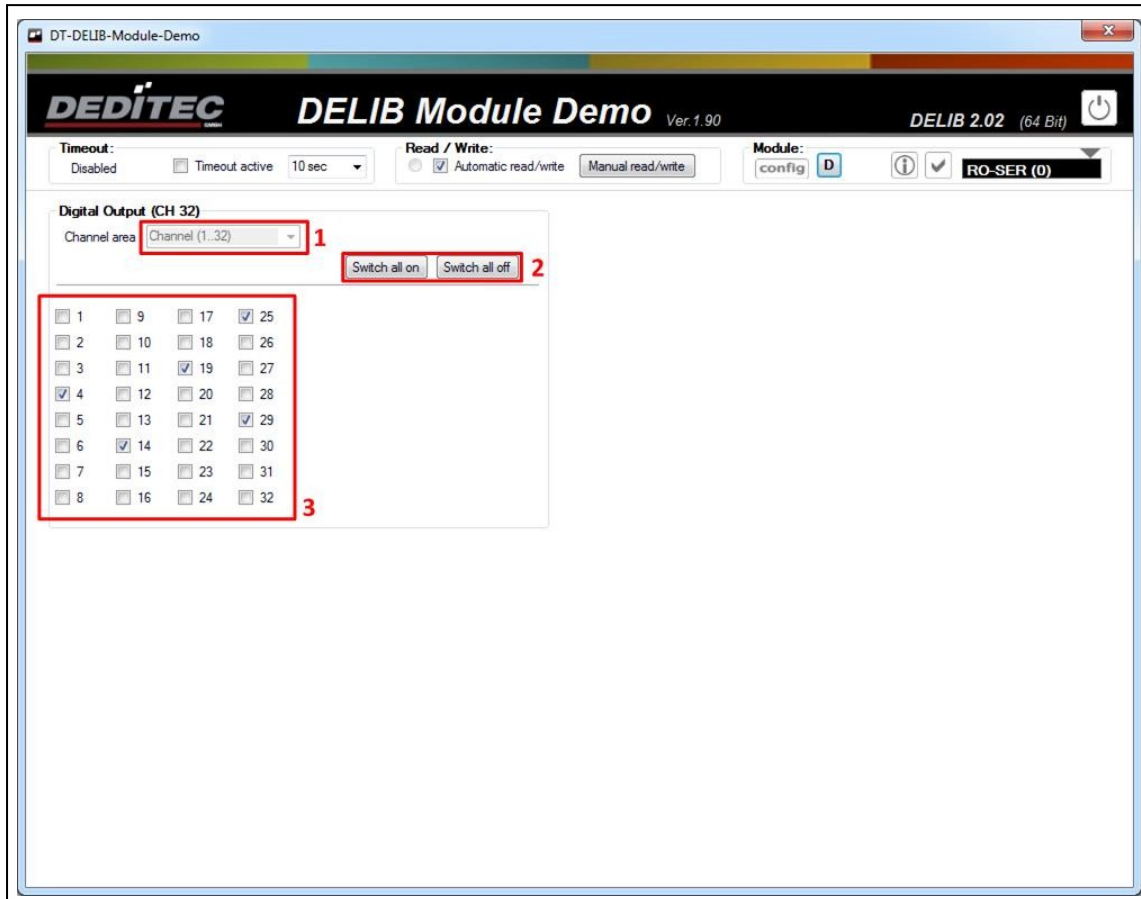
- Digital Input Counter (CH 16):** This section has a 'Channel area' dropdown set to 'Channel (1..8)' (labeled 1). A 'Read with reset' checkbox is checked (labeled 2). Below this is a table of counters for DI 1 through DI 8 (labeled 3):

DI	Counter
DI 1	1
DI 2	2
DI 3	1
DI 4	0
DI 5	0
DI 6	2
DI 7	1
DI 8	0
- Digital Input (CH 16):** This section has a 'Channel area' dropdown set to 'Channel (1..16)'. It displays a grid of checkboxes for inputs 1 through 16 (labeled 4). Inputs 1, 2, 3, 6, and 8 are checked.
- Digital Input FF (CH 16):** This section has a 'Channel area' dropdown set to 'Channel (1..16)'. It displays a grid of checkboxes for inputs 1 through 16 (labeled 5). Inputs 3 and 6 are checked.

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).

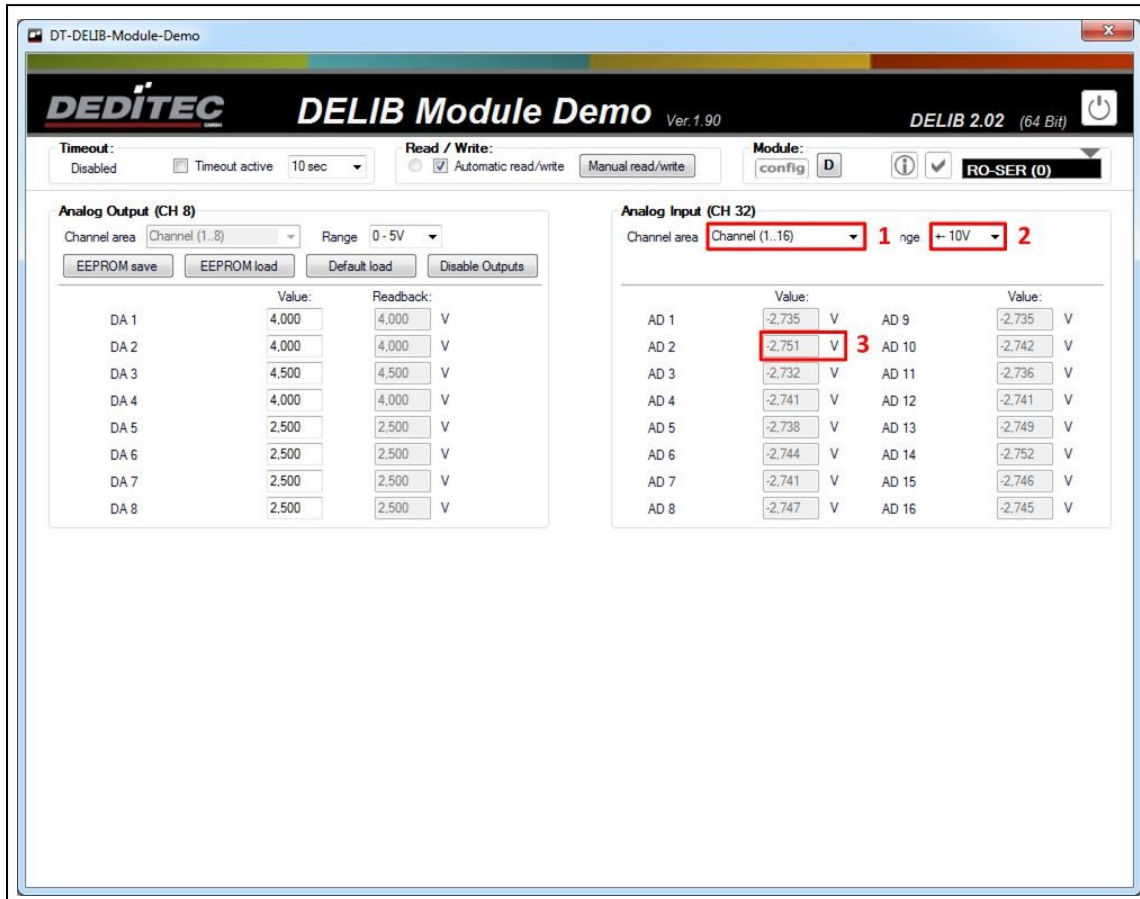
1.2.6.4. Digital Output



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.

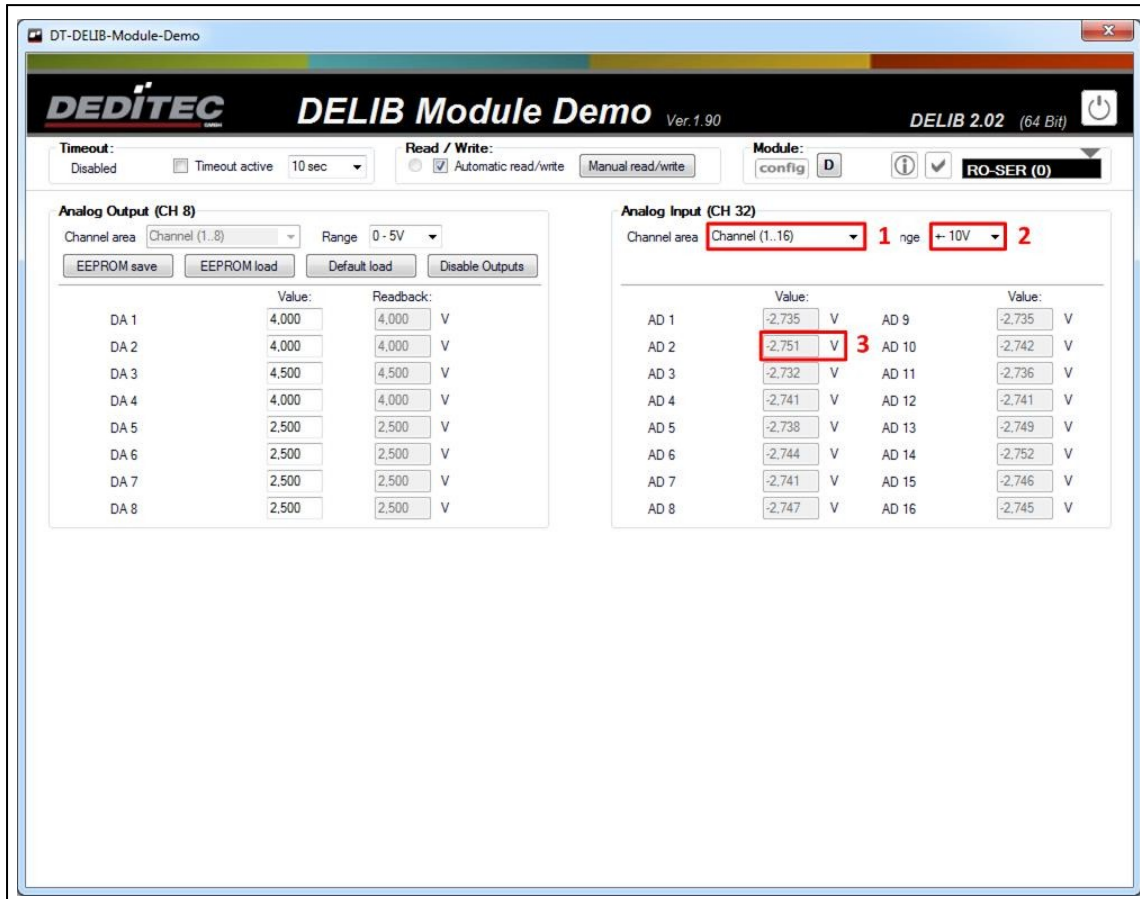
1.2.6.5. Analog Input



Dieses Beispiel zeigt die analogen Eingänge eines RO-SER-AD32-DA8 Moduls.

1. Auswahl des Kanal-Bereichs, der angezeigt werden soll.
2. Auswahl des Spannungs-/Strombereichs in dem gemessen werden sollen. Wird der Modus nicht unterstützt, erscheint die Meldung "illegal" rechts neben dem Drop-Down Menü.
3. Aktueller A/D-Wert an A/D-Kanal 2.

1.2.6.6. Analog Output



Dieses Beispiel zeigt die analogen Eingänge eines RO-SER-AD32-DA8 Moduls.

1. Auswahl des Kanal-Bereichs, der angezeigt werden soll.
2. Auswahl des Spannungs-/Strombereichs in dem gemessen werden sollen. Wird der Modus nicht unterstützt, erscheint die Meldung "illegal" rechts neben dem Drop-Down Menü.
3. Aktueller A/D-Wert an A/D-Kanal 2.

1.2.7. CAN Configuration Utility

Hinweis:

Um ein CAN-Modul konfigurieren zu können, muss dieses zunächst in den Software-Modus gebracht werden.

[Konfiguration Produkte der RO-Serie](#)

[Konfiguration Produkte der BS-Serie](#)

Das CAN-Configuration-Utility ermöglicht eine einfache Konfiguration von Produkten mit einer CAN-Schnittstelle. Es ist möglich Konfigurationen auf Module zu übertragen und auszulesen.

Zusätzlich kann der automatische Empfangsmodus (Auto-RX) und der automatische Sendemodus (Auto-TX) konfiguriert werden.

Der Auto-TX Modus erlaubt ein zyklisches Senden von Datenpaketen, wahlweise mit analogen oder digitalen Eingangszuständen an andere CAN-Adressen. Alternativ kann auch ein Trigger-Event definiert werden. Hierbei wird ein Datenpaket erst dann gesendet, wenn zuvor ein Datenpaket auf einer gewissen CAN-ID empfangen wurde (z.B. CAN-Sync auf ID 0x80).

Mit dem Auto-RX Modus hingegen werden empfangene Datenpakete direkt an analoge oder digitale Ausgänge weitergeleitet. So können beispielsweise Relais-Ausgänge über einen anderen CAN-Bus Teilnehmer gesetzt werden.

Folgende Einstellungen können verändert werden:

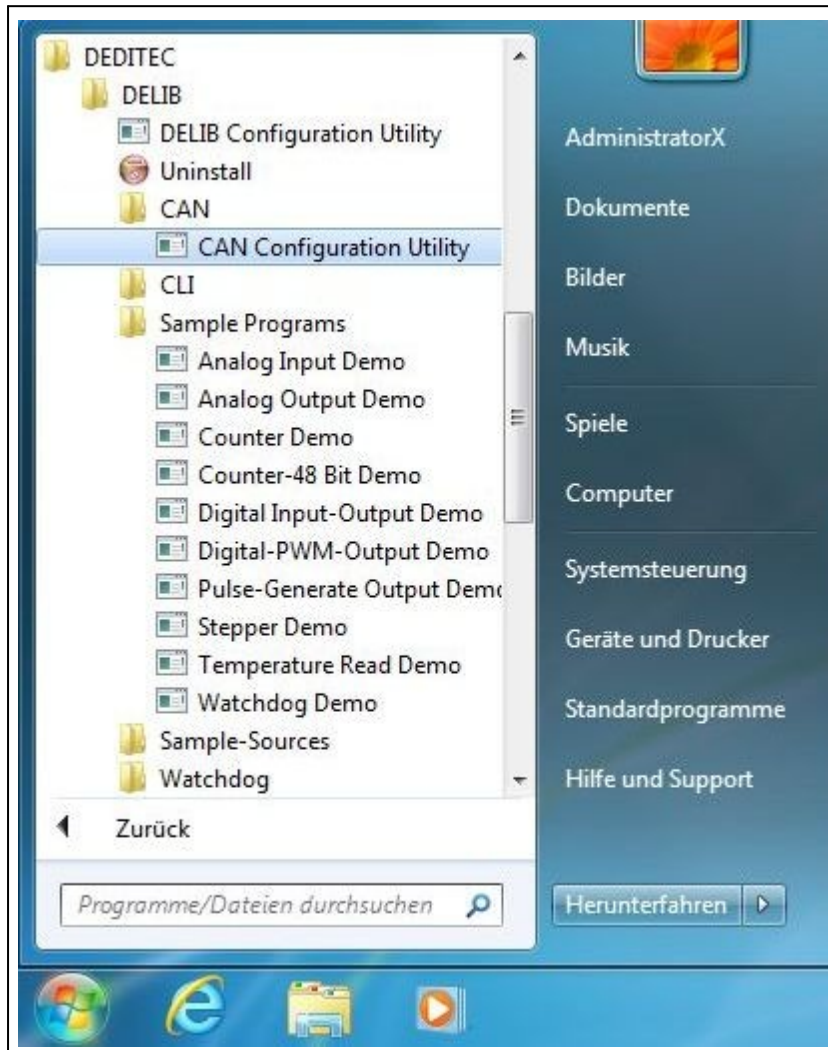
- Baudrate
- Adress-Modus
- Modul-ID
- Response-ID
- Modi, mit denen Submodule gestartet werden
- Automatischer Sendemodus (TX-Modus)
- Automatischer Empfangsmodus (RX-Modus)

Auf den nachfolgenden Seiten wird Schritt für Schritt gezeigt, wie ein CAN-Modul konfiguriert werden kann.

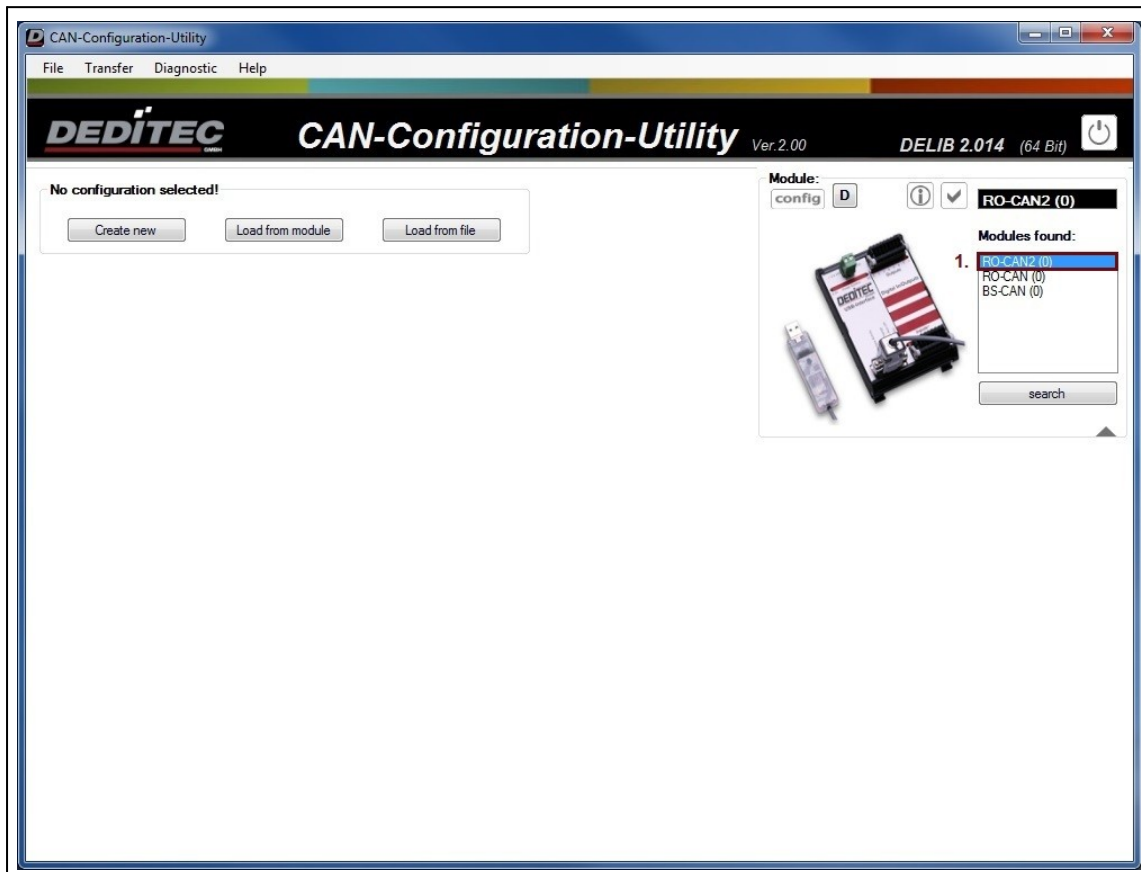
1.2.7.1. Auswahl des Moduls

Starten Sie das CAN-Configuration-Utility über:

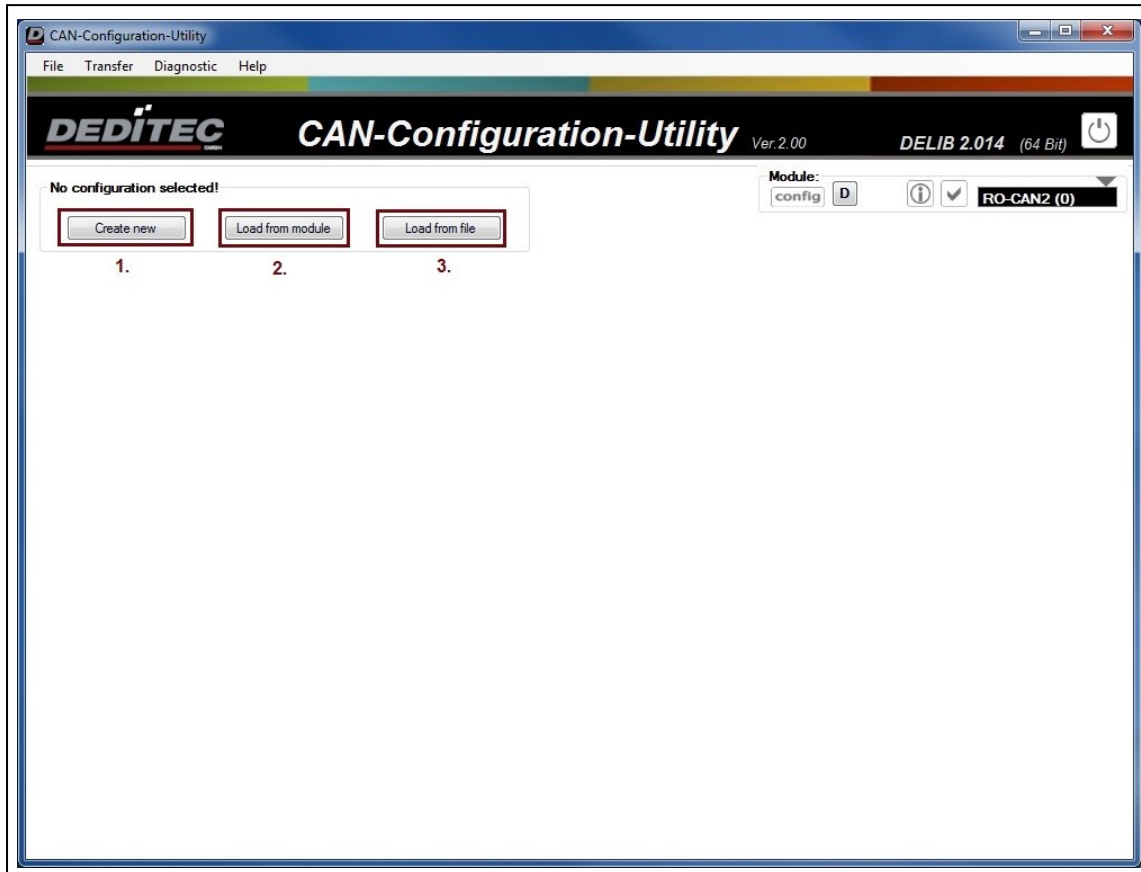
Start → Programme → DEDITEC → DELIB → CAN



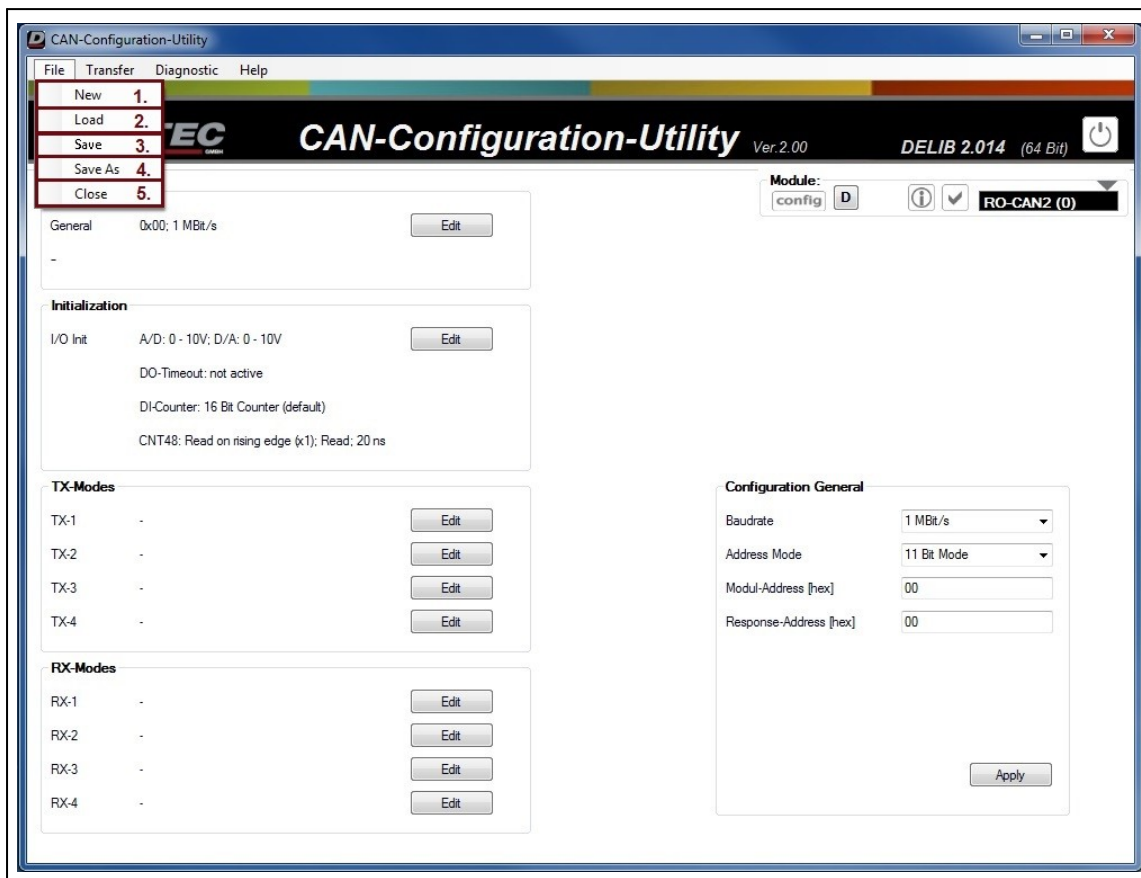
1. In der "Module-Selection" ein entsprechendes CAN-Modul (z.B. das RO-CAN2) auswählen.



1.2.7.2. Neue Konfiguration Erstellen, Laden, Speichern



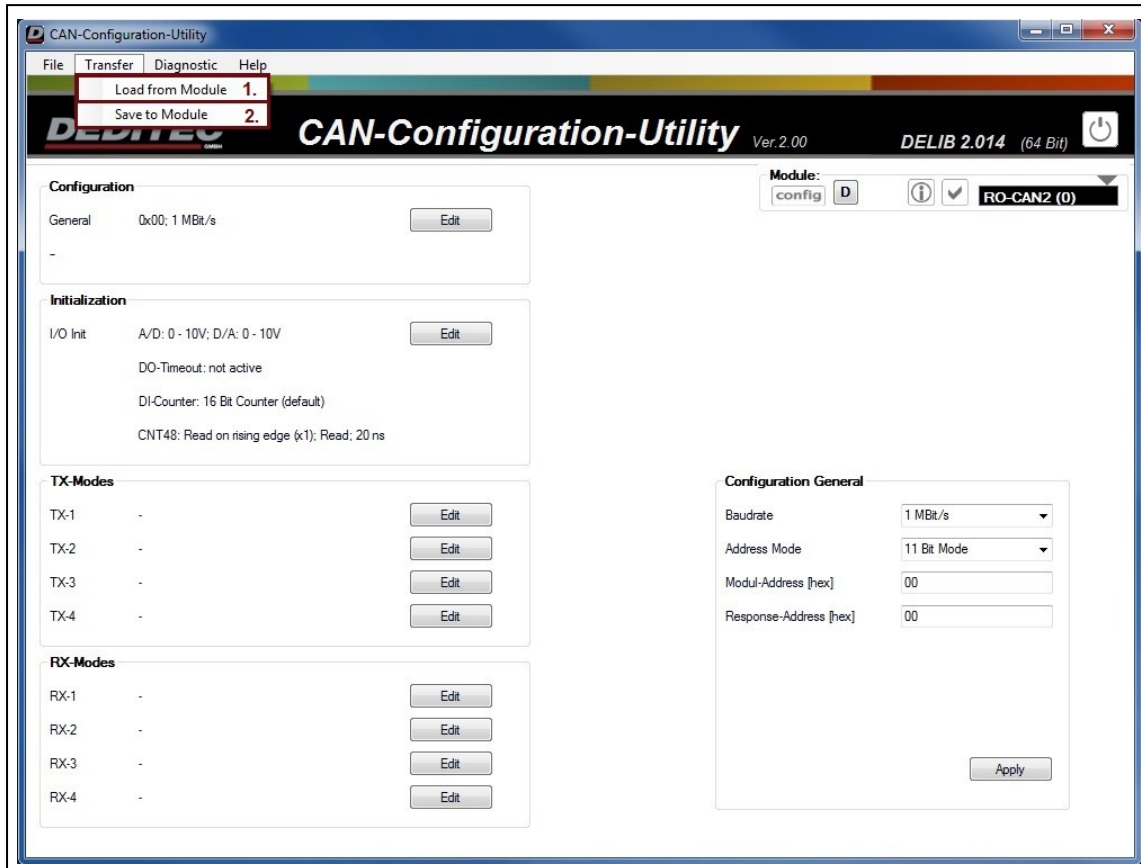
1. Über "Create new" kann eine neue CAN-Konfiguration erstellt werden.
2. Mit "Load from Module" kann die aktuell auf dem Modul vorhandene Konfiguration ausgelesen werden.
3. "Load from File" erlaubt es, zuvor auf einem Datenträger gespeicherte CAN-Konfigurationsdateien zu öffnen.



Mit einem Klick auf "File" in der Menüleiste öffnet sich ein Untermenü:

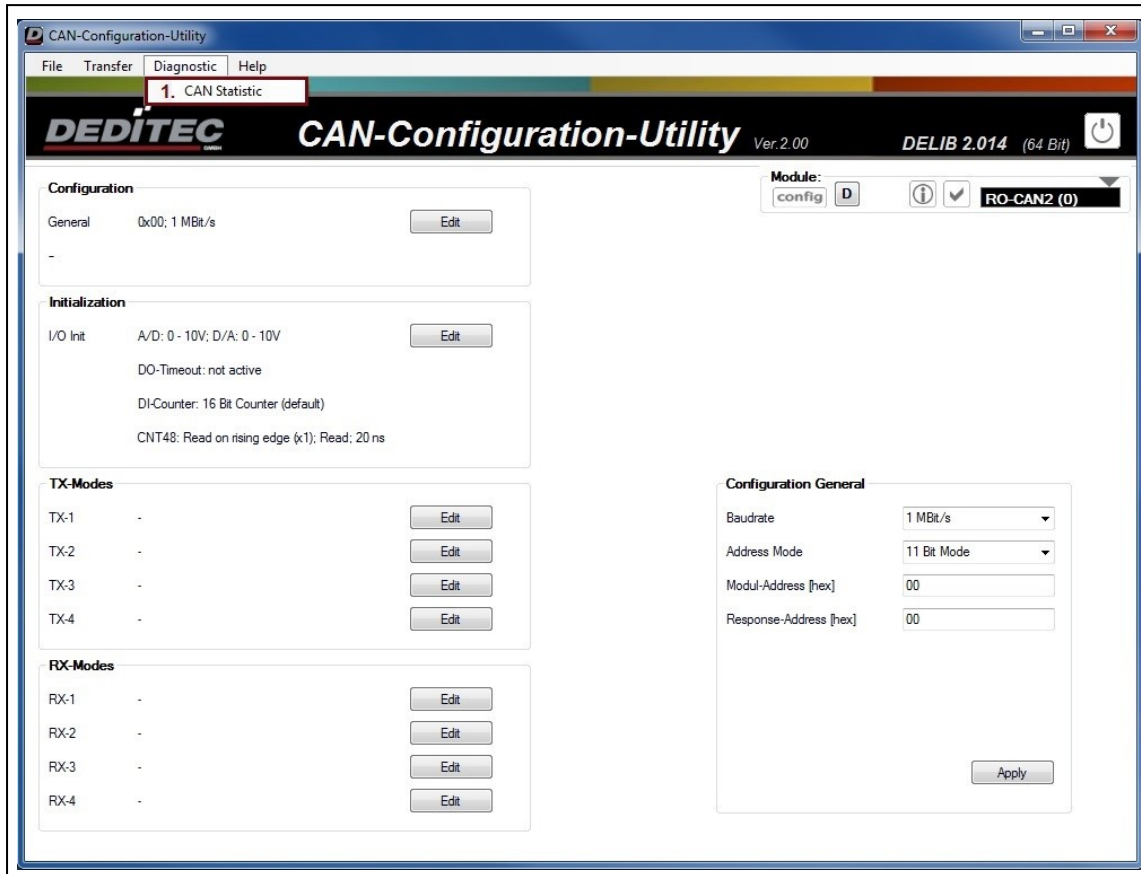
1. Über den Menüpunkt "New" kann eine neue CAN-Konfiguration erstellt werden.
2. "Load" bietet die Möglichkeit, zuvor gespeicherte CAN-Konfigurationsdateien zu öffnen.
3. Wurde eine CAN-Konfigurationsdatei geöffnet, so kann über "Save" die Datei überschrieben und aktuelle Änderungen gespeichert werden.
4. Durch einen Klick auf "Save as" kann die CAN-Konfiguration in einer neuen Datei abgespeichert werden.
5. Hier kann das CAN-Configuration-Utility beendet werden.

