Linux-GPIB, an IEEE488 development environment for Linux

1996 by Claus Schroeter
(clausi@chemie.fu-berlin.de)

Abstract

This Paper describes the features of Linux-GPIB, a powerful and flexible development environment for IEEE488 applications.

What is Linux-GPIB ?

The wide spreaded IEEE488 bus, introduced 1965 by Hewlett Packard as HP-IB, today can be found in nearly all industrial and scientific laboratories where a flexible and scaleable instrumentation system is needed. In comparison to usual measurement hardware, as plug-in A/D D/A boards etc, IEEE488 devices are standalone 'intelligent' devices, so no special knowledge about the internal architecture of measurement devices is necessary. The second important advantage is that a minimal set of functions that is common to all devices can be used for programming, today this concept is called API.
Nearly all companies that are into Measurement and Control business provide IEEE488 hardware together with all the colorful tools that enables programmers to implement just as colorful applications. Until now Linux-Users had to do without IEEE488 because there was no support by this companies for Linux. The same is for other hardware support of course.
One consequence of this is that Linux for long time eked out an existence as 'Hacker Operating System' that could be used at best as X-terminal or for educational purposes only. Since Linux has been known as open and very stable development platform in educational environments, it comes more and more in focus of 'professional' users in industry and commercial development now. One essential is the avaliability of appropriate drivers and toolsets that enables developers to implement their desired software.
The Linux-GPIB development environment fills one gap in this field providing both, hardware support and a basic set of development tools that can be used for instantaneous application programming and testing.

The basic Components of Linux-GPIB

Linux-GPIB consists of three basic parts:

- The linux kernel driver module, that is dynamically loadable and configurable at runtime.
- The user-level library that can be used for programming IEEE488 applications in C or whatever. The library is call-compatible to the popular NI488^{tm} library, so porting applications from other operating systems should be easy.
- The application suite consists of: ibsh a tcl-interpreter extension that contains a basic function set for gpib programming and a lot of helpful graphical widgets as circular dials, plots me-

ters and so on. With ibsh powerful applications with a graphical user interface can be written within minutes.

All this componets comes with full source code (as usual under Linux) so this environment provides a more open architecture than usual IEEE488 packages. The architecture of the driver is easily extensible to other IEEE488 boards so new hardware support can be quickly added.

The driver module, the kernel and the Hardware

The linux-GPIB driver is implemented as loadable driver module that can be loaded into the kernel at runtime without rebooting linux. The basic configuration options as base-address DMA-channel and IRQ-level can be specified at loading time so a recompilation of the module is only necessary if the kernel changes.
The driver supports a wide variety of popular IEEE488 hardware that is availiable on the market, top of all the very powerful National Instruments chipsets followed by the standard nec7210 based architectures. The hardware architecture dependent parts of the driver is isolated as much as possible from the rest of the driver to make porting the code to new IEEE488 hardware as easy as possible.
An set of intrinsic debugging functions enables users to trace any action the driver does inside the kernel's gear, so tracing segmentation faults or other errors is very easy. A debug level switch can be set either at module-loading time or just while applications are running. The formatted debugging output written to the syslog contains all information that is necessary to exactly locate the point where the error occurs. Different levels provide different granularities for function tracing so that debugging can be refined step by step.

In comparison to other IEEE488 drivers the Linux-GPIB drivers provides a simple multiprocessing support that enables different processes to share the bus at the same time.

The driver provides also a VFS interface for sending and receiving data via normal file descriptors. This interface is planned to be extended to a powerful HP-call-style unified language interface (ULI) that enables users to access the bus from normal applications (as matlab, basic etc.).

The user-library and the rGPIB system

The GPIB- user-level library provides an easy-to-use interface to usual programming languages as C or FORTRAN. The call syntax is similar to the very popular National Instruments DOS-library to reduce the psychological barrier for DOS-Users porting their applications to Linux.

The library reads a configuration file at startup that sets up the driver- and the library characteristics. The communication to single devices is implemented using virtual device descriptors so changing a device's characteristic properties as GPIB-address or EOS-Handling modes can be done from the library configuration.

The remote GPIB (rGPIB) system can be used to access the bus from other hosts in the network without significant changes to the application. The GPIB-Library contains both the client and the server stubs so both client and server applications can be implemented using the same library. Fig. 1 shows how the rGPIB system is implemented. As soon the rGPIB-server is started on the machine where the IEEE488 bus is located, remote access is granted for the hosts or domains specified in the configuration. Since the architecture of the

driver allows limited sharing of the bus, more than one host can access the bus. For the applications the only change is that the virtual file descriptor contains the target hostname.

