⇑ Steuer-Computer ⇑	NAV		
⇓ Raspberry PI	⇓ Installation	✘ Optionen	⇓ Backup

Raspberry PI: Optionen

Strom sparen

Funktion

Strom sparen im Dauerbetrieb durch Abschaltung von Funktionen, welche nicht für die Bake verwendet werden. Dies betrifft folgende Themen:

- Bluetooth
- HDMI-Ausgabe

OS-Konfiguration

- Per SSH anmelden
- `sudo nano /boot/config.txt`
- Am Ende der Datei folgende Zeilen hinzufügen:

```
dtoverlay=disable-bt
hdmi_blanking=1
```

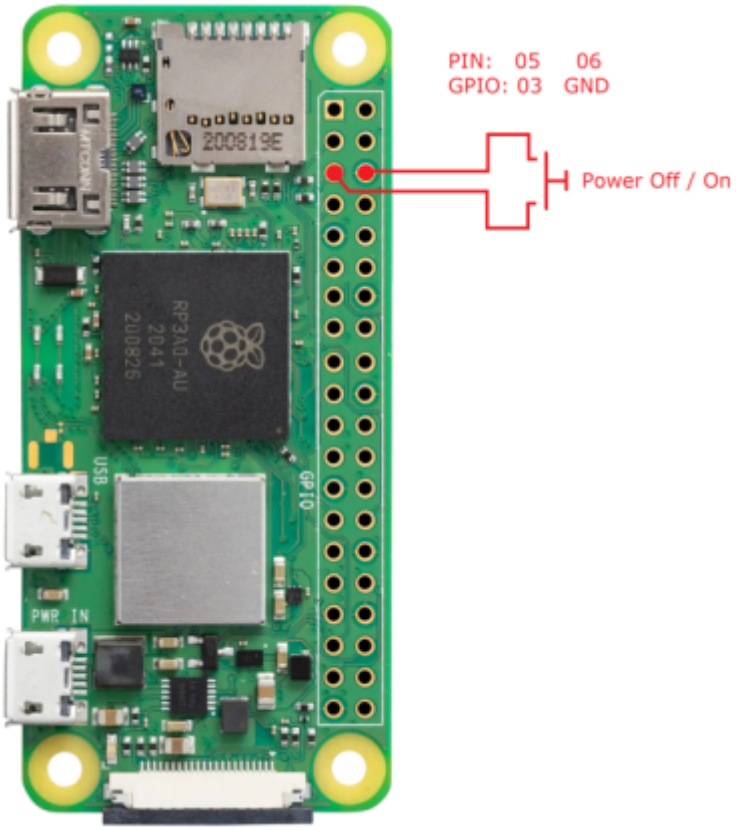
Test

Kann nur mittels eines USB-Strommessgerätes gemessen werden.

Start-/Stop-Taster

Leider besitzt keiner der Raspberry PI ab Werk einen „Ein-/Aus-Schalter“. Das Gerät schaltet sich ein, sobald es mit Strom versorgt wird. Dies hat Vor- und Nachteile. Da man aber speziell ein Linux-Computer-System nicht einfach so durch „Stromverlust“ ausschalten sollte, mag es sinnvoll erscheinen, einen Solchen „Start-Knopf“ einzubauen.

Hardware



An den Pins 05 (GPIO03) und 06 (GND) wird ein Taster angeschlossen. Die Beleung ist bei ALLEN PI-Modellen gleich. Lediglich auf der Seite des Betriebssystemns wird der Taster unterschiedlich behandelt.

Funktion

- Wird der PI mit der Stromversorgung verbunden, fährt er hoch.
- Wird der angeschlossene Taster kurz gedrückt, führt der PI einen Neustart durch (reboot).
- Wird der angeschlossene Taster lange gedrückt (> 5 Sekunden), fährt der PI herunter (shutdown).
- Solange die Stromversorgung NICHT vom PI getrennt wird, kann dieser mit einem kurzen Tastendruck wieder gestartet werden.

