Spielchen mit dem Linksys-Router WRT54g

Thomas Kleffel Christian Daniel

Inhalt

Inhalt

Linksys Wrt54g GPL-Einforderung

GPL-Bedingungen:

- Binaries dürfen nur veröffentlicht werden wenn auch die Sourcen freigegeben werden
- Alle Veränderungen an den Ausgangssourcen müssen zugänglich gemacht werden
- Eigene Projekte dürfen nicht direkt gegen Bibliotheken (etc.) gelinkt werden, die unter der GPL stehen
- Zu den Sourcen gehören auch eventuell notwendige Build-Programme (z.B. Image-Generator)

Linksys Veröffentlichungs-Geschichte:

- Router und Firmware-Update werden als Binär-Dateien veröffentlicht. Die GPL wird nicht erwähnt...
  -> Streit
- Linksys veröffentlich einen Source-Tree, mit dem praktisch nichts anzufangen ist (Teile fehlen)
  -> mehr Streit
- Linksys veröffentlicht ein schickes .tar-Paket mit Cross-Compiler und Anleitung. Alle Tools zum Erzeugen des Flash-Images sind dabei – ein „make“ genügt, um ein Image zu bekommen
  -> alle (fast) glücklich

Inhalt des veröffentlichten Source-Codes:

- Toolchain mit C-Compiler
- µcLibC C-Standard-Bibliothek libc.so
- Linux-Kernel 2.4.20 + Patches für BCM4712KPB-CPU
- Sourcen für alle verwendeten Programme (von einigen kleinen Ausnahmen abgesehen)
- WLAN-Treiber nur als Binär-Module
  (ähnlich den nVidia- oder ATI-Grafiktreibern)

Inhalt

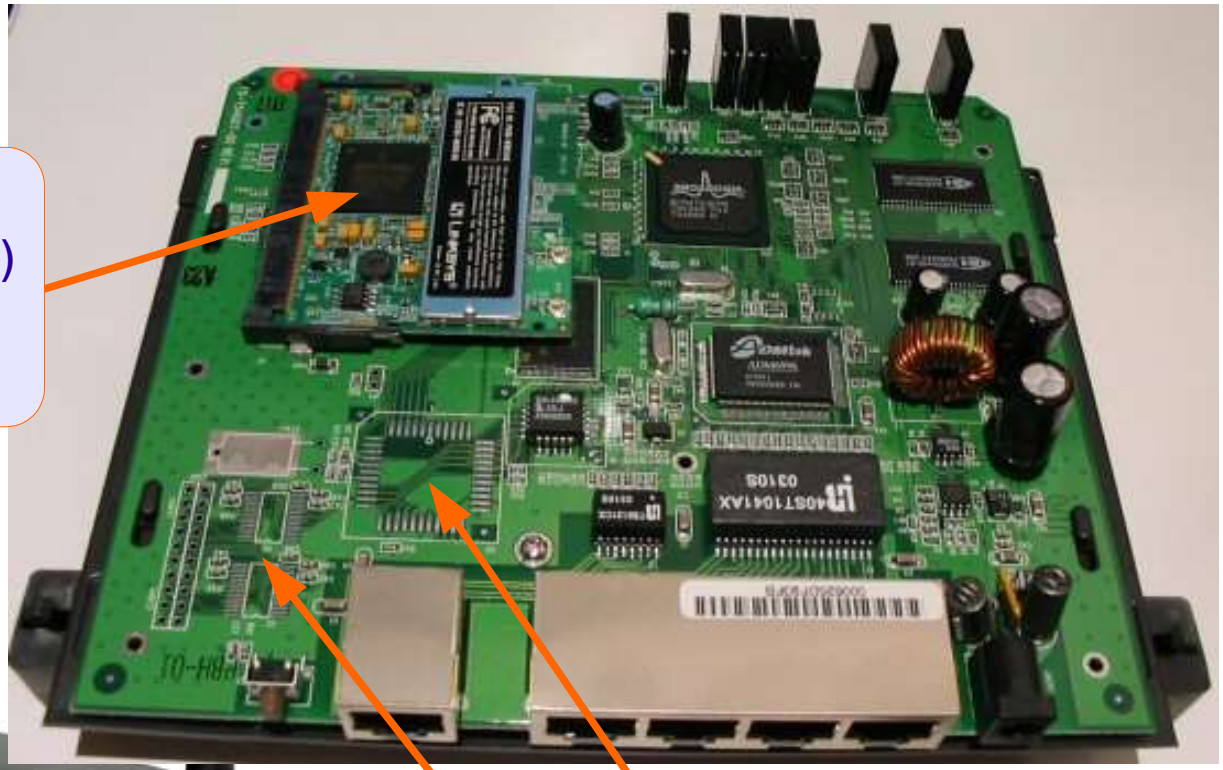
Linksys Wrt54g HW-Revisionen

Bisher wurden vier Revisionen vorgestellt:

- V1.0 (Seriennummer CDF1xxx)
  125MHz, WLAN auf Mini-PCI, 16MB RAM, 4MB Flash
- V1.1 (Seriennummer CDF2xxx-CDF3xxx)
  125MHz, WLAN auf Board, 16MB RAM, 4MB Flash
- V2 (Seriennummer CDF5xxx)
  200MHz, WLAN in CPU integriert, 16MB RAM, 4MB Flash
  Serielle Ports zugänglich
- GS (Seriennummer CGN1xxx)
  200MHz, SpeedBooster-WLAN, 32MB RAM, 8MB Flash
  Serielle Ports zugänglich

Linksys Wrt54g HW-Revision 1.0

Mini-PCI WLAN-Karte
(läuft auch in Notebooks)
Andere Karten
funktionieren auch...



Platz für einen UART 16C450
+ zwei Pegel-Wandler

Bilder ausgeliehen von: http://www.linksysinfo.org/modules.php?name=Content&pa=showpage&pid=6

Linksys Wrt54g HW-Revision 1.1

WLAN direkt auf Board
PCI-Bus noch zugreifbar

JTAG-Port
(nicht dokumentiert)

BUS-Port:

1: D0	2:A0
3: D1	4:A1
5: D2	6:A2
7: D3	8:A3
9: D4	10:/CS
11:D5	12:/RD
13:D6	14:/WR
15:D7	16:RESET
17:GND	18:INT
19:GND	20:GND

