Physik mit dem Raspberry Pi - Systemeinrichtung

29.04.2019, Sebastian Schmidt (sebastian.seb.schmidt@fau.de)

github.com/schmidtseb/RaspberryPiUebung

SD-Karte klonen

Ein bereits bestehendes System soll von einer SD-Karte zu einer anderen kopiert werden. Benötigt wird dabei stets ein PC als Zwischenstelle. Abhängig vom Betriebssystem gibt es nun einige Möglichkeiten, den Transfer des Systems vorzunehmen. Einige davon sind im Folgenden erklärt:

Linux / Mac

Alle Arbeitsschritte erfolgen hierbei im Terminal.

Schritt 1: SD-Karte einlegen und identifizeren

Die SD-Karte, die geklont werden soll, wird eingelegt.

Um nun Operationen an ihr vornehmen zu können, muss zuerst der vom Betriebssystem vergebene Gerätename herausgefunden werden.

Dies geschieht in Linux über den Befehl

```
sudo fdisk -l
```

und am Mac über

```
diskutil list
```

Eine Auflistung der am PC angeschlossenen Geräte sollte nun erscheinen. Die in unserem Fall benutzte SD-Karte sollte in Linux unter dem Namen

```
/dev/mmcblk0
```

und auf dem Mac unter dem Namen

```
/dev/diskX
```

erscheinen. Hierbei ist `X` eine Nummer, die abhängig von den bereits vorher angeschlossenen Geräten ist.

Je nach verwendeter Karte sind jedoch auch andere Gerätenamen denkbar!

Schritt 2: Dumpen der SD-Karte

Der komplette Inhalt der SD-Karte kann mittels des Befehls

```
sudo dd bs=4M status=progress if=<Gerätename> | gzip > <output_file.gz>
```

auf die Festplatte des PCs geschrieben werden.

- `sudo` wird dem Befehl vorangestellt, da Adminrechte für diesen Vorgang benötigt werden (bei falscher Verwendung kann das System beschädigt werden!)
- `dd` ist der eigentliche Kopierbefehl, dessen Parameter `bs` - die zu verwendente Blockgröße, `status=progress` - die Aufforderung zur Ausgabe des aktuellen Status des Kopierens und `if` - das einzulesende File sind. Hierbei muss `<Gerätename>` mit dem vorher bestimmten Gerätenamen ersetzt werden.
- `|` ist der Pipe-operator, der die Ausgabe von `dd` an `gzip` übergibt
- `gzip` ist ein Programm, das die ihm übergebenen Daten komprimiert. Über `>` wird die Ausgabe in ein File, hierbei `<output_file.gz>`, umgeleitet

Die Benutzung von `gzip` ist hierbei optional und dient nur dem Sparen von Speicherplatz. Möchte man das File nicht komprimieren, kann auch der Befehl

```
sudo dd bs=4M status=progress if=<Gerätename> of=<output_file.img>
```

verwendet werden, wobei `of` nun für das Ausgabefile steht.

Anschließend kann die SD-Karte sicher entfernt werden. Dazu muss man diese jedoch vorher unmounten:

```
sudo umount <Gerätename>
```

Schritt 3: Gegebenenfalls formatieren der neuen SD-Karte

Falls die SD-Karte, auf die das gedumpte Image nun geschrieben werden soll, vorher bereits verwendet wurde, empfiehlt es sich die Karte zu formatieren.

Dazu wird sie eingelegt und ihr Gerätename herausgefunden. Anhand diesem muss die Karte nun ungemountet werden.

Anschließend wird bei Linux über

```
sudo mkfs.msdos -F 16 <Gerätename>
```

und beim Mac über

```
sudo newfs_msdos -F 16 <Gerätename>
```

ein neues Filesystem auf die SD-Karte geschrieben.