1.2.7.3. Konfiguration auf das Modul übertragen



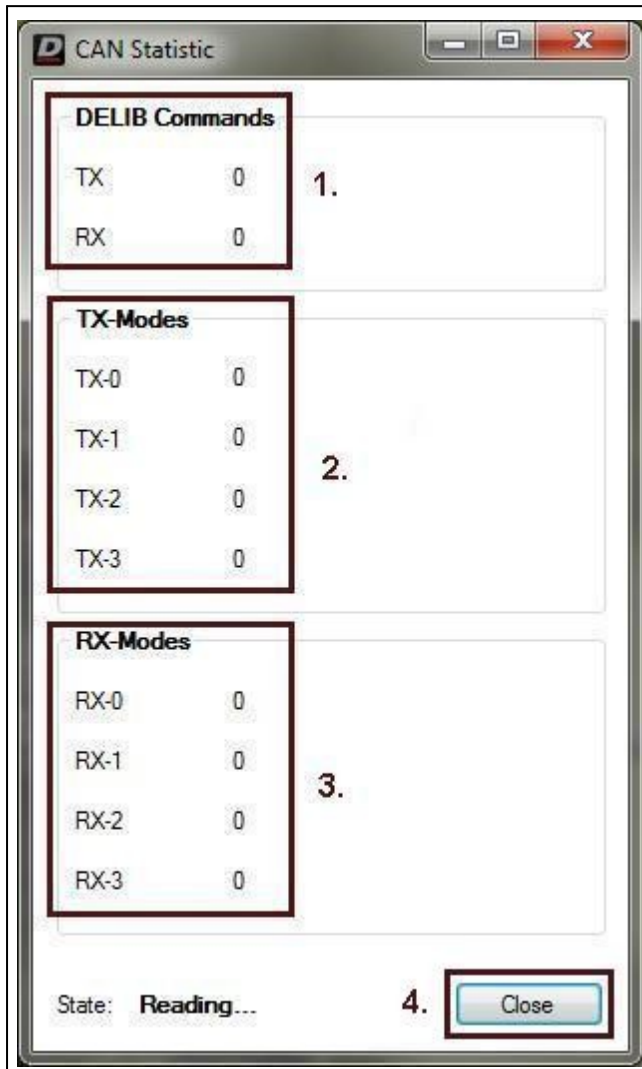
1. Über "Load from Module" kann die aktuelle Konfiguration des ausgewählten Moduls ausgelesen werden.
2. "Save to Module" überträgt die aktuelle CAN-Konfiguration auf das ausgewählte Modul.

1.2.7.4. Statistiken vom Modul abfragen



1. Über den Menüpunkt CAN-Statistic kann ein Informationsfenster angezeigt werden, welches die Anzahl der gesendeten und empfangenen TX- bzw. RX-Pakete darstellt. Ebenfalls wird die Anzahl der gesendeten und empfangenen DELIB Kommandos dargestellt.

Folgendes Fenster zeigt die Statistiken an:



1. Dieser Bereich zeigt die Anzahl der gesendeten und empfangenen DELIB Kommandos an.
2. Für den TX-Modus wird hier die Anzahl der gesendeten Pakete aufgelistet.
3. Die empfangenen RX-Pakete werden hier dargestellt.
4. Über "Close" kann das Fenster geschlossen werden.

Hinweis

Die CAN-Statistik zählt im Hintergrund weiter, auch wenn das Fenster geschlossen ist. Bei einem Neustart des Moduls wird die Statistik auf null zurückgesetzt.

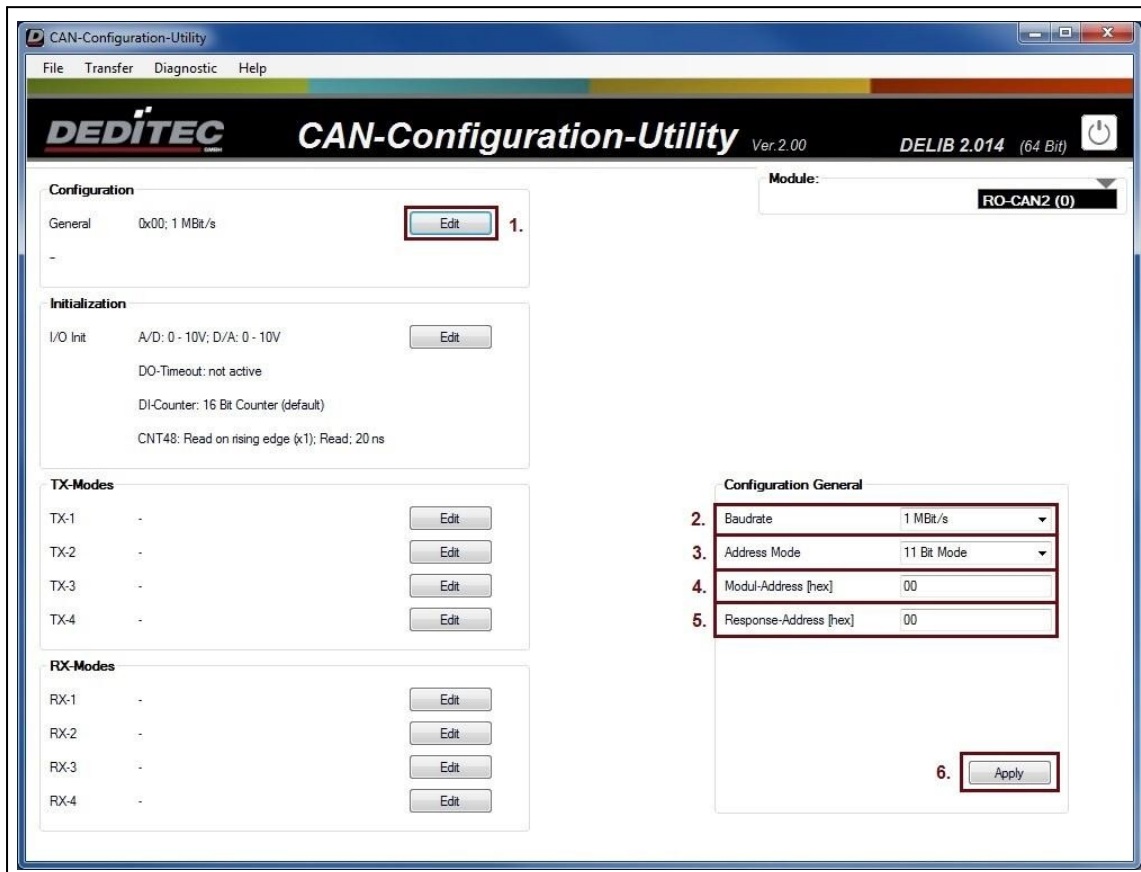
1.2.7.5. Konfiguration

Für jedes CAN-Modul können 4 unterschiedliche Konfigurationen für TX- und RX-Pakete angelegt werden.

Ebenfalls kann definiert werden, in welchem Modus Submodule gestartet werden. So ist es z.B. möglich, dass ein AD-Modul in einem Messbereich von 0-5V und nicht im Standard Messbereich von $\pm 10V$ gestartet wird.

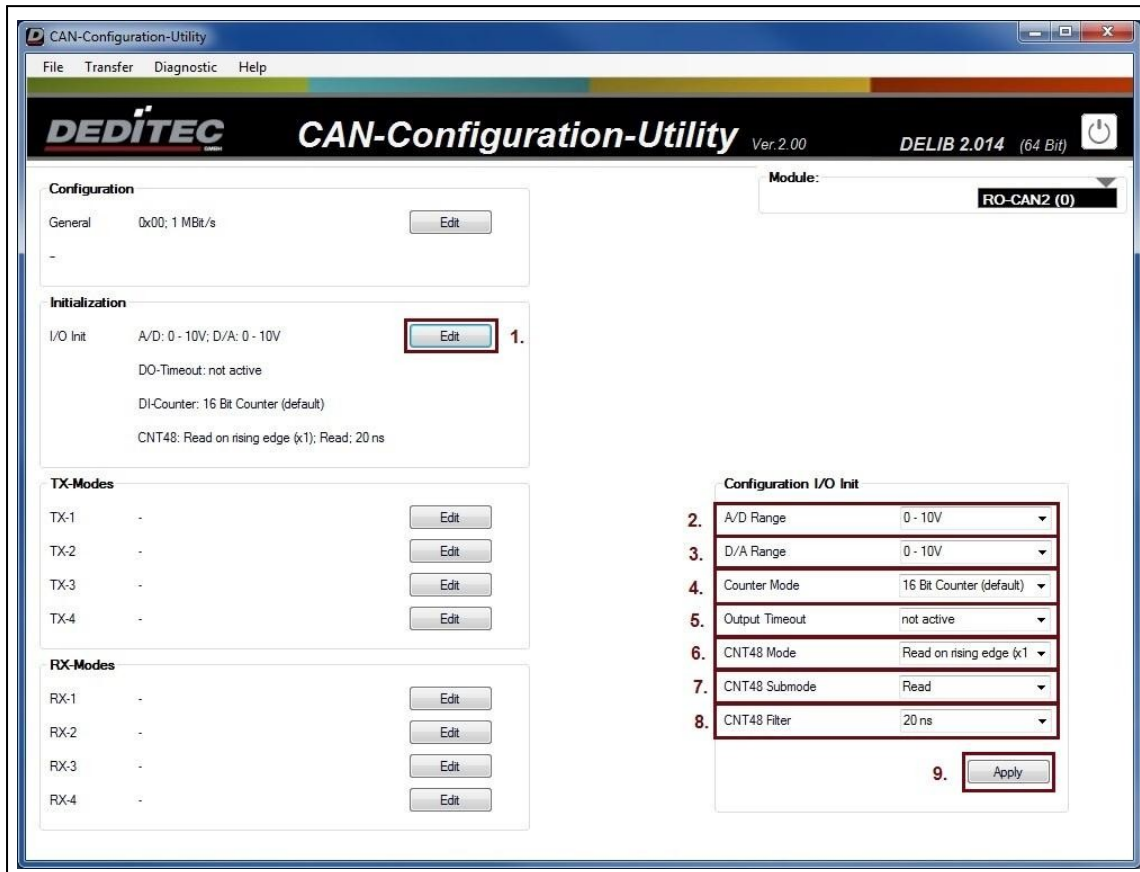
Auf den nachfolgenden Seiten wird gezeigt, wie die verschiedenen Modi konfiguriert werden.

1.2.7.5.1. Modul Konfiguration



1. Über den Menüpunkt "Edit" wird ein Zusatzfenster geöffnet, in dem die Konfiguration vorgenommen werden kann.
2. Hier kann die Baudrate eingestellt werden, mit der das Modul kommunizieren soll.
3. Der Address Mode gibt vor, wie viel Bit zur Adressierung verwendet werden.
4. Die Modul-Address legt fest, unter welcher Adresse das Modul im CAN-Bus identifiziert wird.
5. Die Response-Address gibt vor, an welche Module-Adresse eine Bestätigung gesendet wird, sobald ein Paket empfangen wurde.
6. Über "Apply" werden die Änderungen übernommen.

1.2.7.5.2. I/O Konfiguration



Diese Einstellungen dienen der Konfiguration der angeschlossenen Submodule. Es kann der jeweilige Filter / Modus eingestellt werden, in dem die angeschlossenen Submodule gestartet werden.

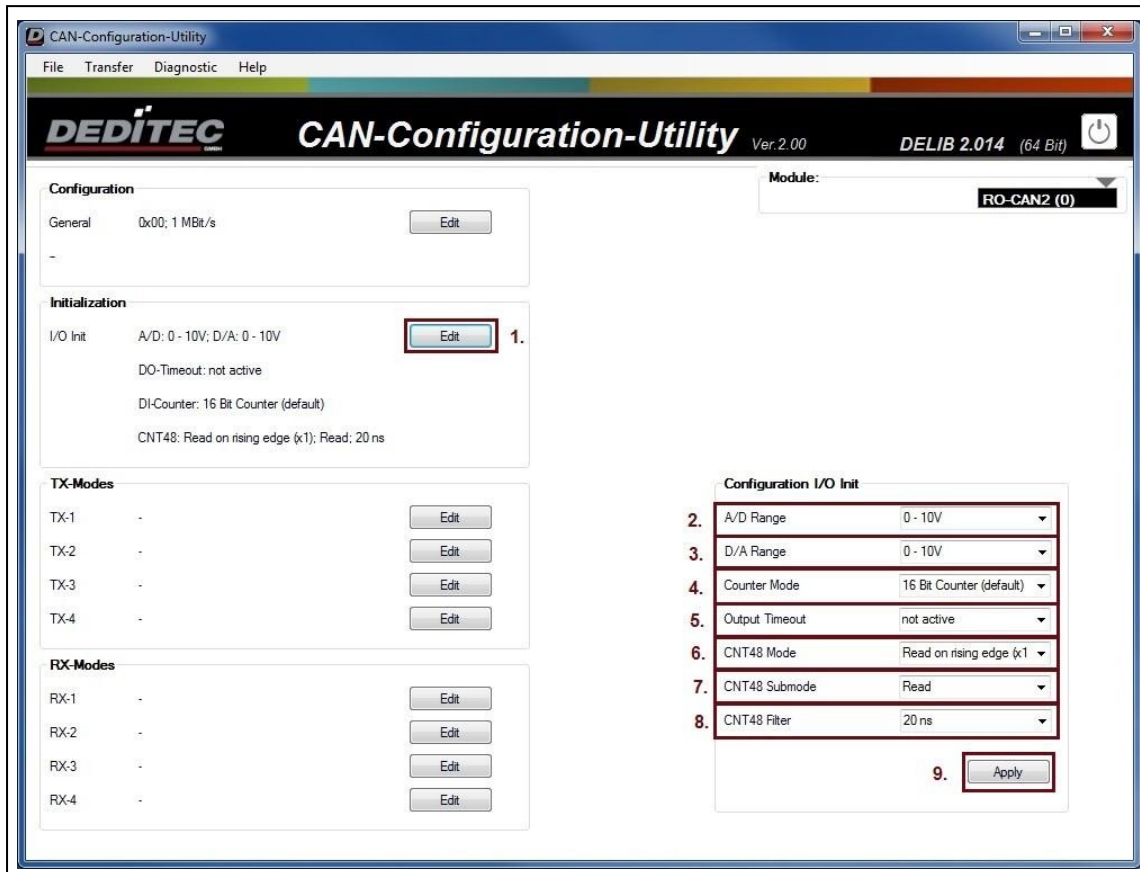
Hinweis:

Sind die entsprechenden Submodule nicht vorhanden, so haben die Einstellungen keine Auswirkung.

1. Über den Menüpunkt "Edit" wird ein Zusatzfenster geöffnet, in dem die Konfiguration vorgenommen werden kann.
2. Der Wertebereich gibt den Bereich an, in dem analoge Signale, digital (z.B. im Bereich 0-10V) umgesetzt werden.
3. Der Wertebereich gibt den Bereich an, in dem digitale Signale, analog (z.B. im Bereich 0-10V) umgesetzt werden.
4. Ist zuständig für den Counter Modus von O8-R8 Modulen. Wahlweise kann

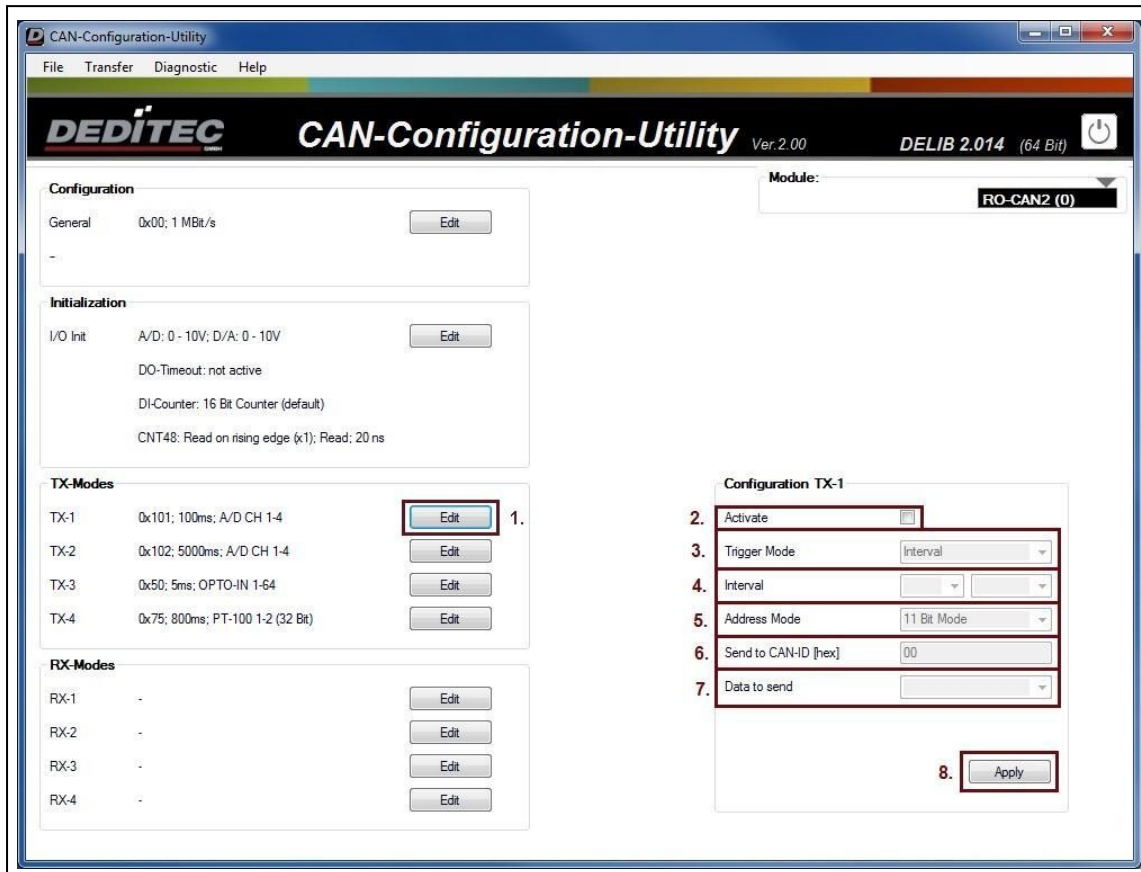
mit 16-Bit hochgezählt werden, oder mit je 8-Bit hoch- und heruntergezählt werden.

5. Gibt die Zeit vor, nach der die Ausgänge abschalten, wenn ein Modul nicht mehr erreicht werden kann. Wird kein Timeout erwünscht wählen Sie bitte die Einstellung "not active".



6. Stellt ein, welcher Counter Modus (nur für RO-CNT8 Module) benutzt werden soll. Es stehen hierbei 5 verschiedene Modi zur Auswahl. Eine genaue Beschreibung der Modi ist im Handbuch "RO-CNT8" zu finden.
7. In Abhängigkeit von dem unter 6. gewählten Modus, stehen hier entsprechende Submodi zur Auswahl. Eine genaue Beschreibung der Submodi ist im Handbuch "RO-CNT8" zu finden.
8. Stellt den Filter ein, wie lange ein Signal mindestens sein muss, damit dieses als High erkannt wird. Es kann aus 16 vorgegebenen Filtern zwischen 20ns und 5ms ausgewählt werden (nur für RO-CNT8 Module).
9. Über "Apply" werden die Änderungen übernommen.

1.2.7.5.3. TX-Konfiguration



Es können bis zu 4 unabhängige TX-Modi eingestellt werden. Die Konfiguration ist für alle Modi identisch. Das folgende Beispiel zeigt die Konfiguration für den ersten TX-Modus.

1. Über den Menüpunkt "Edit" wird ein Zusatzfenster geöffnet, in dem die Konfiguration vorgenommen werden kann.
2. Um die Konfiguration zu aktivieren, muss hier der Haken gesetzt werden.
3. Der Trigger Mode gibt vor, unter welcher Bedingung Daten gesendet werden. Entweder im Intervall oder in Abhängigkeit eines empfangenen RX-Paketes.
4. Hier wird das Intervall konfiguriert (wenn unter Punkt 3. "Intervall" ausgewählt), in dem Pakete versendet werden.
5. Der Address Mode gibt vor, wie viel Bit zur Adressierung verwendet werden.
6. Hier wird festgelegt, an welche Moduladresse CAN-Pakete gesendet werden.
7. Unter diesem Punkt wird definiert, welche Art von Daten an die unter Punkt 6. konfigurierte Adresse gesendet werden. Wie z.B. → der Status der digitalen Eingänge oder → die Spannung von A/D Kanälen.
8. Über "Apply" werden die Änderungen übernommen.

1.2.7.5.3.1. Beispiel Interval

Configuration TX-1

Activate	<input checked="" type="checkbox"/>
Trigger Mode	Interval
Interval	1 * 1sec
Address Mode	11 Bit Mode
Send to CAN-ID [hex]	0x100
Data to send	OPTO-IN 1-64

Apply

Das Beispiel beinhaltet folgende Einstellungen:

Im Intervall von einer Sekunde werden die Daten der digitalen Eingänge 1-64 an die CAN-Adresse 0x100 gesendet.

1.2.7.5.3.2. Beispiel Trigger

Configuration RX-1

Activate

☒

Address Mode

11 Bit Mode

▼

Receive at CAN-ID [hex]

200

Data to send

Trigger Auto TX 1

▼

Apply

Configuration TX-1

Activate

☒

Trigger Mode

RX-Event

▼

Interval

▼

▼

Address Mode

11 Bit Mode

▼

Send to CAN-ID [hex]

100

Data to send

A/D CH 1-4

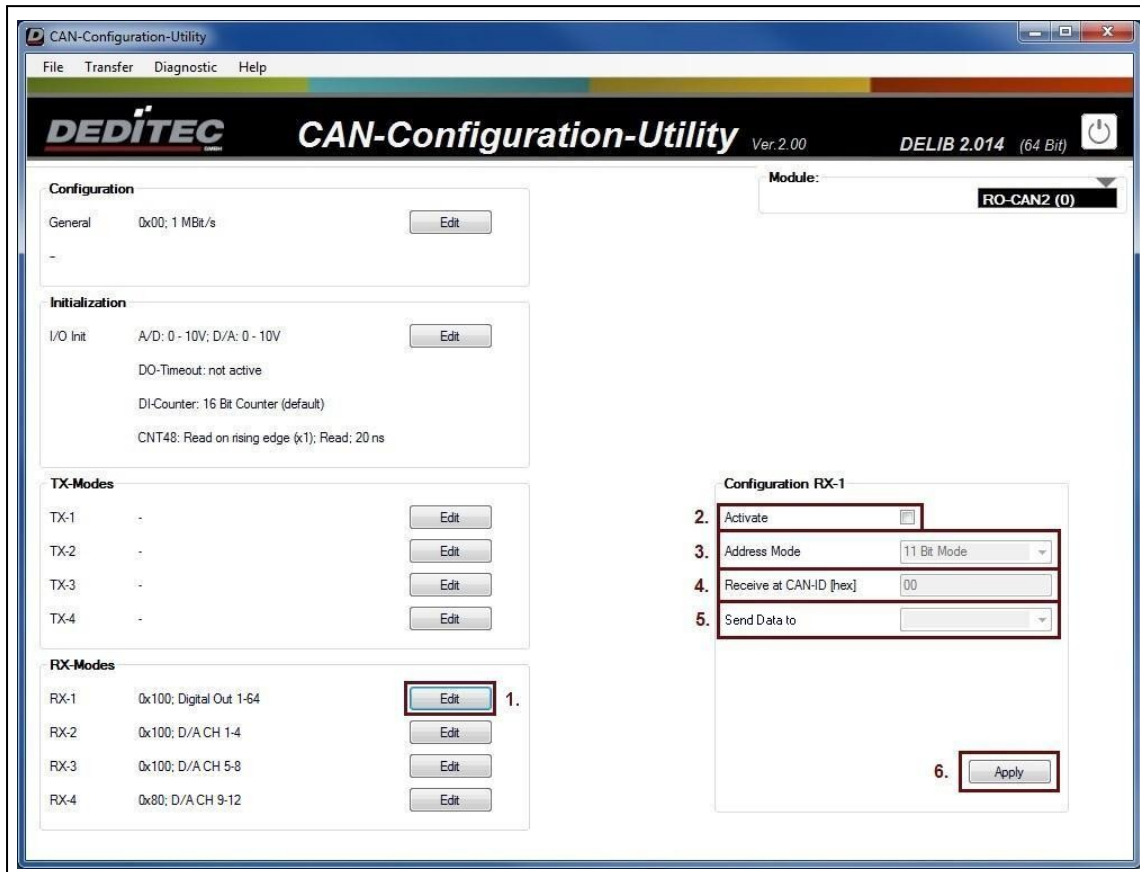
▼

Apply

Das Beispiel beinhaltet folgende Einstellungen:

Werden Daten auf der CAN-Adresse 0x200 empfangen, wird TX-1 ausgeführt (Bild oben), welches die Daten der A/D Kanäle 1-4 an die CAN-Adresse 0x100 versendet (Bild unten).

1.2.7.5.4. RX-Konfiguration



Es können bis zu 4 unabhängige RX-Modi eingestellt werden. Die Konfiguration ist für alle Modi identisch. Das folgende Beispiel zeigt die Konfiguration für den ersten RX-Modus.

1. Über den Menüpunkt "Edit" wird ein Zusatzfenster geöffnet, in dem die Konfiguration vorgenommen werden kann.
2. Um die Konfiguration zu aktivieren muss hier der Haken gesetzt werden.
3. Der Address Mode gibt vor, wie viel Bit zur Adressierung verwendet werden.
4. Hier wird festgelegt, auf welcher CAN-ID die Datenpakete erwartet werden.
5. Wurden auf der unter Punkt 4. konfigurierten ID Daten empfangen, so wird hier definiert, welche Aktion vom Modul ausgeführt werden soll. Wie z.B. → automatisch die Ausgangsspannung bei einem analogen Ausgangsmodul setzen oder → Relais-Ausgänge schalten.
6. Über "Apply" werden die Änderungen übernommen.

1.2.7.5.4.1. Beispiel RX-DA

Configuration RX-1

Activate	<input checked="" type="checkbox"/>
Address Mode	11 Bit Mode
Receive at CAN-ID [hex]	201
Data to send	D/A CH 5-8

Apply

Das Beispiel beinhaltet folgende Einstellungen:

Wurde ein CAN-Paket auf der Adresse 0x201 empfangen, wird der Inhalt des Datenpaketes an den analogen Ausgängen 5-8, unter Berücksichtigung des ausgewählten D/A-Modus gesetzt.

1.2.7.5.4.2. Beispiel RX-DO

Configuration RX-1

Activate ☒

Address Mode 11 Bit Mode ▼

Receive at CAN-ID [hex] 100

Data to send Digital Out 1-64 ▼

Apply

Das Beispiel beinhaltet folgende Einstellungen:

Wurde ein CAN-Paket auf der Adresse 0x100 empfangen, wird der Inhalt des Datenpaketes an die digitalen Ausgänge 1-64 weitergeleitet, woraufhin dort die Ausgänge ein- oder ausgeschaltet werden.

1.2.7.6. Aufbau der CAN-Pakete

Die folgenden Seiten zeigen, wie die CAN-Pakete der unterschiedlichen I/O-Module aufgebaut sind. Ebenfalls wird dargestellt, wie A/D- und D/A-Werte berechnet werden können.

Die Daten der CAN-Pakete sind jeweils 8 Byte groß. Welche Daten an welcher Stelle zu finden sind, entnehmen Sie bitte den folgenden Seiten.

1.2.7.6.1. Digitale Eingänge

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	DI channel 1-8 (Bit 0-7)
2	DI channel 9-16 (Bit 0-7)
3	DI channel 17-24 (Bit 0-7)
4	DI channel 25-32 (Bit 0-7)
5	DI channel 33-40 (Bit 0-7)
6	DI channel 41-48 (Bit 0-7)
7	DI channel 49-56 (Bit 0-7)
8	DI channel 57-64 (Bit 0-7)

1.2.7.6.2. Digitale Ausgänge

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	DO channel 1-8 (Bit 0-7)
2	DO channel 9-16 (Bit 0-7)
3	DO channel 17-24 (Bit 0-7)
4	DO channel 25-32 (Bit 0-7)
5	DO channel 33-40 (Bit 0-7)
6	DO channel 41-48 (Bit 0-7)
7	DO channel 49-56 (Bit 0-7)
8	DO channel 57-64 (Bit 0-7)

1.2.7.6.3. Digitale Eingangszähler (16-Bit)

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	DI counter 1 (Bit 0-7)
2	DI counter 1 (Bit 8-15)
3	DI counter 2 (Bit 0-7)
4	DI counter 2 (Bit 8-15)
5	DI counter 3 (Bit 0-7)
6	DI counter 3 (Bit 8-15)
7	DI counter 4 (Bit 0-7)
8	DI counter 4 (Bit 8-15)

1.2.7.6.4. Digitale Eingangszähler (48-Bit) - 32-Bit Paket

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	CNT8 counter 1 (Bit 0-7)
2	CNT8 counter 1 (Bit 8-15)
3	CNT8 counter 1 (Bit 16-23)
4	CNT8 counter 1 (Bit 24-31)
5	CNT8 counter 2 (Bit 0-7)
6	CNT8 counter 2 (Bit 8-15)
7	CNT8 counter 2 (Bit 16-23)
8	CNT8 counter 2 (Bit 24-31)

Hinweis:

Beachten Sie, dass jeweils nur die ersten 32-Bit der beiden 48-Bit Eingangszähler automatisch in einem CAN-Paket versendet werden können.

1.2.7.6.5. Digitale Eingangszähler (48-Bit) - 64-Bit Paket

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Bit	Inhalt
1	0-7	CNT8 counter 1 (Bit 0-7)
2	0-7	CNT8 counter 1 (Bit 8-15)
3	0-7	CNT8 counter 1 (Bit 16-23)
4	0-7	CNT8 counter 1 (Bit 24-31)
5	0-7	CNT8 counter 1 (Bit 31-39)
6	0-7	CNT8 counter 1 (Bit 40-47)
7	0-3	CNT8 counter 1 Zähler-Modus
7	4-7	CNT8 counter 1 Sub-Modus
8	0	CNT8 counter 1 Eingangszustand (0/1)
8	1-3	Nicht benutzt
8	4-7	CNT8 counter 1 Eingangsfiler

1.2.7.6.6. Analoge Ein- / Ausgänge

1.2.7.6.6.1. Analoge Eingänge

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	A/D channel 5 (Bit 0-7)
2	A/D channel 5 (Bit 8-15)
3	A/D channel 6 (Bit 0-7)
4	A/D channel 6 (Bit 8-15)
5	A/D channel 7 (Bit 0-7)
6	A/D channel 7 (Bit 8-15)
7	A/D channel 8 (Bit 0-7)
8	A/D channel 8 (Bit 8-15)

Der Wertebereich eines A/D Wandlers gibt an, in welchem Bereich analoge Signale (z.B. im Bereich 0-5V), digital umgesetzt werden. Die Einstellungen bezüglich des Wertebereiches können im CAN-Configuration-Utility vorgenommen werden.

Anmerkung:

Der hexadezimal Wert FFFF kennzeichnet immer die obere Grenze eines Wertebereiches, der Wert 0000 die untere.

Formel (A/D-Modus +/-10V, +/-5V oder +/-2,5V)

Spannung = $(\text{Wert} * (\text{max. Spannungswert} * 2) / 0xFFFF) - \text{max. Spannungswert}$

Formel (A/D-Modus 0..10V, 0..5V oder 0..2,5V)

Spannung = $\text{Wert} * \text{max. Spannungswert} / 0xFFFF$

Formel (A/D-Modus 0..20mA, 4..20mA oder 0..24mA)

Stromstärke = $\text{Wert} * 25 / 0xFFFF$

1.2.7.6.6.2. Analoge Ausgänge

Aufbau eines 8 Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	D/A channel 1 (Bit 0-7)
2	D/A channel 1 (Bit 8-15)
3	D/A channel 2 (Bit 0-7)
4	D/A channel 2 (Bit 8-15)
5	D/A channel 3 (Bit 0-7)
6	D/A channel 3 (Bit 8-15)
7	D/A channel 4 (Bit 0-7)
8	D/A channel 4 (Bit 8-15)

Der Wertebereich eines D/A Wandlers gibt an, in welchem Bereich digitale Signale analog (z.B. im Bereich 0-5V) umgesetzt werden. Die Einstellungen bezüglich des Wertebereiches können im CAN-Configuration-Utility vorgenommen werden.

