EXDUL-393E

EDV-Nr.: A-382320

EXDUL-393S

EDV-Nr.: 382310

6 Temperaturmesseinheiten für PT100 und PT1000
1 Eingang bipolar über Optokoppler
1 Ausgang über Optokoppler
1 Zähler 32 Bit
LCD-Anzeige (nur EXDUL-393E)

wasco®

Handbuch

Geschützte Warenzeichen

Windows®, Visual Basic®, Visual C++®, Visual C#® sind eingetragene Warenzeichen von Microsoft.
wasco® ist ein eingetragenes Warenzeichen.
EXDUL® ist ein eingetragenes Warenzeichen.
LabVIEW® ist ein eingetragenes Warenzeichen.
Bei anderen genannten Produkt- und Firmennamen kann es sich um Warenzeichen der jeweiligen Inhaber handeln.

Haftungsbeschränkung

Die Firma Messcomp Datentechnik GmbH haftet für keinerlei durch den Gebrauch des Multifunktionsmoduls EXDUL-393 und dieser Dokumentation direkt oder indirekt entstandenen Schäden.

Wichtiger Hinweis:

Dieses Handbuch wurde für die Module EXDUL-393E und EXDUL-393S erstellt. Das EXDUL-393E bietet zusätzlich eine LCD-Anzeige, alle weiteren Funktionen der Module sind identisch. Für das EXDUL-393S sind die Befehle und Funktionen, die das Display betreffen, nicht zutreffend.

wasco®

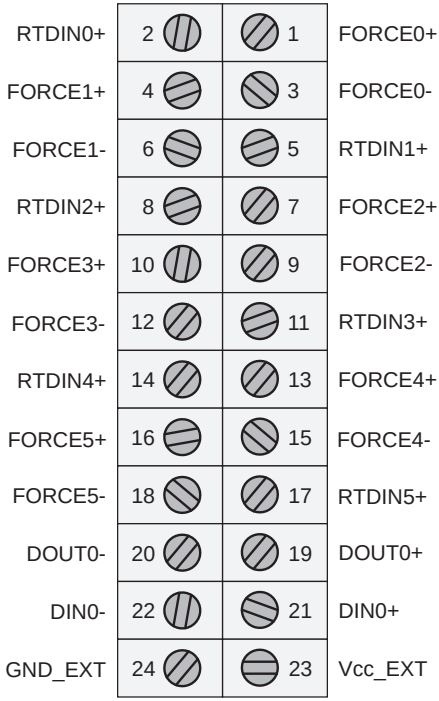
Inhaltsverzeichnis

1. Produktbeschreibung

Das EXDUL-393E verfügt über 6 Temperaturmesseinheiten für die Sensoren PT100 und PT1000 mit jeweils eigener Stromquelle und Messeingängen. Die Messung der einzelnen Sensoren erfolgt per Software-Befehl. Dabei kann sowohl die Temperatur als auch der Sensorwiderstand gemessen werden.
Zusätzlich hat das Modul einen digitalen Eingang und einen digitalen Ausgang mit galvanischer Trennung über hochwertige Optokoppler und zusätzlichen Schutzdioden. Der spezielle leistungsfähige Ausgangsoptokoppler bewältigen einen Schaltstrom von bis zu 150 mA.
Die programmierbare LCD-Anzeige beim EXDUL-393E ermöglicht die Darstellung von digitalen und analogen I/O-Statusinformationen oder programmierbaren anwenderspezifischen Daten.
Über USB oder eine externe Spannungquelle wird das Modul mit der notwendigen Betriebsspannung versorgt. Die Anschlüsse für die Spannungsversorgung sind wie die Anschlüsse des Eingangs- und Ausgangsoptokopplers einer 24poligen Schraubklemmleiste zugeführt.
Das kompakte Gehäuse erlaubt den Einsatz als mobiles Modul am Notebook sowie als Steuermodul im Steuerungs- und Maschinenbau mit einfacher Wandmontage oder unkomplizierter Montage auf DIN EN-Tragschienen.

2. Anschlussklemmen

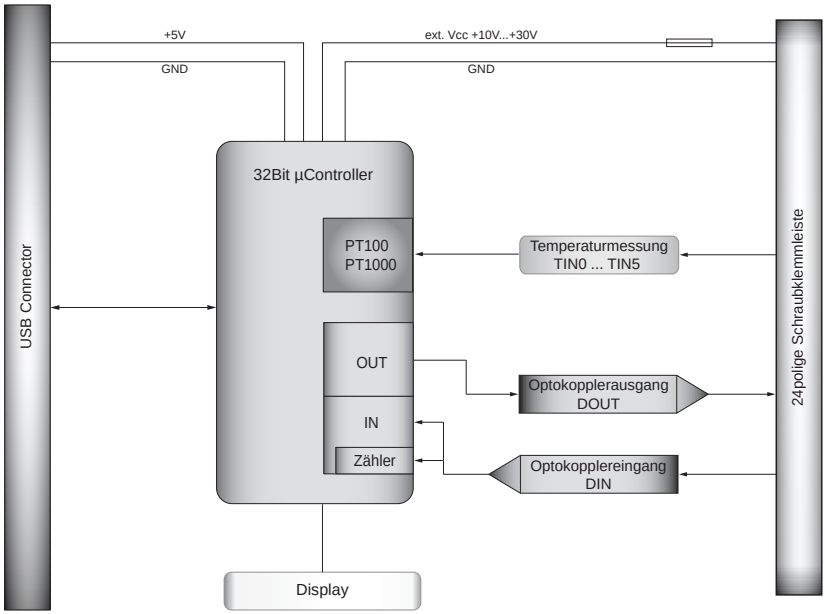
2.1 Klemmenbelegung von CN1



Vcc_EXT:
Anschlussklemme für externe Versorgungsspannung
GND_EXT:
Masse-Anschluss bei Verwendung einer externen Versorgungsspannung

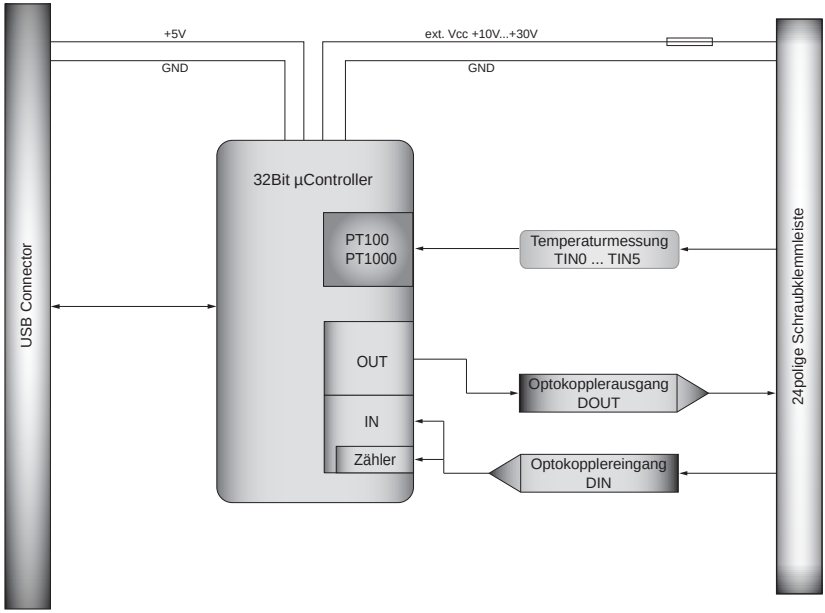
3. Systemkomponenten

3.1 Blockschaltbild EXDUL-393E



Grafik 3.1 Blockschaltbild EXDUL-393E

3.2 Blockschaltbild EXDUL-393S



Grafik 3.2 Blockschaltbild EXDUL-393S

3.3 Digitaler Eingang über Optokoppler

1 bipolarer Kanal
Überspannungsschutz-Dioden
Eingangsspannungsbereich
high = 10..30 Volt
low = 0..3 Volt

3.4 Digitaler Ausgang über Optokoppler

1 Kanal
Leistungsoptokoppler
Verpolungsschutz-Dioden
Ausgangsstrom: max. 150 mA
Spannung-CE: max. 50 V

3.5 Zähler

1 programmierbarer Zähler 32 Bit (belegt den Optokoppler-Eingang)
Zählfrequenz: max. 5 kHz
Automatische Sicherung der Zählerstände im 10kHz Takt

3.6 6 Temperaturmesseinheiten PT100

Sensortyp PT100 und PT1000 je Einheit individuell über Jumper wählbar
3-Leiteranschluss
Temperaturbereich: -200 bis 800°C
Auflösung typ. 0.03°C
Überspannungsschutz +/-45V

3.7 LCD Anzeige (nur EXDUL-393E)

Matrixanzeige mit 2 Zeilen und 16 Spalten zur Darstellung von 16 Zeichen je Zeile
Programmierbar zur Darstellung anwendungsspezifischer Daten oder als I/O-Zustandsanzeige

4. Inbetriebnahme

Der PC-Anschluss erfolgt einfach und unkompliziert Plug & Play über eine USB-Schnittstelle. Über USB oder eine externe Spannungsquelle wird das Modul mit der notwendigen Betriebsspannung versorgt.

4.1 Anschluss an einen USB-Port

Das EXDUL-393E / EXDUL-393S verfügt über ein USB 2.0 Interface und wird über die beiliegende USB-Anschlussleitung direkt an einen PC oder an einen USB-Hub angeschlossen. Der Anschluss erfolgt hotpluggable, d.h. das Modul ist auch im laufenden Betrieb anschließbar.

4.2 Spannungsversorgung über den USB-Port

Das Modul EXDUL-393 kann bei Bedarf ohne Einschränkungen ausschließlich über die USB-Schnittstelle versorgt werden. Dafür muss sichergestellt werden, dass der PC über das USB-Interface 500mA liefern kann.

4.3 Spannungsversorgung über externe Spannungsquelle

Die Firmware des EXDUL-393E / EXDUL-393S erkennt selbständig die Spannungsversorgung über eine externe Spannungsquelle. Wird an den Klemmen Vcc_EXT und GND_EXT (siehe Klemmenbelegung) eine Spannung von +10 V...+30 V DC angelegt, schaltet das Modul sofort auf Betriebsspannung „extern“ um. Die Spannungsversorgung über den USB-Port wird automatisch unterbrochen.
Achtung: Die Spannungsversorgung des Moduls darf während des Betriebs nicht mehr gewechselt werden!

4.4 LCD-Anzeige während der Inbetriebnahme (nur EXDUL-393E)

Während der Inbetriebnahme bzw. Start des Moduls erscheint im Display eine Infoanzeige in Form des Modulnamens. Nach fünf Sekunden wird der Modulname je nach LCD-Anzeigen-Konfiguration entweder durch die I/O-Statusanzeige oder UserLCD-Anzeige ersetzt.