Schritt 4: Schreiben auf neue SD-Karte

Das Schreiben erfolgt nun ähnlich wie das vorherige Lesen der SD-Karte über den Befehl

```
gzip -dc <output_file.gz> | sudo dd bs=4m status=progress of=<Gerätename>
```

Hierbei ist die Erklärung analog zum vorherigen Fall, nur dass nun die Reihenfolge getauscht ist und bei `dd` der Parameter `of` benutzt wird, der das Ausgabefile beschreibt, benutzt wird.

Wurde vorher auf `gzip` verzichtet, so muss der Befehl

```
sudo dd bs=4m status=progress if=<output_file.img> of=<Gerätename>
```

zum Schreiben auf die SD-Karte benutzt werden.

Windows

Beispielhaft wird der `Win32 Disk Imager` verwendet. Die Benutzung ist relativ selbsterklärend. Über einen Klick auf das Ordner-Icon kann das File, in das die SD-Karte gedumpt werden soll, ausgewählt werden. Über die nebenstehende Dropdown-Liste wird schließlich das entsprechende Laufwerk ausgewählt. Der Vorgang selbst wird dann über einen Klick auf `Read` gestartet.

Das Schreiben auf die Karte erfolgt analog. Das zu schreibende File und das entsprechende Laufwerk der SD-Karte werden ausgewählt und der Prozess über einen Klick auf `Write` gestartet.

Aktuelle Anmeldedaten

Die Verbindung zum RaspberryPi erfolgt über WLAN. Die aktuellen Anmeldedaten sind:

- SSID (service set identifier): `seminar01`
- Passwort: `raspberrypi01`

Nach erfolgreichem Verbinden erfolgt die Anmeldung über putty via ssh anhand der Daten:

- IP: `192.168.178.1`, Port: `24`
- Nutzername: `pi`
- Passwort: `raspberrypi01`

Über Linux/Mac kann die Anmeldung auch per ssh im Terminal über

```
ssh -Y pi@192.168.178.1
```

erfolgen.

Änderung der Anmeldedaten

Zum Ändern der Anmeldedaten wird im Folgenden das Programm `sed` benutzt. Dieses ermöglicht das Suchen und Ersetzen von Strings in einem File. Die Syntax ist hierbei

```
sed s/<Suchstring>/<Ersetzstring>/g <infile> > <outfile>
```

Auf das Ausgabefile `<outfile>` kann verzichtet werden, wenn man den Parameter `-i` - für inplace-Bearbeitung - benutzt.

Nun sollen die SSID zu `seminar0X` und die Passwörter zu `raspberrypi0X` geändert werden. Dabei ist `X` jeweils eure Gruppennummer.

Die Befehlskette wird nun am besten in ein File geschrieben, das dann anschließend einfach ausgeführt werden kann. Das Programm, das im Terminal die eigentliche Arbeit erledigt, ist `bash`. Skripte, die von `bash` ausgeführt werden, tragen üblicherweise die Endung `.sh`. Über die Shebang-Kombination `#!` in der ersten Zeile des Skripts kann vermittelt werden, was das auszführende Programm sein soll, d.h. in unserem Fall

```
#!/bin/bash
```

Die folgenden Befehle nehmen nun die eigentlichen Änderungen an Benutzernamen und Passwörtern vor:

- Die Änderung des Passworts erfolgt über:

  ```
  echo "pi:raspberrypiX" | chpasswd
  ```

- Der Hostname, d.h. die SSID die im WLAN ersichtlich ist und die zugehörigen Passwörter müssen über folgende Befehle geändert werden

  ```
  sed -i s/seminar01/seminar0X/g /etc/hostapd/hostapd.conf
  sed -i s/raspberrypi01/raspberrypi0X/g /etc/hostapd/hostapd.conf
  sed -i s/channel\=1/channel\=X/g /etc/hostapd/hostapd.conf
  sed -i s/raspberrypi01/raspberrypi0X/g /etc/hostname
  sed -i s/raspberrypi01/raspberrypi0X/g /etc/hosts
  ```

Das geschriebene Skript `<Skriptname>` kann im Terminal nun über `chmod +x <Skriptname>` ausführbar gemacht werden. Gestartet wird es schließlich über

```
sudo ./<Skriptname>
```

Da Änderungen am System gemacht werden, ist das `sudo` zwingend notwendig!

