



## 1.101: Hardware und System- architektur



Quellen u.a.:

<http://www.lpi.org>

<http://www.lpi-test.de>

<http://www.lpi-academy.de>

Dean, Jeffrey: LPI Linux Certification (LPI Level1), O'Reilly

Linup Front GmbH: Linux-Systemadministration 1



Copyright (©) 2009 by Dr. W. Kicherer 2008 by A. Grupp, R. Boenninger. This work is licensed under the Creative Commons Attribution-NonCommercial-Share Alike 2.0 Germany License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/2.0/de/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.



## Hardware und Systemarchitektur

### Inhalt:

BIOS

PCI-Bus

ATA

SCSI

USB

Geräte

udev

HAL/D-Bus



## BIOS (Basic Input/Output System)

Früher: Schnittstelle zwischen Hardware und Betriebssystem (DOS)

Heutige Hauptaufgaben:

- Starten von Linux. Die Platte, von der gebootet wird, muss vom BIOS richtig erkannt werden; andere falsch erkannte: deaktivieren.
- "Einschalten" aller erforderlichen HW-Komponenten (z.B. Onboard-Grafik, USB-Tastatur bzw. -maus)
- "Abschalten" der zwar verfügbaren aber nicht erwünschten Hardware-Komponenten (z.B. IrDA-Port)

Für die LPIC-1-Zertifizierung keine tiefen Kenntnisse erforderlich. Wichtig ist bei Problemen mit Hardware auch im BIOS zu überprüfen ob Hardware überhaupt erkannt wird.



## BIOS Einstellungen

**Hardware-Uhr:** am besten MEZ oder Weltzeit UTC einstellen. Linux muss dann noch die Zeitzone bekannt gegeben werden.

**Bootreihenfolge:** C: A: SCSI: CD/DVD: ...

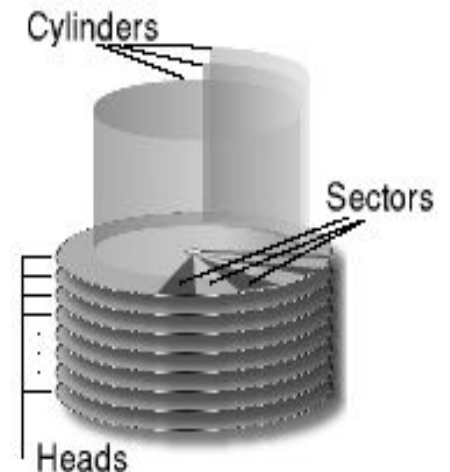
**BIOS-Passwort:** für Systemsicherheit

**Festplatten** ...teilen dem BIOS ihre Plattengeometrie in Form von Zylinder /Köpfen / Sektoren (CHS) mit. Je nach BIOS ist Variablen-Größe beschränkt (z.B. max. 1024 Zylinder)

Dadurch können Datenblöcke (je 512 Byte) eindeutig adressiert werden. Beispiel für die Größe einer Platte, die vom BIOS noch erkannt wird:

1024 Cylinders, (10 bit)	16 Heads, (4 bit)	63 Sectors (6 bit)	$\Sigma$ : 20bit
$1024 * 16 * 63 * 512 = 528 \text{ MB}$			

Werden mehr Bit für die Adressierung verwendet, kann die Größe gesteigert werden.





## BIOS Einstellungen

Moderne Platte halten sich längst nicht mehr an dieses starre System

- Die Adressierung der Datenblöcke geschieht intern und ist nach außen nicht transparent
- Trotzdem meldet die Platte dem BIOS eine Geometrie üblicherweise von CHS 16383/16/63, damit das BIOS die Platte erkennt und überhaupt booten kann.
- Im BIOS kann der LBA-Modus (Logical Block Addressing) eingestellt werden. Sektoren werden einfach sequentiell von außen nach innen durchgezählt.
- LBA muss nur aktiviert werden (heute: **Standard**), wenn der Rechner sonst nicht bootet. Nach dem Booten ignoriert Linux die Angaben des BIOS sowieso.



