Dieser Artikel ist veraltet!
Seit Januar 2017 findet Ihr unter **Projekte → Cubietruck → Neuauflage #2** oder hier eine komplett überabeitet Anleitung.

Cubietruck als headless Video Disk Recorder (VDR)

Vorbereitungen

Zum Kompilieren der verschiedenen VDR Bestandteile und für die Zusammenstellug des Gesamtsystems auf der einer SD-Karte wird ein Host-System benötigt. Ich verwende ein Dell Vostro 3560 mit Intel Core i7 und 8GB RAM als Host-System auf dem ein Ubuntu 12.04 LTS 64bit läuft.
Um z.B. den Cubietruck Kernel kompilieren zu können sollten mindestens die folgende Pakete installiert sein:

```
sudo su
apt-get install git build-essential gcc-arm-linux-gnueabi libncurses5-dev u-boot-tools qemu-user-static
apt-get install debootstrap binfmt-support libusb-1.0-0-dev pkg-config gcc-arm-linux-gnueabihf
```

Info-Quellen

Die meisten Informationen für die erfolgreiche Einrichtung meines Cubietruck und des darauf laufenden VDR habe ich aus folgenden Quellen bezogen:

- http://linux-sunxi.org
- http://cubieboard.org/
- https://github.com/linux-sunxi/u-boot-sunxi/wiki
- http://linux-sunxi.org/FirstSteps
- http://linux-sunxi.org/U-boot
- http://www.debian.org/
- http://www.vdr-wiki.de/wiki/index.php/Hauptseite

Sundtek:

- http://wiki.sundtek.de/index.php/Main_Page

Installationssystem

Als Medium für die Installation eines Debian 7 (Codename „Wheezy“) auf dem Cubietruck soll eine Micro-SD-Karte dienen. Ich setze hier eine Micro-SD-Karte mit zwei Partitionen und min. 2GB Speicherplatz voraus. Mit folgenden Schritten entsteht ein bootfähiges Ausgangssystem auf der Micro-SD-Karte:

SD-Karte vorbereiten

Die SD-Karte (hier `/dev/sdf`) muss wie folgt partitioniert und die ersten 2048 Blöcke der SD-Karte sollten blank geputzt sein:

```
Gerät  boot.     Anfang        Ende     Blöcke   Id  System
/dev/sdf1   *        2048       34815       16384   83  Linux
/dev/sdf2           34816     7774207     3869696   83  Linux
```

Die erste Prtition ist 16 MB groß, die zweite erstreckt sich über den gesamten Rest. Dabei nimmt der unpartitionierte Bereich (bis Block 2047) die folgenden Daten auf:

```
Sector Start Größe Kommentar:
   0      0    8KB Partitionstabelle
  16      8   32KB Initial SPL loader
  80     40  504KB u-boot  (sector 64 / 32KB for 2013.07 and earlier)
1088    544  128KB environment
1344    672  128KB Falcon mode boot params
1600    800   ---- Falcon mode kernel start
2048   1024   16MB ab hier beginnt dann /dev/sdf1
```

Die folgende Kommandozeile erzeugt diese Partitionierung. Natürlich muss `/dev/sdfx` durch das tatsächlich verwendete Gerät ersetzt werden 😉
Das Kommando 'partprobe' veranlasst das Host-Betriebssystem die Änderungen am Partitionsaufbau der Micro-SD-Karte neu einzulesen.

```
dd if=/dev/zero of=/dev/sdf bs=1M count=1
(echo o; echo n; echo p; echo 1; echo; echo "+16M"; echo n; echo p; echo 2;
echo; echo; echo w) | fdisk /dev/sdf >> /dev/null
partprobe /dev/sdf
```

Anschließend die neuen Partitionen formatieren:

```
mkfs.vfat -n boot /dev/sdf1
mkfs.ext4 -L root /dev/sdf2
```

Aus Kompatibilitätsgründen ist für die erste Partition das Dateisystem VFAT unumgänglich.

