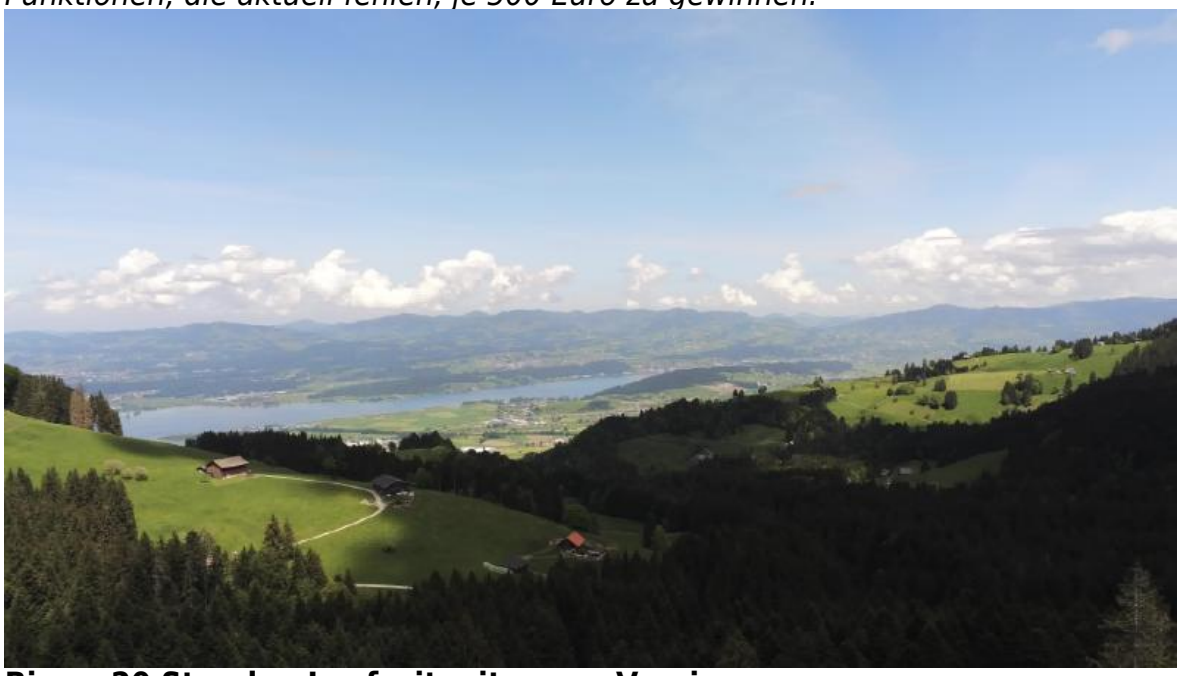
AVMultiPhone verbessern und gewinnen
Egg, 30. Juni 2020: *Im nachfolgenden Blog geht es darum, was die neue Version von AVMultiPhone bietet. Weiter wird aufgezeigt, wie AVMultiPhone an eigene Bedürfnisse angepasst werden kann und zudem gibt es für zwei Funktionen, die aktuell fehlen, je 500 Euro zu gewinnen.*



Bis zu 20 Stunden Laufzeit mit neuer Version
In der neuen Version von 30. Juni ist der Schlafmodus (deep sleep) enthalten. Damit läuft das AVMultiPhone ohne Nachladen bis zu 20 Stunden. Der Schlafmodus wird durch Drücken der Power-Taste aktiviert. Dies im übrigen nur dann, wenn das AVMutliPhone im Batteriemodus läuft. Wird das AVMultiPhone über USB geladen, ergibt der Schlafmodus keinen Sinn, da genügend Energie zur Verfügung steht.
Durch nochmaliges Drücken auf die Power-Taste erwacht das Telefon wieder aus dem Tiefschlaf. Dabei steht direkt der Desktop zur Verfügung. Wer AVMultiPhone entsperren möchte, erreicht dies damit, indem zuvor der Schirm gesperrt wird (Lock-Screen im Hauptmenü). Abhängig von der Länge des «Tiefschlafs» kann es notwendig sein, WLAN erneut freizuschalten (viertes Icon von links). Und noch etwas, wer den Tiefschlaf über die Konsole auslösen möchte, kann dies über ‚pm-suspend' tun.
Kamera «obscura»
Ebenfalls im aktuellen Release enthalten sind die notwendigen Kernel-Module für die Hauptkamera. Über das Menü ‚Graphics' und ‚Camera' lässt sich jederzeit ein Foto machen. Allerdings geht im Moment einzig die Auflösung 1280×720 Pixel, und eine Vorschau gibt es ebenfalls nicht. Das mag wenig erscheinen, doch läuft die Kamera aktuell auf keiner anderen PhinePhone-Distribution.
Das aktuelle Kamera-Skript sei hier publiziert:

```
#!/bin/bash

pfad="/home/archivista/Desktop"
dat=`date +'%Y-%m-%d-%T'`
file="$pfad/camera-$dat.jpg"
/usr/bin/media-ctl -d /dev/media1 --set-v4l2 '"ov5640
3-004c":0[fmt:UYVY8_2X8/1280x720]'
/usr/bin/ffmpeg -s 1280x720 -f video4linux2 -i /dev/video1 -
vframes 1 $file
chown archivista.archivista $file
```

