Digitale Ein- und Ausgänge mit einer RS-232-Schnittstelle

Version 1-00

Bedienungsanleitung

Dokument-Version A, erstellt am 26.07.2009

CGC Instruments
Hübschmannstr. 18 | D–09112 Chemnitz
Tel.: +49 (371) 355 098–55
Fax: +49 (371) 355 098–60
Internet: www.cgc-instruments.com
E–Mail: info@cgc-instruments.com

Inhalt

Technische Daten

Charakteristik

- fünf digitale Eingänge, fünf digitale Ausgänge
- RS-232-Schnittstelle
- Stromversorgung über die RS-232-Schnittstelle oder aus einem externen Netzteil
- Metallgehäuse

Schnittstelle

- Serielle Schnittstelle nach dem RS-232-Standard
- Anschluss: D-SUB-Stecker, 9 Pins
- Datenübertragungsrate: 9600 kBaud
- Schnittstellenparameter: 8 Datenbits, 1 Stopbit, gerade Parität

digitale Ein- und Ausgänge

- Anschluss: 25polige D-SUB-Buchse
- Nennspannung der Eingänge: 24 V=
- Stromverbrauch der Eingänge: 5,3 mA bei 24 V=
- Schaltleistung der Ausgänge: 150 V= / 125 V~, 1 A, 30 W / 60 VA

Externes Netzteil

- Nennspannung: 12 bis 24 V=
- Anschluss: Buchse für Hohlstecker mit Stift ø2,1 mm,
  Polarität: Innen +, Außen –
- Stromverbrauch:
  1 mA typ. mit deaktivierten Ausgängen,
  27 mA typ. mit aktivierten Ausgängen
- Verpolungs- und Überspannungsschutz

Allgemein

- Metallgehäuse:
  Material: Aluminium, naturfarbig eloxiert
  Abmessungen: 84 x 69 x 24 mm^3
  (Länge × Breite × Höhe ohne Stecker)
- Gewicht: ca. 100 g

Lieferungsumfang

- komplett getestetes Modul COM-DIO5+5
- Diagnosesoftware
- Bedienungsanleitung
- optional:
    - RS-232-Anschlusskabel 2x D-SUB-Buchse, 9 Pin
    - externes Netzteil 12 V=, 500 mA
    - RS-232-Schnittstellenkarte mit 1 bis 8 Schnittsstellen

CGC Instruments
Hübschmannstr. 18 | D–09112 Chemnitz
Tel.: +49 (371) 355 098–55
Fax: +49 (371) 355 098–60
Internet: www.cgc-instruments.com
E–Mail: info@cgc-instruments.com

Anschlüsse und Konfigurationselemente

Das Modul verfügt über einen D-SUB-Stecker mit 9 Pins (CN2 in Abb. 1) zum Anschluss an einen PC mit einer seriellen Schnittstelle nach dem RS-232-Standard. Die Schnittstelle dient der Datenübertragung vom/zum Modul und der Stromversorgung des Moduls, die Datenübertragungsrate beträgt 9600 kBaud.

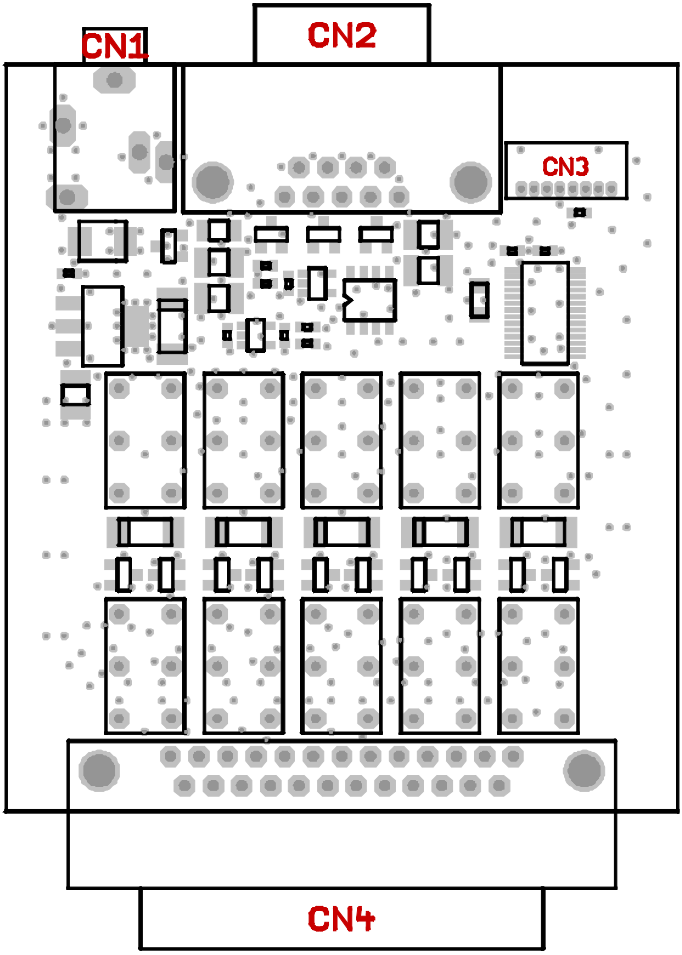

Abb. 1. Anschlüsse des Moduls.
CN1: Klinkenstecker-Buchse für das externe Netzteil, CN2: D-SUB-Stecker für die RS-232-Schnittstelle, CN4: D-SUB-Buchse für die digitalen Ein- und Ausgänge.

