

INTRODUCTION

LINUX DRIVERS



DIN EN ISO 9001:2008 certified

Edition: 02.01-08/2011

Product information

This manual contains the technical installation and important instructions for correct commissioning and usage, as well as production information according to the current status before printing. The content of this manual and the technical product data may be changed without prior notice. ADDI-DATA GmbH reserves the right to make changes to the technical data and the materials included herein.

Warranty and liability

The user is not permitted to make changes to the product beyond the intended use, or to interfere with the product in any other way.

ADDI-DATA shall not be liable for obvious printing and phrasing errors. In addition, ADDI DATA, if legally permissible, shall not be liable for personal injury or damage to materials caused by improper installation and/or commissioning of the product by the user or improper use, for example, if the product is operated despite faulty safety and protection devices, or if notes in the operating instructions regarding transport, storage, installation, commissioning, operation, thresholds, etc. are not taken into consideration. Liability is further excluded if the operator changes the product or the source code files without authorisation and/or if the operator is guilty of not monitoring the permanent operational capability of working parts and this has led to damage.

Copyright

This manual, which is intended for the operator and its staff only, is protected by copyright. Duplication of the information contained in the operating instructions and of any other product information, or disclosure of this information for use by third parties, is not permitted, unless this right has been granted by the product licence issued. Non-compliance with this could lead to civil and criminal proceedings.

ADDI-DATA software product licence

Please read this licence carefully before using the standard software. The customer is only granted the right to use this software if he/she agrees with the conditions of this licence.

The software must only be used to set up the ADDI-DATA products.

Reproduction of the software is forbidden (except for back-up and for exchange of faulty data carriers). Disassembly, decompilation, decryption and reverse engineering of the software are forbidden. This licence and the software may be transferred to a third party if this party has acquired a product by purchase, has agreed to all the conditions in this licence contract and the original owner does not keep any copies of the software.

Trademarks

- ADDI-DATA, APCI-1500, MSX-Box and MSX-E are registered trademarks of ADDI-DATA GmbH.
- Turbo Pascal, Delphi, Borland C, Borland C++ are registered trademarks of Borland Software Corporation.
- Microsoft .NET, Microsoft C, Visual C++, MS-DOS, Windows 95, Windows 98, Windows 2000, Windows NT, Windows EmbeddedNT, Windows XP, Windows Vista, Windows 7, Windows Server 2000, Windows Server 2003, Windows Embedded and Internet Explorer are registered trademarks of Microsoft Corporation.
- LabVIEW, LabWindows/CVI, DASyLab, DIAdem are registered trademarks of National Instruments Corporation.
- CompactPCI is a registered trademark of PCI Industrial Computer Manufacturers Group.
- VxWorks is a registered trademark of Wind River Systems, Inc.
- RTX is a registered trademark of IntervalZero.

Warning

The following risks result from improper implementation and from use of the board contrary to the regulations:



Personal injury



Damage to the board, the PC and peripherals



Pollution of the environment

■ Protect yourself, others and the environment!

■ Read the safety precautions (yellow leaflet) carefully!

If this leaflet is not enclosed with the documentation, please contact us and ask for it.

■ Observe the instructions of this manual!

Make sure that you do not forget or skip any step. We are not liable for damages resulting from a wrong use of the board.

■ Pay attention to the following symbols:



IMPORTANT!

Designates hints and other useful information.



WARNING!

Designates a possibly dangerous situation.

If the instructions are ignored, the board, the PC and/or peripherals may be **destroyed**.



WARNING!

Designates a possibly dangerous situation.

If the instructions are ignored, the board, the PC and/or peripherals may be **destroyed** and persons may be **endangered**.

Contents

Warning	3
1 Linux – General	5
1.1 Why Linux?	5
1.2 What is Linux?	5
2 Linux versions and distributions	6
2.1 Linux version.....	6
2.2 Linux distribution	6
2.3 Linux structure.....	7
2.4 Differences between user level and kernel level.....	8
3 Linux driver types by ADDI-DATA	11
3.1 Which driver for my application?	12
3.2 Selecting an adequate driver	13
4 Linux driver table	14
5 Appendix	15
5.1 Glossary	15
5.2 Further sources	17
6 Contact and support	18

Figures

Fig. 2-1: Linux structure.....	7
Fig. 2-2: Example 1: Access to hardware.....	9
Fig. 2-3: Example 2: Saving values in a file (data logger)	10

1 Linux – General

This brochure describes the advantages of Linux for measurement and automation applications and supports you in selecting an adequate driver type. For this you should read this brochure, complete the table in chapter 3.2 and select an adequate driver type by using the driver table.