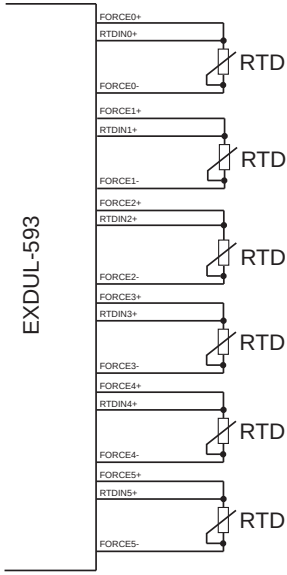
4.5 LCD-Anzeige während des Betriebs (nur EXDUL-393E)

Bei der Inbetriebnahme des Moduls schaltet das Display nach ca. fünf Sekunden, je nach Einstellung, von der Infoanzeige in die I/O-Statusanzeige oder die UserLCD-Anzeige. Während der I/O-Anzeige werden in Zeile1 die aktuellen Zustände der Eingänge, in Zeile2 die Zustände der Ausgänge angezeigt. Falls beim letzten Betrieb des Moduls mit vorgesehenem Befehl der UserLCD-Modus aktiviert wurde, erscheint anstelle der I/O-Statusanzeige die UserLCD-Anzeige mit den Werten aus den Speicherbereichen UserLCD1m und UserLCD2m. Die Daten aus den beiden Registern werden solange angezeigt, bis neue Benutzerdaten über UserLCD-Zeile1 und UserLCD-Zeile2 auf die Anzeige geschrieben werden. Um einen „Screen-Burn“ zu vermeiden, wechselt die Anzeige im laufenden Betrieb etwa jede Minute für ca. fünf Sekunden von der I/O-Statusanzeige oder UserLCD-Anzeige in die Infoanzeige.

5. Temperaturmesseinheiten

Das Modul EXDUL-393 besitzt zur Temperaturmessung mit PT100- und PT1000-Sensoren (IEC 751 α = 0.00385) 6 Messeinheiten, welche zur Bestimmung der Temperatur eine 3-Leiterschaltung verwendet. Die 3-Leiterschaltung sorgt für eine automatische Messfehlerkompensation der Sensorleitungen und führt zu einer genaueren Temperaturmessung. Dabei ist darauf zu achten, dass die Leitungsadern zum Sensor den selben Widerstand besitzen müssen (identische Länge, Querschnitt und Material). Jede Messeinheit liefert während der Temperaturmessung den nötigen Messstrom und übergibt dem Anwender nach abgeschlossener Messung den Temperaturwert und bei Bedarf den Widerstandswert.
Um für die Temperaturmessung den idealen Messstrom zu gewährleisten, besitzt jede Messeinheit einen eigenen Jumper, mit welchem die Sensorart (PT100 oder PT1000) konfiguriert werden kann. Für einen Hardwareschutz, hat jede Messeinheint einen Überspannungsschutz von +/-45V.

5.1 Beschaltung



5.2 Messmöglichkeiten

Bei der Durchführung einer Messung gibt es mehrere Modi, welche bestimmen, wie das Messergebnis verarbeitet oder als Befehlsantwort an den Anwender geschickt werden soll (z.B. mΩ oder °C)

5.2.1 Widerstandsmessung

In diesem Modus wird der angeschlossene Widerstand gemessen und in mΩ an den PC übergeben. Der zu messende Widerstand darf im Bereich von 0 bis 370Ω liegen. Die Messart ist nur in der PT100-Konfiguration verfügbar.

5.2.2 Temperaturmessung PT100/PT100 nach IEC 751

Wird dieser Modus verwendet, so misst das Modul den Sensorwiderstand und berechnet die daraus resultierende Temperatur mit Hilfe der aus IEC 751 (α=0.00385) vorgegebenen Kennlinie des Sensors. Je nach Sensor ist ein Temperaturbereich von -200°C bis 800°C mit einer Auflösung von typ. 0.03°C möglich. Die Temperatur wird mit dem Faktor 100 an den PC zurück geschickt.

Verwendete Callendar-Van Dusen Koeffizienten:
$a = 3.908030 \times 10^{-3}$
$b = -5.7750 \times 10^{-7}$
$c = -4.18301 \times 10^{-12}$

5.3 Fehlererkennung

Um Fehler bei der Temperaturmessung erkennen zu können, gibt es die Möglichkeit, einen Fehlertest durchzuführen. Hier können Fehler wie Leiterbruch, Kurzschlüsse und Über-/Unterspannung erkannt werden.
Wird mit Aufruf des entsprechenden Befehls ein Fehlertest durchgeführt, wird nach einigen wenigen ms ein Fehlerbyte an den Anwender gesendet, welches der Fehleranalyse dient.
Während des Fehlertests kann keine Temperaturmessung durchgeführt werden.

Codes bei Fehlermeldung

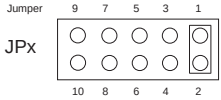
Fehler-Bit	mögliche Fehlerursache	Fehlerbeschreibung
D7	reserviert	
D6	reserviert	
D5	Fehler bei der Verdrahtung	
D4	Fehler bei der Verdrahtung	
D3	Fehler bei der Verdrahtung	
D2	Overvoltage oder Under-voltage	evtl. externe Spannung ein-gespeist

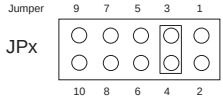
Steht in einem Bit eine 1, so liegt entsprechend dessen Bedeutung ein Fehler vor.

5.4 Sensorart konfigurieren

Jede Messeinheit des Moduls ist mit den Sensoren PT100 und PT1000 kompatibel. Um einen idealen Messstrom generieren zu können, muss der gewüschte Sensortyp zum einen innerhalb der Software (Softwarebefehl oder Webpage) eingestellt werden und zum anderen der passende Jumper im Modul gesetzt werden.
Um Werkszustand sind alle Messeinheiten für den PT100 konfiguriert. Soll mit einer oder mehreren Einheiten ein PT1000 Sensor gemessen werden, so gehen Sie wie folgt vor:

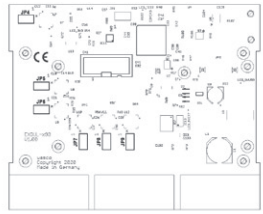
1. Entfernen Sie den Gehäusedeckel. Dafür müssen Sie mit einem Schraubenzieher das Gehäuse an den 4 Laschen auf der Bodenrückseite etwas nach außen drücken und den Deckel herunterziehen.
2. Setzen Sie die Jumper wie gewünscht. Für eine PT100-Messung setzen sie den Jumper zwischen Pin 1 und 2, für eine PT1000-Messung zwischen Pin 3 und 4 (siehe Grafiken).





Folgende Jumperblöcke sind den jeweiligen Messeinheiten zugeordnet.

Jumperblock-Nummer	Messeinheit
JP4	Messeinheit 0
JP5	Messeinheit 1
JP6	Messeinheit 2
JP7	Messeinheit 3
JP8	Messeinheit 4
JP9	Messeinheit 5



3. Anschließend montieren Sie den Deckel wieder auf den Boden. Achten Sie dabei darauf, dass die Kontaktklemmen richtig in die Führungen des Deckels passen.

5.5 Abgleich bzw. Kalibrierung der Messeinheiten

Jede Messeinheit des Moduls wird vor der Auslieferung bereits für die Sensoren PT100 und PT1000 mit Präzisionswiderständen abgeglichen. Sollte ein erneuter Abgleich aufgrund besonderer Einflüsse (hohe oder niedrige Außentemperatur) nötig sein, so kann dieser entweder mit dem Kalibrierungsbefehl im Anwenderprogramm oder über die Webpage abgeglichen werden. Vor der Kalibrierung muss sowohl an die abzugleichende Messeinheit entweder ein genauer Messwiderstand (PT100 = 100R, PT1000 = 1K) oder ein auf 0°C gekühlter Sensor angeschlossen werden als auch die verwendete Sensorart konfiguriert werden. Beachten Sie, dass die Leitungslängen/Leitungswiderstände identisch sind.

6. 1 Optokopplereingang

Das EXDUL-393 verfügt über 1 Eingangskanal, dessen galvanische Trennung mittels Optokoppler erreicht wird. Die Isolationsspannung zwischen Masse des Moduls und Eingang beträgt 500 Volt.

6.1 Pinbelegung des Eingangsoptokopplers

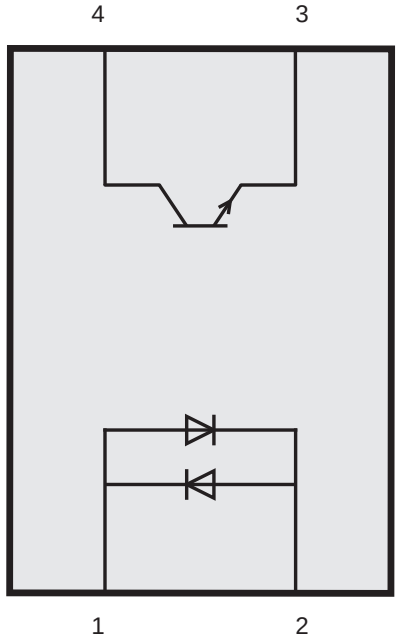


Abb. 7.1

6.2 Eingangsbeschaltung

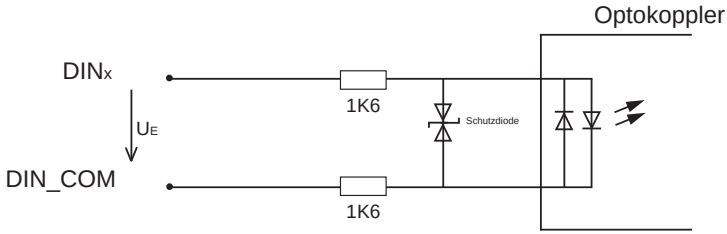


Abb. 6.2 Eingangsbeschaltung

Die Eingänge des Optokopplers sind bipolar ausgeführt. Im Normalfall wird der DIN_COM/DIN0- Anschluß auf Minus gelegt und am DIN0+ -Anschluss eine Spannung angelegt. Sie können jedoch auch, falls es schaltungstechnisch sinnvoller ist, am DIN_COM/DIN0- Anschluß die Plus-Spannung und am DIN0+ Anschluß die Minus-Spannung anlegen.

