

PCAN–Treiber für Linux

Copyright (C) 2002 Peak System–Technik GmbH

www.peak-system.com

linux@peak-system.com

und Klaus Hitschler

klaus.hitschler@gmx.de

Historie des Dokuments

Erster Entwurf	Hi	13.01.2002
Rettung nach Datenverlust	Hi	10.02.2002
Formatüberarbeitung, read/write Tabellen	Hi	20.02.2002
Typografische Fehler entfernt	Hi	21.02.2002

Inhaltsverzeichnis

Disclaimer.....	3
Eigenschaften des Treibers 'pcan.o'.....	3
Eigenschaften der Bibliothek 'libpcan.so'.....	3
Vorbemerkungen.....	4
Installation mit RPM.....	5
Installation des Quellpakets.....	5
Anpassung von 'modules.conf'.....	5
Installation des Treibers.....	6
Manuelle Installation.....	6
Manuelles Entpacken der Dateien.....	6
Manuelle Installation des Treibers.....	7
Manuelle Installation der shared library.....	8
Manuelle Installation der Header-Dateien.....	9
Halbautomatische Installation mit Übersetzung.....	9
Test mit Hilfe der Testprogramme.....	9
Das Wichtigste zu den Quellen.....	11
Übersetzung des Treibers.....	11
Anwendungsfälle	11
Voraussetzungen zur Übersetzung der Quellen.....	12
FAQs.....	12
Anhang.....	13

Disclaimer

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Eigenschaften des Treibers 'pcan.o'

Der Treiber „pcan.o“ unterstützt sowohl die PCAN-PCI, PCAN-ISA als auch die PCAN-Dongle Hardware. Es werden je nach Hardwaretyp (PCI, ISA, Dongle) gleichzeitig bis zu 8 Kanäle unterstützt. Dabei ist, mit der Ausnahme des PCAN-PCI, die Basisadresse und der verwendete Interrupt frei parametrierbar. Die werksseitigen Standardeinstellungen sind Vorgabe. Ein Anwendungsprogramm kann mit dem Treiber auf zwei verschiedene Arten kommunizieren:

1. Über eine direkte „read()“ und „write()“ Schnittstelle können ASCII formatierte Daten bereitgestellt oder akzeptiert werden. Diese Daten enthalten Informationen zu den empfangenen oder zu sendenden Telegrammen oder zur Initialisierung der Kanäle.
2. Über eine „ioctl()“ Schnittstelle. Über diese Schnittstelle können Anwendungsprogramme sowohl CAN-Telegramme empfangen als auch senden. Weiterhin ist es möglich Diagnose zum Zustand der Kanäle abzufragen und die Kanäle zu initialisieren.

Der Treiber kann gleichzeitig von mehreren Anwendungsprogrammen genutzt werden und es wird die gleichzeitige Nutzung mehrerer Interfaces und/oder CAN-Kanäle unterstützt. Wenn mehrere Anwendungsprogramme auf den gleichen CAN-Kanal zugreifen wird die Ein- oder Ausgabe auf Telegrammebene geteilt. (Beispiel: Mehrere lesende Anwendungsprogramme teilen sich zufällig die empfangenen Telegramme.)

Das Lesen oder Schreiben der Telegramme kann blockieren wenn keine Daten mehr im Empfangspuffer verfügbar sind oder der Schreibepuffer gefüllt ist. Diese Eigenschaft kann durch Öffnen des Pfads als „nicht blockierend“ ausgeschaltet werden.

Die „select“-Methode wird unterstützt.

Die Schnittstelle zum Treiber (Konstanten und Strukturen) beschreibt die C-Headerdatei „pcan.h“.

