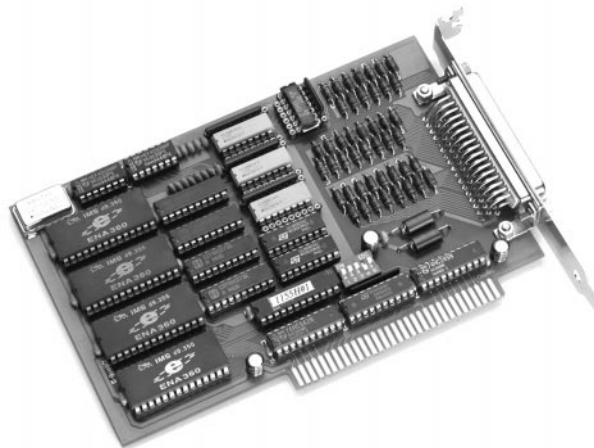
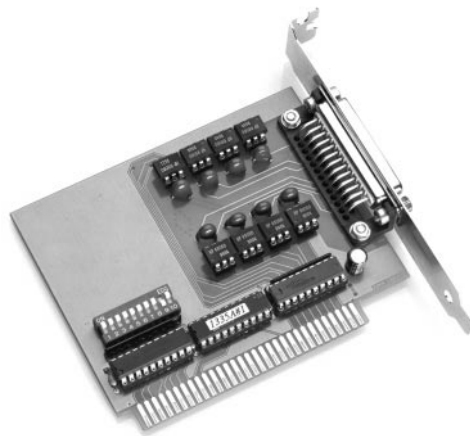

Universaltreiber für ERMA PC-Karten

für Windows NT 4.0

Bedienungsanleitung



ERMA

Electronic GmbH

Warenzeichen

Turbo Pascal, Delphi sind eingetragene Warenzeichen der Borland International, INC.

MS-DOS, Windows, Visual Basic, Visual C++ sind eingetragene Warenzeichen der Microsoft Corporation.

IBM, PC XT/AT, OS/2 sind eingetragene Warenzeichen der International Business Machines Corporation.

Alle ansonsten im Text genannten und abgebildeten Warenzeichen sind Warenzeichen der jeweiligen Inhaber und werden als geschützt anerkannt.

INHALTSVERZEICHNIS

1. Allgemeines	3
2. Installation	3
3. Hardwarekonfiguration	4
4. Einzustellende Parameter für ERMA PC-Karten	5
5. Unterstützung mehrerer PC-Karten	5
6. Programmierung	6
6.1. Programmierung unter Delphi2	7
6.2. Programmierung unter Visual C++ 5.0	7
6.3. Programmierung unter Visual Basic 5.0	8
7. Deinstallation des Treibers	8
8. Notizen	9

1. Allgemeines

Der ERMA-Universaltreiber für Windows NT stellt dem Programmierer eine einfache Möglichkeit zur Verfügung, um ERMA PC-Karten unter Windows NT anzusprechen. Es sind lediglich 8-Bit-Ein- und 8-Bit-Ausgabefunktionen vorhanden. Interrupts oder gar DMA werden hierbei nicht unterstützt.

Durch die Ein-Ausgabefunktionen können die PC-Karten wie unter DOS angesprochen werden.

2. Installation

Die Installation des Universaltreibers besteht darin, daß der Treiber und die DLL in die entsprechenden Systemverzeichnisse kopiert werden und einige Einträge in die Registry erfolgen. Die Unterstützung für die Programmiersprachen verbleibt auf der Diskette und muß bei Bedarf in die Projektverzeichnisse kopiert werden.

- Um die Installation zu starten müssen folgende Schritte ausgeführt werden:
- Im Startmenü den Punkt Ausführen wählen
- Im Ausführen-Dialog x:\Setup eingeben. (x ist durch den Buchstaben des verwendeten Diskettenlaufwerkes zu ersetzen)
- Nach Betätigung des OK-Schalters wird die Installation durchgeführt.

Nach Beendigung der Installation muß der Rechner neu gestartet werden, um den Treiber zu aktivieren.



Hinweis: Um den Treiber installieren zu können, müssen Sie Administratorrechte auf dem Rechner haben!

3. Hardwarekonfiguration

Der Treiber benötigt für seine Arbeit zwei Parameter aus der Registry. Der Parameter **IoPortAddress** definiert die Basisadresse der Karte und der Parameter **IoPortCount** die Anzahl der verwendeten Adressen. Die Standardeinstellung ist für IoPortAddress **0x100** und für IoPortCount **0x4**. Dies sind die benötigten Einstellungen für eine IO 1188 PC-Karte.