The application suite

John Ousterhout's Tcl programming language together with the Tk extension provides a powerful tool that can be used to implement very user-friendly applications within the minute-to-hour range. It can be easily extended with own C-Code to import own functions to Tcl.
`ibsh` that comes with the linux-GPIB package is an extended Tcl interpreter that contains Tk, a basic set of linux-GPIB functions and some very useful visualization extensions as plots, meters, bargraphs and so on. With `ibsh` it is possible to write own Tcl/Tk measurement and control applications completely in Tcl that have a graphical user interface without fiddling with all the X11 or Motif stuff. A wide variety of availiable Tcl/Tk extensions, from database management to socket programming, is now availiable directly to measurement and control applications without any costs and limitation to the phantasy of application programmers.
Together with `ibsh` a library configuration and test utility `ibconf` that comes with the package that can be used for easy point-and-click configuration of the library characteristics and device and bus testing purposes (See Fig. 2). `ibconf` is only a raw demonstration what is possible with `ibsh`. The testing facility of `ibconf` is able to test device functions or low-level bus control functions just by pressing some buttons and enter some commands, so beginners are enabled to get familiar with their devices without writing test programs in C or whatever.
In the near future visual programming tools will be also availiable for Tcl (e.g. SpecTcl) that enables programming GUI via drag-and-drop.

Additional Tools and Support

Since Linux-GPIB has so many different configuration options there are a lot possible sources for mistakes. To provide a quick and easy installation process, the configuration of the source code and its adaption to the current kernel version will be done by a menu-driven setup utility that eases the choice of the possible options.
`ibchk` is a check utility that checks the for different 'frequently done mistakes' and assists with helpful error messages if something is going wrong.
Additionally documentation material like a user-manual, a quick reference guide and various manual pages come with the package to help understanding the linux-GPIB package.
Support is provided via the linux-GPIB mailing list or the linux-lab-project[1].