Zur Versorgung des Moduls wird an die Klinkenstecker-Buchse CN1 in Abb. 1 ein externes Netzteil mit der Gleichspannung von 12 bis 24 V angeschlossen. Alternativ dazu kann die Betriebsspannung über die RS-232-Schnittstelle bereitgestellt werden. Dazu sind spezielle, nicht standardisierte Schnittstellenkarten erforderlich, welche auf dem Pin 9 eine Versorgungsspannung von 12 V zur Verfügung stellen (siehe Tab. 2).

Die Stromversorgung des Moduls kann auch über die standardisierte RS-232-Schnittstelle erfolgen, aber nur dann, wenn die Ausgänge nicht verwendet werden.

Das Modul nutzt in jeder Konfiguration die Steuerleitungen der seriellen Schnittstelle, um Versorgungsspannungen von etwa ±12 V für den Treiber der seriellen Schnittstelle und den internen Mikrokontroller zu erhalten. Das Modul soll daher mit einem vollbelegten 9poligen Kabel an die RS-232-Schnittstelle des Steuerrechners angeschlossen werden (für mehrere Details siehe Abschnitt "RS-232-Datenkabel" sowie Tab. 2).

! **Achtung:** Wird das externe Netzteil nicht angeschlossen oder stellt die RS-232-Schnittstelle keine Stromversorgung von 12 V zur Verfügung, funktionieren die digitalen Ausgänge nicht. Andere Funktionen des Moduls werden dadurch jedoch nicht beeinträchtigt.

Die digitalen Ein- und Ausgänge sind zusammen mit der Gerätemasse an die D-SUB-Buchse CN4 angeschlossen (siehe Tab. 1). Die fünf digitalen Ausgänge sind als galvanisch getrennte Schließkontakte ausgeführt. Die fünf digitalen Eingänge sind ebenfalls vom Gerät galvanisch getrennt und benötigen zur Aktivierung das Anlegen einer externen Spannung von 24 V der in Tabelle 1 angedeuteten Polarität.

Inbetriebnahme

Das Modul COM-DIO5+5 ist ein Schnittstellen-Konverter, der den Zustand digitaler Ein- und Ausgänge über eine standardisierte RS-232-Schnittstelle übertragen kann. Um die Kommunikation mit dem Modul zu ermöglichen, muss das Modul durch ein entsprechendes Datenkabel mit einer seriellen Schnittstelle eines Steuerrechners (eines PCs) verbunden werden. Eine zusätzliche Installation von Treibern ist nicht erforderlich.

Um die Funktion des Moduls zu überprüfen, gehen Sie wie folgt vor:

- Schließen Sie das Modul mit Hilfe eines RS-232-Kabels an eine serielle Schnittstelle des PCs an. Besitzt der PC keine (freie) serielle Schnittstelle, muss eine Schnittstellenkarte installiert werden (siehe Abschnitt "Installation der RS-232-Schnittstellenkarte").
- Öffnen Sie ein Kommandozeilenfenster, wechseln Sie in das Verzeichnis mit dem Diagnoseprogramm `COM-DIO-Test.exe` (Verzeichnis "Program" des beiliegenden Softwarepakets) und starten Sie es. Das Programm zeigt beim Aufrufen die verfügbaren Optionen. Für eine erfolgreiche Ausführung muss mindestens die Nummer der seriellen Schnittstelle angegeben werden, an die das Modul angeschlossen ist. Ist das Modul beispielsweise an die Schnittstelle COM5 angeschlossen, muss das Diagnoseprogramm durch die folgende Anweisung gestartet werden: "`COM-DIO-Test 5`". Wird die angegebene Schnittstelle nicht gefunden, endet das Programm mit einer Fehlermeldung. In einem solchen Fall muss im Gerätemanager von Windows überprüft werden, ob die Schnittstelle existiert und ob sie funktionsfähig ist. Die Konfiguration der Schnittstelle ist dabei unerheblich, denn sie wird durch das Diagnoseprogramm selbst eingestellt.
- Konnte das Diagnoseprogramm erfolgreich gestartet werden, drücken Sie die Taste 'p' auf der Tastatur des PCs. Das Programm liest aus dem angeschlossenen Modul die Produktbezeichnung aus. Sollte anstelle davon ein Fehler gemeldet werden, wurde die Datenübertragung gestört. Wenn die Übertragung auch bei einer Wiederholung nicht korrekt funktioniert, liegt ein grundlegender Fehler vor. Überprüfen Sie das Anschlusskabel und die Funktion der benutzten seriellen Schnittstelle des PCs.
- Funktioniert alles fehlerfrei, können Sie durch Drücken der Tasten 'n', 'd' und 'v' weitere Produktinformationen abfragen.

- Drücken Sie die Taste 'o' und geben Sie einen neuen Wert für die Ausgänge an. Sie können dekadische, durch 'h' endende hexadezimale (beispielsweise `1Ah`) oder durch 'b' endende binäre (beispielsweise `10011b`) Zahlen eingeben. Sie können die Ausgänge einzeln durchtesten, indem Sie Werte von `00001b` bis `10000b` eingeben. Die Schaltgeräusche der Ausgangsrelais kann man dabei gut hören, so dass man auf die Funktion der Ausgänge auch ohne einen externen Anschluss schließen kann.
- Drücken Sie die Taste 'I', dadurch wird die automatische Datenübermittlung vom Modul eingeschaltet. Aktivieren Sie die Eingänge durch Anschließen einer externen Spannung von 24 V. Eine Änderung des Zustandes der digitalen Eingänge wird durch das Diagnoseprogramm über die Anzeige der aktuell eingelesenen Werte signalisiert.