Der Bootloader u-boot

Zum Starten von der SD-Karte benötigt man einen Bootloader. Folgende Schritte zeigen, wie man sich seinen Bootloader besorgt, kompiliert und auf die SD-KArte an die richtige Stelle bringt.

```
sudo su
cd ~
mkdir -p cubietruck
cd ~/cubietruck
git clone git://github.com/linux-sunxi/u-boot-sunxi.git
cd u-boot-sunxi/
make Cubietruck -j4 CROSS_COMPILE=arm-linux-gnueabihf-
```

Der fertige Bootloader liegt dann unter `u-boot-sunxi/u-boot-sunxi-with-spl.bin`
Bootloader auf die SD-Karte kopieren:

```
dd if=u-boot-sunxi-with-spl.bin of=/dev/sdf bs=1024 seek=8
cd ..
```

Anschließend die Partitionen der SD-Karte ins Dateisystem einhängen:

```
mkdir -p /media/boot
mkdir -p /media/root
mount /dev/sdf1 /media/boot
mount /dev/sdf2 /media/root
```

boot.scr erzeugen

Ohne die Datei `boot.scr` weiß der Bootloader nicht was (uImage aka. Kernel) er von wo (Adresse 0x48000000) starten soll…

```
cat <<EOT > boot.cmd
setenv bootargs console=ttyS0,115200 root=/dev/mmcblk0p2 rootwait panic=10
fatload mmc 0 0x43000000 script.bin
fatload mmc 0 0x48000000 uImage
bootm 0x48000000
EOT
```

```
mkimage -C none -A arm -T script -d boot.cmd boot.scr
cp boot.scr /media/boot/
```

script.bin erzeugen

Die Datei `script.bin` enthält alle Konfigurationsparameter für den A20 Prozessor des Cubietruck. Mit ihr wird festgelegt an welchen Pins des Prozessors welche Peripherie angeschlossen ist. Ebenso lässt sich über diese Datei z.B. der VGA Port permanent ein- oder ausschalten oder die SPI Schnittstelle konfigurieren. An dieser Konfigurationsdatei sollte man nur rumbasteln wenn man genau

weiß was man tut - anderenfalls läuft nix mehr.

```
git clone git://github.com/linux-sunxi/sunxi-tools.git
cd sunxi-tools
make fex2bin
# Cross compile for ARM Plattform:
# make clean && make -j2 'fex2bin' CC=arm-linux-gnueabihf-gcc && make -j2
'bin2fex' CC=arm-linux-gnueabihf-gcc && make -j2 'nand-part' CC=arm-linux-
gnueabihf-gcc
cd ..
cp sunxi-tools/nand-part /media/root/sbin/
./sunxi-tools/fex2bin my_cubietruck.fex script.bin
cp script.bin /media/boot/
```

uEnv.txt erzeugen

Über die Datei `uEnv.txt` werden dem Kernel beim Startvorgang wichtige Parameter wie z.B. das Dateisystem der Root-PArtition (hier ext4) mitgeteilt.

```
cat <<EOT > uEnv.txt
root=/dev/mmcblk0p2
extraargs=console=ttyS0,115200 sunxi_no_mali_mem_reserve
sunxi_g2d_mem_reserve=0 sunxi_ve_mem_reserve=0 hdmi.audio=EDID:0
disp.screen0_output_mode=EDID:1280x720p50 rootwait
panic=10 rootfstype=ext4 rootflags=discard
EOT
cp uEnv.txt /media/boot/
```

Der Kernel

Jetzt kompilieren wir einen eigenen Kernel für den Cubietruck.
Die Pfeile (→) bezeichnen das „Abtauchen“ un Untermenues des Kernel Konfiguraionswerkzeuges `menuconfig`.
Der Zusatz `-j4` bei `make -j4 …` erlaubt dem Kompiler vier CPU-Kerne gleichzeitig zu verwenden. Das beschleunigt den Kompiliervorgang erhebich. Wer nicht so viele oder mehr hat muss den Parameter anpassen.

```
git clone -b sunxi-3.4 https://github.com/linux-sunxi/linux-sunxi.git
cd linux-sunxi/
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- sun7i_defconfig
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig  # ->
CONFIG_JUMP_LABEL auf no setzen
  -> earlyprintk -- disable earlyprintk
     -> Kernel Hacking -> [ ] Kernel low level debugging functions (ganz
unten, vorletzter Punkt)
  -> Ethernet Treiber fest in Kernel kompilieren
     -> Device drivers -> Networking device support -> Ethernet driver
support -> <*> Allwinner Ethernet MAC Support
```

```
     -> Device drivers -> Networking device support -> Ethernet driver
support -> <*> Sunxi Platform 10/100/1000Mbps Ethernet drivers
  -> Device drivers -> Block devices -> <*> SUNXI Nandflash Driver
  -> Device drivers -> Real Time Clock -> <M> Micro Crystal RTC
  -> grep -i sunxi .config ...  ;-)

make clean && make -j4 ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- uImage
modules
make -j4 ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- INSTALL_MOD_PATH=output
modules_install
make -j4 ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- INSTALL_HDR_PATH=output
headers_install
cp arch/arm/boot/uImage /media/boot/
mkdir -p /media/root/lib
cp -R output/lib/firmware /media/root/lib/
rsync -ar --no-links output/lib/modules /media/root/lib/
cd ..
```

