DIN EN ISO 9001:2000
certified

ADDI-DATA GmbH
Dieselstraße 3
D-77833 OTTERSWEIER

Technical support:
+49 (0)7223 / 9493 – 0

Introduction

Linux drivers

Edition: 01.05 – 09/2006

Figures

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 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 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 use the name "Linux" to refer to a complete operating system, which includes the kernel and a lot of other programs required for a useful operating system.
Linux is similar to the operating system Unix and has all features you would expect in a modern fully-fledged Unix, including true multitasking, virtual memory, shared libraries, demand loading, shared copy-on-write executables, proper memory management, 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, the Linux version was always composed of three numbers (W.X.Y).
For a development kernel X is odd. If X is even, the kernel is stable.

Increase of Y for a stable kernel was reserved for security updates, new functionalities 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

Since the 2.6 kernel no unstable branches have been released, but kernel maintainers intend to have a stable version with new functions. The version is 2.6.Y.Z. In which Z is only 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 question.

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.

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, assemble and test the software before releasing their distribution. There are currently more than 300 Linux distribution projects in active development, revising and improving their respective distributions.

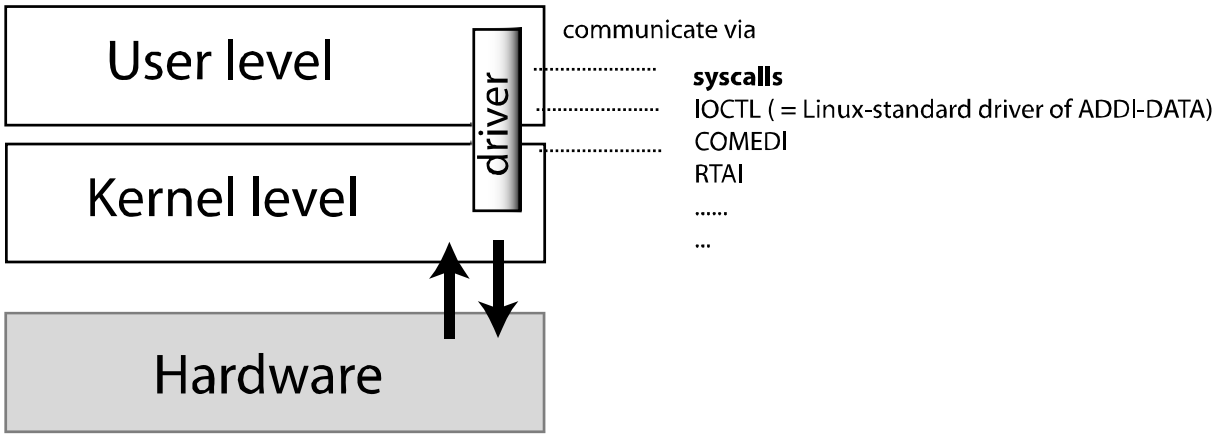
IMPORTANT!
ADDI-DATA uses the Debian and Knoppix distribution.

On request we will send to you the ADDI-DATA MSX-Box Live-CD (Knoppix).

2.3 Linux structure

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

Fig. 2-1: Linux structure



2.4 Difference between user and kernel level

	User level	Kernel level
Characteristics	**User level drivers are 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.	**Kernel API are 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 the user mode processes (on a standard Linux platform) are scheduled with a time of 10 ms. This means that when a **usleep (5)** has to be done, it will sleep at least 10 ms instead of 5 ms. Processes priority is lower than in Kernel mode.	In kernel mode 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 accesses are not allowed in user level. Some specific functions (e.g. **ioperm**) allow these accesses but this is not safe. Only the kernel level has direct access to the hardware. Interruption handles are available through the kernel level by using polling or signals. **Asynchronous interruption**.	**Hardware accesses** have to be realised in the kernel mode. All required functions are available (e.g. inb, outb…) here. Interruptions (**synchronous mode***) with frequencies less than 100 µs are possible (e.g. Linux + RTAI).
Development limitations	**C/C++ and further programming languages** can be used. There are no specific limitations.	**C is the commonly used programming language.** Under Linux files cannot be used, 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.
Use for	It is used for calls on Kernel driver functions to control the hardware, e.g. by **ioctl calls** or **FIFO, shared memory,**…. - Writing configuration, regulation application. - Server/Client applications that send or receive frames from the Ethernet. - Web front-end interface. - applications that read values from hardware through kernel driver and log these values (application is used as data logger).	It is used for driver for the hardware in kernel module form. - Exchanging data with the user level via **ioctl, shared memory, FIFO** or /**proc**. - fast time measurement, fast regulation, and fast data acquisition. - Real time applications

* For more information please refer to the glossary (chapter 5)

	User level	Kernel level
Binary extensions	Applications: No extension, .exe (Cygwin)	Kernel modules: *.o, *.ko
General	User level: Used for applications	Kernel level: Used for drivers and processes that need high priority.
Conclusion	In the user level, applications can be developed to access to the hardware through a kernel level driver. Saving data (in a file or through a socket) from e.g. an analog input is done in user level: The driver in kernel level accesses to the analog input to initialise it and read directly the raw value. In the user level the raw value can be obtained from the kernel level by using FIFO, shared memory, ioctl calls. The raw value can be converted in floating points value to compare it with limits, or log it in a file that then can be sent via the Ethernet (by the user level application).	