Digitale Ein- und Ausgänge

Die digitalen Ein- und Ausgänge sind zusammen mit der Gerätemasse an die D-SUB-Buchse CN4 angeschlossen. Die Tabelle 1 zeigt die Pinbelegung der Buchse.

Abb. 2 zeigt die Beschaltung der Ein- und Ausgänge. Die Ausgänge sind an die Kontakte (Schließer) des jeweiligen Relais angeschlossen, das durch das Modul angesteuert wird. Wird ein Kanal aktiviert, schalten die Kontakte. Im Ruhezustand oder ohne externe Stromversorgung sind die Kontakte geöffnet.

Abb. 2. Beschaltung der Eingänge (links) und der Ausgänge (rechts).

Die Eingangsanschlüsse steuern direkt die Spule des jeweiligen Relais. Das Modul ermittelt den Schaltzustand anhand der Kontakte des Relais. Die Eingänge werden durch das Anlegen der entsprechenden Steuerspannung aktiviert, im stromlosen Zustand sind die Eingänge inaktiv. Bei der Beschaltung muss die Polarität der Steuerspannung beachtet werden.

Da beim Ausschalten der Eingänge durch die in der Induktivität der Relais-Spule akkumulierte Energie hohe Spannungen entstehen können (Selbstinduktion), wurden die Eingänge mit einem aus jeweils einer Diode (siehe Abb. 2) bestehenden Überspannungsschutz ausgestattet. Dadurch werden die angeschlossenen Schaltkreise ausreichend geschützt, weitere Schutzmaßnahmen sind auch bei der Verwendung von Halbleiter-Schaltern (beispielsweise von Näherungssensoren) nicht erforderlich.

! **Vorsicht:** Wird eine Steuerspannung falscher Polarität oder eine Steuerspannung höher als die nominale angelegt, kann dies zu einer dauerhaften Beschädigung des jeweiligen Eingangs oder sogar des gesamten Gerätes führen.

Tab. 1. Pinbelegung der Datenbuchse CN4

Pin	Bezeichnung	Funktion
1	Gnd	Gerätemasse
2	Gnd	Gerätemasse
3	In0+	digitaler Eingang 0, positiv
4	Out0a	digitaler Ausgang 0, Anschluss a
5	In1+	digitaler Eingang 1, positiv
6	Out1a	digitaler Ausgang 1, Anschluss a
7	In2+	digitaler Eingang 2, positiv
8	Out2a	digitaler Ausgang 2, Anschluss a
9	In3+	digitaler Eingang 3, positiv
10	Out3a	digitaler Ausgang 3, Anschluss a
11	In4+	digitaler Eingang 4, positiv
12	Out4a	digitaler Ausgang 4, Anschluss a
13	Gnd	Gerätemasse
14	Gnd	Gerätemasse
15	In0–	digitaler Eingang 0, negativ
16	Out0b	digitaler Ausgang 0, Anschluss b
17	In1–	digitaler Eingang 1, negativ
18	Out1b	digitaler Ausgang 1, Anschluss b
19	In2–	digitaler Eingang 2, negativ
20	Out2b	digitaler Ausgang 2, Anschluss b
21	In3–	digitaler Eingang 3, negativ
22	Out3b	digitaler Ausgang 3, Anschluss b
23	In4–	digitaler Eingang 4, negativ
24	Out4b	digitaler Ausgang 4, Anschluss b
25	Gnd	Gerätemasse

RS-232-Datenkabel

Das Datenkabel verbindet das Modul mit einer seriellen Schnittstelle eines Steuerrechners (eines PCs). Es dient der Datenübertragung vom/zum Modul und der Stromversorgung des Moduls.

Das Datenkabel soll ein 9adriges 1:1-Kabel mit zwei D-SUB-Buchsen, für die Kommunikation reicht jedoch unter Umständen ein einfacheres Kabel. Die folgende Tabelle zeigt die Leitungen, welche für die Funktion des Moduls erforderlich sind. Die Leitungen verbinden gleichnamige Pins der beiden D-SUB-Buchsen.

Tab. 2. Pinbelegung des Datenkabels (Stecker CN2)

Pin	Bezeichnung	Funktion
1	DCD	Eingang, nicht benutzt
2	RxD	Datenleitung Modul → Steuerrechner
3	TxD	Datenleitung Modul ← Steuerrechner
4	DTR	Ausgang, benutzt zur Stromversorgung des Moduls
5	GND	Masse, Abschirmung der Leitungen
6	DSR	Eingang, intern verbunden mit DTR
7	RTS	Ausgang, benutzt zur Stromversorgung des Moduls
8	CTS	Eingang, intern verbunden mit RTS
9	RI	Stromversorgung +12V, optional

Bemerkungen:

1. Das Datenkabel soll für eine höhere Kommunikationssicherheit abgeschirmt sein. Die Abschirmung soll mit der Masse (Pin 5) verbunden sein.
2. Die Eingänge DCD (Pin 1), DSR (Pin 6) und CTS (Pin 8) sind für die Funktion des Moduls nicht erforderlich.
3. Die Leitung RI (Pin 9) ist nur notwendig, wenn das Modul über die Schnittstellenkarte versorgt wird.