Eigenschaften der Bibliothek 'libpcan.so'

Die Bibliothek „libpcan“ stellt eine vereinfachte Schnittstelle zur Nutzung der

Eigenschaften des PCAN-Treibers bereit. Die Treiberaufrufe sind den Treiberaufrufen unter MS-Windows nachempfunden und erleichtern die Portierung entsprechender Applikationen. Hinzu kommen einige Aufrufe die sich besser an die Gegebenheiten von LINUX anpassen. Sie sind mit einem vorangestellten 'LINUX_..' gekennzeichnet. (Beispiel: HANDLE LINUX_CAN_Open(char *szDeviceName, int nFlags);)

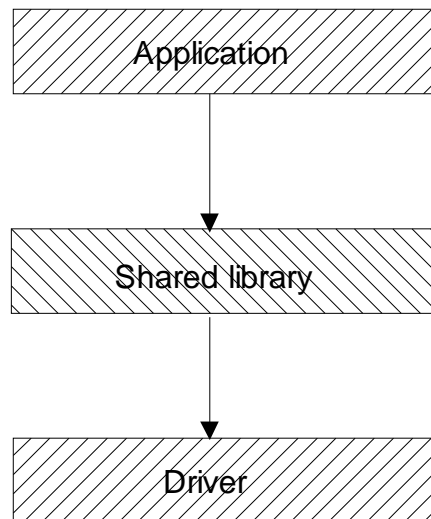


Bild 1. Aufrufhierarchie

Die Schnittstelle zur Bibliothek beschreibt die C/C++-Headerdatei „libpcan.h“. Bitte beachten sie, das „libpcan.h“ die Datei „pcan.h“ einbindet.

Vorbemerkungen

Bemerkung: Vor der Installation der Software muss die Hardware im oder am Rechner installiert sein (Ausnahme PCAN-Dongle, siehe weiter unten (6*)).

Diese Beschreibung beschreibt den Installationsvorgang an einem x86-LINUX-Rechner mit installiertem Kernel 2.4.x. und einem SuSE 7.3 System. Ausnahmen für den Kernel 2.2.x. und x86-RedHat Systeme werden ausdrücklich erwähnt.

Im Verlauf der Installation wird öfters eine Konsole und vielleicht auch ein Editor im „root“-Modus gebraucht. Um innerhalb einer Konsolen oder xterm-Sitzung sich als „root“ anzumelden, können sie das Kommando „su“ oder das in Konsole eingebaute Menu „Datei/Root Konsole“ verwenden.

Um einen Editor im „root“-Modus zu starten, rufen sie am Besten unter KDE das Menu „System/Terminals/Terminal (Systemverwaltungs Modus)“ auf. Dort aus der Kommandozeile heraus können sie dann den Editor „kwrite“ oder „kate“ oder „emacs“ aufrufen.

Bei einigen Distributionen umfasst der Standardpfad nicht den Ordner „/sbin“. Dann müssen einige der Installationskommandos mit vorangestellten „/sbin/“ aufgerufen werden.

Beispiele:

/sbin/modprobe ...	anstatt	modprobe ...
/sbin/insmod ...	anstatt	insmod ...
/sbin/rmmod ...	anstatt	rmmod ...
/sbin/modinfo ...	anstatt	modinfo ...

Installation mit RPM

Die Installation enthält insgesamt drei Dateien:

Installation.pdf	Beschreibung im PDF Format.
peak-linux-driver-20020221-1.36.src.rpm	Die RPM-Quell-Datei.
peak-linux-driver.1.36.tar.gz	Die gepackten tar-Quellen.

(Hinweis: Die Bezeichnungen für das Release bzw. der Version in den Dateinamen können sich ändern.)

Installation des Quellpakets

Rufen sie bitte in der Kommandozeile (als root) auf:

```
rpm --rebuild peak-linux-driver-20020221-1.36.src.rpm
```

vorausgesetzt die Datei „peak-linux-driver-20020221-1.36.src.rpm“ befindet sich in dem Aufrufverzeichnis.

Dieser Aufruf setzt die vorherige Installation des gcc-Compilers, der Kernel-Header und des make-Werkzeugs voraus.

Nach dem erfolgreichen Ablauf dieses „Rebuilds“ legt RPM in dem Verzeichnis

„/usr/src/packages/“ oder „/usr/src/redhat“

für SuSE bzw. RedHat die für diesen Rechner passende binär-Installation ab. Z.B.

/usr/src/redhat/RPMS/i386/peak-linux-driver-20000221-1.36.i386.rpm

Nun können sie mit

```
rpm --install peak-linux-driver-20000221-1.36.i386.rpm
```

die binäre Installation durchführen. Alternativ können sie jetzt auch die GUI-basierten Installer „kpackage“, unter KDE oder „gnorpm“, unter GNOME verwenden.