## 1024 Zylinder Problem

Aus der Endlichkeit der CHS-Adressierung (20Bit) ergab sich bei älteren BIOSen (und bei Lilo\*), dass sich ein zu bootendes System innerhalb der ersten 1024 Zylinder befinden musste.

### Abhilfe:

- LBA einschalten oder
- Plattenkapazität durch Jumper drosseln oder
- andere Platte für Booten nehmen oder
- /boot in eine separate Partition auslagern, die sich innerhalb der ersten 1024 Zylinder befindet (als erste Partition!).

Mit der Adressierung LBA (Logical Block Addressing) werden die Sektoren der Reihe nach, bei 0 und »außen« beginnend, durchnummeriert. Die Abbildung von CHS auf LBA wird durch die Logik der Festplatte selbst übernommen.

*\*) Lilo kann erst ab Version 32 (Revision 3) »hinter« dem Zylinder 1024 booten. Grub hat damit gar kein Problem.*



## Kernel-Module

1. Feste Fähigkeiten eines Linux-Kernels können konfiguriert werden => monolithischer Kernel
2. Zusätzliche Fähigkeiten in Form von Kernelmodulen nachladbar => modulare Struktur
3. Kernel-Module (`*.o`, `*.ko`) in Kategorien unterhalb von => `/lib/modules/<uname -r>/kernel/`
4. `lsmod` zeigt aktuell geladene Module und zugehörige Informationen (Größe des Moduls, welche anderen Module nutzen das Modul) an. Basis der Informationen ist `/proc/modules`



## Laden/Entladen/Anzeigen

- Module können mit *modprobe <MODUL>* geladen werden.
- Entfernt werden Module mit *modprobe -r <Modul>*. Dabei werden auch weitere dadurch nicht mehr benutzte Module entfernt
- Die Konfiguration erfolgt in */etc/modprobe.conf* oder */etc/modprobe.d/...* Dort können Aliase gesetzt werden (der Kernel kennt nur die major/minor Number, *modprobe* übersetzt dann auf das richtige Modul; blacklist von nicht zu installierenden Modulen bei Konflikten, ...)



## PCI-Bus

Es gibt unterschiedliche PCI-Busse (Auswahl):

PCI 32-Bit Bus; 33, 66 MHz

PCI-X 64-Bit Bus; 66,100, 133, 266 MHz

PCIe serieller Bus; 1.25 GHz

## lspci

Zeigt alle PCI-Geräte

**Syntax:** `lspci [OPTION]`

Optionen:

- v Ausführlichere Anzeige (auch -vv, -vvv)
- n zeige Hersteller und Gerätebezeichnung in numerischer Form
- t baumartige Darstellung von Bus-Bridge-Geräte-Verbindungen

Früher war noch `/proc/pci` eine wichtige Infoquelle (jetzt nicht mehr üblich). `/proc/bus/pci/...` enthält noch Dateien zum PCI-Bus (irrelevant), heute über `/sys` (siehe unten)



## lspci – Beispiel mit Anzeige der Netzwerkkarte

`lspci`

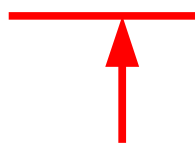
```
00:14.0 Bridge: nVidia Corporation MCP51 Ethernet Controller (rev a1)
```

`lspci -v`

```
00:14.0 Bridge: nVidia Corporation MCP51 Ethernet Controller (rev a1)
Subsystem: Giga-byte Technology Unknown device e000
Flags: bus master, 66MHz, fast devsel, latency 0, IRQ 177
Memory at f5005000 (32-bit, non-prefetchable) [size=4K]
I/O ports at b000 [size=8]
Capabilities: <access denied>
```

`lspci -n`

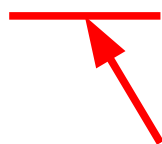
```
00:14.0 0680: 10de:0269 (rev a1)
```



Slot



Hersteller  
(nVidia)



Geräte-ID  
(EthernetController)