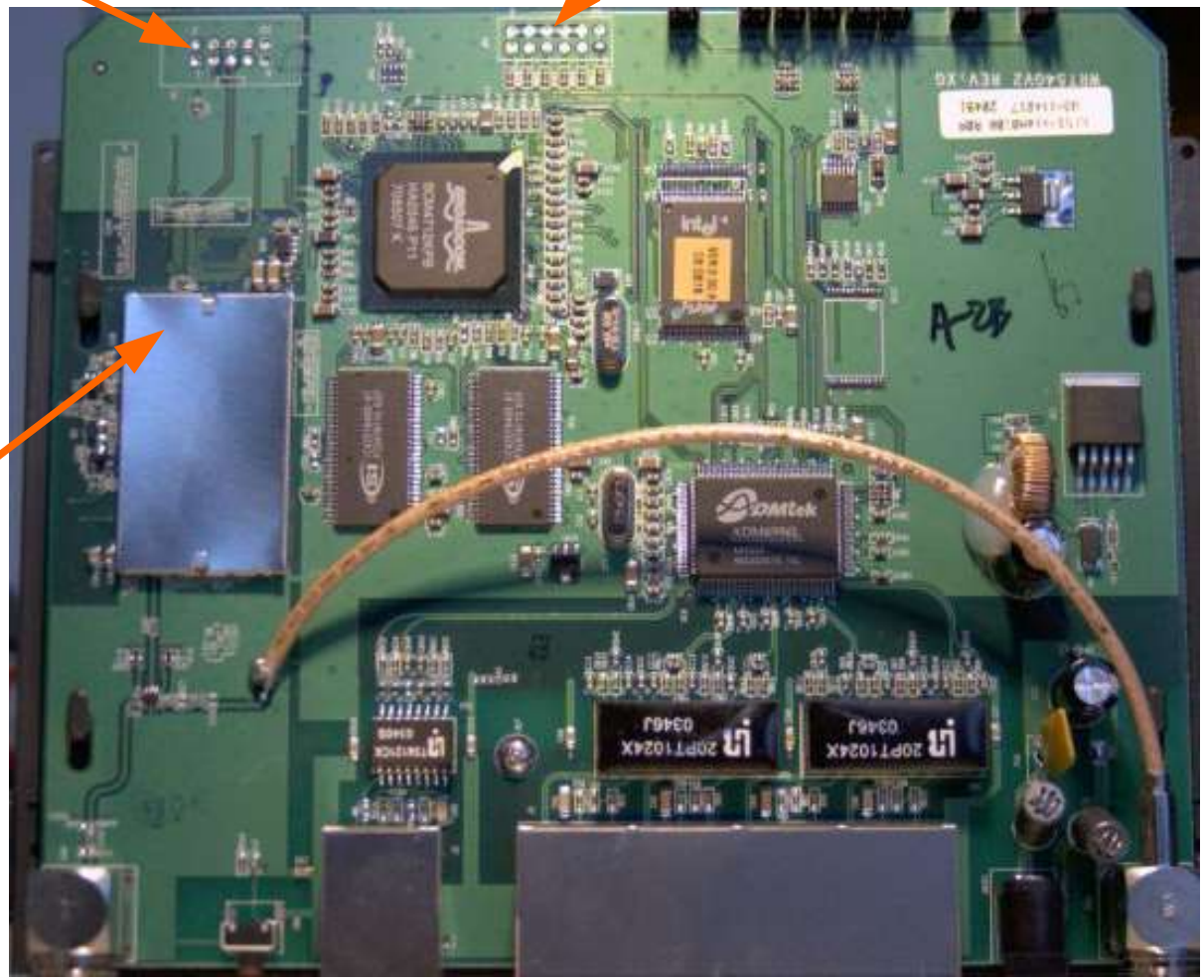
Linksys Wrt54g

HW-Revision 2

Zwei serielle Schnittstellen
auf 3.3V-Basis

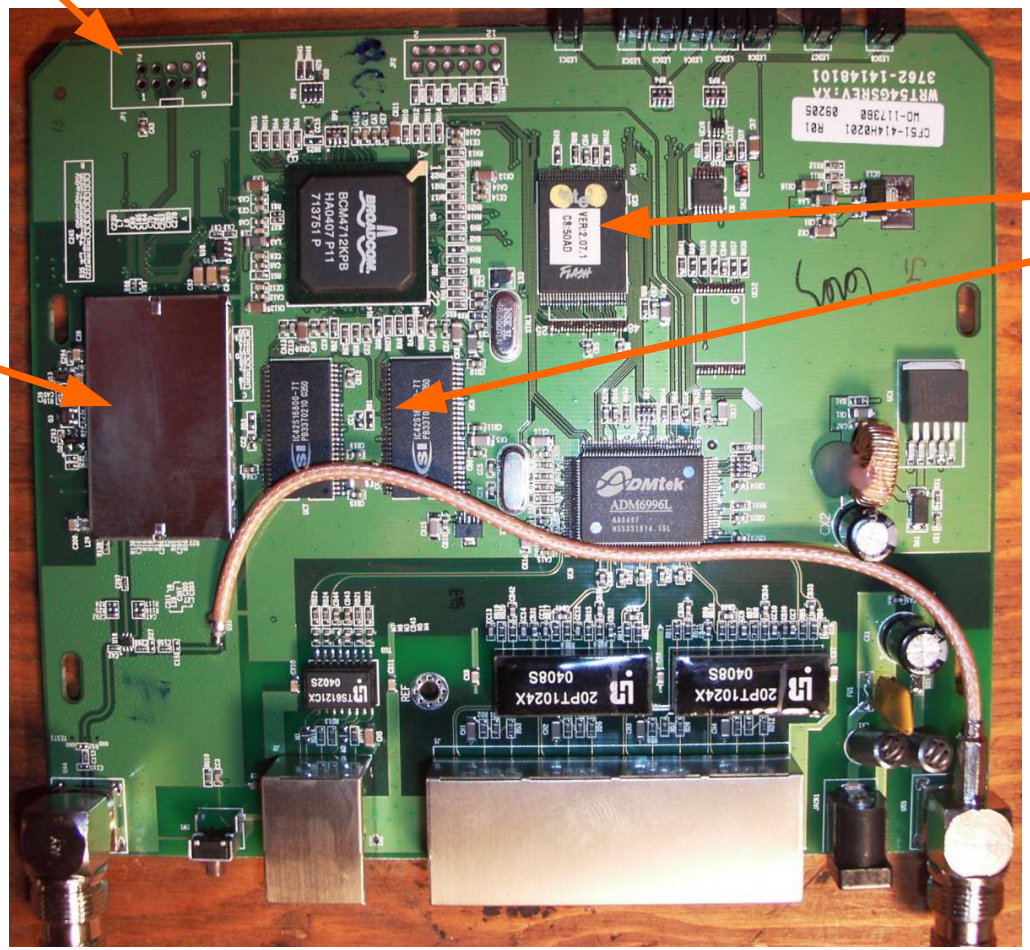
JTAG-Port
(nicht dokumentiert)