Schaltbeispiele

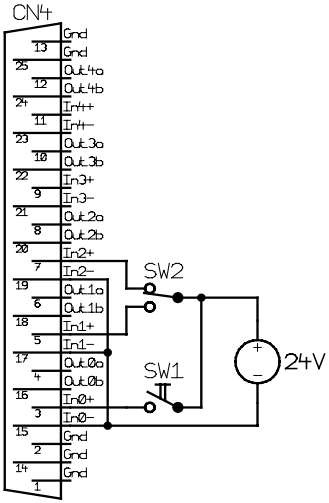

Abb. 3. Anschluss eines Schalters oder Tasters (SW1) und eines Umschalters (SW2) an die Eingänge.

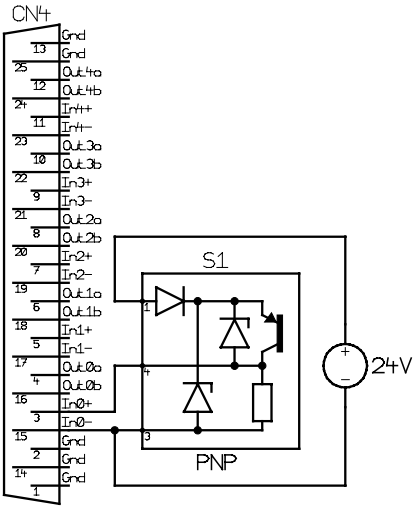

Abb. 4. Anschluss eines photoelektrischen Näherungsschalters an die Eingänge. Der Schalter muss ein PNP-Typ sein, er wird aus dem externen Netzteil versorgt.

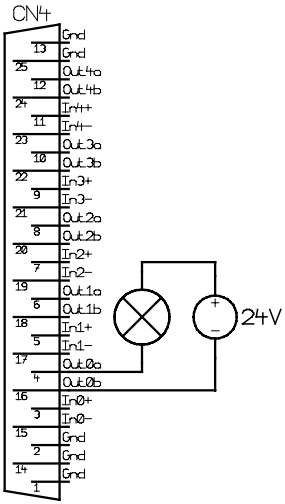

Abb. 5. Anschluss einer Kontrollleuchte an die Ausgänge.
Die Lampe muss so dimensioniert werden, dass die erlaubte Schaltlast der Kontakte des Moduls nicht überschritten wird. Das Netzteil, welches die Lampe versorgt, muss über eine ausreichende Leistung verfügen.

Installation der RS-232-Schnittstellenkarte

Die RS-232-Schnittstellenkarten werden optional als Zubehör zu dem COM-DIO-Modul geliefert. Die Beschreibung der Installation bezieht sich auf die Schnittstellenkarten der Firma EXSYS. Als Beispiel wird hier die Installation der Karte EX-41052 dargestellt, die zwei serielle Schnittstellen zur Verfügung stellt.

Die Installation ist in dem beiliegenden Faltblatt beschrieben. Beachten Sie dabei jedoch folgendes:

- Die Konfiguration der Kurzschlussbrücken auf der Karte soll nur dann geändert werden, wenn die Karte zur optionalen Stromversorgung des COM-DIO-Moduls verwendet werden soll. In solchem Falle müssen die Kurzschlussbrücken so eingestellt werden, dass die Karte auf den Pins 9 der beiden D-SUB-Stecker die Versorgungsspannung von +12 V zur Verfügung stellt.
- Es sollen nicht die Treiber aus der beiliegenden CD installiert werden, sondern sollen zunächst die aktuellen Versionen heruntergeladen werden. Der Hersteller der Karte stellt sie auf seiner Homepage zur Verfügung, für die o. g. Karte EX-41052 befinden sich die Treiber im Verzeichnis "EX-41052" des beiliegenden Softwarepakets oder hier: http://www.exsys.ch/download/driver/EX-41052.zip.
- Zum Installieren der Treiber benötigen Sie Administrationsrechte.
- Die Installation ist in der sich im jeweiligen Verzeichnis befindlichen pdf-Datei detailliert beschrieben. Lesen Sie diese Beschreibung sorgfältig durch.
- Nach der Installation sollen die Nummern der Schnittstellen eingestellt werden. Standardmäßig stellt die Karte die Schnittstellen COM3 und COM4 zur Verfügung. Aus programmtechnischen Gründen kann es günstig sein, für die Schnittstellen höhere Nummern zu verwenden, um Kollisionen mit anderen seriellen Schnittstelen bei verschieden Rechnern zu vermeiden. Dabei muss jedoch beachtet werden, dass die Softwareschnittstelle nur Schnittstellen COM1 bis COM9 ansprechen kann. Empfehlenswert ist es daher, die Schnittstellen auf COM8 und COM9 einzustellen. Dies wird im Gerätemanager durchgeführt, dazu sind Administrationsrechte erforderlich. Die Umstellung der Portnummer geschieht sofort nach der Bestätigung, der Rechner muss dabei nicht neu gestartet werden.

Softwareschnittstelle

Die Softwareschnittstelle zu dem COM-DIO-Modul besteht aus einer dynamischen Linkbibliothek `COM-IO.dll`. Sie befindet sich im Verzeichnis "Program" des beigelegten Softwarepakets. Die Softwareschnittstelle stellt ein selbständiges Softwarepaket dar, für ihre Funktion sind keine weiteren Treiber oder Bibliotheken erforderlich.