Das Root Dateisystem RootFs

Das eigene Root File System (RootFS) schafft die gewünschte Unabhängigkeit von Fremdherstellern - man weiß ja nie was die eifrigen Chinesen so alles in iren RootFS verstecken 😉

```
debootstrap --verbose --arch armhf --variant=minbase --foreign wheezy
/media/root http://ftp.debian.org/debian
cp /usr/bin/qemu-arm-static /media/root/usr/bin/
chroot /media/root
/debootstrap/debootstrap --second-stage
exit
```

Hostname setzten:

```
echo "cubierescue" > /media/root/etc/hostname
```

Und ohne eine `sources.list` können wir später keine Pakete nachinstallieren.

```
echo "deb http://ftp.de.debian.org/debian stable main contrib non-free" >>
/media/root/etc/apt/sources.list
echo "deb-src http://ftp.de.debian.org/debian stable main contrib non-free"
>> /media/root/etc/apt/sources.list
echo "deb http://ftp.debian.org/debian/ wheezy-updates main contrib non-
free" >> /media/root/etc/apt/sources.list
echo "deb-src http://ftp.debian.org/debian/ wheezy-updates main contrib non-
free" >> /media/root/etc/apt/sources.list
echo "deb http://security.debian.org/ wheezy/updates main contrib non-free"
>> /media/root/etc/apt/sources.list
echo "deb-src http://security.debian.org/ wheezy/updates main contrib non-
free" >> /media/root/etc/apt/sources.list
echo "###"  >> /media/root/etc/apt/sources.list
```

```
echo "deb http://http.debian.net/debian wheezy-backports main contrib non-
free" >> /media/root/etc/apt/sources.list
```

Die Datei `fstab` erzeugen. Der zweite Eintrag hängt eine SATA HDD für die VDR-Aufnahmen unter `/mnt/video0` ein.

```
echo "/dev/mmcblk0p2 / ext4 defaults 0 1" >> /media/root/etc/fstab
echo "/dev/sda1 /mnt/video0  ext4 defaults 0 1" >> /media/root/etc/fstab
```

Gigabit LAN aktivieren, d.h. das Laden des Moduls `sunxi-gmac` in die Datei `modules` eintragen.

```
echo "sunxi-gmac" >> /media/root/etc/modules
#WLAN mit bcmdhd funktioniert mit dem Kernel aktuell nicht, benötige ich
aber auch nicht ;-)
#echo "bcmdhd" >> /media/root/etc/modules
```

Ein paar Spracheinstellungen vornehmen und wichtige System-Pakete instalieren. Dazu per `chroot` in die Systemumgebung des neuen Systems wechseln.

```
chroot /media/root/
export LANG=C
apt-get update
apt-get install apt-utils dialog locales
dpkg-reconfigure locales
export LANG=de_DE.UTF-8
apt-get install dhcp3-client udev netbase ifupdown iproute openssh-server
iputils-ping wget net-tools ntpdate vim nano less tzdata console-tools
module-init-tools mc
apt-get install i2c-tools aptitude psmisc wireless-tools wpasupplicant
dosfstools rsync debootstrap bc sysv-rc-conf rsyslog
#./etc/init.d/ntp stop
passwd
exit
```

MAC-Adresse für die LAN Schnittstelle anpassen und WLAN ausschalten: `vi /media/root/etc/network/interfaces`

```
auto lo eth0
allow-hotplug eth0
iface lo inet loopback
iface eth0 inet dhcp
  hwaddress ether b6:f3:87:4c:17:72

#allow-hotplug wlan0
#iface wlan0 inet dhcp
#    wpa-ssid SSID
#    wpa-psk xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
# to generate proper encrypted key: wpa_passphrase yourSSID yourpassword
```