CPU enthält
den WLAN-DSP -
Auf dem Board
findet sich nur
noch der HF-Teil



Linksys Wrt54g HW-Revision GS1

Zwei serielle Schnittstellen auf 3.3V-Basis

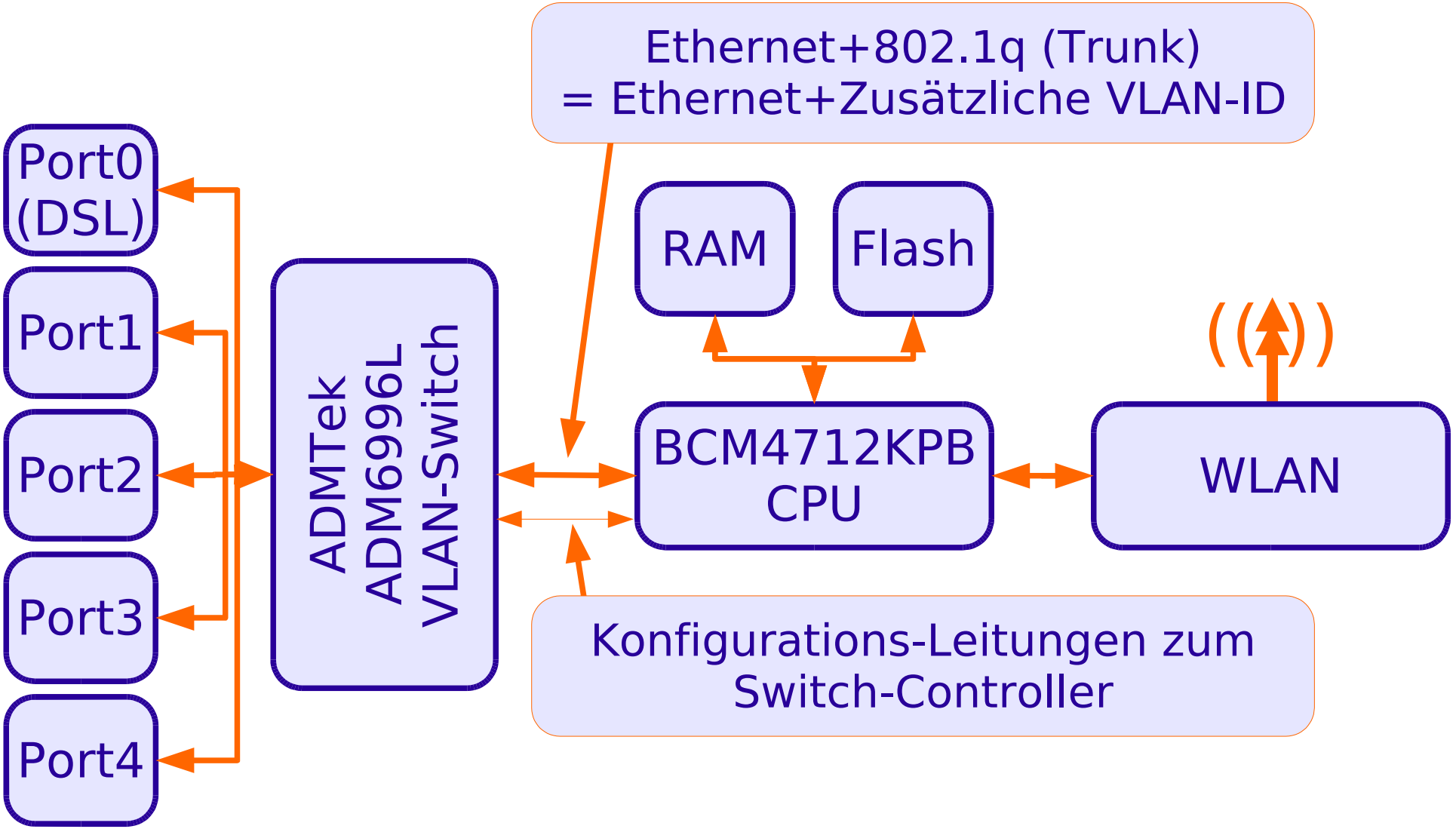


Doppelt so großes Flash und RAM (8MB / 32MB)

WLAN-HF mit SpeedBooster

Inhalt

Linksys Wrt54g Hardware-Details



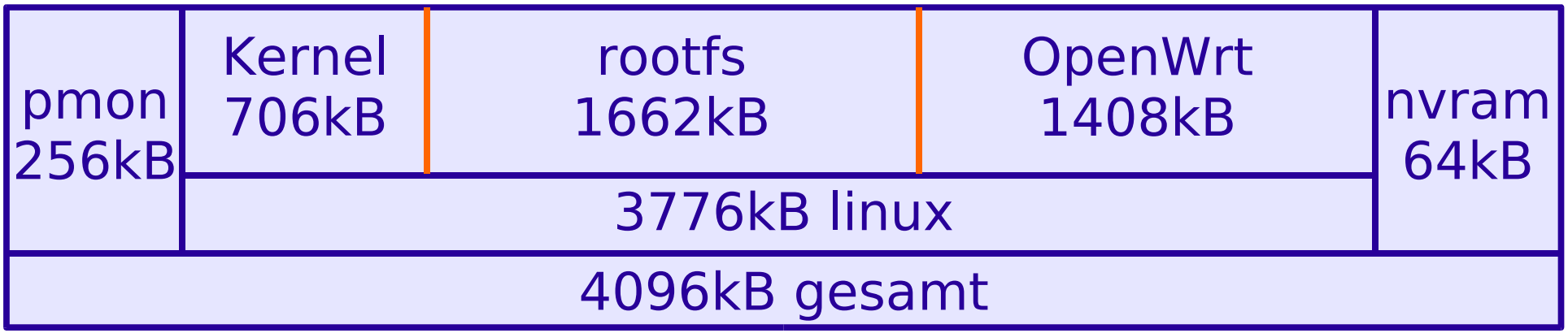
Verteilung der VLANs auf die Ports:

	Port0 (DSL)	Port1	Port2	Port3	Port4
V1-Hardware	1	2	2	2	2
V2&GS-Hardware	1	0	0	0	0

Unter Linux werden die VLANs mit „vconfig“ auf ein Trunk-Device konfiguriert und erscheinen dann als zusätzliche Interfaces, z.B. „**vlan1**“ für den DSL-Port

Aufteilung des Flash-Speichers:

```
0x00000000-0x00040000 : "pmon"      256 kBytes
0x00040000-0x003f0000 : "linux"    3776 kBytes
0x000f834c-0x0028dda0 : "rootfs"   1662 kBytes
0x003f0000-0x00400000 : "nvram"      64 kBytes
0x00290000-0x003f0000 : "OpenWrt" 1408 kBytes
```



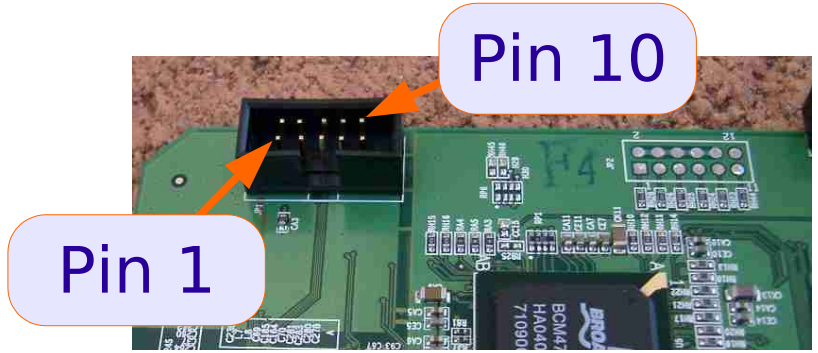
Inhalt

Linksys Wrt54g — Serielle Ports

Benutzung der seriellen Ports:

- V1-Router: Viel Spaß beim Organisieren eines passenden UARTs und Quartzes...
- V2&GS-Router: Es wird ein Pegel-Wandler für 3.3V benötigt (z.B. aufgebaut mit einem MAX3232 oder in der Pfusch-Version mit einem MAX232 und einigen Rs)

```
Pin  1: 3.3V          Pin  2: 3.3V
Pin  3: Tx (ttyS1)    Pin  4: Tx (ttyS0)
Pin  5: Rx (ttyS1)    Pin  6: Rx (ttyS0)
Pin  7: NC            Pin  8: NC
Pin  9: GND           Pin 10: GND
```

