QUANCOM
INFORMATIONSSYSTEME

USBAD8DAC2
USBAD8DAC2/14
USBTTL24

Wesseling, März 2007 Version 4.1.0

Inhaltsverzeichnis

1. Überblick

Wir beglückwünschen Sie zum Kauf Ihres QUANCOM Meß-, Steuer- und Regelmoduls. Sie haben sich für ein Produkt entschieden, dessen Eigenschaften und Funktionalität den neuesten Stand der Technik darstellen. Zu den besonderen Eigenschafen dieser Karte gehören:

- Einfacher Anschluß über USB
- Einfach programmierbar
- Diverse Beispielprogramme in verschiedenen Programmiersprachen
- Treiberunterstützung unter Windows XP, 2000 und ME/98 mit der QLIB **(QUANCOM Driver Library)**

1.1 Unsere Erfahrung ist ihr Gewinn

Wir von QUANCOM sind auf die Entwicklung von Hardware und Software spezialisiert und gehören mittlerweile zu einem der führenden Lieferanten für Meßtechnik und Automatisierung. In ihrem Entwicklungszentrum hat QUANCOM eine eindrucksvolle Produktpalette entwickelt.

1.2 Kommunikation mit unseren Kunden

QUANCOM möchte gerne Ihren Kommentar zu unseren Produkten und zu unseren Handbüchern. erhalten und ist außerdem an Ihren Anwendungen interessiert, die Sie mit unseren Produkten entwickeln. Wir möchten gleichzeitig helfen, wenn Sie Probleme haben und um dies zu vereinfachen enthält dieses Handbuch Kommentar- und Konfigurationsformulare, mit denen man direkt mit uns in Verbindung treten kann. Diese Formulare befinden sich in dem Kapitel **“Dokumentations Formular”** am Ende dieses Handbuches.

1.3 Änderungen zu diesem Handbuch und Softwareupdates

QUANCOM - Produkte zeichnen sich u.a. durch stetige Weiterentwicklung aus. Aktuelle Informationen über Änderungen können Sie den README - Dateien auf der Installationsdiskette oder CD entnehmen. Weitere Informationen und kostenlose Softwareupdates können Sie jederzeit auf den QUANCOM Internet -Seiten unter **www.quancom.de** erhalten.

1.4 Lieferumfang

- USB-Messtechnik-Modul
- 1,8 Meter USB-Kabel
- QUANCOM CD
- Benutzerhandbuch ist als PDF auf der QUANCOM CD enthalten

Sollten eine oder mehrere Komponenten fehlen wenden Sie sich bitte an Ihren Händler. QUANCOM behält sich das Recht vor, Änderungen im Lieferumfang ohne Vorankündigung vorzunehmen.

2. Installationsverfahren

2.1 Systemvorraussetzungen

- Personal Computer: Die QUANCOM Karten laufen mit einem IBM-AT Computer mit 80X86 oder in einem kompatiblen Gerät, wie z.B. Pentium.
- Ihr Computer muß über mindestens eine freie USB Schnittstelle verfügen.

Mehr Informationen hierzu finden Sie im Abschnitt **“Technische Hardwarebeschreibung"**

2.2 Sicherheitsanweisungen

Im Interesse Ihrer eigenen Sicherheit und einer einwandfreien Funktion Ihres neuen QUANCOM-Moduls beachten Sie bitte die folgenden Hinweise:

- Da PC-Karten/Module empfindlich gegen elektrostatische Aufladungen sind ist es wichtig, sich vorher zu entladen bevor die Karte mit den Händen oder dem Werkzeug berührt wird. Dies geschieht am einfachsten, wenn Sie vorher ein metallisches Gehäuseteil berühren.

- Halten Sie die Karte immer am Rand fest und vermeiden Sie ein Anfassen der IC´s.

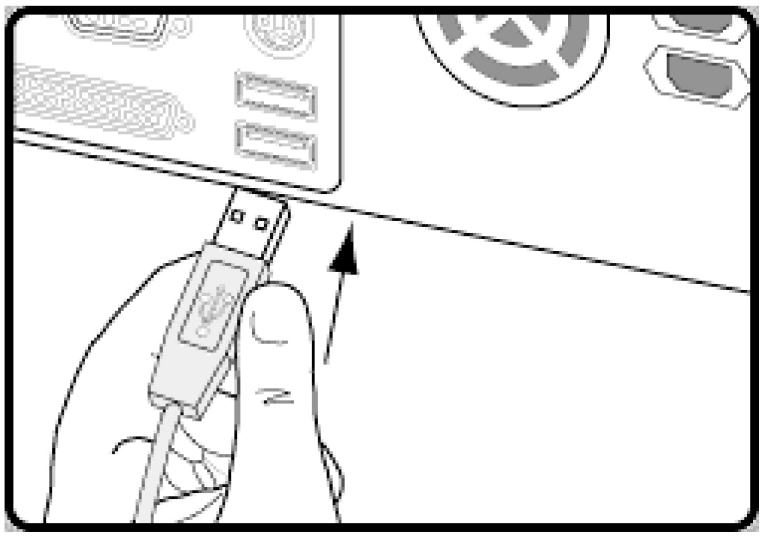
Veränderungen, die ohne ausdrückliche Genehmigung der QUANCOM Informationssysteme GmbH an dem Gerät vorgenommen werden, führen zum Erlöschen der Betriebserlaubnis und der CE Zertifizierung.

2.3 Installation des Moduls

Den PC einschalten, Windows starten und das USB-Verbindungkabel mit dem Gerät und dem USB-Port des PCs verbinden.

Das Modul kann jederzeit während des Betriebs des PCs angeschlossen werden.

Der Ort, an dem sich der USB-Port befindet, ist in der Bedienungsanleitung Ihres PCs beschrieben. Er ist je nach PC-Modell verschieden.



2.4 Technische Hardwarebeschreibung

2.4.1 Beschreibung

Die USB-Module lassen sich sehr einfach über das mitgelieferte USB-Kabel an Ihren PC anschliessen. Durch die zuvor installierte QLIB wird das Gerät sofort von Windows erkannt. Die Stromversorgung erfolgt über den USB-Bus.
Optional haben Sie die Möglichkeit die USB-Module über eine externe 12V Stromversorgung zu betreiben. Hierfür müssen Sie lediglich den Jumper für die Stromversorgung von USB auf Extern stecken und eine separate 12V Spannungsquelle anschließen. Für den Anschluss der zu messenden Signale stehen Ihnen komfortable, steckbare Schraubklemmen zur Verfügung.