Die Kamera selber ist in der Lage, Bilder mit 5 Mio. Pixeln aufzuzeichnen, doch fehlen dazu aktuell die entsprechenden Informationen. Leider lassen sich die Parameter nicht einfach mit anderen Pixelzahlen anpassen und eine Übergabe wie «ov5640 3-004c»:0[fmt:UYVY8…] ist nur nicht selbsterklärend.



AVMultiPhone an eigene Bedürfnisse anpassen

Zunächst, AVMultiPhone kann beliebig an eigene Bedürfnisse angepasst werden. Die Frage stellt sich eher, wie kann verhindert werden, aus Versehen das System zu beschädigen bzw. Sicherungen anzulegen. Zunächst aber, um effizient zu arbeiten, wird aufgezeigt, wie mit Linux-Standard-Tools auf AVMultiPhone zugegriffen werden kann.

Zugriff über SSH und VNC

Im Unterschied zu anderen PinePhone-Distributionen ist der Zugriff zunächst nicht beschränkt. Es stehen sowohl SSH wie VNC zur Verfügung. Am einfachsten ist der Zugriff über das USB-Kabel. Sobald das AVMultiPhone angehängt wird, wird in AVMultiPhone die interne IP-Adresse 172.16.42.1 festgelegt. Idealerweise (sprich auf dem Hauptrechner) kann einfach usb0 aktiviert werden, und der SSH-Zugang steht. Befehl dazu:

```
ifconfig usb0 172.16.42.2 up
ssh root@172.16.42.1
```

Das ‚root'-Passwort lautet 1234. Ebenfalls möglich ist der Zugriff mit dem Benutzer ‚archivista'. Dieser Benutzer kannverwendet werden, um grafische Programme zu starten. Dabei einzig darauf achten, dass ‚export DISPLAY=:0' jeweils gesetzt wird. Beispiel für das Einrichten des VNC-Servers (archivista-User):

```
export DISPLAY=:0
x11vnc -passwd archivista
```

Danach kann vom Hostrechner aus die Verbindung aufgebaut werden:

```
vncviewer -encodings 'tight copyrect' 172.16.42.1
```

Selbstverständlich können auch andere VNC-Programme verwendet werden und auch der Zugriff über WLAN ist möglich. Aus diesem Grunde darf gerne empfohlen werden, die Default-Passwörter zu ändern bzw. den Dienst ssh unterwegs auch abzuschalten. Dazu kann der Befehl ‚/etc/init.d/ssh stop' verwendet werden bzw. ‚rc-update del ssh default', um SSH permanent zu deaktivieren.