Die Benutzerfunktionen in der dynamischen Linkbibliothek `COM-IO.dll` können aus allen gängigen Programmiersprachen aufgerufen werden. Die Details entnehmen Sie dem Benutzerhandbuch Ihres Compilers. Die Definitionen der Benutzerfunktionen sind in der Datei `COM-IO.h` enthalten. Sollte Ihr Compiler keine Importbibliothek erstellen können, steht Ihnen die Datei `COM-IO.lib` zur Verfügung.

Die folgenden Beispiele sind für den Compiler Borland C++ Version 5.0 vorgesehen.

Funktion der Softwareschnittstelle

Die Softwareschnittstelle kann insgesamt 8 Kommunikationskanäle für die Datenübertragung verwalten. Jeder benutzte Kanal muss durch das Öffnen einer seriellen Schnittstelle zugeordnet werden. Ein geöffneter Kanal kann wieder geschlossen werden und danach durch das erneute Öffnen entweder der gleichen oder einer anderen Schnittstelle zugeordnet werden.

Das Öffnen eines Kommunikationskanals konfiguriert die zugeordnete serielle Schnittstelle und löscht ihre Puffer. Direkt nach dem Öffnen kann jede beliebige Funktion zur Datenübertragung aufgerufen werden. D.h. es kann der Zustand der Eingänge abgefragt werden oder können ausgewählte Ausgänge aktiviert werden.

Alle Funktionen erfordern als einen Parameter die Nummer des Kommunikationskanals `PortNumber` (siehe weiter), auf dem die gewünschte Operation ausgeführt werden soll. Es handelt sich um eine vorzeichenlose 16 Bit-Zahl (`WORD`). Die Nummer 0 bezeichnet den ersten Kanal, die Nummer 7 den letzten.

Der Rückgabewert der meisten Funktionen enthält die Aussage über den Erfolg der jeweiligen Operation. Es handelt sich um eine 32 Bit-Zahl mit Vorzeichen (`int`). Der letzte Rückgabewert kann durch die

Funktion `COM_DIO_State` neu abgefragt werden. Tabelle 3 listet die möglichen Rückgabewerte zusammen mit den Fehlermeldungen auf, wie sie durch die Funktion `COM_DIO_ErrorMessage` zurückgegeben werden. Ist ein Fehler bei der Datenübertragung entstanden, kann die Ursache durch Aufrufen der Funktionen `COM_DIO_IO_State` und `COM_DIO_IO_ErrorMessage` herausgefunden werden. Die erste gibt den letzten I/O-Fehler zurück, die zweite die entsprechende Fehlermeldung (siehe Tab. 4.).

Um eine einfachere Integration in Pascal-Softwarepakete zu vereinfachen, verwenden alle Funktionen der Softwareschnittstelle die Pascal-Aufrufkonvention.

Tab. 3. Rückgabewerte der Funktionen

Rückga-bewert	Fehlermeldung	Beschreibung
0	No error	Die Funktion endete erfolgreich.
1	No data is available	Es wurden keine Daten empfangen. Diesen Rückgabewert liefert nur die Funktion `COM_DIO_CheckInput` zurück.
-1	PortNumber out of range	Nummer des Kommunikationskanals `PortNumber` war größer als maximal zulässig.
-2	Memory for the port data could not be allocated	Bei der Initialisierung konnte dem Kommunikationskanal kein Speicher zugeteilt werden.
-3	Error opening the port	Die Schnittstelle konnte nicht geöffnet werden. Für mögliche Ursachen siehe Tab. 4.
-4	Error closing the port	Die Schnittstelle konnte nicht geschlossen werden. Für mögliche Ursachen siehe Tab. 4.
-5	Error purging the port	Die Puffer der Schnittstelle konnten nicht gelöscht werden. Für mögliche Ursachen siehe Tab. 4.
-6	Error sending command	Die Datenübertragung zum Modul ist fehlgeschlagen. Für mögliche Ursachen siehe Tab. 4.
-7	Error receiving command	Die Datenübertragung vom Modul ist fehlgeschlagen. Für mögliche Ursachen siehe Tab. 4.
-8	Wrong command received	Es wurde ein falsche Antwort vom Modul erhalten.
-9	Wrong argument received	Die vom Modul empfangene Antwort enthält falsche Daten.

Tab. 4. I/O-Fehler

Rückgabewert	Fehlermeldung	Beschreibung
0	No error	Die Funktion endete erfolgreich.
-1	PortNumber out of range	Nummer des Kommunikationskanals `PortNumber` war größer als maximal zulässig.
-2	Port not allocated	Bei der Initialisierung konnte dem Kommunikationskanal kein Speicher zugeteilt werden.
1	Port has not been opened yet	Es wurde versucht, einen Kommunikationskanal zu verwenden, der noch nicht geöffnet wurde.
2	Port has already been opened	Es wurde versucht, einen Kommunikationskanal zu öffnen, der bereits geöffnet wurde.
3	Cannot open the port	Die spezifizierte Schnittstelle konnte nicht geöffnet werden. Die Schnittstelle existiert entweder nicht oder wird von einem anderen Programm verwendet.
4	Cannot get the state of the port	Der Zustand der Schnittstelle konnte nicht ermittelt werden.
5	Cannot set the state of the port	Der Zustand der Schnittstelle konnte nicht gesetzt werden.
6	Cannot set the event mask for the port	Die Ereignismaske für die Schnittstelle konnte nicht gesetzt werden.
7	Cannot set the timeouts for the port	Das Timeout für die Schnittstelle konnte nicht gesetzt werden.
8	Cannot clear the port	Die Puffer der Schnittstelle konnten nicht gelöscht werden.
9	Error reading data from the port	Der Empfang der Daten ist fehlgeschlagen.
10	Error writing data to the port	Das Senden der Daten ist fehlgeschlagen.
11	Wrong data amount written to the port	Es konnte nicht die richtige Anzahl an Zeichen gesendet werden.