AD/DA

Bei diesen professionellen USB-Modulen haben Sie die Auswahl zwischen 2 unterschidlichen Eingangsmodi. Dem Differential ended Modus mit 4 A/D Kanälen und einer Auflösung 12 (14) Bit oder dem Single ended Modus mit 8 /AD Kanälen und einer 11 (13) Bit Auflösung. Die Auswahl welchen Modus Sie nutzen wollen, können Sie bequem per Software treffen. Der Single ended Modus nutzt einen Spannungsbereich von +10V bei gleicher Wandlungszeit.
Als Ausgänge stehen Ihnen 2 D/A Kanäle mit einer Auflösung von 10 Bit zur Verfügung. Der D/A Ausgang nutzt einen Spannungsbereich von + 0..5V bei einer Stromaufnahme von max. 10mA

TTL

Zusätzlich haben diese Module noch 2*32Bit Counter und 24 TTL Kanäle die als Ein- oder Ausgänge fungieren können. Die Ein- und Ausgabe der 24 TTL/IO Kanäle und die beiden 32 Bit Counter erfolgen über die 37 polige D-SUB-Buchse. Die Pinbelegung der D-SUB Buchse erfahren Sie in Kapitel 3.6.

2.4.2 A/D Differential-ended (d.e)

A/D Kanäle	4
A/D Auflösung	12 Bit (optional 14 Bit)
A/D Wandlungszeit	20 us
A/D Meßbereich	±20 V, ±10 V, ±5 V, ±4 V, ±2,5 V, ±2 V, ±1,25 V, ±1 V

2.4.3 A/D Single-ended (s.e.)

A/D Kanäle	8
A/D Auflösung	11 Bit (optional 13 Bit)
A/D Wandlungszeit	20 us
A/D Meßbereich	±10 V

2.4.4 D/A Kanäle

D/A Kanäle	2
D/A Auflösung	10 Bit
D/A Spannungsbereich	0..5V
D/A Strom max.	10mA
Steckverbinder	16 pol. Klemmblock

2.4.5 TTL I/O

Kanäle	24 (in 8er Blöcken)
Steckverbinder	37-pol. D-Sub

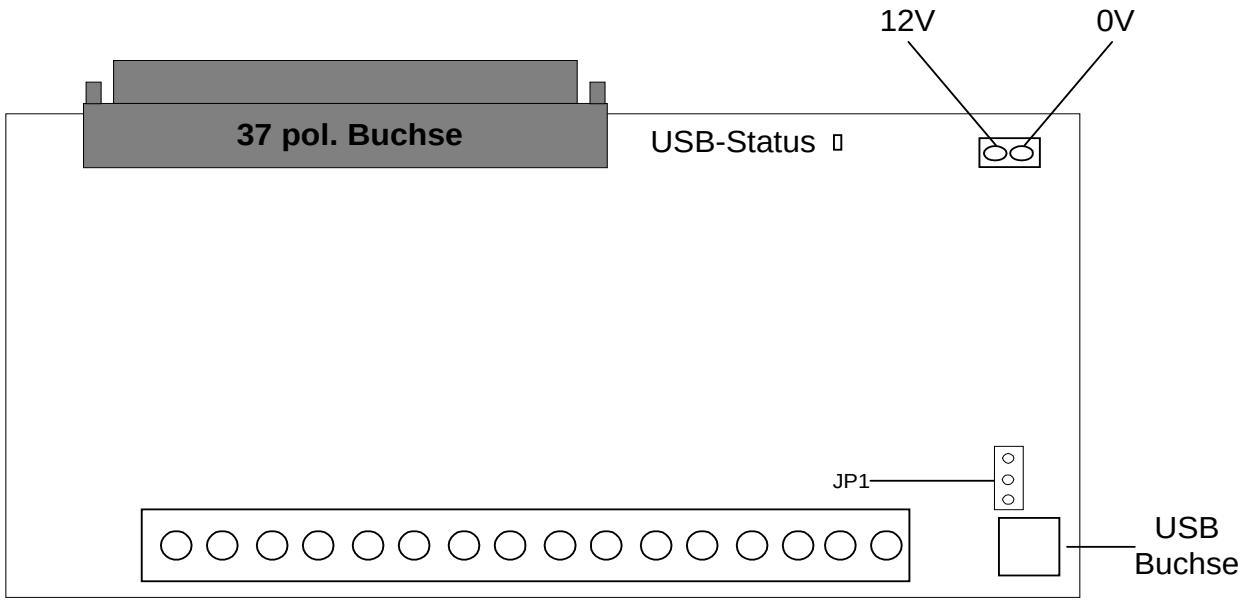
2.4.6 Zähler

Anzahl	2
Auflösung	32 Bit
Frequenz	Max. 1MHz
Steckverbinder	37-pol. D-Sub

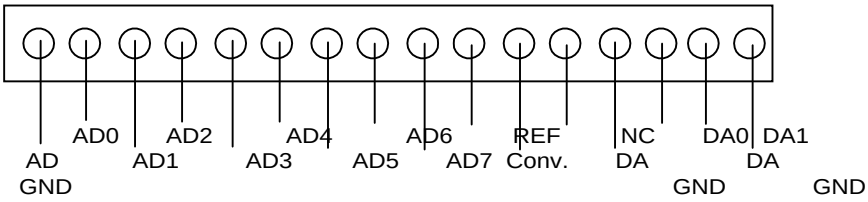
2.4.7 Sonstiges

Abmessungen	132*72 mm
Temperatur Bereich	0...50°

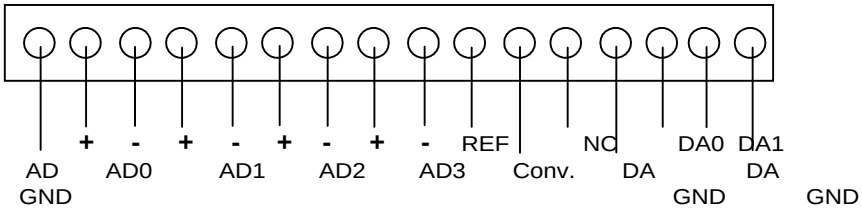
2.5 Kartenübersicht



2.6 Steckerbelegung bei Differential - ended (4 Kanäle)

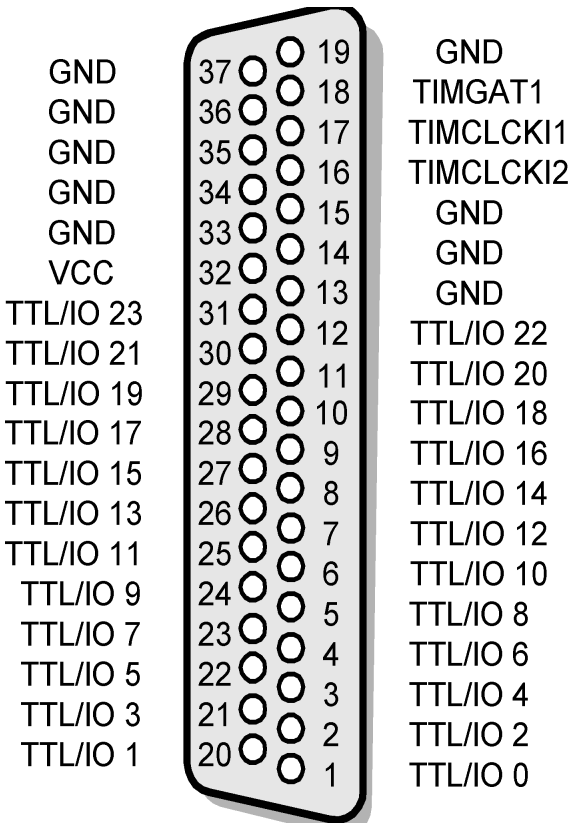


2.7 Steckerbelegung bei Single - ended (8 Kanäle)



AD0 ... AD7: A/D Kanäle
REF: Interne 2,5V Ref. Spannung
DA0,1: D/A Ausgang

2.8 Pinbelegung der 37 pol. D-Sub Buchse



TTL/IO 0 - 23	24 TTL/IO Kanäle
TIMGAT1	Gate für Timer 1 (Timer ist enabled bei Gate =1)
TIMCLKI 1-2	Zähler Eingang 1 - 2
VCC	+5 V
GND	Ground