Um diese Parameter zu ändern muß die Registry geändert werden. Die Parameter sind unter **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\ERMAUNI\Parameters** zu finden. Das Ändern geschieht wie folgt:

- Im Startmenü den Punkt **Ausführen** anwählen.
- Im Ausführen-Dialog **REGEDT32** eingeben und den OK-Schalter betätigen.
- Nach dem Starten des Registrierungseditors wählen Sie das Fenster **HKEY_LOCAL_MACHINE** an.
- Öffnen Sie den SYSTEM-Teilbaum durch doppelklicken auf **SYSTEM**.
- In diesem Teilbaum doppelklicken Sie auf **CurrentControlSet**.
- In diesem Teilbaum doppelklicken Sie auf **Services**.
- In diesem Teilbaum doppelklicken Sie auf **ERMAUNI**.
- Nach einem Doppelklick auf **Parameters** sehen Sie im rechten Fenster die beiden Parameter.

Durch einen Doppelklick auf den jeweiligen Parameter öffnet sich ein Editfenster in dem Sie die Parameter Ihren Bedürfnissen anpassen können.



Hinweis: Nach der Änderung der Parameter muß der Rechner neu gestartet werden!

4. **Einzustellende Parameter für ERMA PC-Karten**

Für alle PC-Karten gilt, daß die Basisadresse frei einstellbar ist. Die Anzahl der benötigten IO-Adressen ist jedoch immer verschieden und muß angepaßt werden. Es ist zu beachten, daß der Universaltreiber keine Interrupts unterstützt!

PC-Karte	IoPortCount
AD 1199	0x2
AD 1285 ¹⁾	0x4
DA 1250 / DA 1256 / DA 1326	0x2
FR 1215	0x10
IO 1188	0x4
IO 1213	0x8
IO 1283 ¹⁾	0x2
RL 1335	0x1
RS1295 ²⁾	1 bis 4 mal 0x8
SM 1251	0x8
SSI 1276	0x8
TTY 1316 ²⁾	0x8
TTY 1294 ²⁾	1 bis 4 mal 0x8
ZIB 1155 / ZIB 1155/OPTO	0x20
ZIB 1177	0x2

1) Für diese Karte werden die 16-Bit-Zugriffe benötigt. Der Interrupt wird nicht unterstützt!

2) Für die Schnittstellenkarten sollte der COM-Treiber von Windows NT benutzt werden. Die eingestellten Parameter der Schnittstellen müssen in der Systemsteuerung über die Anschlüsse eingerichtet werden. Der Vorteil ist, daß man komfortabel mit den COMM-Funktionen arbeiten kann.

5. **Unterstützung mehrerer PC-Karten**

Um mehrere PC-Karten zu unterstützen müssen die Adressbereiche der Karten so eingestellt werden, daß die Adressbereiche direkt hintereinander liegen.

z.B.:

AD 1199 auf 0x100 mit 0x2 Adressen

IO 1188 auf 0x104 mit 0x4 Adressen



Hinweis: 0x102 bei der IO 1188 durch die DIP-Schalter nicht möglich. Deshalb wird die nächstmögliche Adresse gewählt. Die Adressen 0x102 und 0x103 sind zwar ansprechbar, aber haben praktisch keine Funktion.

Somit werden folgende Parameter gesetzt :

- IoPortAddress = 0x100
- IoPortCount = 0x8



Hinweis: Es ist darauf zu achten, daß keine andere Hardware diesen Bereich oder Teile davon beansprucht!

6. Programmierung

Die Funktionen des Treibers sind durch eine DLL gekapselt. Dadurch entfällt der direkte Aufruf des Treibers.

In der DLL sind folgende Funktionen enthalten:

ErmaUniInitDriver

Diese Funktion muß am Anfang des Programmes einmal aufgerufen werden. Dadurch wird die Verbindung zwischen DLL und Treiber hergestellt.

ErmaUniDeInitDriver

Diese Funktion muß am Ende des Programmes einmal aufgerufen werden. Dadurch wird die Verbindung zwischen DLL und Treiber wieder gelöst.

ErmaUnitWriteByte

Durch diese Funktion wird ein Byte auf eine angegebene relative Adresse ausgegeben.

ErmaUniReadByte

Durch diese Funktion wird ein Byte von einer relativen Adresse eingelesen.

ErmaUnitWriteWord

Durch diese Funktion wird ein Wort auf eine angegebene relative Adresse ausgegeben.

ErmaUniReadWord

Durch diese Funktion wird ein Wort von einer relativen Adresse eingelesen.