1.1 Why Linux?

Using Linux has many advantages. This is why ADDI-DATA has developed Linux drivers.

Open Source:

Linux and the source code are freely available. This enables you to develop, integrate, modify and debug drivers and applications.

Control:

You have the control over all running processes and drivers.

Real time:

By applying some patches (e.g. RTAI), Linux operates easily in real time.

Support:

You find several sources of information and help about Linux on the Internet and in books.

1.2 What is Linux?

The term "Linux" basically refers to an operating system "kernel". The kernel is a key component of a complete operating system.

Most of us, however, use the name "Linux" to refer to a complete operating system.

Many Linux features are similar to those of the operating system UNIX such as multitasking, virtual memory, shared libraries and multistack networking including IPv4 and IPv6.

Although originally developed first for 32-bit x86-based PCs (386 or higher), today Linux also runs on different architectures, for example Motorola 68000, PowerPC, ARM, and MIPS.

For more information see also: www.kernel.org

2 Linux versions and distributions

2.1 Linux version

Before the 2.6 kernel was introduced, the Linux version was always composed of three numbers (W.X.Y).

In the development of the kernel, X was odd. If the kernel was stable, X was even.

Increase of Y for a stable kernel was reserved for security updates, new functions or bug corrections.

Example:

2.4.2 = Second revision of the stable kernel 2.4

2.5.3 = Third revision of the development kernel 2.5

From the 2.6 kernel, no unstable branches have been released anymore, but kernel maintainers intend to have a stable version with new functions. The version is 2.6.Y.Z, in which Z is only used for security updates or bug corrections.

In order to find out the kernel version that you are using, you can type "uname -a" in a console:

```
[~]# uname -a
Linux SW08-Linux 2.6.15 #1 SMP PREEMPT Mon Jun 19 16:25:30 CEST 2006 i686
GNU/Linux
```



IMPORTANT!

The ADDI-DATA team needs this information for any request or further enquiry.

2.2 Linux distribution

A Linux distribution is a version of a Unix-like operating system comprising GNU, the Linux kernel and other assorted software.

In commercially backed distributions such as Red Hat, Ubuntu (backed by Canonical Ltd.), SUSE (backed by Novell) and Mandriva and community projects such as Debian and Gentoo, software is tested and assembled before the distribution is released. There are currently more than 300 Linux distribution projects running to actively develop, revise and improve the respective distribution.



IMPORTANT!

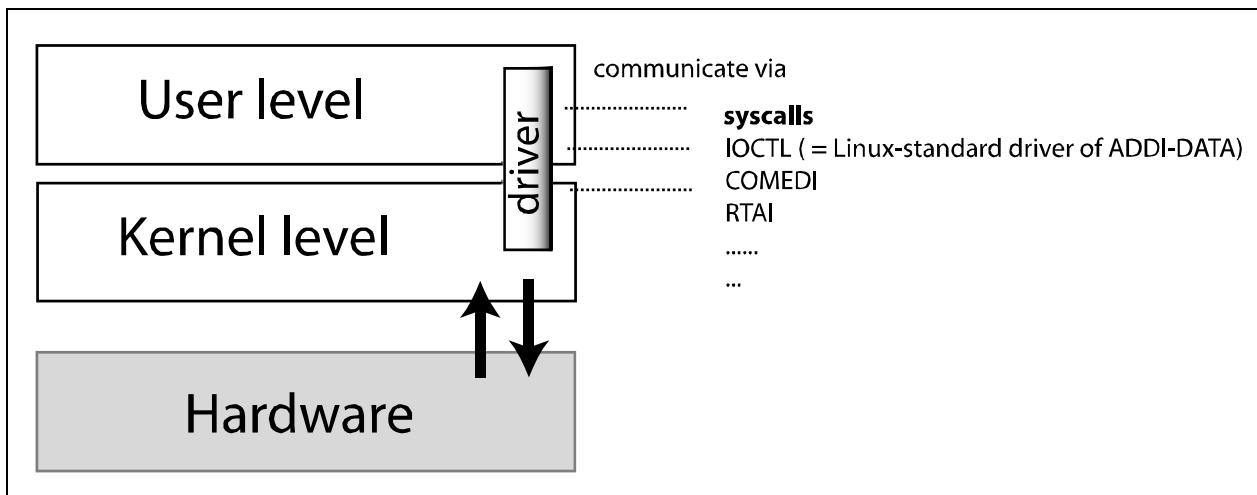
ADDI-DATA uses the Debian and Kubuntu distribution.