2.9 Einstellung der Stromversorgung (JP1)

Bei diesen Modulen können Sie bei der Stromversorgung wahlweise zwischen der internen Versorgung über die USB Schnittstelle und einem externen 12 V Anschluss wählen.
Mit Hilfe von Jumper 1 wählen Sie aus, über welche Art der Stromversorgung die Module betrieben werden soll.

extern USB*

(* Standardeinstellung)

Durch dieses Feature ist es möglich, die USB-ADA8DAC2 auch an Geräten zu betreiben, welche die USB-Spezifikationen, der Spannungsversorgung nicht einhalten.

3. Programmierung des Moduls

3.1 Welche Software brauche ich ?

Die Software die ich brauche ist abhängig von der jeweiligen Anwendung und dem jeweiligen Betriebssystem. **Um Zugriff per Programm auf die Karte zu bekommen bestehen folgende Möglichkeiten:**

- **Methode 1:** High-Level Programmierung (Zugriff auf die Karte über die QLIB) unter Windows XP / 2000 / ME / 98 / 95. Hiermit ist es möglich die Karte z.B. über Visual-C, Visual-Basic, Borland Delphi, Lotus Notes u.a. Compilern und Interpretern anzusprechen.
- **Methode 2:** Installation der QLIB in Zusammenhang mit einem anderen Programm. Beispiele zu dem Einsatz mit Labview von National Instruments finden Sie nach der Installation der QLIB im Verzeichnis samples\USBADA8DAC2.

Wenn Sie **Methode 1** anwenden möchten, benötigen Sie den Quelltext der Anwendung. Sie sind selbst für das Hinzufügen der Befehle in Ihre Anwendung verantwortlich. Um diese Methoden zu benutzen sind Programmierkenntnisse erforderlich..

Methode 2 erlaubt einem das man die QUANCOM Karte mit einer bestehenden Software laufen lassen kann z.B. LabView oder Agilent VEE . Dafür müssen Sie als erstes die QLIB von der Installations CD installieren. Hinweise rund um die QLIB und Installation entnehmen Sie bitte dem QLIB Handbuch, welches sich ebenfalls auf der CD befindet. Auf der CD finden Sie gleichzeitig einige Beispielprogramme für LabView und Agilent VEE.

3.2 QLIB: High Level Programmierung (Windows XP / 2000 / ME / 98)

3.2.1 QLIB (QUANCOM Driver Library)

Die QLIB (die Abkürzung für **QUANCOM LIBRARY**) bietet die Möglichkeit, alle QUANCOM-Karten und Module unter den Betriebssystemen Windows XP/2000 und ME/98 mit den Programmiersprachen C/C++/Delphi/Visual Basic anzusprechen. Sie wird zu allen QUANCOM-Karten mitgeliefert und gestattet durch die Einfachheit der Befehle dem Anwender, die QLIB in eigene Applikationen einzubinden. Die Befehle und Funktionen gelten für alle Betriebssysteme.

Unterstützte Betriebssysteme:

Microsoft Windows Vista / XP / 2000 / ME / 98

Unterstützte Compiler:

C #/ C++

• Borland C/C++

• Microsoft® Visual C/C++ 6.x

• Microsoft® Visual C#.NET

• Microsoft® Visual Studio C/C++.NET

JAVA

• Sun JAVA

Pascal

• Borland Turbo Pascal

Delphi

• Borland Delphi

Basic

• Microsoft® Visual Basic 6.x

• Microsoft® Visual Studio VB.NET

Grafische Programmiersprache:

• Agilent VEE von Agilent

• LabView® 6.0 von National Instruments

4. Schnellinstallation der QLIB für USB

Für die Installation der Treiber und Laufzeitumgebung sind Administratorenrechte erforderlich. Ohne die entsprechenden Rechte kann der Treiber und die Laufzeitumgebung nicht korrekt installiert werden.

Windows XP/2000	Windows ME/98
USB-Modul einstecken	USB-Modul einstecken
Rechner starten	Rechner starten
USB-Version wird vom Betriebsystem erkannt, bitte den Pfad zu den Treibern angeben. (Treiber befinden sich im Verzeichnis **WinXP** oder **Win2000** der Installations-CD)	USB-Version wird vom Betriebsystem erkannt, bitte den Pfad zu den Treibern angeben. (Treiber befinden sich im Verzeichnis **WinME** oder **Win98** der Installations-CD)
QLIB, Testprogramme und Beispiele installieren (Im Hauptverzeichnis der Installations-CD **QUANCOM.EXE** starten)	QLIB, Testprogramme und Beispiele installieren (Im Hauptverzeichnis der Installations-CD **QUANCOM.EXE** starten)
Rechner neu starten	Rechner neu starten

5. QLIB Befehle

Die folgende Auflistung enthält alle von dieser Karte verwendeten QLIB-Befehle. Diese unterscheiden sich in erweiterte (QAPIExt...) und einfache (QAPI...) Funktionen. Bei den einfachen Funktionen werden keine Kartenhandles (Kartenparameter) übergeben. Dadurch wird nur die erste vom System erkannte Karte eines Typs angesprochen und verwendet. Sollen mehrere Karten verwendet werden, so müssen die erweiterten Funktionen angwendet werden.

5.1 Verwaltungsfunktionen

QAPINumOfCards

Mit der Funktion **QAPIExtNumOfCards** wird abgefragt, wieviele unterschiedliche Karten- bzw.

Modultypen von der QLIB unterstützt werden.

ULONG QAPINumOfCards (void);

QAPIGetLastError

Die **QAPIGetLastError** Funktion liefert den letzten Fehlercode des aufrufenden Threads. Der letzte
Fehlercode wird dabei für jeden Thread gesondert gespeichert. Mehrere Threads überschreiben ihre
Fehlercodes nicht gegenseitig.

ULONG QAPIGetLastError (void);

QAPIGetLastErrorCode

Die **QAPIGetLastErrorCode** Funktion liefert den letzten erweiterten Fehlercode eines vorher
aufgerufenen QAPIGetLastError Befehls.