6.3 Eingangsstrom

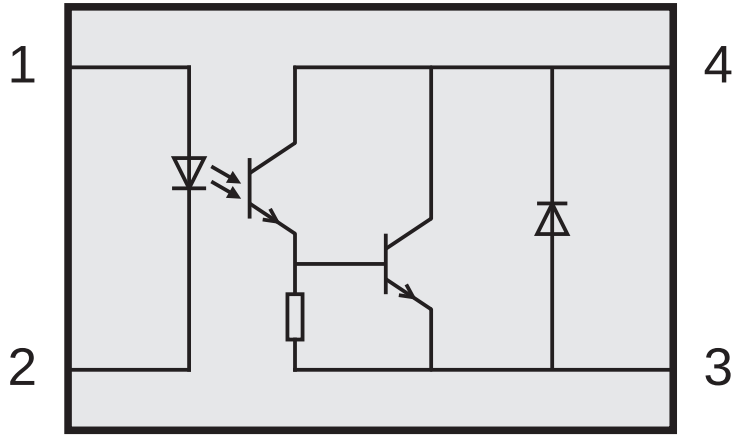
$$I_E \approx \frac{U_E - 1{,}1V}{3200\Omega}$$

Bei einer Eingangsspannung zwischen DIN0+ und DIN0- von 24 Volt ergibt sich ein Eingangsstrom von ca. 7mA, bei 12V von ca. 3,4 mA.

7. 1 Optokopplerausgang

Das EXDUL-Modul verfügt über einen Ausgangskanal, dessen galvanische Trennung ebenfalls mittels Optokoppler erreicht werden. Die Isolationsspannung zwischen Masse des Moduls und Ausgang beträgt 500 Volt.

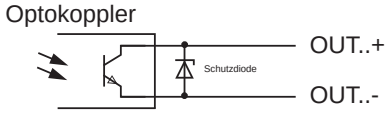
7.1 Pinbelegung des Ausgangsoptokopplers



7.2 Optokopplerdaten

Spannung-CE:	max. 50V
Spannung-EC:	0,1V
Strom-CE:	150 mA

7.3 Ausgangsbeschaltung

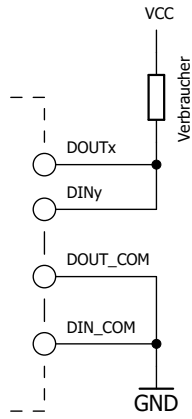


7.4 Programmierung des Optokopplerausgangs

Für die Programmierung des Optokopplerausgangs stehen mehrere Zugriffsfunktionen zur Verfügung. So kann der Ausgangskanal einfach beschrieben werden. Sollte der aktuelle Zustand des Ausgangs im Anwenderprogramm nicht speicherbar sein, so kann dieser über einen Lese-Befehl rückgelesen werden. Hier ist zu beachten, dass der gelesene Schaltzustand nur dem primären Zustand (auf Prozessorseite) entspricht. Soll der tatsächliche Schaltzustand bzw. der Pegel des Ausgangs rückgelesen werden, muss dafür ein Optokopplereingang verwendet werden (siehe Kapitel Optokopplerausgang rücklesen).
Eine detailierte Beschreibung der Programmierung ist im Kapitel Programmierung zu finden.

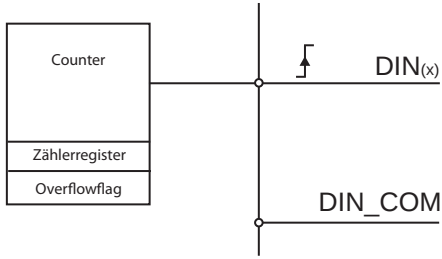
7.5 Optokopplerausgang rücklesen

Gelegentlich ist es bei Applikationen nötig, den tatsächlichen Ausgangszustand in dem Programm zu wissen. Ein Beispiel kann z. B. eine Fehlererkennung im Programm sein. Dies kann durch die Rückführung des Optokopplerausgangs zu dem Optokopplereingang erfolgen. Im folgenden Beschaltungsbeispiel wird der Ausgang x mit dem Eingang y verbunden. Beachten Sie bitte, daß in dieser Schaltungsversion das Ergebnis des Eingangs negiert ist. Wenn der Ausgang geschaltet ist, liegt am Transistor keine Spannung an und somit zeigt der Eingang eine „0“ an. Zudem fließt bei nicht geschaltetem Ausgang über den Optokoppler-Eingang und damit auch über den Verbraucher ein geringer Strom (bei 24V ca. 7mA).



8. Zähler

Das Modul stellt für den Optokopplereingang (DIN0) einen eigenständigen, hardwareunterstützten 32bit Zähler zur Verfügung. Bei Bedarf kann dieser aktiviert werden und reagiert bei jeder steigenden Flanke durch die Inkrementierung des Zählerstandes. Ein abrufbares Flag signalisiert einen Overflow.
Damit bei einer außerplanmäßigen Unterbrechung der Spannungsversorgung der Zählerstand nicht verloren gehen, wird dieser ungefähr alle 100µs gesichert. Liegt dann am Modul wieder eine Spannungsversorgung an, wird der gesicherte Wert automatisch in das Zählerregister geladen.



Funktionen	Beschreibung
Start	Startet den Zähler bzw. gibt den Eingang frei
Stop	Stoppt den Zähler, Signal an dem Eingang wird ignoriert
Reset	Setzt den Zählerstand auf 0
Zählerstand lesen	Liest den aktuellen Zählerstand
Overflowflag lesen	Liest das Overflowflag
Clear Overflowflag	Setzt das Overflowflag zurück

9. Informations-, LCD- und Userregister

9.1 Register HW-Kennung und Seriennummer

Byte	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
HW-Kennung	E	X	D	U	L	-	3	9	3			V	1	.	0	1
	45_{hex}	58_{hex}	44_{hex}	55_{hex}	$4C_{hex}$	$2D_{hex}$	33_{hex}	39_{hex}	33_{ex}	20_{hex}	20_{hex}	56_{hex}	31_{hex}	$3E_{hex}$	30_{hex}	31_{hex}
S/N	1	0	4	4	0	2	6									
	31_{hex}	30_{hex}	34_{hex}	34_{hex}	30_{hex}	32_{hex}	36_{hex}									

Tabelle 9.1 Register HW-Kennung und Seriennummer

Im Register HW-Kennung ist der Modulname sowie die Version der Firmware abgelegt und kann zur Feststellung der Produkt-Identität vom User gelesen werden. In der o. a. Tabelle sind als Beispiel in der Zeile HW-Kennung jeweils der Hex-Wert und das dazugehörige ASCII-Zeichen für das Modul EXDUL-393 mit Firmware-Version 1.01 dargestellt.

Das Register Serien-Nummer kann vom Anwender lediglich gelesen werden. Die Serien-Nummer in der o. a. Tabelle dient als Formatbeispiel.
In der Zeile S/N ist jeweils der Hex-Wert und darüber das dazugehörige ASCII-Zeichen für die Serien-Nummer 1044026 dargestellt.

9.2 Speicherbereiche UserA, UserB, UserLCD1m* und UserLCD2m*

Byte	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
UserA																
	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}
UserB																
	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}
UserLCD1m*																
	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}
UserLCD2m*																
	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}	20_{hex}

In den Registern UserA, UserB, UserLCD1m* und UserLCD2m* können jeweils 16 Stellen (16 Byte) zur eigenen Verwendung genutzt werden. Die Daten bleiben beim Ausschalten erhalten, ein Default-Reset setzt diese Register in die Werkseinstellung (Auslieferungszustand) zurück. Im Auslieferungszustand steht in allen vier User-Speicherbereichen an jeder Stelle der Hex-Wert 20, der im ASCII-Code einem Leer-Zeichen entspricht. In der o. a. Tabelle sind jeweils der Hex-Wert und darüber das dazugehörige ASCII-Zeichen dargestellt.

9.3 Display-Register UserLCD-Zeile1*, UserLCD-Zeile2* und LCD-Kontrast*

Die Register UserLCD-Zeile1 und UserLCD-Zeile2 dienen bei aktivierten UserLCD-Modus zum Beschreiben der beiden LCD-Zeilen mit jeweils 16 beliebigen Zeichen. Mit Übernahme der Daten erfolgt die Anzeige im Display anstelle der Daten aus UserLCD1m* und UserLCD2m*. Die Daten in den Registern UserLCD-Zeile1 und UserLCD-Zeile2 bleiben beim Ausschalten **nicht** erhalten. Über das Register LCD-Kontrast ist der Display-Kontrast einstellbar, der auch beim Ausschalten erhalten bleibt.

*: Nur für EXDUL-393E zutreffend, bei EXDUL-393S ohne Funktion!

10. Installation der Treiber

10.1 Windows-Treiber

Achtung: ab Windows10 muss kein extra Treiber für das Modul installiert werden!

Sobald das USB-Modul EXDUL-393E / EXDUL-393S das erste Mal am PC angeschlossen wird, erkennt Windows automatisch ein neues Gerät und sucht nach einem passenden Treiber.

Geben Sie zur Treiberinstallation dem Windows-Hardwareassistenten den Ordner bzw. das Verzeichnis und den Namen der Setup-Datei „wascoxmfe_v0**x**.inf“ (anstelle von x die Versions-Nr. der INF-Datei eintragen z.B. wascoxmfe_v0**6**.inf) an.

Nach der Aktualisierung der Treiberdatenbank informiert Sie der Hardwareassistent über die erfolgreiche Installation des Treibers.

Im Windows-Gerätemanager wird das EXDUL-393E / EXDUL-393S im Verzeichnis Anschlüsse (COM/LPT) als Wasco-USB-Kommunikationsport oder Serielles USB-Gerät COMx geführt. Jedes Windowsprogramm kann auf die virtuelle Schnittstelle so zugreifen, als handle es sich um einen echten COM-Port.

10.2 Linux-Treiber

Das EXDUL-393 verwendet einen virtuellen Standard COM-Port-Treiber, welcher bei den meistern gängigen Linux-Distributionen bereits installiert ist.
Wird das Modul an die USB-Schnittstelle angeschlossen, wird das Modul im dev-Ordner aufgelistet (z.B. als ttyACM0 unter Ubuntu).

11. Programmierung unter Windows®

11.1 Einführung

Nach erfolgreicher Installation wird das EXDUL-393E / EXDUL-393S im Windows-Gerätemanager als Wasco-Communications-Port COMx bzw. als serielles USB-Gerät COMx (ab WIN10) geführt. Es handelt sich hierbei um ein CDC-Device (Communications Device Class), das über einen virtuellen COM-Port angesprochen wird.
Der Softwarezugriff auf diesen virtuellen COM-Port erfolgt wie über eine normale COM-Schnittstelle über Standard-Windows®-Treiber, eine Installation eines zusätzlichen Treibers ist nicht notwendig.