Zuordnung gemäß  
`/usr/share/misc/pci.ids`  
od. `/usr/share/pci.ids`

`[domain:]bus:device.function`



## ATA (Advanced Technology Attachment)

### parallel

- ATA, PATA, IDE (Integrated Device Electronics) bis 133 MB/s (ATA/133; UDMA 6)
- /dev/hda, /dev/hdb, ... (siehe 1.104 Gerätedateien), bis zu 63 Partitionen pro Festplatte

### seriell

- SATA bis 3GB/s (bald 6 GB/s), hotswapping fähig
- Verwendet die Bibliothek libsata und wird dann über die SCSI-Treiber von Linux angesprochen
- /dev/sda, /dev/sdb, .. bis zu 15 Partitionen pro Festplatte
- Im BIOS: AHCI-Mode (Advanced Host Controller Interface)



## SCSI (Small Computer System Interface)

Neben reinrassigen SCSI-Geräten (z.B. Serverfestplatten) finden sich etliche weitere Geräte, die vom Linux -Kernel als SCSI-Geräte gesehen werden und so über SCSI-Treiber gesteuert werden oder zumindest als SCSI-Geräte erscheinen lässt (libata-Treiber):

- USB-, FireWire
- Serial-ATA-Platten (SATA)
- SCSI-emulierte IDE-Platten

Die SCSI-Emulation war vor Kernel 2.6 bei IDE-CD-Brennern notwendig:

- SCSI-Emulation musste im Kernel eingeschaltet sein (Modul `ide-scsi`)
- Parameterübergabe an den Kernel durch den Bootloader:  
`/dev/hdc=ide-scsi`, damit Gerät von Anfang an als SCSI-Gerät gesehen wird.



## SCSI Standards

Verschiedene SCSI-Standards: (Quelle: Wikipedia )

Interface	Übertragungs			
	geschw. (MByte/sec)	Busbreite (bits)	max. Kabellänge (m)	max. Anzahl an Geräten
SCSI	5	8	6	8
Fast SCSI	10	8	1,5 – 3	8
Differential SCSI	5 – 10	8	12 – 25	8
Wide SCSI	20	16	1,5 – 3	16
Ultra SCSI	20	8	1,5 – 3	5 – 8
Ultra Wide SCSI	40	16	1,5 – 3	5 – 8
Ultra2 SCSI	40	8	12	8
Ultra2 Wide SCSI	80	16	12	16
Ultra3 SCSI	160	16	12	16
Ultra-320 SCSI	320	16	12	16
<b>Serial Attached SCSI(SAS)</b>	600	1 (Seriell)	25	16.256 pro Domain



## SCSI-ID

Jedes Gerät ist durch eine SCSI-ID (meist per Jumper) gekennzeichnet:

0 – 7 (8-Bit-Bus)

0 – 15 (16-Bit-Bus)

Anschluss des SCSI-Busses an einen Hostadapter. Teurere Modelle haben ein eigenes BIOS → Booten von einer SCSI-Platte möglich.

Terminierung eines Busses an beiden Enden:

- Terminierung durch den Hostadapter
- Terminierung durch Terminierungsstecker auf dem letzten Steckplatz oder Jumper auf dem letzten Gerät auf dem Bus.





## Gerätenamen

<code>/dev/sda</code>	Platte
<code>/dev/sdb</code>	Platte
<code>/dev/sdc</code>	Platte
<code>/dev/scd0</code>	CD-Laufwerk
<code>/dev/scd1</code>	CD-Laufwerk
<code>/dev/st0</code>	Bandlaufwerk
<code>/dev/sg0</code>	Generisches Gerät (z.B. Scanner)



## USB

### Merkmale:

- asymmetrisch (Kabelenden unterschiedlich)
- Jeder heutige Rechner hat einen oder mehrere USB-Controller
- An jedem Controller sind max. 127 Geräte anschließbar.
- Baumartige Verschachtelung über Hubs möglich.
- Hotplugging

USB 1.1: 1,5 MBit/s (niedrige Geschw.) für Mäuse, Tastaturen, Modems und 12 MBit/s (volle Geschw.) für Digicams, Scanner.