Software installieren

AVMultiPhone basiert auf pmOS (postmarketOS) bzw. Alpine Linux. Damit stehen

alle Pakete zur Verfügung, die es dort auch gibt. Diese werden mit dem ‚apk'-Paketmanager verwaltet. Folgende Befehle sind dabei nützlich:
apk info Listet alle installierten Pakete auf
apk list –verbose Auflisten aller verfügbaren Pakete
apk search xxx Suchen nach dem Paket xxxx
apk add –force –allow-untrusted xxx Installieren des Paktes xxx (auch lokal)
apk del xxx Entfernen des Paketes
apk update Paketquellen aktualisieren
apk upgrade Neu verfügbare Pakete installieren (auf eigenes Risiko)
System sichern
Der Befehl ‚apk upgrade' kann leider zum Teil- oder Gesamtverlust des Systems führen, weil nicht immer alle Pakete aufeinander abgestimmt sind. Besonders heikel ist dies bei ‚u-boot-pinephone' und beim Kernel ‚linux-postmarketos-allwinner'. Beide überschreiben die Datei ‚/boot/sun50i-a64-pinephone-1.1.dtb'. Dies kann dazu führen, dass danach z.B. kein Ton oder keine SIM-Karte mehr verfügbar ist. Abhilfe verschafft aktuell das Rückkopieren der ausglieferten ‚dtb'-Datei:
cp /boot/sun50i-a64-pinephone-1.1.old cp /boot/sun50i-a64-pinephone-1.1.dtb
Danach ist ein Neustart durchzuführen, Ton und SIM-Karte sollten nun wieder arbeiten. Nebenbemerkung zur Problematik: Sofern die Sachlage richtig verstanden wurde, müsste es beim PinePhone für jede Hardwareversion eine eigene ‚dtb'-Datei geben, doch fehlt dieses Feature aktuell unter pmOS.
Master-Image erstellen
Das Aufspielen eines neuen Images auf eine SD-Karte ist einfach. Abgesehen von **etcher.io,** das den Job grafisch erledigt, sei hier nochmals der Befehl für die Konsole vorgestellt:
gunzip -c postav_200630.img.gz | dd of=/dev/sdX bs=64M
Achtung: Dabei ist /dev/sdX durch den richtigen Buchstaben der SD-Karte zu ersetzen (z.B. /dev/sdh). Ein falscher Buchstabe zerstört das eigene System!
Etwas schwieriger wird es, wenn ein aktuelles Image gesichert werden soll. Das Kopieren des Datenträgers mit ‚dd if=/dev/sdX of=eins.img bs=64M' funktioniert zwar, führt aber zu unendlich grossen Dateien. Dazu wurde ein kleines Skript erstellt:

```
#!/bin/bash

umount /media/archivista/*
device="/dev/sdX"
mkdir -p /sdin
mount ${device}2 /sdin
rm -f /sdin/etc/NetworkManager/system-connections/*
echo "" >/sdin/root/.ssh/known_hosts
echo "" >/sdin/home/archivista/.ssh/known_hosts
rm -f /sdin/home/archivista/.local/share/calls/*
rm -f /sdin/root/.local/share/calls/*
rm /sdin/var/log/Xorg*
echo "" >/sdin/root/.ash_history
echo "" >/sdin/home/archivista/.ash_history
umount /sdin
zerofree ${device}2
dd if=$device of=$1 bs=512 count=14880767
gzip $1
```

Damit wird ein neues Master-Image erstellt. Auch hier gilt, dass /dev/sdX durch den richtigen Buchstaben ersetzt werden muss. Wichtig ist ferner, das die Profile für WLAN, Telefonate und SMS dabei gelöscht werden. Wer dies nicht möchte, kann die entsprechenden Befehle auskommentieren.