`getty` für Login starten:

```
echo "T0:2345:respawn:/sbin/getty -L ttyS0 115200 vt100" >>
/media/root/etc/inittab
```

SSH Key übertragen

```
ssh-copy-id -i ~/.ssh/KEYNAME.pub root@<CUBIETRUCK-IP>
```

NTPD stoppen

Falls der NTP Deamon `ntp` mit installiert wurde sollte er gestoppt und wieder deinstalliert werden. Ich möchte keinen „parasitären“ Zeitserver im lokalen Netz haben - einer Reicht 😉

```
service ntp stop
update-rc.d -f ntp remove
aptitude remove ntp
```

LEDs

Um die LED unter Kontrolle zu bringen helfen folgende Schritte weiter:

Möglichkeiten erfahren

```
cat /sys/class/leds/blue\:ph21\:led1/trigger
[none] battery-charging-or-full battery-charging battery-full battery-
charging-blink-full-solid ac-online usb-online mmc0 mmc1 timer heartbeat
cpu0 cpu1 default-on
 cat /sys/class/leds/orange\:ph20\:led2/trigger
[none] battery-charging-or-full battery-charging battery-full battery-
charging-blink-full-solid ac-online usb-online mmc0 mmc1 timer heartbeat
cpu0 cpu1 default-on
cat /sys/class/leds/white\:ph11\:led3/trigger
[none] battery-charging-or-full battery-charging battery-full battery-
charging-blink-full-solid ac-online usb-online mmc0 mmc1 timer heartbeat
cpu0 cpu1 default-on
cat /sys/class/leds/green\:ph07\:led4/trigger
[none] battery-charging-or-full battery-charging battery-full battery-
charging-blink-full-solid ac-online usb-online mmc0 mmc1 timer heartbeat
cpu0 cpu1 default-on
```

LEDs ausschalten

In `/etc/rc.local` eintragen:

```
# damit der User "vdr" die LEDs auch ein/ausschalten darf:
```

```
chmod -R 777 /sys/class/leds/blue:ph21:led1/
chmod -R 777 /sys/class/leds/orange:ph20:led2/
chmod -R 777 /sys/class/leds/white:ph11:led3/
chmod -R 777 /sys/class/leds/green:ph07:led4/

# alle LEDs ausschalten
echo none > /sys/class/leds/blue\:ph21\:led1/trigger
echo none > /sys/class/leds/orange\:ph20\:led2/trigger
echo none > /sys/class/leds/white\:ph11\:led3/trigger
echo none > /sys/class/leds/green\:ph07\:led4/trigger
```

LEDs separat einschalten

```
echo 1 > /sys/class/leds/blue\:ph21\:led1/brightness
```

ACPI konfigurieren

Um den Cubietruck per Power-Taste herunter zu fahren benötigt man ACPI.

```
aptitude search acpi
apt-get install  acpid
```

Quellen :

- https://groups.google.com/forum/#!msg/cubieboard/eNWzeyDSDYc/4O5TkWY_6xAJ
- https://groups.google.com/forum/#!topic/cubieboard/YxMJhGEfVWk

Stromsparfunktionen nachrüsten

Damit die Festplatte nicht im Dauerbetrieb läuft kann man mit Hilfe der 'laptop-mode-tools' ein sog. Spin-down nach z.B. 5 minuten einstellen. Das spart im IDLE Zustand knapp 500 mW.

```
aptitude search laptop-mode-tools
```

Sundtek DVB-C USB-Stick Installieren

```
wget http://www.sundtek.de/media/sundtek_netinst.sh
chmod +x sundtek_netinst.sh
./sundtek_netinst.sh
/opt/bin/mediaclient -e
/opt/bin/mediaclient -D DVBC
# oder
/opt/bin/mediaclient -d /dev/dvb/adapter[0/1]/frontend0 -m DVBC -D DVBC
```

Tuner in Bulk-Modus versetzen und danach neu anschließen:

```
/opt/bin/mediaclient --dtvtransfermode=bulk -d /dev/dvb/adapter0/frontend0
/opt/bin/mediaclient --dtvtransfermode=bulk -d /dev/dvb/adapter1/frontend0
```

