

# Minix Neo X5 Projektseite

## Projektziel

Basierend auf dem Artikel aus c't 2014 Heft 4 (S.166ff) soll ein Minix Neo X5 (im Folgenden nur noch Minix genannt) mit einem Debian-System bestückt werden. Auf dieser Basis soll der Minix als Mini-Server im lokalen Netz dienen und folgende Dienste bereitstellen:

- owncloud (zum Sync von Adressen/Kalender mit Android-Geräten und Thunderbird-Clients unter Windows/Linux)
- Dateifreigaben (mit samba)

Die Vorteile des Gerätes gegenüber einem Raspberry Pi sind:

- ähnlicher Energieanforderungen
- keine Lüfter
- integrierter Flash (16GB s.u.)
- relativ viel RAM (1GB s.u.)
- relativ hohe Rechenpower (1.4GHz Dual Core)

Nachteil: keine Erweiterbarkeit (ohne das Gehäuse zu modifizieren). Für den Einsatzbereich ist dies aber kein echter Nachteil.

## Vorbereitung oder wie läuft das Ganze ab?

Da das Projekt nicht „plug 'n' play“ ist, sondern relativ viel Handarbeit bedarf ist es sinnvoll sich vorab mit dem Prozedere auseinander zu setzen. Im Folgenden werden die einzelnen Schritte grob zusammengefasst, um einen Überblick zu geben.

1. Debian-VM aufsetzen, in der alle Arbeiten durchgeführt werden.<sup>1)</sup>
2. root-FS(Minix-FS)<sup>2)</sup> erzeugen, dies wird später das filessystem des Minix (offen)
3. Grundkonfigurationen innerhalb des Minix-FS vornehmen (hier kann das Script aus dem c't Artikel helfen) (offen)
4. kopieren des Minix-FS auf eine SD-Card kopieren (offen)
5. initramfs erzeugen, das als Boot-Loader fungiert und später den Kernel lädt (offen)
6. Kernel für Minix bauen (hört sich komplizierter an, als es ist 😊) (offen)
7. aus dem initramfs und dem neuen Kernel in ein Kernel-Image generieren. (offen)
8. Kernel-Image auf den Neo X5 flashen (offen)

Nun geht's aber endlich los.

## Minix-FS auf SD-Card kopieren

Zunächst muss der Name der SD-Card gefunden werden. Mit `lsblk` kann man den Namen finden oder man benutzt `dmesg` kurz nach dem einlegen der Karte. Man erhält eine ähnliche Ausgabe:

```
[ 6502.829511] sdb: sdb1
[ 6502.839812] sd 10:0:0:0: [sdb] No Caching mode page found
[ 6502.839838] sd 10:0:0:0: [sdb] Assuming drive cache: write through
[ 6502.839850] sd 10:0:0:0: [sdb] Attached SCSI removable disk
[ 6503.344347] EXT4-fs (sdb1): mounted filesystem with ordered data mode.
Opts: (null)
```

Bei mir wurde die SD-Card als sdb eingehängt.

Damit es nicht zu Fehlermeldung kommt sollte die SD-Karte zunächst ausgeworfen werden, falls diese bereits automatisch eingebunden wurde.

```
umount /media/USER/SDCARD
```

Für USER und SDCARD müssen selbstverständlich die entsprechenden Werte des eigenen Systems verwendet werden. (ggf. mit weiteren Partitionen auf der SD-Card wiederholen)

## Partitionieren

```
sudo fdisk /dev/sdb
```

Mit **p**<sup>3)</sup> werden alle Partionen angezeigt. So kann man überprüfen, ob das richtige Gerät ausgewählt wurde. Ansonsten zerstört man sich u.U. sein Betriebssystem!! Mit **d**<sup>4)</sup> (VORSICHT!!) können bereits vorhandene Partionen gelöscht werden. Nach **d** muss man per Ziffer die entsprechende Nummer der Partition angeben (s. **p**). Mit **n**<sup>5)</sup> kann eine neue Partition angelegt werden. Man wird gefragt, ob die Partition **p** (primär) oder **e** (extended) sein soll. Wir benötigen eine primäre Partition: also **p**. Falls noch weitere Partionen benötigt werden, sollte bei den nächsten Fragen entsprechender Platz freigelassen werden. Zu ersten Testzwecken ist es sinnvoll die gesamte SD-Card zu verwenden. Wir wählen demnach die Vorgaben als Parameter für den ersten und letzten Sektor.

## Filesystem anlegen

Ist das Partitionieren erledigt, dann wird das Filesystem eingerichtet. Es wird ein EXT4 benötigt. Als Name wird `linuxroot` vorausgesetzt.

```
sudo mkfs.ext4 -L linuxroot /dev/sdb1
```

## SD-Card mit VM verbinden

Dieser Schritt kann je nach System kompliziert oder sehr simpel werden. Der einfachste Weg ist es die SD-Card als USB-Gerät in die VM einzubinden. Fertig!

Der harte Weg: Die SD-Card wird NICHT als USB-Gerät im Host-System eingebunden, sondern als eigene Partition. Dann wird es komplizierter:

[englische Beschreibung](#) **Muss noch eingearbeitet werden**

## Technische Daten des Minix Neo X5

Hier die technischen Eckdaten des Minix Neo X5: (Auszug aus dem offiziellen Datenblatt):

Type	Beschreibung
Prozessor	Rockchip RK3066 Dual Core Cortex A9 1.4GHz (max. 1.6GHz)
GPU	Quad Core Mali 400 (OpenGL ES 2.0/1.1, Opven VG1.1, Flash 11.1)
RAM	1GB DDR3
Int. Speicher	16GB NAND Flash
Funkschnittstellen	802.11 b/g/n WiFi, Bluetooth, 3G über USB-Dongle (nicht enthalten)
OS	Android 4.1.1 Jelly Bean
Video Output	HDMI 1.4a, Full HD 1080p, 3D Filme unterstützt
Audio Output	HDMI 1.4a, optisch, S/PDIF, analog (Kopfhörer, Klinke)
Sonstige Anschlüsse	
	RJ-45 Ethernet (10/100 Mbit/s)
	SD/MMC Slot (SD 3.0, MMC 4.41)
	3x USB 2.0 Ports
	Micro USB OTG Port
	Infrarot-Empfänger ( Fernbedienung ist enthalten)
	Schlitz für Kensington Schloss
Stromversorgung	5V, 3A Adpater (enthalten), Gerät benötigt laut Hersteller weniger als 1A
Videoformate	AVI/RM/RMVB/MKV/WMV/MOV/MP4/WEBM/DAT(VCD format)/VOB/MPEG/MPG/FLV/ASF/TS/TP/3GP u.a.

1)

kann entfallen, wenn man ohnehin unter Debian arbeitet

2)

FS: filesystem

3)

p: print

4)

d: delete

5)

n: new

From:

<https://www.kopfload.de/> - **kopfload - Lad Dein Hirn auf!**

Permanent link:

[https://www.kopfload.de/doku.php?id=allgemein:minix:minix\\_debian&rev=1395770336](https://www.kopfload.de/doku.php?id=allgemein:minix:minix_debian&rev=1395770336)

Last update: **2025/11/19 16:13**