Schließlich wird das System über `sudo reboot` neu gestartet.

Erweiterung der Partitionsgröße

Über `df -h` lassen sich alle aktuellen Partitionen und deren Größen anzeigen. Anhand dieser Werte wird ersichtlich, ob auch wirklich der ganze Speicher der SD-Karte ausgenutzt wird (16 GB in unserem Fall).

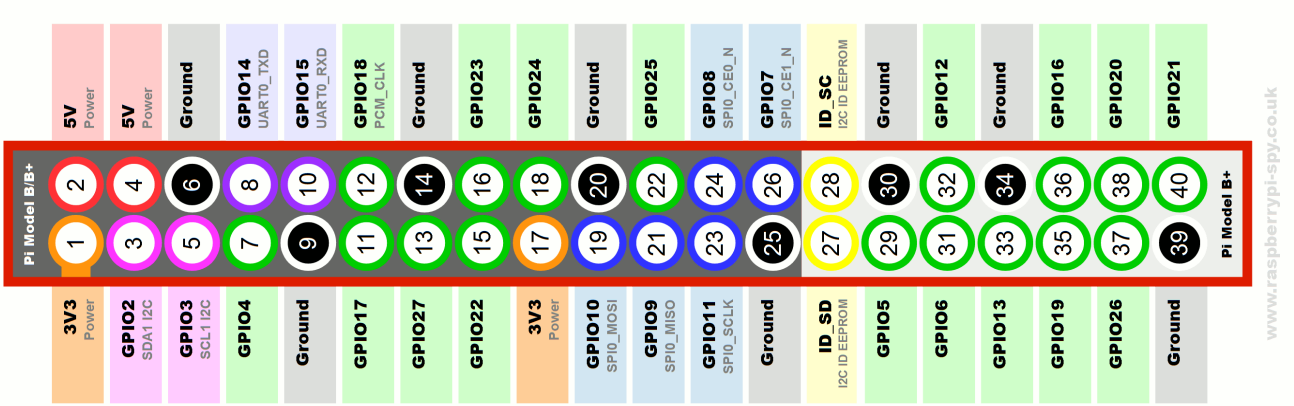
Ist dies nicht der Fall, so kann man eine Erweiterung der Partition vornehmen. Eine einfache Möglichkeit hierfür bietet das Tool `raspi-config`, das einfach über

```
sudo raspi-config
```

im Terminal aufgerufen wird. Durch Wählen der Option `Expand Filesystem` wird schließlich die Erweiterung der Partition vorgenommen.

Beispielprogramm

Eine LED wird an Pin 8 der GPIO angeschlossen und soll zum Blinken gebracht werden.



Dazu wird ein Python-Programm benutzt. Wie vorher beim `bash`-Skript wird über die Shebang-Zeichenfolge `#!` Python als auszuführendes Programm angegeben:

```
#!/usr/bin/env python
import RPi.GPIO as GPIO
from time import sleep

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(8, GPIO.OUT, initial=GPIO.LOW)

while True:
    GPIO.output(8, GPIO.HIGH)
    sleep(1)
    GPIO.output(8, GPIO.LOW)
    sleep(1)
```

Kurze Erklärungen zum Programm:

- Über `import` können zusätzliche Module, die neue Funktionen bereitstellen, geladen werden
- Das Modul `GPIO` regelt die Kommunikation der gleichnamigen Schnittstelle mit der Außenwelt
- In einer Endlosschleife wird der Pin der LED abwechselnd für eine Sekunde an- und abgeschaltet