Rückgabewert	Fehlermeldung	Ursache
12	Error setting the control lines of the port	Die Steuerleitungen der Schnittstelle konnten nicht angesteuert werden.
13	Error reading the status lines of the port	Der Zustand der Statusleitungen der Schnittstelle konnte nicht ermittelt werden.

Steuerung der Kommunikation

Funktion COM_DIO_Open

```
int pascal COM_DIO_Open (WORD PortNumber,
  WORD ComNumber);
```

Öffnet die angegebene Schnittstelle für den Kommunikationskanal und liefert einen Fehlercode nach Tab. 3. Sie muss die erste Funktion sein, die während der Kommunikation mit dem jeweiligen Kommunikationskanal ausgeführt wird. Endet diese Funktion mit einem Fehler, ist keine Kommunikation mit dem Kanal möglich. Die Zahl `ComNumber` bezeichnet die Nummer der COM-Schnittstelle, die dem Kommunikationskanal zugeordnet werden soll. `ComNumber=1` steht beispielsweise für die Schnittstelle COM1.

Funktion COM_DIO_Close

```
int pascal COM_DIO_Close (WORD PortNumber);
```

Schließt den Kommunikationskanal und liefert einen Fehlercode nach Tab. 3. Sie kann verwendet werden, um den Kommunikationskanal freizugeben, um ihn später durch Öffnen der gleichen oder einen anderen Schnittstelle zuzuordnen.

Wird das Programm beendet, das die Softwareschnittstelle `COM-IO.dll` benutzt, werden alle geöffneten Kommunikationskanäle automatisch geschlossen. Der Programmierer muss keine Sorge dafür tragen.

Funktion COM_DIO_Purge

```
int pascal COM_DIO_Purge (WORD PortNumber);
```

Löscht die Schnittstellenpuffer und liefert einen Fehlercode nach Tab. 3. Sie kann verwendet werden, um eine gestörte Kommunikation wiederherzustellen. Wird beispielsweise das COM-DIO-Modul eingeschaltet, soll als erstes diese Funktion aufgerufen werden, um evtl. fehlerhaft empfangene Daten zu löschen.

Die Funktion `COM_DIO_Purge` wird automatisch beim Ausführen der Funktion `COM_DIO_Open` aufgerufen.

Ansteuerung der Ausgänge

Funktion COM_DIO_SetOutput

```
int pascal COM_DIO_SetOutput
  (WORD PortNumber, WORD Data);
```

Steuert die digitalen Ausgänge anhand der Variable `Data` an. Die fünf niedrigsten Bits entsprechen dabei jeweils jedem der Ausgänge. Dem Zustand des Ausgangs Out0 entspricht dabei Bit 0, Bit 4 wiederspiegelt den Zustand des Ausgangs Out4. Ist das jeweilige Bit gelöscht (0), wird das Ausgangsrelais ausgeschaltet und die Kontakte des Ausgangs geöffnet. Ist das jeweilige Bit gesetzt (1), wird das Ausgangsrelais aktiviert und die Kontakte des Ausgangs geschlossen.

Funktion COM_DIO_GetOutput

```
int pascal COM_DIO_GetOutput
  (WORD PortNumber, WORD & Data);
```

Liest den Zustand der digitalen Eingänge aus und speichert ihn in die Variable `Data`. Die Bits der Variable `Data` haben die gleiche Bedeutung wie bei der Funktion `COM_DIO_SetOutput`.

Die Funktion `COM_DIO_GetOutput` kann verwendet werden, um den Zustand der früher gesetzten Ausgänge abzufragen oder den aktuellen Zustand des COM-DIO-Moduls zu ermitteln, wenn die Software neu gestartet wird.

Auslesen der Eingänge

Funktion COM_DIO_GetInput

```
int COM_DIO_GetInput
  (WORD PortNumber, WORD & Data);
```

Liest die digitalen Eingänge aus und speichert den Zustand in die Variable `Data`. Die fünf niedrigsten Bits entsprechen dabei jeweils jedem der Eingänge. Dem Zustand des Eingangs In0 entspricht dabei Bit 0, Bit 4 wiederspiegelt den Zustand des Eingangs In4.

Wird diese Funktion aufgerufen, wird die automatische Datenübermittlung aus dem Modul unterbrochen, welche durch die Funktion `COM_DIO_AutoInput` gestartet wird.

Funktion COM_DIO_AutoInput

```
int COM_DIO_AutoInput (WORD PortNumber);
```

Startet die automatische Datenübermittlung der Zustände der Eingänge aus dem COM-DIO-Modul.