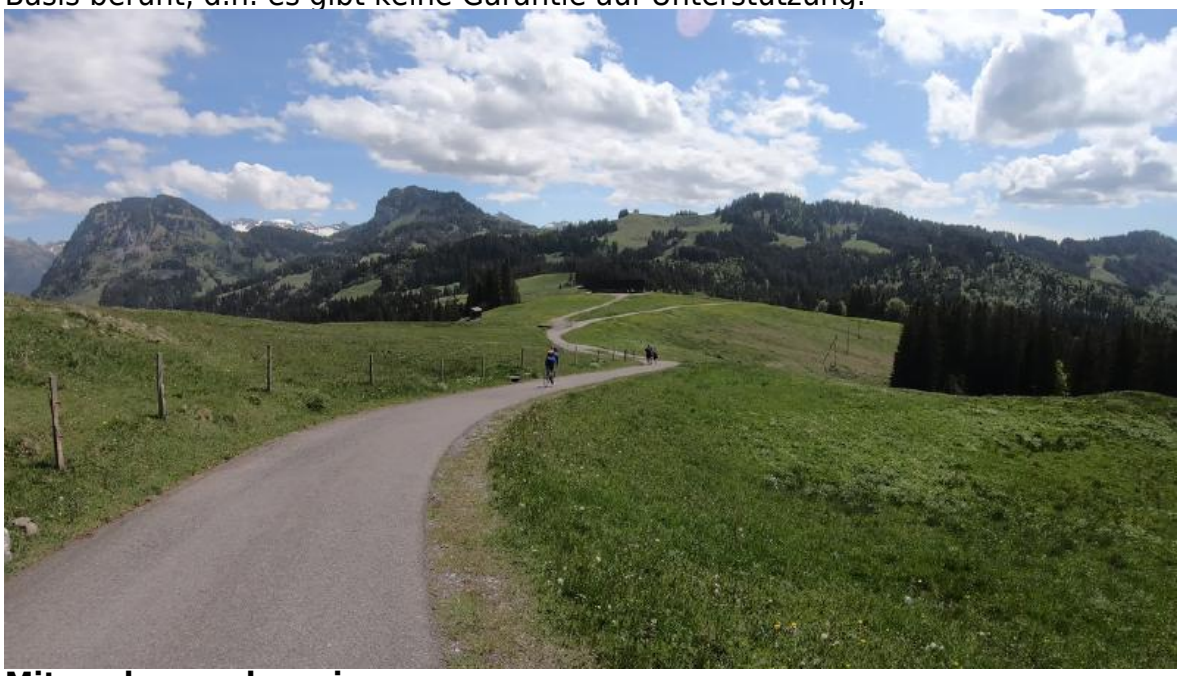
Entwickeln mit pmbootstrap
Wer AVMultiPhone tiefergreifend anpassen möchte, wird früher oder später zur Entwicklungsumgebung von postmarektOS greifen. Dabei handelt es sich um Python3-Skripte. Leider funktioniert der Installationsbefehl ‚pip3 install pmbootstrap' nicht immer. Daher wurde für AVMultimedia ein kleines Skript erstellt:

```
#!/bin/bash

cp -f sudoers /etc
apt-get update
apt-get -y install python3-pip
apt-get -y install git-core
apt-get -y install binfmt-support
pip3 install pmbootstrap
export PATH=/home/archivista/.local/pmbootstrap/bin:$PATH
```

Nachdem die Umgebung verhanden ist, kann mit **pmbootstrap init** gestartet werden. Falls alles klappt, sind einige Fragen zu beantworten, falls nicht, erfolgen Fehlermeldungen. Bei erfolgreichem Vorgang kann danach ein Image erstellt werden:
pmbootstrap install –sdcard=/dev/sdX
Auch hier gilt, einzusetzen ist der richtige Buchstabe. Weitere Befehle, die in der Entwicklungsumgebung hilfreich sind:
pmbootstrap aportgen –fork-alpine xxx Lokale Kopie für Anpassungen erstellen
pmbootstrap kconfig edit linux-postmarketos-allwinner Kernel-Datei verändern
pmbootstrap checksum linux-postmarketos-allwinner Prüfung durchführen
pmbootstrap build linux-postmarketos-allwinner –force Kernel neu erstellen
Die Optionen **checksum** und **build** lassen sich auf beliebige Pakete anwenden, Die Pakete selber werden über die Datei ‚APKBUILD' verwaltet. Beim Kernel liegt diese Datei im Unterordner des pmOS-Hauptordners:
~/pmOS/cache_git/pmaports/main/linux-postmarektos-allwinner
Allenfalls gibt es auch (für geforkte und neue Pakete) Kopien im temp-Ordner:
~/pmOS/cache_git/pmaports/temp/xxx

Wenn immer ein neues Paket erstellt werden soll, ist die Release-Nummer in APKBUILD zu erhöhen. Sofern der ‚build'-Befehl erfolgreich ist, kann das Paket in untenstehendem Order bezogen werden:
~/pmos/packages/edge/aarch64
Wer Hilfe benötigt, kann sein Glück im IRC-Channel #postmarketos versuchen. Ein webbasierte Zugriff ist über **https://freenode.net** (ohne Registrierung) möglich. Ebenfalls lässt sich der Kanal über **https://riot.im** (mit Registrierung) abrufen. Wichtig dabei zu wissen ist, dass die Hilfestellung immer auf freiwilliger Basis beruht, d.h. es gibt keine Garantie auf Unterstützung.