Anpassung von 'modules.conf'

Am Ende der Datei „/etc/modules.conf“ wird durch die Installation ein gekennzeichneteter Zusatz angefügt. Dort ist, falls der Treiber PCAN-ISA oder PCAN-Dongle Interfaces unterstützen soll, der jeweilige Typ und eventuell die vom Standard abweichende Basisadresse oder Interrupt anzugeben. (Siehe auch die Beschreibung zur manuellen Installation des Treibers.)

Installation des Treibers

Nach dem „Rebuild“ der Quellen und der darauffolgenden Binär-Installation auf ihrem Zielrechner reicht der Aufruf (als root) von

```
/sbin/modprobe pcan
```

um den Treiber und alle sonst notwendigen Module zu installieren. Sie müssen jetzt nur noch, je nach Anforderung, diesen Aufruf jedesmal beim Start ihres Rechners oder ihrer Anwendung durchführen.

Manuelle Installation

Manuelles Entpacken der Dateien

Entpacken sie die Datei „peak-linux-driver.x.y.tar.gz“ in ein beliebiges Verzeichnis unterhalb ihres home-Verzeichnisses aus. Beispiel:

```
tar -xzf peak-linux-driver.1.36.tar.gz
```

Sie erhalten dann innerhalb dieses Verzeichnisses folgenden Dateibaum:

peak-linux-driver/	
-- Makefile	Global Makefile
-- Documentation	
-- COPYING	GPL
-- Installation.pdf	Documentation in PDF
-- template.c	File templates
-- template.h	
-- template.make	
-- todo.txt	More to do.
-- driver	
-- Makefile	Makefile only for driver
-- pcan.h	Header for driver access
-- pcan.o	The driver
-- pcan_make_devices	Script to make device files
-- src	
-- pcan_common.h	Driver sources
-- pcan_dongle.c	
-- pcan_dongle.h	
-- pcan_fifo.c	
-- pcan_fifo.h	
-- pcan_fops.c	
-- pcan_fops.h	
-- pcan_isa.c	
-- pcan_isa.h	
-- pcan_main.c	
-- pcan_main.h	
-- pcan_parse.c	
-- pcan_parse.h	
-- pcan_pci.c	
-- pcan_pci.h	

```

| | |-- pcan_sja1000.c
| | '-- pcan_sja1000.h
| |-- test.txt
| '-- wstress
|-- lib
| |-- Makefile
| |-- libpcan.h
| |-- libpcan.so.0.0
| '-- src
|   |-- libpcan.c
|-- test
| |-- Makefile
| |-- receivetest
| |-- src
|   |-- common.c
|   |-- common.h
|   |-- parser.cpp
|   |-- parser.h
|   |-- receivetest.c
|   '-- transmitest.cpp
|-- transmit.txt
|-- transmitest

```

Makefile only for library
Header for library access
The library (version can change)
Library sources
Makfile only for test programs
Test program sources
test programs

Manuelle Installation des Treibers

Melden sie sich als „root“ in einer Konsole, z.B. Konsole oder xterm, an. „Gehen“ sie in das Verzeichnis „.../peak-linux-driver/driver“. Rufen sie

```
modprobe parport
```

auf. Dieser Befehl installiert das „Parport Subsystem“ unter LINUX und ist zur Unterstützung der PCAN-Dongle Einbindung notwendig (1*, 2*). Wenn der Befehl ohne Meldung beendet rufen sie bitte

```
insmod pcan.o type=xxx,yyy,... io=0x300,0x378,... irq=10,7,...
```

auf. Die Platzhalter „xxx,yyy,...“ stehen je für „isa“, „sp“ oder „epp“. Die Parameter für „io“ und „irq“ können weggelassen werden, wenn die Standard Einstellungen konfiguriert sind. PCI-Kanäle werden ohne spezielle Angabe gefunden und installiert, ISA und Parallel-Port gestützte Kanäle müssen angegeben werden.