Anmerkung:

Der hexadezimal Wert FFFF kennzeichnet immer die obere Grenze eines Wertebereiches, der Wert 0000 die untere.

Hinweis:

Die Ausgabe in einen Strombereich ist nur bei Modulen möglich, die diesen Modus auch unterstützen.

1.2.7.6.6.3. Beispiele

Beispiel Spannungsbereich $\pm 10\text{V}$

Wert (hex)	Spannung
FFFF	+10 V
8000	0 V
0000	-10V

Beispiel zur Berechnung:

CAN-Data-Byte0, 1 = 0x4711[hex]

Wertebereich = +/-10 V

Berechnung:

Spannung = $(0x4711 * (10 * 2) / 0xFFFF) - 10 = -4,45 \text{ V}$

Beispiel Spannungsbereich 0-5V

Wert (hex)	Spannung
FFFF	+5 V
8000	+2,5 V
0000	0 V

Beispiel zur Berechnung:

CAN-Data-Byte0, 1 = 0x4711[hex]

Wertebereich = 0-5 V

Berechnung:

Spannung = $0x4711 * 5 / 0xFFFF = 1,38 \text{ V}$

Beispiel Strombereich

Wert (hex)	Stromstärke
FFFF	25 mA
8000	12,5 mA
0000	0 mA

Beispiel zur Berechnung:

CAN-Data-Byte0, 1 = 0x4711[hex]

Wertebereich = 0..20 mA, 4..20 mA oder 0..24 mA

Berechnung:

Stromstärke = $0x4711 * 25 / 0xFFFF = 6,94 \text{ mA}$

Hinweis:

Bitte beachten Sie, dass sich bei einem Auto-TX-Paket mit eingestelltem Strombereich, der Wert eines A/D-Kanals im CAN-Paket auf den Modus 0..25mA bezieht.

1.2.7.6.7. Temperatur Eingänge

Dieses Beispiel zeigt den Aufbau eines Auto-TX-Paketes mit den Einstellungen für PT-100 Kanal 1 und 2.

CAN-Data-Byte	Bit	Inhalt
1	0-7	Wert Kanal 1 (Bit 0-7)
2	0-6	Wert Kanal 1 (Bit 8-14)
	7	Vorzeichen Kanal 1 (0 = positiv, 1 = negativ)
3	0-1	Faktor Kanal 1 (0 [dez] = illegaler Wert / Sensor nicht verbunden, 1 [dez] = Faktor 10, 2 [dez] = Faktor 100)
	2-7	nicht benutzt
4	0	Status PT100 Sensor Kanal 1 (0 = nicht verbunden, 1 = verbunden)
	1-7	nicht benutzt
5	0-7	Wert Kanal 2 (Bit 0-7)
6	0-6	Wert Kanal 2 (Bit 8-14)
	7	Vorzeichen Kanal 2 (0 = positiv, 1 = negativ)
7	0-1	Faktor Kanal 2 (0 [dez] = illegaler Wert / Sensor nicht verbunden, 1 [dez] = Faktor 10, 2 [dez] = Faktor 100)
	2-7	nicht benutzt
8	0	Status PT100 Sensor Kanal 2 (0 = nicht verbunden, 1 = verbunden)

CAN-Data-Byte	Bit	Inhalt
	0-7	nicht benutzt

Berechnung der Temperatur

Temperatur = (Vorzeichen) Wert[dez] / Faktor

1.2.7.6.8. Stepper

Aufbau eines 8-Byte langen CAN-Paketes:

CAN-Data-Byte	Inhalt
1	COMMAND
2	PAR1 (Bit 0-7)
3	PAR1 (Bit 8-15)
4	PAR1 (Bit 16-23)
5	PAR1 (Bit 24-31)
6	PAR2 (Bit 0-7)
7	PAR2 (Bit 8-15)
8	PAR3 (Bit 0-7)

1.2.7.6.8.1. Command-Liste

Kommando DAPI_STEPPER_CMD_	mit Wert (hex)	Bedeutung
SET_MOTORCHARACTERISTIC	1 (hex)	Setzen der Motor Konfiguration
GET_MOTORCHARACTERISTIC	2 (hex)	Abfrage der Motor Konfiguration
SET_POSITION	3 (hex)	Setzen der Motorposition
GO_POSITION	4 (hex)	Anfahren einer bestimmten Position
GET_POSITION	5 (hex)	Abfrage einer bestimmten Position
SET_FREQUENCY	6 (hex)	Einstellung der Motorsollfrequenz

Kommando DAPI_STEPPER_CMD_	mit Wert (hex)	Bedeutung
SET_FREQUENCY_DIRECTLY	7 (hex)	Einstellung der Motorfrequenz
GET_FREQUENCY	8 (hex)	Abfrage der Motorfrequenz
FULLSTOP	9 (hex)	Sofortiges Anhalten des Motors
STOP	10 (hex)	Anhalten des Motors (Bremsrampe wird eingehalten)
GO_REFSWITCH	11 (hex)	Anfahren einer Referenzposition
DISABLE	14 (hex)	Ein-/Ausschalten des Motors
MOTORCHARACTERISTIC_LOAD_DEFAULT	15 (hex)	Setzen der Motorcharakteristik auf Defaultwert
MOTORCHARACTERISTIC_EEPROM_SAVE	16 (hex)	Speichern der Motorcharakteristik im EEPROM
MOTORCHARACTERISTIC_EEPROM_LOAD	17 (hex)	Laden der Motorcharakteristik aus dem EEPROM

Kommando DAPI_STEPPER_CMD_	mit Wert (hex)	Bedeutung
GET_CPU_TEMP	18 (hex)	Abfrage der Temperatur des CPU
GET_MOTOR_SUPPLY_VOLTAGE	19 (hex)	Abfrage der Versorgungsspannung des CPU
GO_POSITION_RELATIVE	20 (hex)	Anfahren einer relativen Position

1.2.7.6.8.2. Werte für par 1 zu Befehl SET_MOTORCHARACTERISTIC

Parameter (DAPI_STEPPER_MOTORCHAR_PAR _ ...)	Wert (dez)	Beschreibung
STEPMODE	1	Stepmode (Full-, 1/2-, 1/4-, 1/8-, 1/16-step)
GOFREQUENCY	2	Geschwindigkeit [Full-step / s] - bezogen auf full-step
STARTFREQUENCY	3	Startfrequenz [Full-step / s]
STOPFREQUENCY	4	Stopp-Frequenz [Full-step / s]
MAXFREQUENCY	5	Maximale Frequenz [Full-step / s]
ACCELERATIONSLOPE	6	Steigung der Beschleunigung [Full-step / ms]
DECELERATIONSLOPE	7	Steigung der Verzögerung [Full- step / 10ms]
PHASECURRENT	8	Phasenstrom [mA]
HOLDPHASECURRENT	9	Phasenstrom für Motorhaltung [mA]
HOLDTIME	10	Zeit, in der der Halt zum Motorstopp geht [ms]
STATUSLEDMODE	11	Modus der Status-LED
INVERT_ENDSW1	12	umkehren des Endschalter1
INVERT_ENDSW2	13	umkehren des Endschalter2
INVERT_REFSW1	14	umkehren des Frequenzschalters1
INVERT_REFSW2	15	umkehren des

Parameter (DAPI_STEPPER_MOTORCHAR_PAR _ ...)	Wert (dez)	Beschreibung
		Frequenzschalters1
INVERT_DIRECTION	16	Alle Richtungsangaben umkehren
ENDSWITCH_STOPMODE	17	Einstellung des Stoppverhaltens (0=Fullstop / 1=Stop)

Parameter (DAPI_STEPPER_MOTORCHAR_PAR _ ...)	Wert (dez)	Beschreibung
GOREFERENCEFREQUENCY_TOEND SWITCH	18	Frequenz vor Endschalter [Full- step / s]
GOREFERENCEFREQUENCY_AFTER ENDSWITCH	19	Frequenz nach Endschalter [Full-step / s]
GOREFERENCEFREQUENCY_TOOFF SET	20	Frequenz bis zum optionalen Offset [Full-step / s]

1.2.7.6.8.3. Werte für par 1 zu Befehl GO_REFSWITCH

1.2.7.6.8.4. Beispiel

Programmbeispiel

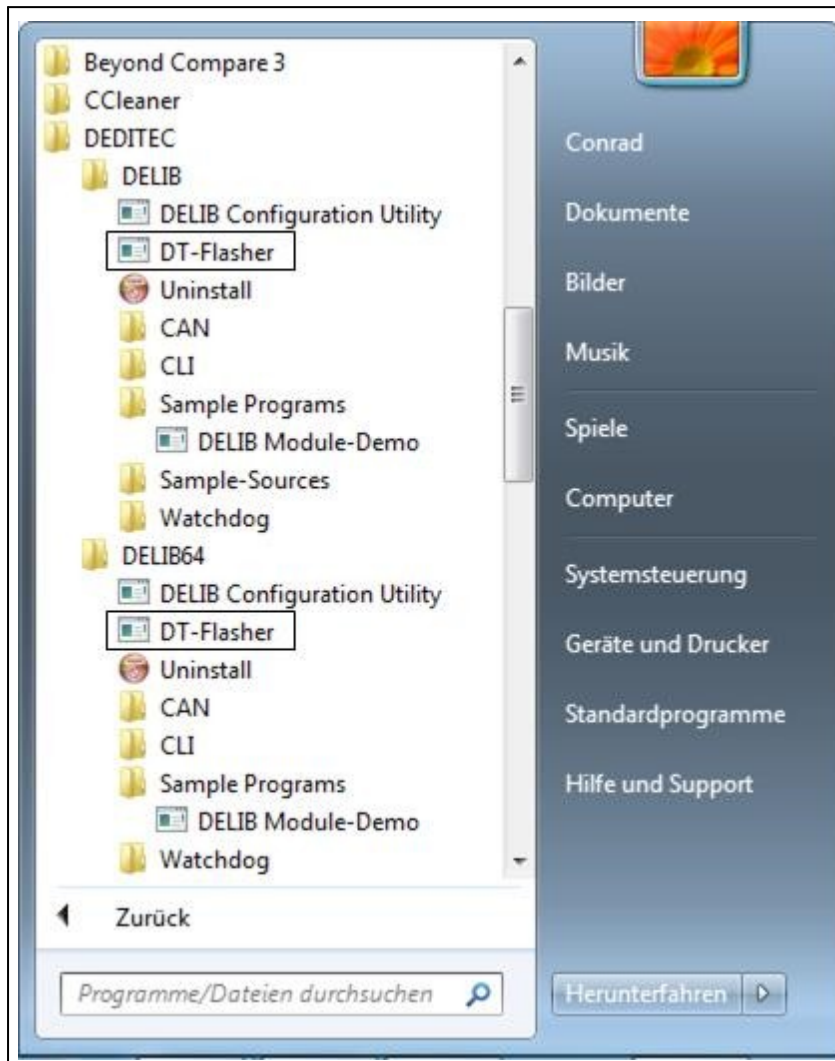
```
DapiStepperCommand(handle, 1,  
DAPI_STEPPER_CMD_GO_POSITION, 3200, 0, 0, 0);
```

wird in einem 8 Byte CAN Paket versendet.

Das Paket hat folgende Struktur:

CAN-Byte	Type	Wert	Byte
1	COMMAND	4	04
2	PAR1 (Bit 0-7)	3200	80
3	PAR1 (Bit 8-15)		0C
4	PAR1 (Bit 16-23)		00
5	PAR1 (Bit 24-31)		00
6	PAR2 (Bit 0-7)	0	00
7	PAR2 (Bit 8-15)		00
8	PAR3 (Bit 0-7)	0	00

1.2.8. 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.

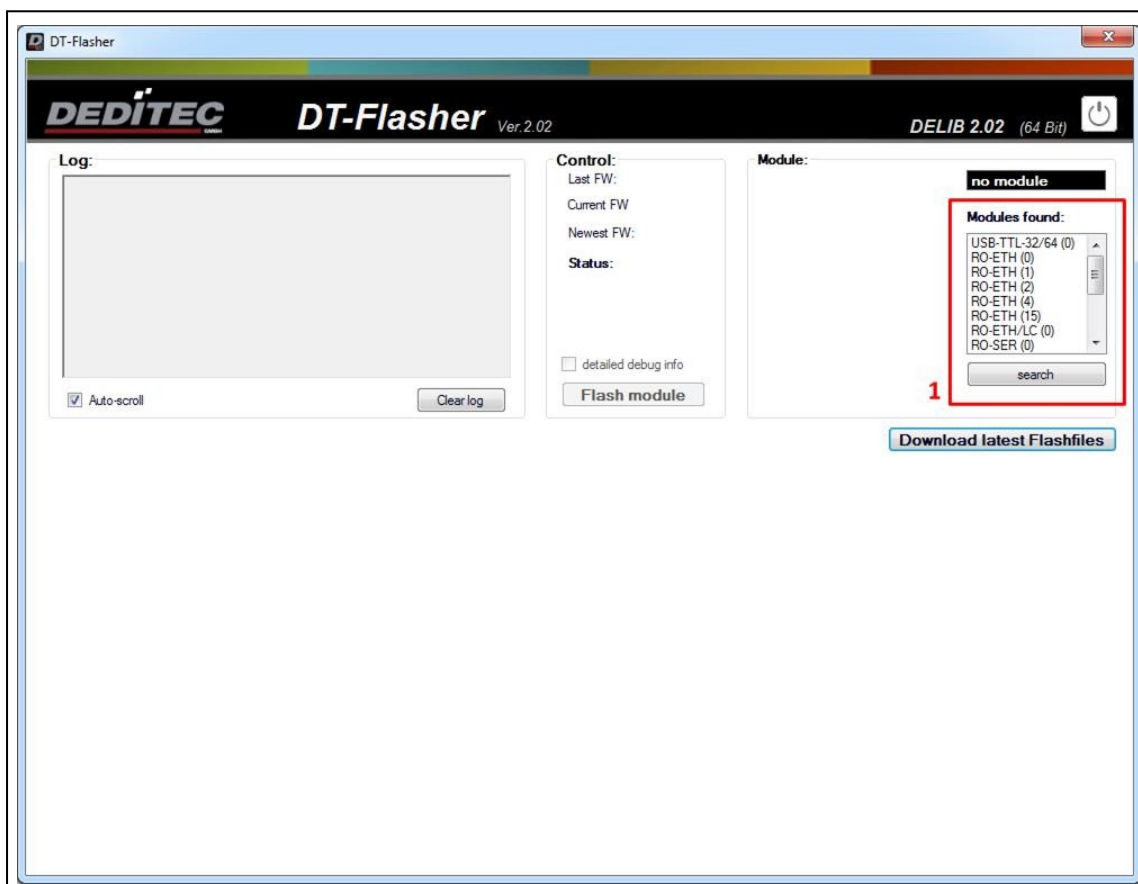
1.2.8.1. Über DEDITEC-Firmware

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:

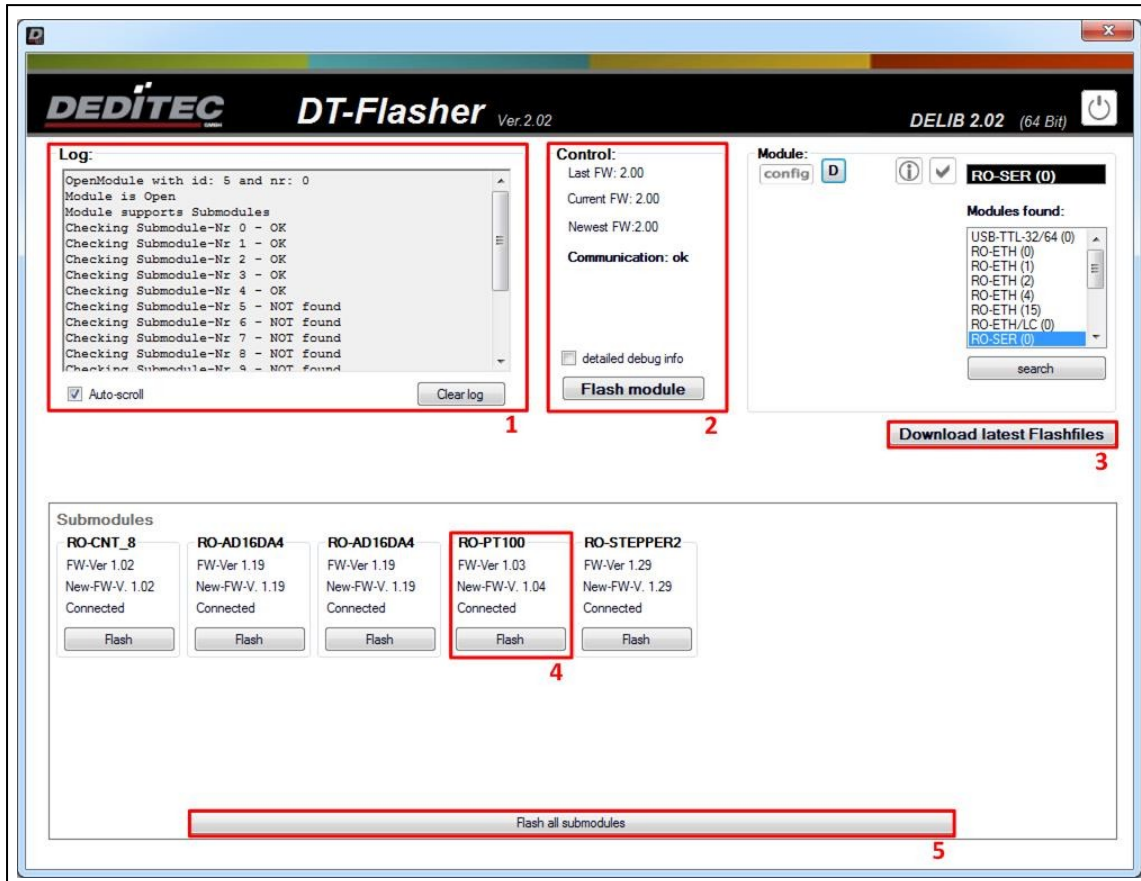
Da neue Firmware-Versionen in der Regel neue Funktionen für Ihr Produkt "freischalten", empfehlen wir daher ein regelmäßiges Firmware-Update Ihrer DEDITEC Produkte.

1.2.8.2. 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"

1.2.8.3. 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.

1.2.8.3.1. 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

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

1.3. 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:

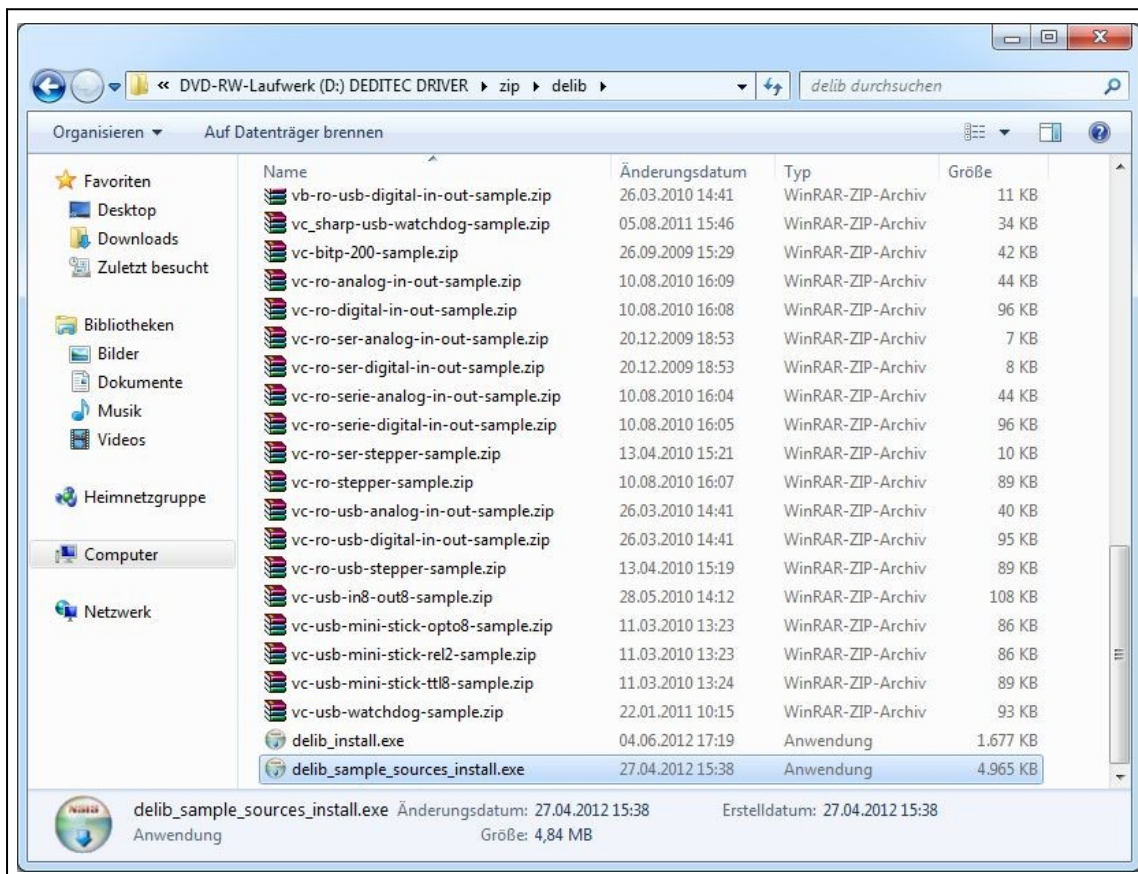
- C
- C++
- C#
- Delphi
- VisualBasic
- VB.NET
- MS-Office
- LabVIEW
- Java

1.3.1. 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`.

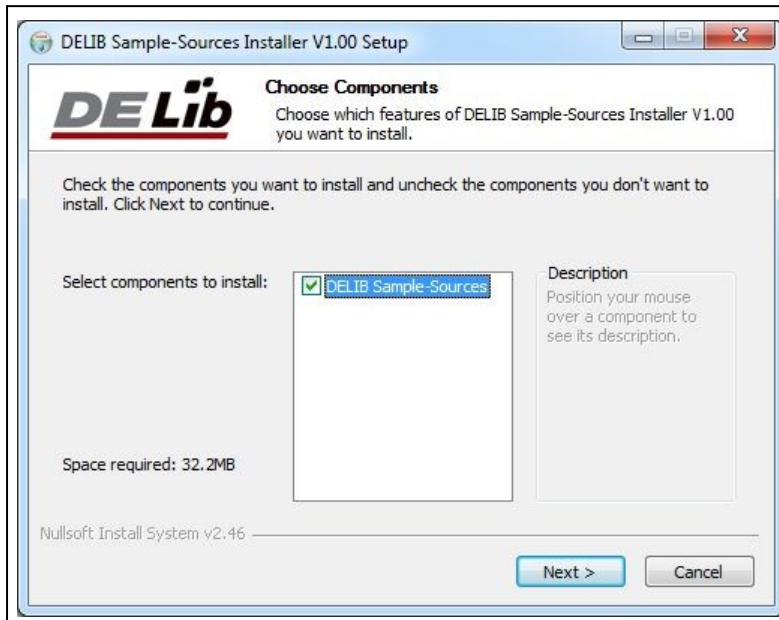
Eine aktuelle Version der Sample Sources finden Sie auch im Internet unter <http://www.deditec.de/de/delib>



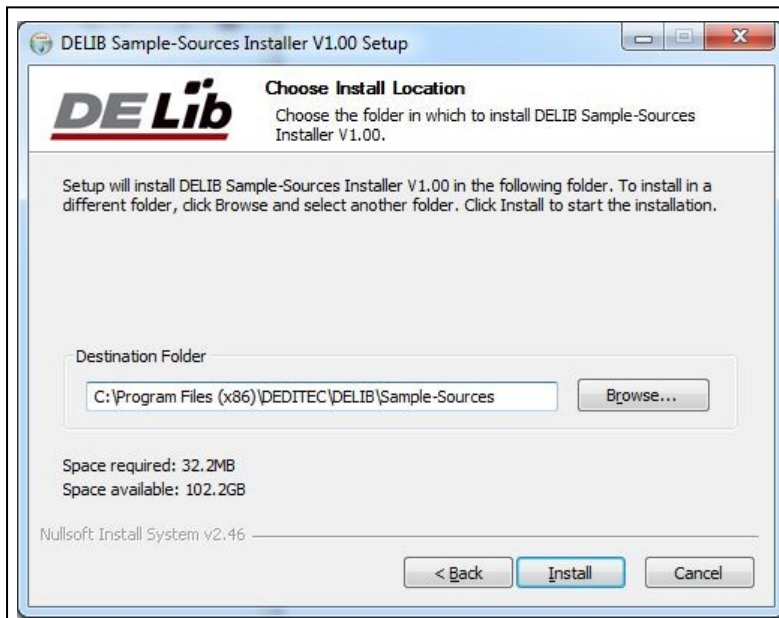
Startbild des DELIB Sample Sources Installer



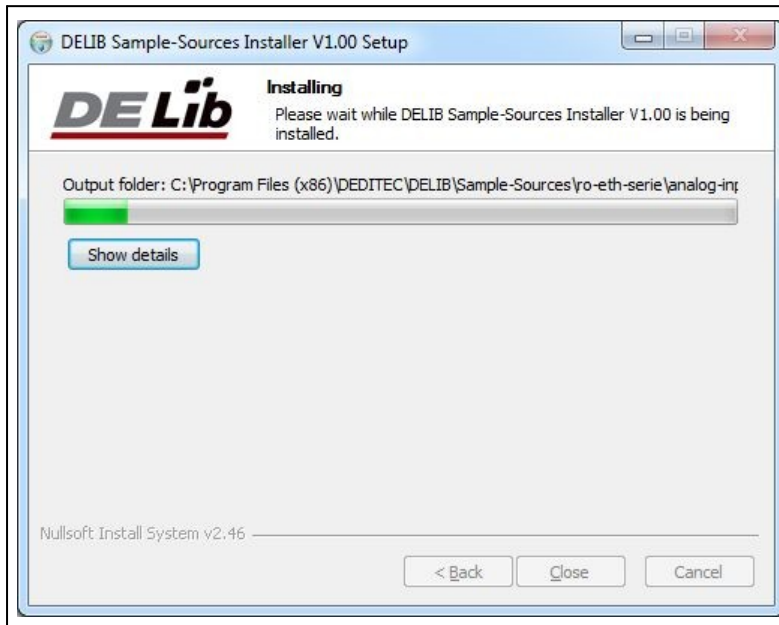
Drücken Sie Next.



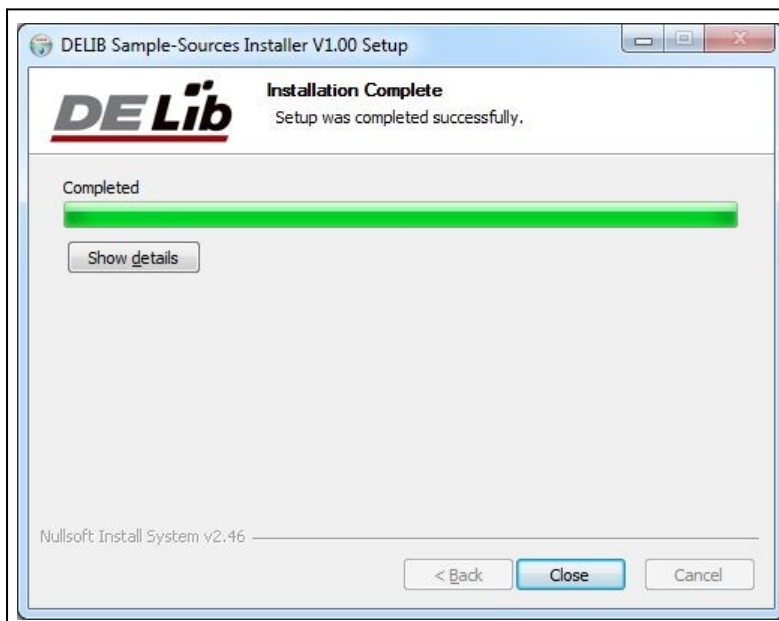
Wählen Sie den Installationsordner und drücken Sie Install.



Die DELIB Sample Sources werden nun installiert.



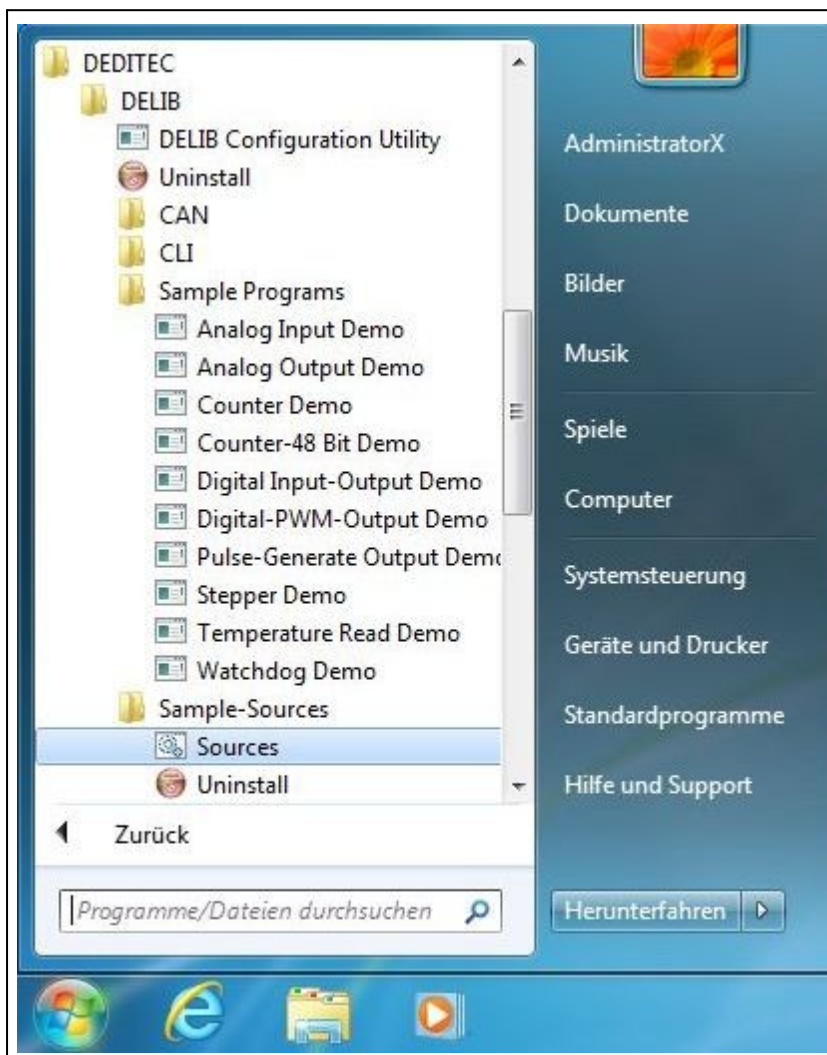
Die DELIB Sample Sources wurden erfolgreich installiert. Drücken Sie Close um die Installation zu beenden.