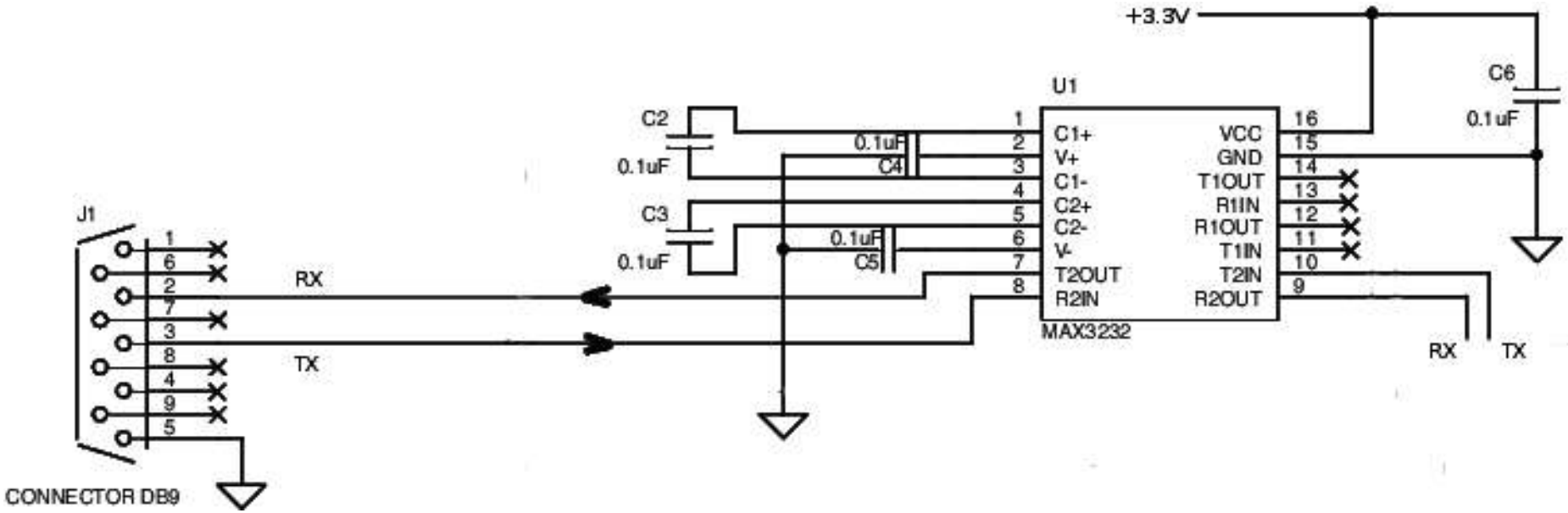


ttyS0 ist als System-Console mit 115200/8N1 konfiguriert

Mehr: http://www.rwhitby.net/wrt54gs/serial.html

Pegel-Konverter:

- Da der Linksys-Router intern mit 3.3V arbeitet und kein Pegel-Konverter für die genormten +12V/-12V eingebaut ist, muß man den von Hand nachrüsten:



Linksys Wrt54g Boot-Loader

```
Initializing Arena.
Initializing Devices.
et0: Broadcom BCM47xx 10/100 Mbps Ethernet Controller 3.50.21.0
CPU type 0x29007: 200MHz
Total memory: 0x2000000 bytes (32MB)

Total memory used by CFE:  0x80334DC0 - 0x8043A310 (1070416)
Initialized Data:          0x80334DC0 - 0x80336F40 (8576)
BSS Area:                  0x80336F40 - 0x80338310 (5072)
Local Heap:                0x80338310 - 0x80438310 (1048576)
Stack Area:                0x80438310 - 0x8043A310 (8192)
Text (code) segment:       0x80300000 - 0x8030F220 (61984)
Boot area (physical):      0x0043B000 - 0x0047B000
Relocation Factor:         I:00000000 - D:00000000

Boot version: v2.3
The boot is CFE

mac_init(): Find mac [00:0F:66:4C:D4:E9] in location 1
Nothing...
Device eth0:  hwaddr 00-0F-66-4C-D4-E9, ipaddr 192.168.1.1, ma
        gateway not set, nameserver not set
Reading :: Failed.: Timeout occured
Loader:raw Filesys:raw Dev:flash0.os File: Options:(null)
Loading: ..... 1830912 bytes read
Entry at 0x80001000
Closing network.
Starting program at 0x80001000
```

- Hier kann mit CTRL-C ein Boot-Prompt gestartet werden! (-> Booten z.B. direkt von TFTP)
- Falls „boot_wait“ auf „yes“ steht, wartet der Bootloader 15 Sekunden lang auf ein neues Firmware-Image zum Flashen
- Danach startet der BL den Flash-Inhalt

Linksys Wrt54g

Console

- Der Kernel auf dem Router verwendet /dev/tts/0 als serielle System-Console – d.h. man sieht hier den Bootvorgang und kann sich nach einem kurzen Umbau in /etc/inittab eine Shell geben lassen...

```
::sysinit:/etc/init.d/rcS
::shutdown:/sbin/halt
ttyS0::respawn:/bin/login
```

Muß werden zu...

```
::sysinit:/etc/init.d/rcS
::shutdown:/sbin/halt
tts/0::respawn:/bin/login
```

Kernel mit devfs legen das /dev-Verzeichnis eigenverantwortlich an – leider sind die Namen oft etwas anders gewählt

Inhalt

Linksys Wrt54g

Distributionen

Linksys-Original
in diversen Versionen
(neuere FW-Versionen laufen
auf älteren Routern)

Diverse kleinere Projekte
- mit SSH / Dropbear
- mit TX-Power-Hack
- mit Client-Mode
- etc...

SVEASOFT
- Web-Oberfläche bleibt
- PPTP-Client und Server
- Wondershaper
- WDS-Mode, Client-Mode
- Mit Wasserzeichen signiert
- ...

OpenWrt
- übernimmt nur den Kernel
- Paketbasiert – Programme können im System nachgerüstet werden
- Beschreibbare Partition im Flash
- Shell-System ohne Klick-Oberfläche

BNMSP-Mitglieder-Firmware
- PPTP-Client
- WLAN-Client-Mode
- NAT, DHCP-Server
- DAU-Tauglich

Linksys Wrt54g Konfiguration

- Die Original-Firmware und alle Abkömmlinge speichern ihre Konfiguration im **NVRAM** – also der letzten Page im Flash.
- Das NVRAM enthält die Konfiguration in Form von Variablen: name=Wert

Anzeigen des NVRAM:

```
$ nvram show
boardrev=0x10
et0macaddr=00:0F:66:4C:D4:E9
boot_wait=on
```

Lesen einer Variablen:

```
$ nvram get boot_wait
on
```

Lesen einer Variablen:

```
$ nvram set bootwait=on
$ nvram set xyz=abc
$ nvram commit
```

- OpenWrt benutzt in der Standard-Einstellung auch die Werte im NVRAM – kann aber z.B. rein über Skripten gesteuert werden, da Änderungen im Dateisystem einen Reboot überleben