Alle Adressen müssen als relative Adressen zur Basisadresse (IoPortAddress) erfolgen. Ist IoPortAddress z.B. auf 0x140 gesetzt, greift man wie folgt auf die Adressen zu:

- 0x00 entspricht 0x140
- 0x01 entspricht 0x141
- 0x02 entspricht 0x142

Die Funktionen haben in der DLL folgende Namen:

```
_ErmaUniInitDriver@0  
_ErmaUniDeInitDriver@0  
_ErmaUniReadByte@4  
_ErmaUniWriteByte@8  
_ErmaUniReadWord@4  
_ErmaUniWriteWord@8
```

Diese Namen werden nur benötigt, wenn von anderen Programmiersprachen als den hier vorgeschlagenen aus auf die DLL zugegriffen werden soll.

6.1. Programmierung unter Delphi2

Die Programmierung unter Delphi gestaltet sich recht einfach:

In das Projekt ist die auf der Diskette befindliche Unit **ERMAUNI.PAS** einzubinden. In der Anwendung wird diese Unit dann nur noch in der USES-Anweisung eingetragen und schon können die Funktionen genutzt werden.

Die Funktionen sind folgendermaßen definiert:

```
- procedure ErmaUniInitDriver;  
- procedure ErmaUniDeInitDriver;  
- function ErmaUniReadByte(Adresse: Word): Byte;  
- procedure ErmaUniWriteByte(Adresse: Word; Data: Byte);  
- function ErmaUniReadWord(Adresse: Word): Word;  
- procedure ErmaUniWriteWord(Adresse: Word; Data: Word);
```

6.2. Programmierung unter Visual C++ 5.0

Für die Einbindung in Visual C++ sind zwei Schritte notwendig:

- Einbinden der Datei ERMAUNI.H in das Projekt und in jede Datei die diese Funktionen benötigt.
- Unter **Projekt-Settings-Link** muß unter **Object/Library modules** die Datei **ERMAUNI.LIB** eingetragen werden.

Die Funktionen sind in VC++ wie folgt definiert:

```
void ErmaUniInitDriver(void);  
void ErmaUniDeInitDriver(void);  
UCHAR ErmaUniReadByte(ULONG Address);  
void ErmaUniWriteByte(ULONG Address, UCHAR Data);  
USHORT ErmaUniReadWord(ULONG Address);  
void ErmaUniWriteWord(ULONG Address, USHORT Data);
```

6.3. **Programmierung unter Visual Basic 5.0**

Die DLL-Funktionen werden durch folgende Deklarationen in Visual Basic 5.0 übernommen:

```
Public Declare Sub InitDriver Lib "ermauni.dll" Alias "_ErmaUniInitDriver@0" ()
Public Declare Sub DeInitDriver Lib "ermauni.dll" Alias
    "_ErmaUniDeInitDriver@0" ()
Public Declare Function ReadBytePort Lib "ermauni.dll" Alias
    "_ErmaUniReadByte@4" (ByVal PortAdr As Long) As Byte
Public Declare Sub WriteBytePort Lib "ermauni.dll" Alias
    "_ErmaUniWriteByte@8" (ByVal PortAdr As Long,
        ByVal Value As Byte)
Public Declare Function ReadWordPort Lib "ermauni.dll" Alias
    "_ErmaUniReadWord@4" (ByVal PortAdr As Long) As Word
Public Declare Sub WriteWordPort Lib "ermauni.dll" Alias
    "_ErmaUniWriteWord@8" (ByVal PortAdr As Long,
        ByVal Value As Word)
```

7. **Deinstallation des Treibers**

Um den Treiber zu deinstallieren müssen die DLL und der Treiber gelöscht werden sowie die Einträge aus der Registry entfernt werden. Da dies vom Setup-Programm noch nicht unterstützt wird, muß dies von Hand geschehen.

Die Deinstallation wird wie folgt durchgeführt:

- Die Datei ERMAUNI.SYS im Verzeichnis WINNT\SYSTEM32\DRIVERS muß gelöscht werden.
- Die Datei ERMAUNI.DLL im Verzeichnis WINNT\SYSTEM32 muß gelöscht werden.
- Die Einträge aus der Registry werden mittels REGEDT32 entfernt. (Siehe Hardwarekonfiguration). Dabei wird der Teilbaum **ERMAUNI** angewählt und die Taste **ENTF**ern betätigt. Alle Unterpunkte werden dabei automatisch gelöscht.
- Nach einem Neustart des Rechners ist der Treiber entfernt.

8. Notizen

ERMA - Electronic GmbH
Max-Eyth-Straße 8
78194 Immendingen

Telefon (07462) 7381
Fax (07462) 7554
email erma-electronic@t-online.de
Web www.erma-electronic.de

ERMA
Electronic GmbH