11.2 Programmierarten

Für den Zugriff auf das EXDUL-Modul gibt es mehrere Möglichkeiten. So kann für die Programmierung unter .NET die Library EXDUL.dll verwendet werden. Diese ermöglicht eine leichten und schnellen Einstieg, um den Zugriff auf das Modul zu programmieren.
Des Weiteren können auch serielle COM-Port-Libraries verwendet werden, welche bei vielen Programmiersprachen wie C oder Delphi vorhanden sind. Sie ermöglichen oft eine breite Einstellmöglichkeit der Schnittstelle und teilweise auch eine Eventprogrammierung (Lesepuffer muss nicht gepolled werden).
LabVIEW-Anwender können ebenfalls mit Hilfe der EXDUL.dll oder den VISA-Funktionsblöcken (Serial Port) leicht auf das Modul zugreifen.

11.3 Programmierung unter Windows mit der .NET EXDUL.dll Library

Wird für den Modul-Zugriff eine .NET-Programmiersprache verwendet (C#, C++.NET oder VB.NET), so kann die Library EXDUL.dll verwendet werden. Sie besitzt einen objektorientierten Aufbau, in welchem jedes EXDUL-Modul durch ein Objekt mit ihren Methoden dargestellt wird.
Bei der Entwicklung der Library wurde auf eine möglichst einheitliche API zwischen den unterschiedlichen EXDUL-Modulen geachtet.
Dies ermöglicht es dem Anwender, bei Bedarf ohne großen Programmieraufwand von z.B. einem USB-EXDUL-Modul auf ein Ethernet-EXDUL-Modul (z.B. EXDUL-393 -> EXDUL-593) zu wechseln.

Open: Verbindung zu Modul aufbauen
bool Open()
Rückgabewerte: true wenn erfolgreich / false bei Fehler

Close
void_Close()
Zusammenfassung: Verbindung zu Modul schließen

Schreiben in Inforegister:
void SetModulInfo (byte type, string info)
Parameter: type: Info-Typ (siehe Handbuch)
info: Bis zu 16 Zeichen langer Info-String
Zusammenfassung: Beschreibt die Modul-Informationsregister

Infobereich	Info-Byte
UserA	0
UserB	1

Lesen aus Inforegister:
string GetModulInfo(byte type)
Parameter: type: Info-Typ (siehe Handbuch)
Rückgabewerte: Gibt das Register "type" als string zurück
Zusammenfassung: Liest die Modul-Information-Register aus

Infobereich	Info-Byte
UserA	0
UserB	1
Hardwarekennung	3
Seriennummer	4

Schreiben in LCD-Register UserLCD:
void SetUserLCD(byte *line*, string *text*)

Parameter:	*line*: 0 = 1. Zeile / 1 = 2. Zeile *text*: Bis zu 16 Zeichen langer LCD-Text
Zusammenfassung:	Beschreibt die UserLCD-Register. Der Parameter *line* legt die Zeile (0 oder 1) fest und *text* den Text aus 16 Zeichen.

Schreiben in LCD-Register UserLCDm:
void SetUserLCDm(byte *line*, string *text*)

Parameter:	*line*: 0 = 1. Zeile / 1 = 2. Zeile *text*: Bis zu 16 Zeichen langer LCD-Text
Zusammenfassung:	Beschreibt die UserLCDm-Register. Der Parameter *line* legt die Zeile (0 oder 1) fest und *text* den Text aus 16 Zeichen

Schreiben des LCD-Modes:
void SetLCDMode(byte *mode*)
Parameter: *mode*: LCD-Modus
Zusammenfassung: Setzt den LCD-Modus fest

LCD-Modus	LCD-Modus-Byte
IO-Mode	0
User-Mode	1

Lesen des LCD-Modes:
byte GetLCDMode()
Rückgabewerte: LCD-Modus
Zusammenfassung: Liest den LCD-Modus aus

LCD-Modus	LCD-Modus-Byte
IO-Mode	0
User-Mode	1

Schreiben LCD-Kontrastwert:
void SetLCDContrast(ushort *contrast*)
Parameter: *contrast*: Wert zwischen 0 und 4095 (empfohlen 800 bis 1800)
Zusammenfassung: Legt den LCD-Kontrast fest

Lesen LCD-Kontrastwert:
ushort GetLCDContrast()
Rückgabewerte: LCD-Kontrast
Zusammenfassung: Liest den LCD-Kontrast aus

Optokopplerausgänge lesen:
uint GetOptoOut()
Rückgabewerte: Zustand der Optokopplerausgänge
Zusammenfassung: Liest den Zustand der Optokopplerausgänge

Optokopplerausgänge schreiben:
void SetOptoOut(uint *value*)
Parameter: *value*: Zustand der Ausgänge
Zusammenfassung: Setzt die Optokopplerausgänge

Optokopplereingänge lesen:
uint GetOptoIn()
Rückgabewerte: Aktueller Zustand der Optokopplereingänge
Zusammenfassung: Liest den aktuellen Zustand an den Optokoppler-eingängen

---

Zähler starten:
void StartCounter(byte *index*)
Parameter: *index*: Counter-Index
Zusammenfassung: Startet den Zähler mit der Nummer index

Zähler stoppen:
void StopCounter(byte *index*)
Parameter: *index*: Counter-Index
Zusammenfassung: Stoppt den Zähler mit der Nummer index

Zähler resetten:
void ResetCounter(byte *index*)
Parameter: *index*: Counter-Index
Zusammenfassung: Setzt den Zählerstand des Zählers mit der Nummer index zurück auf 0

Zählerstand lesen:
uint ReadCounter(byte *index*)
Parameter: *index*: Counter-Index
Rückgabewerte: Zählerstand
Zusammenfassung: Liest den Zählerstand des Zählers mit der Nummer index aus

---

Overflow-Flag lesen:
bool ReadOverflowFlagCounter(byte *index*)
Parameter: *index*: Counter-Index
Rückgabewerte: Overflowflag false = kein Overflow
true = Overflow
Zusammenfassung: Liest das Overflowflag des Zählers mit der Nummer index aus

---

Overflow-Flag rücksetzen:
void ResetOverflowFlagCounter(byte *index*)
Parameter: *index*: Counter-Index
Zusammenfassung: Setzt das Overflowflag des Zählers mit der Nummer index zurück

---

Temperatur messen:
int GetTemperature_PT(byte *channel*)
Parameter: *channel*: Kanalauswahl
Rückgabewerte: Temperatur in °C * 100
Zusammenfassung: Misst die Temperatur durch den angeschlossenen PT-Sensor

---

Temperatursensor-Typ einstellen:
void SetSensortypeTempUnit(byte *channel, byte sensor*)
Parameter: *channel*: Kanalauswahl
senosor: Sensortyp (0=PT100, 1=PT1000)
Zusammenfassung: Stellt den Sensortyp ein und speichert ihn ab

---

Temperatur-Messeinheit kalibrieren:
void CalibrateTempUnit_PT(byte *channel*)
Parameter: *channel*: Kanalauswahl
Zusammenfassung: Mit der Funktion kann die Messeinheit zur Temperaturmessung kalibriert werden. Dazu muss ein im Handbuch festgelegter Widerstand angelegt werden

---

Fehlererkennung an der Temperaturmesseinheit durchführen:
int FaultDetectionTempUnit(byte *channel*)
Parameter: *channel*: Kanalauswahl
Rückgabewerte: Fehlerregister
Zusammenfassung: Führt eine Fehlererkennung an der gewünschten Tempearturmesseinheit channel durch

---

Werksreset:
void DefaultReset()
Zusammenfassung: Setzt das Modul auf die Werkseinstellung zurück. Nach dem Befehl muss das Modul geschlossen und wieder neu geöffnet werden

11.4 Programmierung mit seriellen COM-Port-Libraries

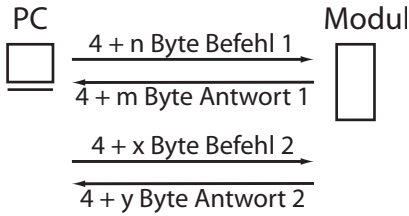
Durch die Möglichkeit mit Standard-COM-Port-Libraries auf das Modul zugreifen zu können, kann der Anwender mit einer Vielzahl an Sprachen seine Anwendung programmieren. So kann neben dem .NET-Framework auch Delphi oder C verwendet werden. Auch können Anwendungen auf vielen Linux basierten Betriebssystemen (falls ein virtueller COM-Port-Treiber vorhanden ist) entworfen werden.

11.4.1 Kommunikation mit dem EXDUL-393

Der Datenaustausch erfolgt durch Senden bzw. Empfangen von Byte-Arrays mit unterschiedlicher Länge über die virtuelle COM-Schnittstelle.

Jeder erlaubte Sendestring wird mit einem definierten Ergebnis- bzw. Bestätigungsstring beantwortet.

Vor dem Senden eines Strings muss der letzte Ergebnis- bzw. Bestätigungsstring gelesen werden.



Grafik 11.4 Kommunikationsmodell

11.4.2 Windows®-Funktionen für die Programmierung

Die Programmierung des EXDUL-393E / EXDUL-393S erfolgt entweder über WIN32 API Funktionen oder sehr komfortabel über ein bereits vorhandenes SerialPort Object in einer Programmiersprache. Beispielprogramme hierzu finden Sie nach der Installation der Software im Installationsverzeichnis auf Ihrem Rechner.

Windows-Funktionen für die Programmierung:

- CreateFile
- GetCommState
- SetCommState
- WriteFile
- ReadFile
- DCB-Struktur (beschreibt die Kontroll-Parameter des Devices)

11.4.3 Befehls- und Datenformat

Der Datenaustausch erfolgt durch Senden und Empfangen von Byte-Arrays. Jedes zu sendende bzw. zu empfangende Byte-Array besteht aus mindestens 4 Bytes. Dabei stellen die ersten drei Bytes den Befehl und das vierte die Anzahl der noch folgenden 4 Byte-Blöcke dar.

Befehl Byte 0	Befehl Byte 1	Befehl Byte 2	Längenbyte

Die Anzahl der 4-Byte-Blöcke variiert von Befehl zu Befehl und ist zum Teil von der zu sendenden Datenmenge abhängig. Genauere Informationen befinden sich bei den einzelnen Befehlsbeschreibungen.

11.4.4 Befehlsübersicht

Hexcode	Beschreibung
0C 00 00	Inforegister lesen und schreiben
0C 00 03	LCD-Register lesen und schreiben
08 00 00	Optokopplerausgänge lesen und schreiben
08 00 01	Optokopplereingänge bearbeiten
09 00 00	Zähler0
0A 04 00	Temperaturmesseinheit Temperatur-/Widerstandsmessung
0A 04 01	Temperaturmesseinheit Fehlererkennung durchführen
0A 04 08	Temperaturmesseinheit Sensortyp einstellen
0A 04 09	Temperaturmesseinheit oberer Schwellewert definieren
0A 04 0A	Temperaturmesseinheit unterer Schwellenwert definieren
0A FF F7	Temperaturmesseinheit kalibrieren

11.4.5 Befehlszusammensetzung

Schreiben in Inforegister

Das EXDUL-Modul stellt mehrere beschreibbare Inforegister zur Verfügung. UserA/B sind zwei 16-Byte-Bereiche für den Anwender, um Informationen in einem nicht-flüchtigen Speicher (FLASH) zu sichern. Die Register sind nur als ganzer 16-Byte-Block beschreibbar.