1.3.2. Benutzung der DELIB Sample Sources

Nach Installation der DELIB Sample Sources finden Sie diese unter

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

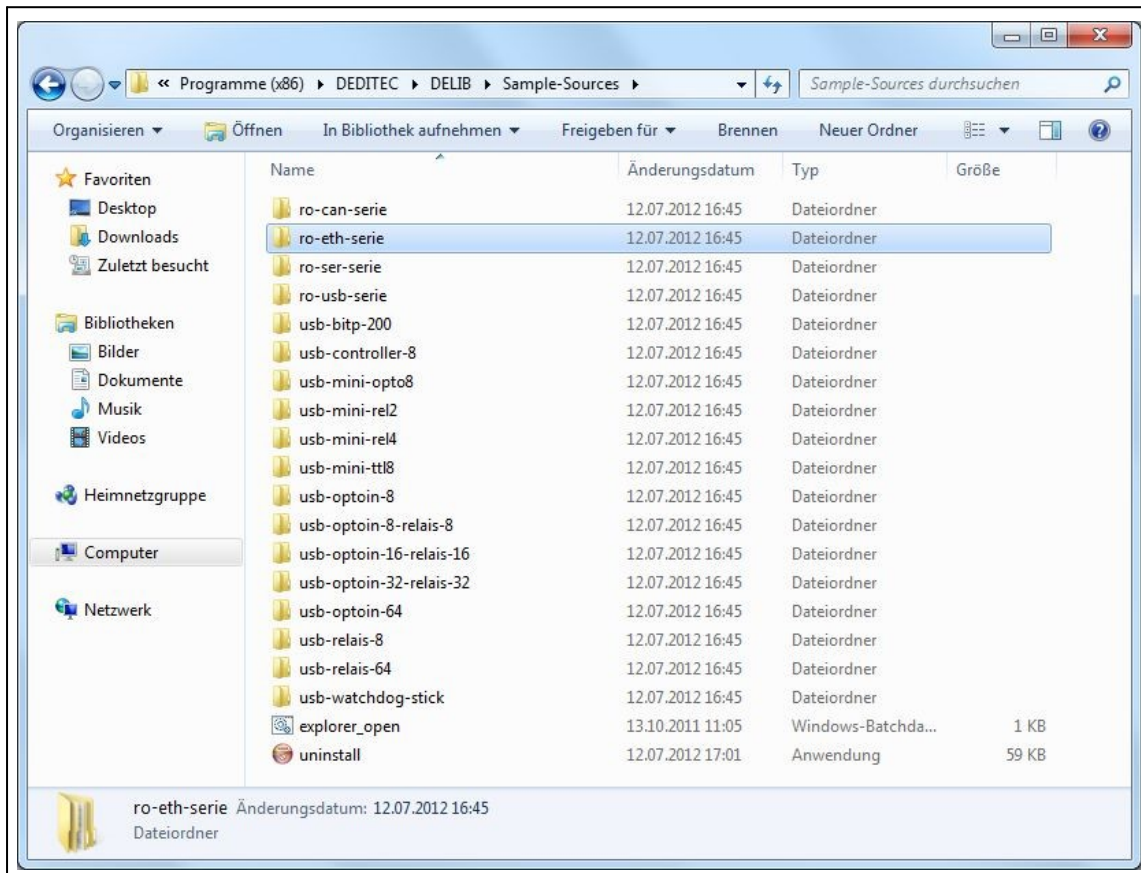


Nun öffnet sich der Windows-Explorer mit einer Übersicht aller Produkte für die ein Beispielprogramm verfügbar ist.

1.3.2.1. 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.

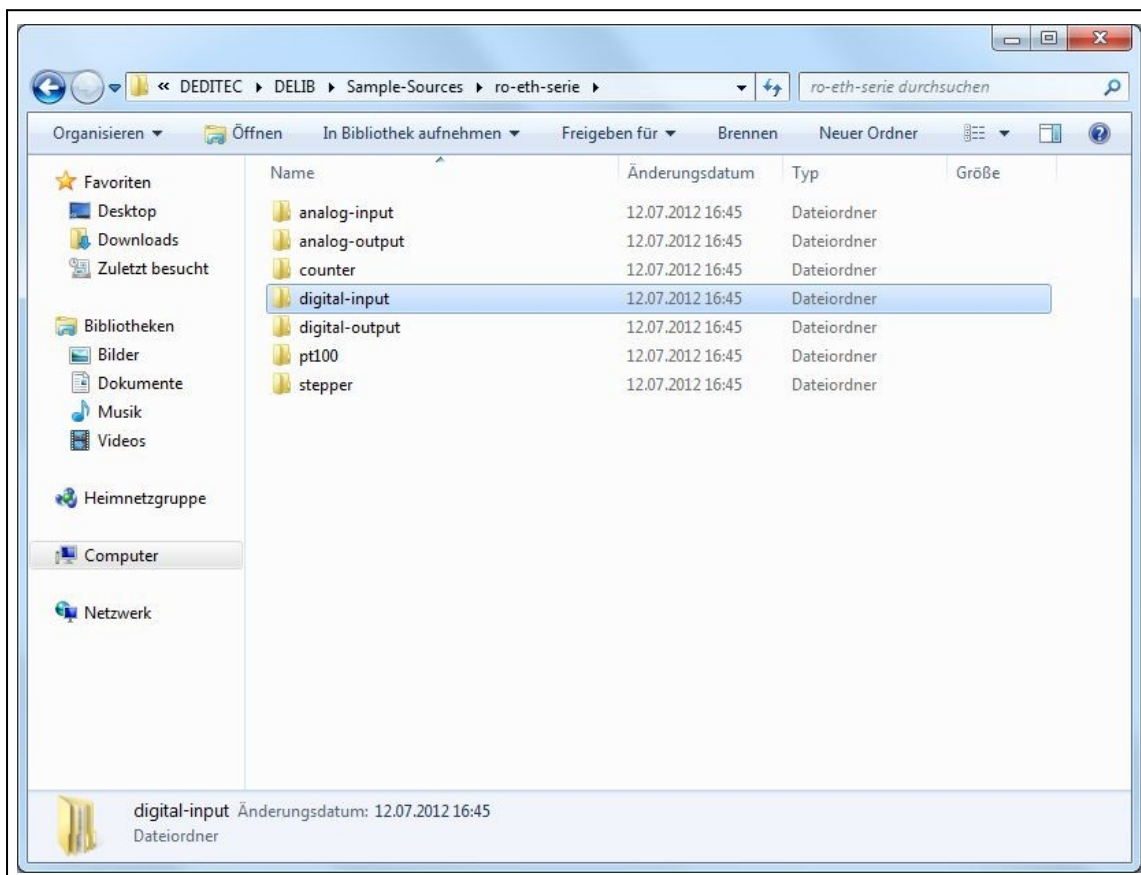
Da es sich um ein RO-ETH-Produkt handelt, wählen bzw. öffnen Sie den Ordner ro-eth-serie.



1.3.2.2. Schritt 2 - Kategorieauswahl

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

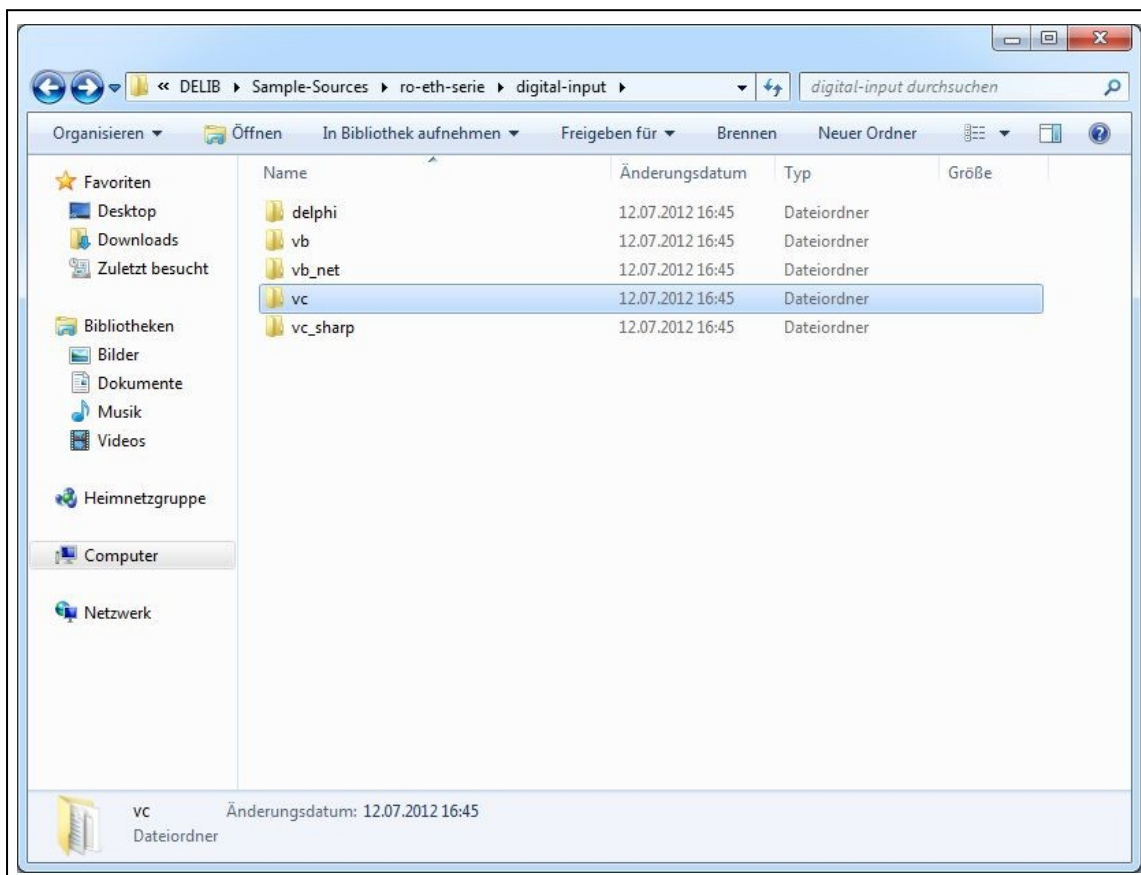
Da wir uns in diesem Beispiel auf die digitalen Eingänge konzentrieren, wählen Sie die Kategorie digital-input



1.3.2.3. Schritt 3 - Programmiersprachenauswahl

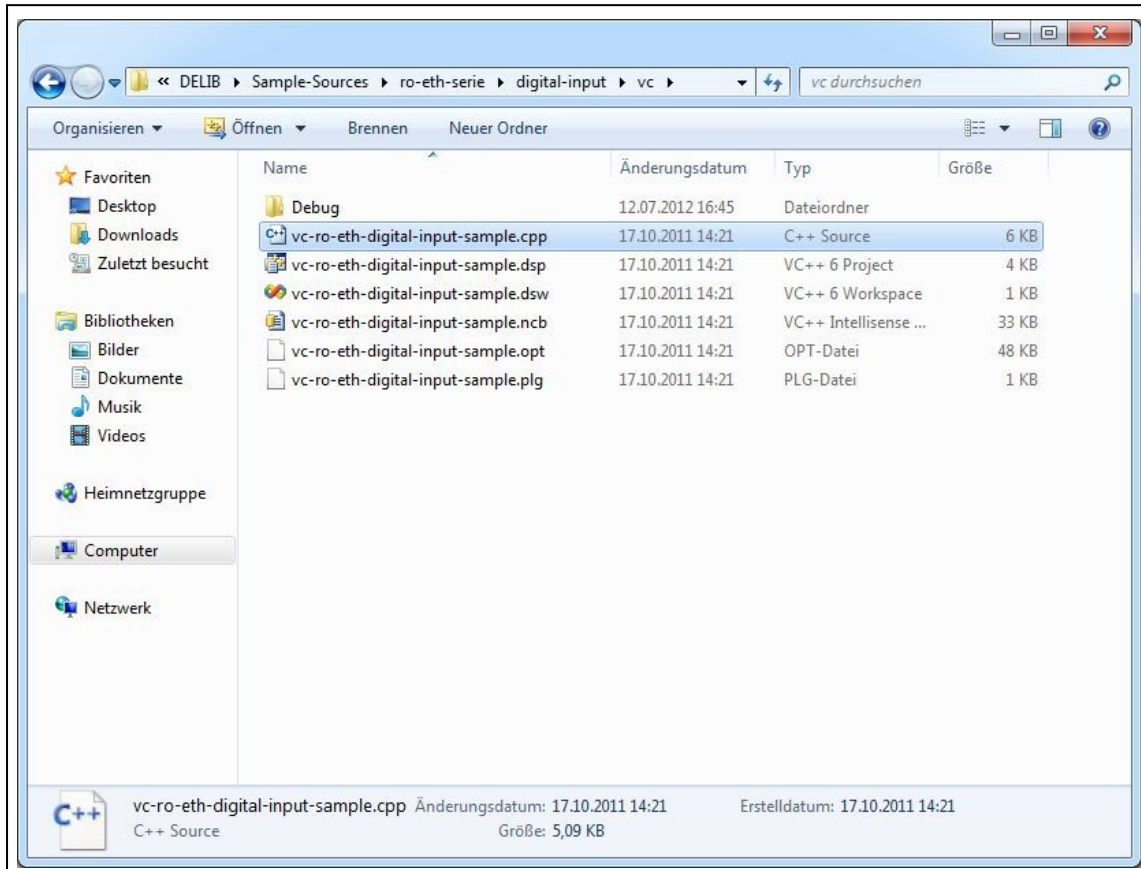
In diesem Schritt sehen Sie alle verfügbaren Programmierbeispiele der gewählten Kategorie, sortiert nach Programmiersprachen.

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

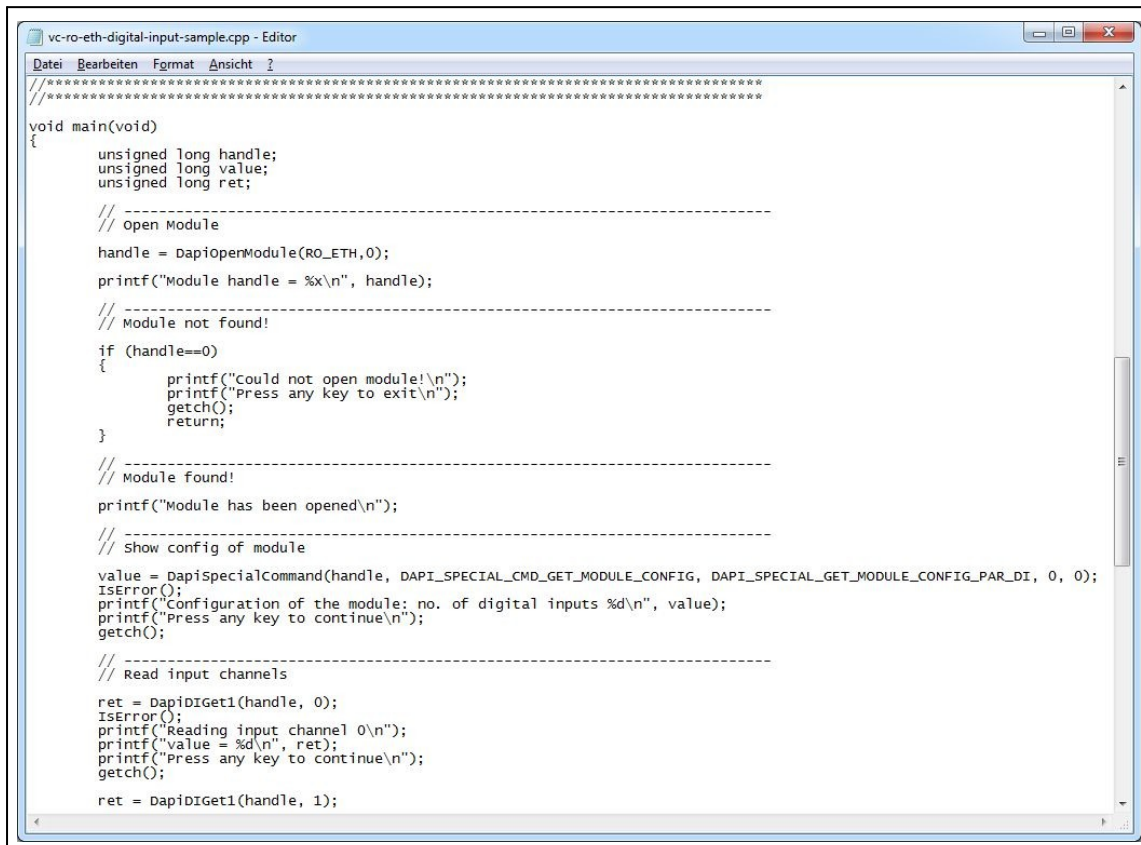


1.3.2.4. Schritt 4 - Quellcode

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

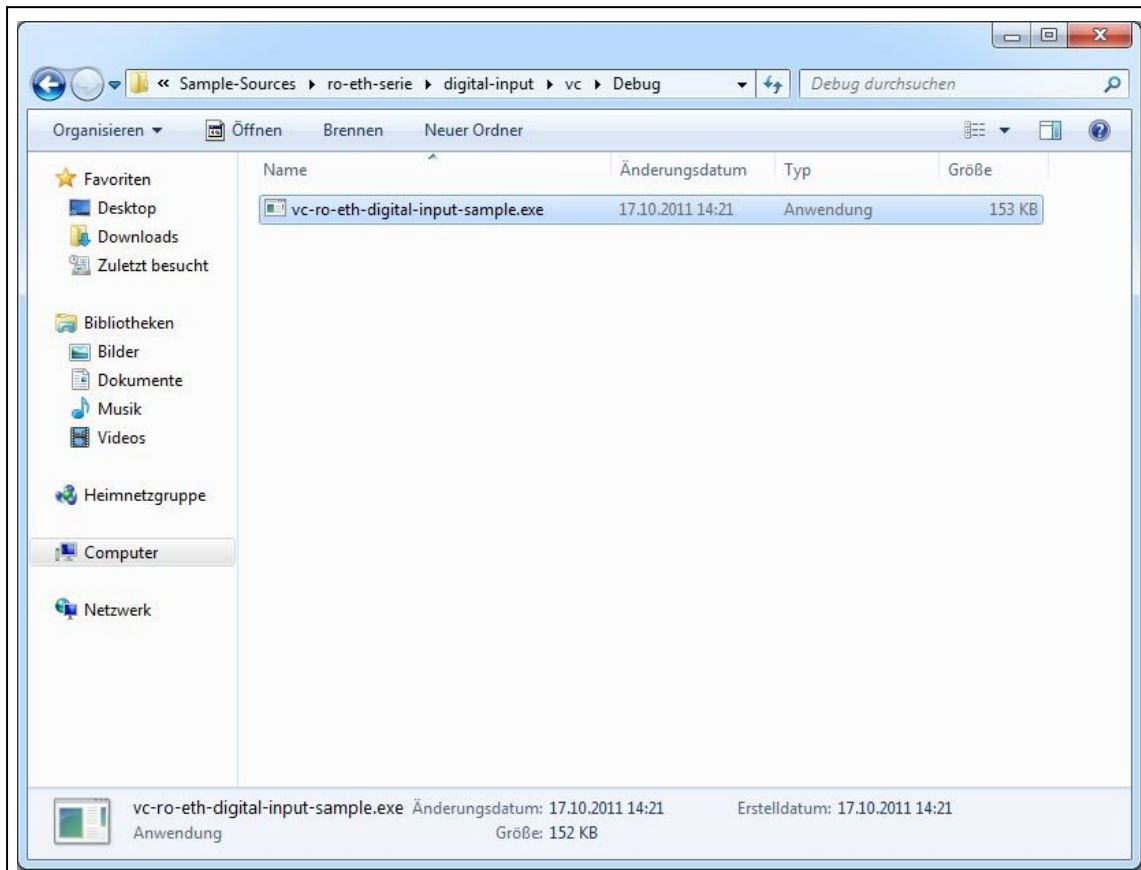
    value = DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_GET_MODULE_CONFIG, DAPI_SPECIAL_GET_MODULE_CONFIG_PAR_DI, 0, 0);
    if (value < 0)
    {
        printf("Configuration of the module: no. of digital inputs %d\n", value);
        printf("Press any key to continue\n");
        getch();
    }

    // -----
    // 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();
    }

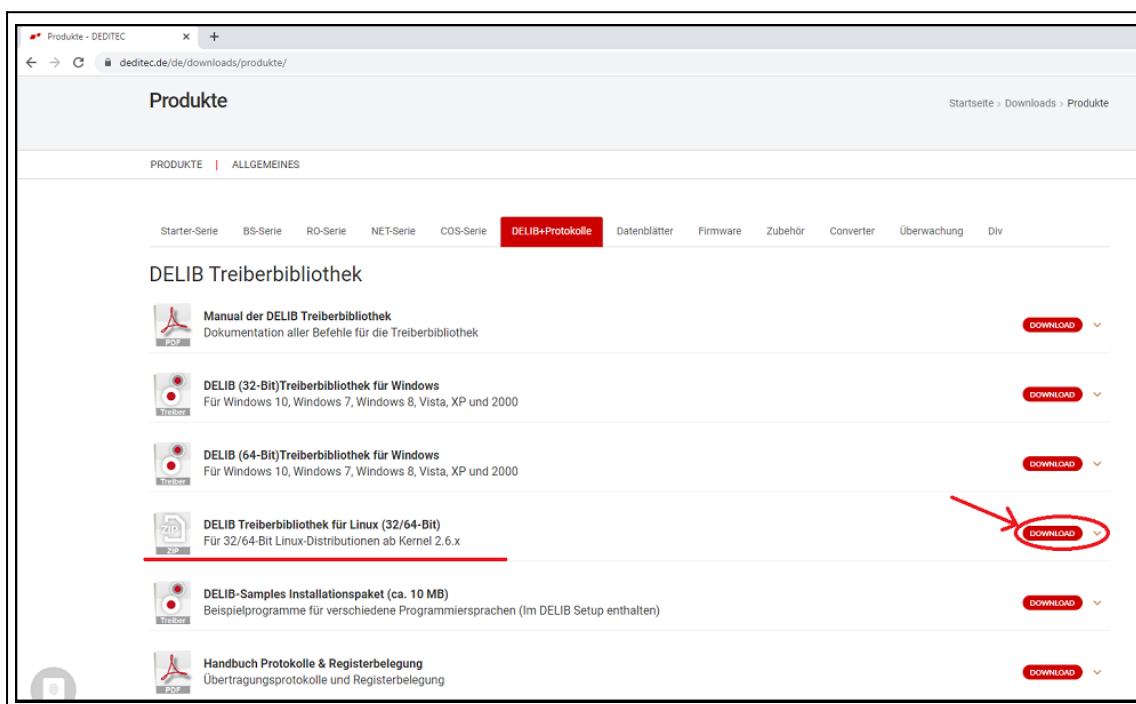
    ret = DapiGetI(handle, 1);
```


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

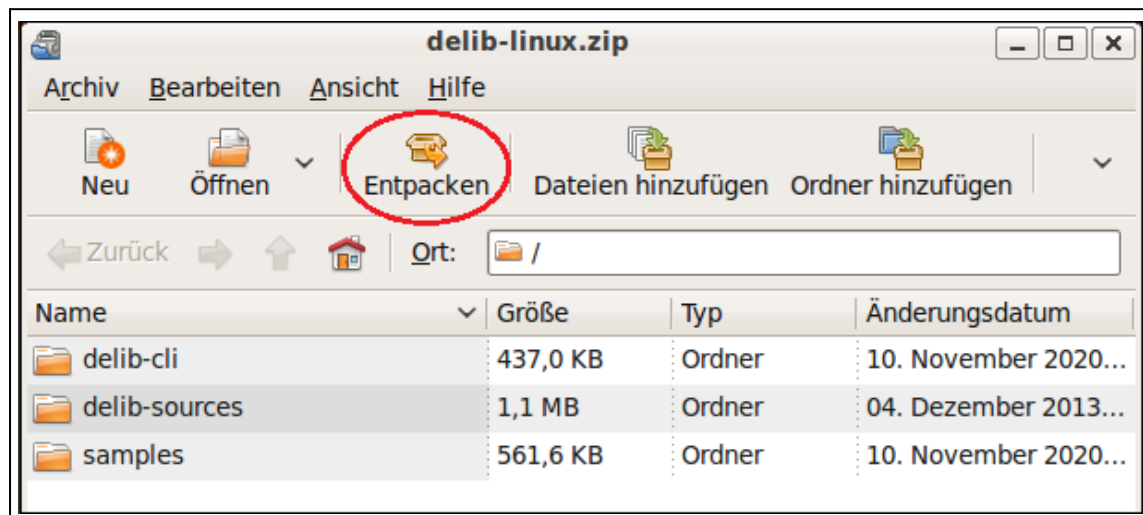


1.4. DELIB für Linux

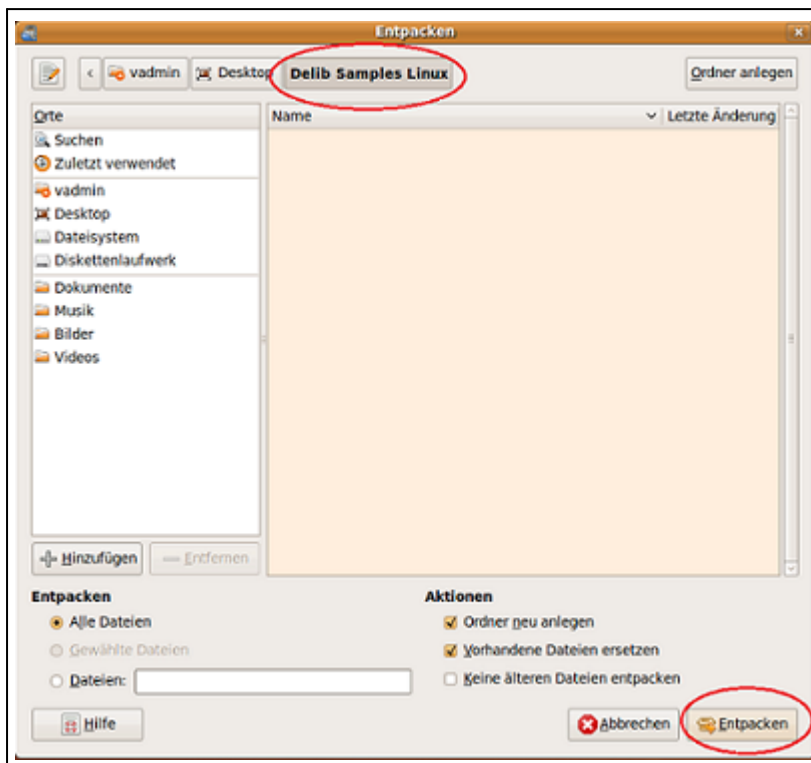
Laden Sie sich die Delib Linux Treiberbibliothek unter "www.deditec.de/de/downloads/produkte/" im Reiter „DELIB+Protokolle“ oder unter "www.deditec.de/media/zip/delib/delib-linux.zip" direkt auf ihr Linux-System.



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.



1.4.1. Verwenden der DELIB Treiberbibliothek für Linux

1.4.1.1. 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“

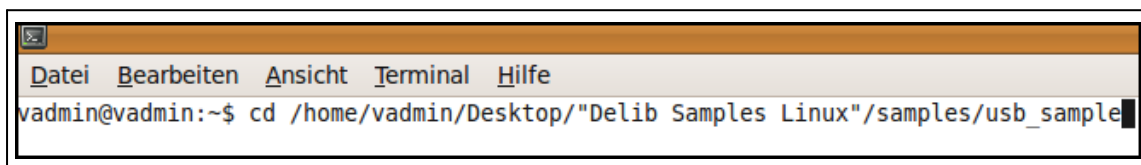
```
23
24 #include <stdio.h>
25 #include <stdlib.h>
26 #include <unistd.h>
27
28 #include "../../delib-sources/delib/library/delib/delib.h"
29
30 int main()
31 {
32     ULONG i;
33     ULONG handle=0;
34
35     printf("\n\n");
36     printf("-----\n");
37     printf("-----\n");
38     printf("-----\n");
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.



```

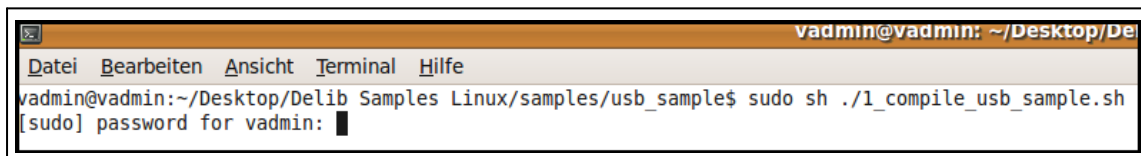
Datei Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~$ cd /home/vadmin/Desktop/"Delib Samples Linux"/samples/usb_sample

```

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:

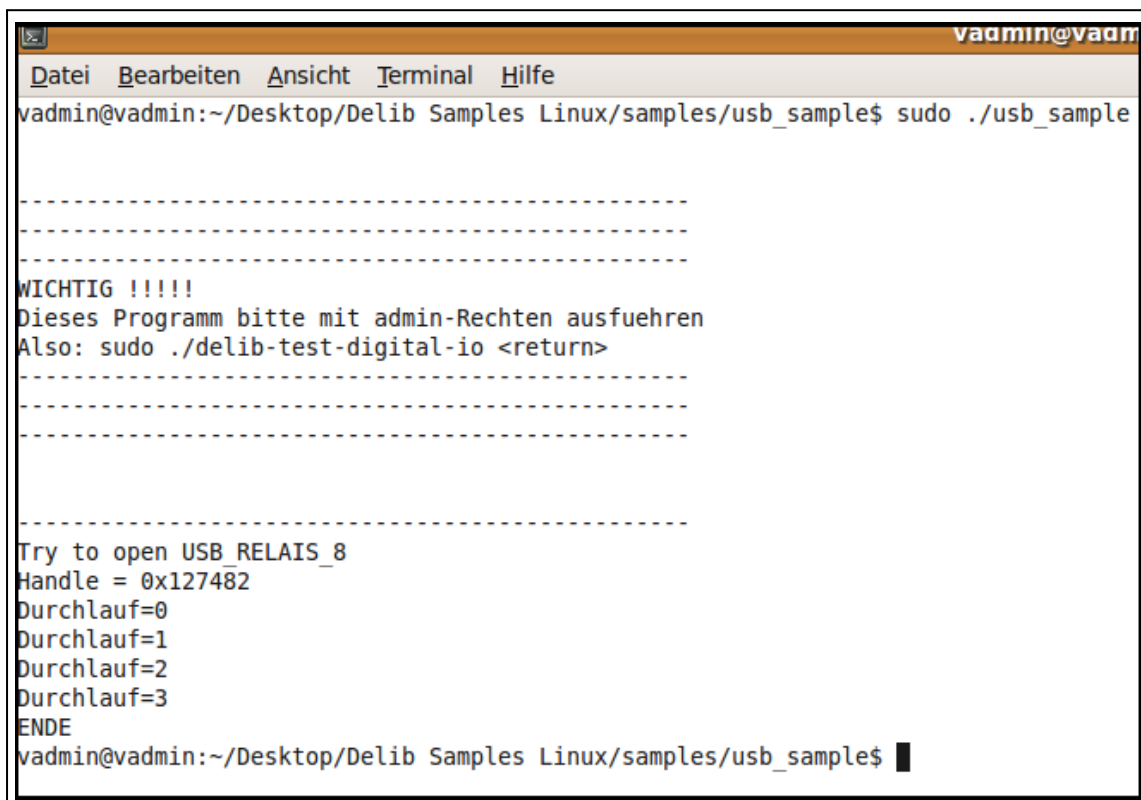
```

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_RELAYS_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_RELAYS_8 in einer Schleife an und wieder ausgeschaltet.

1.4.1.2. 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
```

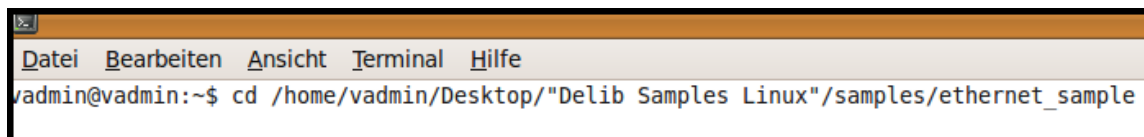
Auskommentieren,
falls keine digitalen
Eingänge vorhanden

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.

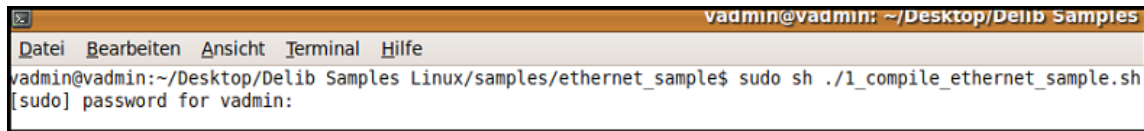
Tipp: Sollten in Ihrem Ordnernamen Leerzeichen enthalten sein, geben Sie diese wie im unteren Beispiel dargestellt in " " an.

A screenshot of a terminal window with a menu bar containing 'Datei', 'Bearbeiten', 'Ansicht', 'Terminal', and 'Hilfe'. The terminal text shows the command 'cd /home/vadmin/Desktop/"Delib Samples Linux"/samples/ethernet_sample' being entered at the prompt 'vadmin@vadmin:~\$'.

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.



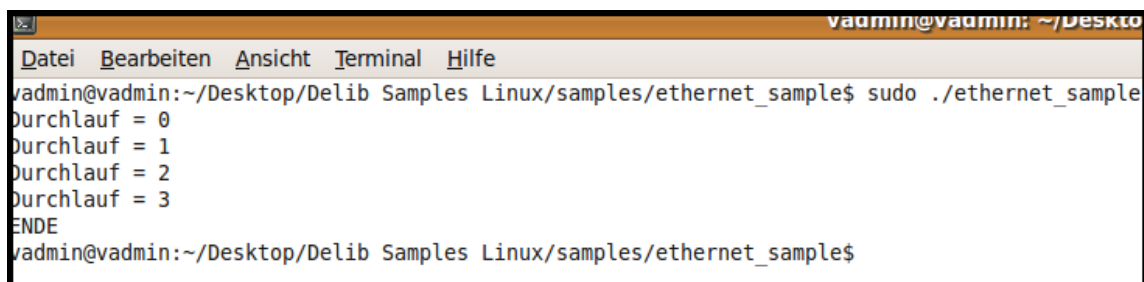
```
vadmin@vadmin: ~/Desktop/Delib Samples
Datei Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~/Desktop/Delib Samples Linux/samples/ethernet_sample$ sudo sh ./1_compile_ethernet_sample.sh
[sudo] password for vadmin:
```

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.