Die Standardvorgaben für ISA und PC/104 Interfaces sind (io/irq): 0x300/10, 0x320/5

Die Standardvorgaben für den Dongle im SP/EPP Modus sind (io/irq): 0x378/7, 0x278/5

Eventuelle Fehlermeldungen können sie mit dem Befehl:

```
tail /var/log/messages
```

anschauen. Normalerweise sieht die Ausgabe ähnlich aus:

```

Jan 13 18:28:57 sylvia kernel: pcan: *Name: Release_20020113_a *
Jan 13 18:28:57 sylvia kernel: pcan: pci device minor 0 found
Jan 13 18:28:57 sylvia kernel: pcan: isa device minor 8 found (io=0x0300,irq=10)
Jan 13 18:28:57 sylvia kernel: pcan: epp-dongle device minor 24 prepared (io=0x0378,irq=7)

```

```
Jan 13 18:28:57 sylvia kernel: pcan: major 254.
```

Sie können die korrekte Installation überprüfen. Rufen sie

```
cat /proc/pcan
```

auf. Sie erhalten im fehlerfreien Fall einen ähnlichen Ausdruck:

```
*----- Peak-Systems CAN interfaces (www.peak-system.com) -----  
*----- *Name: Release_20020113_a * -----  
*----- 3 interfaces @ major 254 found -----  
*n typ ---base--- ir ---read--- ---write--- ---irqs--- ---error--- status  
0 pci 0xe700c000 10 0x00000000 0x00000000 0x00000000 0x00000000 0x0000  
8 isa 0x00000300 10 0x00000000 0x00000000 0x00000000 0x00000000 0x0000  
24 epp 0x00000378 7 0x00000000 0x00000000 0x00000000 0x00000000 0x0000
```

Der Treiber ist für die dynamische Vergabe der „major“ Gerätenummer konfiguriert (4*). Daher müssen nach jeder Installation des Treibers die Geräte-Dateien erneut erzeugt werden. Rufen sie hierzu bitte das Shell-Skript

```
./pcan_make_devices 2
```

auf. Dieses Skript installiert je 2 Geräte-Dateien für PCI, ISA und Dongle Devices im SP und EPP Modus (3*). Bitte prüfen sie das Ergebnis mit dem Befehl

```
ls -l /dev/pcan*
```

Nun können sie schon einen ersten Test der CAN-Übertragung durchführen. Verbinden sie einen CAN-Sender mit der Schnittstelle und senden sie mit der Einstellung 500 kbit/sec. Melden sie sich an einer Benutzer-Konsole an und rufen

```
cat /dev/pcan0 für die erste PCAN-PCI Schnittstelle  
cat /dev/pcan8 für die erste PCAN-ISA oder PCAN-PC/104 Schnittstelle  
cat /dev/pcan16 für die erste PCAN-Dongle Schnittstelle im SP-Modus  
cat /dev/pcan24 für die erste PCAN-Dongle Schnittstelle im EPP-Modus
```

auf. Es sollten die empfangenen Telegramme dargestellt werden (5*). Der Aufruf kann an mehreren Konsolen für mehrere Schnittstellen gleichzeitig erfolgen.

Es ist auch schon die Ausgabe von Daten möglich. Mit der Aufruf von z.B.

```
echo „m s 0x234 2 0x11 0x22“ > /dev/pcan0
```

wird ein CAN-Telegram mit „Standard-Frame“ dem Identifier 0x234 und 2 Datenbytes „0x11 und 0x22“ über die erste PCAN-PCI-Schnittstelle ausgegeben. Über die sogenannte „write-Schnittstelle“ des Treibers lässt sich auch die Bitrate der Schnittstelle einstellen.

Manuelle Installation der shared library

Wechseln sie bitte in das Verzeichnis „../peak-linux-driver/lib“ Melden sie sich bitte als „root“ an und rufen auf:

```
cp libpcan.so.0.0 /usr/lib/libpcan.so.0.0  
ln -sf /usr/lib/libpcan.so.0.0 /usr/lib/libpcan.so.0  
ln -sf /usr/lib/libpcan.so.0 /usr/lib/libpcan.so
```

Dieser Vorgang muss nur beim erstmaligen Installieren oder beim Update der

Bibliothek geschehen. Die Bibliothek muss zur Nutzung der Testprogramme installiert sein.