On request, we will send to you the ADDI-DATA MSX-Box Live DVD (Kubuntu).

2.3 Linux structure

The complete system is composed of three main components (hardware, kernel level and user level), whereas Linux is composed of the user and kernel level. The kernel level and the user level communicate with each other using several calls, the so-called syscalls:

Fig. 2-1: Linux structure



2.4 Differences between user level and kernel level

	User level	Kernel level
Characteristics	User level drivers are available in the form of ioctl calls. ioctl commands are developed to be as similar as possible to the ADDI-DATA Windows driver API. This makes the transition between Linux and Windows easier.	The kernel API is developed to be as similar as possible to the ADDI-DATA Windows driver API. This makes the transition between Linux and Windows easier.
Speed	In user mode (on a standard Linux platform), processes are scheduled with a delay of 10 ms. This means that when a usleep (5) is to be executed, the waiting time is at least 10 ms instead of 5 ms. The process priority is lower than in kernel mode.	In kernel mode, the execution is faster than in user mode. Kernel mode processes have a higher priority than user mode processes. Real time can be reached with patches like RTAI. Time measurement and delay can be done with high resolution (e.g. ns).
Hardware access	Hardware access is not allowed on the user level. Still some functions (e.g. ioperm) allow this access, but it is not secure. Only the kernel level has direct access to the hardware. Interrupt handles are available through the kernel level by using polling or signals (asynchronous interruption).	Hardware access has to be realised in kernel mode. All required functions are available here (e.g. inb, outb etc). Interruptions (synchronous mode ¹) with frequencies of fewer than 100 µs are possible (e.g. Linux + RTAI).
Development limitations	C/C++ and other programming languages can be used. There are no specific limitations.	C is the commonly used programming language. Under Linux, files cannot be used and floating points are not allowed. Many functions of the stdlib are not available. However, ADDI-DATA offers floating points on the MSX-Box.
Development speed	No specific knowledge is needed.	Some specific functions and structures have to be known, but development is as difficult as in user mode.

¹ For more information, please refer to the glossary (chapter 5.1)

	User level	Kernel level
Use	<p>It is used for calls on kernel driver functions to control the hardware, e.g. by ioctl calls or FIFO, shared memory etc.</p> <ul style="list-style-type: none"> - Writing configuration and regulation applications - Server/Client applications that send or receive frames from the Ethernet - Web front-end interface. - Applications that read hardware values through the kernel driver and log these values (application is used as a data logger) 	<p>It is used for drivers for the hardware in kernel module form</p> <ul style="list-style-type: none"> - Exchanging data with the user level via ioctl, shared memory, FIFO or /proc - Fast measurement, regulation and data acquisition - Real-time applications
Binary extensions	Applications: no extension, .exe (Cygwin)	Kernel modules: *.o, *.ko
General	The user level is used for applications.	The kernel level is used for drivers and processes that need the highest priority.
Conclusion	<p>On the user level, applications can be developed to access the hardware over a kernel level driver. On the user level, data is saved (in a file or over a socket), for example the values from an analog input: The driver on the kernel level accesses the analog input to initialise it and directly read the raw value. On the user level, the raw value can be obtained from the kernel level by using FIFO, shared memory and ioctl calls. The raw value can be converted into a floating point value to compare it with the limit values or log it in a file that can be sent via Ethernet (by the user level application).</p>	