ULONG QAPIGetLastErrorCode (void);

5.1.1 AD/DA (Einfach)

QAPIGetAD

Mit der Funktion **QAPIGetAD** wird ein Digitalwert von einem Eingangskanal einer A/D-Karte eingelesen.

ULONG **QAPIGetAD** (**ULONG cardid**,**ULONG channel**,);

Funktionsprototypen für Delphi und Vb

Parameter

cardid

Gibt die ID der Karte an, über welche Informationen abgefragt werden sollen.

channel

Gibt den Kanal an, von welchem der Digitalwert eingelesen werden soll.

QAPIPutDA

Mit dieser Funktion **QAPIPutDA** wird ein Digitalwert auf einen Kanal einer D/A-Karte ausgegeben.

ULONG QAPIPutDA (**ULONG cardid**,**ULONG channel**,**ULONG value**);

Funktionsprototypen für Delphi und Vb

Parameter

cardid

Gibt die ID der Karte an, auf welcher Digitalwerte ausgegeben werden sollen.

channel

Gibt den Kanal an, auf dem ein Digitalwert ausgegeben werden soll.

value

Gibt den Digitalwert an, welcher ausgegeben werden soll.

QAPIConvertDWToVoltage

Mit der Funktion **QAPIConvertDWToVoltage** wird ein Digitalwert in einen Analogwert umgerechnet.

Diese Funktion steht bei Karten bzw. Modulen mit A/D Wandler zur Verfügung.

float **QAPIConvertDWToVoltage** (**ULONG cardid**,**ULONG value**,**ULONG mode**);

Bis QLIB Version 1.96

float **QAPIConvertDWToVoltage** (**ULONG cardid**,**ULONG value**,);

Funktionsprototypen für Delphi und Vb

Parameter

cardid

Gibt die ID der Karte an, von welcher Digitalwerte umgewandelt werden.

value

Gibt den Digitalwert an, welcher umgerechnet werden soll.

mode

Gibt einen kartenabhängigen Parameter an (siehe auch kartenspezifische Parameter).

QAPIConvertVoltageToDW

Mit der Funktion **QAPIConvertVoltageToDW** wird ein Analogwert in einen Digitalwert umgerechnet.

Diese Funktion steht bei Karten mit D/A Wandler zur Verfügung.

ULONG **QAPIConvertVoltageToDW** (**ULONG** **cardid**,float **value**,**ULONG** **mode**);

Funktionsprototypen für Delphi und Vb

Parameter

cardid

Gibt die ID der Karte an, von welcher Digitalwerte umgewandelt werden.

value

Gibt den Analogwert an, welcher umgerechnet werden soll.

mode

Gibt einen kartenabhängigen Parameter an (siehe auch kartenspezifische Parameter).

5.2 QLIB Befehle (Erweitert)

5.2.1 Verwaltungsfunktionen (Erweitert)

QAPIExtOpenCard

Mit der Funktion **QAPIExtOpenCard** wird eine Karte geöffnet.

ULONG **QAPIExtOpenCard** (**ULONG** cardid,**ULONG** devnum);

Funktionsprototypen für Delphi und VB

Parameter

cardid
Gibt die ID der Karte an, welche geöffnet werden soll.

devnum
Gibt die Device-Nummer der Karte an, die geöffnet werden soll.

QAPIExtCloseCard

Mit der Funktion **QAPIExtCloseCard** wird eine Karte geschlossen.

void **QAPIExtCloseCard** (**ULONG** cardhandle);

Funktionsprototypen für Delphi und VB

Parameter

cardhandle
Gibt das Handle einer geöffneten Karte an.

QAPIExtNumOfCards

Mit der Funktion **QAPIExtNumOfCards** wird abgefragt, wieviele unterschiedliche Karten- bzw.
Modultypen von der QLIB unterstützt werden.

ULONG QAPIExtNumOfCards (void);

Funktionsprototypen für Delphi und VB

Parameter

Diese Funktion benötigt keine Parameter.

QAPIGetLastErrorStringEx

Die QAPIGetLastErrorStringEx dient dazu einen String zu erzeugen, der eine lesbare Fehlermeldung
aus den QLIB Fehlercodes erzeugt, die von der Funktion **QAPIGetLastError** bzw. **QAPIGetLastErrorCode** zurückgeliefert werden. Ein Beispiel finden Sie unter Fehler codes .

ULONG QAPIGetLastErrorStringEx(char* buffer, **ULONG** buffersize);

Funktionsprototypen für Delphi und Vb

Parameter

buffer
Zeiger auf einen Puffer der den nullterminierten Fehlerstring erhält.

buffersize
Dieser Parameter gibt die Grösse des übergebenen Puffers in Bytes an.

5.2.2 AD/DA Funktionen Erweitert

QAPIExtReadAD

Mit der Funktion **QAPIExtReadAD** wird ein Digitalwert von einem Eingangskanal einer A/D-Karte
eingelesen.

ULONG **QAPIExtReadAD**(**ULONG cardhandle**,**ULONG channel**,**ULONG mode**);

Funktionsprototypen für Delphi und VB

Parameter

cardhandle
Gibt das Handle einer geöffneten Karte an.

channel
Gibt den Kanal an, von welchem ein Digitalwert eingelesen werden soll.

mode
Gibt einen kartenabhängigen Parameter an (siehe auch kartenspezifische Parameter).

QAPIExtWriteDA

Mit dieser Funktion **QAPIExtWriteDA** wird ein Digitalwert auf einen Kanal einer D/A-Karte ausgegeben.

void **QAPIExtWriteDA**(**ULONG cardhandle**,**ULONG channel**,**ULONG value**,**ULONG mode**);

Funktionsprototypen für Delphi und VB

Parameter

cardhandle
Gibt das Handle einer geöffneten Karte an.

channel
Gibt den Kanal an, auf dem ein Digitalwert ausgegeben werden soll.

value
Gibt den Digitalwert an, welcher ausgegeben werden soll.

mode
Gibt einen kartenabhängigen Parameter an. (siehe auch kartenspezifische Parameter).

QAPIExtLatchDA

Mit der Funktion **QAPIExtLatchDA** werden alle Digitalwerte auf die Kanäle ausgegeben.

void **QAPIExtLatchDA** **(ULONG cardhandle**);

Funktionsprototypen für Delphi und VB

Parameter

cardhandle
Gibt das Handle einer geöffneten Karte an.

QAPIExtConvertDWToVoltage

Mit der Funktion **QAPIExtConvertDWToVoltage** wird ein Digitalwert in einen Analogwert umgerechnet.

Diese Funktion steht bei Karten bzw. Modulen mit A/D Wandler zur Verfügung.

float **QAPIExtConvertDWToVoltage** (**ULONG cardhandle**,**ULONG value**,**ULONG mode**);

Funktionsprototypen für Delphi und VB

Parameter

cardhandle
Gibt das Handle einer geöffneten Karte an.

value
Gibt den Digitalwert an, welcher umgerechnet werden soll.

mode
Gibt einen kartenabhängigen Parameter an (siehe auch kartenspezifische Parameter).