Mitmachen und gewinnen
Aktuell fehlen AVMultiPhone zwei Funktionen, die das Arbeiten deutlich angenehmer machen würden. Darum werden hier je 500 Euro Prämien ausgesetzt, um die beiden Funktionen zu erhalten.
Aufwachen aus Tiefschlaf aufgrund Anruf
Anrufe aus dem Tiefschlaf sind derzeit problemlos aus Mobian und UBPorts möglich. Bei AVMultiPhone bzw. pmOS ist dies aktuell nicht möglich. Gemäss einem UBPorts-Entwickler erfolgt das Aktivieren über die folgenden beiden Befehle:

```
echo "AT+QDAI=1,0,0,2,0,1,1,1" | atinout - /dev/EG25.AT -
echo 'AT+QCFG="risignaltype","physical"' | atinout -
/dev/EG25.AT -
```

Diese Befehle arbeiten aktuell auch unter AVMultiPhone, es wird ‚OK' zurückgemeldet. Telefonanrufe werden im Tiefschlaf auch angenommen, nur wird der Tiefschlaf nicht erfolgreich beendet (d.h. der Anruf kann nicht bemerkt werden). Vielmehr verharrt AVMultiPhone bis zum Drücken der Power-Taste im Tiefschlaf und klingelt danach kurz (der Anruf kommt daher im Teifschlaf an, es gibt im Moment einfach keine Möglichkeit, das Gespräch direkt aus dem Tiefschlaf anzunehmen.
Für die Prämie ist entweder ein AVMultiPhone-Image anzuliefern, bei dem das Aufwachen aus dem Tiefschlaf im Batteriemodus funktioniert oder es ist reproduzierbar zu dokumentieren, wie dies nachgerüstet werden kann.
Befehle, um Kamera-Modi anzusprechen
Die Hauptkamera wird über das Kernel-Modul ‚ov5640' realisiert. In den Sourcen dazu sind folgende Modi dokumentiert:

```
OV5640_MODE_QVGA_176_144,
OV5640_MODE_QVGA_320_240,
```

```
OV5640_MODE_VGA_640_480,
OV5640_MODE_NTSC_720_480,
OV5640_MODE_PAL_720_576,
OV5640_MODE_XGA_1024_768,
OV5640_MODE_720P_1280_720,
OV5640_MODE_1080P_1920_1080,
OV5640_MODE_QSXGA_2592_1944,
```

Diese Auflösungen müssen sowohl als Bild als auch als Video mit 15, 30 und 60 Bildern pro Sekunde verfügbar sein. Selbstvertändlich nur, sofern die Auflösung dies zulässt (z.B. kann der ov5640-Sensor nur HD mit 1920×1080 und 30 fps aufnehmen).
Auch hier gilt, entweder kann ein Image mit AVMultiPhone zur Verfügung gestellt werden oder es ist mit nachvollziehbaren Schritten aufzuzeigen, wie die Funktionalität in AVMultiPhone realisiert werden kann.
Alternativ können auch Offerten für die obigen oder anderen Jobs eingereicht werden. Angebote an **webmaster@archivista.ch** werden gerne wohlwollend und zügig bearbeitet.
Update vom 30. Juni 2020: *Die Prämie für das Ansprechen der Kamera ist vergeben, unter der Seite* ***https://blog.brixit.nl/camera-on-the-pinephone*** *finden sich alle notwendigen Informationen. Herzlichen Dank an den Entwickler Martijn Braam. Weitere Arbeiten für ein grafisches User-Interface für die Kamera dürfen gerne gemeldet werden, eine weitere Honorierung kann zwar nicht zugesichert werden, wird aber gerne geprüft.*
Update vom 2. Juli 2020: *Mit* ***postav_200701final.img.gz*** *werden Anrufe neu aus dem Schlafmodus entgegengenommen, allerdings beträgt die Laufzeit «nur» ca. 10 Stunden. Bis zu 20 Stunden gibt es aktull nur, ohne Auswachen aus dem Tiefschlaf, dazu gibt es* ***postav_200701deeptest.img.gz.*** *Mit anderen Worten, das Problems des Aufwachens besteht aktuell darin, dass der ‚deep'-Modus nicht (zuverlässig) arbeittet. Das «Preisgeld» für den deep-Modus bleibt offen.*
Update vom 28. Dezember 2020: *Leider bestehen bei pmOS (und damit aktuell auch bei AVMultiPhone) nach wie vor Probleme mit dem Tiefschlaf-Modus. Und weil die Entwickler von pmOS auch im Dezember 2020 weder wissen, woran es liegt, noch einen Zeitplan für die Behebung der Probleme haben, musste der Entscheid gefällt werden, dass AVMultiPhone aktuell nicht mehr weiterentwickelt wird.*