Infobereich	Info-Byte
UserA	0
UserB	1

Beispiel: Schreiben der Zeichenfolge EXDUL-393 in Register UserA und UserB

Byte	Senden	Empfangen	Beschreibung
0	0C	0C	Befehlscode 1. Byte
1	00	00	Befehlscode 2. Byte
2	00	00	Befehlscode 3. Byte
3	05	00	Längenbyte → 20 Byte
4	00 (UserA) 01 (UserB)		Info-Byte
5	00		reserviert
6	00		reserviert
7	00		Schreibfunktion Infobereich
8	45		Daten 1. Zeichen E_{as}
9	58		Daten 2. Zeichen X_{asci}
10	44		Daten 3. Zeichen D_{asci}
11	55		Daten 4. Zeichen U_{asci}
12	4C		Daten 5. Zeichen L_{asci}
13	2D		Daten 6. Zeichen $-_{asci}$
14	33		Daten 7. Zeichen 3_{asci}
15	39		Daten 8. Zeichen 9_{asci}
16	33		Daten 9. Zeichen 3_{asci}
17	20		Daten 10. Zeichen $[Leer]_{asci}$
18	20		Daten 11. Zeichen $[Leer]_{asci}$
19	20		Daten 12. Zeichen $[Leer]_{asci}$
20	20		Daten 13. Zeichen $[Leer]_{asci}$
21	20		Daten 14. Zeichen $[Leer]_{asci}$
22	20		Daten 15. Zeichen $[Leer]_{asci}$
23	20		Daten 16. Zeichen $[Leer]_{asci}$

Lesen aus Inforegister

Das EXDUL-Modul besitzt mehrere 16-Byte breite Infobereiche, in welchen Modulinformationen wie die Seriennummer oder die Hardwarekennung stehen. Des Weiteren kann der Anwender auch die beschreibbaren User-Register auslesen.

Infobereich	Info-Byte
UserA	0
UserB	1
Hardwarekennung	3
Seriennummer	4

Info: Alle Infobereiche lassen sich nur als ganzer 16-Byte-Block auslesen.

Beispiel: Infobereich UserA auslesen (User-String = „EXDUL-393“)
Gesendet wird ein 8Byte langer Block und empfangen ein 20Byte langer Block mit Inhalt von UserA bzw. UserB

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte → 4Byte	04	Längenbyte → 16Byte
4	00 (UserA) 01 (UserB)	Info-Byte	45	Daten 1. Zeichen E_{asci}
5	00	reserviert	58	Daten 2. Zeichen X_{asci}
6	00	reserviert	44	Daten 3. Zeichen D_{asci}
7	01	Lesefunktion Infobereich	55	Daten 4. Zeichen U_{asci}
8			4C	Daten 5. Zeichen L_{asci}
9			2D	Daten 6. Zeichen $-_{asci}$
10			33	Daten 7. Zeichen 3_{asci}
11			39	Daten 8. Zeichen 9_{asci}
12			33	Daten 9. Zeichen 3_{asci}
13			20	Daten 10. Zeichen $[Leer]_{asci}$
14			20	Daten 11. Zeichen $[Leer]_{asci}$
15			20	Daten 12. Zeichen $[Leer]_{asci}$
16			20	Daten 13. Zeichen $[Leer]_{asci}$
17			20	Daten 14. Zeichen $[Leer]_{asci}$
18			20	Daten 15. Zeichen $[Leer]_{asci}$
19			20	Daten 16. Zeichen $[Leer]_{asci}$

Beispiel: Infobereich Hardwarekennung auslesen
Gesendet wird ein 8Byte langer Block und empfangen ein 20Byte langer Block mit der Hardwarekennung

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte → 4Byte	04	Längenbyte → 16Byte
4	04	Info-Byte	45	Daten 1. Zeichen E_{asci}
5	00	reserviert	58	Daten 2. Zeichen X_{asci}
6	00	reserviert	44	Daten 3. Zeichen D_{asci}
7	01	Lesefunktion Infobereich	55	Daten 4. Zeichen U_{asci}
8			4C	Daten 5. Zeichen L_{asci}
9			2D	Daten 6. Zeichen $-_{asci}$
10			33	Daten 7. Zeichen 3_{asci}
11			39	Daten 8. Zeichen 9_{asci}
12			33	Daten 9. Zeichen 3_{asci}
13			20	Daten 10. Zeichen $[Leer]_{asci}$
14			20	Daten 11. Zeichen $[Leer]_{asci}$
15			20	Daten 12. Zeichen $[Leer]_{asci}$
16			20	Daten 13. Zeichen $[Leer]_{asci}$
17			20	Daten 14. Zeichen $[Leer]_{asci}$
18			20	Daten 15. Zeichen $[Leer]_{asci}$
19			20	Daten 16. Zeichen $[Leer]_{asci}$

Beispiel: Infobereich Seriennummer auslesen
Gesendet wird ein 8Byte langer Block und empfangen ein 20Byte langer Block mit der Seriennummer

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte → 4Byte	03	Längenbyte → 16Byte
4	04	Info-Byte	31	Daten 1. Zeichen 1_{dez}
5	00	reserviert	30	Daten 2. Zeichen 0_{dez}
6	00	reserviert	34	Daten 3. Zeichen 4_{dez}
7	01	Lesefunktion Infobereich	34	Daten 4. Zeichen 4_{dez}
8			30	Daten 5. Zeichen 0_{dez}
9			32	Daten 6. Zeichen 2_{dez}
10			36	Daten 7. Zeichen 6_{dez}
11				reserviert
12				reserviert
13				reserviert
14				reserviert
15				reserviert
16				reserviert
17				reserviert
18				reserviert
19				reserviert

Schreiben in LCD-Register

Das EXDUL-Modul stellt mehrere beschreibbare LCD-Register zur Verfügung. UserLCD1 und UserLCD2 entsprechen den beiden Zeilen während der UserMode-LCD-Anzeige. UserLCD1m und UserLCD2m sind zwei 16-Byte-Bereiche, welche direkt in einen nicht-flüchtigen Speicher (FLASH) abgelegt werden und beim Modulstart in die Register UserLCD1m bzw. UserLCD2m geladen werden. Alle Register sind nur als ganze 16-Byte-Blöcke beschreibbar.

LCD-Befehl	LCD-Befehl-Byte
UserLCD1	0
UserLCD2	1
UserLCD1m	2
UserLCD2m	3

Beispiel: Schreiben der Zeichenfolge EXDUL-393 in Register

Byte	Senden	Empfangen	Beschreibung
0	0C	0C	Befehlscode 1. Byte
1	00	00	Befehlscode 2. Byte
2	03	03	Befehlscode 3. Byte
3	05	00	Längenbyte → 20 Byte
4	00 (UserLCD1) 01 (UserLCD2) 02 (UserLCD1m) 03 (UserLCD2m)		LCD-Befehl
5	00		reserviert
6	00		reserviert
7	00		Schreibfunktion
8	45		Daten 1. Zeichen E_{as}
9	58		Daten 2. Zeichen X_{asci}
10	44		Daten 3. Zeichen D_{asci}
11	55		Daten 4. Zeichen U_{asci}
12	4C		Daten 5. Zeichen L_{asci}
13	2D		Daten 6. Zeichen $-_{asci}$
14	33		Daten 7. Zeichen 3_{asci}
15	39		Daten 8. Zeichen 9_{asci}
16	33		Daten 9. Zeichen 3_{asci}
17	20		Daten 10. Zeichen $[Leer]_{asci}$
18	20		Daten 11. Zeichen $[Leer]_{asci}$
19	20		Daten 12. Zeichen $[Leer]_{asci}$
20	20		Daten 13. Zeichen $[Leer]_{asci}$
21	20		Daten 14. Zeichen $[Leer]_{asci}$
22	20		Daten 15. Zeichen $[Leer]_{asci}$
23	20		Daten 16. Zeichen $[Leer]_{asci}$

Lesen von LCD-Register

Das EXDUL-Modul stellt mehrere beschreib- bzw. lesbare LCD-Register zur Verfügung. UserLCD1 und UserLCD2 entsprechen den beiden Zeilen während der UserMode-LCD-Anzeige. UserLCD1m und UserLCD2m sind zwei 16-Byte-Bereiche, welche direkt in einen nicht-flüchtigen Speicher (FLASH) abgelegt werden und beim Modulstart in die Register UserLCD1m bzw. UserLCD2m geladen werden. Alle Register sind nur als ganze 16-Byte-Blöcke lesbar.

LCD-Befehl	LCD-Befehl-Byte
UserLCD1 & UserLCD2	0
UserLCD1m & UserLCD2m	2

Beispiel: Lesen der Zeichenfolge EXDUL-393 aus Register

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	03	Befehlscode 3. Byte	03	Befehlscode 3. Byte
3	01	Längenbyte → 20 Byte	08	Längenbyte → 20 Byte
4	00 (UserLCD1&2) 02 (UserLCD1m&2m)	LCD-Befehl	45	Daten Zeile1 1. Zeichen E_{as}
5	00	reserviert	58	Daten Zeile1 2. Zeichen X_{asci}
6	00	reserviert	44	Daten Zeile1 3. Zeichen D_{asci}
7	01	Lesefunktion von LCD-Registern	55	Daten Zeile1 4. Zeichen U_{asci}
8			4C	Daten Zeile1 5. Zeichen L_{asci}
9			2D	Daten Zeile1 6. Zeichen $-_{asci}$
10			33	Daten Zeile1 7. Zeichen 3_{asci}
11			39	Daten Zeile1 8. Zeichen 9_{asci}
12			33	Daten Zeile1 9. Zeichen 3_{asci}
13			20	Daten Zeile1 10. Zeichen $[Leer]_{asci}$
14			20	Daten Zeile1 11. Zeichen $[Leer]_{asci}$
15			20	Daten Zeile1 12. Zeichen $[Leer]_{asci}$
16			20	Daten Zeile1 13. Zeichen $[Leer]_{asci}$
17			20	Daten Zeile1 14. Zeichen $[Leer]_{asci}$
18			20	Daten Zeile1 15. Zeichen $[Leer]_{asci}$
19			20	Daten Zeile1 16. Zeichen $[Leer]_{asci}$
20			45	Daten Zeile2 1. Zeichen E_{as}
21			58	Daten Zeile2 2. Zeichen X_{asci}
22			44	Daten Zeile2 3. Zeichen D_{asci}
23			55	Daten Zeile2 4. Zeichen U_{asci}
24			4C	Daten Zeile2 5. Zeichen L_{asci}
25			2D	Daten Zeile2 6. Zeichen $-_{asci}$
26			33	Daten Zeile2 7. Zeichen 3_{asci}
27			39	Daten Zeile2 8. Zeichen 9_{asci}
28			33	Daten Zeile2 9. Zeichen 3_{asci}
29			20	Daten Zeile2 10. Zeichen $[Leer]_{asci}$
30			20	Daten Zeile2 11. Zeichen $[Leer]_{asci}$
31			20	Daten Zeile2 12. Zeichen $[Leer]_{asci}$
32			20	Daten Zeile2 13. Zeichen $[Leer]_{asci}$
33			20	Daten Zeile2 14. Zeichen $[Leer]_{asci}$
34			20	Daten Zeile2 15. Zeichen $[Leer]_{asci}$
35			20	Daten Zeile2 16. Zeichen $[Leer]_{asci}$