Fig. 2-2: Example 1: Access to hardware

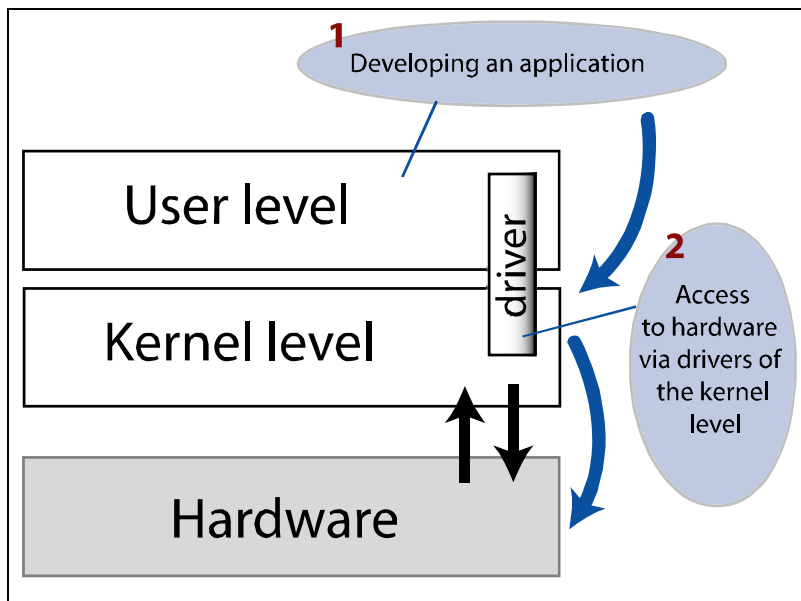
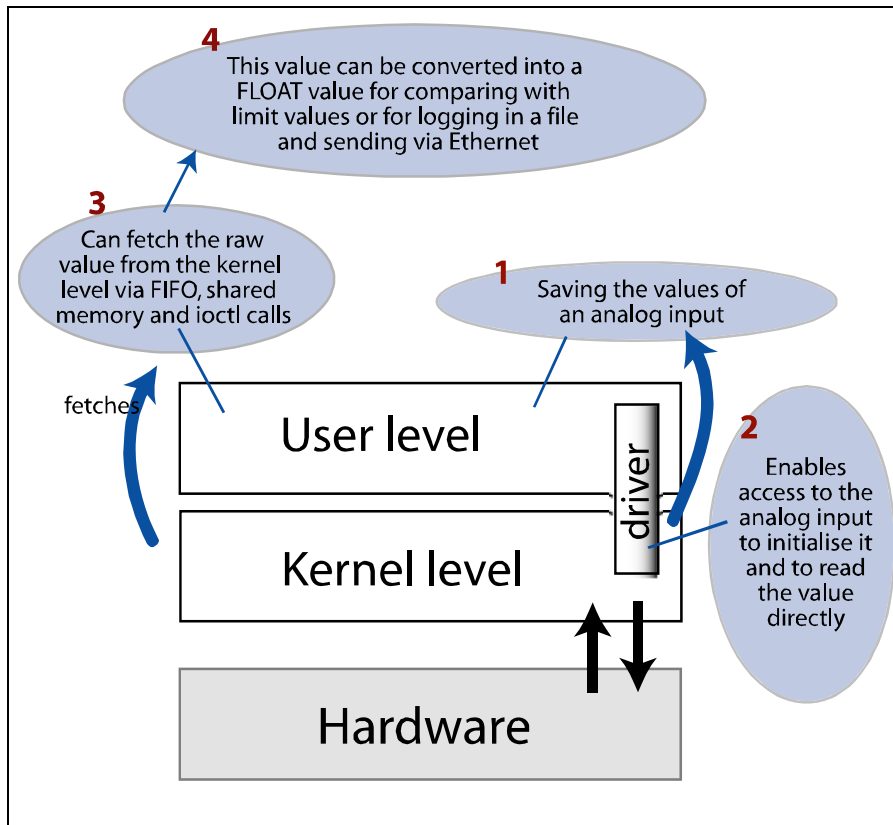


Fig. 2-3: Example 2: Saving values in a file (data logger)



3 Linux driver types by ADDI-DATA

ADDI-DATA offers four categories of drivers. We recommend you to use the "Standard Driver", because it is easy to use and supports nearly all functions of the respective product. However, in certain cases we also offer Comedi and RTAI drivers.