Signalqualität beurteilen:

```
/opt/bin/mediaclient --readsignal=0 -d /dev/dvb/adapter0/frontend0
/opt/bin/mediaclient --readsignal=0 -d /dev/dvb/adapter1/frontend0
```

Check, ob Tuner von `vdr` verwendet wird:

```
/opt/bin/mediaclient --lc
```

Mediaclient

… starten

```
/opt/bin/mediaclient --start
```

… stoppen

```
/opt/bin/mediaclient --shutdown
```

VDR Installation

VDR Version 1.7.28 erkennt problemlos und ohne weitere Konfiguration zwei Sundtek MediaTV Pro III (USB) zur Aufnahme oder Live-Straming von SD/HD DVB-C Sendern.

Konfigurationsdateien und Verzeichnisse

- `/usr/lib/vdr/config-loader.sh` - hier liegt die standard Konfiguration für vdr, sie wird am Ende des Sktiptes mit den in `/etc/default/vdr` gefundenen Parametern überschrieben oder erweitert.
- `/etc/vdr/` und `/var/lib/vdr/` - hier liegen die wichtigsten Konfigurationsdateien
- `/usr/share/vdr/` - command-, recording- und shutdown-hooks

Das VDR Grundsystem installieren

```
aptitude install vdr vdr-plugin-dvbhddevice vdr-plugin-dvbsddevice vdr-
plugin-epgsearch vdr-plugin-femon vdr-plugin-streamdev-server vdr-plugin-
svdrposd vdr-plugin-svdrpservice vdradmin-am vdr-plugin-remote
```

Hostnamen setzten

```
echo "vdr" > /etc/hostname
```

VDR starten oder stoppen

```
cd /; ./etc/init.d/vdr start && ./etc/init.d/vdradmin-am start
cd /; ./etc/init.d/vdr stop && ./etc/init.d/vdradmin-am stop
```

VDR logging anpassen

vi `/etc/default/vdr`

```
...
# set logging level
# 0 = no logging
# 1 = errors only
# 2 = errors and info
# 3 = errors, info and debug (default)
LOGLEVEL=0
```

vi `/etc/init.d/vdr` und `–log $LOGLEVEL` ergänzen

```
startvdr()
{
    if [ "$ENABLED" != "0" ] ; then
        # only start vdr if there is no other instance running
        if start-stop-daemon --start --startas $DAEMON --test \
            --name $(basename $DAEMON) --pidfile $PIDFILE >/dev/null
        then
            getplugins
            mergecommands "commands"
            mergecommands "reccmds"
            configure_console_input

            if [ "$VFAT" == "1" ]; then
                OPTIONS="--vfat $OPTIONS"
            fi

            start-stop-daemon --start --quiet --startas $DAEMON --background
\
                --name $(basename $DAEMON) --pidfile $PIDFILE --make-pidfile
-- \
                -v $VIDEO_DIR -c $CFG_DIR -L $PLUGIN_DIR -r $REC_CMD \
                -s $VDRSHUTDOWN -E $EPG_FILE -u $USER -g /tmp \
                --port $SVDRP_PORT $OPTIONS $PLUGINS $REDIRECT --log
$LOGLEVEL
        else
            echo -n " - seems to be running already"
        fi
```

```
    else
        echo -n " - aborted (to enable the daemon, edit /etc/default/vdr)"
    fi
}
```

Log-Meldungen landen dann entweder unter `/var/log/messages` oder unter `/var/log/user.log`
Wie man alle Log-Meldungen des vdr in eine einzige Datei umleitet steht hier:
http://www.loggn.de/ubuntu-debian-video-disk-recorder-logs-in-einer-extra-datei/.