Schreiben des LCD-Modes

Die LCD-Anzeige des EXDUL-Moduls stellt mehrere Anzeige-Modi bereit. Diese können mit folgendem Befehl eingestellt werden. Der LCD-Modus wird in einem nicht-flüchtigen Speicher abgelegt und wird auch nach einem Neustart des Moduls verwendet

LCD-Modus	LCD-Modus-Byte
IO-Mode	0
User-Mode	1

Beispiel: Schreiben des LCD-Modes

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	03	Befehlscode 3. Byte	03	Befehlscode 3. Byte
3	02	Längenbyte → 8 Byte	00	Längenbyte → 0 Byte
4	04	LCD-Befehl LCD-Mode		
5	00	reserviert		
6	00	reserviert		
7	00	Schreibfunktion		
8	00 (IO-Mode) 01 (User-Mode)	LCD-Modus		
9	00	reserviert		
10	00	reserviert		
11	00	reserviert		

Lesen des LCD-Modes

Die LCD-Anzeige des EXDUL-Moduls stellt mehrere Anzeige-Modi bereit. Der eingestellte LCD-Modus kann mit folgendem Befehl ausgelesen werden.

LCD-Modus	LCD-Modus-Byte
IO-Mode	0
User-Mode	1

Beispiel: Lesen des LCD-Modes

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	03	Befehlscode 3. Byte	03	Befehlscode 3. Byte
3	01	Längenbyte → 4 Byte	01	Längenbyte → 4 Byte
4	04	LCD-Befehl LCD-Mode	00 (IO-Mode) 01 (User-Mode)	LCD-Modus
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	01	Lesefunktion	00	reserviert

Schreiben LCD-Kontrastwert

Über diesen Befehl ist der Display-Kontrast einstellbar. Werte zwischen 0 und 4095 werden akzeptiert. Der Display-Kontrast verringert sich mit ansteigendem Wert. Eine angenehme Darstellung wird im Bereich 800 bis 1800 erreicht.

Beispiel: Schreiben Display-Kontrast-Wert 800

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	03	Befehlscode 3. Byte	03	Befehlscode 3. Byte
3	02	Längenbyte → 8 Byte	00	Längenbyte → 0 Byte
4	0B	LCD-Befehl LCD-Kontrast		
5	00	reserviert		
6	00	reserviert		
7	00	Schreibfunktion		
8	50	Kontrastwert (Lowbyte - 00...FF)		
9	03	Kontrastwert (Highbyte - 00...0F)		
10	00	reserviert		reserviert
11	00	reserviert		reserviert

Lesen LCD-Kontrastwert

Über diesen Befehl ist der Display-Kontrast auslesbar. Der Wert kann zwischen 0 und 4095 liegen. Der Display-Kontrast verringert sich mit ansteigendem Wert. Eine angenehme Darstellung wird im Bereich 800 bis 1800 erreicht.

Beispiel: Lesen Display-Kontrast-Wert 800

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0C	Befehlscode 1. Byte	0C	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	03	Befehlscode 3. Byte	03	Befehlscode 3. Byte
3	01	Längenbyte → 4 Byte	01	Längenbyte → 4 Byte
4	0B	LCD-Befehl LCD-Kontrast	50	Kontrastwert (Lowbyte - 00...FF)
5	00	reserviert	03	Kontrastwert (Highbyte - 00...0F)
6	00	reserviert	00	reserviert
7	01	Lesefunktion	00	reserviert

Optokopplerausgang lesen

Dieser Befehl ermöglicht das Auslesen des aktuellen Zustands des Optokopplerausgangs

Beispiel: Auslesen des Optokopplerausgangszustands
Gesendet wird ein 8Byte langer Block und empfangen ein 8Byte langer Block mit dem Zustand des Optokopplerausgangs

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	08	Befehlscode 1. Byte	08	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01 (→ 4Byte)	Längenbyte	01 (→ 4Byte)	Längenbyte
4	01	r/w Byte (1→ lesen)	0w 00 (LOW an DIN0) 01 (HIGH an DIN0)	Zustand Optokopplerausgang
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert

Optokopplerausgang schreiben

Dieser Befehl ermöglicht dem Anwender, den Ausgangsoptokoppler zu sperren oder durchzuschalten

Beispiel: Ausgabe eines Zustands am Optokopplerausgang
Gesendet wird ein 8Byte langer Block und empfangen ein 4Byte Block als Bestätigung

Byte	Senden	Empfangen	Beschreibung
0	08	08	Befehlscode 1. Byte
1	00	00	Befehlscode 2. Byte
2	00	00	Befehlscode 3. Byte
3	01 (→ 4Byte)	00	Längenbyte
4	00		r/w Byte
5	0w 00 (gesperrt) 01 (durchgeschaltet)		Optokopplerzustand
6	00		reserviert
7	00		reserviert

Optokopplereingang lesen

Dieser Befehl ermöglicht das Auslesen des aktuellen Zustands am Optokopplereingang

Beispiel: Auslesen des Zustands am Optokopplereingang
Gesendet wird ein 4Byte langer Block und empfangen ein 8Byte langer Block mit dem Zustand am Optokopplereingang

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	08	Befehlscode 1. Byte	08	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	01	Befehlscode 3. Byte	01	Befehlscode 3. Byte
3	00	Längenbyte	01 (→ 4Byte)	Längenbyte
4			0w	Zustand Optokopplereingang
5			00	reserviert
6			00	reserviert
7			00	reserviert

Zähler

Dieser Befehl ermöglicht den Zugriff auf den Zähler. So kann der Zähler gestartet, gestoppt, resetted und gelesen werden. Zudem besteht die Möglichkeit, das Overflow-Flag einzulesen und rückzusetzen.

Code	Zähler-Befehlscode
00	Zähler starten
01	Zähler stoppen
02	Zähler resetten
03	Zählerstand lesen
04	reserviert
05	Overflow-Flag lesen
06	Overflow-Flag rücksetzen

Zähler Start / Stop / Reset

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	09	Befehlscode 1. Byte	09	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte	01	Längenbyte
4	bb 00 01 02	Zähler Befehlscode Zähler0 starten Zähler0 stoppen Zähler0 resetten	bb	Zähler Befehlscode
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert

Zähler lesen

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	09	Befehlscode 1. Byte	09	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte	02 (→ 8Byte)	Längenbyte
4	03	Zähler Befehlscode	03	Zähler Befehlscode
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert
8			ww	Zählerstand Byte0
9			ww	Zählerstand Byte1
10			ww	Zählerstand Byte2
11			ww	Zählerstand Byte3

Zählerstand = Zählerstand Byte3 * 0x1000000 + Zählerstand Byte2 * 0x10000 + Zählerstand Byte1 * 0x100 + Zählerstand Byte0

Overflow-Flag lesen

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	09	Befehlscode 1. Byte	09	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte	02 (→ 8Byte)	Längenbyte
4	05	Zähler Befehlscode Overflow-Flag lesen	05	Zähler Befehlscode Overflow-Flag lesen
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	0f	Overflow-Flag

Overflow-Flag rücksetzen

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	09	Befehlscode 1. Byte	09	Befehlscode 1. Byte
1	00	Befehlscode 2. Byte	00	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte	01 (→ 4Byte)	Längenbyte
4	06	Zähler Befehlscode Overflow-Flag rücksetzen	06	Zähler Befehlscode Overflow-Flag rücksetzen
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert

Temperaturmesseinheit Temperatur-/Widerstandsmessung

Dieser Befehl ermöglicht das Messen des Temperatursensors mit der gewünschten Temperaturmesseinheit. Dabei kann der gemessene Wert als Temperatur in °C * 100 oder als Widerstand (nur bei PT100 Konfiguration) übergeben werden.

Beispiel: Temperaturmessung an Messeinheit TIN1
Gesendet wird ein 8Byte langer Block und empfangen ein 12Byte langer Block mit der Temperatur.

Kanal:

Kanal	Messeinheitsindex
TIN0	0
TIN1	1
TIN2	2
TIN3	3
TIN4	4
TIN5	5

Messbereich:

Messart	Messart-Byte
Widerstandsmessung	0
Temperaturmessung	1

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0A	Befehlscode 1. Byte	0A	Befehlscode 1. Byte
1	04	Befehlscode 2. Byte	04	Befehlscode 2. Byte
2	00	Befehlscode 3. Byte	00	Befehlscode 3. Byte
3	01	Längenbyte	02 (→ 8Byte)	Längenbyte
4	01	Messeinheitsindex	01	Messeinheitsindex
5	01	Messart-Byte (=Tempera-turmessung)	01	Messart-Byte (=Temperatur-messung)
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert
8			temp0	Temperatur Byte0
9			temp1	Temperatur Byte1
10			temp2	Temperatur Byte2
11			temp3	Temperatur Byte3

Temperatur = (integer)(temp3*0x1000000 + temp2*0x10000 + temp1*0x100 + temp0) [°C *100]
Widerstand = (integer)(Byte3*0x1000000 + Byte2 * 0x10000 + Byte1 * 0x100 + Byte0) [mΩ]

Temperaturmesseinheit Fehlerüberprüfung

Dieser Befehl führt einen Fehlertest bei der gewünschten Messeinheit durch.

Fehler-Bit	mögliche Fehlerursache wenn gesetzt	Fehlerbeschreibung
D7	reserviert	
D6	reserviert	
D5	Fehler bei der Verdrahtung	
D4	Fehler bei der Verdrahtung	
D3	Fehler bei der Verdrahtung	
D2	Overvoltage oder Undervoltage	evtl. externe Spannung eingespeist
D1	reserviert	
D0	reserviert	

Beispiel: Fehlerüberprüfung an Messeinheit TIN1
Gesendet wird ein 8Byte langer Block und empfangen ein 12Byte langer Block mit dem Fehlerbyte.