Inhalt

Schritte für die Installation

- Herunterladen des Build-Systems (ca. 5MB)
- Build anstoßen (es werden weitere 200MB benötigt)
- Pakete kompilieren
- Fertiges Image installieren

http://openwrt.org/userguide.html

Herunterladen des Build-Systems vom CVS

Bei der Passwort-Abfrage einfach Enter drücken

```
$ cvs -d:pserver:anonymous@openwrt.org:/openwrt login
$ cvs -d:pserver:anonymous@openwrt.org:/openwrt co buildroot
$ cd buildroot
$ make
$ make packages
```

Das dauert dann etwas

Was ist herausgekommen?

```
-rw-r--r--   1 cd users 1537024 Nov 20 21:29 openwrt-g-code.bin
-rw-r--r--   1 cd users 1537024 Nov 20 21:29 openwrt-gs-code.bin
-rw-r--r--   1 cd users 1536000 Nov 20 21:29 openwrt-linux.trx
-rw-r--r--   1 cd users  623758 Nov 20 21:27 openwrt-kmodules.tar.bz2
```

Flash-Image für V1- und V2-Modelle

Flash-Image GS-Modelle

Flash Image für andere Geräte, z.B. Asus ap500

Kernel-Module zum Nachinstallieren

Was ist herausgekommen?
-> Verzeichnis „packages“

Packages
Packages.filelist
chillispot_0.96-1_mipsel.ipk
dhcp-fwd_0.7-1_mipsel.ipk
dropbear_0.44test3_mipsel.ipk
fprobe_1.0.5-1_mipsel.ipk
ip6tables_1.2.11-1_mipsel.ipk
ip_2.0_mipsel.ipk
kmod-ipt6_2.4.20-1_mipsel.ipk
kmod-ipv6_2.4.20-1_mipsel.ipk
kmod-nfs_2.4.20-1_mipsel.ipk
kmod-ppp-async_2.4.20-1_mipsel.ipk
kmod-ppp-mppe-mppc_2.4.20-1_mipsel.ipk
kmod-sched_2.4.20-1_mipsel.ipk
kmod-tun_2.4.20-1_mipsel.ipk
libmatrixssl_1.2.1-1_mipsel.ipk
libpcap_0.8.3-1_mipsel.ipk
libpthread_0.9.26-1_mipsel.ipk
libssl_0.9.7d-1_mipsel.ipk
ntpclient_2003.194-1_mipsel.ipk
oidentd_2.0.7_mipsel.ipk
openssh-client-extras_3.8p1-1_mipsel.ipk
openssh-client_3.8p1-1_mipsel.ipk
openssh-server_3.8p1-1_mipsel.ipk
openssh-sftp-client_3.8p1-1_mipsel.ipk
openssh-sftp-server_3.8p1-1_mipsel.ipk
ppp-radius-plugin_2.4.2-1_mipsel.ipk
ppp_2.4.2-1_mipsel.ipk
pppoecd_1.0_mipsel.ipk
pptp-client_1.5.0-1_mipsel.ipk
pptp-server_1.1.3-1_mipsel.ipk
radvd_0.7.2-1_mipsel.ipk
strace_4.5.6-1_mipsel.ipk
tc_2.0_mipsel.ipk
vsftpd_1.2.2-1_mipsel.ipk
wondershaper_1.1a_mipsel.ipk
zlib_1.1.4-1_mipsel.ipk

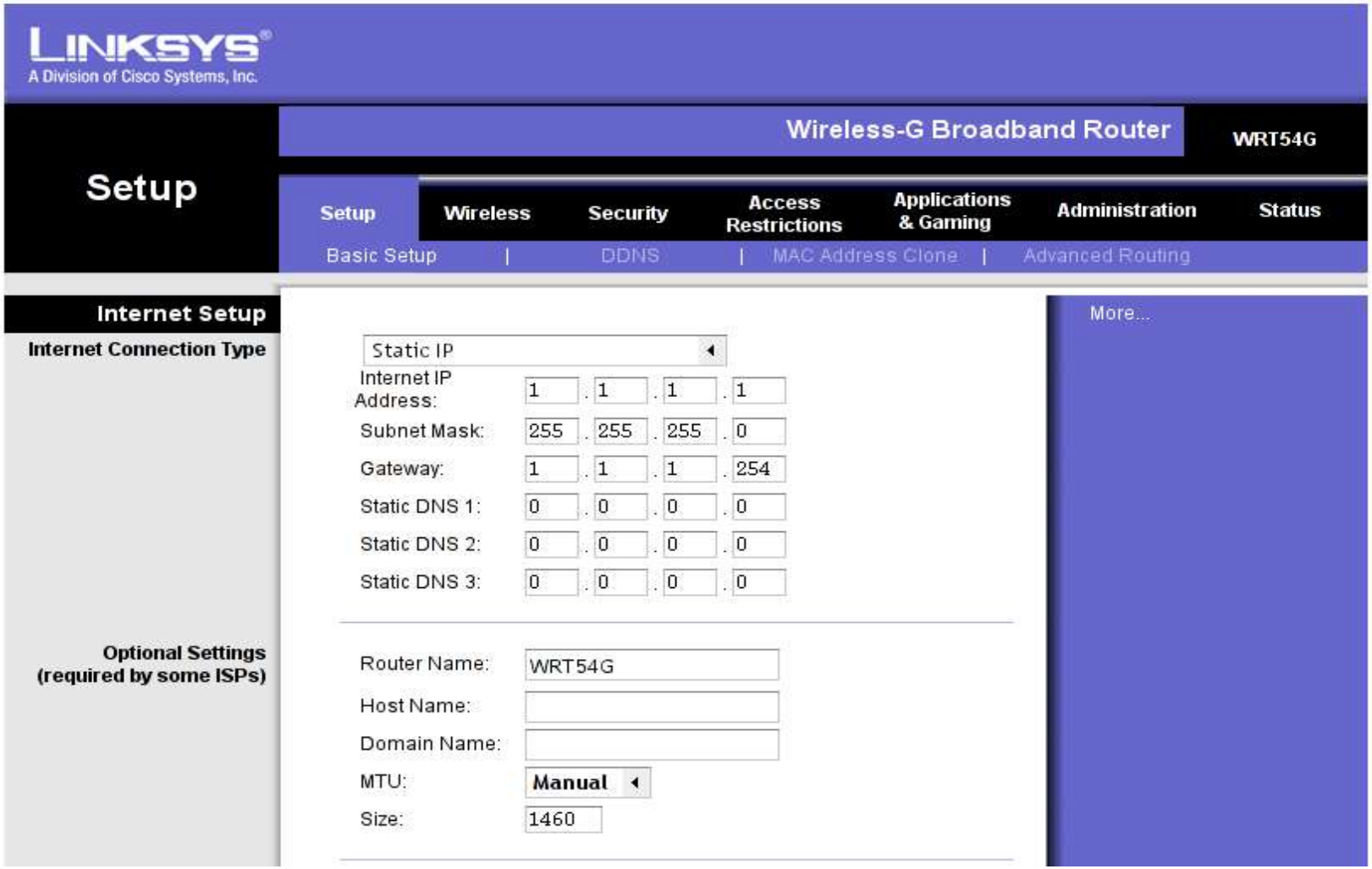
Installation auf dem Router

- Es wird ein tftp-Client benötigt, der ein Passwort mit übermittelt. Einen passenden Client gibt es z.B. bei http://redsand.net/projects/linksys-tftp/linksys-tftp.php
- Gewünschtes Firmware-File ins tftp-Verzeichnis kopieren und „**code.bin**“ nennen.