Voraussetzungen

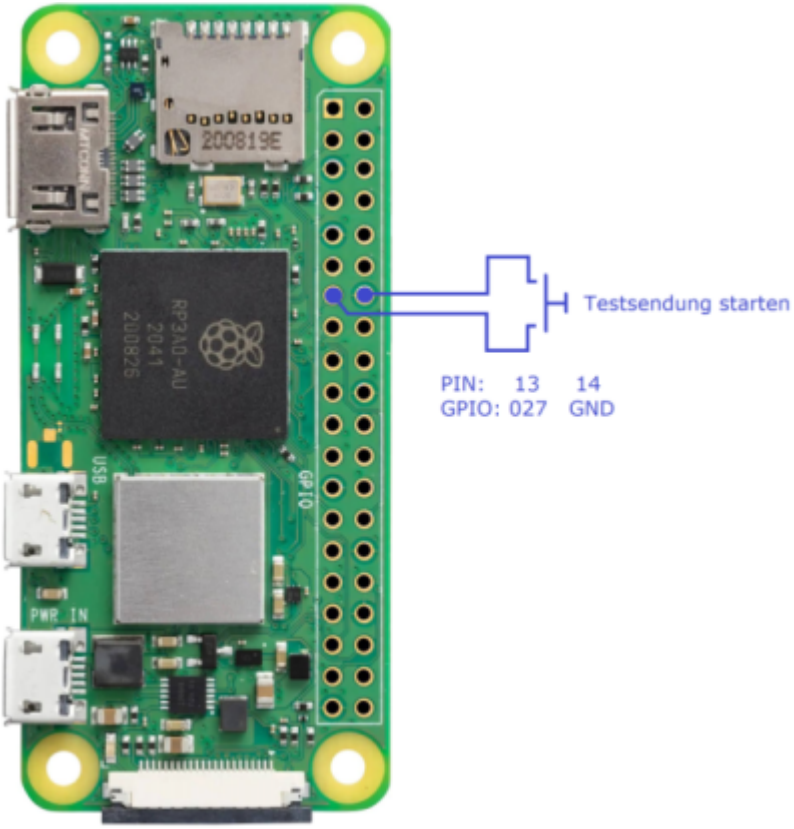
Die Installation-Prozedur für Variante 4, Variante 5 oder Variante 6 wurde erfolgreich durchgeführt und getestet.

OS-Konfiguration

- Per SSH anmelden (pi)
- `sudo wget -O pishutdown.zip wget https:`*`deutschland-funkt.de/bake/lib/exe/fetch.php?media=grundlagen:computer:rapsberrypi:pishutdown.zip * sudo unzip pishutdown.zip * sudo cp`*

*pishutdown/pishutdown.py /usr/local/bin * sudo cp pishutdown/pishutdown.* /etc/systemd/system * sudo systemctl enable pishutdown && sudo systemctl start pishutdown === Test === Kurz drücken (< 3 Sekunden): PI startet neu*
Lang drücken (> 3 Sekunden): Pi fährt herunter
*Erneut drücken: PI fährt wieder hoch —- ===== Start-Meldung per Funk abspielen ===== ==== Funktion ==== * Wird der PI gestartet, wird per Funk eine Startmeldung ausgegeben. ==== Voraussetzungen ==== Die Installation-Prozedur für Variante 4, Variante 5 oder Variante 6 wurde erfolgreich durchgeführt und getestet. ==== Software ==== * Per SSH anmelden (pi) * nano /home/pi/batch/startmeldung * Inhalt der Datei: #!/bin/bash*
batch=„/home/pi/batch“
audio=„/home/pi/audio“
$batch/ptt.py a ein
mplayer -ao alsa $audio/start.mp3 $audio/CALL-Notfallbake.mp3 $audio/CALL-Betriebsbereit.mp3 $audio/CALL-Ende.mp3 $audio/stop.mp3
*$batch/ptt.py a aus * Editor beenden und Datei speichern * crontab -e * Folgende Zeile VOR dem Dateiende einfügen: @reboot sleep 15; /home/pi/batch/startmeldung; * Editor beenden und Datei speichern ==== Test ==== * PI herunterfahren * PI neu starten * Sobald die Meldung per Funk ausgestrahlt wird, kann man sich per SSH anmelden. —- ===== STOP-Meldung per Funk abspielen ===== ==== Funktion ==== Wird der PI herunter gefahren, wird per Funk eine Meldung zum Ende des Betriebes ausgegeben. ==== Voraussetzungen ==== Die Installation-Prozedur für Variante 4, Variante 5 oder Variante 6 wurde erfolgreich durchgeführt und getestet. ==== Software ==== * Per SSH anmelden (pi) * nano /home/pi/batch/call-shutdown * Der Inhalt der Datei: #!/bin/bash*
batch=„/home/pi/batch“
audio=„/home/pi/audio“
$batch/ptt.py a ein
mplayer -ao alsa $audio/start.mp3 $audio/call-betriebsende.mp3 $audio/stop.mp3
*$batch/ptt.py a aus * Editor beenden und Datei speichern * sudo nano /etc/systemd/system/custom-shutdown.service * Die Datei hat folgenden Inhalt: [Unit]*
Description=Schluss-Sound spielen
Before=umount.target
[Service]
Type=oneshot
User=root
WorkingDirectory=/home/pi/batch
ExecStart=/home/pi/batch/call-shutdown
StandardOutput=inherit
StandardError=inherit
[Install]
*WantedBy=reboot.target halt.target poweroff.target * Editor beenden und Datei speichern * sudo systemctl enable —now custom-shutdown.service * sudo systemctl status custom-shutdown.service * Die Ausgabe sollte so aussehen: custom-shutdown.service - Schluss-Sound spielen*
Loaded: loaded (/etc/systemd/system/custom-shutdown.service; enabled;