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0A	Befehlscode 1. Byte	0A	Befehlscode 1. Byte
1	04	Befehlscode 2. Byte	04	Befehlscode 2. Byte
2	01	Befehlscode 3. Byte	01	Befehlscode 3. Byte
3	01	Längenbyte	02 (→ 8Byte)	Längenbyte
4	01	Messeinheitsindex	00	Messeinheitsindex
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert
8			ee	Error-Byte
9			00	reserviert
10			00	reserviert
11			00	reserviert

Temperaturmesseinheit Sensortyp einstellen

Mit diesem Befehl kann der Sensortyp einer jeden Messeinheit bestimmt werden. Zusätzlich muss der entsprechende Jumper gesetzt werden. Im Werkszustand ist der PT100 ausgewählt.

Typbyte	mögliche Fehlerursache wenn gesetzt
0x00 (default)	PT100
0x01	PT1000

Beispiel: Sensortyp PT1000 an Messeinheit TIN1 einstellen
Gesendet wird ein 8Byte langer Block und empfangen ein 8Byte langer Block mit der Temperatur.

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0A	Befehlscode 1. Byte	0A	Befehlscode 1. Byte
1	04	Befehlscode 2. Byte	04	Befehlscode 2. Byte
2	08	Befehlscode 3. Byte	08	Befehlscode 3. Byte
3	01	Längenbyte	01 (→ 4Byte)	Längenbyte
4	01	Messeinheitsindex	00	Messeinheitsindex
5	00	reserviert	00	reserviert
6	01	Typbyte	00	reserviert
7	00	reserviert	00	reserviert

Temperaturmesseinheit kalibrieren/abgleichen

Mit diesem Befehl kann die Kalibrierung bzw. der Abgleich der Messeinheit gestartet werden. Zuvor muss sowohl in der Software als auch an der Hardware der gewünschte Sensortyp ausgewählt werden.

Beispiel: Kalibrierung der Messeinheit TIN1
Gesendet wird ein 8Byte langer Block und empfangen ein 8Byte langer Block mit der Temperatur.

Byte	Senden	Beschreibung	Empfangen	Beschreibung
0	0A	Befehlscode 1. Byte	0A	Befehlscode 1. Byte
1	FF	Befehlscode 2. Byte	FF	Befehlscode 2. Byte
2	F7	Befehlscode 3. Byte	F7	Befehlscode 3. Byte
3	01	Längenbyte	01 (→ 4Byte)	Längenbyte
4	01	Messeinheitsindex	01	Messeinheitsindex
5	00	reserviert	00	reserviert
6	00	reserviert	00	reserviert
7	00	reserviert	00	reserviert

11.5 Modulzugriff über LabVIEW und EXDUL.dll

Dank der EXDUL.dll kann das Modul ohne großen Aufwand in ein LabVIEW-Projekt eingebunden werden. Neben LabVIEW und der EXDUL.dll-Datei wird zudem auf dem Rechner das .NET-Framework benötigt.
Für genauere Informationen lesen Sie sich bitte das EXDUL-LabVIEW-Tutorial durch.

12. Programmierung unter Linux®

12.1 Einführung

Nach dem erfolgreichen Erkennen des EXDUL-393E / EXDUL-393S durch das Betriebssystem wird das Modul im Ordner /dev als ttyACM* Gerät gelistet. Es handelt sich hierbei um ein CDC-Device (Communications Device Class), das über einen virtuellen COM-Port angesprochen wird.
Der Softwarezugriff auf diesen virtuellen COM-Port erfolgt wie über eine normale COM-Schnittstelle über einen Standard-Treiber, eine Installation eines zusätzlichen Treibers ist nicht notwendig.

12.2 Programmierung mit seriellen COM-Port-Libraries

Wurde das Modul erkannt, kann über die Standard Libraries für die serielle Schnittstellen mit ihm kommuniziert werden. Für genauere Informationen lesen Sie ab Kapitel 11.4 weiter.

13. Technische Daten

Digitaler Eingang über Optokoppler
1 bipolarer Kanal mit galvanischer Trennung
Überspannungsschutz-Dioden
Eingangsspannungsbereich
high = 10..30 Volt
low = 0..3 Volt

Digitaler Ausgang über Optokoppler
1 Kanal mit galvanischer Trennung über Leistungsoptokoppler
Verpolungsschutz-Dioden
Spannung-CE: max. 50 V
Ausgangsstrom: max. 150mA

Zähler
1 programmierbarer Zähler 32 Bit (belegt den Optokopplereingang)
Zählfrequenz: max. 5 kHz
Automatische Sicherung des Zählerstandes im 10kHz Takt

6 Temperaturmesseinheiten
Sensortyp PT100 und PT1000 je Einheit idividuell über Jumper wählbar
3-Leiteranschluss

LCD Anzeige (nur EXDUL-393E)
Matrixanzeige mit 2 Zeilen und 16 Spalten zur Darstellung von 16 Zeichen je Zeile
Programmierbar zur Darstellung anwendungsspezifischer Daten oder als I/O-Zustandsanzeige

USB-Schnittstelle
USB 2.0 kompatibel
USB-Anschluss Plug&Play (hotpluggable, auch im laufenden Betrieb anschließbar)

Modul-Anschlüsse
1 * 24polige Schraubklemmleiste
1 * USB-Buchse Typ B

Stromversorgung
USB
Strom: typ. 160mA

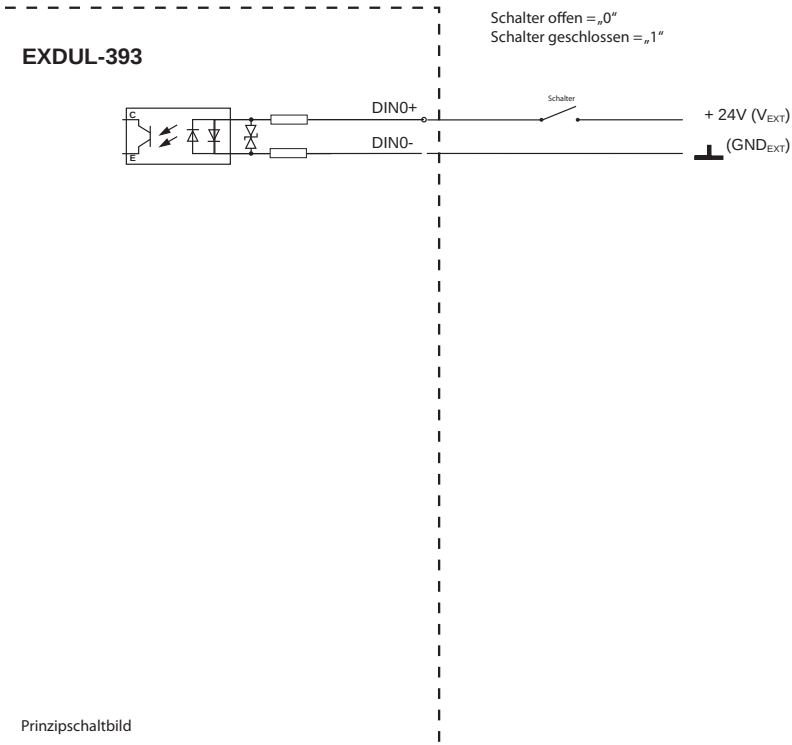
extern
Spannung: 10 .. 30V
Strom: typ. 60mA bei 24V

Abmessungen
105 mm x 89 mm x 59 mm (l x b x h)

Gehäuse
Isolierstoffgehäuse mit integrierter Schnapptechnik zur DIN EN Hutschienen-Montage
Geeignet für Aufbaumontagen, Schaltschrank- und Verteilereinbau sowie für mobile Tischeinsätze

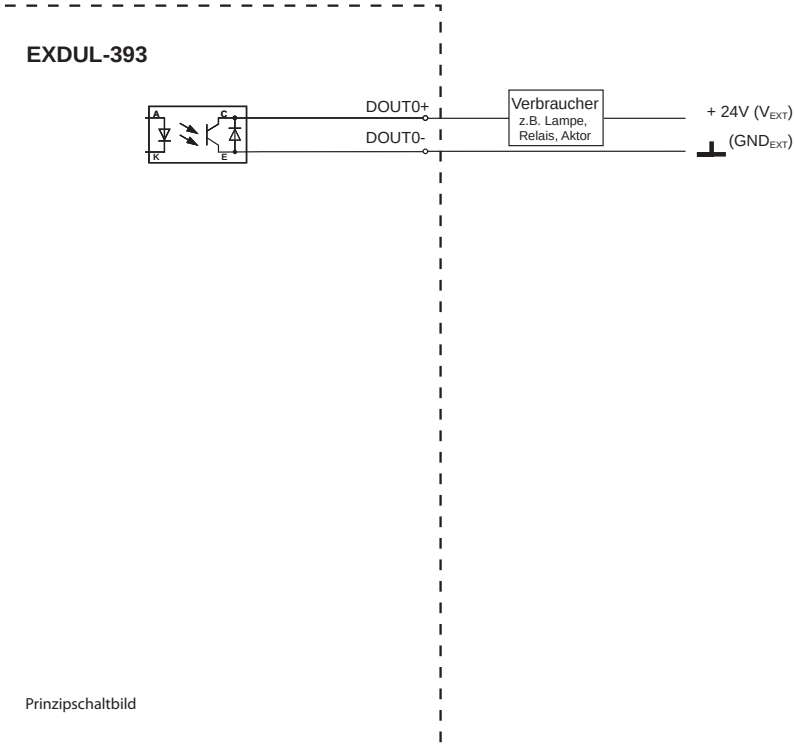
14. Beschaltungsbeispiele

14.1 Beschaltung des Optokoppler-Eingangs



Grafik 14.1 Beschaltung des Optokopplereingangs

14.2 Beschaltung des Optokoppler-Ausgangs



Grafik 14.2 Beschaltung des Optokoppler-Ausgangs

15. ASCII-Tabelle

Hex	Dez	Binär	Zeichen
00	0	00000000	
01	1	00000001	
02	2	00000010	
03	3	00000011	
04	4	00000100	
05	5	00000101	
06	6	00000110	
07	7	00000111	
08	8	00001000	
09	9	00001001	
0A	10	00001010	
0B	11	00001011	
0C	12	00001100	
0D	13	00001101	
0E	14	00001110	
0F	15	00001111	
10	16	00010000	
11	17	00010001	
12	18	00010010	
13	19	00010011	
14	20	00010100	
15	21	00010101	
16	22	00010110	
17	23	00010111	
18	24	00011000	
19	25	00011001	
1A	26	00011010	
1B	27	00011011	
1C	28	00011100	
1D	29	00011101	
1E	30	00011110	
1F	31	00011111	
20	32	00100000	[Leer]
21	33	00100001	!
22	34	00100010	”
23	35	00100011	#
24	36	00100100	$
25	37	00100101	%
26	38	00100110	&
27	39	00100111	’