1.4.2. 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

1.4.2.1. 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.

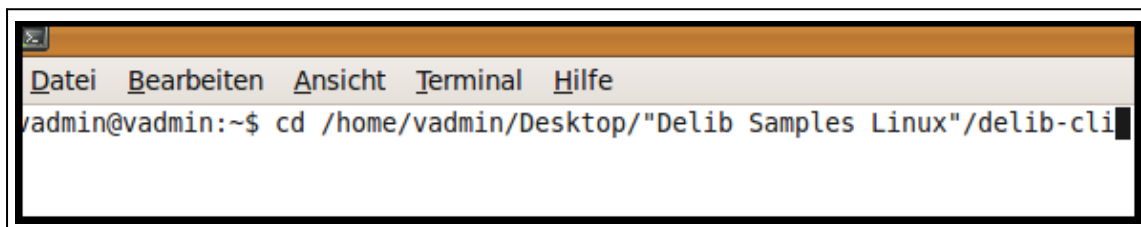
Die IP-Adresse der ETH-Module können über das DELIB Configuration Utility sowie über die Weboberfläche des Moduls eingestellt werden.

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.

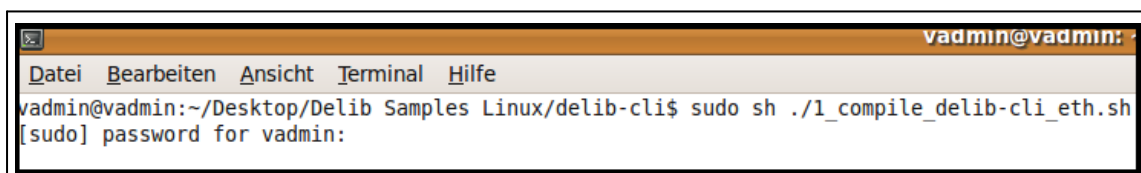


```
File Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~$ cd /home/vadmin/Desktop/"Delib Samples Linux"/delib-cli
```

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.

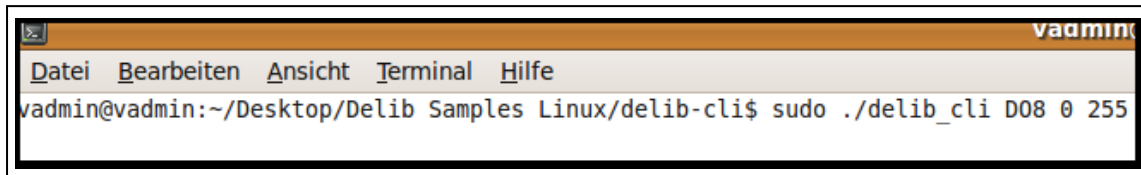


```
vadmin@vadmin:
File Bearbeiten Ansicht Terminal Hilfe
vadmin@vadmin:~/Desktop/Delib Samples Linux/delib-cli$ sudo sh ./1_compile_delib-cli_eth.sh
[sudo] password for vadmin:
```

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 title bar containing a close button icon and the text 'vadmin'. The menu bar includes 'Datei', 'Bearbeiten', 'Ansicht', 'Terminal', and 'Hilfe'. The terminal text shows the user 'vadmin' at host 'vadmin' in the directory '~/Desktop/Delib Samples Linux/delib-cli' running the command 'sudo ./delib_cli D08 0 255'.

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


1.4.2.2. 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

1.5. Weboberfläche

Geben Sie die IP-Adresse (Auslieferungszustand 192.168.1.1) des Moduls in einem Internet-Browser ein.



The screenshot shows a standard web browser security warning. At the top, it says 'Authentifizierung erforderlich' with a close button (X). Below this, it states: 'Für http://192.168.1.1 sind ein Nutzernamen und ein Passwort erforderlich.' and 'Die Verbindung zu dieser Website ist nicht sicher.' There are two input fields: 'Nutzername:' and 'Passwort:'. At the bottom, there are two buttons: 'Anmelden' and 'Abbrechen'.

Der integrierte Web-Server des Moduls erfordert eine Authentifizierung

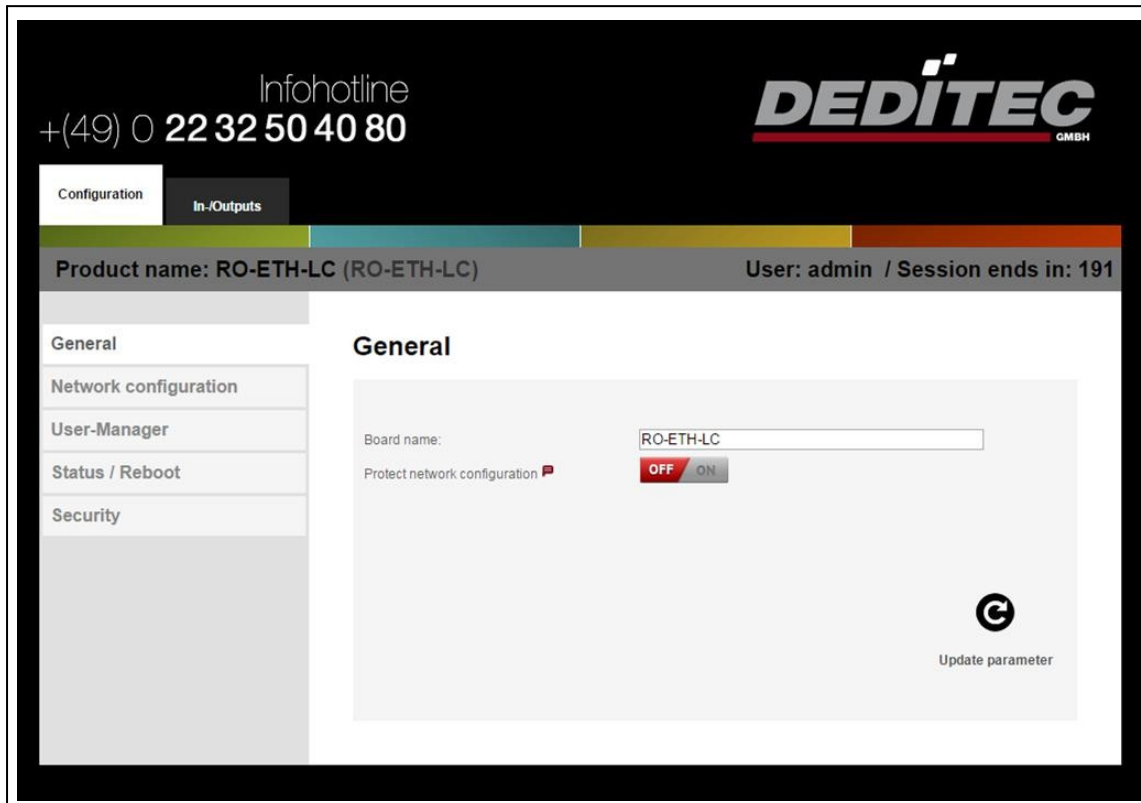
um das Modul vor unberechtigten Zugriffen zu schützen.

Standardmäßig ist folgender Benutzer eingerichtet:

Benutzername	Passwort	Rechte
admin	admin	Administratoren-Rechte

1.5.1. Konfiguration

1.5.1.1. Allgemein



Boardname

Das ist der Name des Moduls, welcher auch im DELIB-Configuration-Utility angezeigt wird.

Dieser Name dient zur Identifizierung mehrerer DEDITEC-Ethernet-Module im Netzwerk.

Protect network configuration

Ist diese Option aktiviert, kann die Netzwerk Konfiguration nur über die Weboberfläche des Moduls geändert werden.

Ein Ändern dieser Konfiguration (z.B. über das DELIB-Configuration Utility) wird in diesem Fall verhindert.

1.5.1.2. Netzwerk Konfiguration

The screenshot displays the DEDITEC web interface. At the top left, there is an 'Infoc hotline' with the phone number '+ (49) 0 22 32 50 40 80'. The DEDITEC logo is in the top right corner. Below the header, there are two tabs: 'Configuration' and 'In-Outputs'. The 'Configuration' tab is active. Below the tabs, a status bar shows 'Product name: RO-ETH-LC (RO-ETH-LC)' and 'User: admin / Session ends in: 197'. On the left side, there is a sidebar menu with options: 'General', 'Network configuration', 'User-Manager', 'Status / Reboot', and 'Security'. The 'Network configuration' option is selected. The main content area is titled 'Network configuration' and contains several input fields: 'MAC' (00:C0:D5:02:00:0D), 'Obtain IP address automatically (DHCP)' (a toggle switch set to 'OFF'), 'IP-address' (192.168.1.25), 'Netmask' (255.255.255.0), 'Std-GW' (192.168.1.254), and 'TCP port' (9912). Below these fields, a notice states: 'Notice: Changing TCP port requires board reboot'. At the bottom right of the configuration area, there is a circular refresh icon and the text 'Update parameter'.

Hier kann die Netzwerk Konfiguration des Moduls geändert werden.

Wird diese Konfiguration geändert, werden Sie automatisch auf die neue IP-Adresse weitergeleitet, sofern diese vom PC erreichbar ist.

Eine Änderung des Ports erfordert einen Neustart des Moduls.

1.5.1.3. Benutzer Manager

The screenshot shows the DEDITEC web interface. At the top, there is an 'Inf hotline' number: +(49) 0 22 32 50 40 80. The DEDITEC logo is in the top right corner. Below the header, there are two tabs: 'Configuration' and 'In-Outputs'. The 'Configuration' tab is active. Below the tabs, there is a status bar showing 'Product name: RO-ETH-LC (RO-ETH-LC)' and 'User: admin / Session ends in: 196'. On the left side, there is a sidebar with a list of menu items: 'General', 'Network configuration', 'User-Manager', 'Status / Reboot', and 'Security'. The 'User-Manager' item is selected. The main content area is titled 'User-Manager'. It shows 'Modul Status: OK'. Below this, there is a section for 'Webinterface requires login' with a toggle switch set to 'ON'. There is a 'Username:' field with the value 'admin' and a 'Set password' button. Below that, there is a 'Session valid time' field with the value '200' and the unit 'sec'. A notice states: 'Notice: Changes apply after board restart'. At the bottom right of the main content area, there is a circular refresh icon and the text 'Update parameter'.

Remove

Löscht das entsprechende Benutzerkonto.

Add User

Erstellt ein neues Benutzerkonto. Die Eingabemaske fordert Sie auf einen Benutzernamen und Passwort einzugeben.

Session valid time

Gibt die Zeit an, wie lange eine Anmeldung gültig ist. Läuft diese Zeit ab, muss der Nutzer sich neu anmelden.

Achtung:

Die Änderung dieser Zeit benötigt einen Neustart des Moduls.

Klicken Sie auf ein Benutzerkonto (z.B. gast) um die Einstellungen zu ändern.

The screenshot shows the DEDITEC web interface. At the top, there is an infohotline number +49 0 22 32 50 40 80 and the DEDITEC GMBH logo. Below this is a navigation bar with tabs for Configuration, In-/Outputs, Custom, and Logout. A status bar indicates the product name as RO-ETH (DEDITEC RO-ETH-Module) and the user as admin, with a session ending in 125 seconds. On the left, a sidebar lists various configuration options: General, Network configuration, Network time (NTP), HTTP-Server, Mail-Server, User-Manager (selected), FW-Update, Log's, Status / Reboot, and Security. The main content area is titled 'User-Manager' and shows the status 'requesting userlist succeeded'. It displays a list of users with 'gast' selected. The 'gast' user configuration is shown in a form with fields for Rights (CONFIG rd, IO rd), available rights (IO_wr, CONFIG_wr, ADMINISTRATION), and set rights (IO_rd, CONFIG_rd). There are buttons for Remove, edit, and update. A password field is also present with a 'set' button. Below the user list, there is a 'Session valid time' field set to 500 seconds and a 'Notice: Changes apply after board restart'. The DEDITEC logo and 'Update parameter' text are at the bottom right.

Edit

Ändert die Zugriffsberechtigungen des aktuellen Benutzerkontos. Klicken Sie auf eine Zugriffsberechtigung um diese entweder aus- oder abzuwählen.

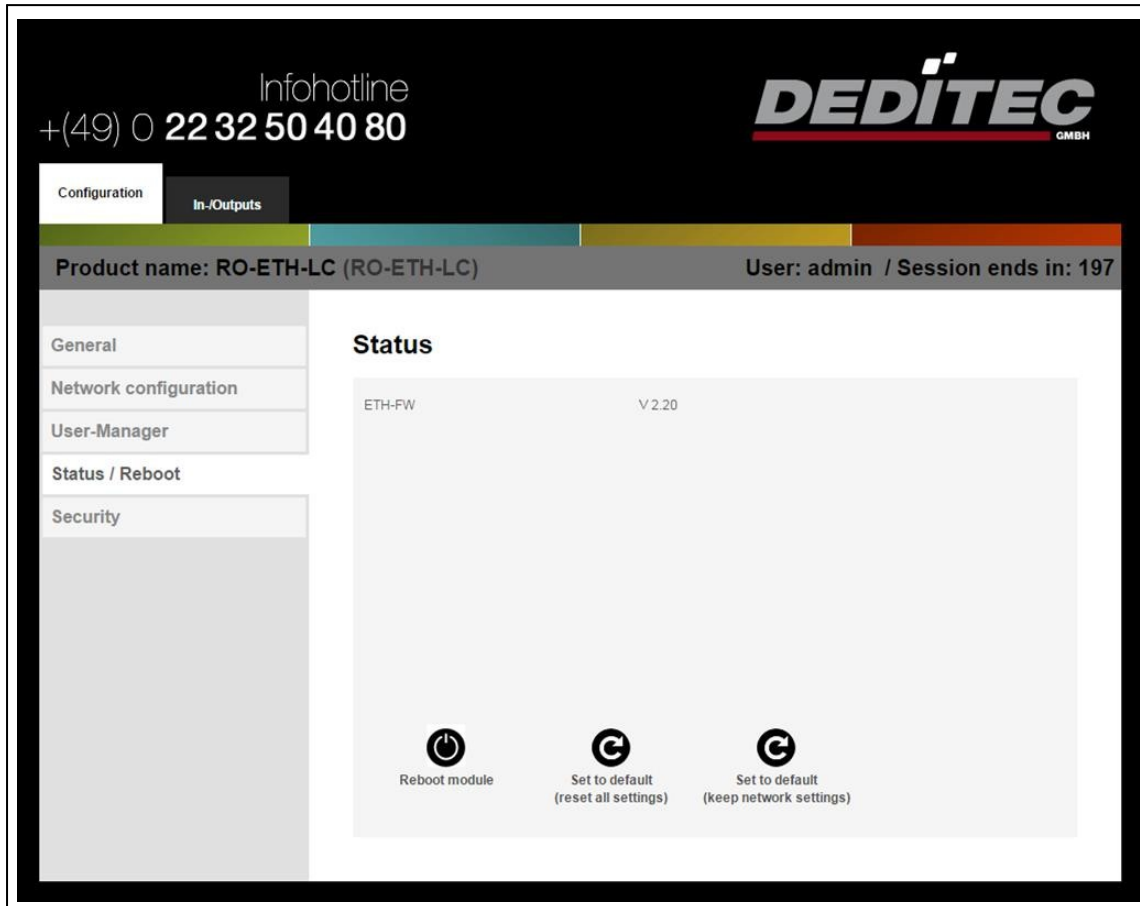
Update

Aktiviert die Zugriffsberechtigungen des aktuellen Benutzerkontos.

Password

Hier kann für das aktuelle Benutzerkonto ein neues Passwort gesetzt werden, welches mit set bestätigt werden muss.

1.5.1.4. Status



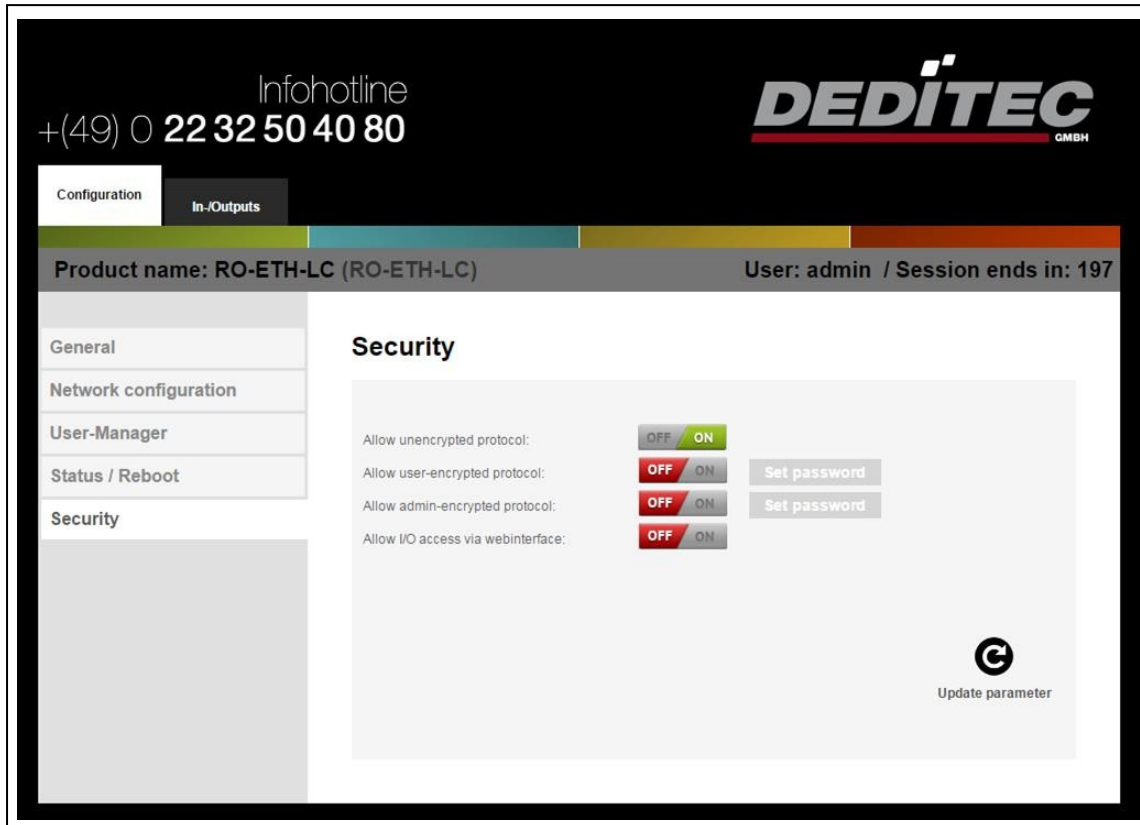
Auf der Status Seite sehen Sie die Revisionsnummern der wichtigsten System Prozesse.

Darüber hinaus, kann an dieser Stelle das Modul per Klick neugestartet, oder die Netzwerk-Einstellungen auf Werkseinstellungen zurückgesetzt werden.

Achtung

Das Neustarten des Moduls, sowie das Zurücksetzen auf Werkseinstellung benötigen Administratoren Berechtigungen.

1.5.1.5. Sicherheit



Allow unencrypted protocol

Diese Option bestimmt, ob ein Zugriff auf das Modul mit einem unverschlüsseltem Protokoll erlaubt wird.

Allow user-encrypted protocol

Diese Option bestimmt, ob ein Zugriff auf das Modul mit einem User-verschlüsseltem Protokoll erlaubt wird.

Dieses Protokoll verfügt über eingeschränkte Zugriffs-Rechte und empfiehlt sich für Benutzer, deren Kommunikation verschlüsselt aber keine System Einstellungen ändern sollen.

Allow admin-encrypted protocol

Diese Option bestimmt, ob ein Zugriff auf das Modul mit einem Admin-verschlüsseltem Protokoll erlaubt wird.

Dieses Protokoll wird benötigt um beispielsweise die Namen der angeschlossenen Ein-/Ausgänge zu lesen oder zu ändern.

Allow I/O access via webinterface

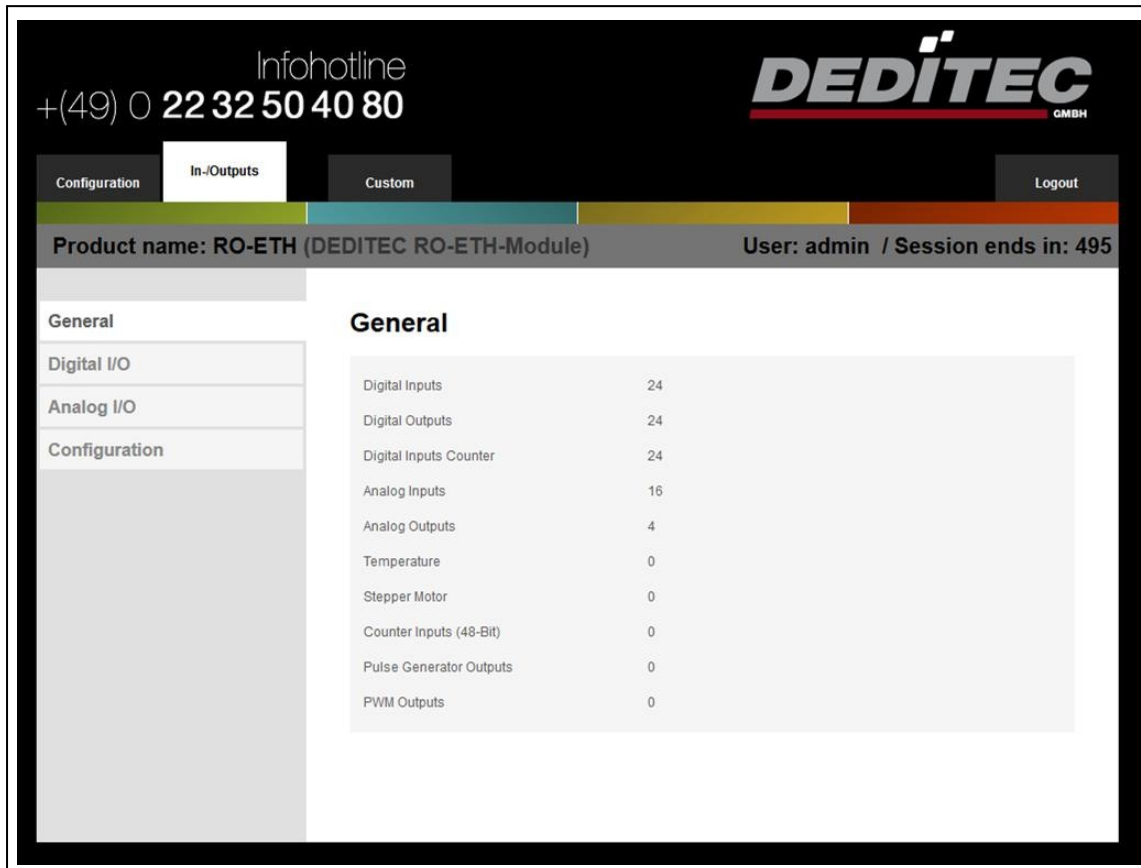
Diese Option bestimmt, ob angeschlossene Ein-/Ausgänge von der Weboberfläche aus gelesen/geschaltet werden können.

Achtung:

Wenn das Passwort zur User- oder Admin-Verschlüsselung geändert wird, muss dieses auch im DELIB-Configuration-Utility nachgetragen werden.

1.5.2. Ein-/Ausgänge

1.5.2.1. Allgemein



Infohotline
+(49) 0 22 32 50 40 80

DEDITEC
GMBH

Configuration In-/Outputs Custom Logout

Product name: RO-ETH (DEDITEC RO-ETH-Module) User: admin / Session ends in: 495

General

Digital Inputs	24
Digital Outputs	24
Digital Inputs Counter	24
Analog Inputs	16
Analog Outputs	4
Temperature	0
Stepper Motor	0
Counter Inputs (48-Bit)	0
Pulse Generator Outputs	0
PWM Outputs	0

Hier sehen Sie eine Auflistung der angeschlossenen Ein-/Ausgänge

1.5.2.2. Digitale Eingänge

Inf hotline
+(49) 0 22 32 50 40 80

DEDITEC
GMBH

Configuration In-/Outputs Custom Logout

Product name: RO-ETH (DEDITEC RO-ETH-Module) User: admin / Session ends in: 355

General
Digital I/O
Digital Inputs
Digital Inputs Counter
Digital Outputs
Analog I/O
Configuration

Digital Inputs

I/O Access to the module: ☐ Activity
Status from modul: OK

Select channel area: CH 0..15

CH	Name	State
0	Digital Input 1	OFF ON
1	Digital Input 2	OFF ON
2	Digital Input 3	OFF ON
3	Digital Input 4	OFF ON
4	Digital Input 5	OFF ON
5	Digital Input 6	OFF ON
6	Digital Input 7	OFF ON
7	Digital Input 8	OFF ON
8	Digital Input 9	OFF ON
9	Digital Input 10	OFF ON
10	Digital Input 11	OFF ON
11	Digital Input 12	OFF ON
12	Digital Input 13	OFF ON
13	Digital Input 14	OFF ON
14	Digital Input 15	OFF ON
15	Digital Input 16	OFF ON

EDIT CH NAMES

Auf dieser Seite werden in regelmäßigen Abständen die Eingänge des Moduls gelesen.

Select channel area

Hier kann der angezeigte Kanal-Bereich (16er Blöcke) festgelegt werden.

State

Zustand der einzelnen Eingänge

Edit CH names

Zur besseren Übersicht, kann hier jedem Kanal ein Name zugeordnet werden.

1.5.2.3. Digitale Eingänge Zähler

Infohotline
+(49) 0 22 32 50 40 80

DEDITEC
GMBH

Configuration In-/Outputs Custom Logout

Product name: RO-ETH (DEDITEC RO-ETH-Module) User: admin / Session ends in: 336

General
Digital I/O
Digital Inputs
Digital Inputs Counter
Digital Outputs
Analog I/O
Configuration

Digital Inputs Counter

IO Access to the module: ☒ Activity
Status from modul: OK

Select channel area: CH 0..15

CH	Name	Counter value
0	Digital Input 1	1
1	Digital Input 2	1
2	Digital Input 3	1
3	Digital Input 4	0
4	Digital Input 5	1
5	Digital Input 6	1
6	Digital Input 7	1
7	Digital Input 8	0
8	Digital Input 9	0
9	Digital Input 10	0
10	Digital Input 11	0
11	Digital Input 12	0
12	Digital Input 13	0
13	Digital Input 14	0
14	Digital Input 15	0
15	Digital Input 16	0

EDIT CH NAMES RESET

Auf dieser Seite werden in regelmäßigen Abständen die Eingangszähler des Moduls gelesen.

Select channel area

Hier kann der angezeigte Kanal-Bereich (16er Blöcke) festgelegt werden.

Counter value

Aktueller Stand der Eingangszähler

Edit CH names

Zur besseren Übersicht, kann hier jedem Kanal ein Name zugeordnet werden.

Reset

Resettet alle Eingangszähler des aktuellen Kanal-Bereichs

1.5.2.4. Digitale Ausgänge

Infohotline
+(49) 0 22 32 50 40 80

DEDITEC
GMBH

Configuration In-/Outputs Custom Logout

Product name: RO-ETH (DEDITEC RO-ETH-Module) User: admin / Session ends in: 295

General
Digital I/O
Digital Inputs
Digital Inputs Counter
Digital Outputs
Analog I/O
Configuration

Digital outputs

I/O Access to the module ☒ Activity
Status from modul: OK

Select channel area CH 0..15

CH	Name	State	Readback
0	Digital Output 1	OFF ON	OFF ON
1	Digital Output 2	OFF ON	OFF ON
2	Digital Output 3	OFF ON	OFF ON
3	Digital Output 4	OFF ON	OFF ON
4	Digital Output 5	OFF ON	OFF ON
5	Digital Output 6	OFF ON	OFF ON
6	Digital Output 7	OFF ON	OFF ON
7	Digital Output 8	OFF ON	OFF ON
8	Digital Output 9	OFF ON	OFF ON
9	Digital Output 10	OFF ON	OFF ON
10	Digital Output 11	OFF ON	OFF ON
11	Digital Output 12	OFF ON	OFF ON
12	Digital Output 13	OFF ON	OFF ON
13	Digital Output 14	OFF ON	OFF ON
14	Digital Output 15	OFF ON	OFF ON
15	Digital Output 16	OFF ON	OFF ON

EDIT CH NAMES ALL OFF ALL ON

Auf dieser Seite werden in regelmäßigen Abständen die Ausgänge des Moduls zurückgelesen. Zusätzlich kann jeder Kanal einzeln oder der aktuelle Kanal-Bereich ein-/ausgeschaltet werden.

Select channel area

Hier kann der angezeigte Kanal-Bereich (16er Blöcke) festgelegt werden.

State

Diesen Kanal oder alle Kanäle ein-/ausschalten

Readback

Aktueller Zustand des Ausgangs

Edit CH names

Zur besseren Übersicht, kann hier jedem Kanal ein Name zugeordnet werden.

1.5.2.5. Analoge Eingänge

The screenshot shows the DEDITEC web interface for the RO-ETH module. The top header includes the contact number +49 0 22 32 50 40 80 and the DEDITEC logo. The navigation bar has tabs for Configuration, In-/Outputs, Custom, and Logout. The main content area is titled 'Product name: RO-ETH (DEDITEC RO-ETH-Module)' and 'User: admin / Session ends in: 212'. On the left, a sidebar menu lists 'General', 'Digital I/O', 'Analog I/O', 'Analog Inputs', 'Analog Outputs', and 'Configuration'. The 'Analog Inputs' section is active, showing a table of 16 channels. Above the table, there are settings for 'I/O Access to the module' (Activity), 'Status from modul' (OK), 'Select channel area' (CH 0..15), and 'A/D Mode' (0..10 V). At the bottom of the table is an 'EDIT CH NAMES' button.

CH	Name	Value
0	Analog Input 1	0.000 V
1	Analog Input 2	0.000 V
2	Analog Input 3	0.000 V
3	Analog Input 4	0.000 V
4	Analog Input 5	0.000 V
5	Analog Input 6	0.000 V
6	Analog Input 7	0.000 V
7	Analog Input 8	0.000 V
8	Analog Input 9	0.000 V
9	Analog Input 10	0.000 V
10	Analog Input 11	0.000 V
11	Analog Input 12	0.000 V
12	Analog Input 13	0.000 V
13	Analog Input 14	0.000 V
14	Analog Input 15	0.000 V
15	Analog Input 16	0.000 V

Auf dieser Seite werden in regelmäßigen Abständen die analogen Eingänge des Moduls gelesen.

Select channel area

Hier kann der angezeigte Kanal-Bereich (16er Blöcke) festgelegt werden.

A/D Mode

Aktueller A/D-Modus des Kanal-Bereichs

Edit CH names

Zur besseren Übersicht, kann hier jedem Kanal ein Name zugeordnet werden.

Value

Aktueller Analog Wert

1.5.2.6. Analoge Ausgänge

Infoc hotline
+(49) 0 22 32 50 40 80

DEDITEC GMBH

Configuration In-/Outputs Custom Logout

Product name: RO-ETH (DEDITEC RO-ETH-Module) User: admin / Session ends in: 239

General
Digital I/O
Analog I/O
Analog Inputs
Analog Outputs
Configuration

Analog Outputs

I/O Access to the module ☒ Activity
Status from modul: OK

Select channel area: CH 0..3

D/A Mode: +/-10 V +/-10 V

CH	Name	Value	Set value	Readback
0	Analog Output 1	1.23 V	SET	1.230 V
1	Analog Output 2	4.56 V	SET	4.560 V
2	Analog Output 3	7.89 V	SET	7.890 V
3	Analog Output 4	9.87 V	SET	9.870 V

EDIT CH NAMES

Auf dieser Seite können analoge Ausgänge gesetzt werden.

Select channel area

Hier kann der angezeigte Kanal-Bereich (16er Blöcke) festgelegt werden.

D/A Mode

Aktueller D/A-Modus des Kanal-Bereichs

Edit CH names

Zur besseren Übersicht, kann hier jedem Kanal ein Name zugeordnet werden.

Value

Analog Wert, der ausgegeben werden soll.

Readback

Aktueller Analog Wert

1.5.2.7. Konfiguration

Infohotline
+(49) 0 22 32 50 40 80

DEDITEC GMBH

Configuration In-/Outputs Custom Logout

Product name: RO-ETH (DEDITEC RO-ETH) User: admin / Session ends in: 1225

General
Digital I/O
Analog I/O
Temperature
Stepper Motor
Configuration

Configuration

Status from modul: OK

Select I/O type: Digital Input
Select channel area: CH 0..7

CH	Name
0	Digital Input 01
1	Digital Input 02
2	Digital Input 03
3	Digital Input 04
4	Digital Input 05
5	Digital Input 06
6	Digital Input 07
7	Digital Input 08

SAVE

Auf dieser Seite können sämtliche Namen der angeschlossenen I/O-Einheiten bearbeitet werden.