QAPIExtConvertVoltageToDW

Mit der Funktion **QAPIExtConvertVoltageToDW** wird ein Analogwert in einen Digitalwert umgerechnet.

Diese Funktion steht nur für Karten mit D/A Teil zur Verfügung.

ULONG **QAPIExtConvertVoltageToDW** (**ULONG cardhandle**,**float value**,**ULONG mode**);

Funktionsprototypen für Delphi und VB

Parameter

cardhandle
Gibt das Handle einer geöffneten Karte an.

value
Gibt den Analogwert an, welcher umgerechnet werden soll.

mode
Gibt einen kartenabhängigen Parameter an (siehe auch kartenspezifische Parameter).

5.2.3 Sonstige Funktionen

QAPIExtGetCardInfo

Mit der Funktion **QAPIExtGetCardInfo** lassen sich Informationen über eine Karte einlesen.

LPCARDDATAS **QAPIExtGetCardInfo**(ULONG **cardid**);

Funktionsprototypen für Delphi und VB

Parameter

cardid

Gibt die ID der Karte an, über welche Informationen abgefragt werden sollen. Der Feld SizeOf muss

mit der Grösse der Struktur initialisiert werden, sonst schlägt die Funktion fehl.

QAPIExtGetCardInfoEx

Mit der Funktion QAPIExtGetCardInfoEx können Informationen einer Karte abgerufen werde.Diese

werden in dem vom der Anwendung übergebenem Speicher geschrieben.

ULONG **QAPIExtGetCardInfoEx**(ULONG **cardid**,LPCARDDATAS **lpcd**);

Funktionsprototypen für Delphi und VB

Parameter

cardnum

Gibt die ID der Karte an, über welche Informationen abgefragt werden sollen

lpcd

Ist ein Zeiger auf eine Struktur vom Typ CARDDATAS

QAPIExtReleaseCardInfo

Mit der Funktion QAPIExtReleaseCardInfo werden die mit QAPIExtGetCardInfo abgefragten Karteninformationen freigegeben.

void **QAPIExtReleaseCardInfo**(LPCARDDATAS **lpcarddatas**);

Funktionsprototypen für Delphi und VB

Parameter

lpcarddatas

Zeigt auf eine Struktur vom Typ CARDDATAS.

5.3 Programmierung mit der QLIB (QUANCOM Driver Library)

Stellen Sie sicher, dass die QLIB (QUANCOM Library) installiert ist.
Weitere Informationen über die Installation und wie man die entsprechenden Dateien in die Anwendung einbindet entnehmen Sie bitte der “32-Bit QLIB API Help” Dokumentation. Das folgende Kapitel erläutert spezielle Befehle die benötigt werden, um die Karte mit der QLIB benutzen zu können.

5.3.1 Einlesen der A/D Kanäle unter VB

```
Function IsError() As Integer

        Dim buffer As String * 256
        Dim Result As Long

        Result = QAPIGetLastError()

        If (Result <> 0) Then

            Result = QAPIGetLastErrorStringEx(buffer, Len(buffer))

            MsgBox (Left(buffer & " ", Result))

        Else

        End If

       IsError = (Result <> 0)

End Function

' read analog inputs

Sub     ButtonReadAI()

       Dim nDevice As Long
       Dim handle As Long
       Dim advalue As Long
       Dim value As Single
       Dim nMode As Long
       Dim channel as long

       ' find first module

       nDevice = 0

       ' open module

       handle = QAPIExtOpenCard(USBAD8DAC2, nDevice)

       if ( handle = 0 ) then
          MsgBox "Unable to open any USBAD8DAC2 module!"
          exit sub
       end if

       ' read all 8 adc channels

       for channel = 0 to 7

          ' 0 = MODE_BI_5V
          ' 1 = MODE_BI_10V
          ' 2 = MODE_BI_3V3
          ' 6 = MODE_BI_2V5
          ' 7 = MODE_BI_1V25
          ' 4 = MODE_UNI_5V
          ' 3 = MODE_UNI_10V
          ' 5 = MODE_UNI_3V3
          ' 8 = MODE_UNI_2V5
          ' 9 = MODE_UNI_1V25

         nMode = MODE_BI_10V

         ' read a/d value

         advalue = QAPIExtReadAD(handle, channel, nMode): If IsError() Then goto terminate_sub

         ' convert digital value to floating point voltage value

         value = QAPIExtConvertDWToVoltage(handle, advalue, nMode): If IsError() Then goto
       terminate_sub

         ' print voltage to console

         debug.print "Channel " & channel & " = " & value & " Volts"
```

```
       next channel
terminate_sub:
       call QAPIExtCloseCard(handle)
end sub
```

5.3.2 Ansteuerung der D/A Kanäle unter VB

```
Function IsError() As Integer

        Dim buffer As String * 256
        Dim Result As Long

        Result = QAPIGetLastError()

        If (Result <> 0) Then

            Result = QAPIGetLastErrorStringEx(buffer, Len(buffer))

            MsgBox (Left(buffer & " ", Result))

        Else

        End If

       IsError = (Result <> 0)

End Function

' read analog inputs

Sub     ButtonWriteAO()

       Dim nDevice As Long
       Dim handle As Long
       Dim davalue As Long
       Dim value As Single
       Dim nMode As Long
       Dim channel as long

       ' find first module

       nDevice = 0

       ' open module

       handle = QAPIExtOpenCard(USBAD8DAC2, nDevice)

       if ( handle = 0 ) then
          MsgBox "Unable to open any USBAD8DAC2 module!"
          exit sub
       end if

       ' write to all 2 dac channels

         nMode = MODE_BI_5V

         ' convert float voltage value to digital voltage and write it to dac output 0

         value = 1.0         ' voltage
              channel = 0            ' channel

         davalue = QAPIExtConvertVoltageToDW(handle, value, MODE_UNI_5V): If IsError() Then GoTo
              Call QAPIExtWriteDA(handle, channel, davalue, 0): If IsError() Then GoTo lab_exit

         ' convert float voltage value to digital voltage and write it to dac output 1

         value = 2.55        ' voltage
              channel = 1            ' channel

         davalue = QAPIExtConvertVoltageToDW(handle, value, MODE_UNI_5V): If IsError() Then GoTo
              Call QAPIExtWriteDA(handle, channel, davalue, 0): If IsError() Then GoTo lab_exit

terminate_sub:

       call QAPIExtCloseCard(handle)

end sub
```