vendor
*Active: inactive (dead) ==== Test ==== * PI herunterfahren * Die SSH-Verbindung (wenn eine bestand) geht verloren. * Die Meldung zum Betriebsende wird ausgestrahlt. Danach wird der PI abgeschaltet. * PI neu starten * Sobald die Meldung per Funk ausgestrahlt wird, kann man sich per SSH anmelden. —- ===== TEST-Sendung per Funk abspielen ===== ==== Funktion ==== Im Baken-Betrieb kann auf Knopfdruck jederzeit eine Testsendung erfolgen! Bitte dabei auf Kollisionen mit dem Sendeplan achten. ==== Voraussetzungen ==== Die Installation-Prozedur für Variante 4, Variante 5 oder Variante 6 wurde erfolgreich durchgeführt und getestet. ==== Hardware ====*



*Am Pin 13 (GPIO027) und 14 (GND) wird ein Taster angeschlossen. ==== Software ==== * Per SSH anmelden (pi) * crontab -e * In der Crontab folgende Zeile hinzufügen: @reboot /home/pi/batch/call-testsendung.py; * Editor beenden und Datei speichern * nano /home/pi/batch/testsendung.py * Der Inhalt der Datei sieht so aus ([TAB] = Einmal Tabulator-Taste):*

```
#!/usr/bin/env python3
from gpiozero import Button
from signal import pause
import subprocess
def play_music():
[TAB]subprocess.Popen([„/home/pi/batch/call-testmeldung“])
def main():
[TAB]try:
[TAB][TAB]button = Button(27)
[TAB][TAB]button.when_pressed = play_music
[TAB][TAB]pause()
[TAB]except KeyboardInterrupt:
```

[TAB][TAB]pass
if name == „main“:
*[TAB][TAB]main() * Editor beenden und Datei speichern * nano /home/pi/batch/call-testmeldung * Der Inhalt der Datei sieht so aus:*
#!/bin/bash
batch=„/home/pi/batch“
audio=„/home/pi/audio“
$batch/ptt.py a ein
mplayer -ao alsa $audio/start.mp3 $audio/call-notfallbake.mp3 $audio/call-testaussendung.mp3 $audio/call-ende.mp3 $audio/stop.mp3
*$batch/ptt.py a aus'' * Editor beenden und speichern ==== Test ====*
Test-Knopf drücken. Die Testmeldung sollte per Funk ausgegeben werden.
|⇑ Steuer-Computer ⇑|NAV||| |⇓ Raspberry PI|⇓ Installation|□ Optionen|⇓ Backup|