	Comedi	ioctl API	Kernel API	Standard Driver (ADDI-DATA standard driver)
Features	Comedi is a collection of drivers for data acquisition hardware. These drivers work under Linux, and also under Linux combined with the real-time extensions RTAI and RTLinux. The Comedi core, which ties all drivers together, allows applications completely independent from the hardware to be written.	ioctl driver commands are the same as Windows ADDI-DATA driver commands that allow for board checking by a user level application.	Kernel drivers do not provide a software API that allows for board checking from the user level. However, the API is also identical to Windows ADDI-DATA drivers.	The standard drivers are a combination of ioctl and Kernel API drivers. They can be used either from the kernel module or from the user level.
ADDI-DATA offers	ADDI-DATA has included some of its boards in the Comedi package. Samples are provided for each supported function (see driver table in Chapter 4 in which the supported boards are listed).	see driver table in Chapter 4 in which the supported boards are listed	on request	see "Standard kernel + ioctl API" in the driver table in Chapter 4

3.1 Which driver for my application?

	Comedi	ioctl API (ADDI-DATA standard driver)	Kernel API	Standard
Recommended use	Used if boards from different manufacturers are to be controlled by the same API.	Used for applications that are written for the user mode in case the application does not require a high priority and maximum execution speed. The application needs to access files, stlib functions etc.	Used if the application is in the form of a kernel module. The application should have a high priority or has to be real time (with e.g. RTAI).	Combination of ioctl and kernel API drivers. Can be used for almost any type of application.
Execution speed	Depends on if used on the user level or kernel level with RTAI (see also chapter 2.4)	User mode executions (see also chapter 2.4)	Kernel mode executions or real time (see also chapter 2.4)	Depends on if used on the user level or kernel level with RTAI (see also chapter 2.4)
Development limitations	see chapter 2.4	see chapter 2.4 The interrupt can only be used in synchronous mode.	see chapter 2.4 Interrupts are available in synchronous ¹ and asynchronous ¹ mode.	see chapter 2.4
Use	Comedi is usually used in user mode for data loggers or regulation applications. (see also www.comedi.org)	see chapter 2.4	see chapter 2.4	see chapter 2.4

¹ For more information, please refer to the glossary (see chapter 5.1)

3.2 Selecting an adequate driver

If you decide to use ADDI-DATA products under Linux, we will support you in selecting an adequate driver type. We recommend you to observe the following points:

- Read this brochure (with driver table).
- Complete the table below.
- Choose an adequate driver type from the driver table.

Contact us if you are not sure (please note your kernel version).

Feature	Notes
Linux kernel version? (uname -a)	
Linux distribution? (Debian version xxx, SUSE version xxxx etc.)	
Which functions do I want to use? (e.g. analog input with or without interrupt etc.)	
Should the driver be accessed from the user level? (application or kernel module)	
Application speed? (is e.g. real time required?)	
The adequate driver type for my requirements is:	

4 Linux driver table

Driver type	Kernel version	APCI-035	APCI-1016	APCI-1024	APCI-1032	PC104 PLUSs1500	APCI-1500	APCI-1508	APCI-1516	APCI-1564	APCI-1648	APCI-1696	APCI-1710	APCI-2016	APCI-2032	APCI-2200	APCI-3000	APCI-3001	APCI-3002	APCI-3003	APCI-3006	APCI-3008	APCI-3010	APCI-3016	APCI-3100	APCI-3106	APCI-3110	APCI-3116	APCI-3120	APCI-3122	APCI-3200	APCI-3300	APCI-3500	APCI-3501	APCI-3504
Comedi 2.6 (Update phase)	2.4	X			X		X		X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X		X	X		X		
	2.6	X			X		X		X	X	X	X	X	X	X	X	X	X	X	X	X		X	X	X	X	X	X		X	X		X		
Standard Kernel + ioctl API	2.4.22 to 2.6.20.6				X	X	X		X	X	X		X	X	X	X	X	X	X	X	X		X	X	X	X	X	X		X		X	X		
					X	X	X		X	X	X		X	X	X	X	X	X	X	X	X		X	X	X	X	X	X		X		X	X		
Native	2.4																																		
	2.6																																		