Installation auf dem Router

- boot_wait im NVRAM auf „on“ setzen (auf einem neuen Router steht es auf „off“)
  - Dafür sorgen, daß der Router auf dem WAN-Interface eine IP hat (ggf. eine fest konfigurieren)
  - Mit dem Browser auf folgende Seite gehen: http://192.168.1.1/Ping.asp
  - Folgende „Ziele“ pingen:
    - `;cp${IFS}*/*/nvram${IFS}/tmp/n`
    - `;*/n${IFS}set${IFS}boot_wait=on`
    - `;*/n${IFS}commit`
    - `;*/n${IFS}show>tmp/ping.log`

Einstellung der festen IP:



Installation auf dem Router

- Dem PC (falls nicht schon geschehen) eine feste IP geben - 192.168.1.10/24 ist z.B. gut
- Router ausschalten / Stecker ziehen
- Router wieder einschalten

```
cd@hinklebaan ~/linksys $ ./linksys-tftp 192.168.1.1
TJ Shelton      redsand [at] redsand.net
Mike Lynn       abaddon [at] 802.11ninja.net
Linksys TFTP Client for *BSD/Linux      The Firmware gets sexier
Modified Berkeley TFTP client Release: !(@) 1.2.1 (10/01/03)

linksys-tftp> put code.bin admin
Sent 3036160 bytes in 2.8 seconds
linksys-tftp>
```

Installation auf dem Router

- Jetzt **W-A-R-T-E-N** – mindestens drei Minuten bzw. so lange bis die Switch-LEDs wieder einen Reset anzeigen
- Wenn man die Reset-Taste hinten am Gerät drückt während der Boot-Loader aktiv ist, dann wird „boot_wait“ wieder auf „off“ gesetzt!!! -> Linksys -> Mülltonne
- Sobald die Power-LED nicht mehr blinkt, ist der Bootloader nicht mehr aktiv

Erste Kontakt-Aufnahme

```
cd@hinklebaan ~ $ telnet 192.168.1.1
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.


BusyBox v1.00 (2004.11.20-20:28+0000) Built-in shell (ash)
Enter 'help' for a list of built-in commands.


 _______                     ________        __
|       |.-----.-----.-----.|  |  |  |.----.|  |_
|   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
|_______||   __|_____|__|__||________||__|  |____|
         |__| W I R E L E S S   F R E E D O M

@OpenWrt:/#
```

Überprüfen, ob das Flash richtig formatiert wurde:

```
@OpenWrt:/# mount
/dev/root on / type squashfs (ro)
none on /dev type devfs (rw)
none on /proc type proc (rw)
/dev/mtdblock/4 on /jffs type jffs2 (rw)
none on /tmp type ramfs (rw)
@OpenWrt:/#
```

Falls diese Zeile **NICHT** erscheint, dann die Schritte auf der folgenden Folie durchführen

Flash von Hand formatieren bzw. Notfallbetrieb:

- Router aus- und wieder einschalten
- Warten bis die DMZ-LED an gegangen ist
- **Sofort** Reset-Taste an der Rückseite drücken
- Einen weiteren Moment warten

Formatierung der jffs2-Partition im Flash anstoßen

```
cd@hinklebaan ~ $ telnet 192.168.1.1
Trying 192.168.1.1...
Connected to 192.168.1.1.
Escape character is '^]'.

@OpenWrt:/# mount
/dev/root on / type squashfs (ro)
none on /dev type devfs (rw)
none on /proc type proc (rw)
none on /tmp type ramfs (rw)
@OpenWrt:/# firstboot
Creating jffs2 partition... done
creating directories... done
setting up symlinks... done
@OpenWrt:/#
```

Installation des Dropbear-SSH-Dienstes:

- /etc/ipkg.conf anpassen (Link nach /rom/etc/ipkg.conf löschen und neue Datei mit folgendem Inhalt anlegen – vi ist vorhanden)

```
src openwrt http://192.168.1.10
dest root /
dest ram /tmp
```

- Auf dem entsprechenden Rechner sollte die URL über einen http-Server freigegeben sein

```
@OpenWrt:/etc# ipkg update
Downloading http://192.168.1.10/Packages ...
Connecting to 192.168.1.10[192.168.1.10]:80
Packages             100%  11245       00:00 ETA
Done.
Updated list of available packages in /usr/lib/ipkg/lists/openwrt
@OpenWrt:/etc#
```

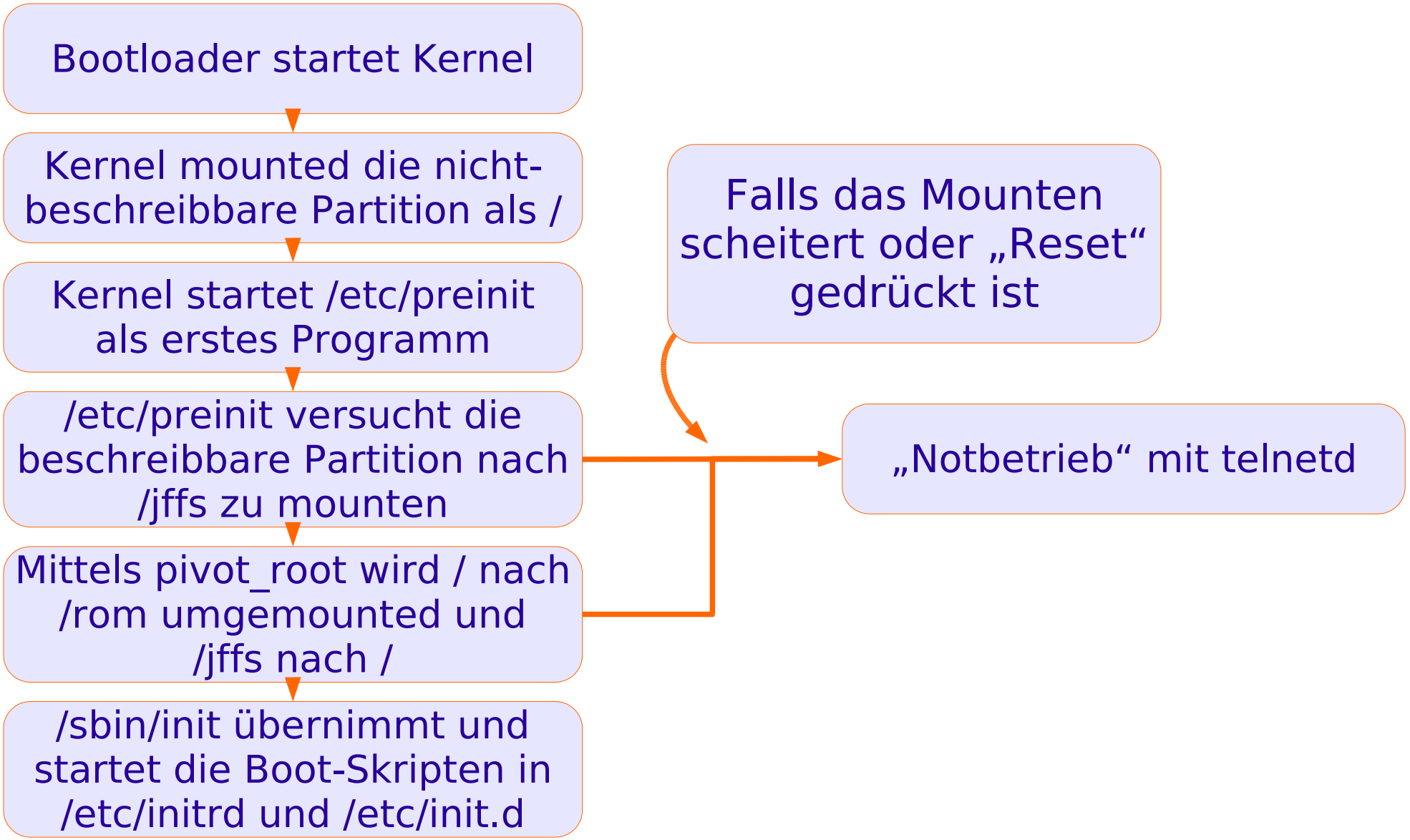
Installation des Dropbear-SSH-Dienstes:

```
@OpenWrt:/etc# ipkg install dropbear

Downloading http://192.168.1.10/dropbear_0.44test3_mipsel.ipk ...
Connecting to 192.168.1.10[192.168.1.10]:80
dropbear_0.44test3_m 100%    125 KB    00:00 ETA
Done.
Unpacking dropbear...Done.
Configuring dropbear...
Will output 1024 bit dss secret key to '/etc/dropbear/dropbear_dss_host_key'
Generating key, this may take a while...
Public key portion is:
ssh-dss AAAA...
Changing password for root
Enter new password:
Re-enter new password:
Password changed.
========================================
dropbear is now configured to run on boot.
========================================
To start it now, run the following command:
/usr/bin/dropbear

Done.
```