Select I/O type

Hierüber kann der I/O-Typ ausgewählt werden.

Select channel area

Sind mehr Ein-/Ausgänge angeschlossen, als auf dieser Form dargestellt werden kann, kann hierüber der Kanalbereich ausgewählt werden.

Save

Hiermit werden die Namen für diesen Kanalbereich gespeichert.

Hinweis:

Der Kanalname darf maximal 16 Zeichen lang sein.



DELIB API Referenz



2. DELIB API Referenz

2.1. 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_REL AIS_8	14
USB_OPTOIN_8_REL AIS_8	15
USB_OPTOIN_16_REL AIS_16	16
USB_OPTOIN_32	16
USB_REL AIS_32	16
USB_OPTOIN_32_REL AIS_32	17
USB_OPTOIN_64	17
USB_REL AIS_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

2.2. Verwaltungsfunktionen

2.2.1. 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;
}
```

2.2.2. 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);
```

2.2.3. 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)
```

2.2.4. 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
```


2.2.5. 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);
```

2.2.6. 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;  
}
```

2.3. Fehlerbehandlung

2.3.1. 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;
}
```

2.3.2. 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;
}
```

2.3.3. 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;
}
```

2.3.4. 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;
}
```

2.3.5. 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;
}
```


2.4. Digitale Eingänge lesen

2.4.1. 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)

2.4.2. 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

2.4.3. 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

2.4.4. 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();
```

2.4.5. 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

2.4.6. 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

2.4.7. 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
```

2.4.8. 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);
```

2.4.9. 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);
```


2.4.10. 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
```

2.4.11. 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.
```

2.4.12. 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
```

2.4.13. 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.
```

2.5. Digitale Ausgänge verwalten

2.5.1. 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

2.5.2. 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

2.5.3. 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

2.5.4. 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();
```

2.5.5. 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

2.5.6. 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
```

2.5.7. 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.

2.5.8. 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.

2.5.9. 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);
```

2.5.10. 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);
```

2.6. Digitale Zähler Funktionen für RO-Counter Module

2.6.1. DapiCnt48ModeSet

Beschreibung

Dieser Befehl setzt einen Zählmodus (optional auch Submodus) und Eingangsfiler für einen bestimmten Eingangszählerkanal.

Definition

```
void DapiCnt48ModeSet(ULONG handle, ULONG ch, ULONG mode);
```

Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Nummer des Eingangszählerkanals, dessen Modus gesetzt werden soll (0, 1, 2, 3, ..)

mode = Gibt den Modus an

Übersicht verfügbare Zähler Modi

Mode	Wert[hex]	CNT8	CNT/IGR
DAPI_CNT48_MODE_COUNT_RISING_EDGE	0	X	X
DAPI_CNT48_MODE_COUNT_RISING_EDGE_X2	1		X
DAPI_CNT48_MODE_COUNT_RISING_EDGE_X4	2		X
DAPI_CNT48_MODE_T	D	X	
DAPI_CNT48_MODE_FREQUENCY	E	X	
DAPI_CNT48_MODE_PWM	F	X	

Submode	Wert[hex]	HW-Reset verfügbar
DAPI_CNT48_SUBMODE_NO_RESET	0	x
DAPI_CNT48_SUBMODE_RESET_WITH_READ	1	x

Submode	Wert[hex]	HW-Reset verfügbar
DAPI_CNT48_SUBMODE_NO_RESET	2	
DAPI_CNT48_SUBMODE_RESET_WITH_READ	3	

Mögliche Werte für mode

*mode=DAPI_CNT48_MODE_COUNT_RISING_EDGE |
DAPI_CNT48_SUBMODE_NO_RESET*

In diesem Modus wird bei steigender Flanke gezählt.

*mode=DAPI_CNT48_MODE_COUNT_RISING_EDGE |
DAPI_CNT48_SUBMODE_RESET_WITH_READ*

In diesem Modus wird bei steigender Flanke gezählt. Zusätzlich wird bei jedem Lesevorgang der Zähler resettet.

*mode=DAPI_CNT48_MODE_COUNT_RISING_EDGE |
DAPI_CNT48_SUBMODE_RESET_ON_CH_7*

In diesem Modus wird bei steigender Flanke gezählt. Zusätzlich kann der Zähler über ein externes Signal (letzter Kanal des Moduls = 1) resettet werden.

*mode=DAPI_CNT48_MODE_COUNT_RISING_EDGE |
DAPI_CNT48_SUBMODE_LATCH_COMMON*

Mit dem Befehl "**DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48, DAPI_SPECIAL_CNT48_LATCH_GROUP8, 0, 0)**" werden alle 8 Zählerstände der Eingangszähler gleichzeitig in einen Zwischenspeicher (Latch) geschrieben. Mit diesem Modus kann dann der gelatchte Zählerstand ausgelesen werden.

mode=DAPI_CNT48_MODE_T

Mit diesem Modus wird die Periodendauer T gemessen. Als Basis hierbei dient ein 100 MHz Zähler.

mode=DAPI_CNT48_MODE_FREQUENCY

Bei diesem Modus lässt sich die Anzahl der steigenden Flanken innerhalb einer Sekunde (=Frequenz) messen.

mode=DAPI_CNT48_MODE_PWM

Mit diesem Modus werden die "high" und "low" Zeit eines Signals gemessen. Dadurch kann anschließend das Verhältnis bestimmt werden (PWM).

Zusätzlich können alle Eingangszähler mit einem Eingangsfiler (mit einer oder Verknüpfung) kombiniert werden. Hierzu stehen folgende Eingangsfiler zur Verfügung:

DAPI_CNT48_FILTER_20ns

DAPI_CNT48_FILTER_100ns

DAPI_CNT48_FILTER_250ns

DAPI_CNT48_FILTER_500ns

DAPI_CNT48_FILTER_1us

DAPI_CNT48_FILTER_2_5us

DAPI_CNT48_FILTER_5us

DAPI_CNT48_FILTER_10us

DAPI_CNT48_FILTER_25us

DAPI_CNT48_FILTER_50us

DAPI_CNT48_FILTER_100us

DAPI_CNT48_FILTER_250us

DAPI_CNT48_FILTER_500us

DAPI_CNT48_FILTER_1ms

DAPI_CNT48_FILTER_2_5ms

DAPI_CNT48_FILTER_5ms

Return-Wert

Keiner

Bemerkung

Dieser Befehl wird nur von unserem Modul RO-CNT8 unterstützt.

Programmierbeispiel

```
DapiCnt48ModeSet(handle, 0, DAPI_CNT48_MODE_COUNT_RISING_EDGE |  
DAPI_CNT48_SUBMODE_RESET_WITH_READ | DAPI_CNT48_FILTER_20ns);  
//Eingangszählerkanal 0 zählt alle Impulse <= 20ns bei steigender Flanke.  
Zusätzlich wird der Zähler nach Abfrage resettet.  
DapiCnt48ModeSet(handle, 1, DAPI_CNT48_MODE_COUNT_RISING_EDGE |  
DAPI_CNT48_SUBMODE_RESET_ON_CH_7 | DAPI_CNT48_FILTER_500us);  
//Eingangszählerkanal 1 zählt alle Impulse <= 500us bei steigender Flanke.  
Dieser Zähler kann mit einem externen Signal (ch7 = 1) resettet werden.  
DapiCnt48ModeSet(handle, 2, DAPI_CNT48_MODE_PWM |  
DAPI_CNT48_FILTER_5ms);  
//Eingangszählerkanal 2 misst alle low-/high Zeiten <= 5ms. Anschließend  
wird das Verhältnis bestimmt (PWM).
```

2.6.2. DapiCnt48ModeGet

Beschreibung

Dieser Befehl liest den Zählmodus eines bestimmten Eingangszählerkanals zurück.

Definition

ULONG DapiCnt48ModeGet(ULONG handle, ULONG ch);

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Nummer des Eingangszählerkanals, dessen Modus ausgegeben werden soll
(0, 1, 2, 3, ..)

Return-Wert

Zählmodus des Eingangszählerkanals.

(Nähere Informationen / Beschreibung der Bits -> siehe delib.h oder Manual "RO-Registerbelegung")

Bemerkung

Dieser Befehl wird nur von unserem Modul RO-CNT8 unterstützt.

Programmierbeispiel

```
value = DapiCnt48ModeGet(handle, 0)
//Gibt den Zählmodus von Eingangszählerkanal 0 zurück
value = DapiCnt48ModeGet(handle, 3)
//Gibt den Zählmodus von Eingangszählerkanal 3 zurück
```

2.6.3. DapiCnt48CounterGet32

Beschreibung

Dieser Befehl liest die ersten 32 Bit eines 48 Bit Eingangszählers.

Definition

ULONG DapiCnt48CounterGet32(ULONG handle, ULONG ch);

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

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

Return-Wert

Ausgabe des Zählerwertes.

Bemerkung

Dieser Befehl wird nur von unseren Modulen RO-CNT8 und RO-CNT/IGR unterstützt.

Programmierbeispiel

```
value = DapiCnt48CounterGet32(handle, 0);  
//gibt den Wert von Eingangszählerkanal 0 aus  
value = DapiCnt48CounterGet32(handle, 3);  
//gibt den Wert von Eingangszählerkanal 3 aus
```

2.6.4. DapiCnt48CounterGet48

Beschreibung

Dieser Befehl liest einen 48 Bit Zähler eines Eingangszählerkanals.

Definition

ULONGLONG DapiCnt48CounterGet48(ULONG handle, ULONG ch);

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

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

Return-Wert

Ausgabe des Zählerwertes.

Bemerkung

Dieser Befehl wird nur von unseren Modulen RO-CNT8 und RO-CNT/IGR unterstützt.

Programmierbeispiel

```
value = DapiCnt48CounterGet48(handle, 0);  
//gibt den Wert von Eingangszählerkanal 0 aus  
value = DapiCnt48CounterGet48(handle, 3);  
//gibt den Wert von Eingangszählerkanal 3 aus
```

2.6.5. DapiSpecialCNT48ResetSingle

Beschreibung

Dieser Befehl resettet den Zählerstand eines einzelnen Eingangszählers.

Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_CNT48,  
DAPI_SPECIAL_CNT48_RESET_SINGLE, ULONG ch, 0)
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangszählers an, dessen Zählerstand resettet werden soll (0, 1, 2, ..)

Return-Wert

Keiner

Bemerkung

Dieser Befehl wird nur von unserem Modul RO-CNT8 unterstützt.

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48,  
DAPI_SPECIAL_CNT48_RESET_SINGLE, 0, 0)  
// Zählerstand von Eingangszähler 0 wird resettet  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48,  
DAPI_SPECIAL_CNT48_RESET_SINGLE, 1, 0)  
// Zählerstand von Eingangszähler 1 wird resettet
```

2.6.6. DapiSpecialCNT48ResetGroup8

Beschreibung

Dieser Befehl resettet gleichzeitig die Zählerstände von 8 Eingangszählern.

Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_CNT48,  
DAPI_SPECIAL_CNT48_RESET_GROUP8, ULONG ch, 0)
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangszählers an, ab dem die Zählerstände von 8 Eingangszählern resettet werden (0, 8, 16, ...)

Return-Wert

Keiner

Bemerkung

Dieser Befehl wird nur von unserem Modul RO-CNT8 unterstützt.

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48,  
DAPI_SPECIAL_CNT48_RESET_GROUP8, 0, 0)  
// Zählerstände der Eingangszähler 0-7 werden resettet  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48,  
DAPI_SPECIAL_CNT48_RESET_GROUP8, 8, 0)  
// Zählerstände der Eingangszähler 8-15 werden resettet
```

2.6.7. DapiSpecialCNT48LatchGroup8

Beschreibung

Dieser Befehl speichert die Zählerstände von 8 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 (langsam) einzeln nacheinander ausgelesen werden können.

Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_CNT48,  
    DAPI_SPECIAL_CNT48_LATCH_GROUP8, ULONG ch, 0)
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangszählers an, ab dem die Zählerstände von 8 Eingangszählern gelatched werden (0, 8, 16, ...)

Return-Wert

Keiner

Bemerkung

Dieser Befehl wird nur von unserem Modul RO-CNT8 unterstützt.

Hinweis

Bitte beachten Sie, dass nur die Zählerstände der Eingangszähler gelatched werden,

bei denen zuvor der Modus "DAPI_CNT48_SUBMODE_LATCH_COMMON" gesetzt wurde. (-> **DapiCnt48ModeSet**)

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48,  
    DAPI_SPECIAL_CNT48_LATCH_GROUP8, 0, 0)  
// Zählerstände der Eingangszähler 0-7 werden gelatched  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_CNT48,  
    DAPI_SPECIAL_CNT48_LATCH_GROUP8, 8, 0)  
// Zählerstände der Eingangszähler 8-15 werden gelatched
```


2.6.8. DapiSpecialCNT48DIGet1

Beschreibung

Dieser Befehl liest den Eingangszustand (0/1) eines digitalen Eingangszählerkanals.

Definition

ULONG DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_CNT48, DAPI_SPECIAL_CNT48_DI_GET1, ULONG ch, 0)

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Eingangszählerkanals an, dessen Eingangszustand gelesen werden soll (0, 1, 2, 3, ..)

Return-Wert

Zustand des Eingangszählers (0/1)

Bemerkung

Dieser Befehl wird nur von unserem Modul RO-CNT8 unterstützt.

Programmierbeispiel

```
value = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_CNT48, DAPI_SPECIAL_CNT48_DI_GET1, 0,
0)
// Liest Eingangszustand von Eingangszählerkanal 1
value = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_CNT48, DAPI_SPECIAL_CNT48_DI_GET1, 1,
0)
// Liest Eingangszustand von Eingangszählerkanal 2
```

2.7. PWM Funktionen

2.7.1. DapiPWMOutSet

Beschreibung

Dieser Befehl setzt das PWM Verhältnis eines PWM-Kanals

Definition

```
void DapiPWMOutSet(ULONG handle, ULONG ch, float data);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, der gesetzt werden soll

data=PWM-Verhältnis in von 0% bis 100% in 1% Schritten

Kleinstes PWM-Verhältnis ist abhängig von der PWM-Frequenz

10Hz data muss $\geq 0\%$ sein

100Hz data muss $\geq 2\%$ sein

250Hz data muss $\geq 3\%$ sein

1000Hz data muss $\geq 9\%$ sein

Return-Wert

Keiner

Programmierbeispiel

```
DapiPWMOutSet(handle, 0, 50);  
// Setzt das PWM Verhältnis des ersten Kanals auf 50% (50% high, 50% low)  
DapiPWMOutSet(handle, 1, 100);  
// Setzt das PWM Verhältnis des zweiten Kanals auf 100% (100% high, 0%  
low)
```

2.7.2. DapiPWMOutReadback

Beschreibung

Dieser Befehl liest das PWM-Verhältnis eines PWM-Kanals

Definition

float DapiPWMOutReadback(ULONG handle, ULONG ch);

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt die Nummer des Ausgangs an, der gelesen werden soll

Return-Wert

PWM Verhältnis des Kanals von 0% bis 100%

Programmierbeispiel

```
float data = DapiPWMOutReadback(handle, 0);  
// Liest das PWM Verhältnis des ersten Kanals  
float data = DapiPWMOutReadback(handle, 2);  
// Liest das PWM Verhältnis des zweiten Kanals
```

2.7.3. DAPI_SPECIAL_PWM_FREQ_SET

Beschreibung

Dieser Befehl setzt die PWM Frequenz des Moduls

Definition

```
void DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_PWM, cmd, par1, par2);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

cmd=DAPI_SPECIAL_PWM_FREQ_SET

par1=channel area 0 (ch 0-15), 16 (ch 16-31) ... usw.

par2=Frequenz = DAPI_PWM_FREQUENCY_10HZ,
DAPI_PWM_FREQUENCY_100HZ, DAPI_PWM_FREQUENCY_250HZ oder
DAPI_PWM_FREQUENCY_1000Hz

Return-Wert

Keiner

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_PWM,  
DAPI_SPECIAL_PWM_FREQ_SET, 0,  
DAPI_PWM_FREQUENCY_100HZ);  
// Setzt die PWM Frequenz des Moduls auf 100Hz
```

2.7.4. DAPI_SPECIAL_PWM_FREQ_READBACK

Beschreibung

Dieser Befehl liest die aktuelle PWM Frequenz des Moduls

Definition

```
void DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_PWM, cmd, par1,  
par2);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

cmd=DAPI_SPECIAL_PWM_FREQ_READBACK

par1=0

par2=0

Return-Wert

uint = DAPI_PWM_FREQUENCY_10HZ, DAPI_PWM_FREQUENCY_100HZ,
DAPI_PWM_FREQUENCY_250HZ oder DAPI_PWM_FREQUENCY_1000Hz

Programmierbeispiel

```
uint frequency = DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_PWM, DAPI_SPECIAL_PWM_FREQ_READBACK,  
0, 0);  
// Liest die PWM Frequenz des Moduls
```

2.8. Pulsgenerator Ausgänge verwalten

2.8.1. DapiPulseGenSet

Beschreibung

Dieser Befehl generiert eine gewisse Anzahl an Impulsen mit vorgegebenen low- und high-Zeiten.

Er wird nur von unseren RO-CNT8 Modulen unterstützt.

Definition

```
void DapiPulseGenSet(ULONG handle, ULONG ch, ULONG mode, ULONG par0,  
    ULONG par1, ULONG par2);
```

Parameter

handle = Dies ist das Handle eines geöffneten Moduls

ch = Gibt die Nummer des zu setzenden Ausgangs an (0, 1, 2, ..)

mode = Modus, mit dem Impulse generiert werden (muss immer 0 sein).

par0 = Anzahl der zu erzeugenden Impulse (par0 = 0 -> unendlich viele Impulse)

par1 = Low-Zeit des Impulses ($t[\text{ns}] / 10 - 1$)

par2 = High-Zeit des Impulses ($t[\text{ns}] / 10 - 1$)

Beispiel für das Einstellen der low-/high-Zeit (Par1/Par2)

500ns -> $500 / 10 - 1 = 49(\text{dez})$

7µ -> $7000 / 10 - 1 = 699(\text{dez})$

2,5ms -> $2500000 / 10 - 1 = 249,999(\text{dez})$

Return-Wert

keiner

Programmierbeispiel

```
DapiPulseGenSet(handle, 0, 0, 10, 29, 69);  
// generiert 10 Impulse an Pulsgenerator-Ausgang 0 mit einer low-Zeit von  
300ns und einer high-Zeit von 700ns.  
DapiPulseGenSet(handle, 3, 0, 100, 7999, 9999);  
// generiert 100 Impulse an Pulsgenerator-Ausgang 3 mit einer low-Zeit von  
80µs und einer high-Zeit von 100µs.  
DapiPulseGenSet(handle, 10, 0, 0, 249999, 299999);
```

// generiert unendlich viele Impulse an Pulsgenerator-Ausgang 10 mit einer low-Zeit von 2,5ms und einer high-Zeit von 3ms.

2.9. A/D Wandler Funktionen

2.9.1. DapiADSetMode

Beschreibung

Dieser Befehl setzt den Modus für einen D/A Wandler.

Definition

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

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0 ..)

mode=Gibt den Modus für den D/A Wandler an (siehe delib.h)

Return-Wert

Keiner

Bemerkung

Folgende Modi werden unterstützt:

(diese sind abhängig von dem verwendeten D/A-Modul)

Unipolare Spannungen:

Modus	Wertebereich
ADDA_MODE_UNIPOL_10V	0V .. 10V
ADDA_MODE_UNIPOL_5V	0V .. 5V
ADDA_MODE_UNIPOL_2V5	0V .. 2,5V

Bipolare Spannungen:

Modus	Wertebereich
ADDA_MODE_BIPOL_10V	-10V .. +10V
ADDA_MODE_BIPOL_5V	-5V .. +5V
ADDA_MODE_BIPOL_2V5	-2,5V .. +2,5V

Ströme:

Modus	Wertebereich
ADDA_MODE_0_20mA	0 .. 20 mA
ADDA_MODE_4_20mA	4 .. 20 mA
ADDA_MODE_0_24mA	0 .. 24 mA
ADDA_MODE_0_25mA	0 .. 25 mA
ADDA_MODE_0_50mA	0 .. 50 mA

2.9.2. DapiADGetMode

Beschreibung

Dieser Befehl liest den eingestellten Modus eines A/D Wandlers zurück. Modus-Beschreibung siehe DapiADSetMode.

Definition

ULONG DapiADGetMode(ULONG handle, ULONG ch);

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des A/D Wandlers an (0 ..)

Return-Wert

Modus des A/D Wandlers

2.9.3. DapiADGet

Beschreibung

Dieser Befehl liest einen Datenwert von einen Kanal eines A/D Wandlers.

Definition

ULONG DapiADGet(ULONG handle, ULONG ch);

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des A/D Wandlers an (0 ..)

Return-Wert

Wert vom A/D Wandler in Digits

2.9.4. DapiADGetVolt

Beschreibung

Dieser Befehl liest einen Datenwert von einen Kanal eines A/D Wandlers in Volt.

Definition

```
float DapiADGetVolt(ULONG handle, ULONG ch);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des A/D Wandlers an (0 ..)

Return-Wert

Wert vom A/D Wandler in Volt

2.9.5. DapiADGetmA

Beschreibung

Dieser Befehl liest einen Datenwert von einen Kanal eines A/D Wandlers in mA.

Definition

```
float DapiADGetmA(ULONG handle, ULONG ch);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des A/D Wandlers an (0 ..)

Return-Wert

Wert vom A/D Wandler in mA.

Bemerkung

Dieser Befehl ist Modul abhängig. Er funktioniert natürlich nur, wenn das Modul auch den Strom-Modus unterstützt.

2.9.6. DapiSpecialADReadMultipleAD

Beschreibung

Dieser Befehl speichert die Werte bestimmter, benachbarter Kanäle eines A/D Wandlers gleichzeitig in einen Zwischenpuffer. So können anschließend die Werte nacheinander ausgelesen werden.

Vorteil hierbei ist, dass die A/D-Werte zum Einen gleichzeitig gepuffert werden, zum Anderen können die Werte mehrerer AD-Kanäle (im Vergleich zu den Befehlen DapiADGetVolt, DapiADGetmA oder DapiADGet) anschließend deutlich schneller abgefragt werden.

Definition

```
void DapiSpecialCommand(ULONG handle, DAPI_SPECIAL_CMD_AD,  
    DAPI_SPECIAL_AD_READ_MULTIPLE_AD, ULONG start_ch, ULONG end_ch);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls.

start_ch=Gibt den Start-Kanal des A/D Wandlers an, ab dem die Werte gepuffert werden (0, 1, 2, ..).

end_ch=Gibt den End-Kanal des A/D Wandlers an, bis zu dem die Werte gepuffert werden (0, 1, 2, ..).

Return-Wert

Keiner.

Bemerkung

Die Werte, die mit Befehl DapiSpecialADReadMultipleAD gepuffert wurden, können anschließend mit den Befehlen DapiADGetVolt, DapiADGetmA oder DapiADGet gelesen werden. Damit auch wirklich der gepufferte Wert gelesen wird, muss bei diesen Funktionen der Parameter "ch" mit 0x8000 logisch "oder" verknüpft werden (siehe Beispiele).

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_AD,  
DAPI_SPECIAL_AD_READ_MULTIPLE_AD, 0, 15);  
// Puffert die Werte von AD-Kanal 0..15  
  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_AD,  
DAPI_SPECIAL_AD_READ_MULTIPLE_AD, 0, 63);  
// Puffert die Werte von AD-Kanal 0..63  
  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_AD,  
DAPI_SPECIAL_AD_READ_MULTIPLE_AD, 16, 31);  
// Puffert die Werte von AD-Kanal 16..31  
  
value = DapiADGetVolt(handle, 0x8000 | 0);  
// Gibt den gepufferten Wert von AD-Kanal 0 in Volt zurück.  
  
value = DapiADGetmA(handle, 0x8000 | 15);  
// Gibt den gepufferten Wert von AD-Kanal 15 in mA zurück.  
  
value = DapiADGet(handle, 0x8000 | 63);  
// Gibt den gepufferten Wert von AD-Kanal 63 in Digits zurück.
```

2.10. D/A Ausgänge verwalten

2.10.1. DapiDASetMode

Beschreibung

Dieser Befehl setzt den Modus für einen D/A Wandler.

Definition

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

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0 ..)

mode=Gibt den Modus für den D/A Wandler an (siehe delib.h)

Return-Wert

Keiner

Bemerkung

Folgende Modi werden unterstützt:

(diese sind abhängig von dem verwendeten D/A-Modul)

Unipolare Spannungen:

Modus	Wertebereich
ADDA_MODE_UNIPOL_10V	0V .. 10V
ADDA_MODE_UNIPOL_5V	0V .. 5V
ADDA_MODE_UNIPOL_2V5	0V .. 2,5V

Bipolare Spannungen:

Modus	Wertebereich
ADDA_MODE_BIPOL_10V	-10V .. +10V
ADDA_MODE_BIPOL_5V	-5V .. +5V
ADDA_MODE_BIPOL_2V5	-2,5V .. +2,5V

Ströme:

Modus	Wertebereich
ADDA_MODE_0_20mA	0 .. 20 mA
ADDA_MODE_4_20mA	4 .. 20 mA
ADDA_MODE_0_24mA	0 .. 24 mA
ADDA_MODE_0_25mA	0 .. 25 mA
ADDA_MODE_0_50mA	0 .. 50 mA

2.10.2. DapiDAGetMode

Beschreibung

Dieser Befehl liest den eingestellten Modus eines D/A Wandlers zurück.

Definition

```
ULONG DapiDAGetMode(ULONG handle, ULONG ch);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0 ..)

Return-Wert

Modus des D/A Wandlers

2.10.3. DapiDASet

Beschreibung

Dieser Befehl übergibt ein Datenwert an einen Kanal eines 16-Bit D/A Wandlers.

Definition

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

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0 ..)

data=Gibt den Datenwert an, der geschrieben wird (16-Bit Datenwert -> Datenwertebereich: 0-65535)

Return-Wert

Keiner

Programmierbeispiel

```
DapiDASet(handle, 0, 65535);  
// Setzt den 1. Ausgang des D/A Wandlers auf maximalen Wert des  
gewählten Modus.  
//(bei ausgewähltem Modus ADDA_MODE_UNIPOL_10V wird der 1. Ausgang  
des D/A Wandlers auf 10V gesetzt)
```

2.10.4. DapiDASetVolt

Beschreibung

Dieser Befehl setzt eine Spannung an einen Kanal eines D/A Wandlers.

Definition

```
void DapiDASetVolt(ULONG handle, ULONG ch, float data);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0 ..)

data=Gibt die Spannung an, die eingestellt werden soll [V]

Return-Wert

Keiner

Programmierbeispiel

```
DapiDASetVolt(handle, 0, 5,4321);  
// Setzt den 1. Ausgang des D/A Wandlers auf 5,4321 V
```

2.10.5. DapiDASetmA

Beschreibung

Dieser Befehl setzt einen Strom an einen Kanal eines D/A Wandlers.

Definition

```
void DapiDASetmA(ULONG handle, ULONG ch, float data);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0 ..)

data=Gibt den Strom an, der geschrieben wird [mA]

Return-Wert

Keiner

Bemerkung

Dieser Befehl ist Modul abhängig. Er funktioniert natürlich nur, wenn das Modul auch den Strom-Modus unterstützt.

2.10.6. DapiSpecialCmd_DA

Beschreibung

Dieser Befehl setzt die Spannungswerte bei einem Kanal beim Einschalten bzw. nach einem Timeout eines D/A Wandlers (EEPROM-Konfiguration).

Definition

```
void DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DA, cmd, ch, 0);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

ch=Gibt den Kanal des D/A Wandlers an (0, 1, 2, ..)

Zurücksetzen der Einstellungen auf Default Konfiguration

cmd=DAPI_SPECIAL_DA_PAR_DA_LOAD_DEFAULT

Speichern der Konfiguration in das EEPROM

cmd=DAPI_SPECIAL_DA_PAR_DA_SAVE_EEPROM_CONFIG

Laden der Konfiguration aus dem EEPROM

cmd=DAPI_SPECIAL_DA_PAR_DA_LOAD_EEPROM_CONFIG

Return-Wert

Keiner

Bemerkung

DAPI_SPECIAL_CMD_DA_PAR_DA_LOAD_DEFAULT

Mit diesem Befehl wird die Default Konfiguration eines D/A Wandlers geladen. Der D/A Wandler hat jetzt als Ausgangsspannung 0V.

DAPI_SPECIAL_DA_PAR_DA_SAVE_EEPROM_CONFIG

Mit diesem Befehl wird die aktuelle D/A Wandler Einstellung (Spannung/Strom-Wert, Enable/Disable und D/A Wandler Modus) in das EEPROM gespeichert.

DAPI_SPECIAL_DA_PAR_DA_LOAD_EEPROM_CONFIG

Mit diesem Befehl wird der D/A Wandler, mit der im EEPROM gespeicherten Konfiguration, gesetzt.

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DA,  
DAPI_SPECIAL_DA_PAR_DA_LOAD_DEFAULT, 1, 0);  
//Zurücksetzen der EEPROM-Konfiguration auf Default Konfiguration bei  
Kanal 1.  
  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DA,  
DAPI_SPECIAL_DA_PAR_DA_SAVE_EEPROM_CONFIG, 3, 0);  
//Speichern der D/A Wandler Einstellungen in das EEPROM bei Kanal 3.  
  
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_DA,  
DAPI_SPECIAL_DA_PAR_DA_LOAD_EEPROM_CONFIG, 2, 0);  
//Setzen des D/A Wandlers, mit der im EEPROM gespeicherten Konfiguration  
bei Kanal 2.
```

2.11. PT100 Funktionen

2.11.1. DapiTempGet

Beschreibung

Dieser Befehl liest einen Temperatur Eingang.

Definition

float DapiTempGet(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

Temperatur [°C]

Programmierbeispiel

```
ret=DapiTempGet(handle, 0)
//gibt die Temperatur von Kanal 0 zurück
```

2.12. TTL-Ein-/Ausgangs Richtungen setzen mit DapiSpecialCommand

2.12.1. 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
```

2.12.2. 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
```

2.12.3. 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
```

2.13. Watchdog Funktionen

2.13.1. DapiWatchdogEnable

Beschreibung

Diese Funktion aktiviert den Watchdog.

Definition

```
void DapiWatchdogEnable(ULONG handle);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

Return-Wert

Keiner

Programmierbeispiel

```
DapiWatchdogEnable(handle) ;  
//Aktiviert den Watchdog
```

2.13.2. DapiWatchdogDisable

Beschreibung

Diese Funktion deaktiviert den Watchdog.

Definition

```
void DapiWatchdogDisable(ULONG handle);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

Return-Wert

Keiner

Programmierbeispiel

```
DapiWatchdogDisable(handle) ;  
//Deaktiviert den Watchdog
```

2.13.3. DapiWatchdogRetrigger

Beschreibung

Diese Funktion retriggert den Watchdog-Timer.

Definition

```
void DapiWatchdogRetrigger(ULONG handle);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

Return-Wert

Keiner

Programmierbeispiel

```
DapiWatchdogRetrigger(handle) ;  
//Retriggert den Watchdog-Timer
```

2.14. Schrittmotoren Funktionen

2.14.1. Befehle mit DapiStepperCommand

2.14.1.1. DAPI_STEPPER_CMD_GO_POSITION

Beschreibung

Hiermit wird eine bestimmte Position angefahren. Dieses Kommando darf nur ausgeführt werden, wenn der Motor nicht "disabled" ist und kein Go_Position oder Go_Referenz ausgeführt wird.

Definition

DapiStepperCommand(handle, motor, DAPI_STEPPER_CMD_GO_POSITION, position, 0, 0, 0);

Programmierbeispiel

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GO_POSITION, go_pos_par, 0, 0, 0);
```

2.14.1.2. DAPI_STEPPER_CMD_GO_POSITION_RELATIVE

Beschreibung

Hiermit wird eine relative Position angefahren. Im Gegensatz zum Befehl GO_POSITION, der eine absolute Position anfährt, wird hier die momentane Position berücksichtigt. Dieses Kommando darf nur ausgeführt werden, wenn der Motor nicht "disabled" ist und kein Go_Position oder Go_Referenz ausgeführt wird.

