Bildschirmübertragung AirPlay mit Raspberry Pi realisieren

Die Anleitung basiert auf dem GitHub Projekt RPiPlay https://github.com/FD-/RPiPlay

Hinweis

RPiPlay setzt grundlegend auf die OpenMAX Video Library, um das Bild direkt über die Grafikkarte auszugeben. Die Bibliothek wird nur in der 32-bit Version und bis Raspbian Buster (10) unterstützt. Eine Unterstützung der neuen Bibliothek v4l2/mesa/drm gibt es derzeit leider noch nicht.

Installation und Einrichtung

1. Image Raspbian lite 32 all legacy 10 (buster) downloaden und auf SD-Karte kopieren, Bsp. mit balena-etcher. Die lite Version ohne grafischer Oberfläche ist ausreichend für das Projekt.
2. Raspberry mit eingelegter SD Karte starten und ggf. anmelden
3. `sudo apt-get update`
4. `sudo apt-get dist-upgrade` Updates durchführen
5. `sudo raspi-config`
localisation Timezone festlegen, Language de_DE.UTF-8, Keyboard, SSH enable, wifi-country Germany, memory split 256 festlegen
6. `sudo hostname -b {NEUER_NAME}` den Namen des Rechners/PiAppleTV Gerätes wählen, Bsp.: sudo hostname -b airplay-mgm-0
7. `sudo apt-get install git cmake libavahi-compat-libdnssd-dev libplist-dev libssl-dev` diverse Pakete installieren
8. Wird die alternative Videoausgabe verwendet werden noch folgende Pakete benötigt: `sudo apt-get install libx264-dev libjpeg-dev libgstreamer1.0-dev libgstreamer-plugins-base1.0-dev libgstreamer-plugins-bad1.0-dev gstreamer1.0-plugins-ugly gstreamer1.0-tools gstreamer1.0-gl gstreamer1.0-gtk3 gstreamer1.0-plugins-good`
8. `git clone https://github.com/FD-/RPiPlay.git` im Homeverzeichnis des aktuellen Benutzers
9. `cd RPiPlay`
10. `mkdir build`
11. `cd build`
12. `cmake --DCMAKE_CXX_FLAGS="-O3" --DCMAKE_C_FLAGS="-O3" ..`
Compilieren
13. `make -j`
14. `sudo make install` Installien unter /usr/local/bin/rpiplay
Testweise aktivieren von RPiPlay mit den Hostname als AirPlay Gerätename: `rpiplay -n $(uname -n) -b auto -vr rpi -ar rpi -a hdmi -l`
15. `sudo systemctl enable avahi-daemon.service`
ggf. optional Avahi aktivieren
16. `sudo nano /etc/systemd/timesyncd.conf`
ggf. NTP Zeitserver anpassen
>
[Time]
NTP=de.pool.ntp.org
FallbackNTP=0.debian.pool.ntp.org 1.debian.pool.ntp.org 2.debian.pool.ntp.org

3.debian.pool.ntp.org
17. `sudo nano /etc/fstab`
Logs ggf. in RAM auslagern um die Speicherkarte zu schonen. varlog und varrun als tmpfs einhängen.
>
```
tmpfs /tmp  tmpfs defaults,nodev,nosuid       0  0
none  /var/log  tmpfs size=10M,noatime,nodiratime       0  0
tmpfs /var/run  tmpfs size=10M,noatime,nodiratime       0  0
```

18. `sudo dphys-swapfile swapoff` Swapfile deaktivieren, um die Speicherkarte zu schonen, ggf. auch deinstallieren
#sudo dphys-swapfile uninstall #sudo update-rc.d dphys-swapfile remove
19. `sudo systemctl disable apt-daily.service`
`sudo systemctl disable apt-daily.timer`
`sudo systemctl disable apt-daily-upgrade.timer`
`sudo systemctl disable apt-daily-upgrade.service`
automatische Updates ggf. deaktivieren.
20. `sudo nano /boot/config.txt` Bootausgabe ausschalten, Raspberry Logo und Farbtestbildschirm ausschalten, Bluetooth deaktivieren, ggf. WiFi deaktivieren
>
```
# GPU Memory 256M
gpu_mem=256
# Rainbow Testscreen off
disable_splash=1
# Start HDMI even without screen
hdmi_force_hotplug=1
# Audio via HDMI
hdmi_drive=2
# DMT-Mode activated
hdmi_group=2
# fix resolution 1920x1080 / 60 Hz (1080p)
hdmi_mode=82
# uncomment to increase signal to HDMI, if you have interference, blanking, or no display
config_hdmi_boost=4
# Enable audio (loads snd_bcm2835)
dtparam=audio=on
# no Autostarting X
start_x=0
enable_uart=0
dtoverlay=disable-bt
#dtoverlay=disable-wifi

# Enable DRM VC4 V3D driver
#dtoverlay=vc4-kms-v3d #turn off with # or use vc4-fkms-v3d
#max_framebuffers=2 #turn off with #
[pi4]
# Enable DRM VC4 V3D driver on top of the dispmanx display stack
dtoverlay=vc4-fkms-v3d
max_framebuffers=2
```