VDR Konfiguratonsdatei

Meine `setup.conf` unter `/etc/vdr/` (link auf `/var/lib/vdr/setup.conf`

setup.conf

```
AntiAlias = 1
AudioLanguages =
ChannelEntryTimeout = 1000
ChannelInfoPos = 0
ChannelInfoTime = 5
ChannelsWrap = 1
CurrentChannel = 1
CurrentDolby = 0
CurrentVolume = 255
DefaultLifetime = 99
DefaultPriority = 50
DelTimeshiftRec = 0
DeviceBondings =
DiSEqC = 0
DisplaySubtitles = 0
EmergencyExit = 1
EPGBugfixLevel = 3
EPGLanguages =
EPGLinger = 60
EPGScanTimeout = 1
FoldersInTimerMenu = 1
FontFix = Courier:Bold
FontFixSize = 15
FontFixSizeP = 0.031000
FontOsd = Sans Serif:Bold
FontOsdSize = 18
FontOsdSizeP = 0.038000
FontSml = Sans Serif
FontSmlSize = 17
FontSmlSizeP = 0.035000
InitialChannel = C-1-1101-28106
InitialVolume = -1
InstantRecordTime = 180
LnbFrequHi = 10600
```

```
LnbFrequLo = 9750
LnbSLOF = 11700
MarginStart = 2
MarginStop = 10
MarkInstantRecord = 1
MaxVideoFileSize = 2000
MenuKeyCloses = 0
MenuScrollPage = 1
MenuScrollWrap = 0
MinEventTimeout = 15
MinUserInactivity = 0
MultiSpeedMode = 0
NameInstantRecord = TITLE EPISODE
NextWakeupTime = 0
NumberKeysForChars = 1
OSDAspect = 1.000000
OSDHeight = 403
OSDHeightP = 0.840000
OSDLanguage =
OSDLeft = 58
OSDLeftP = 0.080000
OSDMessageTime = 1
OSDSkin = sttng
OSDTheme = default
OSDTop = 38
OSDTopP = 0.080000
OSDWidth = 624
OSDWidthP = 0.870000
PauseKeyHandling = 2
PauseLifetime = 1
PausePriority = 10
PrimaryDVB = 1
RecordingDirs = 1
ResumeID = 0
SetSystemTime = 0
ShowInfoOnChSwitch = 1
ShowRemainingTime = 0
ShowReplayMode = 0
SplitEditedFiles = 0
StandardCompliance = 0
SubtitleBgTransparency = 0
SubtitleFgTransparency = 0
SubtitleLanguages =
SubtitleOffset = 0
SVDRPTimeout = 300
TimeoutRequChInfo = 1
TimeSource =
TimeTransponder = 0
UpdateChannels = 0
UseDolbyDigital = 1
UseSmallFont = 1
```

```
UseSubtitle = 1
UseVps = 0
VideoDisplayFormat = 1
VideoFormat = 0
VpsMargin = 120
ZapTimeout = 3
epgsearch.UseSearchTimers = 1
live.LocalNetMask = 192.168.0.0/16
streamdev-server.AllowSuspend = 1
streamdev-server.HideMenuEntry = 0
streamdev-server.HTTPBindIP = 0.0.0.0
streamdev-server.HTTPPriority = 0
streamdev-server.HTTPServerPort = 3000
streamdev-server.HTTPStreamType = 0
streamdev-server.IGMPBindIP = 0.0.0.0
streamdev-server.IGMPClientPort = 1234
streamdev-server.IGMPPriority = 0
streamdev-server.IGMPStreamType = 0
streamdev-server.LoopPrevention = 0
streamdev-server.MaxClients = 5
streamdev-server.ServerPort = 2004
streamdev-server.StartHTTPServer = 1
streamdev-server.StartIGMPServer = 0
streamdev-server.StartServer = 1
streamdev-server.SuspendMode = 1
streamdev-server.VTPBindIP = 0.0.0.0
streamdev-server.VTPPriority = 0
vnsiserver3.TimeshiftBufferFileSize = 3
vnsiserver3.TimeshiftBufferSize = 2
```

DVB-C Kanalliste erstellen

```
aptitude install w-scan
./etc/init.d/vdr stop && ./etc/init.d/vdradmin-am stop
w_scan -fc -c DE >> /var/lib/vdr/channels-DVB-C.conf
cp /var/lib/vdr/channels-DVB-C.conf /var/lib/vdr/channels.conf
./etc/init.d/vdr stop && ./etc/init.d/vdradmin-am stop
```

Standard Video Aufnahmeverzeichnis festlegen

```
echo "VIDEO_DIR=\"/mnt/video0\"" >> /etc/default/vdr
```

TCP-Port für OSD per Telnet konfigurieren

```
echo "-p tcp:2001" > /etc/vdr/plugins/plugin.remote.conf
```

SAMBA installiren und einrichten

```
aptitude install samba
echo "   wins server = eth0:192.168.100.1" > /etc/samba/dhcp.conf
adduser christoph
smbpasswd -a christoph
aptitude install cifs-utils
```