Wird diese Funktion aufgerufen, werden bei jeder Änderung des Zustandes der Eingänge die neuen Daten automatisch von dem COM-DIO-Modul gesendet. Um die Daten zu empfangen, soll periodisch die Funktion `COM_DIO_CheckInput` ausgeführt werden.

! Wurde die automatische Datenübermittlung des Zustandes der Eingänge aus dem COM-DIO-Modul gestartet und wird die Funktion `COM_DIO_CheckInput` nicht periodisch ausgeführt, wird die Kommunikation mit dem COM-DIO-Modul gestört. Die restlichen Funktionen können keine Daten mehr empfangen und enden mit einem Fehler.

Funktion COM_DIO_CheckInput

```
int COM_DIO_CheckInput
  (WORD PortNumber, WORD & Data);
```

Liest analog zu der Funktion `COM_DIO_GetInput` die digitalen Eingänge aus und speichert den Zustand in die Variable `Data`. Im Gegensatz zu der o.g. Funktion werden die Daten jedoch nicht angefordert, sondern nur die Daten ausgewertet, welche von dem COM-DIO-

Modul automatisch übermittelt werden. Die Funktion `COM_DIO_CheckInput` prüft, ob neue Daten durch die Schnittstelle empfangen wurden. Wenn ja, werden sie ausgelesen und ausgewertet. Die Funktion liefert einen der folgenden Rückgabewerte:

Rückgabewert	Bedeutung
0	Die Schnittstelle hat Daten empfangen, sie wurden in die Variable `Data` gespeichert
1	Die Schnittstelle hat keine Daten empfangen, die Variable `Data` wurde nicht geändert
<0	siehe Tab. 3

Die Funktion `COM_DIO_CheckInput` soll periodisch aufgerufen werden, wenn das COM-DIO-Modul Daten automatisch sendet. Die gesendeten Daten werden nach dem Versenden in dem Zwischenspeicher der RS-232-Schnittstelle im Steuerrechner gespeichert. Die Funktion `COM_DIO_CheckInput` soll so oft ausgeführt werden, dass der Zwischenspeicher nicht überlaufen kann. Da für die Übermittlung jedes Eingangszustandes insgesamt 4 Byte übertragen werden müssen, können bei der benutzen Übertragungsgeschwindigkeit theoretisch bis zu 200 Datensätze pro Sekunde übermittelt werden. Dies bestimmt also die maximal sinnvolle Wiederholungsrate der Aufrufe der Funktion `COM_DIO_CheckInput`. Die Zwischenspeicher der gängigen RS-232-Schnittstellen haben eine Länge von 14 Byte, können also bis zu 3 Datensätze aufnehmen. Für eine absolut sichere Übertragung reichen also etwa 70 Aufrufe der Funktion `COM_DIO_CheckInput` pro Sekunde, d.h. Aufrufe mit einem Zeitabstand von maximal 15 ms. In der tatsächlichen Anwendung soll die Funktion jedoch nur so oft aufgerufen werden, dass die beabsichtigte Zeitauflösung erreicht wird und dass keine Daten durch etwaige Überläufe des Zwischenspeichers verloren gehen.

Die automatische Datenübermittlung aus dem Modul wird durch die Funktion `COM_DIO_AutoInput` gestartet und durch die Funktion `COM_DIO_GetInput` unterbrochen.

Fehlerbehandlung

Funktion COM_DIO_State

```
int pascal COM_DIO_State (WORD PortNumber);
```

Liefert einen Fehlercode nach Tab. 3. Sie kann verwendet werden, um das Ergebnis der letzten Operation erneut abzufragen. Die Funktion hat keine Einfluss auf die Kommunikation und kann zu jeder Zeit aufgerufen werden.

Funktion COM_DIO_ErrorMessage

```
const char * pascal COM_DIO_ErrorMessage
  (WORD PortNumber);
```

Liefert eine Fehlermeldung zu dem Ergebnis der letzten Operation. Der Rückgabewert ist ein Zeiger auf eine null-terminierte Zeichenkette, der Inhalt der Fehlermeldung kann Tab. 3 entnommen werden. Die Funktion hat keine Einfluss auf die Kommunikation und kann zu jeder Zeit aufgerufen werden.

Funktion COM_DIO_IO_State

```
int pascal COM_DIO_IO_State
  (WORD PortNumber);
```

Liefert einen I/O-Fehlercode nach Tab. 4. Sie kann verwendet werden, um das Ergebnis der letzten I/O-Operation erneut abzufragen. Die Funktion hat keine Einfluss auf die Kommunikation und kann zu jeder Zeit aufgerufen werden.

Funktion COM_DIO_IO_ErrorMessage

```
const char * pascal COM_DIO_IO_ErrorMessage
  (WORD PortNumber);
```

Liefert eine Fehlermeldung zu dem Ergebnis der letzten I/O-Operation. Der Rückgabewert ist ein Zeiger auf eine null-terminierte Zeichenkette. Der Inhalt der Fehlermeldung kann Tab. 4 entnommen werden. Die Funktion hat keine Einfluss auf die Kommunikation und kann zu jeder Zeit aufgerufen werden.

Verschiedenes

Funktion COM_DIO_Version

```
WORD pascal COM_DIO_Version();
```

Liefert die Version der Softwareschnittstelle (der dynamischen Linkbibliothek `COM-IO.dll`). Sie soll verwendet werden, um festzustellen, ob eine Softwareschnittstelle mit der passenden Version verwendet wird. Die Funktion soll aufgerufen werden, bevor die restlichen Funktionen der Softwareschnittstelle verwendet werden.