Manuelle Installation der Header-Dateien

Bitte kopieren sie die Header-Dateien wie folgt in das Verzeichnis „/usr/include“ und passen sie die Zugriffsrechte entsprechend an.

```
cp peak-linux-driver/driver/pcan.h /usr/include/pcan.h
chmod 644 /usr/include/pcan.h
cp peak-linux-driver/lib/libpcan.h /usr/include/libpcan.h
chmod 644 /usr/include/libpcan.h
```

Halbautomatische Installation mit Übersetzung

Mit wenigen Eingaben lässt sich auch eine halbautomatische Installation mit Übersetzung durchführen. Packen sie die tar.gz-Datei nach „/usr/src“ aus und starten sie die Übersetzung und Installation. Bitte beachten sie jedoch die Voraussetzungen für die Übersetzung der Quellen.

```
cp peak-linux-driver.1.31.tar.gz /usr/src/
cd /usr/src
tar -xzf peak-linux-driver.1.31.tar.gz
rm peak-linux-driver.1.31.tar.gz
cd peak-linux-driver
make clean
make
make install
```

Test mit Hilfe der Testprogramme

Zum ersten Test der Schnittstellen sind zwei Testprogramme beigelegt. Zum Ausführen der Programme wechseln sie bitte in das Verzeichnis „.../peak-linux-driver/test“. Rufen sie bitte „./receivetest --help“ auf. Sie erhalten folgende Antwort:

```
receivetest Version "Release_20020113_a" (www.peak-system.com)
----- Copyright (C) 2002 Peak System-Technik GmbH -----
receivetest comes with ABSOLUTELY NO WARRANTY. This is free
software and you are welcome to redistribute it under certain
conditions. For details see attached COPYING file.

receivetest - a small test program which receives and prints CAN messages.
usage: receivetest [-t=type] [-p=port [-i=irq]] [-b=BTR0BTR1] [-e] [-?]
options: -t - type of interface, e.g. 'pci', 'sp', 'epp' or 'isa' (default: pci).
        -p - port in hex notation if applicable, e.g. 0x378 (default: 1st port of type).
        -i - irq in dec notation if applicable, e.g. 7 (default: irq of 1st port).
        -b - BTR0BTR1 code in hex, e.g. 0x001C (default: 500 kbit).
        -e - accept extended frames. (default: standard frames)
        -? or --help - this help

receivetest: finished (0).
```

Das zugehörige CAN-Interface ist nach Standardvorgabe auf 500 kbit/sec eingestellt. Die Bitrate kann jedoch über einen Kommandozeilenparameter eingestellt werden.

Zur Benutzung eines ISA – oder Dongle-Interfaces mit der Standardeinstellung genügt folgende Angabe:

```
receivetest -t=isa oder receivetest -t=sp oder receivetest -t=epp
```

Als Ergebnis des Programms werden alle empfangenen Daten dieser Schnittstelle formatiert ausgegeben. Mit der Eingabe von Ctrl-C kann das Programm abgebrochen werden.

Zum Senden von Daten kann das Programm „transmitest“ verwendet werden. Bitte rufen sie „./transmitest --help“ auf. Sie erhalten folgende, zu receivetest ähnliche Ausgabe:

```
transmitest Version "Release_20020113_a" (www.peak-system.com)
----- Copyright (C) 2002 Peak System-Technik GmbH -----
transmitest comes with ABSOLUTELY NO WARRANTY. This is free
software and you are welcome to redistribute it under certain
conditions. For details see attached COPYING file.

transmitest - a small test program which sends CAN messages.
usage: transmitest filename [-t=type] [-p=port [-i=irq]] [-b=BTR0BTR1] [-e] [-?]
options: filename - mandatory name of message description file.
        -t - type of interface, e.g. 'pci', 'sp', 'epp' or 'isa' (default: pci).
        -p - port in hex notation if applicable, e.g. 0x378 (default: 1st port of type).
        -i - irq in dec notation if applicable, e.g. 7 (default: irq of 1st port).
        -b - BTR0BTR1 code in hex, e.g. 0x001C (default: 500 kbit).
        -e - accept extended frames. (default: standard frames)
        -? or --help - this help

transmitest: finished (0)
```

Dieses Programm verlangt ähnliche Parameter wie receivetest mit Ausnahme eines Dateinamens. In der Datei mit dieses Namens erwartet das Programm eine Liste von zu sendenden Telegrammbeschreibungen. Diese Telegramme werden dann vom Programm zyklisch, bis zum Abbruch durch Drücken von Ctrl-C, gesendet.