USB 2.0: 480 Mbit/s (hohe Geschwindigkeit)

Resultierende Geschwindigkeit, so wie das „schwächste“ Glied in der Kette.



## USB-Controller

2 Hardware-Typen für USB-Controller sind möglich:

OHCI Open Host Controller Interface von Compaq --oder--

UHCI Universal Host Controller Interface von Intel  
(einfacher, billiger, deswegen aufwändigere Treiber nötig)

Für USB 2.0 zusätzlich:

EHCI Enhanced Host Controller Interface stellt die USB 2.0-Funktionen bereit. Das EHCI nur für Hi-Speed-Modus (480 MBit/s) zuständig.

Falls USB-1.1-Gerät angeschlossen: EHCI reicht Datenverkehr an einen hinter ihm liegenden UHCI- oder OHCI-Controller weiter. (diese Controller sind typischerweise auf demselben Chip).



## USB-Controller

Welche(r) Controller ist eingebaut?

„lspci -v“ oder „/sys/bus/usb/devices/usb.../product“ geben Auskunft.

Welche Kernelmodule (Kerneltreiber) müssen geladen sein?

- uhci\_hcd
  - ohci\_hcd
  - ehci\_hci
- } Benötigen alle "usbcore"



## lsusb

Zeigt alle USB-Geräte

**Syntax:** lsusb [OPTION]

Optionen:

-v Ausführlichere Anzeige  
-t baumartige Darstellung  
von USB-Geräte-  
Verbindungen

```
r-andreas:~ # lsusb -t
Bus# 4
`-Dev# 1 Vendor 0x0000 Product 0x0000
  `--Dev# 2 Vendor 0x058f Product 0x6254
Bus# 3
`-Dev# 1 Vendor 0x0000 Product 0x0000
Bus# 2
`-Dev# 1 Vendor 0x0000 Product 0x0000
Bus# 1
`-Dev# 1 Vendor 0x0000 Product 0x0000
  `--Dev# 3 Vendor 0x11f5 Product 0x0001
```

Infos über die Geräte (Vendor, Product) finden sich in  
`/usr/share/misc/usb.ids`



- /proc beinhaltet Infos zu den Prozessen
- /sys beinhaltet Informationen zum System
- z.B. /sys/bus/usb/devices/usb1/product => Infos zum ersten USB-Controller
- /sys/block => Infos zu den Block-Devices
- /sys/class => Infos zu den Character-Devices
- z.T. sind Einträge per Hard-/Softlink doppelt vorhanden



- Früher: Alle bekannten Devices in /dev
- Heute: Nur die benötigten Devices werden vom udevd in /dev angelegt. /dev liegt nun auf einer RAM-Disk (tmpfs)
- Kernel sendet „uevents“ z.B. beim einstecken eines USB-Sticks => udevd nimmt diese entgegen und legt abhängig von Regeln in /dev Geräte an und/oder führt Aktionen aus
- Die Regeln für den udevd liegen in /etc/udev/



## Aufbau der Konfiguration

- Installierte Regeln: /lib/udev/rules.d/
- Eigene Regeln: /etc/udev/rules.d/
- Beispiel:

```
# CPU
KERNEL==Vergleich"cpu[0-9]*",      NAME=Aktion"cpu/%n/cpuid"
KERNEL=="microcode",              NAME="cpu/microcode", MODE="0600"
# miscellaneous
SUBSYSTEM=="rtc", DRIVERS=="rtc_cmos", SYMLINK+="rtc"
```

- Infos können mit `udevinfo` oder `udevadm info` ermittelt werden (`udevadm info -q all -n /dev/sda`)

Oder in den Log-Dateien



## Plugging und Start von Anwendungen

**coldplug** Anschluss bei ausgeschaltetem Gerät.

**hotplug** Anschluss während des Betriebs.

**hal / d-bus** der HAL (Hardware Abstraction Layer) nimmt u.a. über den udevd Infos entgegen und verteilt die Ereignisse an angemeldete Programm per D-Bus.  
Auskunft gibt: `lshal`