Samba Konfiguration

smb.conf

```
#======================= Global Settings =======================

[global]
   workgroup = WG
   server string = %h server
   wins support = no

   wins server = 192.168.100.1
   dns proxy = no
   name resolve order = lmhosts host wins bcast

#### Networking ####
;   interfaces = 127.0.0.0/8 eth0
;   bind interfaces only = yes


#### Logging ####
   log file = /var/log/samba/log.%m

# Cap the size of the individual log files (in KiB).
   max log size = 1000

   syslog = 0
   panic action = /usr/share/samba/panic-action %d


####### Authentication #######

   security = user
   encrypt passwords = true
   passdb backend = tdbsam
   obey pam restrictions = yes
   unix password sync = yes
   passwd program = /usr/bin/passwd %u
   passwd chat = *Enter\snew\s*\spassword:* %n\n
*Retype\snew\s*\spassword:* %n\n *password\supdated\ssuccessfully* .
```

```
   pam password change = yes
   map to guest = bad user

#======================= Share Definitions =======================
[homes]
   comment = Home Directories
   browseable = yes
   writeable = yes
   read only = no
   create mask = 0700
   directory mask = 0700
   valid users = %S

[Aufnahmen]
   comment = VDR Aufnahmen
   path = /mnt/video0
   browseable = yes
   writeable = yes
   read only = no
   #create mask = 0700
   #directory mask = 0700
   #valid users = %S
   create mask = 0664
   directory mask = 0775
   force group = users

 load printers = no
 printing = bsd
 printcap name = /dev/null
 disable spoolss = yes
```

VDR als XBMC-Backend einrichten

Informationsquellen

VDR als Backend für Raspberry Pi (RaspBMC)

- http://backports.debian.org/Instructions/
- http://www.vdr-portal.de/board18-vdr-hardware/board98-arm-co/121033-vdr-auf-rasberry-problem-mit-vnsiserver-gel%C3%B6st/

VNSI Plugin installieren

```
echo "deb http://http.debian.net/debian wheezy-backports main" >>
/etc/apt/soutces.list
```

```
apt-get update
apt-get install vdr-plugin-vnsiserver
ln -s /var/lib/vdr/plugins/vnsiserver/allowed_hosts.conf
/var/lib/vdr/plugins/vnsiserver3/allowed_hosts.conf
```

alte Dokumente und z.Zt. ungenutztes

NAND-Installation - Installation im NAND-Flash des Cubietruck - aktuell nicht funktionsfähig, wird aber weiter verfolgt...
RTC - eine externe RTC für den Cubietruck - aktuell nicht funktionsfähig, wird aber weiter verfolgt...

```
-----------------------------------------------------------------
 /etc/network/interfaces
   iface eth0 inet dhcp
    hwaddress ether b6:f3:87:4c:17:71

-----------------------------------------------------------------
mount /dev/mmcblk0p1 /mnt/
bin2fex script.bin tmp.fex
vi ...
[dynamic]
MAC = "b6f3874c1771"

fex2bin tmp.fex script.bin
-----------------------------------------------------------------

http://www.igorpecovnik.com/2013/12/24/cubietruck-debian-wheezy-sd-card-imag
e/

dd if=cubietruck-debian-wheezy-2013-12-26.raw of=/dev/sdf bs=1M count=1000

dpkg --get-selections

-----------------------------------------------------------------
dd if=ct-debian-nand.img of=/dev/sdf bs=1024 count=1798741

-----------------------------------------------------------------
ps -AH | grep keyring
ssh-copy-id -i ~/.ssh/cubietruck.wg.pub root@cubietruck.wg

mount /dev/nanda /mnt/boot/ && mount /dev/nandb /mnt/rootfs/

cd ~ && mount /dev/nandb /mnt/rootfs && TIMESTAMP=`date "+%Y%m%d_%H:%M"` &&
tar -zcvf $TIMESTAMP_rootfs.tar.gz /mnt/rootfs && umount /mnt/rootfs
-----------------------------------------------------------------
```

From:
http://xn--vonthlen-b6a.de/ - **Christophs DokuWiki**

Permanent link:
http://xn--vonthlen-b6a.de/doku.php/wiki/projekte/cubietruck/uebersicht

Last update: **2018/01/19 12:54**