Die Syntax der Telegrammbeschreibungen ist:

Tag / Column #1	Description
m	Message string follows
r	RTR-Message string follows
i	Initialisation string follows (write only)
#	Comment follows (write only)

Beschreibung der ersten Spalte der Telegramme der read/write Schnittstelle.

Column	Description
2	s = Standard frame, e = Extended frame
3	Identifier (hex)
4	Length of telegram (dec)
5..12	Telegram bytes (hex)
13	Timestamp (dec, read only)
13	Comment (write only)

Beschreibung der folgenden Spalten der 'normalen' und RTR-Telegramme.

Column	Description
2	BTR0BTR1 Init data (hex, write only)
3	e = allow extended frames (write only)

Beschreibung des Initialisierungsstrings.

Beispiele:

```
# a comment
# a message, standard frame, id=0x123, 0 Databytes
m s 0x123 0
```

```

# a message, standard frame, id=0x123, 1 Databyte, Data
m s 0x123 1 0x11
# a message, standard frame, id=0x123, 1 Databyte, Data
m s 0x123 2 0x11 0x22
# a message, extended frame, id=0x123, 3 Databyte, Data
m e 0x123 3 0x11 0x22 0x33
# a RTR, standard frame, id=0x123, 0 Databyte, comment
r s 0x123 0 # a comment
# a RTR, extended frame, id=0x123, 0 Databyte, comment
r e 0x123 0 # a comment
# initialize
i 0x1234 e

```

Das Wichtigste zu den Quellen

Sämtliche Quellen des Treibers, der Bibliothek und der Testprogramme sind unterhalb des Dateibaums eingefügt. Besonders zu erwähnen sind zwei Dateien:

Die Datei „.../peak-linux-driver/lib/libpcan.h“ beschreibt die Schnittstelle zur dynamisch linkbaren Bibliothek libpcan.so. Diese Datei müssen sie einbinden, wenn ihre Applikation die dynamische Bibliothek nutzen will.

Die Datei „.../peak-linux-driver/driver/pcan.h“ beschreibt die Schnittstelle zum Treiber, sprich die Treiberaufrufe und Kommandokonstanten. Diese Datei müssen sie einbinden, wenn sie direkt mittels „open(), ioctl(), read() etc.“ auf den Treiber zugreifen wollen. Bitte beachten sie, dass libpcan.h die Datei pcan.h einbindet.

Übersetzung des Treibers

Anwendungsfälle

In manchen Anwendungsfällen passt der installierte Treiber nicht zum konfigurierten Betriebssystem. Dies könnten folgende Fälle sein:

1. Der Treiber wird auf eine RedHat LINUX Plattform installiert. RedHat verwendet im Gegensatz zu SuSE standardmässig den „CONFIG_MODVERSIONS“ Modifizierer um die Installation unpassender Treiber zu verhindern. Ob ihr Betriebssystem mit „CONFIG_MODVERSIONS“ erzeugt wurde sehen sie am Einfachsten durch den Aufruf von „cat /proc/ksyms“. Sind hinter den Symbolen kryptische Zahlen-Buchstabenkombinationen dargestellt, dann wurde ihr System mit „CONFIG_MODVERSIONS“ erzeugt.

Hier genügt ein einfaches Neuübersetzen auf ihrem System:

```

cd ~/peak-linux-driver/driver
make clean
make

```

2. Ihr Betriebssystem unterstützt das PARPORT_SUBSYSTEM nicht oder sie stellen eine, dem Betriebssystem unbekannte parallele Schnittstelle bereit um an dieser Schnittstelle den PCAN-Dongle zu betreiben.

Hier müssen sie mit dem Schalter „PAR=NO_PARPORT_SUBSYSTEM“ übersetzen:

```

cd ~/peak-linux-driver/driver

```

```
make clean
make PAR=NO_PARPORT_SUBSYSTEM
```

3. Sie bekommen Laufzeitfehler und wollen mehr Diagnoseinformationen in der Datei „/var/log/messages“ sehen. Hier müssen sie mit dem Schalter „DBG=DEBUG“ übersetzen:

```
cd ~/peak-linux-driver/driver
make clean
make DBG=DEBUG
```

Selbstverständlich sind auch Kombinationen dieser Schalter möglich.