5.3.3 TTL Ausgänge lesen und schreiben unter VB

```
Function IsError() As Integer

        Dim buffer As String * 256
        Dim Result As Long

        Result = QAPIGetLastError()

        If (Result <> 0) Then

            Result = QAPIGetLastErrorStringEx(buffer, Len(buffer))

            MsgBox (Left(buffer & " ", Result))

        Else

        End If

       IsError = (Result <> 0)

End Function

' read analog inputs

Sub     ButtonTTL()

       Dim nDevice As Long
       Dim handle As Long
       Dim nMode As Long
       Dim channel as long
       Dim DDR as long
       Dim value as long

       ' find first module

       nDevice = 0

       ' Step 1: Open the module

       handle = QAPIExtOpenCard(USBAD8DAC2, nDevice)

       if ( handle = 0 ) then
          MsgBox "Unable to open any USBAD8DAC2 module!"
          exit sub
       end if

 ' Step 2: Set the data direction of the lines
 '
 ' The TTL lines of the USBAD8DAC2 module can be grouped in blocks of eight lines.
 '
 ' Bit 0 ( 1 ):    = 1 Select output as data direction for the channels 0...7
 ' Bit 1 ( 2 ):    = 1 Select output as data direction for the channels 8...15
 ' Bit 2 ( 4 ):    = 1 Select output as data direction for the channels 16...23
 '
 ' If the corresponding bit is cleared the lines are set to input mode which is the default mode.
 '
 '     Value written to the DDR             Channels 0...7 Channels 8...15       Channels 16...23
 '     0                                    Inputs                Inputs                Inputs
 '     1                                    Outputs        Inputs                Inputs
 '     2                                    Inputs                Outputs        Inputs
 '     3                                    Outputs        Outputs        Inputs
 '     4                                    Inputs                Inputs                Outputs
 '     5                                    Outputs        Inputs                Outputs
 '     6                                    Inputs                Outputs        Outputs
 '     7                                    Outputs        Outputs        Outputs
 '
 ' Sample:
 '
 ' To set the channels 0...7 and channels 16...23  to output mode and the channels 8...15 to
   input mode
 ' you have to add the decimal values 1 + 4 = 5 and write this value to the DDR
 '


       DDR = 1 + 4

            Call QAPIExtSpecial(handle, JOB_WRITE_DDR, DDR, 0): If IsError() Then GoTo error_exit
```

```
'   Reading and writing to the lines can be done with the eight different commands QAPIExtReadDI1,
    QAPIExtReadDI8
'   QAPIExtReadDI16, QAPIExtReadDI32, QAPIExtWriteDO1, QAPIExtWriteDO8, QAPIExtWriteDO16 and
    QAPIExtWriteDO32.
'
'   The definitions of the functions are as following:
'
'  lines = QAPIExtReadDI1(handle, channel, mode)      ' mode=1 -> don't latch inputs ( only if supported by hardware )
'
'  lines = QAPIExtReadDI8(handle, channel, mode)      ' mode = 1 -> don't latch inputs ( only if supported by hardware )
'
'  lines = QAPIExtReadDI16(handle, channel, mode)    ' mode = 1 -> don't latch inputs ( only if supported by hardware )
'
'  lines = QAPIExtReadDI32(handle, channel, mode)    ' mode = 1 -> don't latch inputs ( only if supported by hardware )
'
'  call QAPIExtWriteDO1(handle, channel, value, mode)     ' mode = 1 -> don't latch outputs ( only if supported by hardware )
'
'  call QAPIExtWriteDO8(handle, channel, value, mode)     ' mode = 1 -> don't latch outputs ( only if supported by hardware )
'
'  call QAPIExtWriteDO16(handle, channel, value, mode)   ' mode = 1 -> don't latch outputs ( only if supported by hardware )
'
'  call QAPIExtWriteDO32(handle, channel, value, mode)   ' mode = 1 -> don't latch outputs ( only if supported by hardware )
'


      ' Step 3: Reading inputs
      ' read the a block of lines from channel 8...15
      ' 0 = channels 0...7    1= channels 8...15    2 = channels 16...23
      '

      channel = 1

      lines = QAPIExtReadDI8(handle, channel, 1)    ' mode = 1 -> don't latch inputs

      ' lines now holds the state of of the channels 8...15

      if ( lines and 1 ) then
          debug.print  "channel 8 is high"
      else
          debug.print  "channel 8 is low"
      end if

      if ( lines and 2 ) then
          debug.print  "channel 9 is high"
      else
          debug.print  "channel 9 is low"
      end if

      if ( lines and 4 ) then
          debug.print  "channel 10 is high"
      else
          debug.print  "channel 10 is low"
      end if

      if ( lines and 8 ) then
          debug.print  "channel 11 is high"
      else
          debug.print  "channel 11 is low"
      end if

      if ( lines and 16 ) then
          debug.print  "channel 12 is high"
      else
          debug.print  "channel 12 is low"
      end if

      if ( lines and 32 ) then
          debug.print  "channel 13 is high"
      else
          debug.print  "channel 13 is low"
      end if

      if ( lines and 64 ) then
          debug.print  "channel 14 is high"
      else
          debug.print  "channel 14 is low"
      end if

      if ( lines and 128 ) then
          debug.print  "channel 15 is high"
      else
          debug.print  "channel 15 is low"
      end if
```

```
       ' Step 4: Writing the outputs
       '
       ' 0 = channels 0...7    1= channels 8...15    2 = channels 16...23
       '
       ' set the lines channel 0, channel 1, channel 2 and channel 7 to high
       '
       '  channel 0:          1
       '  channel 1:          2
       '  channel 2:          4
       '  channel 3:          8
       '  channel 4:          16
       '  channel 5:          32
       '  channel 6:          64
       '  channel 7:          128
       '
       channel = 0

       value = 1 + 2 + 4 + 128

       call QAPIExtWriteDO8(handle, channel, value, 0)

       '
       ' 0 = channels 0...7    1= channels 8...15    2 = channels 16...23
       '
       ' set the lines channel 19 and channel 20 to high
       '
       '  channel 16:         1
       '  channel 17:         2
       '  channel 18:         4
       '  channel 19:         8
       '  channel 20:         16
       '  channel 21:         32
       '  channel 22:         64
       '  channel 23:         128
       '
       channel = 2

       value = 8 + 16

       call QAPIExtWriteDO8(handle, channel, value, 0)

terminate_sub:

       call QAPIExtCloseCard(handle)

end sub
```