Hex	Dez	Binär	Zeichen
28	40	00101000	(
29	41	00101001	)
2A	42	00101010	*
2B	43	00101011	+
2C	44	00101100	,
2D	45	00101101	-
2E	46	00101110	.
2F	47	00101111	/
30	48	00110000	0
31	49	00110001	1
32	50	00110010	2
33	51	00110011	3
34	52	00110100	4
35	53	00110101	5
36	54	00110110	6
37	55	00110111	7
38	56	00111000	8
39	57	00111001	9
3A	58	00111010	:
3B	59	00111011	;
3C	60	00111100	<
3D	61	00111101	=
3E	62	00111110	>
3F	63	00111111	?
40	64	01000000	@
41	65	01000001	A
42	66	01000010	B
43	67	01000011	C
44	68	01000100	D
45	69	01000101	E
46	70	01000110	F
47	71	01000111	G
48	72	01001000	H
49	73	01001001	I
4A	74	01001010	J
4B	75	01001011	K
4C	76	01001100	L
4D	77	01001101	M
4E	78	01001110	N
4F	79	01001111	O

Hex	Dez	Binär	Zeichen
50	80	01010000	P
51	81	01010001	Q
52	82	01010010	R
53	83	01010011	S
54	84	01010100	T
55	85	01010101	U
56	86	01010110	V
57	87	01010111	W
58	88	01011000	X
59	89	01011001	Y
5A	90	01011010	Z
5B	91	01011011	[
5C	92	01011100	
5D	93	01011101	]
5E	94	01011110	^
5F	95	01011111	_
60	96	01100000	`
61	97	01100001	a
62	98	01100010	b
63	99	01100011	c
64	100	01100100	d
65	101	01100101	e
66	102	01100110	f
67	103	01100111	g
68	104	01101000	h
69	105	01101001	i
6A	106	01101010	j
6B	107	01101011	k
6C	108	01101100	l
6D	109	01101101	m
6E	110	01101110	n
6F	111	01101111	o
70	112	01110000	p
71	113	01110001	q
72	114	01110010	r
73	115	01110011	s
74	116	01110100	t
75	117	01110101	u
76	118	01110110	v
77	119	01110111	w
78	120	01111000	x
79	121	01111001	y
7A	122	01111010	z
7B	123	01111011	{

Hex	Dez	Binär	Zeichen
7C	124	01111100	|
7D	125	01111101	}
7E	126	01111110	
7F	127	01111111	
80	128	10000000	
81	129	10000001	
82	130	10000010	
83	131	10000011	
84	132	10000100	
85	133	10000101	
86	134	10000110	
87	135	10000111	
88	136	10001000	
89	137	10001001	
8A	138	10001010	
8B	139	10001011	
8C	140	10001100	
8D	141	10001101	
8E	142	10001110	
8F	143	10001111	
90	144	10010000	
91	145	10010001	
92	146	10010010	
93	147	10010011	
94	148	10010100	
95	149	10010101	
96	150	10010110	
97	151	10010111	
98	152	10011000	
99	153	10011001	
9A	154	10011010	
9B	155	10011011	
9C	156	10011100	
9D	157	10011101	
9E	158	10011110	
9F	159	10011111	
A0	160	10100000	
A1	161	10100001	
A2	162	10100010	
A3	163	10100011	
A4	164	10100100	
A5	165	10100101	
A6	166	10100110	
A7	167	10100111	

Hex	Dez	Binär	Zeichen
A8	168	10101000	
A9	169	10101001	
AA	170	10101010	
AB	171	10101011	
AC	172	10101100	
AD	173	10101101	
AE	174	10101110	
AF	175	10101111	
B0	176	10110000	
B1	177	10110001	
B2	178	10110010	
B3	179	10110011	
B4	180	10110100	
B5	181	10110101	
B6	182	10110110	
B7	183	10110111	
B8	184	10111000	
B9	185	10111001	
BA	186	10111010	
BB	187	10111011	
BC	188	10111100	
BD	189	10111101	
BE	190	10111110	
BF	191	10111111	
C0	192	11000000	
C1	193	11000001	
C2	194	11000010	
C3	195	11000011	
C4	196	11000100	
C5	197	11000101	
C6	198	11000110	
C7	199	11000111	
C8	200	11001000	
C9	201	11001001	
CA	202	11001010	
CB	203	11001011	
CC	204	11001100	
CD	205	11001101	
CE	206	11001110	
CF	207	11001111	
D0	208	11010000	
D1	209	11010001	
D2	210	11010010	
D3	211	11010011	

Hex	Dez	Binär	Zeichen
D4	212	11010100	
D5	213	11010101	
D6	214	11010110	
D7	215	11010111	
D8	216	11011000	
D9	217	11011001	
DA	218	11011010	
DB	219	11011011	
DC	220	11011100	
DD	221	11011101	
DE	222	11011110	
DF	223	11011111	
E0	224	11100000	
E1	225	11100001	
E2	226	11100010	
E3	227	11100011	
E4	228	11100100	
E5	229	11100101	
E6	230	11100110	
E7	231	11100111	
E8	232	11101000	
E9	233	11101001	
EA	234	11101010	
EB	235	11101011	
EC	236	11101100	
ED	237	11101101	
EE	238	11101110	
EF	239	11101111	
F0	240	11110000	
F1	241	11110001	
F2	242	11110010	
F3	243	11110011	
F4	244	11110100	
F5	245	11110101	
F6	246	11110110	
F7	247	11110111	
F8	248	11111000	
F9	249	11111001	
FA	250	11111010	
FB	251	11111011	
FC	252	11111100	
FD	253	11111101	
FE	254	11111110	
FF	255	11111111	

16. Produkthaftungsgesetz

Hinweise zur Produkthaftung

Das Produkthaftungsgesetz (ProdHaftG) regelt die Haftung des Herstellers für Schäden, die durch Fehler eines Produktes verursacht werden.

Die Verpflichtung zu Schadenersatz kann schon gegeben sein, wenn ein Produkt aufgrund der Form der Darbietung bei einem nichtgewerblichen Endverbraucher eine tatsächlich nicht vorhandene Vorstellung über die Sicherheit des Produktes erweckt, aber auch wenn damit zu rechnen ist, dass der Endverbraucher nicht die erforderlichen Vorschriften über die Sicherheit beachtet, die beim Umgang mit diesem Produkt einzuhalten wären.

Es muss daher stets nachweisbar sein, dass der nichtgewerbliche Endverbraucher mit den Sicherheitsregeln vertraut gemacht wurde.

Bitte weisen Sie daher im Interesse der Sicherheit Ihre nichtgewerblichen Abnehmer stets auf Folgendes hin:

Sicherheitsvorschriften

Beim Umgang mit Produkten, die mit elektrischer Spannung in Berührung kommen, müssen die gültigen VDE-Vorschriften beachtet werden.

Besonders sei auf folgende Vorschriften hingewiesen:
VDE0100; VDE0550/0551; VDE0700; VDE0711; VDE0860.
Sie erhalten VDE-Vorschriften beim vde-Verlag GmbH, Bismarckstraße 33, 10625 Berlin.

* Vor Öffnen eines Gerätes den Netzstecker ziehen oder sicherstellen, dass das Gerät stromlos ist.

* Bauteile, Baugruppen oder Geräte dürfen nur in Betrieb genommen werden, wenn sie vorher in ein berührungssicheres Gehäuse eingebaut wurden. Während des Einbaus müssen sie stromlos sein.

* Werkzeuge dürfen an Geräten, Bauteilen oder Baugruppen nur benutzt werden, wenn sichergestellt ist, dass die Geräte von der Versorgungsspannung getrennt sind und elektrische Ladungen, die in im Gerät befindlichen Bauteilen gespeichert sind, vorher entladen wurden.

* Spannungsführende Kabel oder Leitungen, mit denen das Gerät, das Bauteil oder die Baugruppe verbunden sind, müssen stets auf Isolationsfehler oder Bruchstellen untersucht werden. Bei Feststellen eines Fehlers in der Zuleitung muss das Gerät unverzüglich aus dem Betrieb genommen werden, bis die defekte Leitung ausgewechselt worden ist.

* Bei Einsatz von Bauelementen oder Baugruppen muss stets auf die strikte Einhaltung der in der zugehörigen Beschreibung genannten Kenndaten für elektrische Größen hingewiesen werden.

* Wenn aus den vorgelegten Beschreibungen für den nichtgewerblichen Endverbraucher nicht eindeutig hervorgeht, welche elektrischen Kennwerte für ein Bauteil gelten, so muss stets ein Fachmann um Auskunft ersucht werden.

Im Übrigen unterliegt die Einhaltung von Bau- und Sicherheitsvorschriften aller Art (VDE, TÜV, Berufsgenossenschaften usw.) dem Anwender/Käufer.

17. EG-Konformitätserklärung

Für die Erzeugnisse

EXDUL-393E EDV-Nummer A-382320
EXDUL-393S EDV-Nummer A-382310

wird hiermit bestätigt, dass sie den Anforderungen der betreffenden EG-Richtlinien entsprechen. Bei Nichteinhaltung der im Handbuch angegebenen Vorschriften zum bestimmungsgemäßen Betrieb der Produkte verliert diese Erklärung Ihre Gültigkeit.

EN 5502 Klasse B
IEC 801-2
IEC 801-3
IEC 801-4
EN 50082-1
EN 60555-2
EN 60555-3

Diese Erklärung wird verantwortlich für den Hersteller

Messcomp Datentechnik GmbH
Neudecker Str. 11
83512 Wasserburg

abgegeben durch

Dipl.Ing.(FH) Hans Schnellhammer

Wasserburg, 31.01.2019

Referenzsystem-Bestimmungsgemäßer Betrieb

Die Multifunktionsmodule EXDUL-393E und EXDUL-393S sind nichtselbständig betreibbare Geräte, dessen CE-Konformität nur bei gleichzeitiger Verwendung von zusätzlichen Computerkomponenten beurteilt werden kann. Die Angaben zur CE-Konformität beziehen sich deshalb ausschließlich auf den bestimmungsgemäßen Einsatz der Multifunktionsmodule in folgendem Referenzsystem:

Schaltschrank:	Vero IMRAK 3400	804-530061C 802-563424J 802-561589J
19" Gehäuse:	Vero PC-Gehäuse	145-010108L
19" Gehäuse:	Zusatzelektronik	519-112111C
Motherboard:	GA-586HX	PIV 1.55
Floppy-Controller:	auf Motherboard	
Floppy:	TEAC	FD-235HF
Grafikkarte:	Advantech	PCA-6443
Schnittstellen:	EXDUL-393E EXDUL-393S	A-382320 A-382310