Definition

```
void DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GO_POSITION_RELATIVE, go_pos_rel_par, 0, 0, 0);
```

Programmierbeispiel

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GO_POSITION_RELATIVE, 100, 0, 0, 0);  
//Motor fährt, von der aktuellen Position aus gesehen, 100 Schritte nach  
rechts.
```


2.14.1.3. DAPI_STEPPEER_CMD_SET_POSITION

Beschreibung

Dieses Kommando dient zum setzten der Motorposition. Die Auflösung beträgt 1/16 Vollschrirt. Dieses Kommando darf nur bei angehaltenem Motor verwendet werden.

Definition

DapiStepperCommand(handle, motor, DAPI_STEPPEER_CMD_SET_POSITION, par1, 0, 0, 0);

Parameter

par1 = Motorposition

2.14.1.4. DAPI_STEPPEER_CMD_SET_FREQUENCY

Beschreibung

Dieses Kommando dient zur Einstellung der Motorsollfrequenz. Die Motorfrequenzregelung übernimmt dabei die Einhaltung der Beschleunigungs- / Bremsrampe. Schrittverluste treten nicht auf. Die Motorsollfrequenz ist bezogen auf Vollschrittbetrieb. Über das Vorzeichen wird die Richtung ausgewählt. Die Motorsollfrequenz darf nicht über der Maxfrequenz liegen, ansonsten wird das Kommando abgelehnt.

Bei geschlossenem Endschalter1 läßt sich nur in positive Richtung verfahren, bei geschlossenem Endschalter2 läßt sich nur in negative Richtung verfahren, ansonsten wird das Kommando abgelehnt.

Definition

```
DapiStepperCommand(handle, motor, DAPI_STEPPEER_CMD_SET_FREQUENCY,  
par1, 0, 0, 0);
```

Parameter

par1 = Motorsollfrequenz [Hz]

2.14.1.5. DAPI_STEPPEER_CMD_GET_FREQUENCY

Beschreibung

Dieses Kommando dient zum Abfragen der Motorfrequenz. Dieses Kommando darf immer verwendet werden.

Definition

```
DapiStepperCommand(handle, motor, DAPI_STEPPEER_CMD_GET_FREQUENCY,  
par1, 0,0,0);
```

Return-Wert

Motorfrequenz [Hz]

2.14.1.6. DAPI_STEPPER_CMD_SET_FREQUENCY_DIRECTLY

Beschreibung

Dieses Kommando dient zur Einstellung der Motorfrequenz. Die Motorfrequenzregelung übernimmt dabei keine Funktion. Für die Einhaltung der Beschleunigungs- / Bremsrampe ist der Anwender verantwortlich. Schritverluste können bei Nichteinhaltung auftreten.

Die Motorfrequenz ist bezogen auf Vollschrittbetrieb. Über das Vorzeichen wird die Richtung ausgewählt.

Die Frequenz darf nicht über der Maxfrequenz liegen.

Definition

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_FREQUENCY_DIRECTLY, par1, 0,0,0);
```

Parameter

par1 = Motorfrequenz [Hz]

2.14.1.7. DAPI_STEPPEER_CMD_STOP

Beschreibung

Dieses Kommando dient zum Anhalten des Motors, die Bremsrampe wird dabei eingehalten.

Definition

DapiStepperCommand(handle, motor, DAPI_STEPPEER_CMD_STOP, 0, 0, 0, 0);

2.14.1.8. DAPI_STEPPEER_CMD_FULLSTOP

Beschreibung

Dieses Kommando dient zum sofortigen Anhalten des Motors, die Bremsrampe wird dabei nicht eingehalten. Die Motorposition kann vielleicht danach nicht mehr stimmen, da der Motor unkontrolliert angehalten wird.

Definition

DapiStepperCommand(handle, motor, DAPI_STEPPEER_CMD_FULLSTOP, 0, 0, 0, 0);

Programmierbeispiel

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPEER_CMD_FULLSTOP, 0, 0, 0, 0);
```

2.14.1.9. DAPI_STEPPEER_CMD_DISABLE

Beschreibung

Dieses Kommando dient zum deaktivieren/enaktivieren des Motors, der Motor verfährt dann nicht mehr/oder wieder. Dieses Kommando darf nur bei Motorstillstand benutzt werden.

Definition

```
DapiStepperCommand(handle, motor, DAPI_STEPPEER_CMD_DISABLE, par1, 0, 0, 0);
```

Parameter

par1 = Disablemode (0=Normale Funktion / 1=Disable)

2.14.1.10. DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC

Beschreibung

Hiermit werden neue Motor Konfigurationen gesetzt.

Definition

*DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC, par1, par2, 0, 0);*

Parameter

Parameter-Stepmode setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_STEPMODE

par2=0 (Vollschrittbetrieb)

par2=1 (Halbschrittbetrieb)

par2=2 (Viertelschrittbetrieb)

par2=3 (Achtelschrittbetrieb)

par2=4 (Sechzehntelschrittbetrieb)

Parameter-GO-Frequency setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOFREQUENCY

par2=Geschwindigkeit [Vollschritt / s] - bezogen auf Vollschritt Frequenz -
(Maximalwert=5000)

Parameter-Start-Frequency setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_STARTFREQUENCY

par2=Startfrequenz [Vollschritt / s] - bezogen auf Vollschritt Frequenz -
(Maximalwert=5000)

Parameter-Stop-Frequency setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_STOPFREQUENCY

par2=Stopfrequenz [Vollschritt / s] - bezogen auf Vollschritt Frequenz -
(Maximalwert=5000)

Parameter-Max-Frequency setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_MAXFREQUENCY

par2=Maximale Frequenz [Vollschritt / s] - bezogen auf Vollschritt Frequenz - (Maximalwert=5000)

Parameter-Accelerationslope setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_ACCELERATIONSLOPE

par2=Beschleunigungsrampe [Vollschritt / 10ms] - (Maximalwert=1000)

Parameter-Decelerationslope setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_DECELERATIONSLOPE

par2= Bremsrampe [Vollschritt / 10ms] - (Maximalwert=1000)

Parameter-Phasecurrent setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_PHASECURRENT

par2=Phasenstrom [mA] - (Maximalwert = 1500)

Parameter-Hold-Phasecurrent setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_HOLDPHASECURRENT

par2=Phasenstrom bei Motorstillstand [mA] - (Maximalwert=1500)

Parameter-Hold-Time setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_HOLDTIME

par2=Zeit in der der Haltestrom fließt nach Motorstop [ms]

par2=-1 / FFFF hex / 65535 dez (Zeit unendlich)

Parameter-Status-LED-Mode setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_STATUSLEDMODE

par2=Betriebsart der Status-LED

par2=0 = (MOVE - LED leuchtet bei Motorbewegung)

par2=1 = (HALT - LED leuchtet bei Motorstillstand)

par2=2 = (ENDSW1 - LED leuchtet bei geschlossenen Endschanter1)

par2=3 = (ENDSW2 - LED leuchtet bei geschlossenen Endschanter2)

par2=4 = (REFSW1 - LED leuchtet bei geschlossenen Referenzschalterschalter1)

par2=5 = (REFSW2 - LED leuchtet bei geschlossenen Referenzschalterschalter2)

Parameter-Invert-END-Switch1 setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW1

par2=Invertiere Funktion des Endschalter1 (0=normal / 1=invertieren)

Parameter-Invert-END-Switch2 setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW2

par2=Invertiere Funktion des Endschalter2 (0=normal / 1=invertieren)

Parameter-Invert-Ref-Switch1 setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW1

par2=Invertiere Funktion des Referenzschalterschalter1 (0=normal / 1=invertieren)

Parameter-Invert-Ref-Switch2 setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW2

par2=Invertiere Funktion des Referenzschalterschalter2 (0=normal / 1=invertieren)

Parameter-Invert-direction setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_DIRECTION

par2=Invertiere alle Richtungsangaben (0=normal / 1=invertieren)

Parameter-Endswitch-Stopmode setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_ENDSWITCH_STOPMODE

par2=Einstellen des Stopverhaltens (0=Fullstop / 1=Stop)

Parameter-GoReferenceFrequency setzen

(ACHTUNG: Dieser Parameter wird nicht mehr unterstützt!)

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY

Bemerkung

Dieser Parameter wird durch die nachfolgenden drei Parametern vollständig ersetzt.

Parameter-GoReferenceFrequencyToEndSwitch setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOENDSWITCH

par2=Geschwindigkeit, mit der der Enshalter angefahren wird (Frequenz [Vollschritt / s] - (Maximalwert=5000))

Parameter GoReferenceFrequencyAfterEndSwitch setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_AFTERE
NDSWITCH

par2=Geschwindigkeit, mit der vom Enschanter abgefahren wird (Frequenz
[Vollschritt / s] - (Maximalwert=5000))

Parameter GoReferenceFrequencyToOffset setzen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOOFFS
ET

par2=Geschwindigkeit, mit der der optionale Offset angefahren wird (Frequenz
[Vollschritt / s] - (Maximalwert=5000))

Programmierbeispiel

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_STEPMODE, 4,0,0);  
// Schrittmode (Voll-, Halb-, Viertel-, Achtel-, Sechszehntelschritt)  
  
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_GOFREQUENCY, 1000,0,0);  
// Schrittmode (Voll-, Halb-, Viertel-, Achtel-, Sechszehntelschritt)  
  
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_STARTFREQUENCY, 100,0,0);  
// Startfrequenz [Vollschritt / s]  
  
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_STOPFREQUENCY, 100,0,0);  
// Stopfrequenz [Vollschritt / s]  
  
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_MAXFREQUENCY, 3500,0,0);  
// maximale Frequenz [Vollschritt / s]  
  
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_ACCELERATIONSLOPE, 20,0,0);
```

// Beschleunigung in [Vollschritten / ms]

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_DECELERATIONSLOPE, 20,0,0);
```

// Bremsung in [Vollschritten / ms]

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_PHASECURRENT, 750,0,0);
```

// Phasenstrom [mA]

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_HOLDPHASECURRENT, 500,0,0);
```

// Phasenstrom bei Motorstillstand [mA]

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_HOLDTIME, 15000,0,0);
```

// Zeit in der der Haltestrom fließt nach Motorstop [s]

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_STATUSLEDMODE, 0,0,0);
```

// Betriebsart der Status-LED

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW1, 0,0,0);
```

// invertiere Funktion des Endschalter1

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW2, 0,0,0);
```

// invertiere Funktion des Endschalter2

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW1, 0,0,0);
```

// invertiere Funktion des Referenzschalters

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,
```

```

DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW2, 0,0,0);
// invertiere Funktion des Referenzschalterschalter2

DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_DIRECTION, 0,0,0);
// invertiere alle Richtungsangaben

DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_ENDSWITCH_STOPMODE, 0,0,0);
// einstellen des Stopverhaltens

DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC, DAPI_STEPPER_M
OTORCHAR_PAR_GOREFERENCEFREQUENCY_TOENDSWITCH,
100,0,0);
// Einstellung der Geschwindigkeit, mit der zum Endschalter angefahren wird.

DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_AFTEREN
DSWITCH , 200,0,0);
// Einstellung der Geschwindigkeit, mit der vom Endschalter abgefahren wird.

DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOOFFSE
T, 300,0,0);
// Einstellung der Geschwindigkeit, mit der zum optionalen Offset angefahren
wird.

```

2.14.1.11. DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC

Beschreibung

Hiermit wird der Motorspezifische Parameter ausgelesen. Dieses Kommando darf immer benutzt werden. Es teilt sich in Unterkommandos auf, die analog den Parametern von DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC sind.

Definition

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC, par1, 0, 0, 0);
```

Parameter

Parameter-Stepmode abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_STEPMODE
```

Parameter-GO-Frequency abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_GOFREQUENCY
```

Parameter-Start-Frequency abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_STARTFREQUENCY
```

Parameter-Stop-Frequency abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_STOPFREQUENCY
```

Parameter-Max-Frequency abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_MAXFREQUENCY
```

Parameter-Accelerationslope abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_ACCELERATIONSLOPE
```

Parameter-Decelerationslope abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_DECELERATIONSLOPE
```

Parameter-Phasecurrent abfragen

```
par1=DAPI_STEPPER_MOTORCHAR_PAR_PHASECURRENT
```

Parameter-Hold-Phasecurrent abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_HOLDPHASECURRENT

Parameter-Hold-Time abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_HOLDTIME

Parameter-Status-LED-Mode abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_STATUSLEDMODE

Parameter-Invert-END-Switch1 abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW1

Parameter-Invert-END-Switch2 abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW2

Parameter-Invert-Ref-Switch1 abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW1

Parameter-Invert-Ref-Switch2 abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW2

Parameter-Invert-direction abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_DIRECTION

Parameter-Endswitch-Stopmode abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_ENDSWITCH_STOPMODE

Parameter-GoReferenceFrequency abfragen

(ACHTUNG: Dieser Parameter wird nicht mehr unterstützt!)

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY

Bemerkung:

Dieser Parameter wird durch die nachfolgenden drei Parametern vollständig ersetzt.

Parameter-GoReferenceFrequencyToEndSwitch abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOENDSWITCH

Parameter GoReferenceFrequencyAfterEndSwitch abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_AFTERE

NDSWITCH

Parameter GoReferenceFrequencyToOffSet abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOOFFS
ET

Return-Wert

Parameter-Stepmode ablesen

par1=DAPI_STEPPER_MOTORCHAR_PAR_STEPMODE

return=0 (Vollschrittbetrieb)

return=1 (Halbschrittbetrieb)

return=2 (Viertelschrittbetrieb)

return=3 (Achtelschrittbetrieb)

return=4 (Sechzehntelschrittbetrieb)

Parameter-GO-Frequency

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOFREQUENCY

return=Geschwindigkeit [Vollschritt / s] - bezogen auf Vollschritt

Parameter-Start-Frequency

par1=DAPI_STEPPER_MOTORCHAR_PAR_STARTFREQUENCY

return=Startfrequenz [Vollschritt / s]

Parameter-Stop-Frequency

par1=DAPI_STEPPER_MOTORCHAR_PAR_STOPFREQUENCY

return=Stopfrequenz [Vollschritt / s]

Parameter-Max-Frequency

par1=DAPI_STEPPER_MOTORCHAR_PAR_MAXFREQUENCY

return=maximale Frequenz [Vollschritt / s]

Parameter-Accelerationslope

par1=DAPI_STEPPER_MOTORCHAR_PAR_ACCELERATIONSLOPE

return=Beschleunigungsrampe [Vollschritten / ms]

Parameter-Decelerationslope

par1=DAPI_STEPPER_MOTORCHAR_PAR_DECELERATIONSLOPE

return= Bremsrampe [Vollschritten / ms]

Parameter-Phasecurrent

par1=DAPI_STEPPER_MOTORCHAR_PAR_PHASECURRENT

return=Phasenstrom [mA]

Parameter-Hold-Phasecurrent

par1=DAPI_STEPPER_MOTORCHAR_PAR_HOLDPHASECURRENT

return= Phasenstrom bei Motorstillstand [mA]

Parameter-Hold-Time

par1=DAPI_STEPPER_MOTORCHAR_PAR_HOLDTIME

return=Zeit in der der Haltestrom fließt nach Motorstop [ms]

return=-1 / FFFF hex / 65535 dez (Zeit unendlich)

Parameter-Status-LED-Mode

par1=DAPI_STEPPER_MOTORCHAR_PAR_STATUSLEDMODE

return=Betriebsart der Status-LED

return=0 (MOVE - LED leuchtet bei Motorbewegung)

return=1 (HALT - LED leuchtet bei Motorstillstand)

return=2 (ENDSW1 - LED leuchtet bei geschlossenen Endschalter1)

return=3 (ENDSW2 - LED leuchtet bei geschlossenen Endschalter2)

return=4 (REFSW1 - LED leuchtet bei geschlossenen Referenzschalterschalter1)

return=5 (REFSW2 - LED leuchtet bei geschlossenen Referenzschalterschalter2)

Parameter-Invert-END-Switch1

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW1

return=Endschalter1 wird invertiert (0=normal / 1=invertieren)

Parameter-Invert-END-Switch2

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW2

return=Endschalter2 wird invertiert (0=normal / 1=invertieren)

Parameter-Invert-Ref-Switch1

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW1

return=Referenzschalterschalter1 wird invertiert (0=normal / 1=invertieren)

Parameter-Invert-Ref-Switch2

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW2

return=Referenzschalterschalter2 wird invertiert (0=normal / 1=invertieren)

Parameter-Invert-direction

par1=DAPI_STEPPER_MOTORCHAR_PAR_INVERT_DIRECTION

return=Richtungsangaben werden invertiert (0=normal / 1=invertieren)

Parameter-Endswitch-Stopmode

par1= DAPI_STEPPER_MOTORCHAR_PAR_ENDSWITCH_STOPMODE

return=Einstellung des Stopverhaltens (0=Fullstop / 1=Stop)

Parameter-GoReferenceFrequencyToEndSwitch

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOENDSWITCH

return=Frequenz [Vollschritt / s]

Parameter GoReferenceFrequencyAfterEndSwitch abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_AFTERE
NDSWITCH

return=Frequenz [Vollschritt / s]

Parameter GoReferenceFrequencyToOffset abfragen

par1=DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOOFFS
ET

return=Frequenz [Vollschritt / s]

Programmierbeispiel

```
value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_STEPMODE, 0, 0, 0);
// Schrittmode (Voll-, Halb-, Viertel-, Achtel-, Sechszehntelschritt)

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_GOFREQUENCY, 0, 0, 0);
// Schrittmode bei Motorstop (Voll-, Halb-, Viertel-, Achtel-,
Sechszehntelschritt)
```

```

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_STARTFREQUENCY, 0, 0, 0);
// Startfrequenz [Vollschritt / s]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_STOPFREQUENCY, 0, 0, 0);
// Stopfrequenz [Vollschritt / s]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_MAXFREQUENCY, 0, 0, 0);

// maximale Frequenz [Vollschritt / s]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_ACCELERATIONSLOPE, 0, 0, 0);
// Beschleunigung in [Vollschritten / ms]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_DECELERATIONSLOPE, 0, 0, 0);
// Bremsung in [Vollschritten / ms]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_PHASECURRENT, 0, 0, 0);
// Phasenstrom [mA]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_HOLDPHASECURRENT, 0, 0, 0);
// Phasenstrom bei Motorstillstand [mA]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_HOLDTIME, 0, 0, 0);
// Zeit in der der Haltestrom fließt nach Motorstop [s]

value = DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,
DAPI_STEPPER_MOTORCHAR_PAR_STATUSLEDMODE, 0, 0, 0);

```

// Betriebsart der Status-LED

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW1, 0, 0, 0);
```

// invertiere Funktion des Endschaltes1

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_ENDSW2, 0, 0, 0);
```

// invertiere Funktion des Endschaltes2

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW1, 0, 0, 0);
```

// invertiere Funktion des Referenzschalters1

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_REFSW2, 0, 0, 0);
```

// invertiere Funktion des Referenzschalters2

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_INVERT_DIRECTION, 0, 0, 0);
```

// invertiere alle Richtungsangaben

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_ENDSWITCH_STOPMODE, 0, 0,  
0);
```

// einstellen des Stopverhaltens

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOENDSW  
ITCH, 0, 0, 0);
```

// Abfrage der Geschwindigkeit, mit der der Endschaltes angefahren wird.

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_AFTEREN  
DSWITCH, 0, 0, 0);
```

// Abfrage der Geschwindigkeit, mit der vom Endschalter abgefahren wird.

```
value = DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC,  
DAPI_STEPPER_MOTORCHAR_PAR_GOREFERENCEFREQUENCY_TOOFFSE  
T, 0,0,0);
```

// Abfrage der Geschwindigkeit, mit der der optionale Offset angefahren wird.

2.14.1.12. DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_EEPROM_SAVE

Beschreibung

Es wird die aktuelle Motorcharakteristik des Motors ins EEPROM abgespeichert.

Definition

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_EEPROM_SAVE, 0, 0, 0, 0);
```


2.14.1.13. DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_EEPROM_LOAD

Beschreibung

Es wird die Motorcharakteristik des Motors aus dem EEPROM geladen.

Definition

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_EEPROM_LOAD, 0, 0, 0, 0);
```

2.14.1.14. DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_LOAD_DEFAULT

Beschreibung

Es wird die Motorcharakteristik des Motors auf Defaultwerte zurück gesetzt.

Definition

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_MOTORCHARACTERISTIC_LOAD_DEFAULT, 0, 0, 0, 0);
```

Bemerkung

Die Defaultwerte sind folgende:

- Stepmode Vollschrift
- Schrittfrequenz bei GoPosition [Vollschrift / s]: 1000 Hz
- Startfrequenz [Vollschrift / s]: 200Hz
- Stopfrequenz [Vollschrift / s]: 200Hz
- Maximale Schrittfrequenz [Vollschrift / s]: 3000Hz
- Beschleunigungsrampe [Hz/10ms]: 10Hz/10ms
- Bremsrampe [Hz/10ms]: 10Hz/10ms
- Phasenstrom 0..1,5A [1mA]: 750mA
- Haltestrom 0..1,5A [1mA]: 500mA
- Haltezeit 0..unendlich [ms]: 15000ms
- Status_LEDfunktion: Move
- Funktion des Endschalter1: nicht invertiert
- Funktion des Endschalter2: nicht invertiert
- Funktion des Referenzschalter1: nicht invertiert
- Funktion des Referenzschalter2: nicht invertiert
- Funktion aller Richtungsangaben: nicht invertiert
- Endschaltermode: Fullstop
- Schrittfrequenz bei GoReferenz [Vollschrift / s]: 1000 Hz

2.14.1.15. DAPI_STEPPER_CMD_GO_REFSWITCH

Beschreibung

Der Motor fährt zur Referenzposition.

Definition

```
DapiStepperCommand(handle, motor, DAPI_STEPPER_CMD_GO_REFSWITCH,  
par1, par2, par3, 0);
```

Parameter

Mögliche Werte für par1: (werden mehrere benötigt, müssen die einzelnen addiert werden)

DAPI_STEPPER_GO_REFSWITCH_PAR_REF1

DAPI_STEPPER_GO_REFSWITCH_PAR_REF2

DAPI_STEPPER_GO_REFSWITCH_PAR_REF_LEFT

DAPI_STEPPER_GO_REFSWITCH_PAR_REF_RIGHT

DAPI_STEPPER_GO_REFSWITCH_PAR_REF_GO_POSITIVE

DAPI_STEPPER_GO_REFSWITCH_PAR_REF_GO_NEGATIVE

DAPI_STEPPER_GO_REFSWITCH_PAR_SET_POS_0

par2=Motorpositionsoffset (1/16 Vollschrift)

par3=Timeoutzeit [ms]

Bemerkung

Anfahren des Referenzschalters

Zunächst fährt der Motor zur Referenzposition 1 oder 2 (siehe par1).

Hierbei kann angegeben werden, ob der Referenzschalter 1 (DAPI_STEPPER_GO_REFSWITCH_PAR_REF1) oder der Referenzschalter 2 (DAPI_STEPPER_GO_REFSWITCH_PAR_REF2) angefahren wird. Dabei lässt sich die Richtung wählen in die der Motor startet. Mit dem Parameter DAPI_STEPPER_GO_REFSWITCH_PAR_REF_GO_NEGATIVE wird nach links und mit dem Parameter DAPI_STEPPER_GO_REFSWITCH_PAR_REF_GO_POSITIVE wird nach rechts gestartet.

Hierbei wird die Geschwindigkeit GOREFERENCEFREQUENCY_TOENDSWITCH benutzt (siehe **DapiStepperCommand_SetMotorcharacteristic**).

Herausfahren aus dem Referenzschalter

Danach fährt der Motor mit der Geschwindigkeit GOREFERENCEFREQUENCY_AFTERENDSWITCH aus der Referenzposition heraus. Dabei läßt sich wählen, ob der Motor die rechte oder linke Seite des Referenzschalters anfährt. Mit dem Parameter DAPI_STEPPER_GO_REFSWITCH_PAR_REF_LEFT wird die linke Kante angefahren und mit dem Parameter DAPI_STEPPER_GO_REFSWITCH_PAR_REF_RIGHT wird die rechte Kante angefahren.

Optionales Anfahren eines Offsets

Nach dem Herausfahren aus dem Referenzschalter kann noch ein Offset angefahren werden. Falls dieser Parameter nicht = 0 ist (par2), fährt der Motor zu diesem Offset mit der Geschwindigkeit GOREFERENCEFREQUENCY_TOOFFSET.

Nullen der Position des Motors

Mit dem Parameter DAPI_STEPPER_GO_REFSWITCH_PAR_SET_POS_0 kann zusätzlich eingestellt werden, ob der Motor jetzt die Position 0 bekommt.

Programmierbeispiel

```
DapiStepperCommand(handle, motor,
DAPI_STEPPER_CMD_GO_REFSWITCH,
DAPI_STEPPER_GO_REFSWITCH_PAR_REF1 +
DAPI_STEPPER_GO_REFSWITCH_PAR_REF_LEFT +
DAPI_STEPPER_GO_REFSWITCH_PAR_REF_GO_POSITIVE +
DAPI_STEPPER_GO_REFSWITCH_PAR_SET_POS_0, 0, 15000, 0);
```

2.14.1.16. DAPI_STEPPER_CMD_GET_CPU_TEMP

Beschreibung

Die Temperatur des CPU wird abgefragt.

Definition

```
ULONG DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_CPU_TEMP, 0, 0, 0, 0);
```

Parameter

cmd=DAPI_STEPPER_CMD_GET_CPU_TEMP

Return-Wert

Temperatur [°C]

2.14.1.17. DAPI_STEPPER_CMD_GET_MOTOR_SUPPLY_VOLTAGE

Beschreibung

Hiermit wird die Versorgungsspannung des Motors abgefragt.

Definition

```
DapiStepperCommand(handle, motor,  
DAPI_STEPPER_CMD_GET_MOTOR_SUPPLY_VOLTAGE, 0, 0, 0, 0);
```

Parameter

cmd=DAPI_STEPPER_CMD_GET_MOTOR_SUPPLY_VOLTAGE

Return-Wert

Motorversorgungsspannung in [mV]

2.14.2. Status abfragen mit DapiStepperGetStatus

2.14.2.1. DAPI_STEPPER_STATUS_GET_ACTIVITY

Beschreibung

Hiermit werden verschiedene Statusinformationen (z.B. die Aktivität des Motorstroms, etc.) abgefragt.

Definition

```
ULONG DapiStepperGetStatus(handle, motor,  
DAPI_STEPPER_STATUS_GET_ACTIVITY);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

motor=Nummer des anzusprechenden Motors

Return-Wert

Bit	Command	Beschreibung
0	DISABLE	Motor darf nicht verfahren
1	MOTORSTROMACTIV	Motorstrom ist aktiv
2	HALTESTROMACTIV	Haltestrom ist aktiv
3	GOPOSITIONACTIV	GoPosition ist aktiv
4	GOPOSITIONBREMSSEN	GoPosition Bremsung ist aktiv
5	GOREFERENZACTIV	GoReference ist aktiv

Programmierbeispiel

```
ret = DapiStepperGetStatus(handle, motor,  
DAPI_STEPPER_STATUS_GET_ACTIVITY);
```

2.14.2.2. DAPI_STEPPER_STATUS_GET_POSITION

Beschreibung

Hiermit wird eine bestimmte Position abgelesen.

Definition

ULONG DapiStepperGetStatus(handle, motor, cmd);

Parameter

cmd=DAPI_STEPPER_STATUS_GET_POSITION

Return-Wert

Es wird die aktuelle Motorposition in 1/16 Schritteinheiten zurückgegeben

Programmierbeispiel

```
value = DapiStepperGetStatus(handle, motor,  
DAPI_STEPPER_STATUS_GET_POSITION);
```


2.14.2.3. DAPI_STEPPER_STATUS_GET_SWITCH

Beschreibung

Hiermit wird der Zustand der Schalter abgefragt.

Definition

ULONG DapiStepperGetStatus(handle, motor, cmd);

Parameter

cmd=DAPI_STEPPER_STATUS_GET_SWITCH

Return-Wert

Es wird der Zustand der Schalter zurückgeliefert:

Bit0: ENDSCHALTER1; 1 = Endschalter1 ist geschlossen

Bit1: ENDSCHALTER2; 1 = Endschalter2 ist geschlossen

Bit2: REFSCHALTER1; 1 = Referenzschalter1 ist geschlossen

Bit3: REFSCHALTER2; 1 = Referenzschalter2 ist geschlossen

Programmierbeispiel

```
pos = DapiStepperGetStatus(handle, motor,  
DAPI_STEPPER_STATUS_GET_SWITCH);
```

2.14.3. DapiStepperCommandEx

Beschreibung

Dieser erweiterte Befehl steuert Schrittmotoren an.

Definition

```
ULONG DapiStepperCommandEx(ULONG handle, ULONG motor, ULONG cmd,  
ULONG  
par1, ULONG par2, ULONG par3, ULONG par4, ULONG par5, ULONG par6, ULONG  
par7);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

motor=Nummer des anzusprechenden Motors

cmd=Erweitertes Kommando

par1..7=Erweiterte kommandoabhängige Parameter (s. Bemerkung)

Bemerkung

Siehe delib.h für die erweiterten Kommandos und den zugehörigen Parametern.

2.15. CAN Runtime Funktionen

2.15.1. RunTimeVarWriteToModule

Beschreibung

Bei dem Start des Moduls werden die Einstellungen aus dem **Module-Configuration-Memory** geladen und benutzt. Mit Hilfe dieser Befehle lassen sich die Einstellungen während der Laufzeit verändern und auslesen.

Sie werden jedoch nicht in das **Module-Configuration-Memory** gespeichert und gehen daher nach Modulneustart verloren.

Parameter

handle = Dies ist das Handle eines geöffneten Moduls

par = Laufzeitenvariable, die beschrieben oder ausgelesen werden soll

index = Gibt den Index des TX/RX-Paketes an [Wertebereich 0-7]

value = Der Wert um den die Laufzeitenvariable geändert werden soll. Bei der Read-Funktion wird hier eine Referenz übergeben

Bemerkung

Das value muss immer als Hex-Wert angegeben werden. Der Return-Wert ist ebenfalls in Hex. Eine Auflistung von Module, die diese Funktionen unterstützen, können Sie unserer **Delib Übersichtstabelle** entnehmen.

Definition

Für ein besseres Verständnis unserer Beispiele, verwenden wir für das Schreiben die Funktion **RunTimeVarWriteToModule** und für das Lesen **RunTimeVarReadFromModule**

Der darin befindliche Quellcode lautet wie folgt:

//Lesen der Werte

```
public static uint RunTimeVarReadFromModule(uint handle, uint par,
uint index, ref uint value)
{
    byte[] dummy_buff = new byte[] { 0 };
    uint u0 = 0;

    if(DT.Delib.DapiSpecialCommandExt(handle,
        DT.Ext.DAPI_SPECIAL_CMDEXT_CAN_RD_RUNTIME_VALUE,
        par, index, value, ref u0, ref u0,
        dummy_buff, 0, dummy_buff, 0, dummy_buff, 0, ref u0) !=
        DT.RETURN_OK)
    {
        return DT.Error.DAPI_ERR_DEV_CONFIG_READ_ERROR;
    }
    return DT.Error.DAPI_ERR_NONE;
}
```

//Schreiben der Werte

```
public static uint RunTimeVarWriteToModule(uint handle, uint par,
uint index, uint value)
{
    byte[] dummy_buff = new byte[] { 0 };
    uint u0 = 0;

    if(DT.Delib.DapiSpecialCommandExt(handle,
        DT.Ext.DAPI_SPECIAL_CMDEXT_CAN_WR_RUNTIME_VALUE,
        par, index, value, ref u0, ref u0, ref u0,
        dummy_buff, 0, dummy_buff, 0, dummy_buff, 0, ref u0) !=
        DT.RETURN_OK)
    {
        return DT.Error.DAPI_ERR_DEV_CONFIG_READ_ERROR;
    }
    return DT.Error.DAPI_ERR_NONE;
}
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_DEV_BAUDRATE

Mit diesem Befehl kann die Baudrate des Interface eingestellt/ausgelesen werden.

Baudrate	Value
1 MBit/s	0x00
500 KBit/s	0x01
250 KBit/s	0x02
125 KBit/s	0x03
100 KBit/s	0x04
50 KBit/s	0x05
20 KBit/s	0x06
10 KBit/s	0x07

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_DEV_BAUDRATE, 0, 0x01);  
// Hier wird die Baudrate auf 500 KBit/s gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_DEV_BAUDRATE, 0, ref  
val);  
// Hier wird die Baudrate an die Variable val übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_DEV_USEEXTID

Mit diesem Befehl kann der Bit-Mode eingestellt/ausgelesen werden.

useExtID	Value
11 Bit Mode	0x00
29 Bit Mode	0x01

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_DEV_USEEXTID, 0, 0x00);  
// Hier wird die Ext-ID des Interfaces auf den 11 Bit Mode gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_DEV_USEEXTID, 0, ref  
val);  
// Hier wird der verwendete Bit Mode der Variable val übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_IS_ACTIVE

Mit diesem Befehl kann der Trigger-Mode eingestellt/ausgelesen werden.

Bei dem Nutzen des "Interval Mode (0x01)" kann zusätzlich über den Interval-Befehl eingestellt werden, in welchem Zeitintervall die TX-Pakete gesendet werden sollen.

Trigger Mode	Value
OFF	0x00
Interval Mode	0x01
RX-Event	0x02
Fast as possible	0x03

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_IS_ACTIVE, 1, 0x00);  
// Hier wird der Trigger Mode des TX-Paketes[1] auf OFF gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_IS_ACTIVE, 0, ref  
val);  
// Hier wird der Trigger-Mode Status des TX-Paketes[0] der Variable val  
übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_INTERVAL

Mit diesem Befehl kann das Interval eingestellt/ausgelesen werden.

Interval count	Value Bit [4..7]
1	0x01
2	0x02
3	0x03
4	0x04
.. 9	.. 0x09

Interval unit	Value Bit [0..3]
* 1 ms	0x01
* 10 ms	0x02
* 100 ms	0x03
* 1 sec	0x04

Beispiel

Ein Interval von 700ms entspricht einem value von 0x73

Ein Interval von 40ms entspricht einem value von 0x42

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_INTERVAL, 1,  
(((0x02 << 4) & 0xf0) | (0x04 & 0x0f)));  
// Hier wird das Interval des TX-Paketes[1] auf 40ms eingestellt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_INTERVAL, 0, ref  
val);  
// Hier wird das Interval des TX-Paketes[0] der Variable val übergeben.
```


par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_USE_EXT_ID

Mit diesem Befehl kann der Bit-Mode eingestellt/ausgelesen werden.

useExtID	Value
11 Bit Mode	0x00
29 Bit Mode	0x01

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_USE_EXT_ID, 1,  
0x00);  
// Hier wird die Ext-ID des TX-Paketes[1] auf den 11 Bit Mode gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_USE_EXT_ID, 0, ref  
val);  
// Hier wird der verwendete Bit Mode des TX-Paketes[0] der Variable val  
übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_CANID

Mit diesem Befehl kann die CAN-ID eingestellt/ausgelesen werden.