21. `sudo nano /boot/cmdline.txt`
>
rootwait logo.nologo silent quiet loglevel=0 splash vt.global_cursor_default=0

Raspberry Logo off; no Bootup or Shutdown Messages

22. `sudo apt-get install fbi`
23. Datei ipad-rpiplay-appletv-screen.png in /opt/ ipad-rpiplay-appletv-screen.png kopieren, um Splash-Screen angezeigen zu können, solange keine Bildübertragung via AirPlay erfolgt.
24. `sudo nano /etc/update-motd.d/30-sysinfo`
Systemdaten ggf. nach Login anzeigen
>

```
#!/bin/sh

# Show Raspberry Pi Systeminfo

# Datum & Uhrzeit
DATUM=`date +"%A, %e %B %Y %H:%M:%S"`

# Hostname
HOSTNAME=`hostname -f`
#OS
OS1=`grep PRETTY /etc/os-release | cut -c 13-`
KERNEL=`uname -r`

# Letzter Login
LAST1=`last -1 -a | awk 'NR==1{print $3}'`    # Wochentag
LAST2=`last -1 -a | awk 'NR==1{print $5}'`    # Tag
LAST3=`last -1 -a | awk 'NR==1{print $4}'`    # Monat
LAST4=`last -1 -a | awk 'NR==1{print $6}'`    # Uhrzeit
LAST5=`last -1 -a | awk 'NR==1{print $10}'`   # Remote-Computer

# Uptime
UP0=`cut -d. -f1 /proc/uptime`
UP1=$(($UP0/86400))       # Tage
UP2=$(($UP0/3600%24))       # Stunden
UP3=$(($UP0/60%60))       # Minuten
UP4=$(($UP0%60))       # Sekunden

# Durchschnittliche Auslasung
LOAD1=`cat /proc/loadavg | awk '{print $1}'`    # Letzte Minute
LOAD2=`cat /proc/loadavg | awk '{print $2}'`    # Letzte 5 Minuten
LOAD3=`cat /proc/loadavg | awk '{print $3}'`    # Letzte 15 Minuten

# Temperatur
#TEMP=`sensors | grep temp | cut -c15-24`
TEMP=`vcgencmd measure_temp | cut -c 6-`

# Spannung
VOLTAGE=`vcgencmd measure_volts | cut -c 6-`
```

```
# Speicherbelegung
DISK1=`df -h | grep 'dev/root' | awk '{print $2}'`    # Gesamtspeicher
DISK2=`df -h | grep 'dev/root' | awk '{print $3}'`    # Belegt
DISK3=`df -h | grep 'dev/root' | awk '{print $4}'`    # Frei

# Arbeitsspeicher
RAM1=`free -h | grep 'Mem' | awk '{print $2}'`    # Total
RAM2=`free -h | grep 'Mem' | awk '{print $3}'`    # Used
RAM3=`free -h | grep 'Mem' | awk '{print $4}'`    # Free
RAM4=`free -h | grep 'Swap' | awk '{print $3}'`    # Swap used

# IP-Adressen ermitteln
#if ( ifconfig | grep -q "eth0" ) ; then IP_LAN=`ifconfig eth0 | grep "inet" | cut -d " " -f 2 | cut -d " "
-f 1` ; else IP_LAN="---" ; fi ;
if ( ifconfig | grep -q "eth0" ) ; then IP_LAN=`ip -4 addr show eth0 | grep -oP '(?<=inet\s)\d+(\.\d+)
{3}'` ; else IP_LAN="---" ; fi ;
if ( ifconfig | grep -q "wlan0" ) ; then IP_WLAN=`ip -4 addr show wlan0 | grep -oP '(?<=inet\s)\d+
(\.\d+){3}'` ; else IP_WLAN="---" ; fi ;

echo "\033[1;36m$DATUM
OS.............: $OS1 $KERNEL
Hostname......: \033[1;33m$HOSTNAME\033[1;36m
Letzter Login.: $LAST1, $LAST2 $LAST3 $LAST4 von $LAST5
Uptime........: $UP1 Tage $UP2 Stunden $UP3 Minuten
Ø Auslastung..: $LOAD1 (1 Min.) | $LOAD2 (5 Min.) | $LOAD3 (15 Min.)
CPU Temperatur: $TEMP
Spannung......: $VOLTAGE
Speicher......: Gesamt: $DISK1 | Belegt: $DISK2 | Frei: $DISK3
RAM (MB)......: Gesamt: $RAM1 | Belegt: $RAM2 | Frei: $RAM3 | Swap: $RAM4
IP-Adressen...: LAN: $IP_LAN | WiFi: $IP_WLAN
\033[m"
```

26. sudo ln -s /etc/update-motd.d/30-sysinfo /pistatus.sh
27. sudo chmod +x /etc/update-motd.d/30-sysinfo
chmod +x /pistatus.sh