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

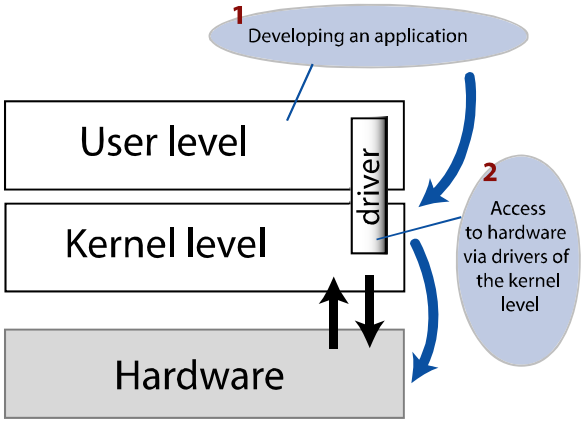
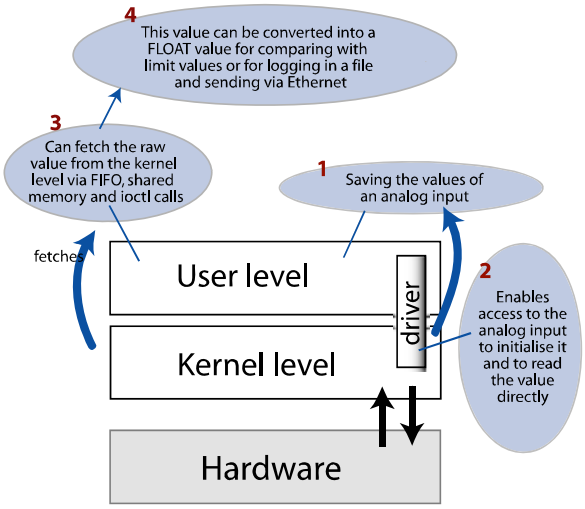


Fig. 2-3: Example 2: Saving values in a file (data logger)



3 LINUX DRIVER TYPES BY ADDI-DATA

ADDI-DATA provides 4 categories of drivers. Therefore we recommend you to use “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 with Linux combined with the real time extensions RTAI and RTLinux. The Comedi core, which ties all drivers together, allows applications to be written that are completely independent from the hardware.	Ioctl-drivers commands are the same as Windows ADDI-DATA drivers commands that permit to control boards by a user level application.	Kernel drivers do not provide a software API that permits to control boards from the user level. However, the API is also like Windows ADDI-DATA drivers.	The standard drivers are a combination of ioctl and Kernel API drivers. It is possible to use them from a kernel module or from the user level.
ADDI-DATA offers	ADDI-DATA included many of its boards in the comedi package. Samples are provided for each functionality. See the driver table below to know which boards are supported.	See driver table below, where the supported product are listed.	On request.	See “Standard kernel + ioctl API” in the driver table below.

3.1 Which driver for my application?

	Comedi	Ioctl API (ADDI-DATA standard driver)	Kernel API	Standard
Recommended use	Used if boards of different manufacturers are to be controlled by the same API.	Used for applications that are written for the user mode. If the application does not require a high priority and execution speed. The application needs to access files, stlib functions,…	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 applications.
Execution speed	Depends on if used in user level or kernel level with RTAI (see also: chapter 2.4)	User mode performances (see also: chapter 2.4)	Kernel mode performances or real time (see also: chapter 2.4)	Depends on if used in user or kernel level with RTAI (see also chapter 2.4)
Development limitations	See: chapter 2.4	See: chapter 2.4 The interrupt can be used only in asynchronous mode.	See: chapter 2.4 Interrupts are available in synchronous* and asynchronous* mode.	See: chapter 2.4
Use for	Comedi is used usually in the user mode for data logger or regulation applications. (See also http://www.comedi.org)	See: chapter 2.4	See: chapter 2.4	See: chapter 2.4

* For more information please refer to the glossary (see chapter 5)

3.2 Selecting an adequate driver

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

1. Read this brochure (with driver table)
2. Complete the table below
3. Choose an adequate driver type from the driver table
Contact ADDI-DATA if you are not sure (please note your kernel version)

Feature	Notes
Version of the Linux kernel? (uname –a)	
Linux distribution? (Debian version xxx, SuSe version xxxx,…)	
Which functionalities to use? (e.g. analog input with or without interrupt…)	
Is the driver to 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 DRIVER TABLE

You can find the current version of this table on www.addi-data.com under Download > Manual download “Introduction Linux Drivers”

Driver type	Kernel version	APCI-035	APCI-1016	APCI-1024	APCI-1032	PCI-104PLUS-1500	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	2.4	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		X	
2.6 (Update phase)	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		X	
ioctl API	2.4				X		X			X	X	X	X																X						
	2.6						X						X																X						
Kernel API	2.4				X						X	X	X																					X	
	2.6																																		
Standard	2.4				X	X	X		X	X	X		X	X	X		X	X	X	X	X		X	X	X	X	X	X			X				
Kernel + ioctl API	2.6				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-7300X	APCI-7420X	APCI-7500X	APCI-7800X
Comedi	2.4						
2.6 (Update phase)	2.6						
ioctl API	2.4						
	2.6						
Kernel API	2.4						
	2.6						
Standard	2.4	X					
Kernel + ioctl API	2.6	X					
Native	2.4			X	X	X	X
	2.6			X	X	X	X

5 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 in 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 in 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.

6 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.knoppix.net
www.kernel.org
www.linuxdoc.org
www.rtai.org
www.wikipedia.org