Voraussetzungen zur Übersetzung der Quellen

Vor dem Übersetzen müssen folgende Voraussetzungen geschaffen werden:

1. Die „kernel-header“ müssen installiert sein. Die Kernel-Header sind installiert wenn der Aufruf von `cat /lib/modules/`uname -r`/build/include/linux/modversions.h` ohne Fehler beendet. („uname -r“ ist hier ein Platzhalter für das Ergebnis des Aufrufs `uname -r`.)
2. Die Datei `/lib/modules/`uname -r`/build/include/linux/version.h` muß vorhanden sein oder zuvor vom Pfad `/boot/vmlinuz.version.h` kopiert werden.
3. Die Werkzeuge wie make, gcc, etc. müssen installiert sein. (Zum Editieren und Erforschen von fremdem Quellcode benutze ich gerne den Source-Navigator von RedHat oder ehemals Cygnus.

FAQs

Q. Der Treiber für meine PCAN-ISA Karte wird nicht installiert. Woran liegt das?

A. Eine Möglichkeit ist, daß sich der SJA-1000 Baustein auf der Karte beim Überprüfen des PeliCAN-Modus zurücksetzt. Wenn die Karte mit einem PLD mit einem roten Punkt bestückt ist und die Steckbrücke JP11 auf „Ein“ gesetzt, dann wird dies die Ursache sein. Abhilfe: Die Steckbrücke auf „Aus“ umlöten.

Q. Meine CAN-Karte oder CAN-Dongle empfängt keine Daten, obwohl ich senden kann. Woran liegt das?

A. Eine wahrscheinliche Ursache ist, daß der falsche Interrupt für die Karte oder die Schnittstelle konfiguriert ist. Dann lassen sich einzelne Telegramme senden, jedoch keine empfangen. Neben der falschen Angabe der Interrupt-Nummer ist oft die fehlende Freischaltung des Interrupts im BIOS („legacy ISA“) für die PCAN-ISA Karte die Ursache.

Q. Mein PCAN-Dongle wird bei der Nutzung abgewiesen obwohl die Installation erfolgreich war. Woran liegt das?

A. In der Regel liegt die Ursache im nicht parametrisierten Interrupt bei der Nutzung

zusammen mit dem PARPORT_SUBSYSTEM (1*).

Q. Welche Hilfsmittel zur Diagnose habe ich bei Problemen?

A. Hier bietet LINUX eine reichhaltige Auswahl. Zunächst lohnt es sich mit "tail -f /var/log/messages" die aktuellen Meldungen zu betrachten. Dann können nähere Informationen mit "cat /proc/pcan", "cat /proc/interrupts", "cat /proc/ioports" und "cat /proc/iomem" erfragt werden. Bitte beachten sie, daß die Interrupts der Hardware erst mit der Nutzung installiert werden.

Q. Wie interpretiere ich die Ausgabe von "cat /proc/pcan" ?

A. Die zweite Zeile gibt Auskunft über das Release des Treibers. In der dritten Zeile wird die Anzahl der gefundenen CAN-Kanäle und die zugeordnete "major"-Nummer angezeigt. Ab der vierten Zeile werden die Eigenschaften der einzelnen CAN-Kanäle beschreiben.

Die erste Spalte zeigt die zum Kanal zugeordnete "minor"-Nummer, die zweite Spalte den Typ des Kanals, die dritte und vierte Spalte die zugeordneten Basisadressen (Ports) und Interrupt-Nummern. Die folgenden vier Spalten zählen die Anzahl der durchgeführten Lese-, Schreib-, Interrupt- und Fehlerereignisse. Die letzte Spalte zeigt den letzten gespeicherten Fehlerstatus des Kanals an. Der Status ist ein Bit-Feld mit folgender Interpretation:

Bit	Description
0	Chip-send-buffer full
1	Chip-receive-buffer overrun
2	BUS warning
3	BUS passive
4	BUS Off
5	Receive-buffer is empty
6	Receive-buffer overrun
7	Send-buffer is full
14	Illegal parameter