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_CANID, 1, 0x1e);  
// Hier wird die CAN-ID des TX-Paketes[1] auf die 30 gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_CANID, 0, ref val);  
// Hier wird der verwendete CAN-ID des TX-Paketes[0] der Variable val  
übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_MODE

Mit diesem Befehl kann der TX-Mode eingestellt/ausgelesen werden.

TX-Mode	Value
OPTO-IN 1-64	0x01
OPTO-IN 65-128	0x24
OPTO-IN 129-192	0x25
OPTO-IN 193-256	0x26
A/D CH 1-4 (16 Bit)	0x02
A/D CH 5-8 (16 Bit)",	0x03
A/D CH 9-12 (16 Bit)	0x04
A/D CH 13-16 (16 Bit)	0x05
A/D CH 17-20 (16 Bit)	0x06
A/D CH 21-24 (16 Bit)	0x07
A/D CH 25-28 (16 Bit)	0x08
A/D CH 29-32 (16 Bit)	0x09
Counter16 1-4 (16 Bit)	0x0a
Counter16 5-8 (16 Bit)	0x0b
Counter16 9-12 (16 Bit)	0x0c
Counter16 13-16 (16 Bit)	0x0d
Counter16 17-20 (16 Bit)	0x0e
Counter16 21-24 (16 Bit)	0x0f
Counter16 25-28 (16 Bit)	0x10
Counter16 29-32 (16 Bit)	0x11

TX-Mode	Value
Cnt48 1-2 (32 Bit)	0x12
Cnt48 3-4 (32 Bit)	0x13
Cnt48 5-6 (32 Bit)	0x14
Cnt48 7-8 (32 Bit)	0x15
PT-100 1-2 (32 Bit)	0x16
PT-100 3-4 (32 Bit)	0x17
PT-100 5-6 (32 Bit)	0x18
PT-100 7-8 (32 Bit)	0x19
Cnt48 1 (64 Bit)	0x1a
Cnt48 2 (64 Bit)	0x1b
Cnt48 3 (64 Bit)	0x1c
Cnt48 4 (64 Bit)	0x1d
Testcounter 8 bit	0x1e
DO Readback 1-64	0x1f
DO Readback 1-32	0x23
Custom1	0x20
Custom2	0x21
Custom3	0x22

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_MODE, 1, 0x0f);  
// Hier wird der Modus des TX-Paketes[1] auf den TX-Mode "Counter16 21-24  
(16 Bit)" gesetzt
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_TX_MODE, 0, ref val);  
// Hier wird der verwendete TX-Mode des TX-Paketes[0] der Variable val  
übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_IS_ACTIVE

Mit diesem Befehl wird das RX-Paket aktiviert/deaktiviert

Trigger Mode	Value
OFF	0x00
ON	0x01

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_IS_ACTIVE, 1, 0x00);  
// Hier wird das RX-Paket[1] auf OFF gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_IS_ACTIVE, 0, ref  
val);  
// Hier wird der Status des RX-Paketes[0] der Variable val übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_USE_EXT_ID

Mit diesem Befehl kann der Bit-Mode eingestellt/ausgelesen werden.

UseExtID	Value
11 Bit Mode	0x00
29 Bit Mode	0x01

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_USE_EXT_ID, 1,  
0x00);  
// Hier wird die Ext-ID des RX-Paketes[1] auf den 11 Bit Mode gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_USE_EXT_ID, 0, ref  
val);  
// Hier wird der verwendete Bit Mode des RX-Paketes[0] der Variable val  
übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_CANID

Mit diesem Befehl kann die CAN-ID eingestellt/ausgelesen werden.

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_CANID, 1, 0x1e);  
// Hier wird die CAN-ID des RX-Paketes[1] auf die 30 gesetzt.
```

```
uint val = 0;  
RunTimeVarReadFromModule(handle,  
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_CANID, 0, ref val);  
// Hier wird der verwendete CAN-ID des RX-Paketes[0] der Variable val  
übergeben.
```

par = DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_MODE

Mit diesem Befehl kann RX-Mode eingestellt/ausgelesen werden.

RX-Mode	Value
Digital Out 1-64	0x01
D/A CH 1-4	0x02
D/A CH 5-8	0x03
D/A CH 9-12	0x04
D/A CH 13-16	0x05
D/A CH 17-20	0x06
D/A CH 21-24	0x07
D/A CH 25-28	0x08
D/A CH 29-32	0x09
Stepper No. 1	0x0a
Stepper No. 2	0x0b
Stepper No. 3	0x0c
Stepper No. 4	0x0d
Stepper No. 5	0x0e
Stepper No. 6	0x0f
Stepper No. 7	0x10
Stepper No. 8	0x11
D/A CH 1-4 (custom)	0x12
D/A CH 5-8 (custom)	0x13
D/A CH 9-12 (custom)	0x14

RX-Mode	Value
D/A CH 13-16 (custom)	0x15
D/A CH 17-20 (custom)	0x16
D/A CH 21-24 (custom)	0x17
D/A CH 25-28 (custom)	0x18
D/A CH 29-32 (custom)	0x19
Trigger Auto TX 1	0x1a
Trigger Auto TX 2	0x1b
Trigger Auto TX 3	0x1c
Trigger Auto TX 4	0x1d
Custom1	0x1e
Custom2	0x1f
Custom3	0x20
PWM CH 1-8	0x21
PWM CH 9-16	0x22
PWM CH 17-24	0x23
PWM CH 25-32	0x24
PWM CH 33-40	0x25
PWM CH 41-48	0x26
PWM CH 49-56	0x27
PWM CH 57-64	0x28
Trigger Auto TX 5	0x29

RX-Mode	Value
Trigger Auto TX 6	0x2a
Trigger Auto TX 7	0x2b
Trigger Auto TX 8	0x2c

Programmierbeispiel

```
RunTimeVarWriteToModule(handle,
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_MODE, 1, 0x0f);
// Hier wird der Modus des RX-Paketes[1] auf den RX-Mode "Stepper No. 6"
gesetzt
```

```
uint val = 0;
RunTimeVarReadFromModule(handle,
DAPI_SPECIAL_CMDEXT_CAN_RUNTIME_RX_MODE, 0, ref val);
// Hier wird der verwendete RX-Mode des RX-Paketes[0] der Variable val
übergeben.
```

2.16. Software FIFO verwalten

2.16.1. DapiSpecialCMDSWFifo

Beschreibung

Dieser Befehl dient zum Einstellen des Software FIFO.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance,  
par2);
```

Parameter

handle = Dies ist das Handle eines geöffneten Moduls

cmd = auszuführende Funktion

fifo_instance = Gibt die Instanz des Software FIFO an.

par2 = Wert, der an die Funktion übergeben wird

Bemerkung

Definieren Sie als erstes immer das Submodul mit dem DapiSpecialSWFifoSetSubmodule-Befehl!

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

2.16.1.1. DapiSpecialSWFifoSetSubmodule

Beschreibung

Dieser Befehl gibt an, an welches NET-Submodul die Daten des Software FIFO übergeben werden.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,  
fif_instance, par2);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_SET_SUBMODULE

fif_instance = Gibt die Instanz des Software FIFO an

par2 = gibt die Nummer des Submoduls an (0, 1, 2, 3, ...)

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_SET_SUBMODULE, fif_instance, 2);  
//Der Software FIFO überträgt die Daten an NET-Submodul 2.
```

2.16.1.2. DapiSpecialSWFifoGetSubmodule

Beschreibung

Dieser Befehl gibt die Nummer des NET-Submoduls, auf welches die Daten übertragen werden, wieder.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,  
fif_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_SUBMODULE

fif_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Nummer des Submodules (0, 1, 2, 3, ...)

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,
```

```
DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_GET_SUBMODULE, fifo_instance, 0);  
printf("Submodule = %lu\n", ret);  
//Die Nummer des NET-Submoduls wird ausgelesen und dargestellt.
```

2.16.1.3. DapiSpecialSWFifoActivate

Beschreibung

Dieser Befehl aktiviert die Fifo-Datenübertragung innerhalb der NET-Module.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_ACTIVATE

fifo_instance = Gibt die Instanz des Software FIFO an

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_ACTIVATE, fifo_instance, 0);  
//Das automatische Ausgeben der Fifo-Datenübertragung wird aktiviert.
```

2.16.1.4. DapiSpecialSWFifoDeactivate

Beschreibung

Dieser Befehl deaktiviert die Fifo-Datenübertragung innerhalb der NET-Module.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_DEACTIVATE

fifo_instance = Gibt die Instanz des Software FIFO an

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_DEACTIVATE, fifo_instance, 0);  
//Das automatische Ausgeben der Fifo-Datenübertragung wird deaktiviert.
```

2.16.1.5. DapiSpecialSWFifoGetActivity

Beschreibung

Mit diesem Befehl wird der Übertragungs-Status des FIFO abgerufen (ob aktiv oder inaktiv).

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_SW_FIFO, cmd,  
fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_ACTIVITY

fifo_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Return = 0 (Übertragung ist deaktiviert)

Return = 1 (Übertragung ist aktiviert)

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_GET_ACTIVITY, fifo_instance, 0);  
printf("Status = %lu\n", ret);  
//Übertragungs-Status wird abgerufen
```

2.16.1.6. DapiSpecialSWFifoIOActivate

Beschreibung

Dieser Befehl aktiviert die FIFO- I/O Ein-/Ausgabe.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_IO_ACTIVATE

fifo_instance = Gibt die Instanz des Software FIFO an

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_IO_ACTIVATE, fifo_instance, 0);  
//Das automatische Ausgeben des FIFO an das Modul wird aktiviert.
```

2.16.1.7. DapiSpecialSWFifoIODeactivate

Beschreibung

Dieser Befehl deaktiviert die FIFO- I/O Ein-/Ausgabe.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_IO_DEACTIVATE

fifo_instance = Gibt die Instanz des Software FIFO an

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_IO_DEACTIVATE, fifo_instance, 0);  
//Das automatische Ausgeben des FIFO an das Modul wird deaktiviert.
```


2.16.1.8. DapiSpecialSWFifoInitAndClear

Beschreibung

Dieser Befehl löscht vorhandene Daten aus dem Software FIFO Speicher und bringt den FIFO-Mechanismus in den Ausgangszustand zurück.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_INIT_AND_CLEAR

fifo_instance = Gibt die Instanz des Software FIFO an

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_INIT_AND_CLEAR, fifo_instance, 0);  
//Vorhandene Daten werden aus dem FIFO Speicher gelöscht.
```

2.16.1.9. DapiSpecialSWFifoSetChannel

Beschreibung

Dieser Befehl gibt unter Angabe von Start- und Endkanal an, in welche Kanäle die Daten des FIFO übertragen werden sollen.

Definition

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, ch);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_SET_CHANNEL

fifo_instance = Gibt die Instanz des Software FIFO an

ch = Angabe des Start- und Endkanals

Programmierbeispiel

```
unsigned long ch_start = 0; //Start with Channel 0
unsigned long ch_end = 1; //End with Channel 1

DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,
DAPI_SPECIAL_SW_FIFO_SET_Channel, fifo_instance,
((ch_end << 8) & 0xff00) | (ch_start & 0xff);
//Der Start- und Endkanal wird festgelegt
```

2.16.1.10. DapiSpecialSWFifoGetChannel

Beschreibung

Dieser Befehl zeigt die Kanäle, in welche die Daten übertragen werden.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,
fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_CHANNEL

fifo_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Nummer der Kanäle

Bit 0-7 Startkanal

Bit 8-15 Endkanal

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_SW_FIFO,
DAPI_SPECIAL_SW_FIFO_GET_CHANNEL, fifo_instance, 0);
printf("Channel = %lu\n", ret);
//Zeigt auf welche Kanäle die Daten übertragen werden.
```

2.16.1.11. DapiSpecialSWFifoSetFrequencyHz

Beschreibung

Dieser Befehl gibt an, in welchem Frequenzintervall (in Hertz) ..

.. bei Eingabe gelesen wird.

.. bei Ausgabe geschrieben wird.

Definition

DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, par2);

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_SET_FREQUENCY_HZ

fifo_instance = Gibt die Instanz des Software FIFO an

par2 = Frequenzintervall in Hertz (Hz)

Bemerkung

Zulässiger Wertebereich: min 1Hz, max. abhängig vom verwendeten Modul

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_SET_FREQUENCY_HZ, fifo_instance,  
10);  
//Setzt das Frequenzintervall auf 10Hz.
```

2.16.1.12. DapiSpecialSWFifoGetFrequencyHz

Beschreibung

Dieser Befehl gibt das vorher eingestellte Frequenzintervall in Hertz wieder.

Definition

ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, 0);

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_FREQUENCY_HZ

fifo_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Frequenzintervall in Hertz (Hz)

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_SW_FIFO,
DAPI_SPECIAL_SW_FIFO_GET_FREQUENCY_HZ, fifo_instance,
0);
printf("Frequency = %lu (Hz)\n", ret);
//Zeigt das vorher eingestellte Frequenzintervall an.
```

2.16.1.13. DapiSpecialSWFifoGetBytesFree

Beschreibung

Dieser Befehl dient zum Auslesen der freien Bytes im Software FIFO Buffer.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,  
    fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_BYTES_FREE

fifo_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Freie Bytes des Software FIFO

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SW_FIFO,  
    DAPI_SPECIAL_SW_FIFO_GET_BYTES_FREE, fifo_instance, 0);  
printf("Freier Speicher = %lu\n", ret);  
//Ausgabe der noch freien Bytes des Speichers.
```

2.16.1.14. DapiSpecialSWFifoGetBytesPerSample

Beschreibung

Dieser Befehl gibt an wieviel Bytes für das Schreiben in den D/A Wandler notwendig sind.

Beispiel: Werden bei einem 16 Bit (2Byte) D/A Wandler 3 D/A Kanäle beschrieben, werden also 3x2 Bytes pro Sample benötigt. Der Wert 6 wird wiedergegeben.

Selbiges gilt auch für A/D Wandler.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,  
    fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_BYTES_PER_SAMPLE

fifo_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Benötigte Bytes pro Sample

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,  
    DAPI_SPECIAL_CMD_SW_FIFO,  
    DAPI_SPECIAL_SW_FIFO_GET_BYTES_PER_SAMPLE,  
    fifo_instance, 0);  
printf("Benötigte Bytes = %lu\n", ret);  
//Ausgabe der notwendigen Bytes pro Sample.
```

2.16.1.15. DapiSpecialSWFifoSetMode

Beschreibung

Dieser Befehl setzt den Software FIFO Mode.

Definition

DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd, fifo_instance, par2);

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_SET_MODE

fifo_instance = Gibt die Instanz des Software FIFO an

par2 = Software FIFO Mode	Wert(hex)
DAPI_SPECIAL_SW_FIFO_MODE_IN_AD16	0x40
DAPI_SPECIAL_SW_FIFO_MODE_IN_AD16_TS	0xc0
DAPI_SPECIAL_SW_FIFO_MODE_IN_AD18	0x41
DAPI_SPECIAL_SW_FIFO_MODE_IN_AD18_TS	0xc1
DAPI_SPECIAL_SW_FIFO_MODE_IN_DI8	0x60
DAPI_SPECIAL_SW_FIFO_MODE_IN_DI8_TS	0xe0
DAPI_SPECIAL_SW_FIFO_MODE_IN_DI16	0x61
DAPI_SPECIAL_SW_FIFO_MODE_IN_DI16_TS	0xe1

Bemerkung

Der Software FIFO-Mode kann sowohl mit als auch ohne Timestamp (TS) eingestellt werden.

Programmierbeispiel

```
DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_SET_MODE, fifo_instance, 0);  
//Der Software FIFO Mode wird gesetzt.
```


2.16.1.16. DapiSpecialSWFifoGetMode

Beschreibung

Dieser Befehl gibt den vorher eingestellten FIFO Mode wieder. Aktuell wird dieser in der Firmware noch nicht unterstützt.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,  
fif_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_MODE

fif_instance = Gibt die Instanz des Software FIFO an

Return-Wert

FIFO Software Mode

Programmierbeispiel

```
unsigned long ret = DapiSpecialCommand(handle,  
DAPI_SPECIAL_CMD_SW_FIFO,  
DAPI_SPECIAL_SW_FIFO_GET_MODE, fif_instance, 0);  
printf("Mode = %lu\n", ret);  
//Gibt den vorher eingestellten FIFO Mode wieder.
```

2.16.1.17. DapiSpecialSWFifoGetStatus

Beschreibung

Mit diesem Befehl können Statuswerte abgerufen werden.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,
fifo_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_STATUS

fifo_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Befehl	Beschreibung (FIFO-Status erzeugt einen Return-Wert...)	Wert(hex)
DAPI_SPECIAL_SW_FIFO_STATUS_IS_ACTIVE	... wenn die Ausgabe des D/A Wandlers aktiv ist	0x01
DAPI_SPECIAL_SW_FIFO_STATUS_IO_IS_ACTIVE	... wenn die Ausgabe des FIFO I/O aktiv ist	0x02
DAPI_SPECIAL_SW_FIFO_STATUS_FIFO_OVERFLOW	... wenn zuviele Daten in den FIFO geschrieben werden	0x04
DAPI_SPECIAL_SW_FIFO_STATUS_FIFO_UNDERRUN	... wenn der FIFO leer läuft	0x08
DAPI_SPECIAL_SW_FIFO_STATUS_FIFO_OUT_OF_SYNC	... wenn die FIFO-Kommunikation innerhalb der Module unterbrochen ist	0x10

Programmierbeispiel

```
unsigned long ret;

ret = DapiSpecialCommand(handle,
DAPI_SPECIAL_CMD_SW_FIFO, DAPI_SPECIAL_SW_FIFO_GET_STATU
S, fifo_instance, 0);
if((ret & 0x01) != 0) {printf("is_active");}
if((ret & 0x02) != 0) {printf("io_is_active");}
if((ret & 0x04) != 0) {printf("fifo_overflow");}
if((ret & 0x08) != 0) {printf("fifo_underrun");}
if((ret & 0x10) != 0) {printf("fifo_out_of_sync");}
```

2.16.1.18. DapiSpecialSWFifoGetInstanceType

Beschreibung

Mit diesem Befehl kann ausgelesen werden, bei welchem Kanal es sich um einen Ein- bzw. Ausgangskanal handelt.

Definition

```
ULONG DapiSpecialCommand(handle, DAPI_SPECIAL_CMD_SW_FIFO, cmd,  
fif_instance, 0);
```

Parameter

cmd = DAPI_SPECIAL_SW_FIFO_GET_INSTANCE_TYPE

fif_instance = Gibt die Instanz des Software FIFO an

Return-Wert

Befehl	Beschreibung	Wert(hex)
DAPI_SPECIAL_INSTANCE_TYPE_FIFO_IN	... gibt an, ob es sich bei dem Kanal um einen Eingang handelt	0x01
DAPI_SPECIAL_INSTANCE_TYPE_FIFO_OUT	... gibt an, ob es sich bei dem Kanal um einen Ausgang handelt	0x02

Programmierbeispiel

```
unsigned long ret;

for(int i=0;i!=10;++i)
{
    ret = DapiSpecialCommand(handle,
    DAPI_SPECIAL_CMD_SW_FIFO,
    DAPI_SPECIAL_SW_FIFO_GET_INSTANCE_TYPE, i, 0);
    switch(ret)
    {
        case DAPI_SPECIAL_INSTANCE_TYPE_FIFO_IN:
printf("Instance %d = FIFO_IN\n\r", i);break;
        case DAPI_SPECIAL_INSTANCE_TYPE_FIFO_OUT:
printf("Instance %d = FIFO_OUT\n\r", i);break;
        default:
printf("Instance %d = INVALID\n\r", i);break;
    }
}
//Gibt wieder ob es sich um einen Ein- oder Ausgangskanal handelt
```

2.16.2. DapiWriteFifo

Beschreibung

Dieser Befehl schreibt Datensätze in den Software FIFO.

Definition

*DapiWriteFifo(ULONG handle, ULONG fifo_instance, ULONG type, UCHAR * buffer, ULONG buffer_length);*

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

fifo_instance=Gibt die Instanz des Software FIFO an

type=Gibt den FIFO-Typ an

buffer=Buffer für den zu sendenden Datensatz

buffer_length=Länge des Buffers

Programmierbeispiel

```
DapiWriteFifo(handle, fifo_instance, type, buffer,
buffer_length);
//Schreibt den Datensatz in den Software FIFO.
```

2.16.3. DapiReadFifo

Beschreibung

Dieser Befehl liest den Software-FIFO aus.

Definition

```
ULONG DapiReadFifo(ULONG handle, ULONG fifo_instance, ULONG type, UCHAR  
* buffer, ULONG buffer_length);
```

Parameter

handle=Dies ist das Handle eines geöffneten Moduls

fifo_instance=Gibt die Instanz des Software FIFO an

type=Gibt den FIFO-Typ an

buffer=Buffer für den zu empfangenden Datensatz

buffer_length=Länge des Buffers

Return-Wert

Länge der ausgelesenen FIFO-Datenstätze

Aufbau eines FIFO-Datensatz (Beispiel mit 2 aktiven AD Kanälen, AD0 und AD4)

Byte	Bedeutung	Wert [hex]
0	RO_FIFO_ID_START	0xf0
1	FIFO-Typ	
2	Zeitstempel (Bit0..Bit7)	
3	Zeitstempel (Bit8..Bit15)	
4	Aktive A/D-Kanäle (Bit0..Bit7)	0x11
5	Aktive A/D-Kanäle (Bit8..Bit15)	0x00
6	A/D Wert Kanal 0 (Bit0..Bit7)	
7	A/D Wert Kanal 0 (Bit8..Bit15)	
8	A/D Wert Kanal 4 (Bit0..Bit7)	
9	A/D Wert Kanal 4 (Bit8..Bit15)	
10	RO_FIFO_ID_END	0xf1

FIFO-Datenstanz = 7 Bytes ID + (2 x Anzahl aktiver A/D-Kanäle) Bytes Daten

RO_FIFO_ID_START

Signalisiert den Anfang eines neuen FIFO-Datensatzes. Die RO_FIFO_ID_START hat immer den Wert 0xf0 [hex]

FIFO Typ

Gibt den FIFO Typ an (z.B. RO_FIFO_ID_TYPE_AD16M0 für A/D-FIFO)

Zeitstempel

Gibt den 16 Bit Zeitstempel des aktuellen Datensatzes an. Zeit-Referenz ist hierbei der Zeitpunkt der Aktivierung des FIFO.

Beim Überlauf des Zeitstempels, wird dieser auf 0 zurückgesetzt.

Aktive A/D-Kanäle

Gibt einen 16 Bit Wert für die aktuell aktiven A/D-Kanäle an. Jedes Bit steht für einen Kanal (Bit0 -> AD0, Bit1 -> AD1, .. Bit15 -> AD15).

Ist das Bit gesetzt, ist der entsprechende A/D-Kanal aktiv

RO_FIFO_ID_END

Signalisiert das Ende eines FIFO-Datensatzes. Die RO_FIFO_ID_END hat immer den Wert 0xf1 [hex]

Bemerkung

Beachten Sie, dass der Software FIFO zuvor mit dem Befehl "DapiSpecialCMDAD" aktiviert, bzw. initialisiert werden muss.

Programmierbeispiel

```
bytes_received = DapiReadFifo(handle, fifo_instance,  
DAPI_FIFO_TYPE_READ_AD_FIFO, buffer, sizeof(buffer));  
//Liest den Software FIFO aus
```

2.17. Ausgabe-Timeout verwalten

2.17.1. 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

2.17.1.1. 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.
```

2.17.1.2. 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.
```

2.17.1.3. 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 weider 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.
```


2.17.1.4. 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.
```

2.17.1.5. 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.
```

2.17.1.6. 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.
```

2.17.1.7. 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.
```

2.17.1.8. 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.
```

2.17.1.9. 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.
```

2.17.1.10. 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.
```

2.17.1.11. 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.
```


2.18. Testfunktionen

2.18.1. 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

2.19. Register Schreib-Befehle

2.19.1. 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.

2.19.2. 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.

2.19.3. 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.

2.19.4. 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.

2.20. Register Lese-Befehle

2.20.1. 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.

2.20.2. 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.

2.20.3. 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, 0xfff4);

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.
```


2.20.4. 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.

2.21. 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 ;
}

```

2.22. 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_SEC 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

Verzeichnisstruktur der DELIB



3. Verzeichnisstruktur der DELIB

Nach erfolgreicher Installation liegt folgender Verzeichnisbaum vor:

\$DELIB_DIR

- |
- | - > include Includes für Programmiersprachen
- | - > lib Library
- | - > lib\bc Borland Compiler Library
- | - > prod_pics Produktbilder
- | - > programs Modul-Testprogramme
- + - > USB-Driver Treiber für USB-Module

Bitte beachten Sie, dass der "\$DELIB_DIR" Ordner, je nach Betriebssystem und DELIB-Version, variieren kann.

DELIB Installation	Windows Installation	Pfad
32 Bit	32 Bit	C:\Programme\DEDITEC\DELIB\
32 Bit	64 Bit	C:\Programme (x86)\DEDITEC\DELIB\
64 Bit	64 Bit	C:\Programme\DEDITEC\DELIB64\

Zudem werden im Windows System Ordner folgende Dateien installiert:

\$SYSDIR\delib.dll, bzw. \$SYSDIR\delib64.dll (32 Bit, bzw. 64 Bit DELIB Version)

\$SYSDIR\delibJNI.dll, bzw. \$SYSDIR\delibJNI64.dll (32 Bit, bzw. 64 Bit DELIB Version)

\$SYSDIR\ftbusui.dll

\$SYSDIR\ftd2xx.dll

\$SYSDIR\FTLang.dll

\$SYSDIR\drivers\ftdibus.sys

Bitte beachten Sie, dass der "\$SYSDIR" Ordner, je nach Betriebssystem und DELIB-Version, variieren kann.

DELIB Installation	Windows Installation	Pfad
32 Bit	32 Bit	C:\Windows\System32
32 Bit	64 Bit	C:\Windows\SysWOW64
64 Bit	64 Bit	C:\Windows\System32

3.1. Include Verzeichnis

Das für die DELIB angelegte Include-Verzeichnis enthält die Dateien, welche die entsprechenden Library-Funktionen beschreiben. Diese sind für die Programmiersprachen C (.h), Delphi (.pas) und Visual Basic (.bas) gegeben.

3.2. Library-Verzeichnis

Hierin befindet sich die Datei "DELIB.lib". Sie dient als Bindeglied für das Compilieren von eigenen Programmen, die die "DELIB.dll" benutzen.

3.3. Library-Verzeichnis für Borland

Für Borland Compiler gibt es eine separate DELIB.lib, die sich im Unterverzeichnis "bc" befindet. Diese dient ebenfalls als Bindeglied für das Compilieren von eigenen Programmen, die die "DELIB.dll" benutzen.

3.4. Umgebungsvariablen

Zwei Umgebungsvariablen weisen auf wichtige Verzeichnisse hin, die Dateien für die Programmiersprachen C, Delphi und Visual Basic enthalten.

"DELIB_INCLUDE" zeigt auf das Include-Verzeichnis.

%DELIB_INCLUDE% à c:\Programme\DEDITEC\DELIB\include"

"DELIB_LIB" zeigt auf das Library-Verzeichnis.

%DELIB_LIB% à c:\ Programme\DEDITEC\DELIB\lib

Anhang



4. Anhang

4.1. Revisionen

Rev 3.01	DEDITEC Design Update 2022
Rev 3.00	DEDITEC Design Update 2021
Rev 2.20	Kapitel "Software" und "DELIB API Referenz" überarbeitet
Rev 2.19	Neue Befehle "DapiDOSetBit32" und "DapiDOClrBit32" hinzugefügt. Diverse Delib Befehle überarbeitet
Rev 2.18	Neuer A/D Befehl "DapiSpecialADReadMultipleAD" und neues Kapitel "Software FIFO verwalten"
Rev 2.17	Kapitel "DELIB CLI (command-line interface)" ergänzt
Rev 2.16	Index hinzugefügt
Rev 2.15	Neuer CNT48 Befehl "DapiSpecialCNT48DIGet1" Ergänzung bei Befehl "DapiSpecialCMDSetDirDX8"
Rev 2.14	Ergänzung Kapitel "Software"
Rev 2.13	Ergänzung Kapitel "DELIB CLI" und Ergänzung der Parameter bei Verwaltungsfunktion "DapiSpecialCMDGetModuleConfig"
Rev 2.12	Ergänzung Kapitel "Digitale Zähler Funktionen", Kapitel "Temperatur Funktionen", Kapitel "Testprogramme" und neue Verwaltungsfunktion "DapiOpenModuleEx"
Rev 2.11	Neue DI Befehle "DapiSpecialDIFFFilterValueSet" und "DapiSpecialDIFFFilterValueGet"
Rev 2.10	Neue CNT48 Befehle "DapiCnt48ModeSet", "DapiCnt48ModeGet", "DapiCnt48CounterGet32" und "DapiCnt48CounterGet48"
Rev 2.09	Neuer Temperatur Befehl "DapiTempGet"
Rev 2.08	Ergänzung DELIB Console Execute und neue Verwaltungsfunktion

	"DapiSpecialCMDGetModuleConfig"
Rev 2.07	Neue Verwaltungsfunktion "DapiGetDELIBVersion", neuer DI Befehl "DapiSpecialCounterLatchAll", "DapiSpecialCounterLatchAllWithReset" und Ergänzung der Modi bei "DapiDIGetCounter"
Rev 2.06	Neuer DO Befehl "DapiDOSet1_WithTimer"
Rev 2.05	Ergänzung des Watchdog-Sticks
Rev 2.04	Neues Programmierbeispiel bei Befehl "DAPI_SPECIAL_CMD_SET_DIR_DX_1" Ergänzung des Return-Werts bei Befehl "DAPI_STEPPER_STATUS_GET_ACTIVITY" Ergänzung des Parameters Hold-Time (Zeit unendlich) bei Befehl "DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC"
Rev 2.03	Neuer Stepper Befehl "DAPI_STEPPER_CMD_GO_POSITION_RELATIVE"
Rev 2.02	Neuer D/A Befehl "DAPI_SPECIAL_CMD_DA" und DO Befehl "DAPI_SPECIAL_CMD_TIMEOUT_GET_STATUS"
Rev 2.01	Ergänzung der DELIB Befehle "DAPI_STEPPER_CMD_SET_MOTORCHARACTERISTIC" "DAPI_STEPPER_CMD_GET_MOTORCHARACTERISTIC" und "DAPI_STEPPER_CMD_GO_REFSWITCH"
Rev 2.00	Designanpassung
Rev 1.3	Softwareinstallation und Verzeichnisstruktur der DELIB
Rev 1.2	Ergänzung des Stepper Motors
Rev 1.1	Ergänzung von diversen AD/DA Befehlen
Rev 1.00	Erste DEDITEC Anleitung

4.2. 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.