Driver type	Kernel version	APCI-3600	APCI-3701	APCI-7300-X	APCI-7420-X	APCI-7500-X	APCI-7800-X
Comedi 2.6 (Update phase)	2.4						
	2.6						
Standard Kernel + ioctl API	2.4.22 to 2.6.20.6	X					
		X					
Native	2.4			X	X	X	X
	2.6			X	X	X	X

5 Appendix

5.1 Glossary

RTAI

= Real Time Application Interface

RTAI is a real-time extension for the Linux kernel, which lets you write applications with strict timing constraints for Linux. Like Linux itself the RTAI software is a community effort. RTAI supports several architectures:

- x86 (with/without FPU and TSC)
- PowerPC
- ARM (StrongARM: clps711x-family, cirrus Logic EP7xxx, CS89712, PXA25x)
- MIPS

RTAI provides deterministic response to interrupts, POSIX compliant and native RTAI real time tasks.

RTAI consists mainly of two parts:

- A patch to the Linux kernel which introduces a hardware abstraction layer
- A broad variety of services which make real time programmers' lives easier.

The latest version of RTAI uses Adeos, providing additional abstraction and much lessened dependencies on the "patched" operating system.

ioctl

The system call `ioctl`, found on Unix-like systems, allows application to control or communicate with a device driver outside the usual read/write of data. This call originated in AT&T Unix version 7. Its name abbreviates the phrase I/O control.

An `ioctl` call takes as parameters:

1. an open file descriptor
2. a request code number
3. a pointer to data (either going to the driver or to come back from it)

The kernel generally dispatches an `ioctl` straight to the device driver, which can interpret the request number and data in whatever way required. The writers of each driver document request number for that particular driver and provide them a constants in a header file. Some systems have conventions encoding the size of

the data in the number, and whether the processing involves input or output. TCSETS exemplifies an `ioctl` on a serial port. The normal read and write calls on a serial port receive and send data bytes. An `ioctl` (`fd`, TCSETS, data) call, separate from such normal I/O, controls various driver options like handling of special characters, or the output signals on the port (such as the DTR signal).

Open Source

Open Source describes practices in production and development that promote access to the end product's sources. Some consider it as a pragmatic methodology. Before open source became widely adopted, developers and producers used a variety of phrases to describe the concept; the term open source gained popularity with the rise of the Internet and its enabling of diverse production models, communication paths, and interactive communities.

Subsequently, open source software became the most prominent face of open source. The open source model can allow for the concurrent use of different agendas and approaches in production, in contrast with more centralized models of development such as those typically used in commercial software companies.

GPL

= (GNU) General Public License

The GNU GPL is a widely used free software license, originally written by Richard Stallmann for the GNU project. The latest version of the license, version 2, was released in 1991. The GNU Lesser General Public License (LGPL) is a modified version of the GPL, intended for some software libraries.

GCC

Gnu C Compiler

Interrupt

The user interrupt routine can be called as follows:

Synchronous mode

Directly by the interrupt routine of the driver (synchronous mode). The code of the user interrupt routine operates on the kernel level.

Asynchronous mode

By the interrupt thread (asynchronous mode). An event is generated and the interrupt thread calls up the user interrupt routine. The code of the user interrupt routine operates on the kernel level.

Real time

A system operates in real time if it receives inputs quantities (e.g. signals, data) within a defined time, and provides the results in time for a partner system or for the system environment.

5.2 Further sources

Would you like to know more about Linux?

Literature:

RUBINI, Alessandro; CORBET Jonathan: *Linux Device Drivers*. O'Reilly & Associates (3rd edition for Linux 2.6)

BOVET, Daniel P.; CESATI, Marco: *Understanding the Linux Kernel*. O'Reilly & Associates 2000

Internet:

www.kubuntu.org

www.debian.org

www.kernel.org

www.linuxdoc.org

www.rtao.org

www.wikipedia.org

6 Contact and support

Do you have any questions? Write or phone us:

Address: ADDI-DATA GmbH
Airpark Business Center
Airport Boulevard B210
77836 Rheinmünster
Germany

Phone: +49 7229 1847-0

Fax: +49 7229 1847-222

E-mail: info@addi-data.com

Manual and software download from the Internet:

www.addi-data.com