Anhang

(1*) = Das „Parport Subsystem“ muß auf „Benutzung des Interrupts“ konfiguriert sein. Hierzu müssen in der Datei „/etc/modules.conf“ folgende Zeilen eingetragen sein:

```
alias parport_lowlevel parport_pc
options parport_pc io=0x378 irq=7
```

In der Regel steht in der „options“-Zeile „irq=none“. Diese Arbeiten können nur als „root“ ausgeführt werden. Sollte das „Parport Subsystem“ schon installiert sein, so kann es mit:

```
rmmod lp und/oder rmmod pcan
rmmod parport_pc
```

`rmmod parport`

entfernt werden.

(2*) = Das „Parport Subsystem“ kann auch nicht beachtet werden. Dann ist jedoch das wechselweise Benutzen des Parallel-Ports durch Dongle und Drucker nicht mehr möglich. Hierzu ist der Treiber mit der Option `PAR=NO_PARPORT_SUBSYSTEM` zu übersetzen. Dann ist die vorherige Angabe von „`modprobe parport`“ beim Installieren nicht notwendig.

(3*) = Für jeden Interface-Typ werden bis zu 8 Geräte-Schnittstellen reserviert. Dies sind für PCI die „minor“-Nummern 0..7, für ISA 8..15, für Dongle im SP-Modus 16..23 und für den Dongle im EPP-Modus 24..31. Entsprechend sind die Geräte-Dateien mit Namen versehen, z.B.:

```
/dev/pcan0   für den ersten PCI-Kanal
/dev/pcan8   für den ersten ISA-Kanal
/dev/pcan16  für den ersten SP-Kanal
/dev/pcan24  für den ersten EPP-Kanal
```

Der Aufruf von „`pcan_make_devices n`“ legt jedoch nur `n` Geräte-Dateien eines Typs an. „`n`“ kann je nach Bedarf von 1..8 angepasst werden.

(4*) = In manchen Fällen ist die dynamische Vergabe der „major“-Gerätenummern unerwünscht. Dies kann jedoch in den Treiberquellen (Datei „`pcan_main.h`“, Konstante „`PCAN_MAJOR`“) geändert werden. Danach muss der Treiber neu übersetzt und installiert werden. Am Besten sollte hier die Nummer „91“ verwendet werden. Diese ist schon für CAN-Devices vorgesehen. Bitte beachten sie, dass hiermit Konflikte zu anderen CAN-Treibern entstehen können.

(5*) = Die „`read`“- und die „`write`“-Schnittstelle des Treibers geben ASCII-formatierte Daten aus oder akzeptieren ASCII-formatierte Daten als Eingang. Das Format der Telegramme ist für „`read`“ und „`write`“ identisch definiert, so daß sogar eine Umleitung der Daten möglich ist:

```
cat /dev/pcan0 > /dev/pcan8
```

Das Format der Daten entspricht der obigen Telegrammbeschreibung. Zusätzlich zu der Telegrammbeschreibung liefert die „`read`“-Ausgabe noch einen groben Zeitstempel in Millisekunden mit. Bei einer Umleitung wie oben wird der Zeitstempel ignoriert.

Die „`write`“-Ausgabe akzeptiert auch die Angabe einer Initialisierung:

```
cat „i 0x1234 e“ > /dev/pcan8
```

Der erste Parameter kennzeichnet die Initialisierung, der zweite Parameter nennt einen Wert für die beiden BTR0/BTR1 Register des SJA1000 Bausteins. Der optionale dritte Parameter, „`e`“, läßt auch „`extended frames`“ zu.

Selbstverständlich ist für die programmtechnische Anbindung auch eine schnellere

„ioctl()“-Schnittstelle definiert.

(6*) = Es wird nur Hardware basierend auf dem Philips SJA1000 Chip akzeptiert. PCAN-ISA oder -PCI Hardware wird direkt beim Installieren verifiziert. Im Gegensatz dazu wird die PCAN-Dongle Hardware erst beim Aufruf eines „open()“ überprüft. Daher wird bei erkanntem Parallelport ohne angesteckten Dongle kein Fehler gemeldet.