5.3.4 Counter 0 und Counter 1 lesen unter VB

```
Function ShowError() As Integer

        Dim buffer As String * 256
        Dim Result As Long

        Result = QAPIGetLastError()

        If (Result <> 0) Then

            Result = QAPIGetLastErrorStringEx(buffer, Len(buffer))

            MsgBox (Left(buffer & " ", Result))

        Else

        End If

       IsError = (Result <> 0)

End Function

' read the counter

Sub     ButtonCounter()

       Dim nDevice As Long
       Dim handle As Long
       Dim nMode As Long
       Dim channel as long
       Dim DDR as long
       Dim value as long

       ' find first module

       nDevice = 0

       ' Step 1: Open the module

       handle = QAPIExtOpenCard(USBAD8DAC2, nDevice)

       if ( handle = 0 ) then
          MsgBox "Unable to open any USBAD8DAC2 module!"
          exit sub
       end if

       '
       '  Step 2: Initialize the counter
       '

       bEnableHWGate = FALSE

       ' Enable Counter 0 ( D-Sub Pin 17 )

       if ( bEnableHWGate ) then

           '
           ' Counting will only take in place if there is high level on the D-Sub Pin 17 line
           '

           Call QAPIExtSetupCounter(handle, 0, MODE_ENABLE_COUNTER_HW_GATE, 0): If ShowError() Then
          GoTo error_exit

       else

           '
           ' Enable counter
           '

          Call QAPIExtSetupCounter(handle, 0, MODE_DEFAULT, 0): If ShowError() Then GoTo error_exit

       end if

       '
       '  Step 3: Reset the counter to zero
       '
```

```
       ' Reset Counter 0

       Call QAPIExtResetCounter(handle, 0, 0, 0): If ShowError() Then GoTo error_exit

       ' Reset Counter 1

            Call QAPIExtResetCounter(handle, 1, 0, 0): If ShowError() Then GoTo error_exit

       ' wait one second

       Sleep(1000)

       '
       '  Step 4: Read the counter ( connect a signal to the lines D-Sub Pin 17-> Channel 0,
          D-Sub Pin 16 -> Channel 1 and  D-Sub Pin 18 -> HW Gate Channel 0
       '

       ' There are two possibilities for reading a counter:
       '
       ' 1.) non-destructive,  no autoreset, mode parameter MODE_DEFAULT
       '
       ' 2.) destructive reading, after reading the counter will be reaset to zero , mode parameter
            MODE_RESET_COUNTER_ON_READ
       '

       for i% = 0 to 10

              ' read the counter 0,  after reading always check  for an error with QAPIGetLastError
              () ,  a common error can be ERROR_QLIB_COUNTER_OVERFLOW

              channel = 0

              counter = QAPIExtReadCounter(handle, channel, MODE_RESET_COUNTER_ON_READ, 0):
              If ShowError() Then GoTo error_exit

              debug.print "Counter 0: " & counter


              ' read the counter 0,  after reading  always check for an error with QAPIGetLastError
              () ,  a common error can be ERROR_QLIB_COUNTER_OVERFLOW

              channel = 1

              counter = QAPIExtReadCounter(handle, channel, MODE_DEFAULT, 0): If ShowError()
              Then GoTo error_exit

              debug.print "Counter 1: " & counter

              ' wait one second

              Sleep(1000)

       next i%


terminate_sub:

       call QAPIExtCloseCard(handle)

end sub
```

6. Anhang

6.1 Frequently asked questions (FAQ)

6.1.1 Allgemeine Informationen

Kann ich Probleme bekommen wenn ich Netzwerkkarten, Soundkarten oder sonstige Erweiterungskarten in meinem PC habe?

Ja, es ist abhängig davon auf welchen I/O-Adressen die QUANCOM-Karte und die anderen liegen. Es können Resourcenkonflikte auftreten, wenn mehrere Geräte die selbe I/O-Adresse benutzen. Entweder ändern Sie die Adresse der QUANCOM-Karte (Kapitel Leitfaden zur Schnellkonfiguration Hard- und Software) oder die Adresse der anderen Komponente.

Welchen Zweck erfüllt das Programm PCIInfo?

Das Programm PCIINFO zeigt die I/O-Adressen von allen PCI-Karten an, die sich im Computer befinden.

Wofür brauche ich das Programm PCISETIO?

Das Programm erlaubt Ihnen ein manuelles Setzen der I/O- Adresse einer QUANCOM PCI Karte.

Wofür brauche ich das Programm PCIGETIO?

Das Programm PCIGETIO liest die Basis I/O-Adresse der QUANCOM PCI Karte aus dem Konfigurationsspeicher des PCI-Decoders aus.

6.1.2 Probleme mit Karten unter Windows 98/95 und Windows 2000/NT

Warum ist in der Systemsteuerung der Karten Dialog QLIB leer?

- Es ist keine QUANCOM PCI Karte im System.

Nach der Installation kommt die Meldung ”QLIBNDRV.SYS nicht gefunden ” oder ”QLIBNDRV.VXD nicht gefunden”- Was kann ich machen?

- Überprüfen Sie, dass die QLIB korrekt installiert ist. Für weitere Informationen rund um die Installation der QLIB sehen Sie im QLIB-Handbuch nach, das auf der CD enthalten ist.

Nach der Installation kommt die Meldung ”Direct-IO interface cannot be initialized qmulti32.dll **could not be initialized” - Was kann ich machen?**

- Überprüfen Sie, ob die QLIB korrekt installiert ist. Für weitere Informationen rund um die Installation der QLIB sehen Sie im QLIB-Handbuch nach, das auf der CD enthalten ist.

Warum gibt QAPIExtOpenCard (...) den Wert 0 zurück, obwohl die Karte installiert ist?

- Überprüfen Sie, ob die QLIB korrekt installiert ist. Für weitere Informationen rund um die Installation der QLIB sehen Sie im QLIB-Handbuch nach, das auf der CD enthalten ist.
- Die Karte ist nicht richtig konfiguriert. (”Systemsteuerung” => QLIB, Kapitel ” QLIB” Handbuch auf der Installations CD)

Warum bekomme ich die Meldung "Driver QLIBNDRV.SYS oder ”Driver QLIBNDRV.VXD konnte nicht geladen werden?”

- Überprüfen Sie, ob die QLIB korrekt installiert ist. Für weitere Informationen rund um die Installation der QLIB sehen Sie im QLIB-Handbuch nach, welches auf der CD enthalten ist.
- Die Treiber für die QUANCOM-Karte ist nicht geladen. (Systemsteuerung => System)

Windows 2000/NT: Kann die QLIB nur mit Administratorenrechten installiert werden?

- Ja, installieren Sie die QLIB nur mit Administratorenrechten.

Windows 2000/NT: Warum bekomme ich die Meldung ”Treiber konnte nicht geladen werden” während der Installation?

- Die Installation wurde ohne Administratorenrechte ausgeführt.
- Die QLIB-Software ist auf einem Netzlaufwerk installiert worden. Installieren Sie die QLIB immer auf der lokalen Festplatte.

Windows 2000/NT: Wie kann ich den QLIBNDRV.SYS Treiber manuell installieren?

Wenn die QLIB-Installation fehlschlägt, kann es nötig sein, dass Sie den Treiber manuell installieren.

- Suchen Sie auf der CD in dem Verzeichnis ”Tools” die Datei **instdrv.exe.** Mit diesem Tool können Sie den Treiber manuell installieren.
- Führen Sie das Programm folgendermaßen aus:

instdrv qlibndrv d:\directory\qlibndrv.sys

Wechseln Sie dazu in das Verzeichnis, in der sich die Datei ”qlibndrv.sys” befindet.

- Wechseln Sie unter Systemsteuerung -> Dienste die Startart auf ” Automatisch” (für den Dienst QLIBNDRV). Starten Sie jetzt Ihren Computer neu.

Warum muß ich den Treiber nach jedem Neustart wieder starten?

Die Startart des Treibers steht auf ”Manuell”. Wenn Sie die Einstellungen ändern möchten, wählen Sie die Startart ”Automatisch” und starten Sie Ihr System neu.