Inhalt

Linksys Wrt54g

Bootvorgang



Boot-Skripten in /etc/init.d:

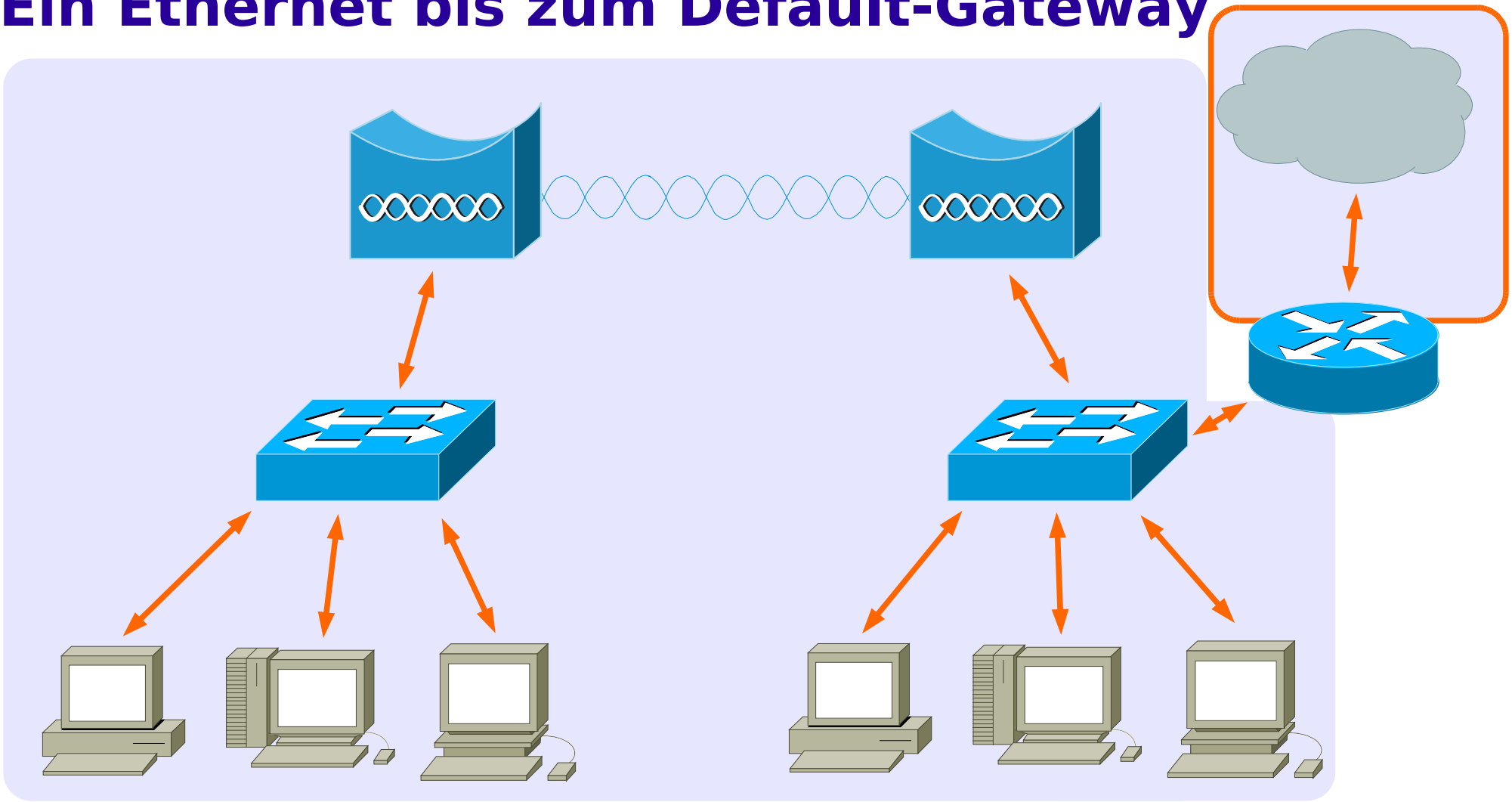
```
.
..
S10boot -> /rom/etc/init.d/S10boot
S40network -> /rom/etc/init.d/S40network
S45firewall -> /rom/etc/init.d/S45firewall
S50dnsmasq -> /rom/etc/init.d/S50dnsmasq
S50httpd -> /rom/etc/init.d/S50httpd
S50telnet -> /rom/etc/init.d/S50telnet
S99done -> /rom/etc/init.d/S99done
rcS -> /rom/etc/init.d/rcS
```

- Möchte man ein Skript verändern, dann den Link löschen und das Skript in die beschreibbare Partition kopieren:

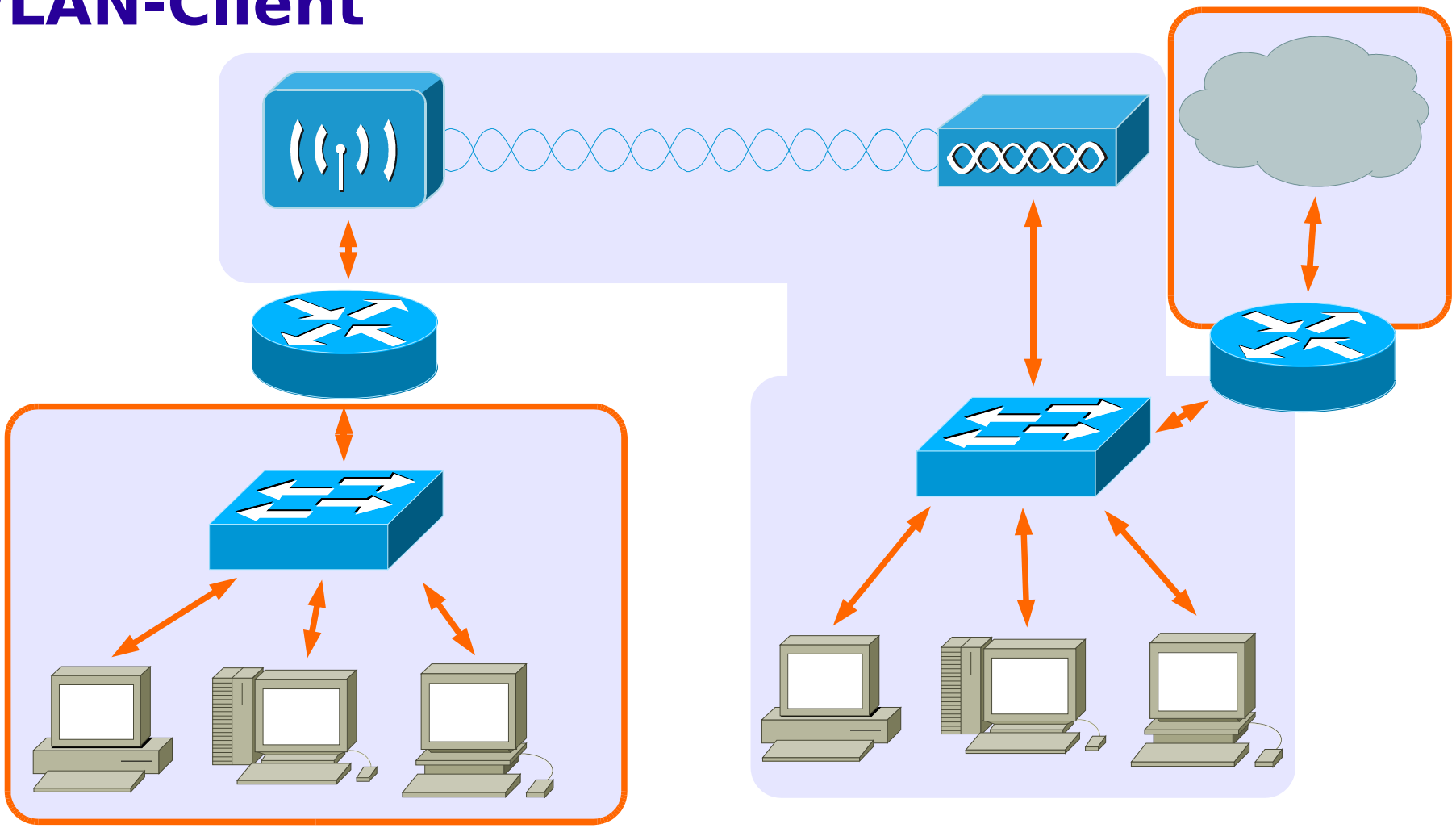
```
@OpenWrt:/etc/init.d# rm S40network
@OpenWrt:/etc/init.d# cp -p /rom/etc/init.d/S40network .
@OpenWrt:/etc/init.d# vi S40network
```

Inhalt

Ein Ethernet bis zum Default-Gateway



Über einen zusätzlichen Router auf den WLAN-Client



Inhalt

Linksys Wrt54g — WLAN-Interface

- Das WLAN-Interface im Wrt54g unterstützt vier unterschiedliche Betriebsarten:

Access-Point

Das Interface kann dazu eine IP haben oder direkt auf ein Draht-Ethernet gebridgt werden

WLAN-Client

Das Interface braucht dann sicher eine IP, zusätzliche Rechner werden über NAT (Routing!) angebunden

WDS-Mode

Im AP-Mode können zusätzlich WDS-Partner angegeben werden, hinter denen sich ein ganzes Ethernet verbirgt.

Monitor-Mode

Jaja, Kismet läuft auch...

Linksys Wrt54g WLAN-Interface

- Leider ist das „**wl**“-Kommando, das die Original-Firmware zum Konfigurieren des WLANs benutzt nicht im OpenWrt enthalten, sondern muß von Hand auf den Router kopiert werden (z.B. mittels netcat)

  Das Tool verbirgt sich im Build-Tree unter
  build_mipsel/WRT54GS/release/src/router/mipsel-uclibc/install/utils/usr/sbin/wl

```
root@OpenWrt:~# wl
Usage: wl [-a|i <adapter>] [-hu] <command> [arguments]

  -a, -i        adapter name or number
  -h, -u        this message

ver     get version information

up      reinitialize and mark adapter up (operational)

down    reset and mark adapter down (disabled)
```

- „wl“ bietet ca. 10000 unterschiedliche Befehle und Optionen an, weswegen für Details auf die eingebaute Anleitung und Experimentieren verwiesen werden muß.
- Zwei Beispiele werden noch gezeigt: Eine sinnvolle Anwendung und ein Hack :-)

Linksys Wrt54g WLAN-Interface

- Anzeige der Netze in der Umgebung und einbuchen als Client

```
root@OpenWrt:~# wl ap 0
root@OpenWrt:~# wl scan
root@OpenWrt:~# wl scanresults
SSID: "chrisnet"
Mode: Managed   RSSI: -84 dBm   noise: -92 dBm  Channel: 11
BSSID: 00:0F:66:24:DE:41        Capability: ESS
Supported Rates: [ 1(b) 2(b) 5.5(b) 11(b) 18 24 36 54 6 9 12 48 ]


root@OpenWrt:~# wl join chrisnet
root@OpenWrt:~# wl status
SSID: "chrisnet"
Mode: Managed   RSSI: -88 dBm   noise: -92 dBm  Channel: 11
BSSID: 00:0F:66:24:DE:41        Capability: ESS
Supported Rates: [ 1(b) 2(b) 5.5(b) 6 9 11(b) 12 18 24 36 48 54 ]


root@OpenWrt:~#
```

- Steigerung der Sendeleistung (Vorsicht! Überhitzung!)

```
root@OpenWrt:~# wl txpwr 84
root@OpenWrt:~# wl txpwr
84
root@OpenWrt:~#
```

- Weniger ist manchmal mehr: Es hat sich gezeigt, daß manche Geräte sehr hitzeempfindlich sind – und daß sie bei niedrigerer Sendeleistung besser funktionieren!

Inhalt

Ausblick

- Aufbau eines großen WLAN-Netzes als Internet-Zugang inklusive dynamischem Routing auf Basis von OSPF
- Verwendung der einzelnen Ports des Switches als getrennte Ethernet-Devices (indem man die Ports in eigene VLANs steckt – im Netz gibt es dazu das admcfg-Tool)
- Benutzung des Routers als billige RS232-Bridge um z.B. Zugangskontroll-Systeme oder Solaranlagen anzubinden
- Aufbau eines integrieten WLAN-Hotspots mit DSL-Einwahl, Benutzer-Authentisierung und Konten-Management - auf Basis eines VPN (z.B. OpenVPN)
  -> realisiert als wVPN-Projekt

VIELEN DANK - Fragen?

Wir schauen genau hin

- Finden des USB-Host-Ports an der CPU und Aufbau eines MP3-Players :-)