Der Rückgabewert ist eine vorzeichenlose 16 Bit-Zahl (WORD), in dem höheren Byte wird die Hauptversionsnummer zurückgegeben, das niedrigere Byte bezeichnet die Reihenfolge innerhalb der Hauptversion. Eine identische Hauptversionsnummer zweier Bibliotheken `COM-IO.dll` bedeutet, dass diese gleiche Funktionen implementieren und dass lediglich interne Korrekturen vorgenommen wurden. Sind die Hauptversionsnummern unterschiedlich, besteht keine Garantie, dass die Bibliothek eingesetzt werden kann, ohne dass das Programm neu kompiliert oder sogar modifiziert werden muss.

Funktion COM_DIO_FW_Ver

```
int pascal COM_DIO_FW_Ver
  (WORD PortNumber, WORD & Version);
```

Liefert die Version der Firmware des COM-DIO-Moduls. Sie soll verwendet werden, um festzustellen, ob die Softwareschnittstelle eine passende Version zu der Firmware hat.

Der Rückgabewert ist wie bei der Funktion `COM_DIO_Version` eine vorzeichenlose 16 Bit-Zahl `Version`. Die dynamische Linkbibliothek `COM-IO.dll`. und die Firmware des COM-DIO-Moduls sind miteinander kompatibel, wenn die Hauptversionsnummern identisch sind.

Funktion COM_DIO_FW_Date

```
int pascal COM_DIO_FW_Date
  (WORD PortNumber, char * DateString);
```

Liefert das Kompilationsdatum der Firmware des COM-DIO-Moduls. Der Rückgabewert ist eine null-terminierte Zeichenkette, sie wird in den Puffer `DateString` kopiert. Der Puffer `DateString` muss vor

dem Aufruf der Funktion definiert werden und muss groß genug sein, um 11 Zeichen des Rückgabewerts (beispielsweise `19 JUN 2009`) und das Abschlusszeichen aufnehmen zu können.

Funktion COM_DIO_prod_No

```
int pascal COM_DIO_prod_No
  (WORD PortNumber, DWORD & Number);
```

Liefert die Produktnummer des COM-DIO-Moduls. Der Rückgabewert `Number` ist eine vorzeichenlose Zahl.

Funktion COM_DIO_prod_ID

```
int pascal COM_DIO_prod_ID
  (WORD PortNumber, char * Identification);
```

Liefert den Identifikationstext des COM-DIO-Moduls. Der Rückgabewert `Identification` ist ein Zeiger auf eine null-terminierte Zeichenkette.

Beispiel eines Kommunikationsprogramms

```
#include <stdio.h>
#include <conio.h>
#include <windows.h>

#include "COM-DIO.h"


int main()
   {
   // define the port number:
   unsigned short PortNum = 0;

   // open the port, attach it to COM1:
   if (COM_DIO_Open (PortNum, 1))
      {
      // handle a possible error:
      printf ("%s\n", COM_DIO_ErrorMessage (PortNum));
      return 1;
      }

   printf ("Press 'o' to set output data, "
               "'i' to get identification data, "
               "or 'Esc' to stop\n");

   // Set automatic transmission of input state:
   if (COM_DIO_AutoInput (PortNum))
      printf ("%s\n", COM_DIO_ErrorMessage (PortNum));

   // main program loop:
   while (true)
      {
      WORD Data;
      // check for data:
      const int Result = COM_DIO_CheckInput (PortNum, Data);

      if (Result < 0) // handle a possible error
         printf ("%s\n", COM_DIO_ErrorMessage (PortNum));

      if (Result == 0) // if data is ready, print it
         printf ("Input data received: %d (%02Xh)\n",
            Data, Data);

      if (kbhit() == 0) continue; // repeat if no key is pressed

      const int Key = getch(); // read the pressed key

      // set the output state:
      if (Key == 'o')
         {
         printf ("Enter new hexadecimal output state: ");
         scanf ("%X", &Data);
```

```
      if (COM_DIO_SetOutput (PortNum, Data))
         printf ("%s\n", COM_DIO_ErrorMessage (PortNum));
      }

   // get identification data:
   if (Key == 'i')
      {
      // get firmware version:
      WORD Version;
      if (COM_DIO_FW_Ver (PortNum, Version))
         printf ("%s\n", COM_DIO_ErrorMessage (PortNum));
      else
         printf ("Firmware version: %d.%02d\n",
            Version / 0x100, Version & 0xFF);

      // get firmware date:
      char DateString [20];
      if (COM_DIO_FW_Date (PortNum, DateString))
         printf ("%s\n", COM_DIO_ErrorMessage (PortNum));
      else
         printf ("Firmware date: %s\n", DateString);

      // get product identification:
      char IDString [20];
      if (COM_DIO_prod_ID (PortNum, IDString))
         printf ("%s\n", COM_DIO_ErrorMessage (PortNum));
      else
         printf ("Product identification: %s\n", IDString);

      // get product number:
      DWORD Number;
      if (COM_DIO_prod_No (PortNum, Number))
         printf ("%s\n", COM_DIO_ErrorMessage (PortNum));
      else
         printf ("Product number: %d\n", Number);
      }

   // stop on 'Esc':
   if (Key == 27)
      break;
   }

//finish the program:
return 0;
}
```