6.2 Kunden Support und Hilfe

Sie benötigen Hilfe?

Wenn Sie nicht wissen was Sie während einer Installation tun müssen, oder wie die Karte in Betrieb genommen wird, lesen Sie bitte dieses Handbuch.

! Tip !

Im Kapitel ”Frequently asked questions” (Häufig gestellte Fragen) sind einige Antworten auf häufig gestellte Fragen. Sie können Ihnen bei der Problemlösung behilflich sein. Auf der QUANCOM Installations CD finden Sie in Textform die Datei README.TXT, welche alle wichtigen Änderungen beinhaltet.

! Wichtig !

Wenn Sie weitere Fragen haben, kontaktieren Sie unser Support-Team. Für diesen Fall halten Sie bitte folgende Informationen bereit:

- Genauer Karten-Typ
- Version der Treiber
- Version der QLIB
- Betriebssystem, Hardware-Ausstattung und Bussystem
- Name und Version von dem Programm, welches den Fehler ausgibt
- Eine genaue Fehlerbeschreibung (versuchen Sie den Fehler zu wiederholen, um diesen besser beschreiben zu können)

Wen kann ich erreichten?

Die QUANCOM Internet Webseite
www.quancom.de

Per Fax
+49 22 36 / 89 92 - 49

Per E-Mail:
support@quancom.de

Adresse:
QUANCOM INFORMATIONSSYSTEME GmbH
In der Flecht 14
50389 Wesseling

Wenn Sie Hilfe brauchen, erreichen Sie uns unter:
QUANCOM Hotline Deutschland
0 22 36 / 89 92 - 20

Montags - Donnerstag
von 9:00 bis 18:00
Freitags
von 9:00 bis 17:00

Aktuelle Treiber

Auf unserer Internetseite http://www.quancom.de können sie immer die neusten Treiber Versionen und Updates finden. Zudem finden Sie ebenfalls viele andere Informationen und die ”Frequently asked questions (FAQ’s)”. Bevor Sie uns kontaktieren, überprüfen Sie ob die neueste Version der QUANCOM Software installiert ist.

Reparatur

Wenn Sie nicht genau wissen, ob die QUANCOM Karte defekt ist, rufen Sie unsere QUANCOM Hotline an:

Tel.: **+49 22 36 / 89 92 – 20**

Bevor Sie uns die Karte zur Reparatur schicken, rufen Sie unsere Hotline an:

Tel.: **+49 22 36 / 89 92 – 20**

Wenn Sie uns die Karte zurückschicken, legen Sie diese bitte in die Originalverpackung oder eine adäquate Verpackung, um einen Transportschaden zu verhindern. Zusätzlich bitten wir Sie, uns eine Kopie der Originalrechnung mitzuschicken.

6.3 Technisches Support Formular

Wenn Sie einen Internetzugang haben, öffnen Sie folgende URL in Ihrem Browser: http://www.quancom.de/quancom/qshop.nsf/techniksupport?OpenForm&deu
Füllen Sie das Formular komplett aus bevor Sie sich an QUANCOM Informationssysteme GmbH wenden. Wenn Sie andere QUANCOM Hardware oder Software nutzen, fügen Sie das bitte dem Formular hinzu.

Name: _____
Firma: _____
Adresse: _____
Telefon: _____
Fax: _____
Computer / Prozessor: _____
Betriebssystem: _____
Grafikkarte: _____
Maus: _____
QUANCOM Karte _____
Andere installierte Karten: _____
Festplatte (Kapazität, frei): _____
Das Problem ist: _____
Auflistung der Fehlermeldung: _____
_____
_____

Folgende Schritte führen zur Wiederholung des Problems:

_____
_____
_____
_____
_____

6.4 Dokumentations Formular

QUANCOM Informationssysteme GmbH möchte Ihren Kommentar zum Produkt und zu der über diese Dokumentation oder eines Produktes. Diese Informationen helfen uns unsere Qualität zu verbessern.

Titel: USBAD8DAC2 USBAD8DAC2/14 USBTTL24

Erstellungsdatum: 07.03.2007

Nehmen Sie Stellung zur Kompetenz, Übersichtlichkeit und Inhalt dieses Handbuches. Wenn Sie Fehler im Handbuch entdecken notieren Sie sich bitte die Seitenzahl.

Vielen Dank für Ihre Hilfe.

Name: _____

Firma: _____

Adresse: _____

Telefon: _____

Fax: _____

Kommentar: _____

_____

_____

_____

_____

Email an: **support@quancom.de**

Fax an: **+49 2236 89 92 49**

Adresse: **QUANCOM Informationssysteme GmbH**
In der Flecht 14
50389 Wesseling

6.5 Hardware und Software Konfigurationsformular

Dieses Formular hilft Ihnen die Einstellungen der Hardware und Software aufzulisten. Füllen Sie das Formular komplett aus bevor Sie sich an QUANCOM Informationssysteme GmbH wenden und nutzen Sie das Formular
ebenfalls um die aktuelle Konfiguration nachzuschlagen.

- **QUANCOM Produkt:**

Name / Name der Karte _____
Interrupt Level _____
DMA Kanal _____
Basis I/O Adresse _____
Betriebssystem _____

- **Andere Informationen**

Computer Model _____
Prozessor _____
Taktfrequenz _____
Grafikkarte _____
Betriebssystem _____
Programmiersprache _____
Programmiersprachen-Version _____

- **Andere Karten im System**

Basis I/O-Adresse anderer Karten _____
DMA Kanäle anderer Karten _____
Interrupt Level anderer Karten _____

6.6 Warenzeichen

Linux ist ein eingetragenes Warenzeichen von Linus Torvalds.
MS, MS-DOS, Microsoft, Visual Basic, Windows, Windows Vista/XP/2000/NT/ME/98/95 sind eingetragene Warenzeichen von Microsoft Corporation.
XT und PS/2 sind Warenzeichen und IBM, OS/2 und AT sind eingetragene Warenzeichen der International Business Machines Corporation.
Intel, Pentium ist ein eingetragenes Warenzeichen von Intel Corporation.
USB ist ein eingetragenes Warenzeichen von USB Implementers Forum Inc.
JAVA ist ein eingetragenes Warenzeichen von Sun Microsystems.
DELPHI und Pascal sind ein eingetragene Warenzeichen von Borland Corporation.
PCI ist ein eingetragenes Warenzeichen von PCI Special Interest Group.
PCI Express ist ein eingetragenes Warenzeichen der PCI-SIG.
Nationalinstruments, LABVIEW ist ein eingetragenes Warenzeichen von Nationalinstruments Corporation.
Agilent VEE ist ein eingetragenes Warenzeichen von Agilent Technologies.
Ethernet ist ein eingetragenes Warenzeichen der Xerox Corporation.

Bei anderen Produkt- und Firmennamen, die in dieser Anleitung erwähnt werden, könnte es sich um Marken ihrer jeweiligen Eigentümer handeln.