28. W-LAN ggf. einrichten

```
sudo nano /etc/network/interfaces.d/wlan0
>
#wlan0
auto wlan0
allow-hotplug wlan0
iface wlan0 inet dhcp
  pre-up wpa_supplicant -B -D wext -i wlan0 -c /etc/wpa_supplicant/wpa_supplicant.conf
  post-down killall -q wpa_supplicant
```

29. `sudo nano /etc/dhcpcd.conf`
Fallback IP am Netzwerkanschluss ggf. einrichten und Wifi Schnittstelle konfigurieren

>
It is possible to fall back to a static IP if DHCP fails:
define static profile
profile static_eth0
static ip_address=192.168.1.23/24
static routers=192.168.1.1
static domain_name_servers=192.168.1.1

fallback to static profile on eth0
interface eth0
fallback static_eth0

enable WiFi
interface wlan0
env ifwireless = 1
env wpa_supplicant_driver = wext , nl80211

30. `sudo nano /etc/wpa_supplicant/wpa_supplicant.conf`
Anpassen an jeweilige W-LAN Verbindung, hier Bsp. Für WPA2-Enterprise
>
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
country=DE
network={
 ssid="meinwifi"
 priority=1
 scan_ssid=1
 key_mgmt=WPA-EAP
 anonymous_identity="appletv_$(uname -n)"
 identity="appletv-01234"
 password=hash:e6eaaaaaaaaaaaaa
 eap=PEAP
 phase1="peaplabel=0"
 phase2="auth=MSCHAPV2"
 ca_cert="/usr/local/share/ca-certificates/extra/rootca.crt"
 subject_match="srv.example.com"
}

#Test: wpa_supplicant -i wlan0 -c /etc/wpa_supplicant/wpa_supplicant.conf

Der NTPasswort Hash (NTLM password hash) für die Wifi Verbindung kann wie folgt selbst generiert werden:
`echo -n plaintext_password_here | iconv -t utf16le | openssl md4`
Prefix it with "hash:" in the wpa_supplicant.conf file, i.e.
password=hash:66aaaaaaaaaaaaaaaaaaaaaa

31. `sudo nano /etc/systemd/system/rpiplay.service`
Service für RPiPlay erstellen mit Hostname des Geräts als AirPlay Gerätename
>
[Unit]

```
Description=RPiPlay AppleTV Service
Documentation=https://github.com/FD-/RPiPlay
DefaultDependencies=yes
After=network.target
StartLimitIntervalSec=30

[Service]
Type=simple
ExecStart=/usr/local/bin/rpiplay -n %H -b auto -vr rpi -ar rpi -a hdmi -l
StandardOutput=inherit
StandardError=journal
LogLevelMax=3
Restart=always
RestartSec=10

[Install]
WantedBy=multi-user.target
```

32. `sudo systemctl enable rpiplay`

33. `sudo nano /etc/rc.local`

IP Adresse und Gerätename in der Konsole anzeigen und FBI nach Booten des Geräts starten und vor der Konsole anzeigen. Es ist dann das angezeigte Bild, wenn keine Daten via AirPlay empfangen werden.

```
clear

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

# Print AppleTV Hostname
_H=$(hostname -s) || true
if [ "$_H" ]; then
  printf "Connect to AppleTV MGM: \033[1;33m%s\033[1;36m\n" "$_H"
fi

/usr/bin/fbi -T 1 -device /dev/fb0 -noverbose -autozoom /opt/ipad-rpiplay-appletv-screen.png &

exit 0
```

34. `sudo passwd root`

root Passwort und Passwort des Standardbenutzers pi ändern

35. `passwd`

36. Standardbenutzer pi ggf. in airplay umbenennen

```
su -
sed -i s/pi/airplay/g /etc/passwd
```

```
sed -i s/pi/airplay/g /etc/shadow
sed -i s/pi/airplay/g /etc/group
sed -i s/pi/airplay/g /etc/sudoers
sed -i s/pi/airplay/g /etc/gshadow
mv /home/pi /home/airplay
```

37. `sudo raspi-config` ggf. Read-Only und Overlay Filesystem enablen um die Speicherkarte zu schonen oder Systemänderungen zu verhindern. Auf Pi Zero wegen kleinem Arbeitsspeicher ist dies jedoch nicht zu empfehlen.
38. `rm /etc/ssh/ssh_host_*`
`dpkg-reconfigure openssh-server`
ggf. SSH Schlüssel neu erzeugen

Images

Ein erstelltes Image kann leicht auf andere Speicherkarten geklont werden. Es ist lediglich der Hostname auf dem geklonten Rechner anzupassen. Mit pishrink.sh aus dem Projekt https://github.com/Drewsif/PiShrink kann das Image auf ein Minimum verkleinert und so ggf. auch auf kleineren Speicherkarten aufgespielt werden.
pishrink.sh -v -s *.img
Sollte das Dateisystem beim ersten Boot nicht auf die maximale Größe expandiert werden kann dies mit folgenden Befehlen manuell geschehen:

```
sudo raspi-config --expand-rootfs
```