[1] For further information see `http://www.fu-berlin.de/~clausi`

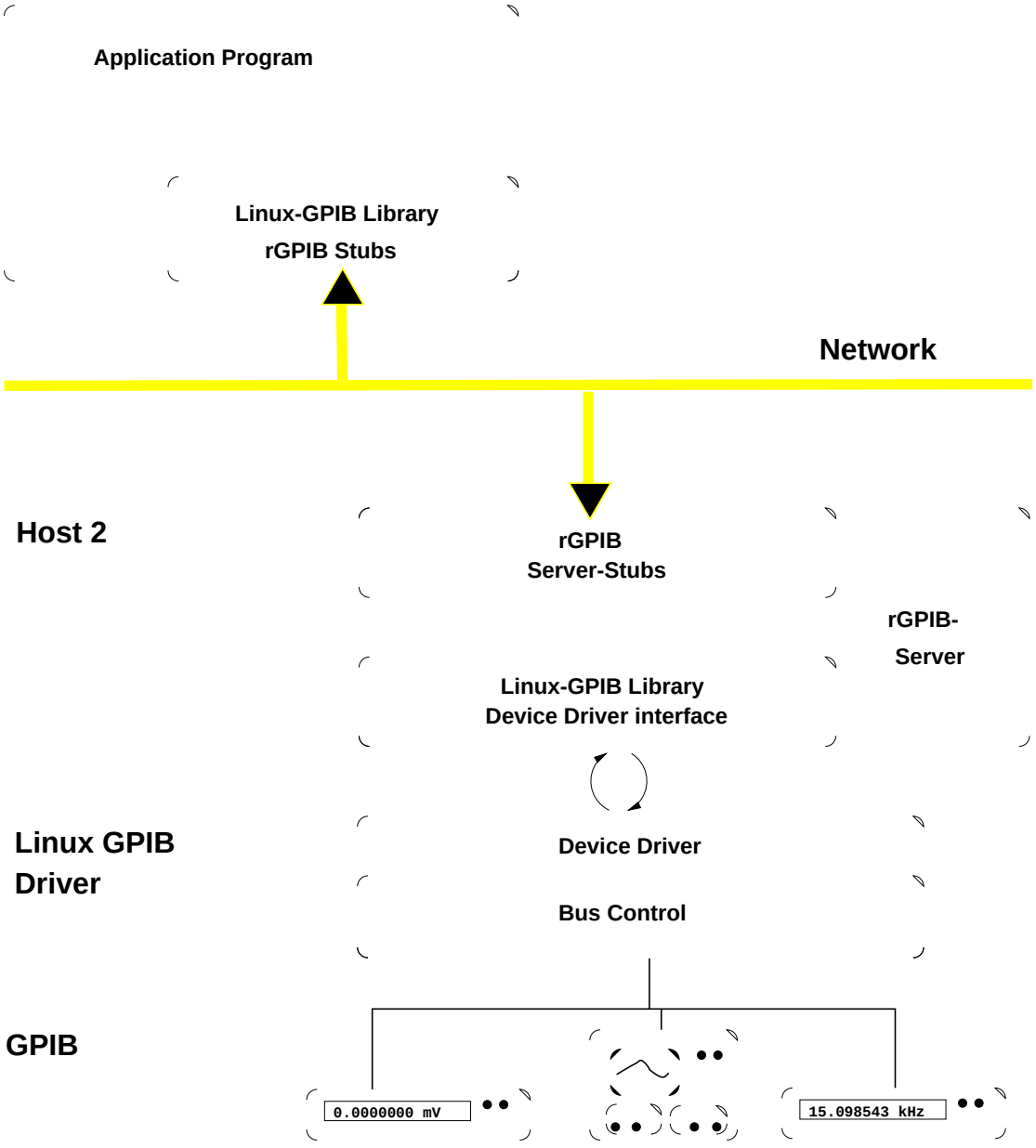

Figure 1: The rGPIB System

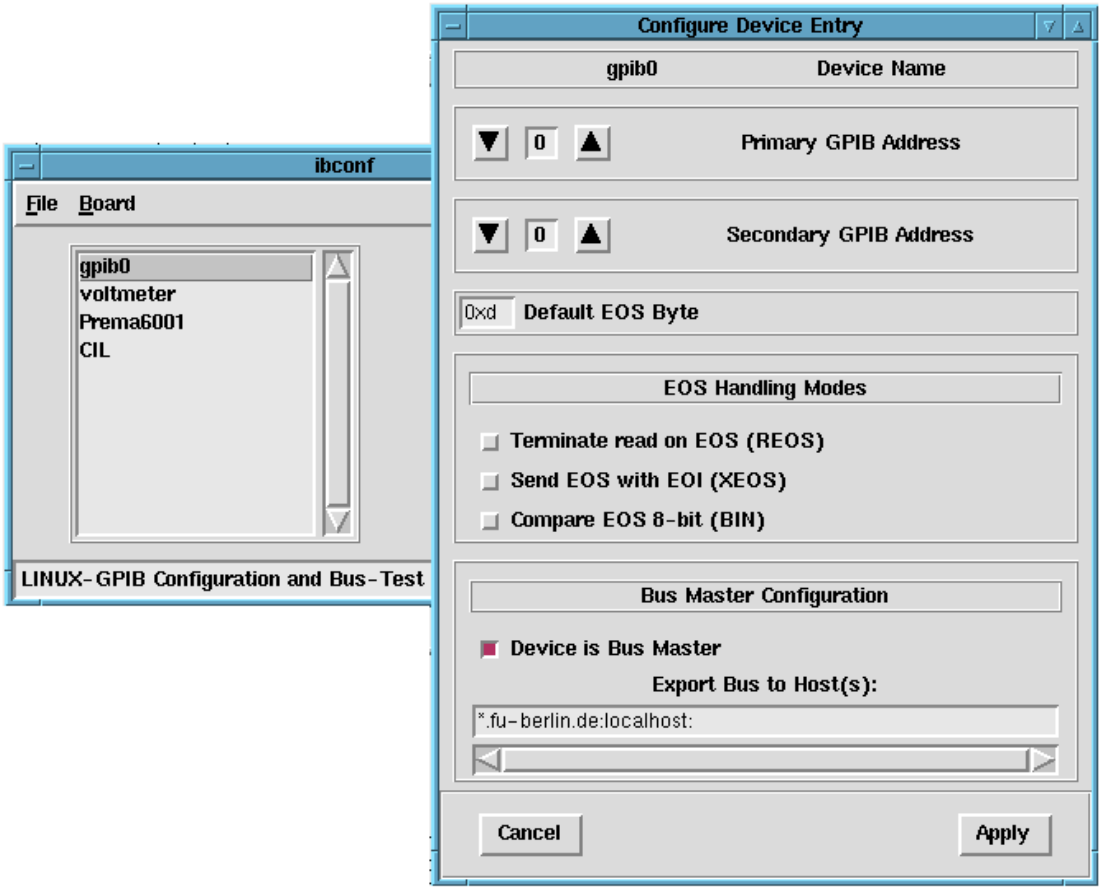

Figure 2: The `ibconf` Utility